碩士學位論文

DiffServ Network에서 혼잡제어를 위한 Extended RED 알고리즘

濟州大學校 大學院

컴 퓨 터 工 學 科

金 恩 範

2003年 12月

DiffServ Network에서 혼잡제어를 위한 Extended RED 알고리즘

指導敎授 宋 旺 暾

金 恩 範

이 論文을 工學 碩士學位 論文으로 提出함.

2003年 12月

金恩範의 工學 碩士學位 論文을 認准함.

審査委員長 金 壯 亨 印

委 員 邊 翔 庸 印

委 員 宋 旺 暾 印

濟州大學校 大學院

2003年 12月

Extended RED Algorithm for Congestion Control on DiffServ Network

Eun-Bum Kim
(Supervised by professor WangCheol Song)

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENT FOR THE DEGREE OF
MASTER OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING
GRADUATE SCHOOL
CHEJU NATIONAL UNIVERSITY

2003. 12.

목 차

그림목차

표목차

Summary

The Internet of nowadays provides only best-effort service and cannot support the specific services which require QoS guarantee. To provide QoS capabilities to Internet services, many efforts are being made to the existing protocol architectures so far and some of the results are embodied as DiffServ and etc. to fullfil the service requirements such as user bandwidth, packet loss, and delay guarantee etc..

In DiffServ architecture, all the traffic are entering the network, classified at the edge routers and are marked and controlled in a PHB associated with DSCP at the core routers to enable QoS aware service provisionings. Differentiated services only define the DS fields and PHBs and the forwarded traffic is classified as Expedited Forwarding(EF), Assured Forwarding(AF) and Best Effort Forwarding(BE) at edge routers.

In the case of AF, RED can be used to control the queue and to improve the througtput and tail drop behaviors. As the operational parameters of RED are usually fixed to the specific situation of network, they cannot support the various network environments.

In this thesis, I extend the existing RED to improve the overall performance under DiffServ architecture, especially with AF services. Various RED parameters are dynamically changed according to the prediction of network load and its variations, enabling the easy adaptation to environment variation.

The status or the degree of network congestion is evaluated with some prediction and the parameters, Minimum Threshold and Drop Probabilities are adjusted dynamically with this feedback to resolve and improve the drawbacks of the existing RED. The simulation results show this extension of RED is improving the various performance measures such as link utilization rate, packet loss and throughput for each class to some extent.

Ⅰ. 서론

최근 인터넷에서는 인터넷 방송, 화상회의, VoIP 등 QoS(Quality of Service)보장을 요구하는 새로운 응용 서비스들의 출현과 인터넷 사용자 수의 급격한 증가와 함께 IP QoS의 문제는 차세대 인터넷에서 가장 주요한 과제의 하나로 등장하고 있다. 하지만 현재의 인터넷은 모든 패킷을 동일하게 전달하는 Best-Effort 서비스만을 제공하고 있기 때문에 서비스에 따른 패킷의 손실 또는 지연 등의 QoS에 대한 요구 사항을 보장해 주지 못하고 있다. 따라서 인터넷에서 서비스의 QoS를 보장해 주기 위해서는 현재의 모델과는 다른 새로운 서비스 모델을 필요로 한다.

실시간 응용 서비스가 요구하는 QoS를 지원하기 위해 새로운 서비스 모델에 기반을 둔 IP 패킷 전달 방식에 대한 연구가 최근 수년간 IETF WG에서 연구되어 왔다. 이 IETF WG에서 제안된 방식 중 대표적인 것이 IntServ(Integrated Service)와 DiffServ (Differentiated Service)이다. IntServ 모델은 실시간 응용 서비스에서 발생되는 패킷의 흐름을 단위로 하여 패킷을 전달한다. 하지만 이 IntServ 모델은 각 패킷 흐름에 대한 상태 정보를 망의 라우터가 유지하고 있어야 하기 때문에 백본망(Backbone Network)에서는 현실적으로 수용하기에 어려움이 있다. (R. Braden, 1994) (J. Wroclawski, 1997)

이에 따라 확장성의 문제를 갖고 있는 IntServ 모델의 한계를 극복하고 인터넷 백본망에서 적용할 수 있는 서비스 모델로서 DiffServ 모델이 90년대 후반부터 IETF WG에서 활발히 논의되기 시작하여 빠른 속도로 구조 및 관련 표준안이 개발되고 있다. DiffServ 모델은 흐름(Flow) 단위로 QoS를 보장하지 않고 흐름들의 집합(Aggregation)을 단위로 서비스를 차별화함으로써 훨씬 간단하고, 따라서 대규모 망에도 적용 가능하도록 하는 모델이다. (S. Blake, 1998)

DiffServ(DS) 구조는 네트워크에 진입하는 트래픽을 망의 경계에서 분류하고 조절하며, 다른 행동 집합(BA : Behavior Aggregate)에 할당하는 모델을 기초로 한다. DiffServ 네트워크의 핵심 부분인 경계라우터(Edge Router)는 TCB, 버퍼

관리, 스케줄러로 구성되어 있다. TCB(Traffic Conditioning Block)는 트래픽 분류 기능과 조절 기능이 조합되어 있는 블록이다. 트래픽 분류기는 BA 분류기를 사용하여, DSCP 필드를 이용하여 패킷을 분류한다. 마킹된 패킷은 큐잉 단계로 넘어가며 버퍼관리를 위해 RED(Random Early Detection), RIO(RED with In and Out), WRED(Weighted RED), FRED(Flow RED) 등의 차별화가 가능한 방법이 사용된다.

AF PHB의 버퍼관리를 위해 사용되는 RED 메커니즘은 라우터에서 체증이 발생하기 전에 능동적인 버퍼관리를 통하여 전역 동기화(global synchronization)에 의해 순간적으로 링크의 이용률이 감소하는 것을 방지하고 라우터 내의 큐 길이를 가능한 작은 크기로 유지케 하는 방법이다. (Sally Floyd, 1993)

RED는 가중 평균 큐 길이에 따라 망의 체증 정도를 결정하여 라우터에 들어오는 패킷을 폐기하는데 적용되는 확률이 결정되며, 이 RED 방식을 사용할 경우 Drop-Tail 방식과 비교하여 링크의 사용 효율을 높일 수 있다.

그러나 RED에서 사용하는 매개변수들은 망의 상황과 상관없이 동일한 값으로 설정되기 때문에 다양한 트래픽 상황에 적용하게 되면 여러 가지 문제점이 발생한다. 일반적으로 버스트한 데이터의 특성을 갖는 망에서는 최소 큐 한계값(TH_{min})을 크게 하면 링크의 사용 효율을 높일 수 있으며, 지속적으로 망에 과부하가 가해지고 있을 때는 최대확률(P_{max}) 값을 높게 설정하여야 효율적인 체증관리가 가능하다고 알려져 있다. 현재의 망의 특성은 버스트한 데이터의 특성과 지속적인 과부하의 망 특성을 동시에 갖고 있어 동일한 매개변수의 값을 설정하면 상이한 두 환경에서 잘 적용되기란 불가능하다.

본 논문에서는 DiffServ Network AF PHB에서 버퍼 관리 메커니즘으로서 망의 전체적인 부하를 감지하는 방법을 제안하고 이 방법을 이용하여 망의 상태에 따라 각각의 매개변수의 값을 자동적으로 조절하여 다양한 망 환경에서도 잘 적용될 수 있는 확장 RED를 제안하고자 하며, 다음과 같이 구성된다.

2장에서는 차별화 서비스 배경과 구성요소, AF PHB에서의 버퍼 관리 메커니즘에 대해서 기술하고, 3장에서는 본 논문에서 사용된 확장 RED 알고리즘을 소개한다. 4장에서는 차별화 서비스의 AF 트래픽에 적용된 확장 RED의 모의실험 및 결과를 제시한다. 5장에서는 결론을 맺는다.

Ⅱ. 차별화 서비스

1. IntServ 모델과의 차이점

지금까지 인터넷은 망 내의 모든 트래픽을 동일하게 처리하는 “best-effort"라 불리는 단일 서비스 모델만을 제공하고 있으며, 응용별로 차별화된 QoS를 제공하지 않는다. 즉, 현재의 대부분의 인터넷 응용들은 어느 정도의 패킷 전송 지연이나 패킷 손실에 영향을 받지 않으므로 이와 같은 best-effort 서비스 모델만으로 인터넷 응용들을 지원할 수 있었다. 그러나 최근 들어 이와 같은 best-effort 모델로는 지원이 어려운 새로운 멀티미디어 응용들이 등장하고 있다. 음성이나 영상과 같은 이런 새로운 응용들은 종단간 지연과 지연 변이에 매우 엄격한 상한 값을 요구한다. 이처럼 새로이 등장하고 있는 멀티미디어 응용들은 지연과 지연 변이 및 패킷 손실에 매우 민감하여 현재의 인터넷 모델로는 서비스 품질 보장을 위한 노력인 Internet Engineering Task Force(IETF)를 중심으로 활발하게 진행 중에 있다. 인터넷에서 QoS 제공을 위한 IETF의 대표적인 표준은 크게 두 가지로 Integrated Service(IntServ)와 Differentiated Service(DiffServ)로 나눌 수 있다. DiffServ 모델이 IntServ 모델과 비교하여 다음과 같은 특징을 가지고 있다. (홍석원, 2000)

첫째로, DiffServ 모델은 하나의 IP 패킷 흐름별로 서로 다른 QoS를 제공한다는 개념에서 벗어나, 여러 흐름의 집합을 단위로 하여 각 집합별로 패킷 전달을 차별화한다.

둘째, IntServ 모델에서는 서비스에 따른 QoS를 보장하기 위해서 망의 모든 라우터는 패킷 헤더 정보에 의해서 패킷 분류를 수행한다. 이와 같은 패킷 분류는 라우터에 있어서 고도의 처리 능력을 요구한다. 반면에 DiffServ에서는 패킷 분류와 같은 트래픽 조절 기능들을 모두 망의 가장자리에서만 일어나게 하고 망의 내부에서는 아주 간단한 패킷 전달 기능만이 수행되도록 하였다. 따라서 망의 경계 라우터는 여러 흐름이 집합된 서비스에 따라서 패킷의 분류를 수행하고 이를

패킷에 표시(mark)한다. 그리고 망 내부의 라우터는 패킷에 표시된 정보에 따라서 단순히 패킷의 전달 기능만을 담당하게 된다.

셋째, IntServ 모델에서 각 라우터에서 자원 예약을 위하여 연결 수락 제어를 수행해야 하는 반면에, DiffServ 모델에서는 이 기능을 망의 경계 라우터에서만 수행하도록 한다. 또한 이러한 자원 예약 절차도 DiffServ에서는 반드시 동적으로 이루어질 필요는 없고 망의 사용자와의 서비스 수준 협약에 따라 고정적으로 이루어질 수도 있다. 사용자가 서비스 수준 협약을 준수하는지의 여부는 망의 경계 노드에서만 감시된다.

마지막으로, DiffServ 모델에서는 IntServ의 보장형 서비스와 같은 절대적인 서비스 요구 사항을 보장하지는 않는다. DiffServ의 기본적인 서비스 개념은 개별적인 응용 흐름(application flow)를 구별하지 않고 단지 여러 흐름을 묶은 집합체로서 서비스를 차별화할 뿐이다.

