폭주회피를 위한 개선된 큐 관리 기법에 관한 연구

A Study on the Modified Queue Management Scheme for Congestion Avoidance

양 진 영* 이 팔 진** 김 종 화***
Jin-Young Yang Pal-Jin Lee Jong-Wha Kim

요 약

본 논문에서는 IP 망에서의 폭주회피를 위한 수정된 RED 알고리즘을 제안한다. RED는 평균 큐 크기를 계산하여 초기 폭주를 탐지하지만, 이를 무작위로 선택된 소스에 통보하여 패킷을 드롭시키기 때문에 글로벌 동기화 또는 공정성 문제를 유발한다. 또한 최적의 평균 큐 크기를 얻기 위해서는 네트워크 트래픽 특성이 파악되어야 한다. 수정된 RED 알고리즘은 평균 큐 크기가 최소 임계치를 초과하면 연결의 상태를 이용하여 각 소스의 패킷을 드롭시킨다. 이는 큐에서 버스트를 허용하면서 평균 큐 크기를 낮게 유지할 수 있기 때문에 망의 성능을 향상시킬 수 있다.

제안된 알고리즘은 goodput, 네트워크 이용률, 공정성의 평가인자를 이용하여 기존의 기법인 DT, RED와 비교 분석을 수행하였으며, 제안된 기법의 타당성을 보인다.

Abstract

In this paper, a Modified RED algorithm for congestion avoidance in IP networks is presented. The RED detects incipient congestion by computing the average queue size. By notifying only a randomly selected fraction of connection, it causes to the global synchronization or fairness problem. And also, the network characteristics need to be known in order to find th optimum average queue length. When the average queue size exceeds a minimum threshold, a modified RED algorithm drops packets based on the state of each connection. Performance is improved because of keeping the average queue size low while allowing occasional bursts of packets in the queue.

we compare performance of modified RED with RED and Drop Tail in terms of goodput, network utilization and fairness.

1. 서 론

인터넷 상에서 사용 중인 대부분 전송 프로토콜은 종단 간 폭주제어를 제공하는 TCP(Transmission Control Protocol)이다. TCP는 적응적인 윈도우 기반의 흐름제어 기법을 제공하며, TCP에 의해 개발된 폭주회피와 제어 알고리즘은 이용 가능한 자원을 최대로 사용하는데 그 목적이 있다[9].

TCP의 처리율(throughput)과 공정성(fairness)을 향상시키기 위해 Floyd와 Jacobson에 의해 제안된 RED(Random Early Detection) 알고리즘이 있다[1][4]. 이 밖에도 처리율과 공정성을 향상시키기 위한 다양한 방법이 연구되어 오고 있다. RED는 TCP와 같은 네트워크 전송이 널리 사용될 때 종종 발생되는 버스트를 허용하면서 게이트웨이에서의 큐를 제어하여 패킷 전송률을 일시적으로 감소시킴으로써 평균 큐 크기를 가능한 낮게 유지하고, 큐의 오버플로우 상태에 따라 목적지에서 폐기될 불완전한 패킷으로 인한 congestion collapse를 피할 수 있다. 그러나 이 방법은 패킷을 무작위로 드롭시키기 때문에 글로벌 동기화 문제를 가져오며, 패킷을 비례적으로 드롭시키기 때문에 공정성에 부정적인 효과가 있다[13]. RED에서 최적의 큐 크기를 결정하기 위해서는 네트워크 특성을 파악하여야 하며, 최대 임계치인 $\max_{th}$의 최적값은 라우터에서 허용되는 최대 평균 지연에 묶

* 정회원 : 목포대학교 정보공학부 박사
jyyang@chodang.ac.kr
** 정회원 : 초당대학교 컴퓨터공학과 조교수
pjlee@chodang.ac.kr
*** 비회원 : 목포대학교 정보공학부 부교수
kimjh@chungye.mokpo.ac.kr

여 있기 때문에 네트워크 상에서 운영되는 다양한 응용 유형과 연관지어져야 한다[7].

