논문 04-29-7B-11 한국통신학회논문지 '04-7 Vol.29 No.7B

다중 DiffServ 도메인 상에서 QoS 보장을 위한 동적 클래스 재협상 알고리즘

이 대 붕*, 정회원 송 황 준

Dynamic Class Mapping Mechanism for Guaranteed Service with Minimum Cost over Differentiated Services Networks

Lee dai-boong, Song hwang-jun *Regular Members*

요 약

네트워 환경에서 quality of service(QoS) 에 대한 요구가 증대함에 따라 여러 가지 방안들이 제시되고 있는 가운데, Differentiated services (DiffServ) 모델은 확장성을 보장하며 QoS를 제공할 수 있는 방법으로서 제시되고 있다. 그러나 대부분의 DiffServ 관련 연구들은 통합된 트래픽에 관한 확장성 문제와, 단일 도메인에서의 per hop behavior(PHB)에 관한 문제에만 관심을 두고 있기 때문에, 시간적으로 변화하는 네트웍 조건하에서 다중 도메인의 단대단 QoS를 제공하는 데 있어서는 많은 어려움들이 있다. 따라서 본 연구에서는 다중 DiffServ 도메인에서 네트웍 cost를 최소화 하는 가운데 멀티미디어 데이터의 단대단 QoS를 제공할 수 있는 동적 클래스 재협상 알고리즘을 제안하였다. 본 논문은 DiffServ의 relative 서비스 모델을 효과적으로 구현할 수 있는 방법과 QoS 레벨 전달 메카니즘, 그리고 동적 클래스 재협상 방법으로 구성되어 있으며, 실험 결과를 통해 제안된 알고리즘의 성능을 측정한다.

ABSTRACT

Differentiated services (DiffServ) model has been prevailed as a scalable approach to provide quality of service in the Internet. However, there are difficulties in providing the guaranteed service in terms of end-to-end systems since differentiated services network considers quality of service of aggregated traffic due to the scalability and many researches have been mainly focused on per hop behavior or a single domain behavior. Furthermore quality of service may be time varying according to the network conditions. In this paper, we study dynamic class mapping mechanism to guarantee the end-to-end quality of service for multimedia traffics with the minimum network cost over differentiated services network. The proposed algorithm consists of an effective implementation of relative differentiated service model, quality of service advertising mechanism and dynamic class mapping mechanism. Finally, the experimental results are provided to show the performance of the proposed algorithm.

Keyword : DiffServ, Network QoS, QoS renegotiation, Packet loss rate, Packet Delay

Ⅰ. 서 론

현재의 현재의 통신망 서비스는 과거의 텍스트 중심 서비스 구조에서 벗어나, 동영상과 음성 등의 멀티미디어 서비스를 제공하는 형태로 변화하고 있다. 특히 동영상의 경우 주식 시세나 여행 정보 등의 각종 데이터베이스를 컴퓨터와 TV등의 단말기를 이용하여 영상 형태로 제공하는 서비스 형태가

* 홍익대학교 전파통신공학과 멀티미디어 통신 시스템 연구실 (neferian@hotmail.com)
논문번호 : 040025-0119, 접수일자 : 2004년 1월 19일

요구되고 있으며, 동영상 회의, 동영상 전자 우편, 동영상 전화, 홈쇼핑, 원거리 학습 시스템 등의 서비스 기술 개발이 이루어지고 있다. 그러나 이와 같은 요구와 기술개발에도 불구하고 통신망을 통해 멀티미디어 데이터를 효율적으로 전송하는 데에는 아직까지 많은 어려움이 존재한다. 멀티미디어 데이터 전송의 대표적인 어려움은 일반적인 텍스트 등의 정보에 비해 훨씬 많은 양의 정보를 필요로 할 뿐 아니라, 압축 변수와 데이터 자체의 변화로 인한 버스티 특성을 갖는다는 점에서 기인한다. 즉 일률적인 대역폭과 데이터 전송 구조로는 멀티미디어 데이터 전송 서비스를 제대로 할 수 없으며, 이에 따라 데이터의 특성을 고려하여 실시간 동영상 전송을 가능하게 하기 위한 다양한 방법들이 개발되어야 한다.

