IP 네트워크에서 혼잡제어를 위한 새로운 Active RED 알고리즘

구자헌[L], 정광수, 오승준
광운대학교 전자공학부 컴퓨터통신연구실
jhkoo@adams.gwu.ac.kr, kchung@daisy.gwu.ac.kr, sjoh@media.gwu.ac.kr

A New Active RED Algorithm for Congestion Control in IP Networks

Jahon Koo[L], Kwangsue Chung, Seoungjun Oh
School of Electronics Engineering, Kwangwoon Univ.

요 약

기존의 인터넷 라우터는 Drop tail 방식으로 패킷을 관리한다. 따라서 네트워크 트래픽의 지수적인 증가로 인한 혼잡 상황으로 발생하는 패킷 손실을 해결 할 수 없다. 이 문제를 해결하기 위해 IETF (Internet Engineering Task Force)에서는 RED(Random Early Detection) 알고리즘과 같은 능동적인 큐 관리 알고리즘을 제시하였다. 하지만 RED 알고리즘은 네트워크 환경에 따른 매개 변수의 설정의 어려움을 가지고 있어 잘못 된 매개변수 설정으로 인하여 네트워크 성능이 저하되는 문제점을 가지고 있다. 본 논문에서는 기존의 RED를 개선한 ARED를 제안했다. ARED는 RED 알고리즘의 문제점을 개선하여 네트워크 특성에 맞추어 동적으로 매개변수를 조절하는 알고리즘이다. ns를 이용한 실험을 통하여 ARED의 성능을 검증하였다.

1. 서론

현재 인터넷의 구조는 Drop tail 방식을 이용하는 라우터로 구성되어져 있으며 IP 프로토콜을 기반으로 하는 최선형 서비스를 하고 있다. 이러한 구조는 지수적으로 증가하는 인터넷 트래픽으로 인하여 발생하는 혼잡상황을 제어 할 수 없다. 그리고 혼잡상황으로 발생하는 패킷 손실률의 증가는 네트워크의 성능을 저하시키며 인터넷 서비스의 열화를 초래한다. 이런 문제를 해결하기 위해서는 네트워크의 트래픽이 집중되는 라우터나 게이트웨이에 향상된 큐 관리 알고리즘(queue management algorithms)이 필요하다. 이러한 문제를 해결하기 위한 큐 관리 알고리즘으로 IETF에서는 RED(Random Early Detection) 알고리즘을 권고하고 있다[1,2,3]. RED 알고리즘은 기존의 다른 큐 관리 알고리즘이 가지고 있던 여러 문제를 완화시키며 전역동기화 및 패킷 손실 문제를 해결하였다[4].

그러나 RED 알고리즘의 경우 네트워크 부하에 맞추어 매개변수(parameter)를 설정하지 않을 경우 기존의 Drop tail 방식 보다 효율이 떨어지며 다양한 네트워크 환경에 따른 매개변수 설정의 어려움을 가지고 있다.

본 논문에서는 RED 알고리즘의 문제점을 개선하여 네트워크 특성에 맞추어 매개 변수를 동적으로 설정하는 새로운 큐 관리 알고리즘인 ARED(Active RED)를 제안하였다. 제안한 알고리즘은 기존의 큐 관리 알고리즘 보다 효율적으로 큐를 관리하며 좋은 성능을 제공한다.

2장에서는 큐 관리 알고리즘과 혼잡상황 제어 동작에 대해 기술하였고 3장에서는 RED 알고리즘의 동작과 문제점에 대하여 기술하였다. 4장에서는 새로 제안한 ARED 알고리즘에 대하여 기술하였다. 5장에서는 시뮬레이터를 이용하여 제안한 알고리즘을 검증하였다. 마지막으로 6장에는 결론을 맺었다.

2. 관련 연구

2.1 큐 관리 알고리즘

현재 인터넷의 라우터나 게이트웨이에서 발생 할 수 있는 혼잡상황을 해결하기 위해서 여러 가지 혼잡제어 방법이 논의되고 있다. 그 중에서도 프로토콜 레벨의 혼잡 제어 방법으로 TCP와 같은 전송 프로토콜을 기반으로 한 혼잡 제어 방법이 많이 제시가 되었다. 또 다른 방법으로 네트워크 레벨의 혼잡 제어 방법으로 트래픽이 집중되는 라우터나 게이트웨이에서 향상된 큐 관리 알고리즘을 지원하여 혼잡 상황을 해결하는 방법이 제시되었다[1].