2. DiffServ 구성 요소

1) 기본구조

DiffServ 망의 기본 구조와 구성 요소는 Fig.1 과 같다. DiffServ 제공 능력을 갖는 DS망(혹은 DS 도메인)은 여러 ISP망으로 구성될 수 있다. ISP를 연결하는 링크 사이의 경계에 경계 라우터(edge router)가 존재하며, 또한 DS망과 Non-DS 망과 연결되는 위치에도 경계 라우터가 존재하게 된다. DiffServ 구조는 여러 ISP에 걸친 양종단간(end-to-end) 서비스(Inter-domain Service)와 하나의 ISP 망에서 시작되고 끝나는 Intra-domain 서비스의 두 형태를 지원한다. 따라서 일반 개인 뿐 아니라 ISP 망 자체가 DS 망의 사용자가 될 수 있다. DiffServ의 망 구조는 다음과 같은 세 가지 요소 기능으로 구성된다.

◎ 서비스 수준 협약 (Service Level Agreement)
◎ 트래픽 조절 기능 (Traffic Conditioning)
◎ DS 바이트와 패킷 전달 기능 (Per-Hop Behavior)

DS 망의 사용자는 먼저 DS 망의 관리자와 서비스 사용을 위한 협약이 이루어진다. 이것은 서비스 수준 협약(SLA)이라는 쌍방간에 합의된 내용이어야 한다. DS 망의 사용자는 이러한 SLA에 의해서 DS 망을 통해 전달하고자 하는 패킷 흐름의 집합체를 정의하게 된다.

DS 망 경계의 입구 라우터는 이와 같은 정의된 패킷 흐름의 집합체에 트래픽 분류와 조절(conditioning) 기능을 수행한다. 패킷 조절 기능에는 트래픽 분류에 따른 패킷의 표시(mark), 흐름의 측정(meter), 쉐이핑(shaping), 그리고 감시 기능(policing)을 포함한다.

그리고 DS 망의 내부 라우터에서는 이와 같이 경계 라우터에 의해서 표시된 코드에 의해서 단순히 패킷을 전달하게 된다. 이러한 내부 라우터에서의 패킷 전달 기능을 DiffServ에서는 Per-Hop Behavior(PHB)라는 용어를 사용하고 있다.

2) 서비스 수준 협약과 대역폭 브로커(Bandwidth Broker)

사용자는 DS망의 사용을 신청할 때 자신의 트래픽과 원하는 서비스에 대한 사항을 DS 망의 관리자에게 알려준다. DS 망의 관리자는 이러한 사용자의 트래픽을 수용할 것인지의 여부를 결정한다. 이때 연결 수락 여부의 결정은 현재의 망의 상황과 사용자의 요구에 따라 정적으로 이루어질 수도 있으며, 동적으로 이루어질 수도 있다. 정적으로 이루어질 경우 상호 협약된 내용에 따라서 휴먼 관리자의 수작업에 의해서 일정 시간 동안 미리 정해진 경로와 대역을 할당하게 된다. 만약 DiffServ에 의한 서비스가 이루어질 경우 초기에는 이러한 정적인 방식으로 사용될 것이다.

하지만 이러한 서비스 협약이 동적으로 이루어질 경우 망의 부하나 가격 정책의 변화에 따라서 서비스 할당 내용이 달라질 수 있다. 이를 위해서는 RSVP와 같은 동적인 신호 프로토콜이 필요하게 된다. 그리고 DS 망의 자원을 관리하고 감시할 수 있는 노드가 필요한데 DiffServ에서는 이것을 대역폭 브로커(bandwidth broker)라고 부르고 있다. (Y. Bernet, 1998)

망의 경계 라우터는 이러한 신호 프로토콜에 의한 요청을 받았을 경우 연결 수

락 여부 및 감시 기능에 대해서 대역폭 브로커에게 문의를 하게 되며 이후의 모든 트래픽 조절 기능에 대한 사항을 대역폭 브로커를 통해 전달 받게 된다. 각 DS 망에는 대역폭 브로커가 요구되며 각 대역폭 브로커는 서로 망 자원에 관련된 정보를 주고 받을 필요가 있다.

이러한 대역폭 브로커의 존재와 상호 관련에 대해서 Fig. 1에 나와 있다.

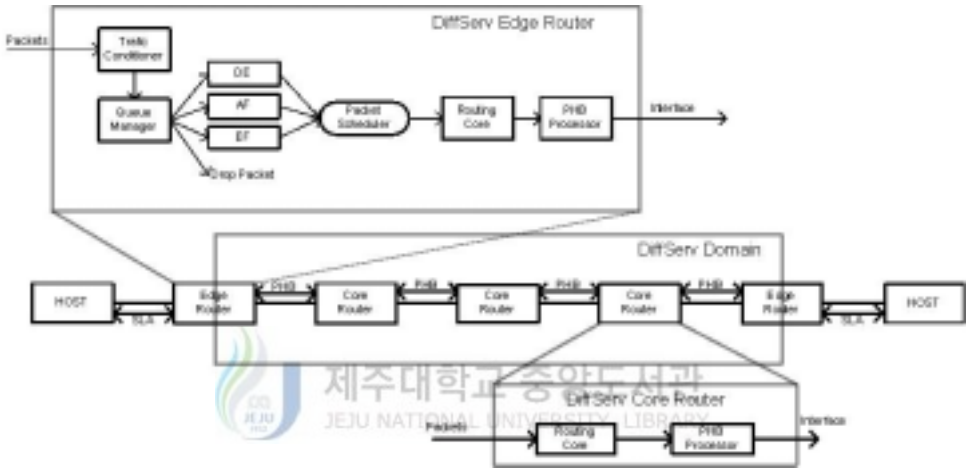


Fig. 1 Differentiated Services Architecture

3) 트래픽 조절(Traffic Conditioning)

사용자의 서비스 계약에 의해서 DS망의 경계 라우터는 DS 망의 내부 라우터에서 패킷을 전달하는 방식을 결정하기 위해서 패킷에 표시(mark)를 하게 된다. DiffServ에서는 IPv4의 경우 Type of Service(TOS) 바이트의 6비트를 사용하여 표시를 하도록 정하고 있다. 이렇게 표시되는 TOS 바이트를 DiffServ에서는 DS 바이트라고 부르고 있다.

DS 망의 경계 라우터는 DS 망 내에서 패킷을 어떻게 전달할 것인가를 결정하기 위해서 패킷을 분류하고 표시하는 기능을 수행한다. DiffServ 용어로는 이러한 기능을 트래픽 조절이라는 용어로 표현하고 있다. 트래픽 조절은 몇 가지 요소 기능으로 구성되는데 트래픽 분류, 측정(meter), 표시(mark), 쉐이핑(shaping), 그리고 폐기(drop) 기능이 여기에 속한다 (K. Nichols, 1999)

트래픽의 분류는 다음의 두 가지 방법이 존재한다. 즉 DS 바이트 외에 소스(source) 및 목적지 주소 필드 등 IP 헤더내의 필드에 근거하는 MF (Multi-Field) 분류와 단지 DS 바이트만을 사용하여 분류하는 BA (Behavior Aggregate) 분류로 나뉜다.

트래픽 측정기(meter)는 SLA에서 약속한 트래픽 프로필을 기준으로 패킷 흐름을 측정하여 프로필 준수 또는 위반과 같은 결과를 다음의 요소 기능에 전달한다. DS 망 내부에서 동일한 패킷 전달 방식으로 전달되는 패킷들은 동일한 DS 바이트를 갖는다. 동일한 DS 바이트를 갖고 DS망으로 들어오는 패킷들의 집합을 Behavior Aggregate(BA)라고 한다.

트래픽 표시기(marker)는 특정 패킷 전달 방식에 해당하는 코드를 DS 바이트에 기록하여 패킷을 특정 BA에 속하도록 한다.

트래픽 쉐이퍼(shaper)는 BA의 트래픽 패턴을 미리 약속한 트래픽 프로필에 준수하도록 조치를 취한다. 트래픽 폐기(Dropper)는 약정된 트래픽 프로필(예, 패킷 도착율이나 버스트 길이)에 어긋나는 패킷에 대해서 패킷 폐기와 같은 적절한 조치를 취한다. 이들 구성 요소의 상관 관계가 Fig. 2에 예시되어 있다.

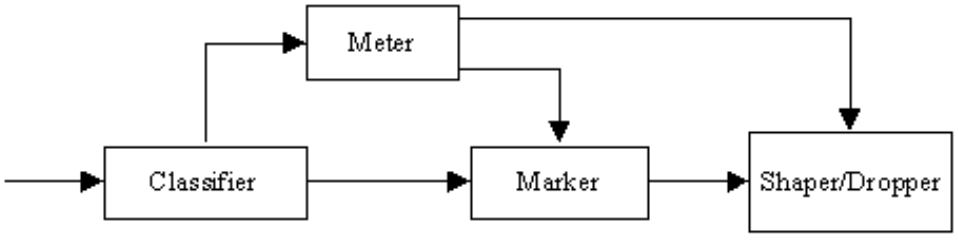
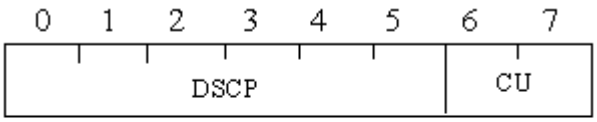


Fig. 2 Traffic Conditioner

4) DiffServ 모델에서의 패킷 전달 방식 (Per-Hop Behavior)

(1) DS 바이트와 PHB

DS 망의 경계 라우터는 IP 패킷이 중간 경유 라우터에서 어떠한 방식으로 전달될지를 구분하여 이것을 IPv4의 TOS 필드 혹은 IPv6의 Class Field 필드에 표시하게 된다. DiffServ에서는 이 필드를 DS 바이트, 그리고 DS 바이트에 표시되는 값을 코드 포인트(codepoint)라고 부른다. 현재 RFC 2474에서 정의하고 있는 DS 바이트의 형식은 Fig.3과 같다. (K. Nichols, 1998)



DSCP : DS Codepoint

CU : Currently Unused

Fig. 3 DS Byte Format

이 그림에서 보는 바와 같이 6 비트가 DS 코드 포인트(DSCP)로 할당되었다. 이 코드 값은 패킷이 경유하는 라우터에서 패킷이 전달되는 순서(즉 패킷 스케쥴링)와 버퍼 할당과 같은 패킷 전달 방식을 결정하게 된다. DiffServ에서는 이러한 망 내부 라우터에서의 패킷 전달 방식을 PHB라고 부르고 있다. 표준화의 초기 논의 과정에서 이 6 비트 중 한 개의 비트를 IN 비트로 따로 규정하여, 패킷이 약속된 서비스 프로필에 맞는지(In-profile)와 맞지 않는지(Out-of-profile)를 나타내도록 사용하려고 했다가 6 비트의 DSCP 내에 묵시적으로 규정하기로 하였다. CU 비트는 앞으로 명시적 체증 표시(Explicit Congestion Notification)와 같은 용도를 위해 예비한 필드이다. DS 바이트는 패킷이 다른 도메인에 전달되면서 재 기록될 수도 있다.

DS 망 내부의 라우터는 패킷 전달 메커니즘으로 다양한 버퍼 관리 및 패킷 스케쥴링 기능을 제공한다. ISP 관리자는 이들을 잘 선택 구성하여, 특정의 몇 가

지 패킷 전달방식을 제공하게 된다. 이와 같이 PHB는 라우터에서 패킷이 받게 되는 공통된 전달 방식을 규정한 것이다. PHB는 라우터의 내부 구현 메커니즘은 어떤 구현 기술을 사용하던지 상관하지 않으며 외부에 드러난 패킷 흐름의 입출력 동작만을 규정하고 있다.

