IP 네트워크에서 혼잡제어를 위한 새로운 Active RED 알고리즘

(A New Active RED Algorithm for Congestion Control in IP Networks)

구 자 헌 † 정 광 수 ††
(Jahon Koo) (Kwangsue Chung)

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 잘못된 매개변수 설정으로 인하여 네트워크 성능이 저하되는 문제점을 가지고 있다. 본 논문에서는 기존의 RED 알고리즘을 분석하여 문제점을 파악하고 이 문제점을 개선한 새로운 ARED (Active Random Early Detection) 알고리즘을 제안했다. ARED 알고리즘은 네트워크 특성에 맞추어 동적으로 매개변수를 조절하므로 서 기존의 RED 알고리즘을 개선한 알고리즘이다. ns(network simulation)를 이용한 실험을 통하여 ARED 알고리즘의 성능을 검증하였다.

키워드 : 혼잡제어, RED, 능동적 큐 관리, 스케줄링 알고리즘, 인터넷

Abstract In order to reduce the increasing packet loss rates caused by an exponential increase in network traffic, the IETF (Internet Engineering Task Force) is considering the deployment of active queue management techniques such as RED (Random Early Detection). While active queue management in routers and gateways can potentially reduce packet loss rates in the Internet, this paper has demonstrated the inherent weakness of current techniques and shows that they are ineffective in preventing high loss rates. The inherent problem with these queue management algorithms is that they all use static parameter setting. So, in case where these parameters do not match the requirement of the network load, the performance of these algorithms can approach that of a traditional Drop-tail.

In this paper, in order to solve this problem, a new active queue management algorithm called ARED (Active RED) is proposed. ARED computes the parameter based on our heuristic method. This algorithm can effectively reduce packet loss while maintaining high link utilizations.

Key words : Congestion Control, RED, Active Queue Management, Internet

1. 서 론

현재 인터넷을 구성하는 네트워크의 구조는 Drop tail 방식을 이용하는 라우터로 구성되어져 있으며 IP 프로토콜을 기반으로 하는 최선형 서비스를 사용하고 있다. 이러한 구조는 지수 적으로 증가하는 인터넷 트래픽으로 인하여 발생하는 혼잡상황(congestion collapse)을 제대로 제어하지 못 한다. 그리고 혼잡상황으로 인하여 발생하는 패킷 손실률의 증가는 네트워크의 성능을 저하시키며 인터넷 서비스의 전체적인 열화를 초래한다. 이러한 문제를 효과적으로 해결하기 위해서는 네트워크의 트래픽이 집중되는 라우터나 게이트웨이에 트래픽을 효율적으로 관리하는 메커니즘인 큐 관리 알고리즘(queue management algorithms)이 필요하다. 이 문제를 해결하기 위한 대표적인 큐 관리 알고리즘으로 IETF(Internet Engi-

† 학생회원 : 광운대학교 전자통신공학과
jhkoo@adams.kwangwoon.ac.kr
†† 종신회원 : 광운대학교 전자통신공학과 교수
kchung@daisy.kwangwoon.ac.kr
논문접수 : 2001년 12월 6일
심사완료 : 2002년 5월 17일

neering Task Force)에서는 RED(Random Early Detection) 알고리즘을 권고하고 있다[1,2,3].

인터넷 서비스의 빠른 확산과 이용자의 급증에 따른 다양한 사용자의 요구에 대한 처리와 사용자의 서비스 질(Quality of Service)의 개선에 대한 연구가 많이 진행 중에 있다. 특히, 최근 인터넷 서비스의 성능을 개선하기 위해서 라우터 알고리즘에 대한 연구가 활발하게 진행되고 있으며 그 중에서도 구현이 용이하며 적은 복잡성(complexity)을 가지는 큐 관리 알고리즘에 대한 연구가 활발히 진행되어 지고 있다. 대표적인 큐 관리 알고리즘인 RED 알고리즘은 기존의 다른 큐 관리 알고리즘이 가지고 있던 여러 문제를 완화시키며 기존의 라우터 메커니즘이 가지고 있던 전역동기화 및 패킷 손실 문제를 해결하였다. 그러나 RED 알고리즘의 경우 네트워크 부하(overload) 특성에 맞추어 알고리즘의 매개변수(parameter)를 적절히 설정하지 않을 경우 기존의 Drop tail 방식 보다 효율이 떨어지는 특성을 가지는 문제점을 가지고 있다. 또한, 다양한 네트워크 환경의 특성에 따른 매개 변수 설정의 어려움을 가지고 있다. 이러한 문제들로 인하여 트래픽이 동적인 실제 네트워크에 적용하기 어려운 단점을 가지고 있다[4,5].

본 논문에서는 기존의 RED 알고리즘의 문제점을 분석하여 네트워크 특성에 맞추어 매개 변수를 동적으로 설정하는 새로운 큐 관리 알고리즘인 ARED(Active RED) 알고리즘을 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 동작을 분석하여 문제점 및 개선 방안에 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

최근 인터넷 트래픽의 지수 적인 증가로 인한 라우터나 게이트웨이에서 발생 할 수 있는 혼잡상황을 해결하기 위해서 여러 가지 혼잡제어 방법이 논의되고 있다. 대표적인 혼잡제어 방법으로는 종단간 흐름 제어(End-to-End Flow Control)방법의 동작 알고리즘을 수정하여 성능을 개선한 방법이 있다. 이 방법은 프로토콜 레벨의 혼잡 제어 방법으로 혼잡 상황 발생 시 TCP와 같은 전송 프로토콜을 기반으로 패킷 전송 윈도우(window) 크기를 조절한다. 최근 기존의 프로토콜 레벨의 혼잡 제어 방법을 개선한 ECN(Explicit Congestion Notification)-TCP, TCP Vegas와 같은 다양한 알고리즘 및 방법이 제시되었다. 또 다른 혼잡 제어 방법으로 트래픽이 집중되는 라우터나 게이트웨이에서 향상된 큐 관리 알고리즘을 이용하여 네트워크 레벨의 혼잡 상황을 개선하는 방법이 제시되었다[1].