따라서, 본 논문에서는 각 연결에 대한 패킷 드롭핑의 공정성을 유지하고, 글로벌 동기화 문제를 해결하기 위하여 평균 큐 크기가 최소 임계치와 최대 임계치 사이에 존재할 때 들어오는 패킷을 연결(C_i)의 상태에 따라 선택적으로 드롭시킴으로써 RED가 가지고 있는 문제를 해결한다. 그리고 제안된 방법은 goodput, 네트워크 이용률, 그리고 공정성의 평가인자를 이용한 기존의 폭주제어 기법인 Drop Tail, RED와 비교 및 분석을 수행하며, 시뮬레이션을 통해 제안된 방법의 타당성을 검증한다.

본 논문의 구성은 다음과 같다. 2장에서는 폭주회피를 위한 수정된 RED를 제안하며, 3장에서는 시뮬레이터를 이용하여 제안된 기법의 시뮬레이션을 수행하며, 얻어진 결과와 기존의 기법과의 비교 분석한다. 마지막으로 4장에서는 결론을 다루며, 향후 수행되어야 할 방향을 제시한다.

2. 개선된 큐 관리 알고리즘

RED에서 가지고 있는 불공정성 효과를 줄이기 위해 패킷을 비례적으로 드롭시킴으로써 선택된 연결(C_i)에 무작위로 폭주를 지시하는 대신 큐에 입력된 가장 많은 패킷을 갖는 연결들에 대해 선택적으로 드롭시키는 것이다[13]. 또한 Floyd는 서로 다른 네트워크와 트래픽 조건에 따라 드롭 수준과 드롭 확률을 동적으로 조정하여 최적의 평균 큐 크기를 결정하는 것을 강조하고 있다[4].

게이트웨이가 폭주를 소스에 통보하는 방법은 패킷 드롭에 의해 이루어지며, 이것은 버퍼가 풀(full) 일 때 자동적으로 이루어진다. RED(Random Early Detection) 알고리즘은 큐 크기가 일시적으로 변함과 동시에 평균 큐잉 지연을 제어하는 것으로, 평균 큐의 크기가 임계치보다 높으면 평균 큐 크기의 선형적인 함수로 통계적 확률을 이용하여 패킷을 무작위로 드롭시킨다. RED 게이트웨이에 대한 일반적인 알고리즘은 그림 1과 같다.

```
for each packet arrival
    calculate the average queue size avg
        if min_th ≤ avg ≤ max_th
            calculate probability p_a
            with probability p_a :
                mark the arriving packet
    else if max_th ≤ avg
        mark the arriving packet
```

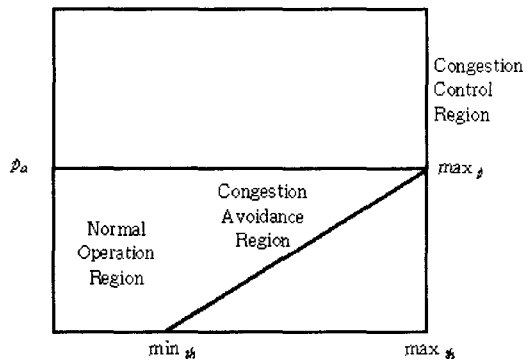
(그림 1) RED 알고리즘

이 알고리즘은 2단계로 운영되는데, 첫 번째로 평균 큐 크기를 계산하기 위해 식 1과 같은 EWMA (exponentially weighted moving average)를 이용한다.

$$avg = (1 - w_q)avg + w_q * Queue\_Size \quad (식\ 1)$$

두 번째로 두 개의 파라미터인 최소 임계치와 최대 임계치인 $\min_{th}$와 $\max_{th}$를 이용하여 그림 2에서 보는 바와 같이 0에서 $\max_p$까지 변하는 확률 p_a를 결정한다.

패킷을 드롭시키는 확률 p_a는 average queue length가 최소 임계치 $\min_{th}$에서 최대 임계 $\max_{th}$에 이르기까지 변하는 것처럼 0에서 $\max_p$까지 선형적으로 변하며, 마지막에 드롭된 패킷, 카운트 이후 패킷수가 증가함에 따라 증가한다.



(그림 2) 임계치 변화에 따른 p_a 값

$$p_a = p_b/(1 - count * p_b) \quad \text{(식 2)}$$

$$p_b = \max_p(avg - \min_{th})/(\max_{th} - \min_{th}) \quad \text{(식 3)}$$

그러나 이 알고리즘은 식 3의 p_b로 인하여 폭주가 증가함에 따라 많은 패킷이 드롭된다. 특히 자원을 적게 사용하는 플로우보다는 더 많이 사용하는 플로우로부터 드롭되는 패킷 수가 훨씬 많다.

