

석사학위 논문

DiffServ망에서 Class와 Scheduling모
델에 따른 특성 연구



제주대학교 대학원

허 지 완

2004년 11월

DiffServ망에서 Class와 Scheduling모 델에 따른 특성 연구

지도교수 송 왕 철

허 지 완

이 논문을 공학 석사학위 논문으로 제출함

2004년 11월



제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

허지완의 공학 석사학위 논문을 확인함

심사위원장 _____인

위 원 _____인

위 원 _____인

제주대학교 대학원

2004년 11월

목 차

제 I 장 서론.....	6
제 II 장 DiffServ의 구성요소.....	8
제 1 절 DiffServ의 구조	8
1.1 트래픽 클래스의 정의.....	9
1.2 트래픽 조절기(Traffic Shaper).....	13
1.3 큐잉(Queueing)과 스케줄링(Scheduling).....	14
제 III 장 스케줄링 알고리즘의 시뮬레이션.....	16
제 1 절 EF PHB Class 와 AF PHB Class.....	17
제 2 절 스케줄링 기법의 비교	21
제 IV 장 결론.....	28
참고 문헌	30
영문 요약 Summary.....	32



표 목 차

표 1 IPv4와 IPv6의 DSCP영역-----	10
표 2 IP TOS Field-----	11
표 3 IP TOS 필드 정의-----	11
표 4 AF code point-----	12
표 5 EF Code point-----	13
표 6 스케줄링 알고리즘의 비교-----	16
표 7 EF 트래픽 설정-----	18
표 8 EF/AF 트래픽 전송 수치-----	20
표 9 PRI Scheduling code-----	22
표 10 WRR Scheduling code-----	23
표 11 스케줄링 알고리즘 별 종합 결과값-----	24
표 12 RR 스케줄링의 폐기된 패킷량-----	26
표 13 PRI 스케줄링의 폐기된 패킷량-----	26
표 14 WRR 스케줄링 폐기 패킷량-----	26
표 15 WIRR 스케줄링 폐기 패킷량-----	27

그림 목 차

그림 1 DiffServ의 구성-----	8
그림 2 PHB-----	9
그림 3 큐 관리 모델과 스케줄러-----	14
그림 4 시뮬레이션 모델-----	17
그림 5 EF, AF Traffic Graph-----	19
그림 6 EF와 AF 트래픽 전송/폐기량 그래프-----	20
그림 7 RR scheduling-----	21
그림 8 PRI scheduling-----	22
그림 9 WRR scheduling-----	23
그림 10 WIRR scheduling-----	23
그림 11 AF PHB Class Traffic-----	24
그림 12 EF Class Traffic Drop-----	25

제 I 장 서론

현재의 일반적인 인터넷 망에서는 단순한 매커니즘을 가진 서비스만을 지원하고 있다. 지난 수년간 급속히 성장해온 인터넷은 전통적인 FTP나 email 등의 통신이나 자료 전달형 서비스 외에 VoIP, VOD와 같은 실시간 서비스나 고속 전송이 필요한 응용들을 등장시켰다. 이러한 시간이나 속도에 민감한 응용들은 네트워크에서 빠른속도를 보장 받거나 균등 시간, 적은 지연을 가지고 전송 될 것을 요구하고 있다.

위와 같은 요구를 만족시키기 위해 종단간 RSVP 프로토콜에 의해 이루어지는 IntServ(Integrated Service)와 DiffServ(Differentiated Service) 모델 등과 같은 QoS(Quality of Service)를 지원하는 네트워크 모델이 제안되었다.

IntServ 모델은 종단간의 호스트 간에 RSVP(Resource reservation Setup Protocol)를 이용하여 합의된 플로어에 대하여 미리 정의된 속도를 보장하게 된다. 이 모델은 네트워크가 방대해 지고 RSVP에 의해 대역폭을 예약하고자 하는 호스트의 수가 많아짐에 따라 확장성에 문제가 도출되었다. 이 문제를 해결하고자 하는 DiffServ 모델은 트래픽을 플로어 단위로 관리하지 않고 일련의 플로어 즉, 클래스 별로 분류, 처리한다.

DiffServ 모델에서는 각기 다른 가입자에게 네트워크 버퍼나 회선의 전송속도와 같은 네트워크 자원을 분배하는 것이 가장 중요하다. 이를 위해 QoS 지원 네트워크 모델에서는 자원을 분배하기 위해 트래픽을 분류하고 큐를 관리하며 혼잡 상태를 벗어나기 위한 방법들이 연구 되고 있다.

차별화된 네트워크 품질을 보장하는 DiffServ 네트워크의 구성 요소는 다음과 같다.

- Packet Classifier
- Admission Control, Traffic Shaping

- Queueing, Scheduling
- Packet Drop

패킷의 분류과정에서는 각 가입자의 IP와 같은 정보는 물론 특정한 어플리케이션에 대한 분류나 특정한 프로토콜로 들어오는 플로우에 따른 분류가 필요하다. 차별화 서비스에서는 IP 패킷 헤더의 TOS중 DSCP 필드를 마킹하여 분류하게 된다.

패킷 분류과정 후에 Admission Control은 이 패킷의 클래스에 따른 충분한 네트워크 리소스가 있는지는 체크하고 만약 있다면 그 클래스에 따라 핸들링을 수행하게 된다. 트래픽 셰이핑 과정에서는 일련의 플로우에 대하여 전송률을 일정하게 하는 과정이다. 이때 쓰이는 대표적인 셰이핑 기술에는 널리 알려진 토큰 큰 버킷이 있다.

이제 패킷이 큐에 들어오게 되면 해당하는 큐에 따라 전송 우선 순위와 폐기 우선 순위를 정해 패킷을 전송한다. 이러한 큐잉 방법은 FIFO, PQ, FQ, WFQ, CBQ와 같은 방법들이 쓰이고 있거나 제안되어 있다.

그리고 네트워크에 혼잡 제어를 위해 패킷 폐기 알고리즘으로 Tail Drop, Early Drop과 RED 등의 메커니즘이 제안되었다.

본 논문에서는 IETF에서 제안한 EF와 AF 트래픽이 어떻게 정의한 특성에 맞게 전송할 수 있는가에 대하여 큐잉과 스케줄링을 이용하여 구현하고, 클래스에 따른 트래픽이 혼재한 DiffServ 망에서 적합한 스케줄링 알고리즘을 찾기 위해 시뮬레이터를 통해 동작 특성과 성능을 비교한다. 2장에서는 큐관리 메커니즘과 스케줄링 방법에 대하여 분석하고, 3장에서는 각 스케줄링 알고리즘에 대한 시뮬레이션 결과를 토대로 특성을 분석한다. 4장에서는 EF, AF 트래픽이 어떤 큐잉 알고리즘과 스케줄링 기법으로 구현될 수 있는 하는 것과 구현하고자 하는 차별화 서비스에서 가장 적합한 스케줄링 방법에 대하여 설명하고 끝을 맺는다.

제 II 장 DiffServ의 구성요소

<그림 1>에서 DiffServ 네트워크 모델의 구성을 보여준다. 패킷은 분류기 (Classifier)에서 IP나 TCP port와 같은 정보와 함께 DSCP 코드 포인트를 기초로 하여 패킷을 분류한다.

미터와 마커는 패킷을 프로파일과 비교하여 적합한 코드포인트로 패킷을 마킹하는 역할을 하게 되고, 트래픽 조절기(Shaper)는 트래픽 프로파일에 따라 패킷의 속도를 일정하게 조절한다.

