동적 노드 설정을 이용한 인터넷상의 QoS 제공

(Providing QoS in the Internet using Dynamic Node Setting Method)

김 중규*
(Jung-Gyu Kim)

요 약 최근 인터넷에서는 인터넷 방송, 영상회의, VoIP 등 QoS 보장을 요구하는 새로운 응용 서비스들의 출현으로 서비스 품질 제공은 차세대 인터넷에서 가장 주요한 과제의 하나다. 하지만 현재의 인터넷은 모든 패킷을 동일하게 전달하는 BE(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. 본 연구에서는 IP QoS와 관련하여 네트워크에서 QoS 지원을 위한 고려사항 등을 살펴보고, DiffServ 모델에서 QoS를 지원할 수 있는 동적노드 설정방법을 제시하고 이의 성능을 분석한다.

Abstract Historically, IP-based internets have been able to provide a simple best-effort delivery service to all applications they carry. Best effort treats all packets equally, with no service level, packet loss, and delay. The best-effort style has generally worked fine. But the needs of users have changed. The want to use the new real-time, multimedia, and multicasting applications. Thus, there is a strong need to be able to support a variety of traffic with a variety of quality-of-service requirements, within the TCP/IP architecture. This paper propose a framework that offers QoS in a DS domain using dynamic node setting method.

1. 서 론

최근 인터넷에서는 인터넷 방송, 화상회의, VoIP(Voice over IP) 등 서비스 품질(QoS : Quality of Service) 보장을 요구하는 새로운 멀티미디어 응용 서비스들의 출현함으로서, IP QoS의 문제는 차세대 인터넷에서 가장 주요한 과제의 하나로 등장하고 있다. 하지만 현재의 인터넷은 모든 패킷을 동일하게 전달하는 BE(Best Effort) 서비스만을 제공하고 있기 때문에, 데이터 전송중에 발생하는 패킷의 손실 또는 지연 등에 대한 요구 사항을 보장해 주지 못하고 있다. 따라서 인터넷에서 서비스 품질을 보장해 주기 위해서는 현재와는 다른 새로운 서비스 모델을 필요로 한다[1][8].

실시간 응용 서비스가 요구하는 QoS를 지원하기 위해 새로운 서비스 모델에 기반을 둔 IP 패킷 전달 방식에 대한 연구가 최근 수년간 IETF 작업그룹에서 진행되어 왔다[3]. IETF에서 제안한 방식 중 대표적인 것이 통합 서비스(IntServ : Integrated Service)와 차등 서비스(DiffServ : Differentiated Service)이다[2][4]. IntServ 모델은 실시간 응용 서비스에서 발생되는 패킷의 흐름을 단위로 하여 패킷을 전달한다. 즉 신호 프로토콜인 RSVP(Resource Reservation) 프로토콜을 이용하여 연결 수락 제어와 자원예약을 수행하여 패킷의 전달 지연을 보장해 주는 모델이다. 하지만 이 모델은 각 패킷 흐름에 대한 상태 정보를 망의 라우터가 유지하고 있어야 하기 때문에 백본망에서는 현실적으로 수용하기에는 어려움이 있다.

이에 따라 확장성의 문제를 갖고 있는 IntServ 모델의

*대구대학교 정보통신공학부

한계를 극복하고 인터넷 백본 망에서 적용할 수 있는 서비스 모델로서 DiffServ 모델이 90년대 후반부터 IETF 작업그룹에서 활발히 논의되기 시작되어 빠른 속도로 구조 및 관련 표준안이 개발되고 있다. DiffServ 모델은 흐름 단위로 QoS를 보장하지 않고 흐름들의 집합을 단위로 서비스를 차별화 함으로써 훨씬 간단하게 구현할 수 있어, 대규모 망에도 적용 가능한 모델이다.

한편, DiffServ 구조는 전 도메인이나 상호연결된 도메인을 통하여 QoS를 제고하기 위하여, 일관된 자원 공급을 필요로 한다. 이를 위해서는 서비스 품질의 하락을 피하면서 자원 이용도를 증가시키야 하므로 에지노드에서 트래픽 컨디셔너와 네트워크 코어의 메카니즘은 빈번하게 재구성되어야 한다. 장비가 서로 다르거나 복잡한 네트워크 토폴로지를 가지는 경우, 재구성 과정은 상당히 복잡한 작업이며, 이것으로 인해 심각한 불안정성이나 QoS 하락이 발생할 수도 있다[11][15].