라우터나 게이트웨이에서 효과적으로 큐를 관리하는 혼잡제어 알고리즘으로 IETF에서 논의하고 있는 것은 크게 두 가지로 분류 할 수 있다. 첫 번째는 스케줄링 알고리즘으로 대표적인 방법으로는 FQ(Fair Queueing) 알고리즘이 있다. 이 알고리즘은 각 flow에 대하여 개별적인 큐를 분리하여 관리하는 per-flow방식을 사용한다. 그러나 흐름들에 관한 정보를 유지하고 처리하기 위해서 복잡한 알고리즘을 필요로 하기 때문에 고속의 라우터 처리 능력을 요구한다. 많은 flow를 갖는 네트워크 환경에서 널리 사용하기에는 많은 비용이 필요하다.

두 번째로 처음부터 간단히 설계된 큐 관리 알고리즘이 있다. 이 방법은 간단하면서도 어느 정도의 공정성을 유지하는 기능을 제공한다. 대표적인 알고리즘으로는 RED(Random Early Detection)가 있다. 이 방법은 모든 flow가 하나의 FIFO 큐를 통하여 처리가 되며

네트워크 상태에 따라 관리된다. 따라서 적은 비용으로도 네트워크에 발생하는 혼잡 상황 문제를 해결할 수 있다.

2.2 혼잡 제어의 동작

기존의 인터넷에서는 혼잡상황을 해결하기 위해서 종단간 흐름 제어 (End-to-End Flow Control)방법을 이용한다. 이 방법은 TCP와 같은 전송 프로토콜을 이용하여 피드백(feedback) 받은 네트워크 정보를 기반으로 TCP 송신측(TCP Sender)에서 각 연결의 전송률을 윈도우 기반(window-based)으로 제어한다[2].

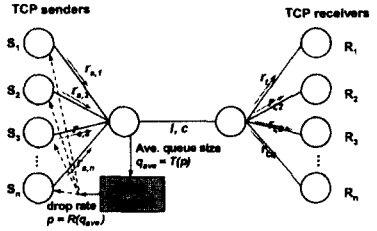

그림 1. n개의 플로우 피드백 시스템

그림 1과 같이 n개의 TCP 플로우에 대하여 혼잡제어 동작을 하는 피드백 시스템을 표시 할 수 있다. TCP의 혼잡제어 알고리즘은 큐의 오버플로우(overflow)가 발생하거나 패킷이 폐기되어 패킷이 손실되면, 피드백 정보를 통하여 혼잡상황을 인지하고 소스의 전송률을 줄여 혼잡상황을 해결하려 한다. 일반적으로 TCP 송신측에서 조절하는 소스의 전송률은 네트워크에서 발생하는 패킷 폐기율(drop rate)에 위해서 결정된다. 또한 RED 알고리즘이나 큐 관리 알고리즘과 같이 라우터에서 네트워크의 평균 큐 크기나 트래픽의 로드에 따라 큐를 관리하는 폐기 모듈(drop module)은 TCP 송신측에서 보내는 전송률에 위해서 패킷 폐기율을 결정한다. 이러한 동작의 상관 관계는 그림 2와 같이 표현 할 수 있다.

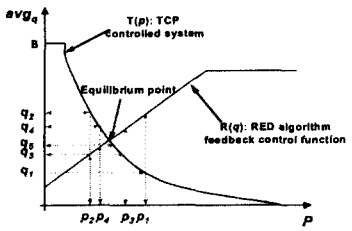

그림 2. RED를 이용하는 피드백 시스템의 평형 상태

그림 2와 같이 전송률과 폐기율은 평균 큐 크기와 패킷 폐기 확률 값으로 표현이 되며 두 상관 관계에 있는 함수($q_{ave} = T(p)$, $p = R(q_{ave})$)는 평형상태(Equilibrium)로 수렴하도록 동작한다. 일반적으로 안정적인 혼잡제어 동작을 할 경우 평형 상태로 수렴한다. 만약 평형 상태로 수렴을 하지 못 할 경우 높은 패킷 폐기율을 가지게 되며 큐의 크기가 심하게 변동한다[5].

3. RED 알고리즘의 결점

3.1 RED 알고리즘의 동작

RED 알고리즘은 혼잡 상황이 발생하기 이전에 평균 큐 크기를 기반으로 그림 3과 같이 혼잡 상태를 판별하여 혼잡 상태에 대한 정보를 패킷의 폐기나 ECN(Explicit Congestion Notification)을 이용한 표시를 이용하여 종단 호스트에게 혼잡정보를 알려주는 방법이다.