평균 큐 크기가 최소 임계치와 최대 임계치 사이에 존재할 때 들어오는 패킷을 연결(Connection)의 상태에 따라 패킷을 선택적으로 드롭시키기 위한 확률 p_a는 다음과 같이 구한다. 연결(C_i)에 대한 Load Ratio(L)는 식 4와 같다.

$$L = (\#\ \text{of packet from}\ C_i)\ /\ (\text{Fair Allocation}) \quad \text{(식 4)}$$

여기서 Fair Allocation(A_{fair})은 식 5와 같이 구할 수 있다.

$$A_{fair} = queue\_size\ /\ \#\ of\ active\ C_i \quad \text{(식 5)}$$

N_a를 연결(C_i) 개수이며, Y_i를 각 C_i에서 발생된 패킷 수라 할 때, C_i의 L은 식 6과 같이 구할 수 있다.

$$L = Y_i \cdot N_a\ /\ queue\_size \quad \text{(식 6)}$$

따라서 P_b는 식 7과 같이 표현할 수 있다.

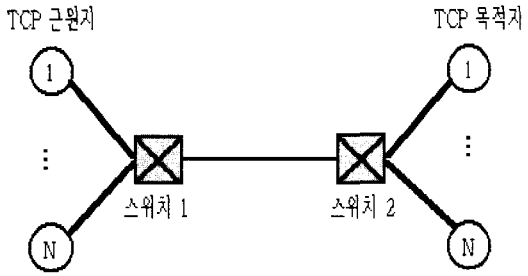
$$P_b = P_b \cdot Y_i \cdot N_a\ /\ queue\_size \quad \text{(식 7)}$$

최종 드롭핑 확률 p_a는 평균 큐 크기와 카운터가 증가함에 따라 증가하며, RED에서 패킷을 무작위로 드롭시켜 발생되는 글로벌 동기화 및 공정성의 문제는 패킷을 선택적으로 드롭시킴으로써 해결할 수 있다.

3. 시뮬레이션

3.1 시뮬레이션 환경

제안된 기법의 시뮬레이션을 수행하기 위한 시뮬레이션 네트워크는 그림 3과 같으며, UNIX 환경을 기반으로 ns[5]와 Tcl and Tk[15]를 이용하여 수행하였다. 각 TCP 연결에는 서로 다른 전파 지연을 허용하며, 링크 대역은 TCP 근원지[목적지]에서 스위치까지는 155.52Mbit/s, 그리고 각 스위치 사이는 45Mbit/s로 설정하였다. 또한 트래픽은 단방향(unidirectional)으로 설정하고, 모든 TCP 근원지는 무한대의 파일 전송을 수행한다.



(그림 3) 시뮬레이션 네트워크

5개의 TCP 근원지와 스위치1 사이에 주어진 전파 지연은 각각 0.25ms, 0.5ms, 1ms, 1.5ms, 2ms이며, 스위치1과 스위치2 사이는 1ms, 목적지와 스위치2 사이는 null로 설정 하였다.

RED 알고리즘의 시뮬레이션 수행을 위해 사용된 파라미터는 표 1과 같다[4]. 버퍼 크기는 오버플로우로 인한 패킷이 결코 드롭되지 않도록 하

(표 1) 시뮬레이션 파라미터

Parameter	Value
W_q	0.002
$\min_{th}$	5 packets
$\max_{th}$	15 packets
$\max_p$	1/50

기 위해 충분히 크게 유지하였으며, 33~112 packets으로 설정하였다.

사용된 시뮬레이터는 4.3-Tahoe BSD Release를 이용하여 수행하였는데, TCP의 윈도우 조정 알고리즘은 두 단계로 구성된다. 초기 임계치는 수신자가 광고한 윈도우 크기의 절반으로 설정된다. 연결(connection)은 slow-start 단계에서 시작하며, 윈도우가 임계치에 도달할 때까지 현재의 윈도우는 매 RTT(Round Trip Time) 마다 2배로 증가한다. 다음에 폭주회피 단계에 접어들면 현재의 윈도우는 매 RTT 마다 하나의 패킷 단위로 증가하며, 수신자가 광고한 윈도우 이상으로 증가될 수는 없다.