프로토콜 레벨의 제어 방법과 네트워크 레벨의 혼잡 제어 방법은 네트워크 상황에 따라서 상호 동작을 해야 한다. 만약 둘 중에 하나의 혼잡제어 방법만으로 혼잡상황을 해결하려고 할 경우 효과적으로 혼잡 상황을 해결할 수 없다. 예를 들어, TCP와 같은 전송 프로토콜을 이용한 혼잡제어 방법의 경우 네트워크 혼잡상황을 해결하기 위해서 윈도우 기반(window-based)의 흐름 제어를 사용을 한다. 하지만, 이 방법으로는 의도적이거나 동적인 네트워크 혼잡 상황을 근본적으로 해결하지는 못하는 한계가 있다. 따라서, 이 문제를 해결하기 위해서는 망에서 추가적으로 상호 연동 할 수 있는 네트워크 레벨의 라우터 메커니즘이 필요하다. 즉, 네트워크 레벨의 라우터 혼잡제어 방법은 인터넷의 근본적인 혼잡상황 문제를 명시적으로 해결 할 수 있게 해준다. 네트워크 레벨의 혼잡 제어 방법은 기존의 Drop tail 방법과 같은 라우터 메커니즘을 개선하여 지수 적인 트래픽 증가 상황에서도 혼잡제어 가능한 망을 설계 할 수 있게 해주며 TCP와 같은 전송 프로토콜 레벨의 혼잡 제어 방법과도 효과적으로 상호 연동한다.

라우터나 게이트웨이에서 효과적으로 큐를 관리하는 네트워크 레벨의 혼잡제어 알고리즘으로 IETF에서 논의하고 있는 라우터 알고리즘은 크게 두 가지로 분류할 수 있다. 첫 번째는 스케줄링 알고리즘으로 대표적인 방법으로는 FQ(Fair Queueing) 알고리즘이 있다[6]. 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. 따라서 많은 flow를 갖는 네트워크 환경에서 널리 사용하기에는 많은 비용이 필요하다.

라우터에서 혼잡상황을 제어하는 두 번째 알고리즘으로는 처음부터 간단히 설계된 큐 관리 알고리즘이 있다. 이 방법은 간단하면서도 어느 정도의 공정성을 유지하는 기능을 제공한다. 대표적인 알고리즘으로는 Floyd가 제안한 RED 알고리즘이 있다[4]. 이 방법은 모든 flow가 하나의 FIFO 큐를 통하여 처리가 되며 네트워크 상태에 따라 관리된다. 따라서 적은 비용으로도 네트워크

에 발생하는 혼잡 상황 문제를 해결할 수 있다.

2.2 혼잡 제어의 동작

기존의 인터넷에서는 혼잡상황을 해결하기 위한 방법으로 프로토콜 레벨의 종단간 흐름 제어(End-to-End Flow Control) 방법을 이용하고 있다. 이 방법은 TCP와 같은 전송 프로토콜을 기반으로 하며 수신측(TCP Receiver)으로부터 피드백(feedback) 받은 네트워크 정보를 이용하여 TCP 송신측(TCP Sender)에서 각 연결의 전송률을 윈도우 기반으로 제어한다. 혼잡 제어 시 사용되는 네트워크 정보는 네트워크를 구성하는 각 라우터의 메커니즘에 의해서 결정된다. 즉, 이러한 방법으로 상호 동작하는 시스템을 피드백 시스템이라 한다. 현재 인터넷을 통하여 사용하고 있는 TCP는 기본적으로 혼잡상황을 제어하기 위한 동작 메커니즘으로 아래와 같은 congestion avoidance(혼잡 회피) 방법을 가지고 있으며 동작은 다음과 같다.

예를 들어 TCP 플로우가 B 바이트의 패킷을 전송한다고 가정하면, 패킷이 망에서 폐기되거나 손실 될 때 TCP의 혼잡 윈도우는 W 개의 크기를 갖는다. TCP는 패킷 폐기나 손실에 의해 혼잡 윈도우의 크기를 반으로 줄이게 되고 다음 패킷 폐기 때까지 각 RTT 기간 동안 혼잡 윈도우의 크기를 하나씩 증가시킨다. 이를 도식하면 그림 1과 같다.

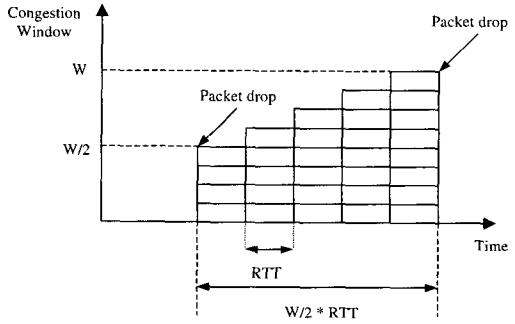


그림 1 TCP 혼잡 윈도우 크기의 변화

이러한 TCP의 동작을 혼잡상황이 발생하는 망에서 다수의 플로우에 대하여 표현하여 보면 그림 2와 같이 n개의 TCP 플로우에 대하여 혼잡제어 동작을 하는 피드백 시스템을 표시 할 수 있다[7].

이 시스템은 크게 두 가지 모듈로 구성이 되며, 각 모듈은 전송률을 제어하는 TCP 송신측 모듈(TCP sender)과 라우터에서 패킷 폐기율을 결정하는 폐기 모듈(drop module)로 구성이 된다.

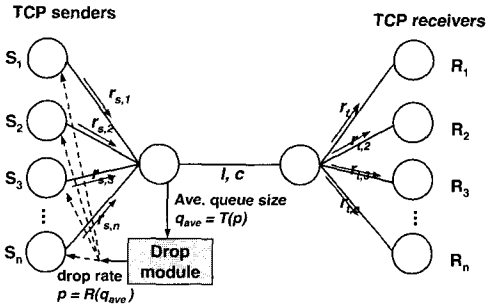


그림 2 n개의 플로우 피드백 시스템

먼저 TCP 송신측에서는 앞에서 설명한 현재 TCP에서 사용하는 congestion avoidance 메커니즘을 사용한다. 다시 말해, TCP의 혼잡제어 알고리즘은 큐의 오버플로우(overflow)가 발생하거나 패킷이 폐기되어 패킷이 손실되면, TCP 송신측은 수신측으로부터 받은 피드백 정보를 통하여 혼잡상황을 인지하고 소스의 전송률을 반으로 줄여 혼잡상황을 해결하려 한다. 그리고 TCP 송신측에서 인지하는 혼잡상황은 피드백을 통하여 전송 받는 ACK 메시지를 통하여 네트워크에서 발생된 패킷 손실 정보를 통하여 알아내며, 패킷 손실 정보는 라우터에서 발생되는 패킷 폐기율(drop rate)에 비례적으로 결정된다. 그리고 패킷 폐기율은 라우터나 게이트웨이에서 사용하는 라우터 메커니즘인 큐 관리 알고리즘이나 스케줄링 알고리즘으로 구성된 폐기 모듈에 위해서 결정된다.

