

## DiffServ 망에서의 혼잡제어 알고리즘

김은범\* · 안기중\*\*

## Congestion Control in DiffServ Networks

Eun-Bum Kim\* · Khi-Jung Ahn\*\*

### ABSTRACT

In this paper, I extend 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, 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, packet loss rate and throughput for each class to some extent.

**Key Words** : DiffServ, QoS, AF PHB, RED, Congestion Control

### 1. 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 기존의 인터넷은 모든 패킷을 동일하게 전달하는 Best-Effort 서비스만을 제공하고 있기 때문에 서비스에 따른 패킷의 손실 또는 지연 등의 QoS에 대한

요구사항을 보장해 주지 못하고 있다. 따라서 인터넷에서 서비스의 QoS를 보장해 주기 위해서는 현재의 모델과는 다른 새로운 서비스 모델을 필요로 하며 이에 따라 등장한 모델이 DiffServ이다. DiffServ 모델은 흐름 단위로 QoS를 보장하지 않고 흐름들의 집합(aggregation)을 단위로 서비스를 차별화함으로써 훨씬 간단하고, 따라서 대규모 망에도 확장 적용할 수 있는 모델이다[1]. DiffServ 구조는 네트워크에 진입하는 트래픽을 망의 경계에서 분류하고 조절하며, 다른 행동 집합(BA)에 할당하는 모델을 기초로 한다. DiffServ의 핵심부분인 경계라우터는 TCB, 버퍼관리, 스케줄러로 구성되어있다. TCB는 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, RIO, WRED, FRED

\* 제주대학교 대학원

Graduate School, Cheju Nat'l Univ.

\*\* 제주대학교 통신·컴퓨터공학부, 첨단기술연구소

Faculty of Telecommunication and Computer Eng., Cheju Nat'l Univ., Res. Inst. Adv. Tech.

등의 방법이 사용된다[2,3]. AF PHB의 버퍼관리를 위해 사용되는 RED 메커니즘은 라우터에서 체증이 발생하기 전에 능동적인 버퍼관리를 통하여 전역 동기화에 의해 순간적으로 링크의 이용률이 감소하는 것을 방지하고 라우터 내의 큐 길이를 가능한 작은 크기로 유지케 하는 방법이다[4,5]. 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 상관 없이 동일한 값으로 설정되기 때문에 다양한 트래픽 상황에 일괄 적용하게 되면 여러 가지 문제점이 발생한다. 일반적으로 버스트한 데이터의 특성을 갖는 망에서는 최소 큐 한계값을 크게 하면 링크의 사용 효율을 높일 수 있으며, 지속적으로 망에 과부하가 가해지고 있을 때는 최대 확률 값을 높게 설정하여야 효율적인 체증 관리가 가능하다고 알려져 있다. 기존의 망의 특성은 버스트한 데이터의 특성과 지속적인 과부하의 망 특성을 동시에 갖고 있어 동일한 매개변수의 값을 설정하면 상이한 두 환경에서 잘 적용되기가 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 적응적으로 조절하여 다양한 망 환경에서도 잘 적용될 수 있는 확장 RED를 제안한다.

## II. AF PHB에서의 버퍼관리 메커니즘

### 2.1. RED(Random Early Detection)

RED의 목표는 혼잡 회피와 전역동기화 회피이다. 이를 위해 알고리즘은 FIFO 출력 큐에 새로운 패킷이 도착할 때마다 두 기능을 수행하는 데 첫 번째 절차는 평균 큐길이( $avg$ )를 계산하며 이 평균 큐길이를 두 한계치와 비교한다.  $avg$ 가 하한치  $TH_{min}$ 보다 작으면 혼잡이 최소이거나 없는 것으로 가정하여 이 패

킷을 큐에 넣는다.  $avg$ 가 상한치  $TH_{max}$ 보다 크면 혼잡이 심각하다고 가정하여 이 패킷을 버린다.  $avg$ 의 정확한 값에 따라  $avg$ 를 상한선에 이르게 증가시키는 확률  $P_a$ 를 계산한다. 큐가 이 영역에 있으면 이 패킷을 확률  $P_a$ 로 버리며 확률 $(1-P_a)$ 로 큐에 들어가게 한다. 기본적으로 알고리즘의 첫 단계(큐 크기 계산)는 허용할 수 있는 폭주 정도를 결정하며 알고리즘의 두 번째 단계(패킷 폐기 결정)는 현 수준의 혼잡이 있을 때 폐기되는 패킷들의 빈도를 결정한다.