(2) Default(DE) PHB와 Class Selector PHB

DE PHB는 현재 인터넷 라우터에서 널리 사용되고 있는 패킷 전달 방식인 베스트 에포트 전달 방식을 정의하고 있다. 이 방식에서 패킷은 입력된 순서대로 출력되고, 손실이 일어날 수도 있다. 지연은 가능한 최소화되고 대역폭은 가능한 많이 이용된다. DE PHB는 Diff-Serv를 지원하지 않는 사용자를 허용하기 위함이다. DE PHB에 해당하는 DS값은 '000000'이다.

Class Selector PHB는 현재 사용하고 있는 IP Precedence 필드와의 호환성을 위해 정의되었다. 현재 IP 헤더의 TOS 필드는 3비트의 IP precedence 비트와 4비트의 Type of Service 필드로 구성되어있다. (P. Almquist, 1992)

현재 인터넷에서 IP precedence 값이 보편적으로 사용되고 있지 않지만 일부 라우터와 ISP 망에서 부분적으로 이용되고 있기 때문에 DS 망에서도 이미 사용중인 값과의 호환성을 그대로 유지하기 위해서 Class Selector PHB를 정의하였다. 이 PHB에 해당되는 DS 값은 'xxx000'(x는 1이거나 0)이다.

(3) Expedited Forwarding(EF) PHB

EF PHB는 라우팅 정보 갱신과 같은 망 제어 트래픽과 같이 우선순위가 가장 높은 전달 방식을 의미한다. (V. Jacobson, 1998)

EF PHB방식으로 패킷을 전달할 경우 라우터에서는 이 그룹의 패킷의 출력률(departure rate)을 도착률(arrival rate) 보다 같거나 크게 설정하여 버퍼에서의 지연이 가능한 없도록 한다. EF PHB의 DS값은 '101110'이다.

EF PHB를 이용하는 사용자는 그 만큼의 대가를 지불하면서 빠르고 보장된 서비스를 제공받게 되는데 마치 비싼 값을 지불하면서 비행기 1등석을 사용하는 승객과 같다고 할 수 있다. ISP망 사업자는 EF PHB를 토대로 프리미엄 서비스라고도 불리는 가상 전용선(Virtual Leased Line) 서비스를 실현할 수 있다.

위에서 정의된 DS 코드포인트 값을 Table.1은 보여주고 있다. DE PHB, Class Selector PHB, 그리고 EF PHB의 코드포인트는 DS 바이트의 총 6비트에서 8개를 차지한다.

Table. 1 PHB and codepoint that is proposed by standard

PHB	Codepoint
Default	000000
Class Selector	xxx000
Expedited Forwarding	101110

(4) Assured Forwarding(AF) PHB

사용자에 따라서 인터넷을 통해 IP 패킷을 전달할 때 어느 정도의 보장을 요구할 수가 있다. 어느 회사는 인터넷을 사용할 때 회사 내부의 각 사이트로부터의 합쳐진 트래픽이 망 사업자와 약정한 트래픽 프로필을 초과하지 않는 이상 트래픽은 높은 확률을 가지고 전달된다는 보증을 원한다. 한편 때로는 약정된 트래픽 프로필을 벗어나는 트래픽의 발생이 허용되는 것을 원할 수도 있다. 이러한 경우, 프로필을 벗어난 과잉 트래픽은 프로필을 준수한 트래픽 보다는 낮은 수준의 패킷 전달 서비스를 받게 됨을 망 사업자와 사용자가 미리 이해하고 있다는 것을 전제로 한다. AF PHB는 이러한 요구를 만족시키기 위해 제안된 PHB 안이다. (J. Heinanen, 1999)

AF PHB에서 DS망 사업자는 사용자에게 받은 패킷들을 여러 가지의 종류로 구분된 순서에 의해서 패킷 전달을 한다. AF PHB에서 패킷 전달 보장 순서는 패킷 전달의 순서를 결정하는 클래스와 체증(congestion) 발생시 패킷을 폐기하는 순서에 의해서 결정된다. 클래스는 이 그룹의 패킷 전달을 위해서 할당된 전달 자원에 의하며, 폐기 우선순위(drop procedence)는 이 그룹의 패킷에 할당된 버퍼의 크기에 따라 결정된다.

현재 AF PHB은 4개의 클래스와 3개의 폐기 우선순위로 구분하고 있다.

Table. 2는 AF PHB에서 각 클래스와 폐기 우선순위에 따라 구분하여 할당되는 DS 코드포인트들을 나타내고 있다. AF PHB는 각 클래스 안에 긴 기간의 체증은 최소화하는 반면 폭주(burst)를 일으키는 짧은 기간의 체증은 허용하도록 구현하려고 한다. 이와 같은 요구를 충족하기 위해서 라우터에서는 버퍼 관리 알고리즘으로 RED(Random Early Detection)의 사용을 권고하고 있다.

AF PHB를 토대로 하여 제안된 서비스의 예로 흔히 인용되는 것이 올림픽 서비스이다. 이 경우 금메달 서비스는 은메달 서비스 보다, 은메달 서비스는 동메달 서비스 보다 항상 더 나은 수준의 패킷 전달 서비스를 받도록 패킷들은 세 개의 클래스로 할당된다. 각 클래스 안에 있는 패킷들은 다시 상중하의 패킷 폐기 우선순위를 할당받는다.

Table 2. AF PHB and Codepoint

Drop Procedence	Class 1	Class 2	Class 3	Class 4
Low	001010	010010	011010	100010
Medium	001100	010100	011100	100100
High	001110	010110	011110	100110

3. AF PHB에서의 버퍼관리 메커니즘

1) RED(Random Early Detection)

네트워크 상에 혼잡이 갑자기 생기면 라우터의 버퍼가 차버리게 되어 라우터가 패킷들을 버리기 시작한다. TCP 트래픽의 경우에는 slow start 단계로 들어간다는 신호이며 이 단계에서 네트워크상의 부하를 줄여 혼잡을 해소하게 된다. 이러한 상황일 때 두 가지 어려움이 있다. 첫째 폐기된 패킷을 재전송하게 되므로 네트워크 상에 부하를 증가시키게 되어 TCP 흐름상에 심각한 지연을 일으키게 한다. 더욱더 심각한 것은 전역 동기화(global synchronization)라는 현상으로 트래픽이 폭주하면 큐들이 차버려 많은 패킷들을 버리게 된다. 이러한 결과로 많은 TCP 커넥션들이 영향을 받게 되어 slow start에 들어가게 된다.

이렇게 되면 네트워크 트래픽이 굉장히 줄어들어 당분간 네트워크가 엉뚱하게 활용이 덜 되는 상황이 되어 버린다. 거의 동시에 많은 TCP 커넥션들이 slow start에 들어갔기 때문에 거의 동시에 slow start에서 빠져나오게 되므로 또 다시 폭주가 생겨 slow start에 들어가고 네트워크가 활용이 덜 되고 하는 상황이 계속되게 된다.

한 방법은 각 라우터에서 패킷을 버리는 확률을 줄이기 위하여 더 큰 버퍼들을 사용하는 것인데 이는 두 가지 이유 때문에 좋지 않다. 첫째 이렇게 큰 버퍼들이 차게 되면 모든 커넥션들이 겪는 지연이 아주 커지게 된다. 만약 이 트래픽이 자기상사형이면 더 나빠지기 쉬우며 기본적으로 버퍼를 충분히 크게 만들 수가 없다. 큰 폭주가 이어서 생기게 되므로 혼잡이 계속되고 버퍼 요구가 더 많아지게 된다. 더 나은 해결책은 혼잡의 시작을 예상하여 한 TCP 커넥션에 slow start할 시기를 알려주고 필요하면 다른 커넥션의 속도를 줄이기 전에 slow start한 결과를 측정하는 것이다. 혼잡이 시작될 때 이러한 식으로 하면 트래픽 부하를 서서히 줄이는 브레이크 역할을 하게 되어 TCP 커넥션들에는 최소한의 영향만 주며 전역동기화가 생기지 않게 된다. RED는 이러한 해결책을 제공하고 있다. (S. Floyd, 1993)

RED의 설계 목표로 다음과 같은 것들이 있다.

◎ 혼잡 회피 (congestion avoidance)

RED는 혼잡에 대처하는 것이 아니라 혼잡을 피할 수 있도록 설계되어 있다. 따라서 RED는 지연이 작고 스루풋을 높게 네트워크를 유지하기 위하여 혼잡의 시작을 찾아야 한다.

◎ 전역동기화 회피 (global synchronization avoidance)

혼잡의 시작을 알게 되면 라우터가 어느 커넥션(들)에 백오프하도록 알려줄 것인가를 결정해야 한다. 현재의 구현에서는 이러한 통보가 암묵적으로 이루어지는데 패킷을 폐기하기만 한다. 혼잡을 미리 발견하고 필요한 만큼의 커넥션들에만 백오프를 알려주기 때문에 전역동기화를 피할 수 있게 된다.

◎ 폭주 트래픽만 폐기하지 않도록 함

혼잡의 시작은 하나나 소수의 Source들로부터 폭주 트래픽이 도착할 때 일어나기 쉽다. 이러한 폭주는 라우터가 이미 지원하고 있는 부하에 더욱더 부담을 주게 된다. 도착하는 패킷들만 선택하여 버리게 하면 평균적으로 같은 트래픽으로 Source들을 조정하고 있는 것에 비하여 폐기 알고리즘은 폭주 Source들에만 편중되게 된다.

◎ 평균 큐 길이의 한계

RED가 평균 큐 크기를 제어할 수 있어야 한다. 즉 평균지연을 제어할 수 있어야 한다.

일반적으로 패킷이 도착할 때마다 RED 알고리즘은 다음과 같은 절차를 수행한다.

Calculate the average queue size avg
  if avg < TH_{min}
    queue packet
  else if TH_{min} ≤ avg < TH_{max}
    calculate probability P_a
    with probability P_a
      discard packet
    else with probability 1-P_a
      queue packet
  else if avg ≥ TH_{max}
    discard packet

알고리즘은 FIFO 출력 큐에 새로운 패킷이 도착할 때마다 두 기능을 수행하는데 첫 번째 절차는 평균 큐길이 avg를 계산하며 이 평균 큐길이를 두 한계치와 비교한다. avg가 하한치 TH_{min}보다 작으면 혼잡이 최소이거나 없는 것으로 가정하여 이 패킷을 큐에 넣는다. avg가 상한치 TH_{max}보다 크면 혼잡이 심각하다고 가정하여 이 패킷을 버린다. avg의 정확한 값에 따라 avg를 상한선에 이르게 증가시키는 확률 P_a를 계산한다. 큐가 이 영역에 있으면 이 패킷을 확률 P_a로 버리며 확률(1-Pa)로 큐에 들어가게 한다.

기본적으로 알고리즘의 첫 부분(큐 크기 계산)은 허용할 수 있는 폭주 정도를 결정하며 알고리즘의 두 번째 부분(패킷 폐기 결정)은 현 수준의 혼잡이 있을 때 폐기되는 패킷들의 빈도를 결정한다.