효과적인 멀티미디어 데이터 전송 서비스에 관한 연구는 통신망 형태를 구조적으로 변화시키는 방법과 데이터의 특성을 고려한 전송 기법에 관한 연구로서 크게 구분할 수 있다. 기본적으로는 Differentiated services (DiffServ) 와 Intergrated service(IntServ)와 같이 통신망의 서비스 구조를 달리하는 방법을 이용하여, 멀티미디어 서비스뿐만 아니라 다른 요구를 갖는 서비스들을 지원하고자 하는 방법이 있으며, 최근에는 데이터의 특성을 고려한 quality of service (QoS) 재협상에 기반을 둔 방법과 같이 송신단과 수신단의 통신에 기반한 다양한 전송 기법들도 제안되어지고 있다.

확장성 있는 네트워 QoS 제공 방법으로서 DiffServ[1]는 개개 flow 대신에 통합된 flow를 제어함으로서 QoS를 제공하고, 또한 단일 DiffServ 도메인 내의 per hop behavior (PHB) 만 제어하면 되기 때문에 QoS 제공에 있어 가장 확실하고, 편리한 방법으로 인식되고 있다. 그러나 두 개의 끝단 시스템 사이에는 여러개의 DiffServ 도메인이 존재 할 수 있기 때문에, 전체 전송 과정에 있어 단대단 OoS를 제공하고자 할 때에는 다중 DiffServ 도메인의 QoS에 관한 부분이 고려되어야 한다. 그림1은 이러한 다중 DiffServ 도메인의 일반적인 구성을 나타내고 있다.

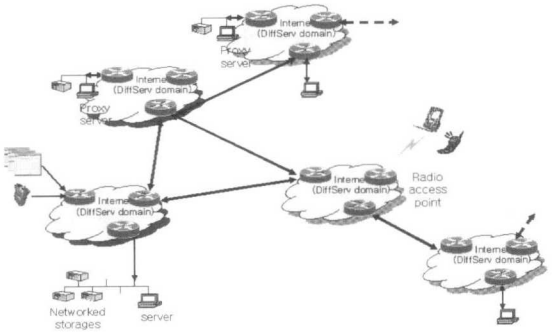


그림 1. 일반적인 다중 DiffServ 구성

그림1의 구성에서 보듯이 DiffServ가 확장성 있는 네트워 QoS 제공방법임에도 불구하고, 단일 도메인의 QoS 제공만으로는 충분하지 않기 때문에, 본 연구에서는 단대단 QoS 제공을 위해서 단일 도메인만이 아닌 전송과정상의 모든 DiffServ 도메인 상에서의 QoS를 고려한다.

일반적으로 각 DiffServ에 따라, 그리고 각 DiffServ의 클래스에 따라 측정되는 QoS(경험적 QoS)는 다르게 되며, 또한 각 class 마다 부여되는 네트워 사용요금(network cost)도 달라지게 된다. 그러므로 단대단 QoS를 제공할 수 있는 최적의 클래스 재협상 알고리즘은 이러한 각 네트워의 상황을 고려할 수 있어야 하며, 전체 전송 상황에서의 네트워 사용률을 향상시킬 수 있는 알고리즘 이어야 한다.

이러한 단대단 QoS를 제공하기 위한 방법으로 제안된 여러 가지 방법들 중 PHB를 확장하여 단일 도메인의 edge-to-edge전송 특성을 예측하고 이에 따라 각 도메인의 per domain behavior (PDB)[2]를 정의하는 방법이 있으나, 이 방법 역시 절대이거나 확률적인 DiffServ의 서비스 제공 방법에 대한 부분에는 아직까지 더욱 연구가 진행되어야 하는 과제가 남아 있다. 또 다른 방법으로는 각 도메인이나 다중 도메인 전체에 걸친 특성을 파악하는데 있어 도메인 서버(bandwidth broker (BB)[3])를 이용하는 방법이 있다. 이 방법의 경우 BB를 이용하여 BB가 관할하는 도메인의 자원을 관리하거나(admission control 등), 이웃한 도메인 BB와의 통신을 통하여 서로간의 서비스 레벨을 협상하는 방법 등을 이용할 수 있으며, QoS 보장과 확장 가능한 서비스 구조를 위해서 제어 구조를 BB에게 집중 시키는 몇 가지 구조가 제안되고 있다.[4]-[5]

일반적인 DiffServ 구조에서는 Traffic aggregates (TAs)를 이용하기 때문에 제공되는 보장형 서비스(guaranteed services)는 절