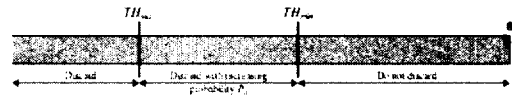


Fig. 1. RED Buffer

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

$$P_b = \frac{P_{max}(avg - TH_{min})}{TH_{max} - TH_{min}} \quad (1)$$

$P_a$ 는 폭주 단원지가 불이익을 당하지 않게 비교적 공평하게 간격을 두고 폐기를 결정한다.  $P_a$ 가  $P_b$ 에 비해 전역 동기화가 일어날 가능성이 적다.

$$P_a = \frac{P_b}{1 - count \times P_b} \quad (2)$$

### 2.2. 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  $P_a$ 에 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.  $Q_{avg}$ 를 구할 때 실제 큐 길이가  $Q_{avg}$ 가 변하는데 미치는 영향을

나타내는 값이 큐 가중치( $W_q$ )이다.  $W_q$ 가 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.  $W_q$ 는 0.001 이상을 쓰도록 권하고 있고 일반적으로 0.002를 사용하고 있다.  $P_{max}$ 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. 실제 망에 RED를 적용했을 때  $TH_{min}$ 과  $TH_{max}$  값을 설정하는 것은 상당히 중요한 일이지만 이것은 해석적으로나 모의실험만으로 정할 수 있는 매개변수는 아니다. 이 매개변수의 범위는 다음과 같이 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 지연시간이 길어진다.  $TH_{min}$ 과  $TH_{max}$ 의 차이는 왕복시간 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.

### 2.3. 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 초래

하게 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 수율(throughput)에 영향을 주어 성능이 나빠진다[6]. 따라서, RED에서는 4개의 매개변수를 고정된 값으로 사용하기 때문에 지속적으로 트래픽이 버스트할 때는 RED가 제대로 기능을 발휘하지 못하는 상황이 발생할 수 있으며 이를 보완하기 위해서는 매개변수의 값을 망의 상태에 따라 조절하여 적용적으로 반응할 수 있도록 하여야 한다. RED의 고정된 매개변수들은 적절한 트래픽 부하일 경우에 잘 적용되도록 정해지므로 버스트 한 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.

## III. Extended RED Algorithm

위에서 기술한 RED의 문제점들은 고정적인 매개변수들로 인해 망의 지속적인 트래픽 상태를 반영할 수 없다는 알고리즘의 특성 때문에 야기되는 문제들이다. RED에서 최대 확률 외에 성능에 중대한 영향을 미치는 매개변수는 여러 가지가 있다. 이 중에서 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 지속적으로 트래픽 상태를 감지하는 새로운 방법을 제시하고 이를 이용하여 최소 큐 한계값과 최대 확률을 트

래픽 상태에 따라 적응적으로 반응하도록 확장된 RED 알고리즘을 제안한다.

### 3.1. 트래픽 상태 감지 방법

망에서 지속적으로 변화하고 있는 트래픽 상태에 효율적으로 적응하기 위하여 일반적으로 사용되는  $W_q=0.002$ 를  $1/20$  이하로 감소시켜 식 (3)처럼 가중평균을 계산하여  $Q_L$ 라 하고 망 트래픽 상태의 척도로 사용한다.

$$Q_L \leftarrow (1 - W_L) \times Q_L + W_L \times Q_{size} \quad (3)$$

단,  $W_L < W_q/20$ ,  $Q_L$ 의 초기값 = 0

망 트래픽 상태의 척도로 사용하는  $Q_L$ 은 지수가중(exponentially weighted) 평균을 이용하여 계산한 평균 큐 길이이다. 비중  $W_L$ 은 실제 큐 크기의 변화에 따라  $Q_L$ 이 얼마나 빨리 변하는가를 결정한다.  $W_L$ 이 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  $W_L$ 을 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. 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  $W_L$ 이 너무 작기 때문에 이미 올라간 값이 떨어질 때에도 많은 시간이 필요하게 된다.  $W_L$ 이 너무 클 경우에는 이와는 반대로  $Q_{avg}$  값이 실제 큐 길이의 영향을 많이 받게 된다.  $Q_{avg}$ 가 실제 큐 길이의 영향을 받아 기복이 크다는 것은 RED를 사용하는 의의를 줄이는 일이다. 결과적으로  $Q_L$ 은 실제 큐 크기의 변화보다 상당히 뒤에 일어나게 한다. 이렇게 작은  $W_L$ 을 사용하면 알고리즘이 짧은 폭주의 혼잡에 반응하지 않게 된다. 이 값이  $TH_{min} \times \beta$  부근에 있을