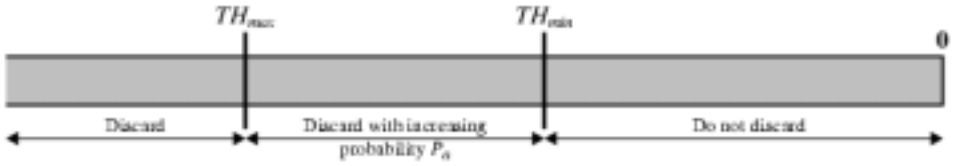


Fig. 4 RED Buffer

임시확률값 P_b는 avg = TH_{min}에서의 0부터 시작하여 avg = TH_{max}에서의 최대치 P_{max}까지 선형으로 증가하는 값이다.

$$P_b = \frac{P_{\max}(avg - TH_{\min})}{TH_{\max} - TH_{\min}}$$

P_a는 폭주 원천이 불이익을 당하지 않게 비교적 공평하게 간격을 두고 폐기를 결정한다. P_a가 P_b에 비해 전역 동기화가 일어날 가능성이 적다.

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

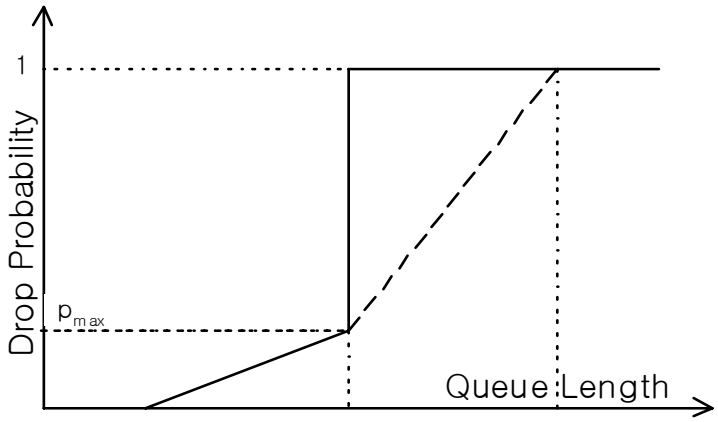


Fig. 5 Drop Probability

2) RIO (RED with In/Out)

RIO 메커니즘은 RED를 기반으로 라우터에서 혼잡 상황 동안 차별화된 패킷 드롭핑을 수행하도록 하는 방식이다. RIO에서 종단 사용자를 위한 트래픽 프로필은 네트워크 가장자리에서 유지되고, 사용자 트래픽이 계약을 초과할 때 패킷은 *out-profile*로 마킹된다. 그렇지 않으면 *in-profile*로 마킹된다. RIO 메커니즘은 단일 큐에서 이용되고 모든 사용자 패킷은 동일한 큐로부터 전달되고 서비스된다. (David D. Clark, 1998)

RIO 메커니즘은 하나의 FIFO 큐를 사용하고, 네트워크 가장자리의 리마킹 Policer에 의존한다. 그 결과로써 최선형 라우팅 디바이스의 향상에 근거하여 차별화된 패킷 전달을 가능하게 한다.

RIO 알고리즘은 게이트웨이에 유입된 패킷이 In 인지 Out 인지 검사하여 *in-profile*인 경우 In 패킷의 평균 큐 크기(Q_{avg})를 계산하여 2종류의 매개변수 즉, TH_{min_in}, TH_{max_in}, P_{max_in}과 TH_{min_out}, TH_{max_out}, P_{max_out}으로서 다섯 가지의 Congestion 상태를 나타낼 수가 있다.

RIO 알고리즘은 다음과 같은 과정을 수행한다.

For each packet arrival
    if it is an *In* packet
        Calculate the average *In* queue size *avg_in*;
    Calculate the average queue size *avg_total*;

If it is an In packet
    if $min\_in < avg\_in < max\_in$
        calculate probability P_{in};
        with probability P_{in}, drop this packet;
    else if $max\_in < avg\_in$
        drop this packet
If this is an *Out* packet
    if $min\_out < avg\_total < max\_out$
        calculate probability P_{out};
        with probability P_{out}, drop this packet;
    else if $max\_out < avg\_total$
        drop this packet

Ⅲ. Extended RED Algorithm

1. RED 매개변수의 동작특성

RED는 버퍼에서의 평균 큐 길이(Q_{avg})를 이용하여 체증 제어를 수행한다. Q_{avg}가 최소 큐 한계값(TH_{min}) 보다 작을 때는 모든 패킷이 정상적으로 처리되며 TH_{min}과 최대 큐 한계값(TH_{max}) 사이에 있을 때는 확률 P_a에 의해서 패킷이 폐기된다. Q_{avg}가 TH_{max}보다 클 때는 최대 확률에 의해서 패킷이 폐기된다. RED에서 이러한 매개변수 값을 어떻게 정하느냐에 따라서 성능 차이가 심하게 된다. 따라서 RED 매개변수의 동작특성을 먼저 이해하여야 하며, 아래에서 각각의 매개변수에 대한 동작특성을 설명한다.

1) 큐 가중치(W_q)와 평균 큐 길이(Q_{avg})

평균 큐길이(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 현재 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 0.001 이상을 쓰도록 권하고 있고 일반적으로 0.002를 사용하고 있다. (S. Floyd, 1993)

2) 최대 확률(P_{max})

P_{max}의 값을 조절할 경우 가장 영향을 받는 부분은 Q_{avg}이다. 이 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 주어야 한다.

3) 최소 큐 한계값(TH_{min}), 최대 큐 한계값(TH_{max})

실제 망에 RED를 적용했을 때 TH_{min}과 TH_{max} 값을 설정하는 것은 상당히 중요한 일이지만 이것은 수학적으로나 모의실험만으로 정할 수 있는 매개변수는 아니다. 이 매개변수의 범위는 다음과 같다.

트래픽이 버스트한 특성을 가질수록 TH_{min}을 크게 주어 링크의 사용효율을 유지할 수 있도록 해준다. 그러나 정상적인 환경에서는 TH_{min}을 가급적 작게 주는 것이 유리하다. TH_{max}는 Drop Tail의 최대 버퍼 크기와 비슷하게 동작하므로 이 매개변수의 값이 클 경우는 큐 길이가 전체적으로 높게 유지되기 때문에 큐잉 지연시간(queueing delay time)이 길어진다. TH_{min}와 TH_{max}의 차이는 왕복시간(roundtrip-time) 동안에 계산되는 평균 큐 길이의 추정치보다 커야 RED 라우터는 효율적으로 동작할 수 있다. 대략 TH_{max}는 최소한 TH_{min}의 두배 이상으로 할 것을 권고하고 있다.

2. RED 매개변수 적용의 문제점

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

이 매개변수의 최적 값은 망의 트래픽 상황에 따라 달라지기 때문에 결국은 트래픽 상황의 변화에 따라 그 값을 변화할 수 있어야 할 것이다. RED의 매개변수를 고정시켜 놓고 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 RED 매개변수가 원래의 의도대로 동작하지 못하기 때문이다. 또한 이러한 상황에 맞도록 매개변수의 값을 정하게 되면 트래픽 부하가 적을 경우에는 throughput에 영향을 주어 성능이 나빠진다. (홍석원, 1999)

따라서, RED에서는 4개의 매개변수를 고정된 값으로 사용하기 때문에 지속적으로 트래픽이 버스트한 상황일 때는 RED가 제대로 기능을 발휘하지 못하는 상황

이 발생할 수 있으며 이를 보완하기 위해서는 매개변수의 값을 망의 상태에 따라 조절하여 적응적으로 반응할 수 있도록 하여야 한다.

RED의 고정된 매개변수들은 적정한 트래픽 부하일 경우에 잘 적용되도록 정해지므로 버스트한 트래픽 부하가 지속적으로 가해졌을 때도 문제가 발생하지만 낮은 경우에도 개선의 여지가 있다. 4개의 매개변수 중 TH_{min}의 값은 트래픽 부하가 적을 경우 링크 이용률을 높게 유지하기 위해 사용되어지며 높게 할수록 적은 트래픽 상황일 경우에 좋은 링크 이용률을 유지한다. 그러나 이러한 TH_{min}을 적정한 트래픽 부하일 때에도 동일하게 적용된다면 링크 이용률에는 향상이 없고 불필요한 평균 큐의 크기만 증가시키게 되므로 이 경우에는 TH_{min}은 작게 유지되어야 한다.

3. 제안 알고리즘

위에서 기술한 RED의 문제점들은 고정적인 매개변수들로 인해 망의 지속적인 트래픽 상태를 감지할 수 없다는 알고리즘의 특성 때문에 야기되는 문제이다. 최근의 연구에서는 이러한 RED의 문제점을 개선하기 위해서 ARED(Adaptive RED)가 제안되었다. ARED는 라우터가 처리하는 트래픽의 부하에 따른 큐 길이의 변화를 추적하여 최대 확률을 변화시킴으로써 링크의 효율을 높이는 방안이다. (W. Feng, 1997)

RED에서 최대 확률 외에 성능에 중대한 영향을 미치는 매개변수는 여러 가지가 있다. 이중에서 최소 큐 한계값의 경우 평균 큐 길이에 가중치를 줌으로써 패킷이 폐기되지 않고 정상적으로 처리되는 상황에서도 빠른 복귀가 늦어지며 이 때문에 링크의 이용이 비효율적이 된다. 이러한 문제는 정상적인 패킷 처리 상태일 경우 최소 큐 한계값을 높여줌으로써 성능의 개선을 이룰 수 있다.

따라서 본 논문에서는 트래픽의 부하에 따른 큐 길이의 변화를 통해 지속적인 트래픽 상태를 감지하는 방법을 제시하고 이를 이용하여 최소 큐 한계값(TH_{min})과 최대 확률(P_{max})을 트래픽 상태에 따라 적응적으로 반응하는 확장된 RED 알고리즘을 제안한다.

1) 트래픽 상태 감지 방법 제시

일반적으로 사용되는 W_q=0.002를 1/20 이하로 감소시켜 식 [1]처럼 가중평균(Q_{avg})을 계산하여 Q_L라 하고 망 트래픽 상태의 척도로 사용한다.

$$Q_L \leftarrow (1 - W_L) \times Q_L + W_L \times Q_{size} \qquad [1]$$

단, $W_L < W_q/20$, Q_L의 초기값 = 0

망 트래픽 상태의 척도로 사용하는 Q_L은 지수비중(exponentially weighted) 평균을 이용하여 계산한 평균 큐 길이이다. 비중 W_L은 실제 큐 크기의 변화에 따라 Q_L이 얼마나 빨리 변하는가를 결정한다. W_L이 작으면 작을수록 현재의 큐의 실제 길이가 Q_{avg}가 변하는데 미치는 영향이 작게 된다. 이런 경우에는 일시적인 트래픽의 증가로 인해 단시간 내에 큐의 길이가 증가되더라도 Q_{avg}가 증가하는데 미치는 영향이 작기 때문에 이를 허용할 수 있다. 하지만 W_L을 마냥 작게만 주는 것도 문제가 있다. Q_{avg}가 TH_{max}를 넘었을 경우가 그것이다. Q_{avg}가 TH_{max}를 넘었을 경우에는 도착하는 모든 패킷에 마크를 하기 때문에 이렇게 마크된 패킷에 대한 응답 패킷을 받은 송신 호스트는 모두 slow start 단계에 들어가기 때문에 전송 효율이 크게 떨어지게 된다. 즉 Tail-Drop 방식과 비슷한 상황이 되는 것이다. 그래서 Q_{avg}가 TH_{max}를 넘었을 때는 가능한 한 빨리 Q_{avg}가 TH_{max} 밑으로 떨어져야 하는데 실제 큐 길이는 매우 감소했다 하더라도 W_L이 너무 작기 때문에 이미 올라간 값이 떨어질 때에도 많은 시간이 필요하게 된다. W_L이 너무 클 경우에는 이와는 반대로 Q_{avg} 값이 실제 큐의 길이의 영향을 많이 받게 된다. Q_{avg}가 실제 큐 길이의 영향을 받아 기복이 크다는 것은 RED를 사용하는 의의를 줄이는 일이다. 결과적으로 Q_L은 실제 큐 크기의 변화보다 상당히 뒤에 일어나게 한다. 이렇게 작은 W_L을 사용하면 알고리즘이 짧은 폭주의 혼잡에 반응하지 않게 된다. 이 값이 TH_{min}×β 부근에 있을 경우를 정상적인 상태, 클 경우를 과부하 상태, 작을 경우를 낮은 부하 상태라 판단한다. 여기서 β는 정상상태의 기준을 결정하는 매개변수로 망의 특성에 따라 달라질 수 있다. 작은 큐잉 지연을 요구하면 작게 하고, 큐잉 지연보다 좋은 링크 이용률을 원한다면 큰 값을 선

택할 수 있다. 본 논문에서는 β = 0.5, β = 0.33으로 설정하여 모의실험을 수행하였다.