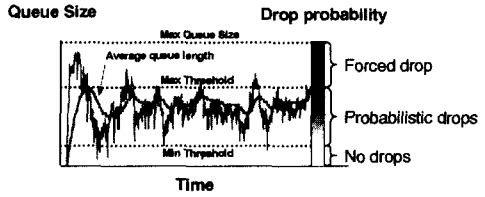

그림 3. RED 알고리즘의 동작

일반적으로 미리 혼잡 상황을 발견하여 혼잡제어를 수행하므로 네트워크 자원의 낭비가 적어 Drop tail 보다는 좋은 성능을 제공한다.

RED 알고리즘은 표 1과 같이 다양한 매개 변수에 의해서 제어되며 그 중에서도 max_p 값에 따라 전체적인 특성이 결정된다.

표 1. RED 제어 매개 변수

qlen	최대로 수용 할 수 있는 큐 공간 크기
min_{th}	폐기 확률을 결정하는 최소 임계 값
max_{th}	폐기 확률을 결정하는 최대 임계 값
w_q	Avg_q 변화의 특성을 결정하는 가중치 값
max_p	폐기 특성을 결정하는 최대 확률 값

그러나 RED 알고리즘의 경우 다양한 매개변수로 설정되어 동작하기 때문에 잘못 된 매개변수를 설정 할 경우 전체적인 성능이 Drop tail 보다 떨어진다[6].

3.2 RED 알고리즘의 문제점

RED 알고리즘의 동작 특성을 자세히 알기 위해서 그림 4와 같은 네트워크 환경에서 ns(network simulation)[7]을 이용하여 최대 확률 값(max_p)의 값을 변경하며 다양한 플로우 수에 따른 패킷 폐기율의 변화에 대하여 실험하였다.

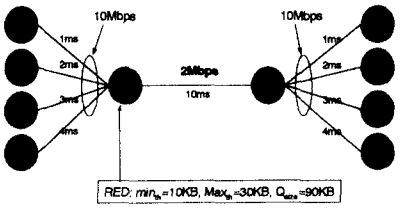

그림 4. 실험 네트워크 설정

그림 5와 같이 max_p 값의 변화에 따라 패킷 폐기율은 변화하였다. 실험 결과에서와 같이 플로우의 수에 따라 최소 패킷 폐기율을 결정하는 max_p 값이 가변적인 것을 알 수 있다. 즉 적절한 max_p 값이 설정되어 있지 않을 경우 그림 2의 평형상태로 수렴을 알 수 없기 때문이다. 따라서 네트워크 특성(플로우의 수)에 따라 능동적으로 동작하는 새로운 큐 관리 알고리즘이 필요하다. 즉 플로우 수에 따라 적절

한 max_p 값을 능동적으로 조절하는 방법이 필요하다[8].

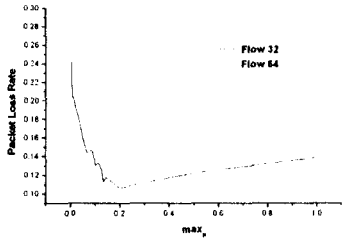

그림 5. 패킷 폐기(손실)률에 대한 실험 결과

본 논문에서는 잘못 설정된 max_p 값으로 발생하는 높은 패킷 폐기율을 해결하기 위해서 능동적으로 매개변수를 설정하는 ARED(Active RED) 알고리즘을 제안한다.

4. Active RED 알고리즘

기존의 RED 알고리즘의 문제점을 해결하기 위해서 제안하는 ARED 알고리즘은 패킷들의 양에 따라서 변화는 큐 크기의 변화율을 줄여 가능하면 평형 상태에 수렴하도록 max_p 값을 조절하는 알고리즘이다. 제안한 ARED 알고리즘은 그림 6과 같다.

```
if (평균 큐의 변화의 정도가 큼
  && (⊿W old_gain < margin1 * ⊿W current_gain))
  max_p 증가

if (⊿T 동안 평균 큐의 크기 감소
  && (평균 큐의 크기 < margin2 * 최대 임계 값))
  maxp 감소

if (max_p 증가)
  max_p = max_p + d1;

else if (max_p 감소)
  max_p = max_p - d2;
```