대적인 값이나 확률적인 값으로 제공될 수 있다. 최근에 Christin[6] 등이 제안한 방법인, quantitative assured forwarding (AF) 서비스의 경우 QoS 요소로서 패킷 손실률과 전송 지연을 이용하고 있으며, 이에 따라 절대적인 값으로, 각 클래스에 따른 차등 서비스를 비례적으로 제공하고 있다. 또한 TA를 이용한 AF 서비스의 경우, 개개 flow와 통합된 flow의 서비스 차이를 비교한 연구[7] 결과에 따르면 두개의 QoS가 큰 차이가 없는 것으로 나타나고 있다. 이러한 연구들은 IETF에서 제안하고 있는 DiffServ의 기본적인 정의를 확대 적용한 방법들로서 절대적인 QoS 값으로서 서비스를 제공하는 것을 기본적인 방법으로 하고 있으며, 절대적인 QoS를 제공하는 또 다른 방법으로는 수신자의 피드백 신호를 이용하여, 송신자가 요구하는 QoS가 만족됐는지를 확인하고 이에 따라 클래스를 재 선택하는 방법이 있다.

언급했던 것처럼, 본 논문의 연구는 시간에 따라 QoS가 변화 할 수 있는 다중 DiffServ 도메인 환경에서 특정한 멀티미디어 서비스의 QoS 요구나, 혹은 특정 사용자의 QoS 요구를 단대단 관점에서 어떻게 제공해 줄 것인가 하는 문제를 다루고 있다. 따라서 본 연구에서는 다중 DiffServ 도메인에서 네트웍 cost를 최소화 하는 가운데 멀티미디어 데이터의 단대단 QoS를 제공할 수 있는 동적 클래스 재협상 알고리즘을 제안한다.

본 논문은 2장에서 효과적인 relative 서비스 모델의 구현 방법 및 QoS 레벨 전달 메카니즘, 그리고 동적 클래스 재협상 방법으로 구성되어 있으며, 3장에서는 실험 결과를 통해 제안된 알고리즘의 성능을 측정한다. 마지막으로 4장에서 결론 및 향후 연구 과제를 제시하는 것으로 구성되어 있다.

II. 동적 클래스 재협상 알고리즘

본 논문에서는 QoS 요소로서 패킷 손실률과 전송 지연을 이용하고 있다. 단대단 전송지연의 경우 전체 전송과정상의 각 DiffServ 도메인에서 발생하는 전송지연의 합으로 나타나게 되고, 패킷 손실률의 경우 각 DiffServ 도메인에서 발생하는 손실률의 곱으로 표현된다. 그리고 전체 전송과정에서 발생하는 cost는 각 도메인에서 발생하는 cost의 합이 된다. 이러한 조건들을 만족하는 가운데 다음 가정을 세울 수 있다.

가 정:

1. 각각의 DiffServ 도메인은 relative 서비스를 제공하며, 서로 독립적인 서비스를 제공한다.

2. 각각의 DiffServ 도메인이 제공하는 클래스의 수는 서로 다를 수 있으며, 이에 따라 각 클래스에서 제공하는 QoS 및, 클래스 사용 cost도 서로 다르다.

가정1의 경우 현재 인터넷 등의 각 ISP에서 운영하는 도메인 서비스는 서로 독립적이기 때문에 가정으로서 전혀 문제가 없으며, 가정2의 경우 가정1에 기반 했을 때 당연히 유추할 수 있는 결론이다.

다중 DiffServ 도메인 상에서의 QoS 제공의 문제는 현재 서비스되고 있는 flow의 서비스 클래스를 각 도메인에서 무엇으로 할 것인가에 관한 문제로 요약할 수 있다. 본 논문에서는 이러한 재협상 과정에 있어 우선 첫 번째로 어느 시점에 재협상이 이루어 질 것인가 하는 문제와(2.1.2), 두 번째로 재협상이 어떻게 이루어지는가에 관한 두 가지 문제(2.2)로 구분하여 논의를 전개한다.

재협상 시점의 선정에 관한 문제에 있어서는 2.1.2에서 자세하게 언급할 내용으로서, 주기적인 경우와 비 주기적인 재협상 시점의 선정이 있다. 주기적인 재협상의 경우 일정 시간 간격으로 각 도메인에서 flow가 사용하는 서비스 클래스를 재 선정하는 방법이며, 비 주기적인 재협상의 경우 QoS에 관한 제한조건을 만족하지 않을 경우에만 클래스를 재협상 함으로서 불필요한 오버헤드를 줄이고자 한다.