4.3-Tahoe BSD TCP에서 패킷 손실은 congestion experienced 신호로 취급된다. 근원지에서의 손실된 패킷의 탐지는 fast retransmit 절차를 이용하며, 똑같은 패킷을 확인하는 4개의 패킷이 수신되면 패킷은 드롭된 것으로 판단한다. 패킷이 드롭되면 근원지는 임계치를 현재의 윈도우의 1/2로 설정하고, 현재의 윈도우를 1로 설정하여 Slow-Start 단계로 진입한다. 근원지는 또한 잃어버린 패킷을 탐지하기 위해 재전송 타이머를 사용한다.

3.2 시뮬레이션 결과

4개의 연결 각각에 대한 전송시간은 노드 1에 대해서는 time 0에서 패킷 전송을 시작하고, 노드 2에 대해서는 time 0.2초 후에, 노드 3에 대해서는 time 0.4초 후에, 노드 4에 대해서는 time 0.6초 후에 패킷 전송을 시작하였다.

비교인자는 비교인자 goodput, 링크 이용률, 공정성(Fairness)을 이용하였다. goodput은 주어진 시뮬레이션 시간 동안에 성공적으로 확인 신호를 받은 패킷들의 수를 이용하여 계산되며, 공정성은 식 9를 이용하여 구할 수 있다[7].

$$Fairness\ Index = 1 - \frac{1}{N}\sum_{i=1}^{N}\left|\frac{t_i - \bar{t}}{\bar{t}}\right| \quad (식\ 9)$$

(표 2) 폭주 회피 알고리즘의 비교

비교인자 \ 알고리즘	DT	RED	개선된 RED
Goodput	0.973	0.980	0.983
Fairness	0.895	0.950	0.975
Utilization	0.952	0.962	0.965

여기서 N은 소스의 수를 나타내며, t_i은 connection i의 처리율, $\bar{t}$는 평균 처리율을 나타낸다. Throughput은 목적지에 전송된 패킷의 전체 수를 시뮬레이션 시간으로 나눈 것으로 정의할 수 있다.

또한 본 연구에서 제안된 기법의 우수함을 보이기 위해 벤치마크로서 DT(Drop Tail)와 RED (Random Early Drop)를 이용하였다. Drop Tail에 대해서 버퍼 크기를 15~140 packet으로 설정하고 phase effects를 피하기 위해 전송을 위한 FTP 패킷을 준비하는데, [o, t]초 사이에서 균일 분포로 무작위 시간을 취했다.

표 2는 버퍼 크기가 20 패킷으로 되어있을 때의 기존의 기법인 Drop Tail과 RED, 제안된 방법과의 시뮬레이션 결과를 정리하였다.

버퍼 크기가 적으면 평균 큐 길이는 불안정하여 평가인자들의 결과치는 전반적으로 적게 나타나지만 버퍼 크기가 증가함에 따라 안정된 결과를 가져왔다. 비록 제안된 기법이 기존의 RED에 비해 약간 향상된 결과치를 가져왔지만, 대부분 게이트웨이에서 사용 중인 RED와 비교한다면 다소 뜻밖의 결과라 할 수 있다. 그러나 네트워크 구조 및 연결 수를 증가시켜 다양한 시뮬레이션을 수행한다면 훨씬 개선된 결과를 기대할 수 있다.

5. 결 론

본 논문에서는 IP 네트워크에서 데이터 트래픽에 대하여 폭주 회피를 효율적으로 수행하기 위해 연결의 상태에 따른 수정된 RED를 제안하였다. 제안된 방법은 각 연결에 대한 패킷 드롭핑의

공정성을 유지하고, 글로벌 동기화 문제를 해결하기 위하여 평균 큐 크기가 최소 임계치와 최대 임계치 사이에 존재할 때 들어오는 패킷은 연결(C_i) 수, 연결에 따른 패킷 수, 큐 상태를 이용한 부하를 계산하고, 이를 기반으로 들어오는 패킷의 드롭여부를 결정한다. 또한 패킷이 드롭될 연결의 결정은 RED에서처럼 무작위로 이루어지는 것이 아니라 선택적으로 이루어지기 때문에 동기화와 공정성을 더욱 더 향상시킬 수 있다.