2) 최소 큐 한계값(TH_{min}) 결정

최소 큐 한계값은 낮은 망 부하 상태일 때는 높게, 높은 망 부하상태일 때는 낮게 설정할 수 있도록 식 [2] 와 같이 결정한다.

$$TH_{\min} = \alpha \times (TH_{\max} \times \beta - Q_L) + \gamma \qquad [2]$$

단, $0 < \beta < 1$

여기서 α와 γ 값은 최소 큐 한계값을 결정하는 매개변수이다. 반복적 실험 결과에 의해 TH_{min}값이 TH_{max}의 절반 이하 값을 가지도록 하는 조건을 만족하고 동시에 가장 좋은 성능을 보여줄 때의 경험적인 매개변수 값이다. 본 논문에서는 α = 5, γ = 5 로 설정하여 모의실험을 수행하였다.

3) 최대확률(P_{max}) 결정

망의 트래픽 상태에 따라 P_{max} 값을 자동적으로 식 [3]을 이용하여 결정한다. 망의 부하가 적을 때는 낮게 하고 많을 때는 높게 하여 다양한 망의 환경에 효율적으로 적용할 수 있도록 결정한다.

$$P_{ar} = \frac{Q_L}{TH_{\max} - Q_L \times (\beta - 1)} \qquad [3]$$

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

P_{inini} : $P_{\max}$의 초기 값으로 주어지는 값

본 논문에서는 δ = 5로 설정하여 모의실험을 하였다.

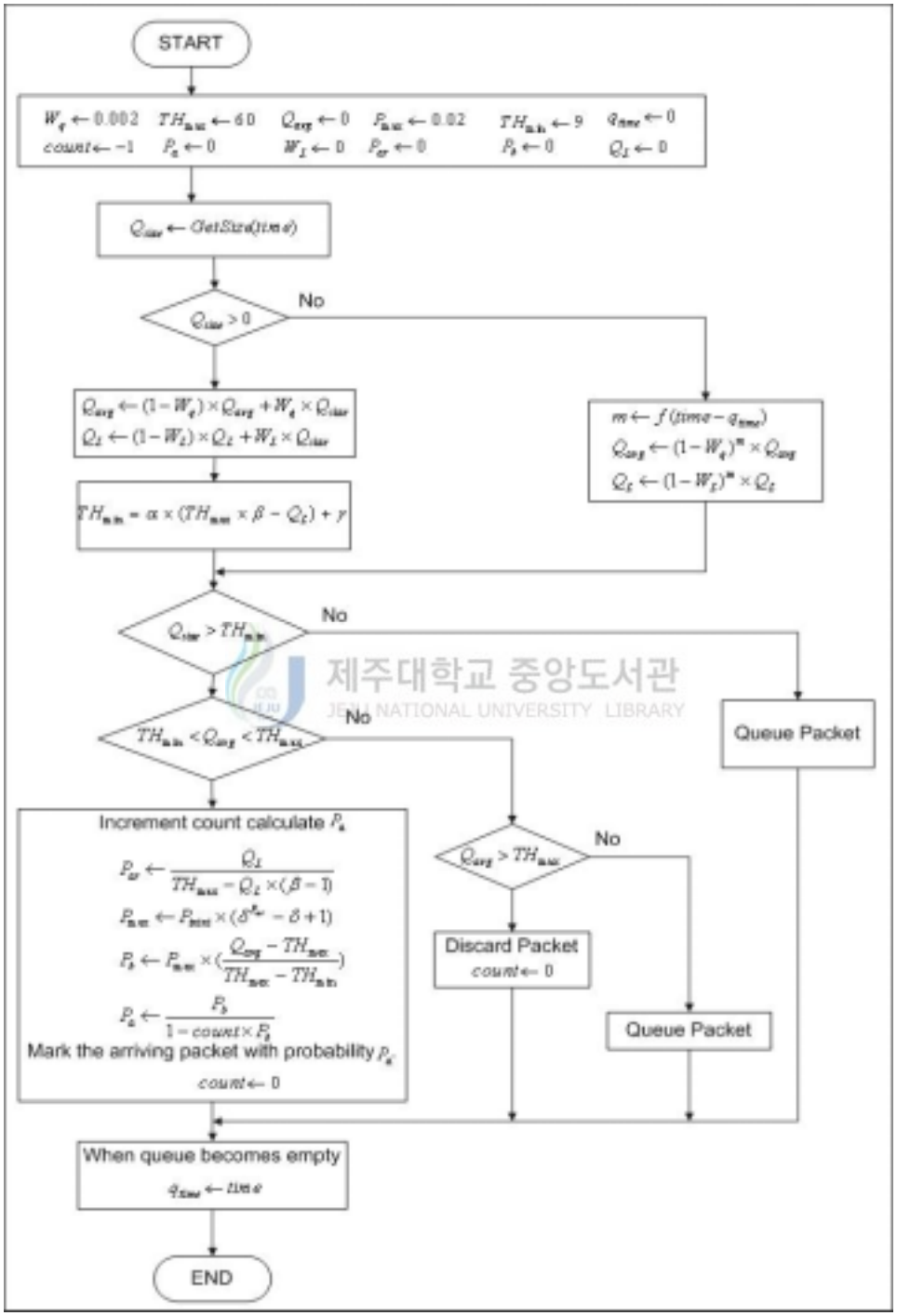


Fig. 6 Extended RED Algorithm

Extended RED 알고리즘의 동작과정을 간단히 살펴보면 먼저 패킷이 라우터에 도착하면 현재 큐 크기를 구한다. 현재 큐 크기가 0보다 크다면 평균 큐 크기와 망 트래픽 상태의 척도인 Q_L을 식 [1]을 이용하여 결정하고, 식 [2]를 이용하여 최소 큐 한계값을 결정한다. 이후는 기존 RED의 동작과정과 유사하다. 현재 평균 큐 크기를 최소 큐 한계값 및 최대 큐 한계값과 비교하여 최소 큐 한계값보다 작으면 패킷의 폐기는 없으며 정상적으로 처리된다. 평균 큐 크기가 최소 큐 한계값과 최대 큐 한계값 사이에 있을 경우 확률 Pa에 의해서 폐기가 되며 최대 큐 한계값보다 클 경우에는 식 [3]에 의해서 구해진 최대 확률을 이용하여 패킷의 폐기가 이루어진다.

Ⅳ. 모의실험 결과 및 성능분석

1. 모의실험 환경

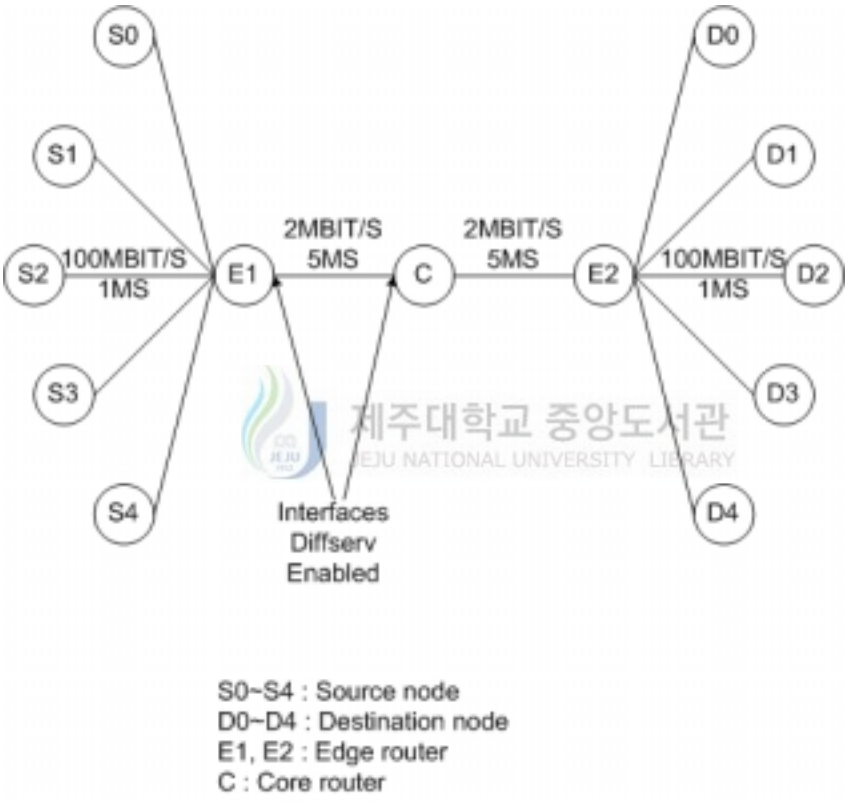


Fig. 7 Simulation Network Topology

모의실험을 위한 Network Topology는 Fig. 7과 같다. 송신측은 S0부터 S4까지 5개의 노드이며 Edge Node(E1) 와는 100Mbit/s 대역폭과 1ms 지연을 가지고 있다. E1에서 Core Router(C)까지는 2Mbit/s 대역폭, 5ms 지연을 가지며 이 구간에서 본 논문에서 제안한 ERED 알고리즘의 성능을 측정하였다. Traffic은

Default(DE) PHB, Expedited Forwarding(EF) PHB, Assured Forwarding(AF) PHB의 세 가지 패킷전달방식(PHB)으로 발생되고 전달된다. EF PHB는 우선순위가 가장 높은 전달 방식이며 Traffic Class는 Premium으로, AF PHB Group은 사용자의 요구를 만족시키는 보장 서비스로서 Olympic Service 중에 Gold Service를 Traffic Class로 할당한다. 이외에 DE PHB는 현재 인터넷 라우터에서 널리 사용되고 있는 Best-Effort 방식을 정의하며 DiffServ를 지원하지 않는 사용자를 허용하기 위함이다. DE PHB의 Traffic Class 역시 Best-Effort이다. Premium Service에는 300kbit/s, Gold Service에는 1Mbit/s, Best-Effort Service에는 700kbit/s 대역폭을 할당함으로서 이 구간의 전체 대역폭은 2Mbit/s 이다. 특히 AF PHB에는 RED와 ERED가 적용되며 트래픽 유형은 FTP와 Telnet 이다. 정리하면 Table. 3과 같다.