그림 6. Active RED 알고리즘

본 논문에서 제안하는 ARED 알고리즘의 동작은 임의의 시간 ⊿T 동안의 평균 큐 크기의 변화의 정도를 감시하며 큐 크기의 변화 중 큐의 최대 크기와 최소 크기 사이의 값인 폭(width) ⊿W를 감시하여 변화의 폭을 줄이도록 max_p 값을 조절하는 알고리즘이다. 예를 들면 현재 네트워크 특성에 적합하지 않은 max_p 값이 설정되었을 경우 큐 크기의 변화는 심하게 변동(oscillation)된다. ARED 알고리즘은 이러한 큐의 크기의 변동의 정도를 줄여 전체적인 패킷 폐기(손실)율을 줄여준다.

5. 실험 및 성능 평가

제안한 알고리즘의 성능을 알아보기 위해서 먼저 큐의 크기의 변동이 얼마나 개선되었는지 알아보았다. 그림 4와 같은 네트워크 환경에서 총 16개의 플로우가 전송되는 혼잡링크에 대하여 큐의 변화를 라우터 n5에서 모니터링 하였다. 그림 7과 같이 기존의 RED 방법보다 큐 크기의 변동의 정도를 줄이도록 동작하고 있음을 확인하였다.

플로우의 수의 변화에 따른 패킷 폐기율을 알아보기 위해서 그림 4와 같은 네트워크 환경해서 플로우의 수를 변화 시켰을 때의 패킷 폐기율을 알아보았다. 그림 8과 같이 ARED 알고리즘을 사용 할 경우 기존의 RED 방법보다 낮은 패킷 폐기율을 나타냈다. 따라서 ARED 알고리즘은 특별히 매개 변수를 수동적으로 설정하지 않아도 네트워크 특성에 따라 능동적으로 max_p 값을 조절 및 설정한다.

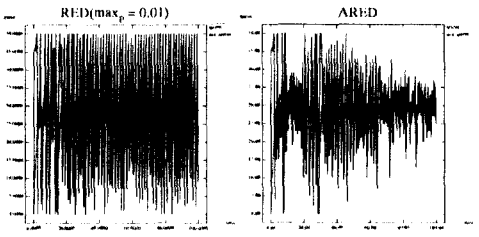

그림 7. RED와 ARED의 큐 크기의 변화

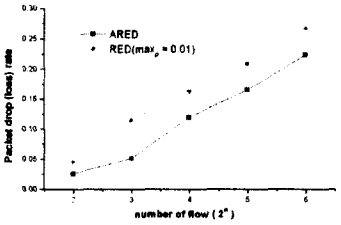

그림 8. RED와 ARED의 패킷 폐기율

6. 결론 및 향후 과제

본 논문에서는 동적으로 변화하는 네트워크 트래픽으로 인하여 발생하는 혼잡상황에 대하여 고정적인 매개 변수 설정으로 동작하는 RED 알고리즘의 문제점을 개선한 새로운 ARED 알고리즘을 제안하였다. ARED는 능동적으로 네트워크 상태에 적합한 매개변수 max_p 값을 설정하여 기존의 RED 알고리즘의 문제점을 개선하였다. 제안한 알고리즘의 성능을 알아보기 위해서 큐 크기의 변화 및 패킷 폐기율에 대하여 실험 및 검증하였다.

향후 연구 과제로는 비반응(unresponsive) flow에 대한 공정성을 개선하는 방법에 대한 연구가 수행되어야 하며, 실제 네트워크 환경에서의 적용에 대한 연구가 수행되어야 할 것이다.

참 고 문 헌

[1] Braden, B., "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, April 1998.

[2] Jacobson, V., "Congestion Avoidance and Control", Proceeding of SIGCOMM88, August 1988.

[3] Floyd, S., and Fall, K., "Router Mechanisms to Support End-to-End Congestion Control", LBL Technical report, February 1997.

[4] Floyd, S., and Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Transaction on Networking, August 1993.

[5] Firoiu, V., and Borden, M., "A Study of Active Queue Management for Congestion Control", Proceedings of INFOCOM 2000, 2000.

[6] Christiansen, M., Jeffay, K., Ott, D., and Smith F.D., "Tuning RED for Web Traffic", IEEE/ACM Transactions, June 2001.

[7] UCB/ LBNL/ VINT, "Network Simulator ns (Version 2)", http://www-mash.cs.berkeley.edu/ns/.

[8] Feng, W., et al "A self-configuring RED gateway", IEEE INFOCOM 99, April 1999.