이와는 반대로 네트워크 레벨에서 혼잡상황을 제어하는 패킷 폐기 모듈인 큐 관리 알고리즘이나 스케줄링 알고리즘은 평균 큐 크기나 트래픽 로드의 양에 비례적인 동작을 한다. 즉 네트워크 트래픽의 증가로 인하여 변화하는 라우터의 큐 크기 변화 정보를 이용하여 네트워크의 혼잡상황을 인지하며 이러한 큐의 크기 변화 정보를 이용하여 네트워크 트래픽 상황에 적합한 패킷 폐기율을 결정한다. 폐기 모듈에서 계산되는 패킷 폐기율은 네트워크의 트래픽 상황을 결정하는 TCP 송신측의 전송률에 의존적이며 라우터에서의 패킷 폐기율의 크기와 TCP 송신측의 전송률의 크기는 반비례적인 관계를 가지고 있다. 즉, 폐기 모듈은 TCP 송신측에서 보내는 전송률의 크기의 정도에 따라 패킷 폐기율을 결정한다. 이러한 동작의 상관 관계는 Victor Firoiu의 논문[7]에 근거하여 TCP 플로우의 전송률을 나타내는 f 함수와 패킷 드랍 확률을 나타내는 p와 TCP 연결의 RTT (round trip time)인 R, TCP의 윈도우 크기인 W, 그리

고 평균 패킷 크기인 M로 다음과 같은 수식을 유도 할 수 있다[8].

$$W(p)=\frac{2}{3}+\sqrt{\frac{-8}{6p}+\frac{4}{9}} \quad (2.1)$$

$$Q(p,w)=\min(1,\frac{(1-(1-p)^3)(1+(1-p)^3(1-(1-p)^{w-3}))}{1-(1-p)^w} \quad (2.2)$$

$$F(p)=1+p+2p^2+4p^3+8p^4+16p^5+32p^6 \quad (2.3)$$

$$f(p,R)=\begin{cases}M\Big/\left(R(W(p)+1)+\dfrac{Q(p,W(p))F(p)T_0}{1-p}\right) & \text{if } W(p)<W_{\max}\\ M\Big/\left(R(\frac{1}{4}W_{\max}+\dfrac{1-p}{pW_{\max}}+2)+\dfrac{Q(p,Wmax)F(p)T_0}{1-p}\right) & otherwise\end{cases} \quad (2.4)$$

TCP 각 플로우의 전송률로 인하여 발생되는 라우터 평균 큐 크기 $T(p)$는 식 (2.5)로 표현 할 수 있다.

$$T(p)=\begin{cases}\max(B, c(f^{-1}(p,c/n)-R_0)), & p\le p_0\\ 0, & otherwise\end{cases} \quad (2.5)$$

폐기 모듈에서 결정하는 평균 큐 크기에 따른 패킷 폐기 함수는 RED 알고리즘을 가정할 경우 그림 3 (B는 라우터의 최대 큐 크기)과 같은 패킷 폐기 함수를 가지며 식(2.6)으로 패킷 폐기 함수 $R(q_{ave})$을 유도 할 수 있다.

$$R(q_{ave})=\begin{cases}0, & 0\le q_{ave}<\min_{th}\\ \dfrac{q_{ave}-\min_{th}}{\max_{th}-\min_{th}}P_{\max}, & \min_{th}\le q_{ave}<\max_{th}\\ 1, & \max_{th}\le q_{ave}\le B\end{cases} \quad (2.6)$$

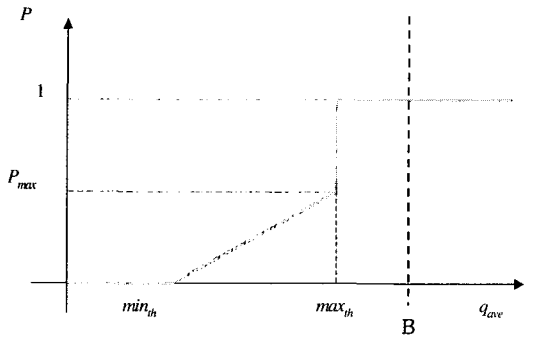


그림 3 RED 알고리즘의 패킷 폐기함수

앞에서 유도한 $T(p)$와 $R(q_{ave})$을 이용하여 그림 4와 같은 관계를 도시 할 수 있다.

그림 4와 같이 전송률과 폐기율의 관계는 두 값에 비례적인 평균 큐 크기(q_{ave})와 패킷 폐기 확률 값(p)으로 표현이 되며 TCP 송신 측과 폐기 모듈사이의 상관관계는 두 개의 함수($q_{ave}=T(p)$, $p=R(q_{ave})$)로 표현 할

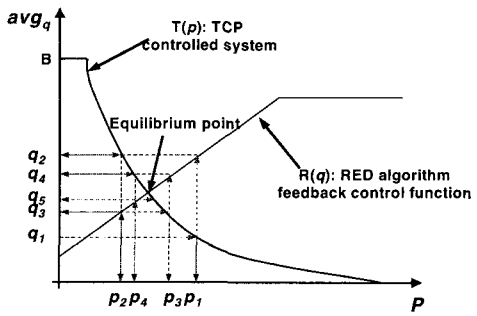


그림 4 RED를 이용하는 피드백 시스템의 평형 상태

수 있다. 그리고 두 함수는 서로 상관관계를 통하여 임의의 평형 상태 점 (Equilibrium point)으로 수렴하도록 동작한다. 즉 네트워크 혼잡상황이 발생하면 네트워크 트래픽 특성에 맞도록 송신 측의 전송률과 라우터의 패킷 폐기 확률이 결정되어 평형 상태로 수렴된다. 예를 들어 라우터의 폐기 모듈과 송신 측 모듈이 네트워크 혼잡상황에 대하여 적절한 혼잡 제어를 할 경우 두 모듈은 평형 상태를 가지는 하나의 값으로 수렴한다. 만약 네트워크 혼잡상황에 대하여 적절한 혼잡제어를 하지 못할 경우 두 모듈은 평형 상태로 수렴 할 수 없게 된다. 즉, 두 모듈이 평형 상태로 수렴을 하지 못 할 경우 라우터의 폐기 모듈에서는 높은 패킷 폐기율을 가지게 되며 송신 측 모듈은 적절한 전송률을 선택하지 못하여 불규칙적인 전송을 하게 된다. 이로 인하여 네트워크의 높은 패킷 손실에 따른 네트워크 성능 감소와 라우터 버퍼 큐의 크기의 심한 변동에 따른 높은 지터(jitter)를 가지는 문제가 발생한다. 따라서 두 모듈이 평행 상태로 수렴 할 수 있는 적절한 각 모듈의 설정이 필요하다. 다음 장에서는 이러한 RED 알고리즘의 문제점을 알아보기 위해서 간단한 실험을 통하여 RED 알고리즘의 동작 특성을 분석하였다.