RED의 매개변수의 경우 W_q는 0.002, P_{max}는 0.02, TH_{max}는 30, TH_{min}는 5, Q_{size}는 60, 패킷 크기는 1500byte로 설정하였다. 제안 알고리즘인 ERED의 경우 Ⅲ장에서 설명한 바와 같이 W_q는 0.002, P_{max}는 0.02, W_L는 0.00004, α는 5, β는 0.5, γ는 5, δ는 5로 설정하였다.

모의실험은 두개의 시나리오로 구분하여 진행한다. 첫 번째 시나리오는 Network의 정상 상태를 의미하는 기준값 β를 0.5로 설정하여 수행하고, 두 번째 시나리오는 기준값 β를 0.33으로 설정하여 수행한다. 모의실험 결과는 링크 이용률, 패킷 손실률, 큐 사이즈 등으로 나타낼 수 있으며 기준값 β를 0.5와 0.33으로 했을 때 결과를 비교, 분석한다.

모의실험을 위해서 UC Berkeley에서 개발하고 현재 널리 사용되고 있는 네트워크 모의실험 도구인 NS-2를 사용하였다. 큐잉 방식은 WFQ(Weighted Fair Queueing)를 사용하였으며 AF PHB에 RED와 ERED를 적용하여 수행 결과를 비교, 분석하였다.

Table. 3 Network Interface

PHB	Traffic class	Bandwidth	Marking Rule	Dropper
EF	Premium	300 kbit/s	S0 → D0 (DSCP=46)	Drop out of profile
AF	Gold	1 Mbit/s	Telnet (DSCP=10) FTP (DSCP=12)	RED, ERED
Default	Best Effort	700 kbit/s	Everything else (DSCP=0)	Drop out of profile

2. 모의실험 결과 및 분석

Fig. 8은 기준값 β를 0.5로 설정했을 때의 AF PHB에서 RED와 ERED의 큐 길이를 보여준다. RED의 경우 큐 길이가 급격하게 변하고 있는데 비해서 제안 알고리즘 ERED는 최소 큐 한계값(TH_{min}) 근처에 머물며 좀더 나은 큐 관리를 보여주고 있다. ERED가 지속적인 부하를 감지, TH_{min}을 적응적으로 반응시키는 것을 보여주고 있다. Fig. 9는 기준값 β를 0.33으로 설정했을 때의 큐 길이 결과값이다. β를 0.33으로 설정했을 때의 큐 길이가 β를 0.5로 설정했을 때의 큐 길이보다 평균적으로 더 작다는 것을 볼 수 있다.

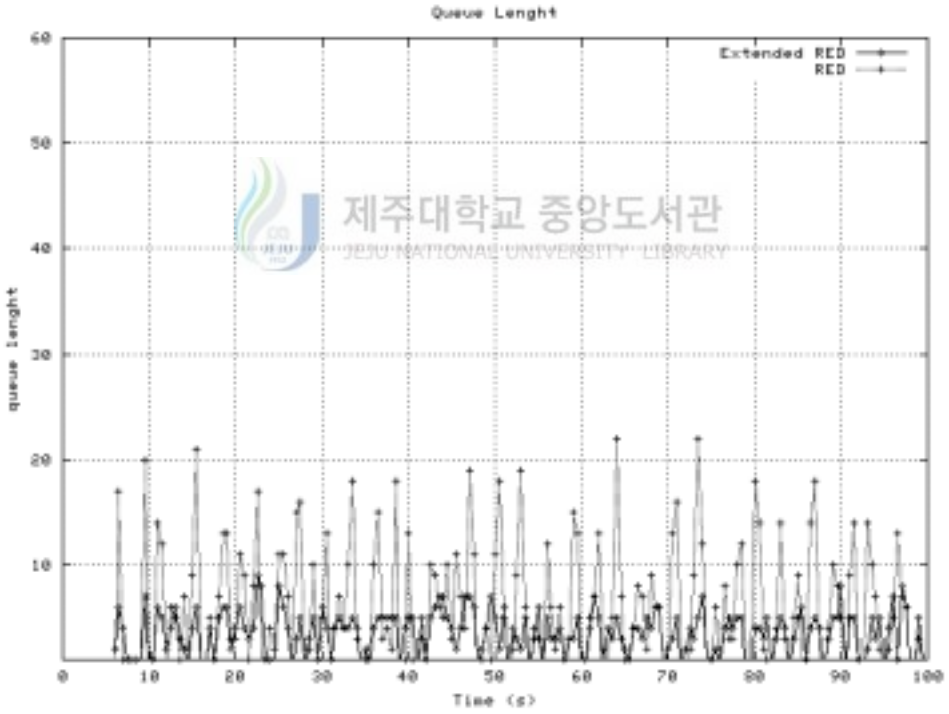


Fig. 8 Queue Length (β = 0.5)

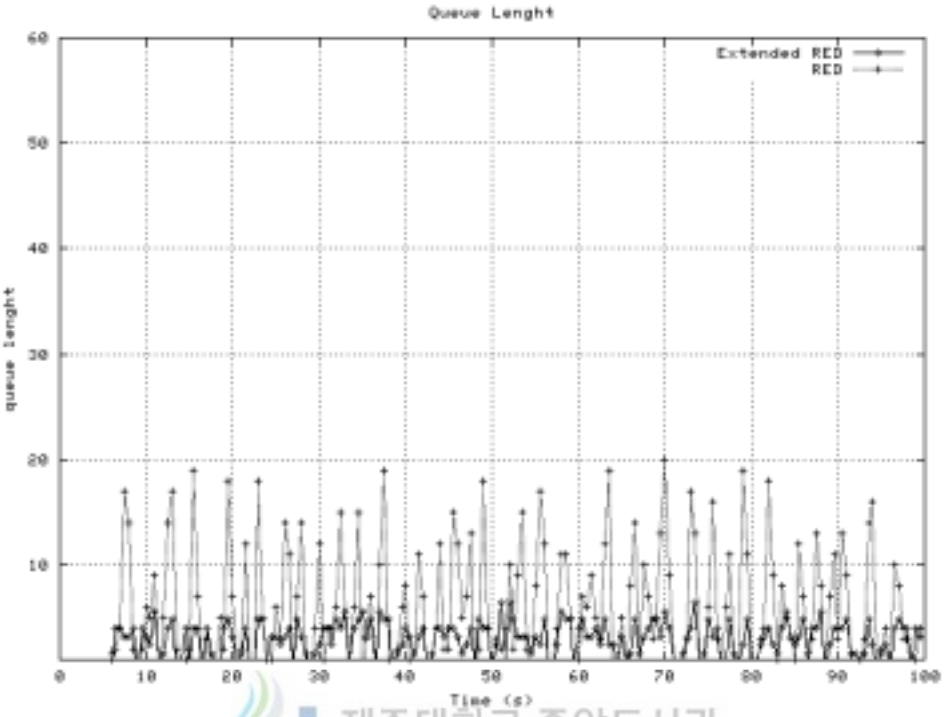


Fig. 9 Queue Length (β = 0.33)

Fig. 10은 기준값 β를 0.5로 설정했을 때의 AF PHB에서 RED와 ERED의 패킷 손실률을 보여 준다. ERED가 RED보다 패킷 손실률이 적은 것을 볼 수 있다. Fig. 11은 기준값 β를 0.33으로 설정했을 때의 패킷 손실률을 보여 주는데 기준값 β를 0.5로 설정했을 때보다 패킷 손실률이 더 높다는 것을 볼 수 있다. 이러한 결과는 기준값을 크게 설정함으로서 최대 한계값을 낮추는 것에서 기인한다. Table. 4의 평균값을 보면 기준값 β를 0.5로 설정한 ERED의 경우 RED에 비해 33% 정도의 패킷 손실률 감소를, 기준값 β를 0.33으로 설정한 ERED에 비해서는 21% 정도의 패킷 손실률 감소를 가져온다는 것을 알 수 있다.

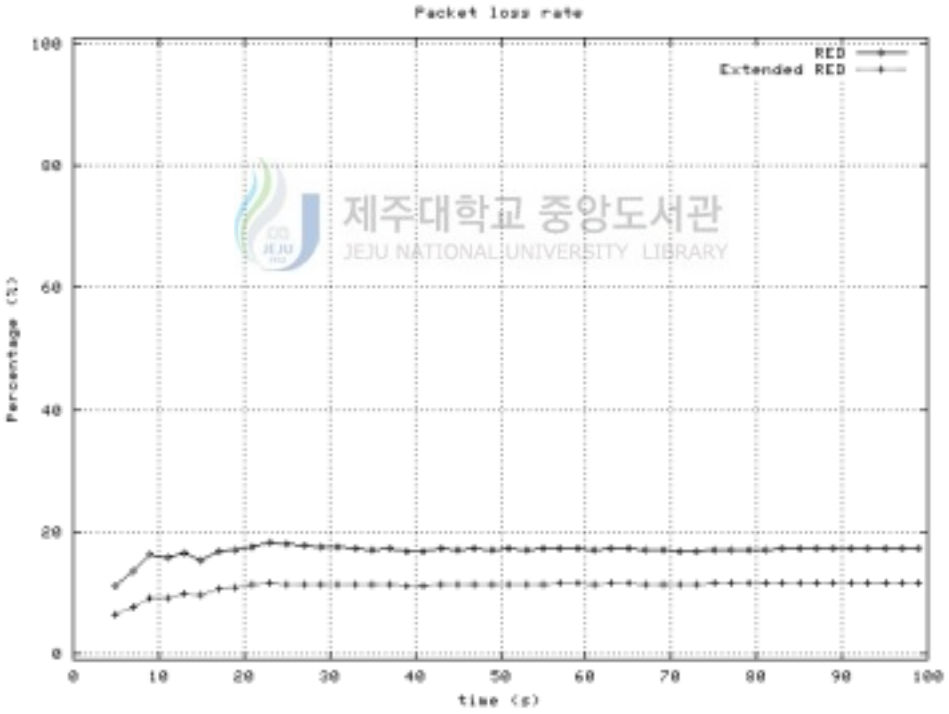


Fig. 10 Packet Loss Rate (β = 0.5)

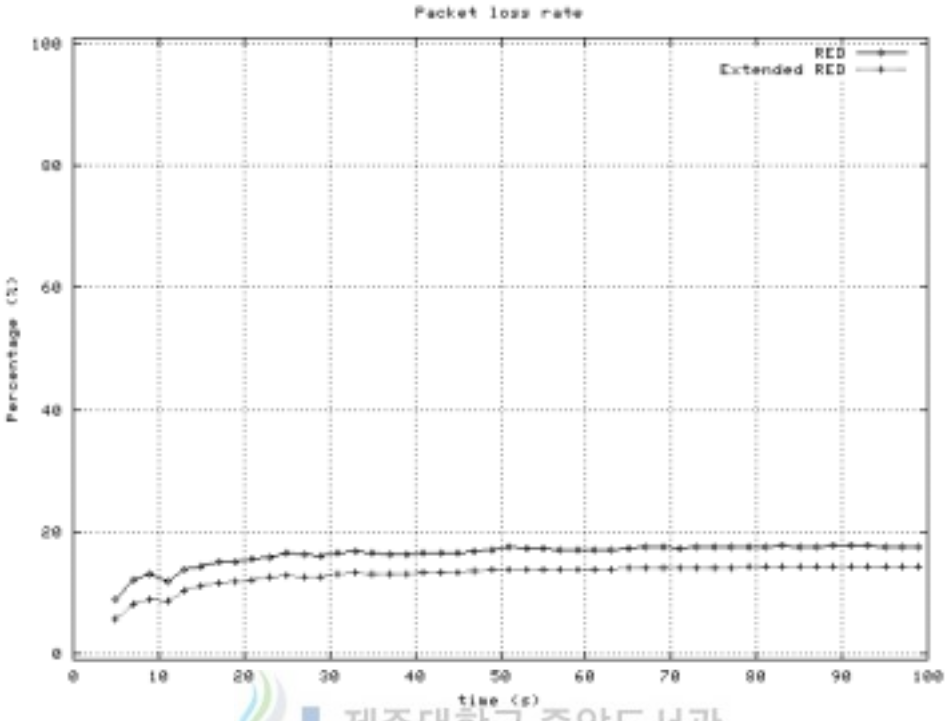


Fig. 11 Packet Loss Rate (β = 0.33)