향후 본 연구는 다양한 네트워크 트래픽 특성을 고려한 시뮬레이션을 지속적으로 수행하여 RED가 가지고 있는 다양한 문제에 접근하고, 또한 이에 대한 해결방안을 연구할 것이다.

참 고 문 헌

[1] S. Floyd, et el. "Recommendations on Queue Management and Congestion Avoidance in the Internet," RFC2309, April. 1998.

[2] Darius Buntinas. "Congestion Control Schemes for TCP/IP Networks," http://www.cis.ohio-state.edu/~jain/cis788-95/tc pip_cong.

[3] Mark Gaynor. "Proactive Packet Dropping Methods for TCP Gateways," http://www.eecs.harvard.edu/~gaynor/final.ps.

[4] Sally Flyod and Van Jacobson. "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Transactions on Networking, Aug. 1993.

[5] S. McCanne and S. Floyd. ns Network Simulator. http://www.isi.edu/nsnam/ns/.

[6] Kevin Fall. Kannan Varadhan, The VINT project : ns Notes and Documentation, Feb. 25, 2000.

[7] Miguel A.Labrador and Sujata Banerjee. "Packet Dropping Policies for ATM and IP Networks," IEEE Communications Surveys, Vol.2, No.3, 1999.

[8] Vincent Rosolin, Olivier Bonaventure and Guy Leduc. "A RED discard strategy for ATM networks and its performance evaluation with TCP/IP traffic," http://montefiore.ulg.ac.be.

[9] Omar Elloumi, Hossam Afifi, "Improving RED Algorithm Performance in ATM Networks," GLOBECOM'97 IEEE Global Telecommunication Conference, 1997.

[10] Sally Floyd, Kevin Fall. "Promoting the of End-to-End Congestion Control in the Internet" IEEE/ACM Transactions on Networking, Vol.7, No.4, August. 1999.

[11] J.Heinanen, K. Kikki. "A Fair Buffer Allocation Scheme," Unpublished Manuscript.

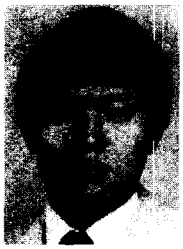
[12] A. Mankin, K. Ramakrishnan. "Gateway Congestion Control Survey," RFC:1254, Aug.1991. http://www-uxsup.csx.cam.ac.uk/netdoc/rfc/rfc1254.txt.

[13] Kevin Fall and Sally Floyd. "Simulation-based Comparison of Tahoe, Reno, and SACK TCP," http://ee.lbl.gov.

[14] M. Allman, V. Paxson, W. Stevens, "TCP Congestion Control,"" Network Working Group RFC: 2581, April. 1999. http://www.isoc.org/inet2000/cdproceedings/rfc/rfc/2581.txt.

[15] Brent B. Welch, Practical Programming in Tcl and Tk, Prentice Hall PTR, 1997.

◐ 저 자 소 개 ◑



양 진 영
1983년 조선대학교 경영학과(경영학사)
1988년 조선대학교 대학원 전자계산학과(공학석사)
2001년 목포대학교 대학원 컴퓨터공학과(박사)
1997~현재 초당대학교 컴퓨터과학과 조교수
관심분야 : TCP/IP, Traffic Control, MMI
E-mail : jyyang@chodang.ac.kr

이 팔 진
1986년 조선대학교 전산기공학과(공학사)
1988년 중앙대학교 대학원 전자계산학과(이학석사)
1995년 전북대학교 대학원 전자계산기공학과(공학박사)
1995~현재 초당대학교 컴퓨터과학과 조교수
관심분야 : TCP/IP, Networking QoS, Wireless Network
E-mail : pjlee@chodang.ac.kr

김 종 화
1983년 조선대학교 전자공학과(공학사)
1985년 조선대학교 대학원 전자공학과(공학석사)
1991년 일본 동북대학 대학원 전자공학과(공학박사)
1991~현재 목포대학교 정보공학부 부교수
관심분야 : Embeded System IBS, Web-based Control, HSI
E-mail : kimjh@chungye.mokpo.ac.kr