따라서 이 같은 문제를 해결하고 QoS 제어를 원활히 하기 위하여 전 도메인상의 관리 동작을 사용하는 것이다. 정책기반 관리가 DiffServ와 IntServ 구조에서 QoS 보장을 위한 기반 시설로서 제안되었다. 본 연구에서는 정책기반 관리에 기초하여 DiffServ 도메인내에서 동적인 노드 설정방법으로 퍼지 제어기를 제안하였다. 퍼지 제어기는 데이터 흐름 추정의 불확실성과 부정확성에 대응하기 위해 사용되는데, 이 경우, 퍼지 제어기는 기존의 디지털 제어기와 비교하여 시스템의 복잡성을 낮추면서 부정확한 변수를 처리하는 데 장점을 가진다. 한편 이의 성능을 분석하기 위해 시뮬레이션을 이용하여, 음성 흐름의 지연과 지연변동 그리고 손실율이 다른 지연에 민감하기 않은 트래픽과 함께 사용하여 제어기의 유효성을 입증하였다.

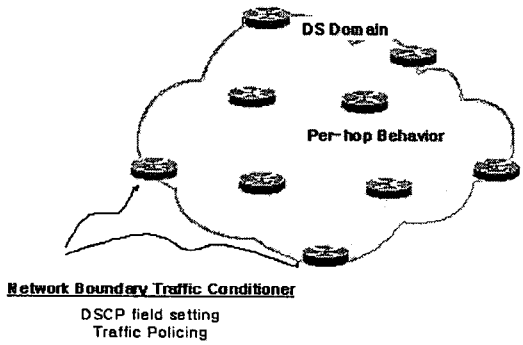
2. DiffServ 구조와 정책기반 관리

2.1 차등서비스

1998년 IETF DiffServ 작업그룹에서 제안된 이 모델은 QoS 레벨로 보면 오늘날 인터넷과 같은 BE 서비스 제공과 IntServ 모델에 의한 QoS를 보장하는 모델 사이의 중간적인 위치를 차지한다. 하지만 이 모델은 앞에서 언급한 IntServ 모델의 문제점들을 극복하기 위한 모델로 제안된 구조이다[3].

이 모델에서는 우선적으로 QoS를 몇 개의 클래스로 분류하여 이 분류된 클래스에 따라 서비스를 보장하도록 하는 것이다. 이에 따라 우선적으로 IP 헤더의 특정 필드(IPv4 : ToS, IPv6 : Traffic Class Field)에 표시를 이용하여 차등 서비스(DS : Differentiated Service)를 설정하게 된다. 이렇게 특정 값으로 설정된 DS 값들에 의해 적절한 포워딩(PHB : Per-hop Behavior)을 수행하도록 하는 구조이다[12].

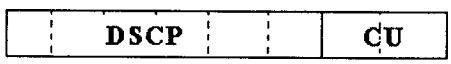
<그림 1>은 DiffServ의 개요도로서, DS 도메인 내에서는 각각 경계 라우터에서 설정된 차등 서비스 코드 포인트(DSCP : Differentiated Services Codepoint) 값에 따른 PHB 처리를 하게되며, 경계 라우터에서는 DSCP 설정 및 트래픽 폴라이싱을 수행하게 된다. 즉 DiffServ 구조에서는 DS 도메인의 경계라우터에서 IP 헤더의 ToS에 DSCP를 설정하여 PHB를 할당하고, 네트워크 내에서는 이 분류된 PHB에 따라 자원 할당, 패킷 폴라이싱, 스케줄링 등 PHB 프로세스를 수행하도록 하는 구조이다. 즉 경계라우터와 코어 라우터의 역할을 분리시켜 통합 서비스 구조의 확장성 문제를 해결하고자 한 모델이다[11].



<그림 1> 차등서비스 개요

이 모델에서 제시한 IP Header에 ToS 필드를 DS 필드로 사용한 경우를 <그림 2>에 나타내었다. 이 DSCP의 설정값에 따라 해당 패킷의 요구되는 QoS를 나타내게 되는데 크게 다음과 같이 분류되고 있다[3].