일단 재협상 시점이 선정된 후에는 각 도메인에 속해 있는 모든 클래스의 현재 서비스 상태를 기반으로 하여, 2.2의 수식(3)에서의 Problem Formulation과 같이 QoS 한계 조건을 만족하는 가운데, 가장 cost를 최소화시킬 수 있는 클래스를 선택하게 된다.

2.1 Relative 서비스 모델과 QoS 레벨 전달 메카니즘

본 절에서는 효과적인 relative 서비스 모델의 구현 방법 및 QoS 레벨 전달 메카니즘, 그리고 cost를 최소화 하는 가운데 단대단 QoS를 제공할 수 있는 동적 클래스 재협상 방법을 설명하고자 한다.

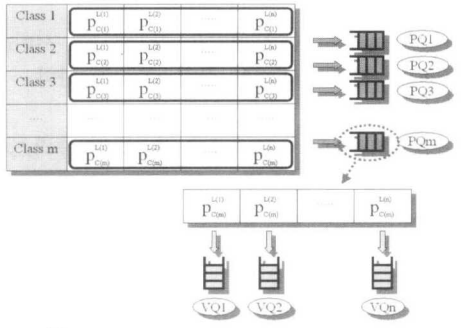


그림 2. RFC 2597에서 제안하는 Queueing 구조

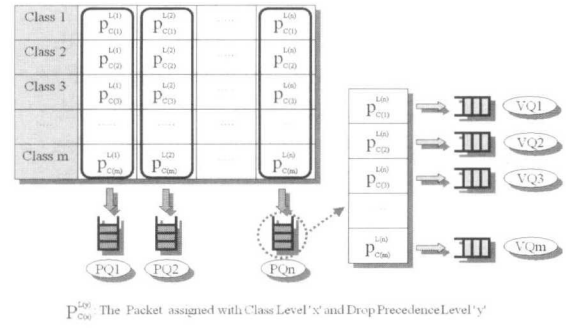


그림 3. 본 연구에서 제안하는 Queueing 구조

2.1.1 Relative 서비스 모델의 구현

DiffServ의 AF(Assured Forwarding) 서비스의 경우 큐(Queue)의 구조는, 일반적으로 독립적인 forwarding 서비스를 제공받는 네 개의 PQ(Physical Queue)와, 각각의 PQ안에 세 가지의 drop level이 가능하도록 세 개의 VQ(Virtual Queue)로서 제안되고 있다.[9] 하나의 PQ에는 같은 종류의 클래스 레벨을 갖는 패킷들만이 유입될 수 있으며, 각 PQ에 유입된 패킷들은 drop precedence에 따라 세 개의 VQ로 구분되게 된다. 그러나 이러한 큐잉(Queueing) 구조와 forwarding 메카니즘을 이용 할 경우, relative 서비스를 위해서는 서비스 레벨에 따른 QoS 상태 정보를 라우터에서 계속 Monitoring하고, 이에 따른 적절한 조치를 취해야만 relative 서비스를 구현 할 수 있게 된다.[10] 그러나 이 경우 core 라우터의 기능을 간소화시키는 DiffServ의 근본 취지에도 어긋날 뿐더러, 복잡한 알고리즘을 요구하게 된다.

따라서 본 논문에서는 큐잉 구조를 달리하여 간단하게 relative서비스를 구현하고자 하였다. 우선 각 클래스의 drop precedence는 edge 라우터로 패킷이 유입될 때 적용되는 policy에 따라 결정된다. 사용될 수 있는 Policer Types 로는 TSW3CM[11], Single Rate Three Color Marker[12], Two Rate Three Color Marker[13] 등이 있으나, 간단한 구현을 위해서 Token Bucket Policer를 사용하였다. Token bucket policer를 사용 할 경우 token에 의해서 forwarding 버퍼로 들어가는 패킷에 대해서는 high precedence가 적용되고, 드롭되는 패킷들은 low precedence가 적용되어 forwarding 버퍼로 유입된다.