경우를 정상적인 상태, 큰 경우를 과부하 상태, 작은 경우를 낮은 부하 상태라 판단한다. 여기서  $\beta$ 는 정상 상태의 기준을 결정하는 매개변수로 망의 특성에 따라 달라질 수 있다. 작은 큐잉 지연을 요구하면 작게 하고, 큐잉 지연보다 좋은 링크 이용률을 원한다면 큰 값을 선택할 수 있다.

### 3.2. 최소 큐 한계값(THmin) 결정

최소 큐 한계값은 망 부하 상태에 적응하여 설정할 수 있도록 식 (4)와 같이 결정한다.

$$TH_{min} = \alpha \times (TH_{max} \times \beta - Q_L) + \gamma \quad (4)$$

단,  $0 < \beta < 1$

여기서  $\alpha$ 와  $\gamma$  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.

### 3.3. 최대확률(Pmax) 결정

망의 트래픽 상태에 따라  $P_{max}$  값을 자동적으로 식 (5)를 이용하여 결정하며 망의 부하가 적을 때는 낮게 하고 많을 때는 높게 하여 다양한 망의 환경에 효율적으로 적용할 수 있도록 결정한다.

$$P_{ar} = \frac{Q_L}{TH_{max} - Q_L \times (\beta - 1)} \quad (5)$$

$$P_{max} = P_{min_i} \times (\delta^{P_{ar}} - \delta + 1)$$

$P_{min_i}$  :  $P_{max}$ 의 초기 값으로 주어지는 값

## IV. 모의실험 결과 및 성능분석

### 4.1. 모의실험 환경

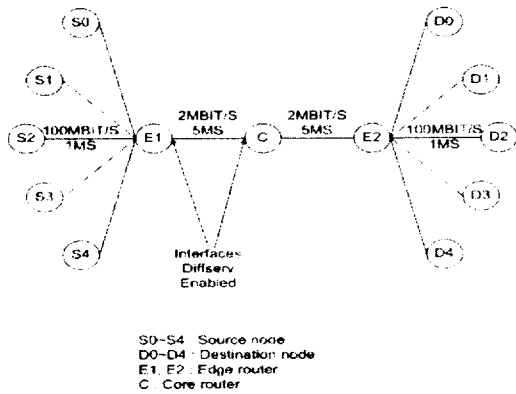


Fig. 2. Simulation Network Topology

모의실험을 위한 망 형상은 Fig. 2와 같다. 송신측은 S0부터 S4까지 5개의 노드이며 경계 라우터(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. 트래픽은 DE PHB, EF PHB, AF PHB의 세 가지 패킷전달방식(PHB)으로 발생되고 전달된다. EF PHB는 우선순위가 가장 높은 전달 방식이며 트래픽 클래스는 프리미엄 서비스로, AF PHB Group은 사용자의 요구를 만족시키는 보장 서비스로서 골드 서비스를 트래픽 클래스로 할당한다. 이외에 DE PHB는 현재 인터넷 라우터에서 널리 사용되고 있는 Best-Effort 방식을 정의하며 DiffServ를 지원하지 않는 사용자를 허용하기 위함이다. 프리미엄 서비스에는 300kbit/s, 골드 서비스에는 1Mbit/s, Best-Effort 서비스에는 700kbit/s 대역폭을 할당함으로써 이 구간의 전체 대역폭은 2Mbit/s 이다. 특히 AF PHB에는 RED와 ERED가 적용되며 트래픽 유형은 FTP와 Telnet 이다. RED의 매개변수의 경우  $W_q$ 는 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  $W_q$ 는 0.002,  $P_{max}$ 는 0.02,  $W_L$ 는 0.00004,  $a$ 는 5,  $\beta$ 는 0.5,  $r$ 는 5,  $\delta$ 는 5로 설정하였다. 모의실험은 두개의 시나리오로 구분하여 진행한다. 첫 번째 시나리오는 망의 정상 상태를 의미하는 기준값  $\beta$ 를 0.5로 설정하여 수행하고, 두 번째 시나리오는 기준값  $\beta$ 를 0.33으로 설정하여 수행한다. 모의실험 결과는 링크

이용률, 패킷 손실률 등으로 나타낼 수 있으며 기준값  $\beta$ 를 0.5와 0.33으로 했을 때 결과를 비교, 분석한다. 모의실험을 위해서 UC Berkeley에서 개발하고 현재 널리 사용되고 있는 네트워크 모의실험 도구인 NS-2를 사용하였다[7,8,9]. 큐잉 방식은 WFQ를 사용하였으며 AF PHB에 RED와 ERED를 적용하여 수행 결과를 비교, 분석하였다.