트래픽 조절기로도 과도한 전송 패킷을 처리할 수 없어 더 이상 저장할 버퍼가 없다면 패킷은 Dropper에 의해 폐기 된다.

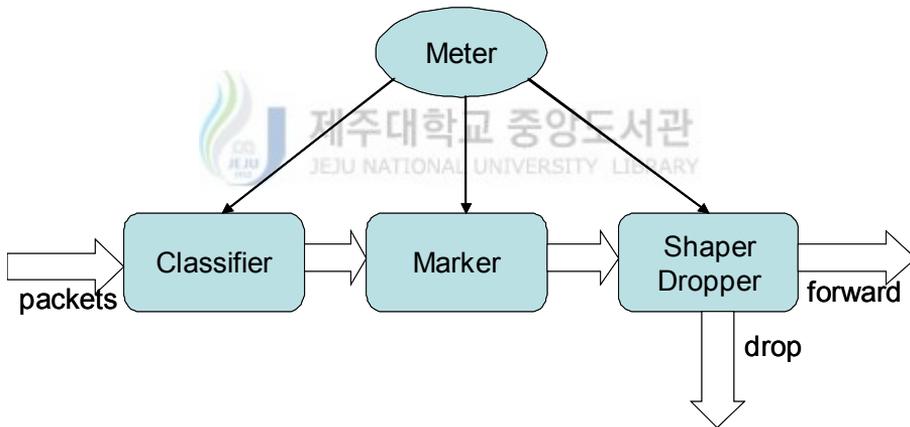


그림 1 DiffServ의 구성

제 1 절 DiffServ의 구조

DiffServ 네트워크에서는 들어오는 패킷을 먼저 분류하고 패킷의 헤더에 적절한 코드를 할당함으로써 각 패킷에 서로 다른 BA(Behavior Aggregate)를 할당한다. 각 BA는 DSCP(DiffServ Code Point)에 따라 분류된다. 이렇게 분류된 패킷은 코어망에서 분류에 맞는 PHB에 의하여 전달되게 된다.

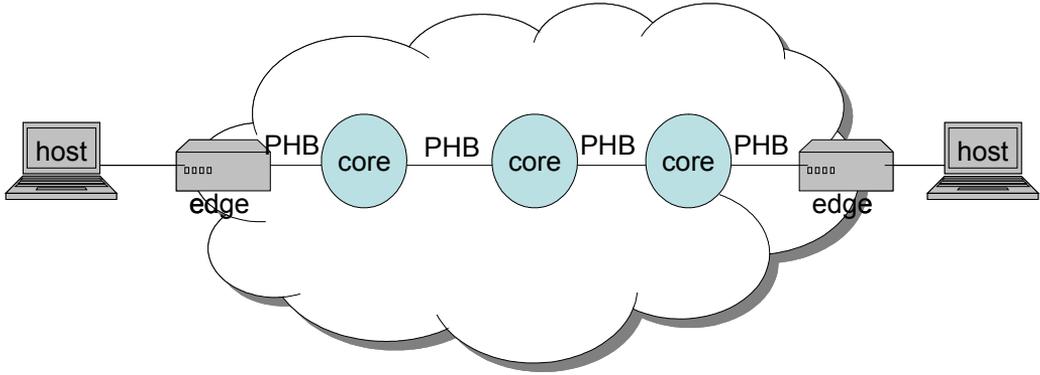


그림 2 PHB

<그림 2>와 같이 edge router 사이의 트래픽은 코어 네트워크에 다른 DiffServ 네트워크나 일반적인 QoS를 지원하지 않는 네트워크를 지나게 되며, 각 노드들은 DSCP에 정의된 PHB를 적용하여 라우팅 하게 된다.

1.1 트래픽 클래스의 정의

PHB는 앞서 기술한 것과 같이 DSCP로 정의되며, 장치들에 대한 버퍼나 대역폭의 할당 또는 우선 순위, 지연, 손실에 관한 특성을 정의한다. 이들은 다시 블럭화 되어 하나의 PHB 그룹으로 정의된다. 이에 따라 각 장치들은 큐 매니지먼트나 스케줄링을 수행하게 된다. 이러한 PHB는 노드에서 정책적으로 변화될 수 있으며, 특정한 매커니즘에 따라 정해지는 것은 아니다. 또한 PHB는 또한 한개, 또는 여러개의 그룹이 하나의 영역내에서 사용 될 수 있다. 표준 권고안의 코드포인트는 정의되어 있다. 하지만, 이 코드포인트들이 모두 PHB에 일대일로 대응되는 것은 아니며, 여러개의 코드포인트가 하나의 PHB에 대응될 수 있다. 이는 DSCP에 정의된 AF(Assured Forward) Class, EF(Exoedited

forward) Class가 반드시 어떤 특정한 전송 품질을 보장 받는 것이 아니며, 네트워크에서 정의한 정책에 의하여 변화 될 수 있다는 것을 의미한다. 이는 아주 많은 PHB 그룹을 정의하여 계층화를 많이 할 수도, 그렇지 않고 단순화 할 수도 있다는 것이 된다.

또한 어떤 PHB그룹에도 속하지 못한 코드포인트 트래픽들은 Default PHB를 두어 대응시켜야 한다. DiffServ 네트워크가 아닌 특정한 서비스를 요구하지 않는 BE(Best Effort) Class의 트래픽들이 대표적으로 이 PHB를 사용하는 트래픽이 될 것이다.

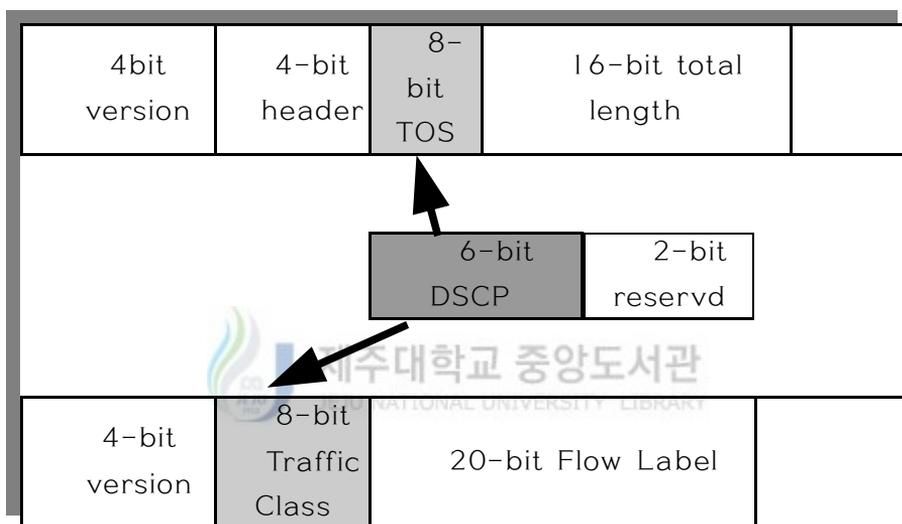


표 1 IPv4와 IPv6의 DSCP영역

code point의 마킹은 <표 1>의 IP헤더 TOS(TypeOfService)에서 2bit의 영역을 제외한 6bit의 영역을 DSCP 필드로 정의하여 사용한다.