Table. 4 Simulation result (Packet Loss Rate)

		19	39	59	79	99	Average
RED		15.3	16.5	17.2	17.1	17.1	16.64
ERED	β=0.5	10.1	11.1	11.6	11.5	11.6	11.18
	β=0.33	11.8	13.5	13.3	13.8	13.8	13.24

Fig. 12는 기준값 β를 0.5로 설정했을 때의 AF PHB에서 RED와 ERED의 링크 이용률을 보여 준다. ERED가 RED보다 링크 이용률이 높은 것을 볼 수 있다. Fig. 13은 기준값 β를 0.33으로 설정했을 때의 링크 이용률을 보여 주는데 기준값 β를 0.5로 설정했을 때보다 링크 이용률이 더 낮다는 것을 볼 수 있다. 이러한 결과는 패킷 손실률과 마찬가지로 기준값 β를 크게 설정함으로서 최대 한계값을 낮추는 것에서 기인한다. Table. 5의 평균값을 보면 기준값 β를 0.5로 설정한 ERED의 경우 RED에 비해 8.7% 정도의 링크 이용률 증가를, 기준값 β를 0.33으로 설정한 ERED에 비해서는 5.1% 정도의 링크 이용률 증가를 가져온다는 것을 알 수 있다.

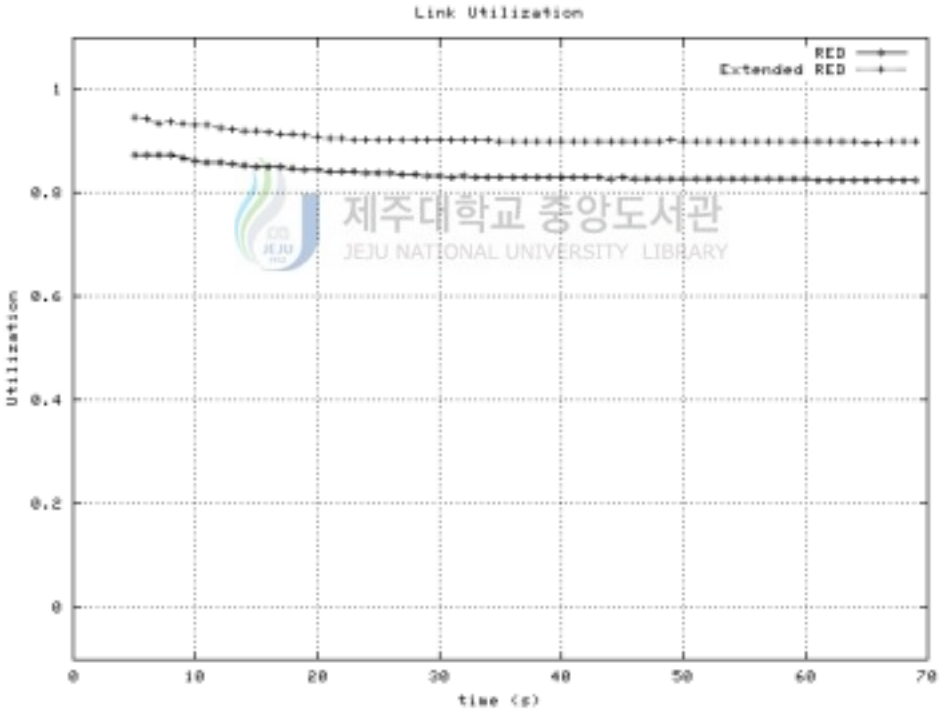


Fig. 12 Link Utilization (β = 0.5)

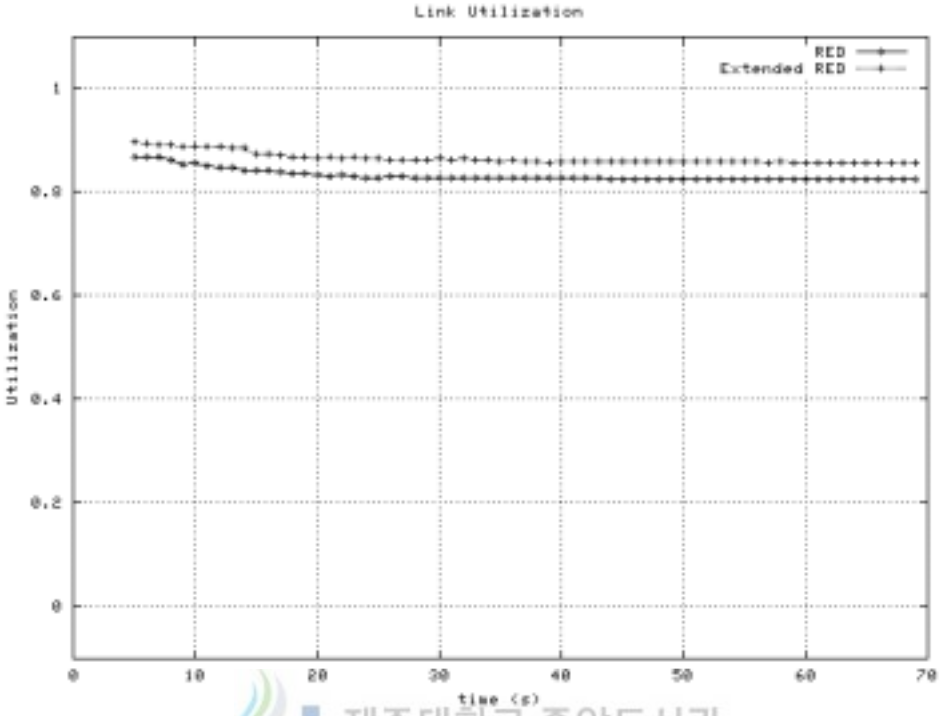


Fig. 13 Link Utilization (β = 0.33)

Table. 5 Simulation result (Link Utilization)

		20	40	60	80	100	Average
RED		0.84	0.82	0.82	0.81	0.81	0.820
ERED	β=0.5	0.91	0.89	0.89	0.88	0.89	0.892
	β=0.33	0.86	0.85	0.85	0.84	0.84	0.848

Fig. 14와 Fig. 15, Fig. 16은 RED와 ERED(β=0.33), ERED(β=0.5)를 적용했을 때 각각의 클래스 전송률을 보여주고 있다. Fig. 14를 보면 FTP aggregate의 경우 AF PHB에 할당된 1Mbit/s 대역폭에 훨씬 못 미치는 대역폭을 사용하고 있다. 반면에 Fig. 15의 FTP aggregate를 보면 기준값 β를 0.33으로 설정했을 때 RED보다는 좀더 최대 대역폭에 가까운 대역폭을 사용하고 있다. Fig. 16은 기준값 β를 0.5로 설정했을 때 AF PHB의 최대 대역폭인 1Mbit/s에 근접함을 볼 수 있다.

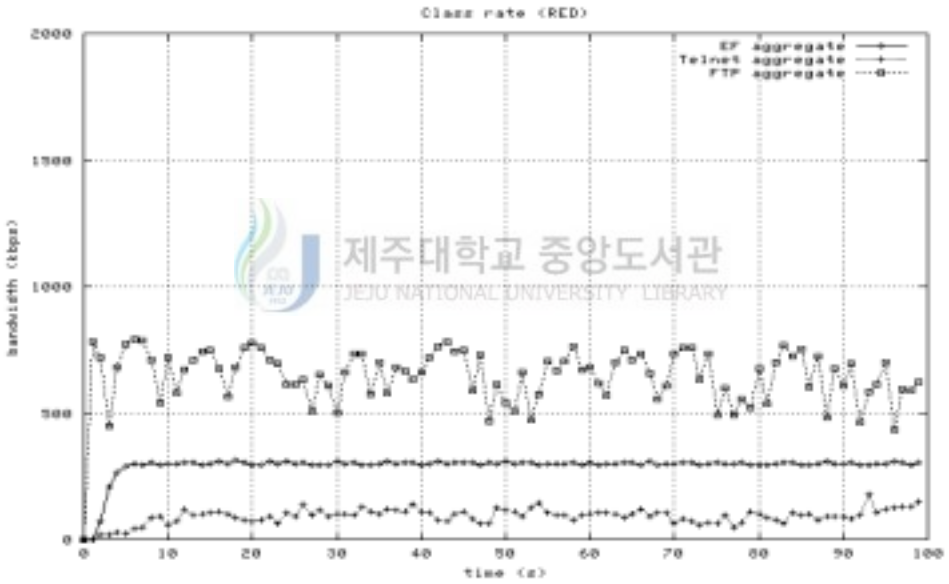


Fig. 14 Class Rate (RED)

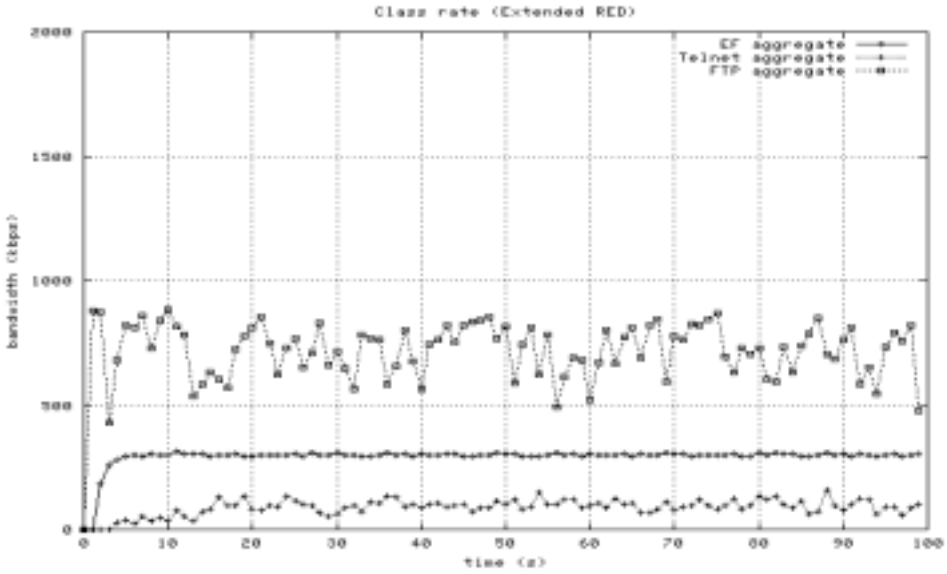


Fig. 15 Class Rate (Extended RED, β = 0.33)