① Default DSCP[000 000]
② Class Selector DSCP
③ Expedited Forwarding(EF) PHB
④ Assured Forwarding(AF) PHB

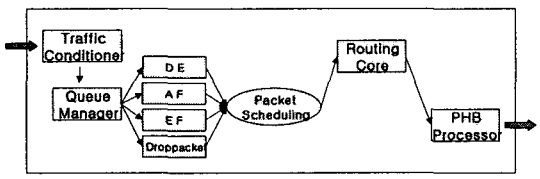


DSCP : differentiated services codepoint
CU : current unused

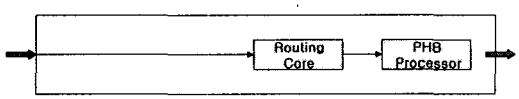
<그림 2> DiffServ 모델을 위한 ToS

2.2 라우터 기능

<그림 3>에 나타낸 트래픽 컨디셔너는 네트워크의 경계라우터에서 수행되는 다양한 QoS 함수들로 구현되어진다. 즉 적절한 DSCP를 설정하고 들어오는 트래픽을 감시하여 적절한지 등을 감시하는 기능을 수행하게 된다. 이러한 DiffServ 모델에서의 경계라우터의 구조는 다음 그림과 같다. DiffServ 구조에서는 트래픽에 대한 그룹화 과정을 통하여 패킷들을 분류하고 이에 따라 차별화 시켜 QoS를 보장하는 모델로서, IntServ의 확장성 문제를 해결하기 위하여 경계 라우터와 코어 라우터에서 담당하는 역할을 구분하였다. 즉 상대적으로 트래픽이 적은 경계 라우터에서 <그림 3a>와 같은 복잡한 기능을 수행하고, 코어 라우터에서는 <그림 3b>와 같이 단순히 포워딩만 수행하도록 하여 IntServ의 문제점을 해결하려는 모델이다[3].



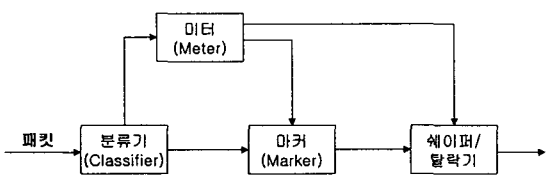
(a) 경계 라우터 기능도



(b) 코어 라우터 기능도

<그림 3> 경계 라우터와 코어 라우터의 개념적 구조

경계 라우터에서 네트워크에 진입하는 트래픽을 조절하기 위하여 <그림 4>에서 보여주는 바와 같으며, 분류기, 미터, 마커, 쉐이퍼와 같은 기능들이 에지 부분에 할당된다. 이 4가지 요소를 반드시 포함할 필요는 없다.



<그림 4> 컨디셔너 기능블럭

컨디셔너(Conditioner)의 주요한 기능은 다음과 같다.

1) 분류기(Classifier) : 차별화된 서비스를 지원하기 위해서는 각각의 패킷을 보고 그것이 어디에 속하는 지를 확인할 수 있어야 한다. 분류기에서 사용하는 방법으로는 두가지가 있다. 하나는 BA(Behavior Aggregate) 분류기로 DS 필드만을 기초로 패킷을 분류한다. 다른 방법은 MF(Multi-Field) 분류기로 근원지 주소, 목적지 주소, DS 필드, 프로토콜 ID, 근원지/목적지 포트(TCP, UDP)와 같은 전송 계층 헤더 필드등의 패킷 헤더에 있는 여러 필드의 조합으로 분류를 한다.

2) 미터(Meter) : 트래픽 미터는 TCA(Traffic Conditioning Agreement)에서 명시된 트래픽 프로파일에 대해 분류기에 의해 선택된 패킷 스트림의 시간적 특성을 측정한다. 이 기능 블록은 패킷이 분류기에 의해 분류된 다음 그 패킷이 소비하는 자원ㅇ을 알기 위한 블록이다. 미터는 어떤 트래픽 흐름이 자원의 소비 제한 내에 있는지(In-profile) 아니면 초과하는지(Out-of-profile)를 판단하기 위하여 각 패킷에 대하여 검사하고, 그리고 특정한 행동을 개시하기 위하여 다른 조절 기능에 상태 정보를 전달한다. 패킷을 측정하는 파라미터로는 흐름율(Flow Rate)과 버스트 크기가 있다. 이 트래픽 측정에는 토큰 버킷(Token Bucket) 방식이 사용되며 이 결과를 바탕으로 Shaping 또는 Dropping을 수행하게 된다.