네트워크 자원이 남아 있는 한 BE 트래픽은 네트워크 자원을 최대한 사용하여 전송된다는 것을 의미한다. DSCP필드 다시 3비트씩 둘로 나누어 진다. <표 2>에서와 같이 처음 3비트는 클래스를 정의하며, 이는 EF, AF를 나타낸다. 나머지 3비트는 폐기 우선권(drop precedence)을 정의하는데 쓰인다.

Drop Precedence (3bit)	Class Selector (3bit)				
Precedence	D	T	R	0	0

표 2 IP TOS Field

Bit	Description
0,1,2	Precedence
3	0 = Normal delay
	1 = Low delay
4	0 = Normal throughput
	1 = High throughput
5	0 = Normal reliability
	1 = High reliability
6,7	사용하지 않음

표 3 IP TOS 필드 정의



폐기 우선권(Drop Precedence)은 표 3과 같이 정의되었다.

AF PHB는 네개의 클래스로 분류되고, 이 클래스에 따라 대역폭이 보장된다. AF 클래스의 패킷은 세가지 폐기 우선권을 가지므로 총 12가지의 속성을 가질 수 있게 된다.

AF PHB는 네개의 클래스로 분류되고, 각 클래스에는 세가지 폐기 우선 순위가 정의되어 총 12가지의 속성을 가질 수 있다. <표 4>와 같이 각 클래스는 버퍼 공간이나 대역폭과 같은 네트워크 자원을 특정 양 만큼 할당하기 위해 사용되고, 폐기 우선 순위는 혼잡 발생시 AF 클래스는 상대적인 우선권을 고려하여 패킷을 폐기하기 위해 사용된다.

	Class 1	Class 2	Class 3	Class 4
Low Drop	AF11 : 001010	AF21: 010010	AF31: 011010	AF41: 100010
Med Drop	AF12: 001100	AF22: 010100	AF32: 011100	AF42: 100100
High Drop	AF13: 001110	AF23: 010110	AF33: 011110	AF43: 100110

표 4 AF code point

그러므로 AF PHB는 최소의 품질을 보장하려는 서비스에 적당하다고 할 수 있다.

예를 들어 어떤 AF 클래스의 트래픽이 평균 속도 이하이면 낮은 순위의 폐기 우선권을 적용하고 최대 속도로 전송되어 오는 트래픽에서는 높은 폐기 우선 순위를 적용한다. 그 분류에 속하지 않는 트래픽들은 중간 정도의 우선 순위를 적용하게 된다. 이로써 하나의 AF 클래스내에서 서로 다른 세가지 폐기 우선권을 가지게 할 수 있다.

이런 네트워크 운영 정책은 금, 은, 동 올림픽 서비스의 구현이 가능하게 한다.

EF PHB 트래픽은 적은 지연과 적은 손실을 가지며 우선 순위가 높다. 이 EF PHB에 의한 트래픽은 가장 양질의 네트워크 자원을 사용해야 하는 서비스에 적절하다 할 수 있다.

EF 클래스 트래픽들은 큐에 머무는 시간이 없거나 아주 작아야 한다. EF 클래스를 정의한 목적인 다른 어떤 트래픽보다 안정적으로 전송되기 위해서는 이외의 트래픽에 대해 큐 매니지먼트에서 가장 폐기율이 높은 패킷 폐기 매커니즘을 사용하고, 스케줄러에서는 이 클래스의 트래픽이 가장 먼저 디큐잉 되도록 하여야 한다.

그러나 EF PHB 클래스의 트래픽이 전송중이라 하더라도 AF 클래스와 BE 클래스의 트래픽들이 동일한 네트워크 상에서 전송되어야 하므로 EF PHB 클래스와 AF PHB 트래픽은 BE 클래스의 트래픽을 위하여 계약된 대역폭에 대해서 적용되도록 하여야 한다.

만약 EF 클래스 트래픽이 프로파일에 정의된 이상의 트래픽으로 전송되면,

쉐이핑을 통해 속도를 조정하거나 폐기를 결정해야 한다.

그렇지만 EF PHB 클래스는 다른 어떤 트래픽보다 전송 품질을 최대한 보장하므로 이는 프리미엄 서비스의 구현을 가능하게 한다.

표 5 는 EF 클래스를 위한 표준 코드포인트를 나타낸다.

1	0	1	1	1	0
Precedence			D	T	R

표 5 EF Code point

그러므로 EF 클래스의 서비스는 가장 높은 전송량을 요구하거나 균등 시간, 즉, 고정 비트율 전송을 요구하는 VoIP 응용과 같은 서비스에 적당하다.

1.2 트래픽 조절기(Traffic Shaper)

네트워크 상에서 과다한 트래픽이 발생하는 경우 DiffServ 모델에서는 트래픽 프로파일에 의하여 속도를 조절하게 된다. Meter는 가입자와 네트워크 제공자 간의 SLA(Service Level Agreement)에 의해 트래픽 프로파일과 일치하는지를 검사한 후 Marker에 의해 PHB 그룹에 따른 DSCP 필드를 마킹한 후 나가게 된다.

트래픽 조절기는 패킷의 속도를 평균화하기 위해 트래픽을 지연시킨다. 트래픽 조절기에 의해서도 더 이상 저장할 버퍼가 존재하지 않는다면 dropper에 의해 패킷은 폐기 될 것이다.

리키 버킷(leaky bucket)과 토큰 버킷(token bucket)은 대표적인 조절기 알고리즘이다.

1.3 큐잉(Queueing)과 스케줄링(Scheduling)

egress edge router(출구 경계 라우터)에 도착한 패킷은 큐에 저장되게 되는데, 큐는 모든 우선권에 대하여 하나씩 존재하는 것이 바람직하다. 이는 큐 매니지먼트를 간단하게 할 수 있다. EF, AF, BE 트래픽에 대하여 각각 다른 큐에 저장한다면, 스케줄링의 처리가 단순해 질 것이다.

네트워크의 혼잡 상황 제어를 위해 IETF에서는 라우팅 알고리즘을 스케줄링 알고리즘과 큐 관리 알고리즘으로 분류하고 있다.^[11] 이중 스케줄링 알고리즘은 라우터에 도착한 패킷들이 어떻게 전송 될 것인가를 결정하는 방법이다.

큐 관리 메소드와 스케줄링에 관한 도해는 <그림 3>와 같다.

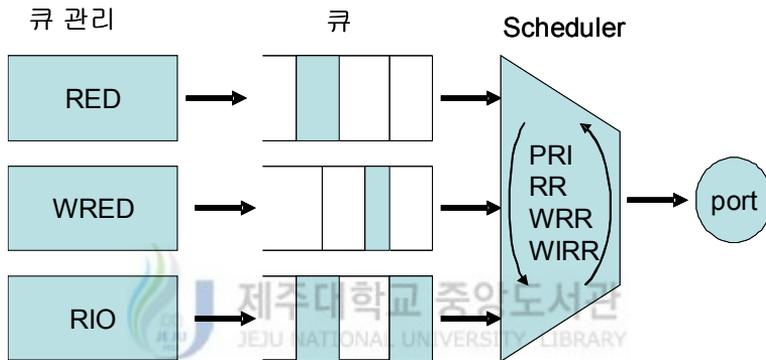


그림 3 큐 관리 모델과 스케줄러

일반적으로 큐는 네트워크상의 라우터나 스위치들의 메모리 버퍼로 존재하게 되며, 아래와 같은 스케줄링 알고리즘들이 있다.