3. RED 알고리즘의 분석

본 장에서는 기존의 RED 알고리즘의 문제점을 알아보기 위해서 간단한 실험을 통하여 RED 알고리즘의 동작 특성을 분석하여 보았다.

3.1 RED 알고리즘의 동작

RED 알고리즘은 혼잡 상황이 발생하기 이전에 평균 큐 크기를 기반으로 그림 5와 같이 평균 큐 크기의 변화의 정도를 기반으로 혼잡 상황을 판별하며, 혼잡 상태에 대한 피드백 정보를 패킷의 폐기나 패킷 헤더에 혼잡 상황을 표시(marking)하는 ECN을 이용하여 종단

호스트인 송신 측에게 혼잡정보를 알려 준다.

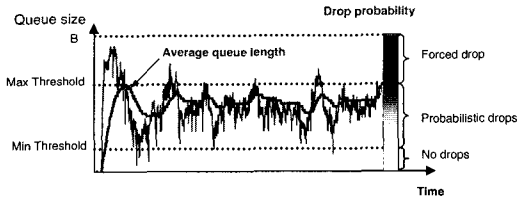


그림 5 RED 알고리즘의 동작

RED 알고리즘은 혼잡상황이 발견되면 EWMA (Exponential Weighted Moving Average)를 이용하여 평균 큐 크기를 계산하며 평균 큐 크기에 따라 적절한 패킷 폐기 확률 값을 계산한다. 예를 들어 평균 큐 크기가 최소 경계 값(minimum threshold: min_{th})을 초과하면 패킷들은 패킷 폐기 확률 값에 따라 랜덤 하게 폐기되거나 패킷 헤더의 ECN bit를 이용하여 표시한다. RED 알고리즘은 네트워크 혼잡상황을 피하기 위해서 TCP와 같이 피드백 정보를 기반으로 혼잡제어를 하는 전송 프로토콜과 상호 동작한다. 그리고 평균 큐 크기가 최대 경계 값(maximum threshold: max_{th})을 초과할 경우 모든 패킷들은 폐기 또는 표시한다.

RED 알고리즘의 동작 방법은 이처럼 기존의 혼잡제어 방법과는 다르게 혼잡상황을 예상하여 도착하는 임의의 플로우의 패킷을 폐기하기 때문에 drop tail 방법이 가지고 있는 "락 아웃(lock out)" 과 "풀 큐(full queue)" 문제를 해결 할 수 있다. 또한 미리 혼잡상황을 발견하여 TCP의 전송률을 줄여 혼잡제어를 수행하기 때문에 불필요한 패킷 손실로 발생하는 네트워크 자원의 낭비를 drop tail 방법 보다 줄여 좋은 성능을 나타낼 수 있다. 그리고 효과적인 큐 관리로 평균 큐 크기를 작게 유지할 수 있어 큐잉(queueing)으로 생기는 전송지연 시간을 줄일 수 있다. 일반적으로 RED 알고리즘을 이용하는 방법이 Drop tail 방법을 사용하는 것보다는 좋은 성능을 제공한다[9].

Drop tail보다 좋은 성능을 제공하는 RED 알고리즘은 표 1과 같이 다양한 매개 변수에 의해서 제어되며 그 중에서도 max_p 값에 따라 전체적인 알고리즘의 특성이 결정된다[10].

RED 알고리즘은 위의 표 1과 같이 다양한 매개변수로 설정되어 동작하기 때문에 만약 네트워크 특성에 맞지 않는 잘못 된 매개변수로 설정이 될 경우 혼잡상황에 대하여 적절한 혼잡제어를 하지 못 한다. 최악의 경우 잘 못 된 매개변수 설정으로 인하여 전체적인 성능이 Drop tail 보다 떨어질 수도 있다[11].

표 1 RED 제어 매개 변수

alen	최대로 수용할 수 있는 큐 공간 크기
min_{th}	폐기 확률을 결정하는 최소 임계 값
max_{th}	폐기 확률을 결정하는 최대 임계 값
w_a	Avg_a 변화의 특성을 결정하는 가중치 값
max_b	폐기 특성을 결정하는 최대 확률 값

3.2 RED 알고리즘의 문제점

RED 알고리즘의 동작 특성을 자세히 알기 위해서 그림 6과 같은 네트워크 환경에서 ns(network simulation) [12]을 이용하여 RED 알고리즘의 매개변수인 최대 확률 값(P_{max} or max_p)에 따른 네트워크 트래픽의 특성과의 상관관계에 대하여 실험하였다.

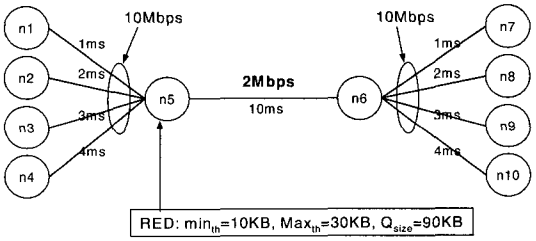


그림 6 실험 네트워크 설정

그림 6과 같은 네트워크 환경에서 왼쪽의 임의의 호스트(n1,n2,n3,n4)들로부터 오른쪽의 임의의 호스트(n7,n8,n9,n10)로 32개의 TCP 플로우가 연결되어 트래픽을 전송할 때와 64개의 TCP 플로우가 연결되어 트래픽을 전송할 때, 혼잡상황이 발생되는 중간 라우터에서 RED 알고리즘의 매개변수 max_p를 0부터 1까지의 다양한 값으로 조절할 때 라우터를 지나는 트래픽들의 패킷 폐기율을 알아보았다. 그림 7과 같이 max_p 값의 변화에 따라 패킷 폐기율은 가변적으로 변화하였다.