3) 마커(Marker) : 분류기 단계를 통해 분류된 패킷들을 해당하는 DSCP(DS Code Point) 값으로 IP 헤더의 해당 필드에 마킹하게 된다. 이때에는 DSCP 값을 새로이 표시하거나 이미 설정되어 있어도 이들 값을 새로이 표시할 수 있으며, 이를 'Packet Coloring'이라고도 한다.

4) 쉐이퍼(Shaper) : 트래픽 프로파일에 따라 스트림을 보내기 위해 스트림에서 약간의 패킷이나 혹은 모든 패킷을 지연시킨다. 쉐이퍼는 일반적으로 유한 크기 버퍼를 가지며, 만약 지연된 패킷을 잡아두기에 충분한 버퍼공간이 없다면 패킷은 탈락될 것이다. 이와 같이 버스트 데이터를 처리하거나 트래픽의 속도를 조절하는 것을 쉐이핑이라고 하며, 버스트로 도착하는 트래픽을 평활화하기 위해 사용된다.

5) 탈락기(Dropper) : 탈락기는 트래픽 프로파일에 따라 스트림을 보내기 위해 트래픽 스트림에서 약간의 패킷 혹은 모든 패킷을 탈락시킨다. 탈락기는 패킷이 없거나 적은 양의 패킷을 위해 쉐이퍼 버퍼 크기를 설정함으로서 쉐이퍼의 특별한 경우로 구현될 수 있다. 탈락기는 단시간의 버스트를 허용하면서 장시간의 폭주를 막기 위하여 능동적인 대기관리 알고리즘등이 사용된다. 미터/마커, 쉐이퍼, 탈락기의 기능을 종합하여 폴리싱이라고 한다. 트래픽 조절기는 대개 DS 입구/출구 경계 노드내에 위치하지만, DS 영역의 내부 노드나 non-DS 영역

내에 위치할 수도 있다.

2.3 정책기반 QoS 관리

위에서 언급된 트래픽 관리 기술들은 각 네트워크의 요소들에서 개별적으로 제공되는 기능들이며, QoS 보장 기술은 인터넷에서 QoS를 제공하기 위해 해당 응용 서비스 마다 자원을 예약하거나 할당하는 기능을 수행하며, 이 경우 일관성 있고 효율적인 종단간의 QoS 보장을 위해서는 네트워크 도메인 내부와 도메인간 전반적인 입장에서 자원관리를 할 필요가 있는데, 정책기반의 관리를 통해서 이 역할을 수행할 수 있다. 먼저 네트워크 도메인 내부에서는 도메인 특성에 맞는 정책에 따라 QoS 관리를 수행하며 도메인 간에는 서로의 협상을 통해서 종단간의 QoS를 보장하여야 한다[3][14].

정책 기반의 QoS 관리는 크게 정책 편집, 정책 충돌 방지, 정책 생성, 정책 분배, 정책 진화 기능으로 구성되어 있다. 정책 편집 기능을 통해서 네트워크 관리자는 네트워크와 사용자의 정책을 생성한다. 생성된 정책은 일단 기존의 정책과 충돌이 일어나는가를 정책 충돌 방지 기능을 통해서 검토한다. 그 후 정책 생성 기능은 네트워크 장비가 이해할 수 있는 형태로 정책을 변경 생성한다. 변경 생성된 정책 정보는 필요한 네트워크 장비에 분배되며 각 장비는 받은 정책을 적용하여 트래픽을 제어한다. 일단 분배된 정책은 네트워크의 상태나 특정 일시 등과 같은 영향으로 변경을 요하게 되기도 하는데 정책 진화 기능을 이러한 변화를 감지하는데 필요한 기능을 한다[10][13].

정책 기반 관리 기술은 단순히 QoS 관리뿐만 아니라 보안, 경로 제어 등을 위한 용도로 광범위하게 사용되므로, 정책 기반 관리 기술은 확장성 있는 구조를 가지는 것이 매우 중요하다.