- FIFO (First-in First-out)

가장 단순하면서 기본적인 큐잉 방법이다. 처음 들어온 패킷이 가장 먼저 디큐잉 된다. 모든 패킷은 동일한 방법으로 제공되며 하나의 큐로 구현할 수 있다. 큐가 가득차게 되면 혼잡 제어를 위해 새로운 패킷이 폐기되게 된다.

FIFO는 패킷을 재 정렬 할 필요도 없으며 최대 지연 시간을 측정하기에 용이한 장점이 있다. 하지만, 서로 다른 클래스에 대하여 동일한 큐잉을 제공함으로써 차별화 서비스 네트워크에서는 적합하다고 볼 수 없다.

- RR(Round Robin)

큐를 정해진 순서대로 순환하면서 전송한다. 현재의 큐를 전송한 후에 다음 큐에 전송할 패킷이 있으면 차례대로 전송한다. 이 알고리즘은 간단하면서 큐안의 패킷들이 일정 시간에 전송될 것을 보장한다.

- PRI(Priority Queueing)

차별화 서비스를 위한 가장 간단한 방법을 제공한다. 여러개의 큐를 가지고 있으면서 우선 순위가 높은 큐를 먼저 처리한다.

들어오는 패킷의 우선순위에 따라 해당하는 우선 순위 큐에 대기 시킨다. 패킷에 대한 우선권으로 인해 차별화 서비스를 제공할 수 있지만, 패킷의 우선 순위가 낮은 패킷은 기아현상에 빠질 수 있다. 또한 특정한 패킷에 대해 차별화된 품질을 보장해주지는 못한다.

- WRR(Weighted Round Robin)

일련의 트래픽 플로우 집합에 대하여 가중치를 주어 차별화된 서비스를 제공하는 알고리즘이다. 단순한 FQ(Fair-Queueing)의 경우 각 큐에는 가중치 1을 받게 되는데 이는 각 큐가 서비스 될 때마다 1비트씩 전송되는 것을 의미한다. 이는 n개의 플로우에서 각 플로우는 $1/n$ 의 대역폭을 할당 받을 수 있다.

WRR에서는 EF, AF, BE 트래픽에 각각 5,3,2의 가중치를 주면 대역폭을 50%, 30%, 20%를 사용하게 되고 혼잡 상황에서도 대처할 수 있으며, BE 트래픽의 기아상태도 막을 수 있다.

각 스케줄링 방식의 장·단점을 표로 정리하면 표 6와 같다.

스케줄링	장점	단점
FIFO	<ul style="list-style-type: none"> 가장 간단하다. 패킷의 재 배열이 필요없다. 최대 지연 시간 계산이 용이하다 	<ul style="list-style-type: none"> 클래스별 Guaranteed Service를 지원하지 않는다. 네트워크의 혼잡상황에 지연이 심하다. 기아현상에 빠질 수 있다.
RR	<ul style="list-style-type: none"> 정해진 시간에 존재하는 패킷이 전송되는 것을 보장할 수 있다. 	<ul style="list-style-type: none"> FIFO와 같이 클래스별 대역폭 할당을 할 수 없다.
PRI	<ul style="list-style-type: none"> DiffServ를 지원할 수 있는 서비스 클래스를 사용할 수 있다. 	<ul style="list-style-type: none"> 우선순위(가중치)가 낮은 패킷은 기아현상 (starvation)에 빠질 수 있다.
WRR	<ul style="list-style-type: none"> 네트워크 자원을 클래스에 따라 효율적으로 분배 가능하다. Busy 플로우에 대해 대처할 수 있다. 	<ul style="list-style-type: none"> 복잡하다.

표 6 스케줄링 알고리즘의 비교

제 III 장 스케줄링 알고리즘의 시뮬레이션

본 논문에서는 Network Simulator 2(NS-2)를 이용하여 실험을 하였다. NS-2에서 제공하는 네트워크 토폴로지와 트래픽 모델을 이용하여 Ingress edge 라우터와 core 라우터를 Tcl 스크립트로 만들고 각각의 노드에 대역폭과 스케줄링 알고리즘을 적용하였다. 시뮬레이션에 사용한 알고리즘은 Round Robin(RR), Rriority(PRI), Weighted Round Robin(WRR)과 Weighted Interleaved Round Robin(WIRR) 스케줄링 알고리즘이다.

트래픽 셰이퍼로는 토큰 버킷을 공히 적용하고 MRED와 RED Active 큐 관리 알고리즘을 사용하였다.

제 1 절 *EF PHB Class* 와 *AF PHB Class*



2장에서 기술한대로 PHB의 표준 코드 포인트는 정의되어 있지만, 이에 따른 PHB 그룹의 정의와 그 수는 네트워크 정책에 의존한다. 그러므로, 시뮬레이터에서 제공하는 PHB 변경 만으로는 실제적인 EF 트래픽을 생성할 수 없었다.

PHB 파라미터 셋팅에서 physical queue의 값을 설정한 후 큐의 설정에서 폐기 우선권(Drop priority)을 가장 높게 설정하여 EF 클래스를 구현하고 AF 클래스는 상대적으로 낮은 값을 설정하여 구현하였다.

트래픽은 CBR(Constant Bit Rate)하고 전송 프로토콜은 UDP(User Datagram Protocol)를 사용하였다.

실험에 사용한 기본적인 시뮬레이션 모델은 그림 4과 같다.

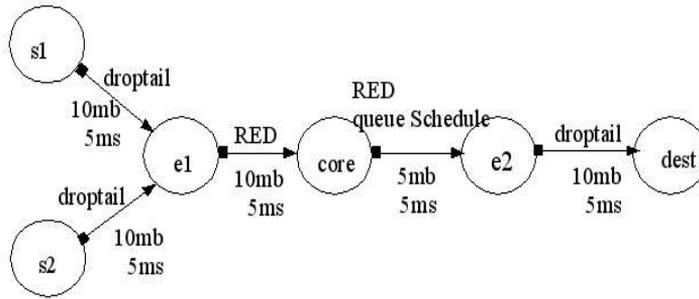


그림 4 시뮬레이션 모델

두개의 전송측(s0, s1)에서는 각각 AF 트래픽과 EF트래픽을 전송하고, core router(core)에서 edge router(e2)사이에 스케줄링 알고리즘을 적용하였다.

e1과 core, core와 e2사이에는 MRED 큐를 적용했다. core와 e2사이는 5MB의 대역폭을 가지는 혼잡(bottleneck) 구간으로 설정하고 이 구간의 패킷 패기율과 전송 패킷량을 측정하였다.