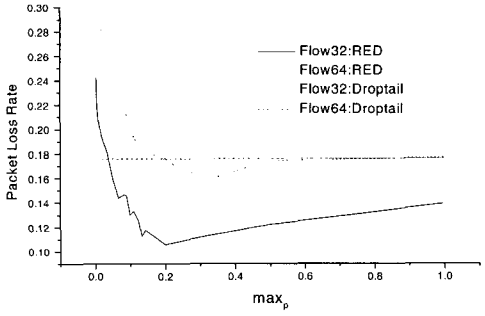


그림 7 패킷 폐기(손실)율에 대한 실험 결과

실험 결과에서와 같이 플로우의 수에 따라 최소 패킷 폐기율을 결정하는 max_p 값이 서로 다른 것을 확인할 수 있다. 즉 트래픽의 특성에 따라 RED 알고리즘의 매개변수가 설정되어야만 최소 패킷 폐기율 가지며 효과적으로 혼잡상황을 제어할 수 있다. 예를 들어 적절한 max_p 값이 설정되어 있지 않을 경우 앞장에서 설명한 피드백 시스템의 두 모듈의 상관 관계는 그림 8과 같은 분포를 가지게 되며 두 상관 함수 $q_{ave}=T(p)$, $p=R(q_{ave})$은 평형 상태로 수렴하는 평형 상태 점(p, q_{ave})을 가지지 못하게 된다.

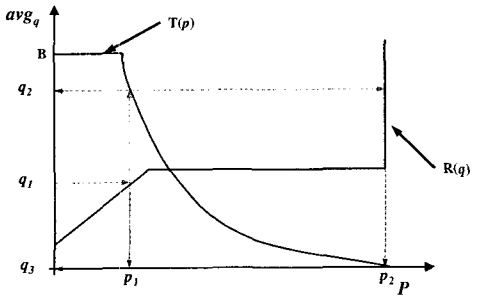
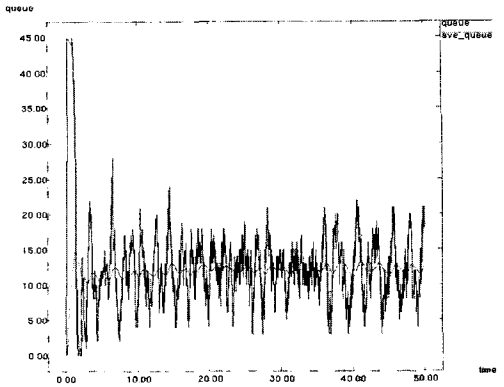


그림 8 잘못 설정된 maxp 값을 가진 RED 알고리즘의 피드백 동작

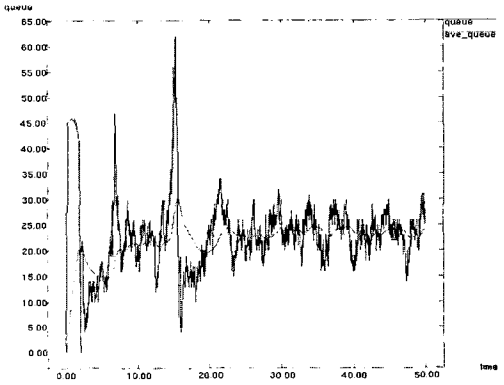
추가적으로 RED 알고리즘의 파라메터 설정에 따른 라우터 큐 크기 변화를 알아보기 위해서 8개의 플로우에 대하여 서로 다른 max_p를 설정했을 때의 큐 크기 변화를 모니터링 하여 보았다. 그림 8과 같이 서로 다른 max_p 값에 의해서 큐 크기의 변화가 심하게 차이가 남을 확인 할 수 있었다.

그림 9의 (a) 결과에서와 같이 적절한 max_p 값보다 너무 높은 max_p값이 설정되어 있을 경우 필요 이상의 패킷 폐기로 인하여 전체적인 큐 크기 및 평균 큐 크기가 작아져 좋은 전송 성능을 가지지 못한다. 그리고 적절한 max_p 값보다 너무 적은 max_p값을 가질 경우 그림 9의 (c)와 같이 큐의 변동이 심해진다. 이로 인하여 불규칙적인 패킷 손실 및 재전송으로 인한 각 플로우의 지터(jitter)의 크기가 커지며, 라우터 버퍼의 overflow로 인하여 많은 패킷이 손실되어 전체적인 네트워크 성능이 감소한다. 그림 9의 (b)와 같이 네트워크 특성에 적합한 max_p값이 선택되어지는 경우 적당한 평균 큐 크기를 유지하며, 큐 크기 변동이 적어져 안정적인 네트워크 서비스를 제공할 수 있다[13,14,15].

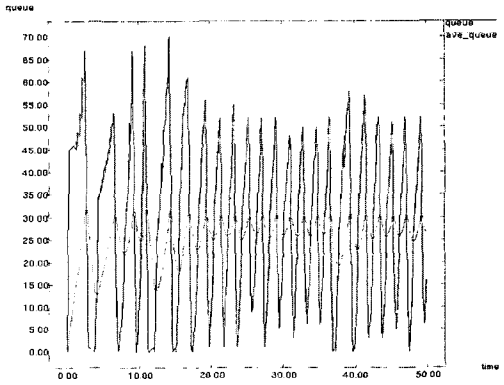
네트워크 트래픽 특성과 RED 알고리즘의 max_p와의 상관관계는 그림 10과 같이 두 모듈의 상관관계 그래프



(a) max_p = 0.90



(b) max_p = 0.1



(c) max_p = 0.004

그림 9 max_p 값에 따른 큐 크기의 변화

로 표현 할 수 있다. 그림에서와 같이 플로우의 수가 증가 할 수록 T(p)는 오른쪽으로 점점 크기가 커지는 특성을 가지고 있다. 예를 들어 RED 알고리즘의 매개변수들이 고정적일 경우 트래픽의 양이 증가하게 되면 고정적으로 설정되어 있는 RED 알고리즘의 매개변수

max_p 값 밖으로 T(p)가 커져 평형 상태로 수렴을 하지 못하는 관계를 가지게 된다. 또한 필요이상의 큰 max_p 값이 설정되어 질 경우 높은 패킷 폐기율로 인하여 낮은 평균 큐 크기를 가지게 된다. 이 경우 전체적인 전송률이 떨어지고 또 큐의 크기 변화가 심해져 네트워크 효율성이 나빠진다.