2.3.1 정책기반 QoS 관리 시스템 구조

정책 기반의 QoS 관리를 위해서 IETF, DMTF (Distributed Management Task Force)[9] 등에서는 공통된 시스템 구조를 정의하기 위해 상호 협력중에 있다. 정책 기반 QoS 관리 시스템은 크게 정책 툴, 정책 저장 및 검색을 위한 디렉토리 시스템, 정책 결정을 책임지는 정책 결정 포인트(PDP : Policy Decision Points)와 정책 실행 포인트(PEP : Policy Enforcement Points), PDP Proxy로 구성되어 있다.

PDP와 PEP는 통합된 시스템이거나 분산된 형태로 존재할 수 있으며, 분산 시스템이 확장성 면에서 유리하다. PDP, PEP 및 디렉토리 간에는 통신을 위한 프로토콜이 필요한데, 현재 거론되고 있는 대표적인 프로토콜로 COPS(Common Open Policy Service)와 LDAP(Light Weight Directory Access Protocol)이 있다. COPS는 PDP와 PEP 사이에 정책 정보를 전달하기 위해 필요하며, LDAP은 PDP나 PEP가 정책 정보를 저장, 검색, 획득하는데 필요한 프로토콜이다[9][10].

2.3.2 정책기반 QoS 관리 시스템 프로토콜

정책프로토콜은 각 라우팅 시스템에서 사용자 데이터 전달 요구를 수신하였을 때 사용자에 적용할 수 있는 정책에 따라 사용자를 받아들이거나 거절하는 기능을 수행하기 위하여 라우터와 서버간에 동작하는 프로토콜이다.

이러한 정책 프로토콜에 대하여 다음과 같은 사항들이 요구되어진다.

- 신뢰성
- 작은 지연
- PEP-초기화, 양방향 처리 기능
- 비동기 통보 기능
- 멀티캐스트 그룹 처리 기능
- QoS 명세 기능

기존에 정책 프로토콜로 이용되어 왔던 프로토콜로는 RADIUS, LDAP, SNMP(Simple Network Management Protocol) 등이 있으나 이들은 신뢰성이 부족하고 서버가 시작하는 메시지 전달 기능 등을 제공하지 못하고 있다.

서버에서 정책프로토콜 기능을 가지는 것을 고려해본다면 현재 정책 프로토콜에 대한 완전한 표준 규격이 나오지 않은 상태이기 때문에 COPS나 DIAMETER와 같은 프로토콜의 사전 도입 방안보다는 기존 프로토콜을 이용하는 단계적 접근 및 통합적인 접근 방안이 필요하다. 즉, 초기에는 RADIUS, SNMP, CLI(Common Line Interface) 등과 같이 기존 라우터들이 이용하는 프로토콜을 이용하여 정책 프로토콜로 이용하고 추후 정책 프로토콜에 대한 국제 표준이 결정된 후 그것을 수용하도록 하는 단계적 접근과 PDP는 다양한 라우터를 수용할 수 있는 능력을 가지고 있어야 하므로 기존의 다양한 프로토콜들을 대부분 수용할 수 있어야 할 것이다.

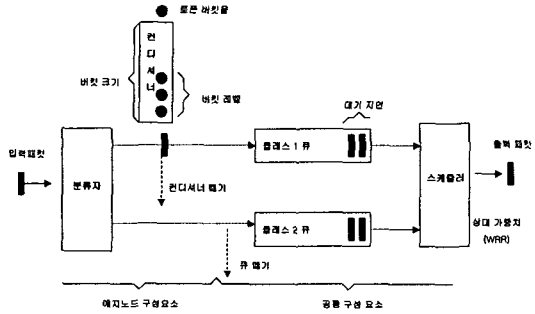
3. 제어기를 이용한 QoS 제공

인터넷에서 QoS 제어와 관련된 정책기반 관리구조와 방법에 대해 보여준다. 정책기반관리구조에서 QoS 변수를 제어하는 것은 일정시간동안 트래픽 예측을 필요로 하기 때문에 복잡한 작업이다. 기존의 트래픽 예측 방법은 트래픽의 임의성으로 인해, 예측에 신뢰성이 떨어지거나, 트래픽의 부정확성과 불확실성이 네트워크에 미치는 영향을 반영하지 못하였다. 이로 인해 경로선택에 어

려움이 있거나, 도메인에서 종단간 지연의 보장에 어려움이 있었다. 이를 해결하기 위해서 복잡성을 줄이면서 트래픽의 불확실성와 부정확성 문제를 해결하기 위해 퍼지 논리에 기초한 제어기를 이용한다.