전송측 s1에서 수신측 dest에까지의 모든 경로에 EF 트래픽을 생성하기 위하여 PHB 코드포인트를 설정하고 해당 노드에는 두개의 큐를 두어 각각의 스케줄링 모드와 패킷 폐기 순위를 달리 하였다. 아래 <표 7> 에서의 예시와 같이 코드 포인트 10,11의 큐에는 엄격한 패킷 폐기 메커니즘을 가지는 PRI를 스케줄러로 하고 NS-2에서 제공하는 MRED의 4가지모드 중 모든 대역폭을 사용할 수 있는 DROP 으로 가장 높은 우선 순위를 할당하였다. AF PHB에는 MRED 모드중 가장 적은 우선권을 할당하는 RIO-C를 이용하여 이로써 우선 순위가 높은 EF PHB 클래스의 트래픽은 다른 클래스의 패킷들을 더 많이 폐기 시키면서 전송되었다.

```

# Core to Edge parameters
$qCE1 set numQueues_ 2           //두개의 큐를 생성
$qCE1 setNumPrec 2              //두개의 virtual queue
$qCE1 meanPktSize $packetSize   //RED 큐의 평균 길이
#Define PHB
$qCE1 addPHBEntry 10 0 0        //10,11 두개의 DSCP
$qCE1 addPHBEntry 11 0 1        // virtual queue
$qCE1 addPHBEntry 20 1 0
$qCE1 addPHBEntry 21 1 1
# codepoint 10,11 PRI scheduler
$qE1C setSchedulerMode PRI      //PRI 모드의 적용
# MRED mode is DROP
$qE1C setMREDMode DROP 0        //EF 큐에 DROP모드
$qE1C configQ 0 0 20 40 0.02    //$Q_num,$V_num,$min_Th, $maxTh,$maxP
$qE1C configQ 0 1 10 20 0.10
# codepoint 20,21 MRED mode is RIO-C
$qE1C setMREDMode RIO-C 1       //AF 큐
$qE1C configQ 1 0 20 40 0.02    //$Q_num,$V_num,$min_Th, $maxTh,$maxP
$qE1C configQ 1 1 10 20 0.10

# $Q_num : queue number
# $V_num : virtual queue number
# $min_Th : minimum threshold, dropped when the min_Th value is
reached
# $maxTh : maximum threshold
# $maxP : maximum dropping probability

```

표 7 EF 트래픽 설정

Core에서 Edge1으로 가는 경로에 두개의 큐를 정의하고, 표준안 데로 10,11을 EF DSCP로, 20, 21를 AF DSCP로 정하였다. 각각 하나씩의 가상큐(virtual queue)를 정의하였다. 여기서 해당 노드의 큐 파라미터 설정은 In-profile과 Out-profile에 관련된 수치를 지정할 수 있으며 디폴트로 정의된 값

을 그대로 적용하였다. 예를 들어 들어오는 패킷이 $\$min_Th$ 에 다다르게 되면 모든 패킷이 폐기된다.

두개의 송신측(s1,s2)에서 수신측(dest)로 가는 경로 중에 혼잡 구간(core to e2)에서의 패킷 폐기 상태를 그래프로 출력하였다.

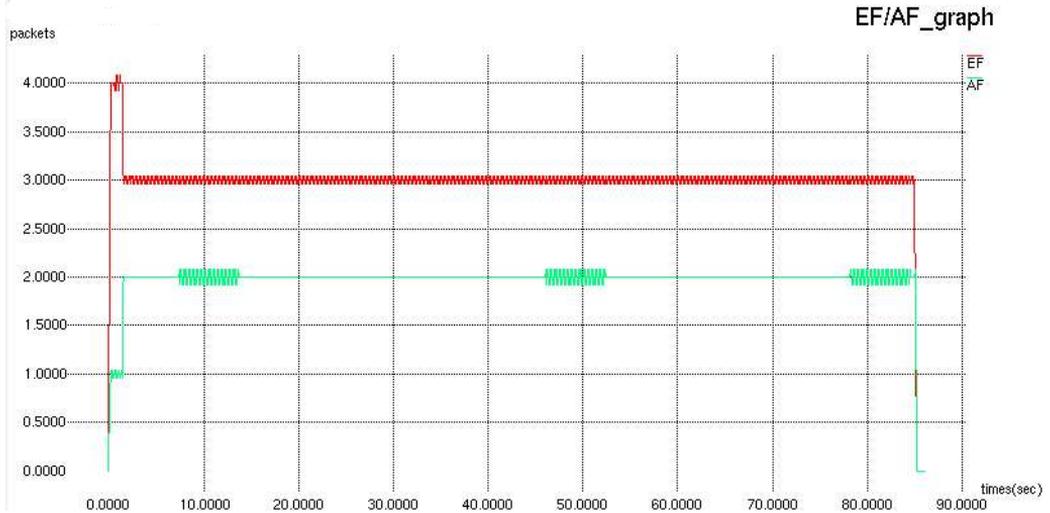
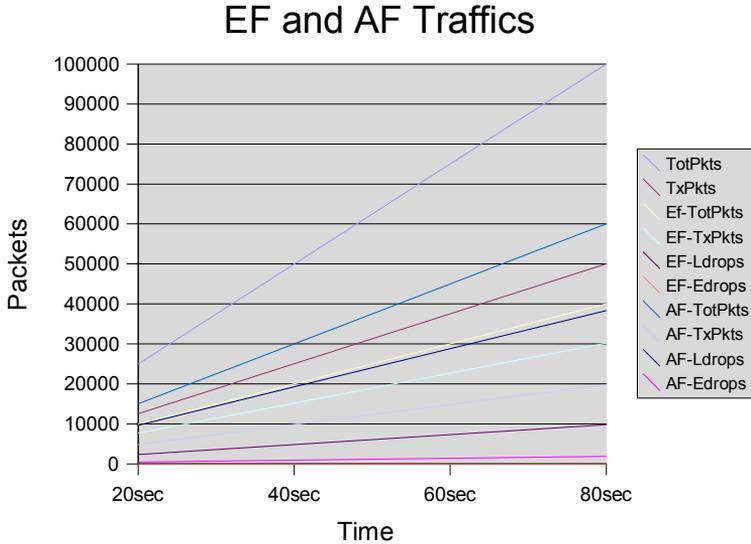


그림 5 EF, AF Traffic Graph

상단 그래프는 EF PHB 클래스 데이터를 하단의 그래프는 AF PHB 클래스의 그래프를 보여주고 있다. EF PHB 그래프는 높은 패킷 폐기정책에 따라 높은 전송량을 보여주고 있으며 하단의 AF PHB 그래프는 전송 속도는 적으나 비교적 안정적인 전송 상태를 보여주고 있다.

시간에 따른 전송량과 폐기량을 그래프로 나타내면 다음 그래프와 같다.

그림 6 EF와 AF 트래픽 전송/폐기량 그래프



아래 표는 전송 상황에 대한 결과값을 표로 나타낸 것이다.

	20sec	40sec	60sec	80sec
EF 총 패킷	9994	19995	29995	39995
EF전송 패킷	7678	15178	22677	30176
EF Link drop	2316	4817	7318	9819
EF RED Drop	0	0	0	0
AF 총 패킷	14992	29991	44991	59991
AF 전송 패킷	4844	9847	14844	19822
AF Link Drop	9680	19238	28771	38304

표 8 EF/AF 트래픽 전송 수치

도출된 데이터에 의하면 EF 클래스 트래픽은 AF 트래픽에 비하여 지연을 통해 전송하고자 큐에 도착한 패킷은 39,995와 59,991로 많으나 폐기된 패킷의 수도 AF트래픽은 Link over flow에 의한 38,304와 RED 알고리즘에 의한 1,843으로 많은 것을 볼 수 있고 EF 트래픽은 RED 폐기가 아예 없는 것으로 큐에 머무는 시간이 아주 작았다는 것을 알 수 있다.

제 2 절 스케줄링 기법의 비교

앞서 기술한 테스트 환경에서 동일한 조건하에서 스케줄링 메커니즘만 교체하면서 PHB 클래스가 혼재한 상황에서 혼잡 구간을 다시 측정하였다.

시뮬레이션에 사용한 알고리즘은 Round Robin(RR), Rriority(PRI), Weighted Round Robin(WRR)과 Weighted Interleaved Round Robin (WIRR) 스케줄링 알고리즘이다.

NS-2에서는 노드에 큐를 설정하게 되면 기본적인 스케줄링 모드는 RR모드로 하게 된다.