이러한 drop precedence 상황에서 RFC 2597[9]에서는 그림2와 같이 동일한 클래스를 가지는 패킷들은 같은 PQ로 유입되고, 하나의 PQ에서 drop precedence에 따라 PQ내에 있는 각각의 VQ로 구분되는 방법을 제안하고 있다. 그러나 본 연구에서 제안하는 방법은 그림3과 같이 동일한 클래스 레벨을 하나의 PQ에 넣는 방법 대신에, 서로 같은 drop precedence 레벨을 갖는 패킷들을 하나의 PQ에 유입 시키고, PQ 안에서 각각의 클래스 레벨에 따라 VQ를 달리하는 방법을 이용하였다. 이 방법을 이용할 경우 수식 (1)의 조건을 만족하도록 각 클래스의 token rate를 조절(CAC 단계에서 입력 트래픽의 평균 rate등의 정보를 제공받기 때문에 이에 따른 token rate 조절이 가능)한 후 패킷 forwarding 메카님즘에서 forwarding의 우선 순위를 수식(2)를 만족하도록(클래스1이 최 상위 클래스라고 할 때) 해주면 relative 서비스 구현이 가능해진다. 본 논문에서는 이러한 forwarding 메카니즘으로 WRR(Weighted Round Robin)을 이용하였으며, 각 클래스별 weight를 조절함으로서 DiffServ의 relative 서비스를 구현하였다.

$$\frac{R_{C(x)}^{L(y)}}{R_{C(x)}^{L(y+1)}} > \frac{R_{C(x+1)}^{L(y)}}{R_{C(x+1)}^{L(y+1)}} \qquad \text{수식 (1)}$$

$$\frac{R_{C(1)}^{L(y)}}{R_{C(1)}^{L(y+1)}} < \frac{W_{PQ(y)}}{W_{PQ(y+1)}} \qquad \text{수식 (2)}$$

x : 클래스 번호 $x = 1, 2, 3, \ldots\ldots, m-1$

y : drop precedence 레벨 번호

$y = 1, 2, 3, \ldots, n-1$

m : 클래스의 수

n : drop precedence의 수

$R_{C(x)}^{L(y)}$: 클래스 x 중 drop precedence 레벨 y 로 유입되는 패킷 rate

$W_{PQ(y)}$: Physical Queue y의 weight

2.1.2 QoS 레벨 전달 메카니즘

Relative 서비스를 제공하는 각 DiffServ 도메인에서의 QoS 요소는 네트워 상황과 시간에 따라 변화할 수 있기 때문에 랜덤한 값으로 나타낼 수 있으며, 이에 따라 전체 전송 과정상의 모든 DiffServ를 고려한 단대단 QoS 역시 랜덤 함수로 표현 할 수 가 있다. 또한 [17]에 따르면 두 끝단 시스템 사이에 존재하는 독립적인 도메인의 수가 증가 할수록 단대단 QoS의 분산은 증가하게 된다. 결과적으로 이러한 분산의 증가는 연속적인 서비스를 요구하는 응용에 대하여 심각한 전송효율의 감소를 가져 올 수 있게 된다. 따라서 본 절에서는 네트워 cost를 최소화 하는 가운데 단대단 QoS를 보장 할 수 있는 동적 클래스 재협상 알고리즘을 제안하고자 한다.

실험결과의 그림2를 살펴보면 개개 flow의 QoS 레벨과 그 flow가 속해있는 클래스의 QoS 레벨이 거의 유사한 것을 볼 수 가 있다. 이 결과에 따르면 flow가 속해있는 클래스의 QoS 레벨을 측정함으로서, 우리는 개개 flow의 QoS 레벨을 유추할 수 있으며, TA를 기반으로 하는 DiffServ기본 개념상 개개 flow의 QoS레벨을 파악하는 것이 어렵다고 할 때, 이는 계산량을 줄일 수 있는 매우 유용한 방법이라고 할 수 있으며 [10]에서의 결론도 이를 증명하고 있다.

네트워 상황에 따른 수락제어(CAC : Call Admission Control)에 따라 relative 서비스 환경에서의 QoS 레벨은 매우 유동적으로 변화하며, 따라서 이러한 환경 하에서 QoS를 보장하기 위한 QoS 레벨 전달 메카니즘이 필요하다. 그러나 QoS 레벨 전달 등의 과정은 불필요한 오버헤드와 계산을 필요로 하기 때문에, 본 절에서는 이를 줄이기 위한 효율적인 전달 메카니즘을 제안한다. 또한 이러한 QoS 레벨 전달의 과정을 주기적인 경우와 비 주기적인 경우로 구분하고, QoS 레벨 전달 횟수와 이에 따른 QoS의 개선을 알아보고자 한다.