3.1 DiffServ 노드 구조

DiffServ 구조에서 제어기는 <그림 5>와 같다. DiffServ 구조에서 모든 노드는 각 클래스에 대해 다른 큐를 가지며, 분류자가 패킷을 대응하는 큐에 넣고, 스케줄러가 큐에서 패킷을 처리한다. 이 기능 이외에도, 에지 노드는 각 패킷에 표시를 하기위해 마커를 포함하고, 계약대로 입력 흐름을 유지하기 위해 컨디셔너를 포함한다. 제안된 구조는 2개의 제어기로 구성되며, 하나는 코어와 에지 노드에서 사용하는 큐와 스케줄러를 제어하고, 다른 하나는 에지노드에서 진입 트래픽을 조정하기 위해 사용되는 컨디셔너이다[3].



<그림 5> DiffServ 노드 구조

3.2 퍼지 제어기

트래픽은 EF(Expedited Forwarding)와 BE(Best Effort) 클래스로 분류되는데, EF는 VoIP와 같은 실시간 응용의 최선 클래스이다. 이것은 각 노드에서 적은 지연과 지연 변동을 제공한다. BE 클래스는 어떠한 보장도 없는 IP 네트워크 트래픽을 나타낸다. EF 클래스에 최대 우선순위를 주는 정책을 정의하였다. 제안된 제어기는 좀더 효율적인 코드로 구성하기 위해 트라이앵글과 트레페조이드 집합을 사용한다[5][7].

3.2.1 스케줄러 제어기

스케줄러는 제어가 가능하기 때문에 WRR(Weighted Round Robin) 혹은 WFQ(Weighted Fair Queueing) 타입을 사용하며, 큐는 동작하는 동안 변경되는 정해진 가중치에 따라 서비스된다. 각 큐에서 패킷의 지연과 폐기는 이 가중치에 의해 제어된다. 스케줄러의 첫 번째 입력 변수는 EF/BE 상대 가중치로서, 이것은 전체 가중치(EF + BE)에 대해 EF 큐의 WRR 스케줄러 가중치이다[5][13]. 입력 가중치는 우선순위에 따라 5단계(P0, P1, P2, P3, P4)로 구분하여 퍼지 집합을 이용한다.

두 번째 입력변수는 큐에서 패킷 지연인데, 이것은 노드를 통해 패킷의 지연을 나타내는 것으로, 지연의 정도에 따라 3단계(D0, D1, D2)로 구분하여 퍼지 함수를 이용한다.

세 번째 입력변수는 큐 오버프로우에 의한 폐기율이다. 다수의 동적 큐 관리 메카니즘이 사용되는데, 중요한 것은 폐기가 발생하였다면 자원이 부족하다는 것이다. BE 큐에서 폐기는 시간 구간동안 BE 클래스에서 폐기되는 패킷의 수로 표시된다.

3.2.2 컨디셔너 제어기

컨디셔너의 첫 번째 입력 변수는 퍼킷에 저장되어 있는 토큰의 수이다. 만약 버킷이 가득 차 있다면 이것은 노드에서 입력 트래픽이 출력 트래픽 보다 적다는 것이다. 만약 버킷이 비었다면 이것은 버킷율이 입력 트래픽과 유사하거나 적다는 것이다. 두 번째 변수는 컨디셔너에서 폐기되는 패킷의 수이다. 패킷은 컨디셔너에 도착했을 때 버킷이 비어있다면 패킷은 폐기된다. 이 경우 입력 트래픽이 버킷의 토큰 발생율보다 크다는 것이다. 그러나 컨디셔너는 목적이 트래픽을 계약한 내용대로 유지하는 것이 목적이므로 버킷율의 값을 조정할 수 없다.

DiffServ 영역은 오직 에지노드에서만 패킷을 폐기할 수 있다. 도메인 코어 노드에서 더 이상 자원이 없다면 에지 노드에서 입력율 감소를 버킷 생성율 감소를 통하여 표시해야 한다. 코어에서 높은 지연이 발생하면 에지 노드는 입력율을 감소시켜야 한다. 코어에서 높은 지연이 발생하면 에지 노드는 입력율을 감소시켜야 한다. 컨디셔너의 멤버쉽 함수는 스케줄러 함수와 유사하다. 컨디셔너의 입력 함수는 토큰 버킷율, 버킷 집합, EF 큐에서 폐기율 그리고 코아노드에서 최대 큐 지연이다.