4.2. 모의실험 결과 및 분석

Fig. 3은 기준값  $\beta$ 를 0.5로 설정했을 때의 AF PHB에서 RED와 ERED의 패킷 손실률을 보여 준다. Fig. 4는 기준값  $\beta$ 를 0.33으로 설정했을 때의 패킷 손실률을 보여 주는데 기준값  $\beta$ 를 0.5로 설정했을 때보다 패킷 손실률이 더 높다는 것을 볼 수 있다. 이러한 결과는 기준값을 크게 설정함으로써 최대 한계값을 낮추는 것에서 기인한다. Table. 1의 평균값을 보면 기준값  $\beta$ 를 0.5와 0.33으로 설정한 ERED의 경우 RED에 비해 각각 33%, 21% 정도의 패킷 손실률 감소를 가져온다는 것을 알 수 있다.

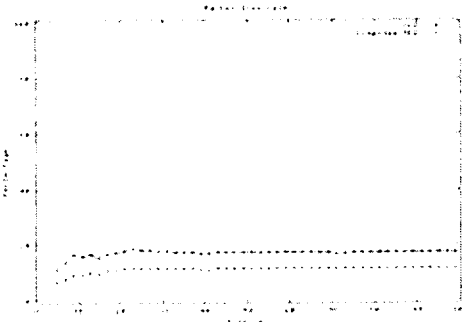


Fig. 3. Packet Loss Rate ( $\beta = 0.5$ )

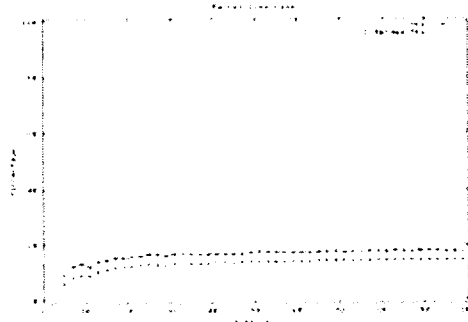


Fig. 4. Packet Loss Rate ( $\beta = 0.33$ )

Table. 1. Simulation result (Packet Loss Rate)

		19	39	59	79	99	Average
RED		15.3	16.5	17.2	17.1	17.1	16.64
ER	$\beta=0.5$	10.1	11.1	11.6	11.5	11.6	11.18
ED	$\beta=0.33$	11.8	13.5	13.3	13.8	13.8	13.24

Table. 2. Simulation result (Link Utilization)

		20	40	60	80	100	Average
RED		0.84	0.82	0.82	0.81	0.81	0.820
ER	$\beta=0.5$	0.82	0.89	0.89	0.88	0.89	0.892
ED	$\beta=0.33$	0.86	0.85	0.85	0.84	0.84	0.848

Fig. 5는 기준값  $\beta$ 를 0.5로 설정했을 때의 AF PHB에서 RED와 ERED의 링크 이용률을 보여 준다. ERED가 RED보다 링크 이용률이 높은 것을 볼 수 있다. Fig. 6은 기준값  $\beta$ 를 0.33으로 설정했을 때의 링크 이용률을 보여 주는데 기준값  $\beta$ 를 0.5로 설정했을 때보다 링크 이용률이 더 낮다는 것을 볼 수 있다. 이러한 결과는 패킷 손실률과 마찬가지로 기준값  $\beta$ 를 크게 설정함으로써 최대 한계값을 낮추는 것에서 기인한다. Table. 2의 평균값을 보면 기준값  $\beta$ 를 0.5와 0.33으로 설정한 ERED의 경우 RED에 비해 각각 8.7%, 5.1% 정도의 링크 이용률 증가를 가져온다는 것을 알 수 있다.