2.1.2.1 주기적인 QoS 레벨 전달

주기적인 시각 간격으로 각 도메인은 자신에게 속해있는 클래스의 QoS 레벨을 계산하고, 다른 도메인에게 새롭게 계산된 QoS 레벨을 전달한다. 이를 통해 사용자는 자신이 이용하는 전체 경로상의 도메인에서 앞으로 이용할 클래스를 재 선택하게 된다.

2.1.2.2 비주기적인 QoS 레벨 전달

불필요한 QoS 전달 과정과 이에 따른 클래스 재협상의 문제, 그리고 신호의 오버헤드를 줄이기 위하여, 비 주기적인 시간 간격의 QoS 레벨 전달 메카니즘을 제안 할 수 있다. 비 주기적인 QoS 레벨 전달의 경우 사용자의 flow가 속해있는 클래스의 과거 QoS 레벨(이를 계산하기 위하여 sliding window를 이용한다.)과 현재 측정된 QoS 레벨의 차이가 일정 값(threshold value)이상일 경우에만 QoS 레벨을 전달하게 된다. 이를 간략히 수식화 하면 다음과 같다.

$$\text{if } \frac{|QoS_{cur} - QoS_{prev}|}{QoS_{prev}} > T \text{ ,}$$

then : QoS 레벨 전달

QoS_{cur} : 현재의 QoS 레벨

QoS_{prev} : 과거의 QoS 레벨

T : 임계값

이 경우 window를 이용하는 것은 주파수 도메인에서의 Low Pass Filter 역할을 하기 때문에, window의 길이는 과거의 QoS 레벨을 결정하는데 있어 상당한 영향을 주게 된다.

2.2 문제 Formulation 및 Trellis를 이용한 동적 클래스 재협상 알고리즘

일단 새로운 QoS 레벨이 전달된 후에는, 단대단 QoS를 보장하면서 cost를 최소화 시킬 수 있는 클래스를 재 선택해야 하며, 다음과 같은 과정을 이용한다.

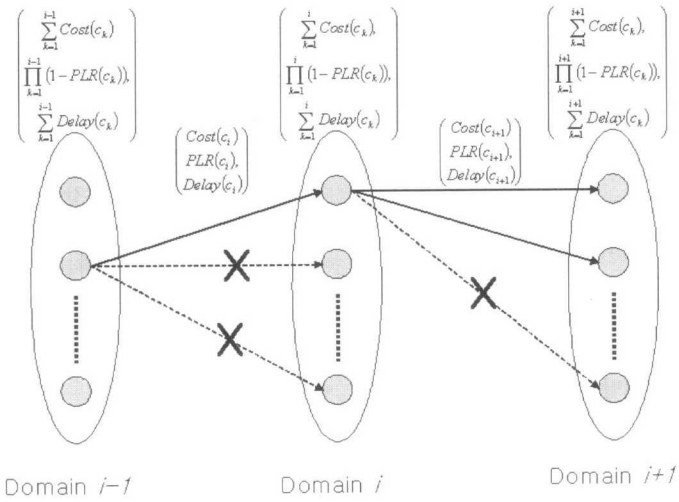


그림 4. 최적의 클래스 선택을 위한 Trellis 알고리즘

Problem Formulation:

Determine $(c_1, c_2, c_3, \ldots\ldots, c_N)$

to minimize $\sum_{i=1}^{N} Cost(c_i)$

subject to,

$$\sum_{i=1}^{N} Delay(c_i) \leq Delay_{req},$$

$$\Pi i = 1N(1 - PLR(c_i) \leq 1 - PLR_{req}, \quad \text{수식(3)}$$

c_i : 도메인 i에서 선택된 클래스

N : 양 끝단 사이에 존재하는 Diffserv의 수

$Cost(c_i)$: 도메인 i에서 선택된 클래스의 네트웍 cost

$PLR(c_i)$: 도메인 i에서 선택된 클래스의 패킷 손실률

$Delay(c_i)$: 도메인 i에서 선택된 클래스의 패킷 전송지연

PLR_{req} : 사용자가 요구하는 패킷 손실률의 한계

$Delay_{req}$: 사용자가 요구하는 패킷 전송지연의 한계

최적의 path를 구하는 문제에 있어서는 재협상 시점에 전달된, 각 도메인에 속해 있는 모든 클래스의 현재 서비스 상태를 기반으로 하여 가능한 한 모든 path를 검색한다.