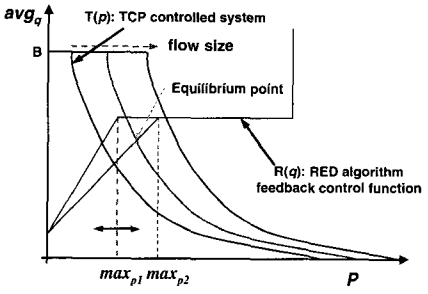


그림 10 max_p와 트래픽 특성

따라서 네트워크 특성(플로우의 수)에 따라 능동적으로 동작하는 새로운 큐 관리 알고리즘이 필요하다. 즉 플로우 수에 따라 적절한 max_p 값을 능동적으로 조절하는 방법이 필요하다[14].

본 논문에서는 동적으로 변화하는 네트워크 트래픽으로 인하여 RED 알고리즘의 중요 매개변수인 max_p 값이 잘 못 설정되어 발생하는 높은 패킷 폐기율을 해결하기 위해서 능동적으로 매개변수를 설정하는 ARED (Active RED) 알고리즘을 제안한다.

4. Active RED 알고리즘

본 논문에서는 기존의 RED 알고리즘의 문제점을 해결하기 위해서 ARED 알고리즘을 제안하였다. 제안하는 ARED 알고리즘은 네트워크 트래픽의 특성에 따라서 RED 알고리즘의 매개변수를 능동적으로 조절하는 알고리즘이다. ARED 알고리즘의 구성은 그림 11과 같다.

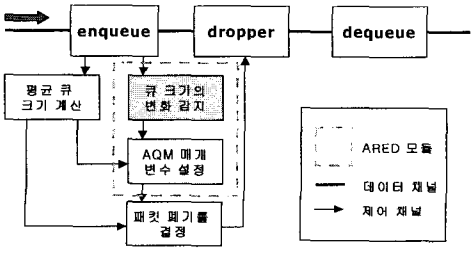


그림 11 Active RED 알고리즘의 구성

그림 11과 같이 ARED 알고리즘의 동작 방법은 네트워크 트래픽의 변화가 발생 할 때 큐 크기의 변화율을 통하여 혼잡상황을 인지한다. 만약, 네트워크에 적합하지 않은 라우터 알고리즘의 매개변수가 설정되어 있을 경우 큐의 크기는 심하게 진동한다. 이러한 진동은 보통 적합한 max_p 값보다 너무 작은 값이 설정되어 있을 경우 발생한다. 그리고 ARED 알고리즘은 필요 이상의 패킷 손실을 방지하기 위해서 그림 11과 같이 평균 큐 크기의 변화 정도에 따라 매개변수 값을 설정한다. 따라서, 큐 크기가 심하게 진동하거나 특정 임계값 미만으로 평균 큐 크기가 작아질 경우 ARED 알고리즘은 큐 크기의 변화율을 줄여 가능하면 평형 상태에 수렴하도록 max_p 값을 조절하고 또한, 필요 이상의 패킷 폐기율이 발생하지 않도록 max_p값을 조절한다.

ARED 알고리즘은 동적인 네트워크 변화에 효과적으로 최적의 매개변수 값을 가지도록 그림 11의 AQM 매개변수 설정 모듈은 다음과 같은 동작을 한다. 매개변수 max_p 값이 최적의 값(현재 네트워크 특성에 적합한 값)보다 작게 설정되어 있을 경우 큐의 변화가 심하면서 오랜 시간 동안 평균 큐의 크기가 줄지 않는다. 이러한 상황을 발견하기 위해서 특정 주기 동안의 큐의 변화를 감지하여 주기 동안 변화가 계속 심하게 발생할 경우 max_p 값이 잘못 설정됐다고 가정한다. ARED 알고리즘은 이런 상황을 발견하여 d_1 만큼 max_p 값을 증가시켜 매개변수 값을 조절한다. 매개변수 값을 재설정하는 주기는 라우터에서 패킷대하여 폐기한 정보가 수신 측에서 송신 측으로 전달되어 송신 측의 전송률을 조절하여 네트워크에 트래픽을 줄일 수 있을 정도의 시간을 고려하여 결정되어야한다. 만약 너무 자주 매개 변수를 재설정 할 경우 이전에 폐기한 패킷에 의해서 변화되는 트래픽에 대하여 잘 적응하지 못해서 큐 크기의 변화가 심해질 수 있다. 따라서 매개변수를 재설정하는 주기도 트래픽의 유입량을 고려하여 설정해야한다. 본 논문에서는 간단한 실험 환경을 고려하여 고정적인 주기를 가정하여 시험 하였다.

다음 주기 동안 반복적으로 매개변수를 조절하여 매개변수 값이 최적의 값으로 설정하여 평형 상태로 수렴하도 동작한다. 그리고, max_p가 최적의 값보다 너무 클 경우 불필요한 패킷 폐기로 인하여 평균 큐 크기가 최대 임계 값 보다 낮은 크기를 유지하여 큐를 비효율적으로 관리하게 된다. 이러한 상황을 발견하기 위해서 평균 큐 크기의 최대 임계 값을 설정하여 임계 값 미만으로 평균 큐 크기가 작아질 경우 최적의 값보다 너무 큰 값을 설정했다고 가정한다. 이 경우 ARED 알고리즘은