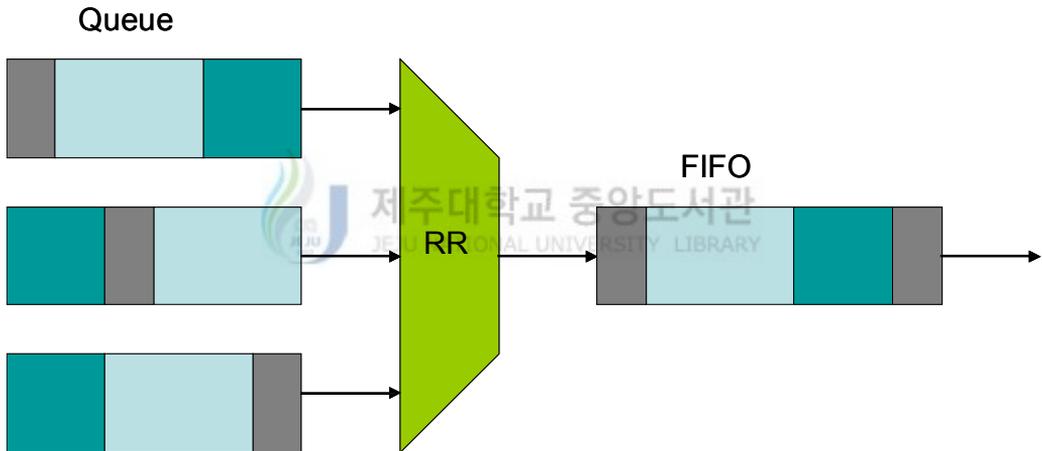


그림 7 RR scheduling

PRI모드는 EF PHB 클래스에 우선권(priority)를 가장 높이 설정하는 것으로 실험하였다.

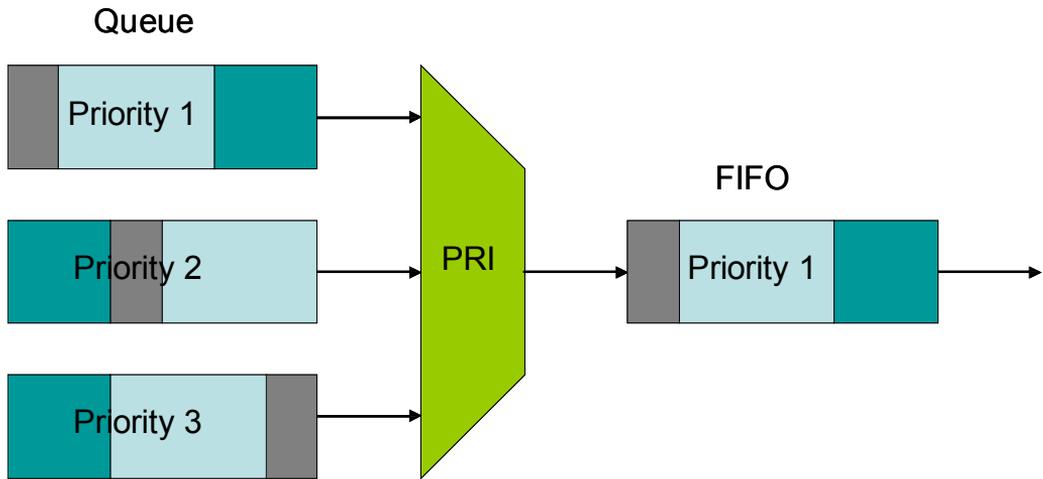


그림 8 PRI scheduling

```
#set PRI scheduler
$dsredq setSchedulerMode PRI
#set maximum bandwidth 5MB to queue 0
$dsredq addQueueRate 0 5000000
```

표 9 PRI Scheduling code

WRR 모드는 각 큐에 가중치를 설정할 수 있다. <표 9>에서 큐 0과 1에 각각 7과 3의 가중치를 적용하고 있다.

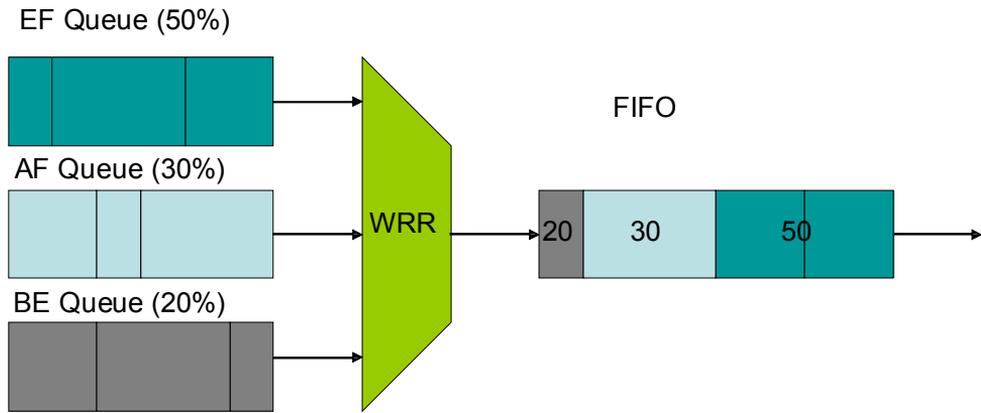


그림 9 WRR scheduling

```
#set WRR scheduler
$qCE2 setSchedulerMode WRR
# set priority 7 to queue 0
$qCE2 addQueueWeights 0 7
# set priority 3 to queue 1
$qCE2 addQueueWeights 1 3
```

표 10 WRR Scheduling code

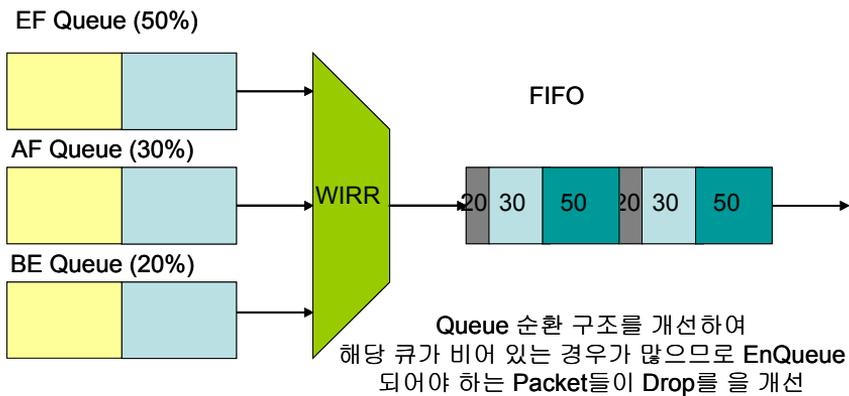


그림 10 WIRR scheduling

측정한 결과 종합적으로 스케줄링 알고리즘에 따라 <표 10>과 같은 폐기된 패킷 수치가 산출되었다. 클래스와 관계없이 총 전송량과 폐기된 패킷량을 나타낸 것이다. s1과 s2에서 80000패킷을 전송했고 가중치는 AF와 EF클래스에 각각 3:7비율로 주었을 때 산출된 값이다.

	TxPkts	Ldrops	Edrops
RR	50027	23627	6335
PRI	50030	23564	6395
WRR	50026	24222	5741
WIRR	50024	18175	5773

표 11 스케줄링 알고리즘 별 종합 결과값

Assured forward(AF) Class 트래픽의 각 스케줄링 알고리즘을 도입한 패킷 폐기량은 <그림 6>과 같다.