이렇게 검색된 모든 path들 중 수식(3)의 제한조건을 만족하는 가운데 가장 cost를 최소화시킬 수 있는 클래스를 선택하게 된다. 그러나 이러한 full search의 경우 연산 시간이 오래 걸 릴 수 있으므로, 본 논문에서는 Trellis[9]-[10] 방법과 fast pruning 알고리즘을 이용하여 연산 시간을 줄이고자 하였다.

최적의 path 선택은 사용자의 QoS 요구를 만족할 수 있는 path의 선택과 함께, 사용자가 네트웍을 사용함으로 지불해야 하는 네트웍 cost(price)를 최소화 하고자 하는 의미를 갖는다. 일반적으로 DiffServ 도메인에서는 사용하는 클래스의 priority에 따라 과금도 달라지게 된다. 따라서 사용자의 QoS에 따라 cost를 줄 일 수 있는 적절한 서비스 클래스를 선택하는 것이 중요하며, 본 논문에서 제시한 path 재협상 알고리즘을 이용하여 최저의 비용으로 원하는 QoS 조건을

표 1. 테스트 파일의 수치적 특성

Trace File	Minimum value (Byte)	Maximum value (Byte)	Average (Byte)	Standard deviation (Byte)
Star Wars	275	124816	9313.2	12902.725
Termi-nator-2	312	79560	10904.75	10158.031

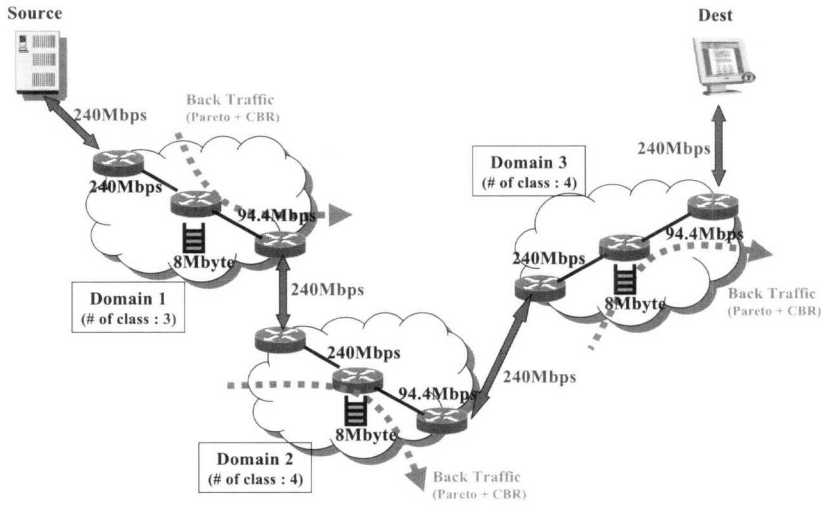


그림 5. 전체 시스템 구성

만족 할 수 있는 서비스를 제공받게 된다.

전체적인 클래스 선택 과정은 그림4과 같으며, 그림에서 볼 수 있듯이 i-1번째 도메인의 클래스 중 조건을 만족하지 않는 클래스의 경우, 이 클래스를 포함하는 i번째 도메인의 클래스는 고려할 필요가 없게 되므로, 요구되는 계산량을 줄 일 수 있게 된다.

Ⅲ. 실험 및 결과

제안된 알고리즘의 성능평가를 위해 NS-2[23]를 이용하였다. 앞에서 언급했던 것처럼 QoS 보장을 위한 QoS 레벨 전달 메카니즘과 동적 클래스 재협상 방법의 성능을 측정하기 위하여, 재협상이 없을 때와 주기적 시간 간격으로 재협상 있을 때, 그리고 비주기적인 시간간격으로 재협상을 했을 때의 3가지 조건을 비교 하였다.

콘트롤 신호 오버헤드의 척도로는 QoS 레벨 전달의 횟수와 이에 따른 클래스 재협상 횟수를 이용하였으며, 전송 과정의 QoS 요소로는 패킷 손실률과 전송 지연을 이용하였다. 전체적인 실험 환경과 시스템 구성은 그림5와 같다.