불필요한 패킷 폐기율을 낮추기 위해서 d_2 만큼 max_p를 줄인다. 여기서, 최대 임계 값은 네트워크 서비스의 특징을 결정하는 파라메터이다. 즉 최대 임계 값을 크게 설정 할 경우 전반적인 전송률은 증가하지만 지연시간이 증가하고 작게 설정할 경우 전송률은 감소하지만 지연시간이 감소하는 특징을 가지고 있다. 따라서 최대 임계값의 설정은 네트워크 서비스 사업자가 네트워크 서비스의 특징을 결정할 때 고려되어야 한다. 제안한 ARED 알고리즘은 그림 12와 같다.

```
if (평균 큐의 변화의 정도가 큼
  && (ΔW_avg_que < margin1 * ΔW_current_max))
  max_p 증가

if (ΔT 동안 평균 큐의 크기 감소
  && (평균 큐의 크기 < margin2 * 최대 임계 값))
  max_p 감소

if (max_p 증가)
  max_p = max_p + d1;

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

그림 12 Active RED 알고리즘

본 논문에서 제안하는 ARED 알고리즘의 동작은 임의의 시간 ΔT 동안의 평균 큐 크기의 변화의 정도를 감시하며 큐 크기의 변화 중 ΔT 동안의 큐 최대 크기와 ΔT 동안의 큐 최소 크기의 차인 최대 변화 폭 ΔW(width)를 계산하여 변화 폭이 점점 커질 경우 max_p 값을 조절한다. 즉, 잘 못 된 매개변수가 설정되어있을 경우 발생하는 심한 큐 크기의 변동에 대하여 큐 크기 변화의 폭을 가능하면 줄어들도록 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 변화의 원인을 파악하여 매개변수 max_p 값이 최적의 값을 가지도록 동작을 한다. 결국, ARED 알고리즘은 이러한 큐의 크기의 변동의 정도를 줄여 최적의 매개변수를 유지하면서 동작한다. 또한 평균 큐의 최대 임계값을 설정하여 필요이상의 패킷 폐기율이 발생하지 않도록 max_p 값을 조절한다. 따라서, ARED 알고리즘의 기본동작의 목적은 잘 못된 매개변수 설정으로 발생되는 전체적인 패킷 폐기(손실)율을 줄여 효율적인 라우터의 큐를 관리하는 알고리즘이다.

5. 실험 및 성능 평가

본 논문에서 제안한 ARED 알고리즘의 성능을 알아보기 위해서 두 가지 실험을 하였다. 첫 번째 실험은 잘 못된 매개변수 설정에 대하여 ARED 알고리즘이 매개변수를 능동적으로 조절하는지를 알기 위해서 RED 알고리즘과 ARED 알고리즘의 큐 크기의 변화를 알아보았다. 두 번째 실험은 플로우의 수를 변화 시켰을 때 전체 패킷 손실률을 RED 알고리즘과 ARED 알고리즘에 대하여 비교하여 보았다.

5.1 큐 크기의 변화

앞장에서 설명한 피드백 시스템에서의 라우터 큐 크기의 변화가 진동하는 것은 매우 일반적인 현상이다. 이러한 현상은 피드백 시스템에서 사용하는 TCP의 혼잡제어 메커니즘 때문에 발생하며, 네트워크의 혼잡상황에 따라 매우 가변적인 특성을 가진다. 특히, 잘 못된 네트워크 설정으로 인하여 혼잡상황을 제대로 제어하지 못할 경우, 라우터에서 큐 크기의 변화가 큰 폭으로 발생하며 전반적으로 네트워크의 성능을 저하시키는 결과를 가져온다. 예를 들어, 큐 크기의 변화 폭이 클 경우 큐에서는 오버플로우 현상이 발생하여 높은 패킷 손실률이 발생하며 재 전송으로 인한 네트워크 리소스를 낭비하게 된다. 또한 큐 크기의 변화가 클 경우 큐잉 지연의 변화가 커져 네트워크 서비스 품질이 저하된다.

라우터 알고리즘에 따라 변화하는 큐 크기의 변화를 알아보기 위해서 앞에서 실험한 실험 환경인 그림 6과 같은 네트워크 환경에서 실험하였다.

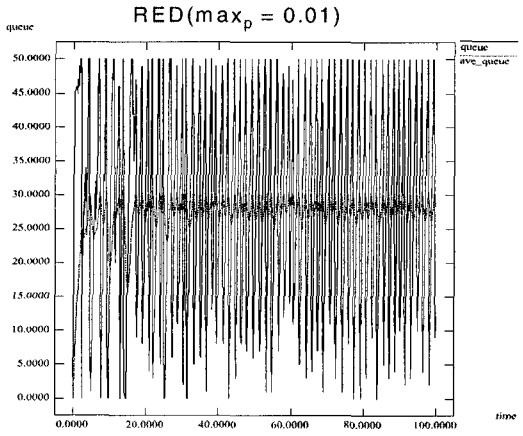
표 2 RED과 ARED 알고리즘의 매개변수 설정 값

	RED	*ARED*
W_q	*0.002*	*0.002*
qlen	*90KB*	*90KB*
min_{th}	*10KB*	*10KB*
max_{th}	*30KB*	*30KB*
P_{max}	*0.01*	*0.01*
ΔT	–	*1500packet/ ΔT*
$margin_1$	–	*0.9*
$margin_2$	–	*0.9*
d_1	–	*0.01*
d_2	–	*0.03*

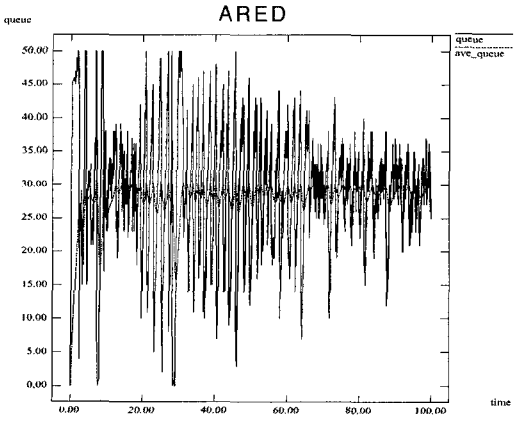
본 논문에서 실험하는 각 알고리즘의 매개변수는 표 2와 같이 설정을 하였다. RED 알고리즘의 설정은 [9]에서 권고하는 설정을 이용하였다. 그리고 ARED 알고리즘의 각 매개변수 설정은 1500개의 패킷 단위 주기로 1500 packet/ ΔT, ΔW의 변화를 10%로 가정하여 $margin_1$을 0.9로 최대 임계 값에 10% 범위 안으로 가정하여 $margin_2$을 0.9로 주기 당 증가치인 d_1은 0.01로

주기 당 파라메터 감소치인 d_2는 0.03으로 설정하였다.

실험 네트워크에 총 16개의 플로우를 전송하여 혼잡 링크인 라우터 n5에서의 큐 크기의 변화를 모니터링 하였다. 시간당 혼잡상황이 발생되는 라우터의 큐 크기 및 평균 큐 크기를 도시하면 그림 13과 같다. 그림 13의 (a)와 같이 기존의 RED 알고리즘을 사용할 때, 현재 네트워크 트래픽의 특성에 RED 알고리즘의 매개변수가 적합하지 않을 경우 큐 크기가 심하게 변화됨을 확인할 수 있다. 본 논문에서 제안한 ARED 알고리즘을 적용하였을 경우 현재 네트워크 트래픽 특성에 맞게 매개변수를 조절하여 그림 13의 (b)와 같이 큐 크기의 변동의 정도를 줄이도록 동작하고 있음을 확인하였다. 불필요한 큐 크기의 변화는 응용프로그램에게 높은 지터를 가지게 하여 네트워크의 서비스를 나쁘게 한다.



(a) RED 알고리즘



(b) ARED 알고리즘

그림 13 RED와 ARED의 큐 크기의 변화

5.2 패킷 폐기율

라우터에서의 패킷 폐기율의 결정은 네트워크의 효율성을 결정한다. 필요 이상의 패킷을 폐기하거나 적절한 패킷 폐기율을 가지지 못 할 경우 네트워크의 효율성은 급격히 떨어진다. 예를 들어 패킷 폐기율을 너무 높을 경우 네트워크에서 처리 할 수 있는 트래픽의 양을 감소시켜 네트워크의 자원을 완전히 활용하지 못하는 경우를 발생시킨다. 그리고 패킷 폐기율이 적정 폐기율보다 너무 낮을 경우 혼잡상황을 계속 야기 시켜 네트워크의 성능을 급격히 저하시킨다. 따라서 라우터에서 네트워크 상황에 맞는 패킷 폐기율을 결정하는 것은 매우 중요하다.