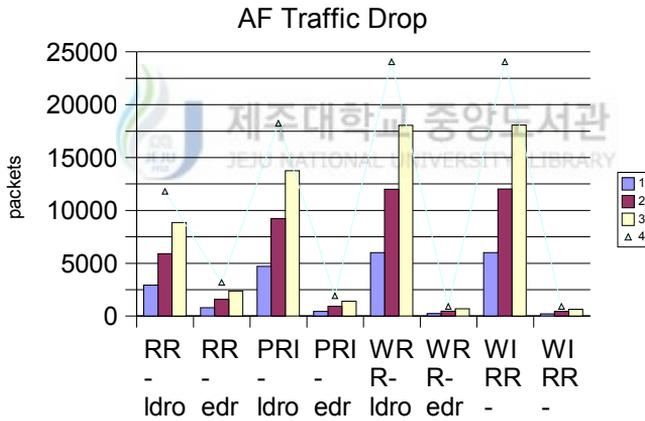


그림 11 AF PHB Class Traffic

Expedited Forward(EF) 트래픽에 대한 그래프는 <그림 7>과 같이 표시되었다.

EF Traffic Drop

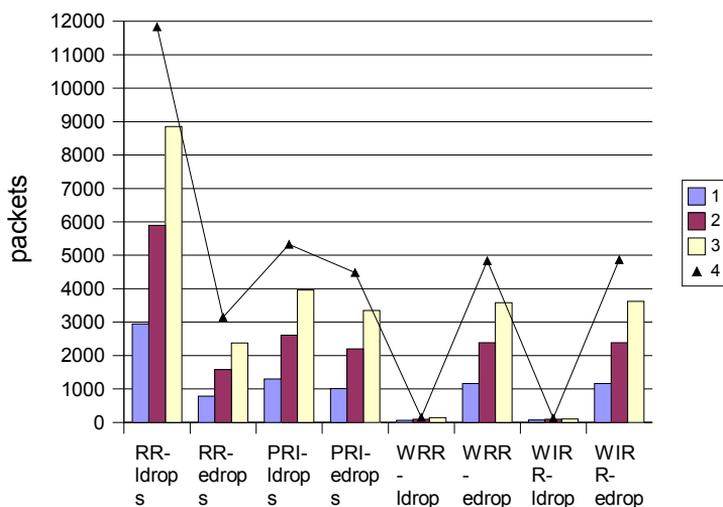


그림 12 EF Class Traffic Drop

링크 오버플로어에 의한 폐기량과 RED큐에 의한 폐기량을 X축에서 사용한 스케줄링 알고리즘 이름과 함께 표시하였다.

ldrops는 core와 e2간의 혼잡(bottleneck) 구간에서 링크 오버플로어에 의해 폐기된 패킷을 나타내며, edrops는 RED 큐에 의해 사전 폐기(early dropping)된 패킷을 나타낸다. 즉 ldrops와 edrop를 합친 값은 해당 클래스 트래픽의 총 패킷 폐기량이 된다.

시뮬레이션 하는 동안 시간 차를 두어 4번의 결과값을 색깔로 표시하고 선으로 표시한 값은 최고치의 최종 폐기된 총량을 나타낸다.

RoundRobin(RR) 스케줄링은 두개의 서로 다른 큐에 AF 클래스의 패킷과 EF 클래스의 패킷이 있더라도 큐를 순환하면서 차례대로 전송하므로 AF 클래스 이건 EF클래스이건 서로 비슷한 비율의 패킷 폐기를 하는 것을 볼 수 있다. 실제로 폐기된 패킷의 총량은 14,981 패킷으로 두 클래스가 같았다. 즉 일정한 수의 EF와 AF 클래스의 트래픽 폐기량을 보인다.

	20sec	40sec	60sec	80sec
EF-drops	2944	5897	8849	11832
AF-drops	2936	5900	8850	11795

표 12 RR 스케줄링의 폐기된 패킷량

Priority(PRI) 스케줄링은 가중치(priority)에 의해 패킷을 전송한다. 그러므로 EF 클래스 트래픽이 가장 순위가 높으므로 가장 빨리 서비스되고, AF클래스, BE 트래픽의 순서로 전송된다. 결과 값은 AF 클래스 트래픽이 20,149, EF 클래스는 9,810으로 결과가 나왔다. EF 클래스가 시간에 따라 증가할 수록 급격하게 AF 클래스 트래픽의 폐기가 늘어나는 것을 볼 수 있다.

	20sec	40sec	60sec	80sec
EF-drops	2309	4807	7307	9810
AF-drops	5149	10147	15150	20149

표 13 PRI 스케줄링의 폐기된 패킷량

Weighted Round Robin(WRR) 스케줄링은 core에서 e2의 링크에 두개의 큐를 두어 WRR 스케줄러를 적용하고 가중치를 다르게 두어 시뮬레이션 하였다. 가중치의 변경에 따라 EF traffic이 전혀 폐기되지 않을 수도 있고 AF클래스에 좀더 많은 가중치를 부여하여 폐기율을 조절할 수 있었다.

EF 클래스 7, AF 클래스 3으로 가중치를 설정한 후 산출된 값은 <표 13>과 같다.

	20sec	40sec	60sec	80sec
EF-drops	1233	2474	3728	4989
AF-drops	6221	12475	18721	24974

표 14 WRR 스케줄링 폐기 패킷량

Weighted Interleave Round Robin(WIRR) 스케줄링은 WRR의 큐 순환을 개선한 것으로 WRR이 하나의 큐를 전송한계치에 이를때까지 전송하거나 큐가 완전히 비워질 때까지 전송한 후에 다음 큐로 이동하는데 반해 WIRR은 하나의 큐를 정해진 수치만큼만 전송하고 다른 큐로 이동한다. 이때 들어오는 패킷은

새로운 큐가 아닌 같은 클래스의 큐에 계속해서 쌓일 수 있고 그 큐는 계속해서 같은 가중치를 가지고 있을 수 있다. WRR과 동일한 설정 값으로 산출된 값은 <표 13>과 같다.

	20sec	40sec	60sec	80sec
EF-drops	1235	2478	3731	4993
AF-drops	6220	12473	18721	24972

표 15 WRR 스케줄링 폐기 패킷량

두 가중치 기반 스케줄링 기법에서는 코드 포인트에 따른 가중치를 변경 설정 함으로써 폐기되는 폐킷량이 변화하는 것을 볼 수 있으며, 하나의 큐에 가중치를 전부 할당하면 PRI 스케줄링과 같은 결과가 나왔다.



제 IV 장 결론

QoS 제공에 있어서 가장 중요한 것은 네트워크 자원, 즉 대역폭의 관리가 가장 중요하다. 본 논문에서는 DiffServ를 통해 QoS를 제공함에 있어서 여러가지 클래스의 데이터가 혼재한 상황에서 여러 스케줄링 기법의 특성에 대하여 비교 분석하였다.

DiffServ에서는 PHB에 의한 자원할당과 패킷 폐기에 관한 일련의 작업을 수행한다. 자원할당과 관련하여 본 논문에서 EF PHB 클래스와 AF PHB 클래스를 대상으로 시뮬레이터를 이용하여 큐잉과 스케줄링 기법의 조합을 통해 이를 구현하였고, 두 클래스가 동시에 전송되는 상황에서 트래픽 전송상황을 비교 분석하였다. 같은 상태에서 스케줄링 알고리즘의 파라미터를 조절하면서 어떠한 스케줄링 기법이 DiffServ에서 유용한가를 연구하였다.