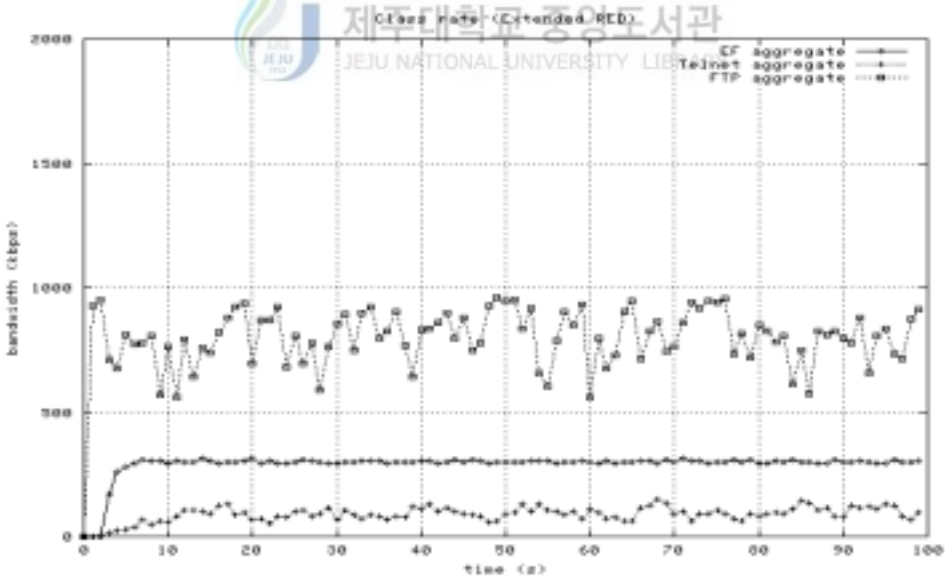


Fig. 16 Class Rate (Extended RED, β = 0.5)

위의 모의실험 결과에서 RED 매개변수의 주요 성능 지표인 패킷 손실률과 링크 이용률을 살펴보자. Fig. 11과 Table. 4, Fig. 13과 Table. 5를 보면 제안 알고리즘 ERED의 부하 상태 기준값 p_b를 0.33으로 설정했을 때 RED와 비교하여 패킷 손실률에서는 약 21%의 감소를, 링크 이용률에서는 약 3.4% 증가를 보여준다. 마찬가지로 Fig. 10과 Table. 4, Fig. 12와 Table. 5를 보면 기준값 p_b를 0.5로 설정했을 때 RED와 비교하여 패킷 손실률에서는 약 33% 감소를, 링크 이용률에서는 약 8.7%의 증가를 보여준다. 패킷 손실률이나 링크 이용률에서 기준값 p_b를 0.5로 설정했을 때가 0.33으로 설정했을 때보다 약 16%의 패킷 손실률 감소와 약 5.1%의 이용률 증가를 가져온다. 따라서 기준값 p_b를 0.5로 설정했을 때가 기준값 b를 0.33으로 설정했을 때보다 패킷 손실률과 링크 이용률 면에서 좀더 개선된 성능을 보여준다. 하지만 기준값 p_b를 0.5로 설정하면 Fig. 8과 Fig. 9에서 볼 수 있듯이 기준값 p_b를 0.33으로 설정했을 때보다 큐잉 지연이 커지는 것을 감수해야 한다. 결국 큐잉 지연의 증가를 가져오지만 패킷 손실률과 링크 이용률의 개선이 이루어진다.

Ⅴ. 결론

DiffServ Network는 인터넷의 새로운 응용 서비스와 사용자들의 다양한 요구를 충족하기 위해 제안된 모델이다. DiffServ 모델은 사용자들과의 서비스 계약(SLA)에 따라 차별화된 서비스를 제공하지만 인터넷의 불확실성에 따른 폭주 트래픽으로 인한 전역 동기화 문제(global synchronization), 사용계약을 위반하는 트래픽 등에서 자유로울 수가 없다. 이러한 전역 동기화 문제와 사용계약을 위반하는 트래픽을 제어하기 위한 방법으로 RED를 사용한다. RED는 라우터 버퍼의 평균 큐 길이를 계산하여 최대 큐 한계값, 최소 큐 한계값, 최대 패킷 폐기 확률, 적응 패킷 폐기 확률 등의 매개변수를 이용하여 버퍼가 가득 차기 전에 패킷을 미리 폐기해버리는 메커니즘을 구현한다. 하지만 RED는 고정된 매개변수 값으로 인해 다양한 Network 상태에 따라 탄력적으로 동작할 수가 없다. 이러한 문제점을 극복하기 위해 본 논문에서는 Network의 부하 상태를 측정할 수 있는 방법을 제시하고 이를 이용하여 최소 큐 한계값과 최대 패킷 폐기 확률을 Network 상태에 따라 적응적으로 반응하는 확장 RED(Extended RED) 알고리즘을 제안하였다.

모의실험 수행 결과 링크 이용률은 부하 상태 기준값 β를 0.5로 설정한 ERED의 경우 RED에 비해 8.7% 정도의 링크 이용률 증가를, 기준값 β를 0.33으로 설정한 ERED에 비해서는 5.1% 정도의 링크 이용률 증가를 보였으며, 패킷 손실률은 기준값 β를 0.5로 설정한 ERED의 경우 RED에 비해 33% 정도의 패킷 손실률 감소를, 기준값 β를 0.33으로 설정한 ERED에 비해서는 16% 정도의 패킷 손실률 감소를 보였다. 기준값 β의 경우 값이 커질수록 링크 이용률이 증가하고 패킷 손실률이 감소하는 반면에 지연은 증가하였으며, 값이 작아질수록 링크 이용률은 감소하고 패킷 손실률은 증가하는 대신 지연은 감소하였다.

결국 제안 알고리즘 ERED는 RED에 비해서 패킷 손실률과 링크 이용률에서 좀더 나은 성능의 개선을 보여주지만 적응적으로 망의 부하 상태를 측정하고 이를 최소 큐 한계값과 최대 확률에 반영하는 복잡한 과정이 추가되어야 한다. 따

라서 단순한 RED에 비해 라우터의 부담이 커질 수 있으며 큐잉 지연의 증가를 감수해야 하지만 패킷 손실률의 감소와 링크 이용률의 개선을 가져온다.

향후 연구과제는 RED의 큐 가중치(W_q)처럼 ERED의 기준값 P를 결정하기 위한 방안과 제안 알고리즘에서 사용된 TH_{min}과 P_{max}이외의 매개변수, 즉 TH_{max}와 P_a의 적용에 대한 연구가 필요할 것이다.

참고문헌

David D. Clark, Wenjia Fang, "Explicit Allocation of Best Effort Packet Delivery Service", IEEE/ACM Trans. on Networking, August 1998.

J. Heinanen, et al., "Assured Forwarding PHB Group", IETF RFC 2597, June 1999.

J. Wroclawski, "The Use of RSVP with IETF Integrated Services", IETF RFC 2210, September 1997.

James Aweya, et al., "Service differentiation using a multi-level RED mechanism", Int. J. Network Mgmt, 2002.

K. Nichols, et al., "Definition of the Differentiated Services Field(DS Field) in the IPv4 and IPv6 Headers", IETF RFC 2474, December 1998.

K. Nichols, et al., "Format for Diffserv Working Group Traffic Conditioner Drafts", IETF Internet Draft, February 1999.

Kevin Fall, et al., "The NS Manual (NS Notes and Documentation)", The VINT Project, February 2000.

P. Almquist, "Type of Service in the Internet Protocol Suite", IETF RFC 1349, July 1992

R. Braden, et al., "Integrated Services in the Internet Architecture : An Overview", IETF RFC 1633, June 1994.

S. Blake, et al.. "An Architecture for Differentiated Services", IETF RFC 2475, December 1998.

S. Floyd, et. al., "NS Simulator Tests for Random Early Detection(RED) Queue Management", LBL, Technical report, October 1996.

Sally Floyd and Van Jacobson. "Random Early Detection Gateways for Congestion Avoidance", IEEE/ACM Trans. on Networking, August 1993.

Sergio Andreozzi, "DiffServ simulations using the Network Simulator : requirements, issues and solutions", ANNO ACCADEMICO, 2000.

UCB/LBNL/VINT, "The Network Simulator - ns2", 1995.

V. Jacobson et al., "An Expedited Forwarding PHB", IETF Internet Draft, November 1998.

W. Feng et al., "Technique for Eliminating Packet Loss in Congested TCP/IP Networks", U. Michigan CSE-TR-349-97, November 1997.

Y. Bernet et al., "Requirement of Diff-Serv Boundary Routers", IETF Internet Draft, May 1998

박현정, “인터넷에서 DiffServ 기반의 QoS 제어를 위한 버퍼관리 기법”, 호남대학교 석사학위논문, February 2002.

홍석원, 유영석, “Random Early Detection(RED) 최적화를 위한 자체적응 알고리즘” 정보처리학회 논문지, 1999.

홍석원, 장재준, “인터넷 QoS 모델”, Netmanias.com, August 2000.

황종규, “IP QoS 보장을 위한 DiffServ 구조 및 연구동향”, Netmanias.com, January 2002.

감사의 글

학문의 큰 꿈을 품고 대학원 문을 두드린지 벌써 2년이 흘렀습니다. 호기심과 설레임으로 가득했던 마음이 어느덧 아쉬움으로 남습니다. 처음 연구실 들어왔을 때 안기중 교수님께서 하셨던 “석사과정은 공부를 어떻게 하는지 배우는 과정”이라는 말을 이제야 알 것 같습니다. 이젠 아쉬움을 새로운 출발을 위한 발판으로 삼아 짧지만 길었던 대학원 생활을 이 한편의 논문으로 마치려 합니다.

그동안 학업에는 엄하게 생활에서는 아버지처럼 자애롭게 지도하시고 이끌어주셨던 안기중 교수님께 진심으로 감사의 마음을 드립니다. 안기중 교수님께서 안계신 동안 정신적으로 의지할 수 있었던, 항상 저를 믿어 주시고 세심하게 지도해 주신 송왕철 교수님께 진심으로 감사의 마음을 드립니다. 그리고 아낌없는 가르침을 주신 김장형 교수님, 곽호영 교수님, 변상용 교수님, 이상준 교수님, 변영철 교수님께도 감사의 마음을 드립니다.

연구실 생활하면서 많은 도움을 주신 김동춘 선배님, 김대영 선배님, 정태백 선배님, 현병철 선배님, 영미 누나, 은수 누나, 그리고 온갖 궂은일 도맡아 하고 같이 밤샘을 했던 후배 관홍에게 감사의 마음을 전합니다.

그리고 서로 의지하며 큰 힘이 되었던 진영이형, 남식, 봉수, 일남이형, 혁준이형, 은경 누나 등 대학원 동기들과 강석이 형, 영도 형, 진석이 형, 정하 누나, 영수 형, 그 외 모든 선배님들, 후배들에게도 감사의 마음을 전하며, 앞으로 항상 좋은 일만 가득하길 기원합니다.

바쁘다는 핑계로 많은 모임에 빠지기 일쑤였지만 모든 것을 이해하고 힘이 되어준 경중, 용주, 재경, 용순 등 모든 친구들에게도 감사의 마음을 전합니다.

항상 옆에서 힘들 때는 격려하고 용기를 주며 기쁠 때는 기쁨을 같이 하며 공부에만 전념할 수 있도록 배려를 해주며 너무나도 큰 힘이 되었던 사랑하는 선영에게 사랑을 가득 담은 감사의 마음을 전합니다.

마지막으로 사랑하는 동생 기범, 민범이와 진정으로 아들을 믿고 한없는 희생과 사랑을 베푸시며 인생의 든든한 버팀목이 되어 주신 아버님, 어머님께 이 논문을 바칩니다.