3.2.3 규칙기반과 추론

규칙기반은 퍼지시스템의 원하는 동작을 표현하는 퍼지집합을 가진 If-Then 규칙 그룹이다. 제어기는 크게 스케줄러와 컨디셔너로 나누어서 규칙을 지정하는데, 먼저 스케줄러 제어기의 규칙은 다음과 같다.

① EF 큐의 지연이 D1 이면, 큐 가중치 우선순위는 1을 증가시킨다. 우선순위가 P1이면 이것은 P2로 조정한다. EF 지연이 D2면 큐 가중치 우선순위를 2 증가시킨다.

② EF 큐 지연이 D0이고 BE 큐의 폐기율이 D1이면 큐 우선 순위는 1 감소시킨다. 우선 순위가 D1이면 P1으

로 조정한다. BE 큐의 폐기율이 D2이면 큐 우선 순위는 2 감소시킨다.

한편, 컨디셔너 제어기를 위한 규칙은 다음과 같다.

① 컨디셔너 버킷 레벨이 낮고 EF 큐 폐기율이 중간이면 컨시셔너율을 1 증가시킨다. EF 큐 폐기율이 높으면 컨디셔너율을 2 증가시킨다.

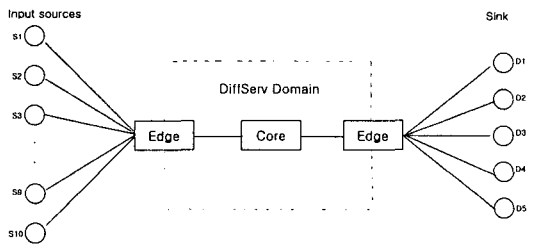
② 컨디셔너 버킷 레벨이 중간이거나 높고 또한 코어 노드에서 EF 큐 지연이 중간이면 컨디셔너율을 1 감소시킨다. 만약 EF 큐 지연이 높으면 컨디셔너율을 2 감소시킨다.

4. 시뮬레이션 및 결과 분석

4.1 시뮬레이션 환경

시뮬레이션은 EF(Expedited Forwarding)와 BE(Best Effort) 클래스에 대해 수행하였다. EF는 VoIP와 같은 실시간 응용의 최선 클래스이다. 이것은 각 노드에서 적은 지연과 지연 변동을 제공한다. BE 클래스는 어떠한 보장도 없는 IP 네트워크를 나타낸다[6][7]. 시뮬레이션을 위한 토폴로지는 <그림 6> 이다.

VoIP 응용은 UDP 상에서 항등비트율(CBR)과 지수분포 on-off 트래픽으로 구현하였다. CBR 트래픽은 네트워크 QoS를 위해서 최악의 경우이고, on-off 트래픽은 정상대화와 유사하다. 음성 트래픽은 EF 클래스와 BE 클래스로 분류된다. 음성연결에서 EF 클래스의 음성 트래픽 수는 변화하며, 각 BE 소스는 64 Kbps이다. on-off 소스에서 버스트시간은 600ms이고 아이들 시간은 400ms라고 하면 평균 전송율은 38.4 Kbps이다[13][14].

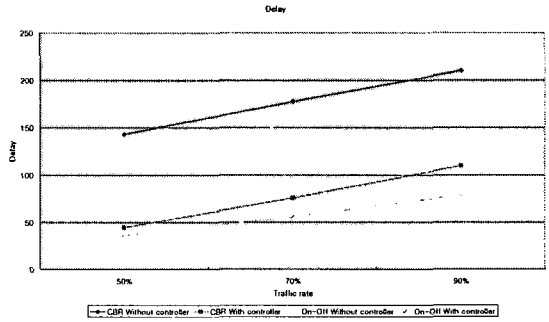


<그림 6> 시뮬레이션 토폴로지

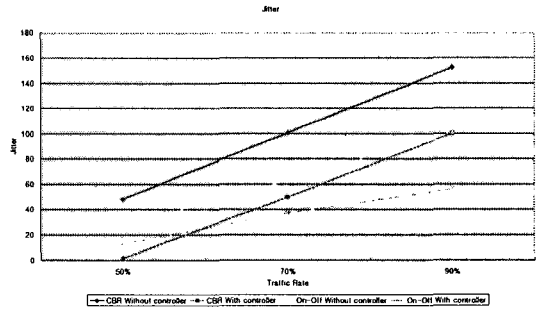
4.2 결과 고찰

성능측정은 EF 클래스의 종단간 지연과 지터의 값을 구하였다. DiffServ 도메인에서 제어기가 없는 경우와 퍼지 제어기를 사용한 경우이다. 각 시뮬레이션의 초기에 각 클래스 별로 50%의 초기 스케줄러 구성을 가지고 시작하였다.