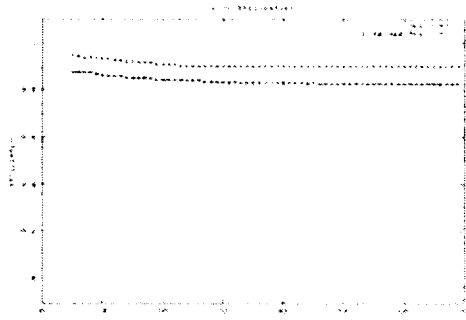


Fig. 5. Link Utilization ( $\beta = 0.5$ )

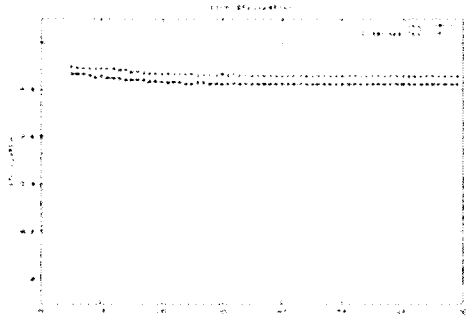


Fig. 6. Link Utilization ( $\beta = 0.33$ )

## V. 결론

DiffServ Networks에서 전역 동기화 문제와 사용 계약을 위반하는 트래픽을 제어하기 위한 방법으로 RED를 사용한다. 하지만 RED는 고정된 매개변수 값으로 인해 다양한 망 상태에 따라 탄력적으로 동작할 수가 없다. 이러한 문제점을 극복하기 위해 본 논문에서는 망의 부하 상태를 측정할 수 있는 방법을 제시하고 이를 이용하여 최소 큐 한계값과 최대 패킷 폐기 확률을 망 상태에 따라 적용적으로 반응하는 확장 ERED(Extended RED) 알고리즘을 제안하였다. 모의실험 수행 결과 링크 이용률은 부하 상태 기준값  $\beta$ 를 0.5와 0.33으로 설정한 ERED의 경우 RED에 비해 각각 8.7%, 5.1% 정도의 링크 이용률 증가를 보였으며, 패킷 손실률은 기준값  $\beta$ 를 0.5와 0.33으로 설정한 ERED의 경우 RED에 비해 각각 33%, 16% 정도의 패킷 손실률 감소를 보였다. 기준값  $\beta$ 의 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 적용적으로 망의 부하 상태를 측정하고 이를 최소 큐 한계값과 최대 확률에 반영하는 복잡한 과정이 추가되어야 한다. 따라서 단순한 RED에 비해 라우터의 부담이 커질 수 있으며 큐잉 지연의 증가를 감수해야 하지만 패킷 손실률의 감소와 링크 이용률의 개선을 가져온다. 향후 연구과제는 RED의 큐 가중치처럼 ERED의 기준값  $\beta$ 를 결정하기 위한 방안과 제안 알고리즘에서 사용된  $TH_{min}$ 과  $P_{max}$ 이외의 매개변수, 즉  $TH_{max}$ 와  $Pa$ 의 적용에 대한 연구가 필요할 것이다.

## 참고문헌

- 1) S. Blake, et al., 1998, An Architecture for Differentiated Services, IETF RFC 2475, pp. 2-31.
- 2) K. Nichols, et al., 1998, Definition of the Differentiated Services Field(DS Field) in the IPv4 and IPv6 Headers, IETF RFC 2474, pp. 3-16.
- 3) K. Nichols, et al., 1999, Format for Diffserv Working Group Traffic Conditioner Drafts, IETF Internet Draft, pp. 1-2.
- 4) Sally Floyd and Van Jacobson, 1993, Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Trans. on Networking, Vol. 1, No. 4, pp. 397-413.
- 5) James Aweya, et al., 2002, Service differentiation using a multi-level RED mechanism, Int. J. Network Mgmt, pp. 81-98
- 6) 홍석원, 유영석, 1999, 버퍼 관리 기반 체증 제어 방식의 최적화를 위한 자체 적응 알고리즘 정보처리학회 논문지, Vol. 6, No. 11, pp. 3097-3108
- 7) S. Floyd, et. al., 1996, NS Simulator Tests for Random Early Detection(RED) Queue Management, LBL, Technical report.
- 8) Sergio Androozzi, 2000, DiffServ simulations using the Network Simulator : requirements, issues and solutions, ANNO ACCADEMICO, pp. 1-116.
- 9) UCB/LBNL/VINT, 1995, The Network Simulator-ns2.