본 장에서는 앞장에서 설명한 잘못된 매개변수의 설정 시 불필요한 패킷 폐기율을 가지는 라우터 알고리즘을 개선한 ARED 알고리즘의 성능을 실험하였다. 네트워크의 트래픽 특성을 결정하는 플로우의 수의 변화에 따른 ARED 알고리즘의 동작을 알아보기 위해서 라우터에서 발생하는 패킷 폐기율을 계산하였다. 그림 6과 같은 네트워크 환경해서 FTP 플로우의 수를 4개에서 64개로 변화 시키면서 150초 동안 전체 플로우의 패킷 폐기율을 알아보았다. 2^n 플로우당 각 큐 관리 알고리즘에서 폐기하는 패킷의 비율을 계산하면 그림 14와 같다. 그림 14와 같이 두 알고리즘 모두 플로우 수가 증가함에 따라 패킷 폐기율이 증가하는 것을 알 수 있다. 그림 14에서 보는 것과 같이 본 논문에서 제안한 ARED 알고리즘을 사용 할 경우 기존의 RED 방법보다 전반적으로 낮은 패킷 폐기율을 나타냄을 알 수 있다. ARED 알고리즘은 RED 알고리즘에 비하여 네트워크 상황에 맞게 적절한 매개변수를 설정하여 불필요한 큐 크기의 변화를 줄였다. 이를 통하여 라우터에서의 오버플로우 및 불필요한 패킷 재 전송을 줄여 혼잡상황에서도 안정적으로 동적인 네트워크 트래픽을 처리한다.

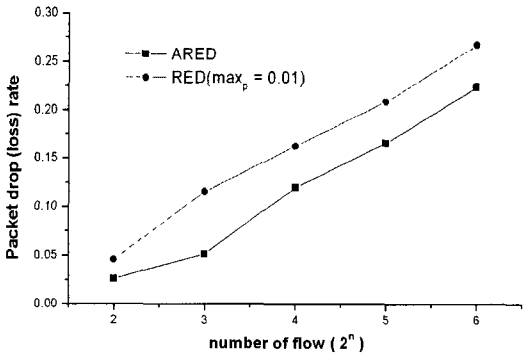


그림 14 RED와 ARED의 패킷 폐기율

두 실험 결과를 통하여 본 논문에서 제안한 ARED 알고리즘은 라우터 알고리즘의 중요한 매개변수를 수동적으로 설정하지 않아도 네트워크 특성에 따라 능동적으로 매개변수 max_p 값을 조절 및 설정하여 동적인 네트워크 트래픽에 적응 적인 동작을 한다.

6. 결론 및 향후 과제

현재 인터넷을 구성하고 있는 라우터에서는 Drop tail 방법을 사용하고 있다. 이 방법은 네트워크 혼잡상황을 명시적으로 해결하지 못하는 문제점을 가지고 있다. 이 문제점을 해결하기 위해 IETF에서는 RED 알고리즘과 같은 큐 관리 알고리즘을 권고하고 있다. 본 논문에서는 동적으로 변화하는 네트워크 트래픽으로 인하여 발생하는 혼잡상황에 대하여 고정적인 매개변수 설정으로 동작하는 RED 알고리즘의 문제점을 분석하고 이를 개선한 새로운 ARED 알고리즘을 제안하였다. ARED 알고리즘은 능동적으로 네트워크 상태에 따라서 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. 결과를 통하여 기존의 방법 보다 개선된 성능을 보임을 확인하였다.

향후 연구 과제로는 비반응(unresponsive) flow에 대한 공정성을 개선하는 방법에 대한 연구가 수행되어야 하며, 실제 네트워크 환경에서의 적용에 대한 관련 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] W. Feng, D. Kandlur, D. Saha, and K. Shin., "Techniques for Eliminating Packet Loss in Congested TCP/IP Network," U. Michigan CSE-TR-349-97, November 1997.

[6] Demers, A., Keshav, S. and Shenker, S., "Analysis and simulation of a fair queueing algorithm," Journal of Internetworking Research and Experience, October 1990. Also in Proceedings of ACM SIGCOMM' 89.

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

[8] Padhy, J., Firoiu, V., Towsley, D., and Kurose, J., "A Stchastic Model of TCP Reno Congestion Avoidance and Control," Technical Report CMPSCI TR 99-02, Univ. of Massachusetts, Amherst, 1999.

[9] G. Iannaccone, M. May, and C. Diot, "Aggregate Traffic Performance with Active Queue Management and Drop from Tail," Computer Communication Review, July 2001.

[10] S. Floyd, "RED: Discussions of Setting Parameters," November 1997. http://www.aciri.org/floyd/REDparameters.txt.

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

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

[13] C. Hollot, V. Misra, D. Towsley, and W. Gong, "On Designing Improved Controllers for AQM Routers Supporting TCP Flows," INFOCOM 2001, 2001.

[14] J. Aweya, M. Ouellette, D. Y. Montuno and A. Chapman., "An Optimization-oriented View of Random Early Detection," Computer Communications, 2001, to appear.

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

구 자 헌

1999년 광운대학교 전자통신공학과 학사. 2001년 광운대학교 전자통신공학과 석사. 2001년 ~ 현재 광운대학교 전자통신공학과 박사과정. 관심분야는 컴퓨터통신, 인터넷, QoS, 멀티미디어통신

정 광 수

정보과학기회논문지 : 정보통신 제 29 권 제 3 호 참조