<그림 7>은 지연을 기준으로 전체 패킷중 주어진 비율하에서, 가지는 최대 지연의 값을 을 나타낸 것이고, <그림 8>는 지터를 기준으로 전체 패킷중 주어진 비율하에서, 가지는 최대 지터의 값을 표시한 것이다. 그림에서 보듯이 제어기를 사용하는 경우 지연과 지터에서 우수한 성능을 나타냄을 알 수 있다.



<그림 7> EF 트래픽 지연



<그림 8> EF 트래픽 지연 변동

5. 결 론

본 연구에서는 DiffServ 도메인에서 QoS 보장을 위한 방법을 제시하고 이의 성능을 시뮬레이션을 이용하여 분석하였다. 보다 좋은 QoS를 제공하기 위해 노드에서 동적으로 변수를 재구성하여야 하는데, 이를 위한 퍼지 제어기를 제안하였다. 퍼지 논리의 사용은 도메인에서 유입되는 트래픽의 부정확성과 불확실성의 처리를 향상시킨다.

또한 이 제어기를 사용함으로서 시스템의 복잡성을 낮추어서, 시스템의 확장이 얼마든지 가능하게 됨을 알 수 있다. 성능 평가를 위해 EF와 BE 클래스에 대해 전송지연과 지터를 측정하였다. 측정결과 우수한 성능을 나타냄을 알 수 있다. 앞으로 주어진 정책기반구조에서 좀더 확장된 클래스에 대해서도 동적으로 변수를 설정할 수 있는 제어기에 대한 연구가 따라야 할 것이다.

참 고 문 헌

[1] Srinivas Vergsna, IP Quality of Service, Cisco Press, 2001.

[2] Uyless Black, "Voive over IP, Prentice-Hall, 2000.

[3] M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services", IETF RFC 2475, Dec. 1998.

[4] R. Braden, D. Clark, S. Shenker, "Integrated Services in the Internet Architecture: An Overview, " RFC1633, June 1994.

[5] George J. Klir and Tina A. Folger, "Fuzzy sets, Uncertainty and Information", Prentice-Hall Int., 1988.

[6] NS, "The network Simulator - ns version 2", http://www.isi.edu/nsnam/ns.

[7] S. Floyd, and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance, " IEEE/ACM Transactions on Networking, V.1, N.4, August 1993.

[8] D. Kandlur, D. Saha, and K. Shin, "Understanding TCP Dynamics in an Integrated Services Internet," NOSSDAV '97, May 1997.

[9] DMTF, Distributed Management Task Force, http://www.dmtf.org.

[10] J. Boyle, R. Cohen, D. Duram, S. Herzog, R. Rajan, and A. Sastry, " The COPS (Common Opern Policy Service) Protocol, Internet Draft, December 1998.

[11] UCLA Internet Research Lab., RSVP Diagnostics Tool, http://irl.cs.ucla.edu/

[12] "Differentiated Services-Implementation", http://www.ittc.ukans.edu/~kdrao/BB/.

[13] Andrea Francini and Fabio M. Chiussi, "Providing QoS Guarantees to Unicast and Multicast Flows in Multistage Packet Switches", IEEE Journal on Selected Areas in Communications, Vol. 20, No. 8, pp 1589~1601, 2002.

[14] Constantinos Dovrolis and Dimitrios Stiiliadis, "Proportional Differentiated Services : Delay Differentiation and Packet Scheduling", IEEE/ACM Transactions on Networking, Vol. 10, No. 1, pp 12~26, 2002

[15] David D.Clark and Wenjia Fang, "Explicit Allocation of Best-Effort Packet Delivery Service", IEEE/ACM transactions on Networking, Vol. 6, No. 4, 1998.