EF 트래픽은 큐에 머무는 시간이 가장 짧거나 아예 없어야 하므로 MRED (Multi-level RED) 큐의 4가지 모드 중 Drop 모드를 적용하고, 동시 전송되는 AF 트래픽에 대해서는 MRED 모드 중 RIO-C (RED In and Out - Coupled) 모드를 적용하여 패킷 폐기에 대한 엄격함을 차별화 하였다. 스케줄링은 PRI 모드로 단순하면서 EF와 AF 트래픽의 전송 특성을 가장 잘 보여줄 수 있도록 하였다. 실험 결과에 따르면, MRED의 DROP을 적용한 EF 트래픽은 총 39,995 패킷의 전송 시도 중에 9,819의 링크 오버플로에 의한 폐기만이 나왔으며, MRED 모드 중 가장 느슨한 폐기 폐기를 보여주는 RIO-C를 적용한 AF 트래픽 전송 상태에서는 지연을 통해 총 59,991개의 패킷 전송시도 중에 링크 오버플로에 의해 38,304 패킷과 RED 알고리즘에 의한 폐기가 1,843으로 도출되었다. 이로써 EF 트래픽은 가장 짧은 큐 대기 시간을 보이면서 RED에 의한 폐기가 0으로 EF 트래픽의 특성은 MRED 모드 중 DROP을 이용하였을때라는 것을 보여주고 있다. 또한 AF 트래픽은 많은 전송 요구와 지연에 의해 38,304개의 패킷을 폐기하면서도 총 전송 요구 패킷의 수는 EF(39,995)에 비해 59,991로 많았다는 것을 알 수 있었다. 이는 가장 느슨한 즉, 폐기 마킹을 가장 느슨하게 하는 MRED 모드 중 RIO-C를 통해서 AF 트래픽을 구현했다.

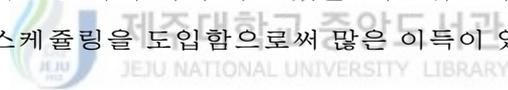
수행한 시뮬레이션에서 RR 스케줄링을 제외한 다른 스케줄링 기법들은 모두 설정한 우선 순위 값에 기인하여 결과값을 도출하므로 여러가지 클래스의 트래픽이 혼재하는 경우 네트워크에서 적절한 매커니즘의 선택이 QoS제공에 매우 중요한 요인이 된다.

클래스에 관계없이 동등한 폐기량을 보여준 RR스케줄링 기법이나 우선권을 가진 패킷을 최우선 전송하는 PRI 스케줄링 방식은 RR 알고리즘은 차별화된 서비스를 할 수 없다는 점에서, PRI 알고리즘은 QoS를 제공할 수는 있으나 EF 클래스가 많을 경우에도 AF, BE 클래스 트래픽이 전송되어야 한다는 점에서 두 스케줄링 기법은 DiffServ모델에는 적합하지 않다.

WRR 방식은 클래스 별로 가중치에 따라 EF트래픽의 전송을 보장해 주면서도 AF 트래픽의 전송도 일정 부분 보장하는 것을 보였다.

이로써 IETF에서 정의한 EF와 AF PHB 트래픽들은 큐와 스케줄링에 의해 정의한 의미대로 전송 품질을 보장할 수 있으며, 실험한 큐 스케줄링 방식들에 대해서는 DiffServ에 가장 적합한 알고리즘은 WRR이나 WIRR과 같은 가중치 기반 방식이라고 할 수 있다.

갈 수록 방대해지고 고속화 되어 지고 있는 네트워크에서 QoS를 도입하는데 있어 가중치 기반 스케줄링을 도입함으로써 많은 이득이 있을 것이라 기대한다.



· 참고 문헌

[1] Braden, D. Clark, S. Shenker “Integrated Services in the Internet Architecture: an Overview.” June 1994.

[2].D. Grossman.”New Terminology and Clarifications for Diffserv.” RFC 3260 April 2002.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss. “An Architecture for Differentiated Service.” December RFC 2475 1998.

[4] Ramakrishnan, S. Shenker, J. Wroclawski, L. Zhang.”Recommendations on Queue Management and Congestion Avoidance in the Internet.” RFC 2309 April 1998.

[5] Zheng Wang “Internet QoS -Architectures and Mechanisms for Quality of Service“ Morgan Kaufmann 2001

[6] “NS-2 Simulator 2” <http://www.isi.edu/nsnam/ns/>

[7] Sergio Androozzi “DiffServ Simulations Using the Network Simulator” <http://www.cnaf.infn.it/~sergio/diffserv/ns/features.htm>

[8] IETF ”Differentiated Services (DiffServ)”<http://www.ietf.org/html.charters/diffserv-charter.html>

[9] Cisco systems.”Implementing DiffServ for End-to-End Quality of Service Overview”http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/fqos_c/fqcprt7/qcfdfsrv.htm June. 2003

[10] V. Jacobson, K. Nichols, K. Poduri. “An Expedited Forwarding PHB.” RFC 2598 June 1999.

[11] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, L. Zhang. “Recommendations on Queue Management and Congestion Avoidance

in the Internet.” RFC 2309 April 1998

[12] Qbone Signaling Design Team “Bandwidth Broker Architecture”

<http://qbone.internet2.edu/bb/>



· 영문 요약 **Summary**

This paper describes a number of basic methods for scheduling algorithm for DiffServ(Differentiated Service) model. Several scheduling disciplines are analyzed, like RR(Round Robin), PRI(Priority), WRR(Weighted Round Robin) and WIRR(Weighted Interleave Round Robin).Advantages and disadvantages for each methods are outlined.

Scheduling dicipline used in NS-2(Network Simulator 2) is described.

In DiffServ model, resource are allocated on a per-class basis. AF (assuerd fowarding) class and EF(Expedited forwarding) class have been standardized.

According to result, RR scheduling dicipline is no available make a distinction between AF and EF PHBs class traffic. This dicipline is unworthy of DiffServ.

PRI scheduling dicipline can support priority of PHPs class but this dicipline is possible die of starvation about low priority PHBs class traffic.

WRR scheduling dicipline is support priority by PHBs codepoint. This dicipline based on weight also. Therefore Weighted based dicipline can support many class of traffic without starvation of low priority class traffic.. And it meet the situation a incoming busy traffic

We can conclude that based on Weighted fair scheduling dicipline is best algorithm for DiffServ.

감사의 글

늦게나마 학문의 길을 다시 열어주신 송왕철 교수님과 더 나은 논문을 위해 질책과 격려를 해주신 안기중 교수님, 김도현 교수님께 진심으로 깊은 감사를 드립니다.

함께 고민해 주고 작은 일까지 일일이 도와준 김강석 학형과 이경진, 이정운 후배께도 감사 드립니다. 살아온 어느 2년동안보다도 더 열심히 같이 해주었습니다.

끝으로 공부할 수 있게 도와준 아내와 하느님의 가장 큰 선물인 두딸, 그리고 그 이름만으로 포근함을 느낄 수 있는 부모님께 감사의말씀을 전합니다. 그 정성에 비하여 너무 작은 결실이나마 이 논문을 바칩니다.

2004년 12



제주대학교 중앙도서관
JEJU NATIONAL UNIVERSITY LIBRARY

허 지 완 올림