연속적인 서비스를 요구하는 응용의 경우 패킷 손실률의 최대 값의 증가는 서비스 품질을 떨어뜨리는 요소가 되기 때문에, 패킷 손실률의 평균과 표준편차, 최대 값을 성능 평가의 척도로 이용하였다.

실험에 이용된 동영상으로는 MPEG-1[21, 22]으로 인코딩 된 Star Wars (240*352 size) 와 Terminator-2 (240*352 size)가 이용되었으며, 각각의 길이는 40,000 프레임으로 구성되어 있다. 두 파일의 인코딩 구조는 IBBPBBPBBPBB (1GOP 는 12 프레임으로 구성)으로서, I 프레임과 P 프레임, 그리고 B 프레임의 양자화 계수는 각각 10, 14, 18로 되어 있다. 또한 초당 25프레임으로 인코딩 되어 있으며, 결과적으로 출력되는 트래픽은 VBR (Variable Bit Rate)로 나타난다. 표1은 이러한 테스트 파일의 수치적 특성을 나타낸다.

3.1 Relative Differentiated 서비스 모델에서의 개개 flow와 통합된 flow(TAs)의 QoS 비교

3.1.1 구현한 Relative 서비스 모델의 성능 평가

우선 제안된 relative 서비스 모델의 성능 파악하기위하여 입력으로 제공되는 테스트 트래픽은 CBR과 VBR의 두 가지 경우를 실험 하였으며, VBR의 경우 Pareto[23] 랜덤 함수를 이용하여 생성하였다.

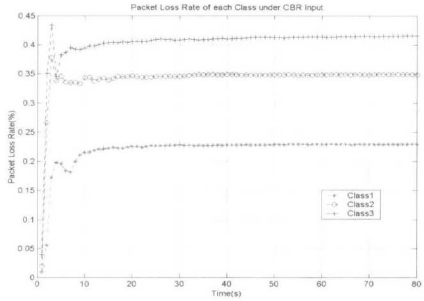


그림 6. Relative 서비스 환경에서의 패킷 손실률 (입력 트래픽 : CBR)

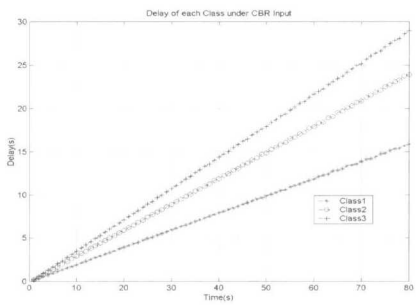


그림 7. Relative 서비스 환경에서의 전송지연 (입력 트래픽 : CBR)

그림6, 7 에서 보는 것처럼 입력 트래픽이 CBR일 경우 패킷 손실률과 전송 지연은 클래스의 우선순위에 따라 비례적으로 나타나는 것을 볼 수 있으며, 입력 트래픽이 Pareto일 경우에는 그림

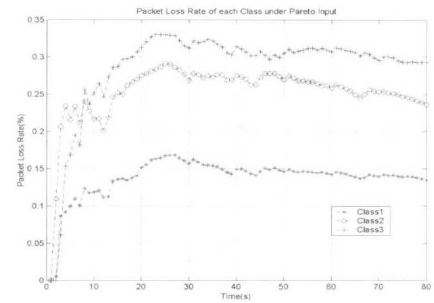


그림 8. Relative 서비스 환경에서의 패킷 손실률 (입력 트래픽 : Pareto)

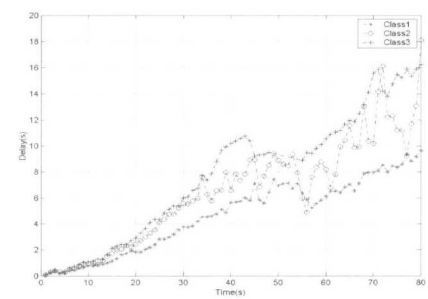


그림 9. Relative 서비스 환경에서의 전송지연 (입력 트래픽 : Pareto)

8,9와 같이 클래스2의 QoS 레벨이 클래스1의 QoS 레벨과 클래스3의 QoS 레벨 사이에서 유지되는 것을 볼 수 있다.

실험 결과에서 보듯이 제안된 relative 서비스

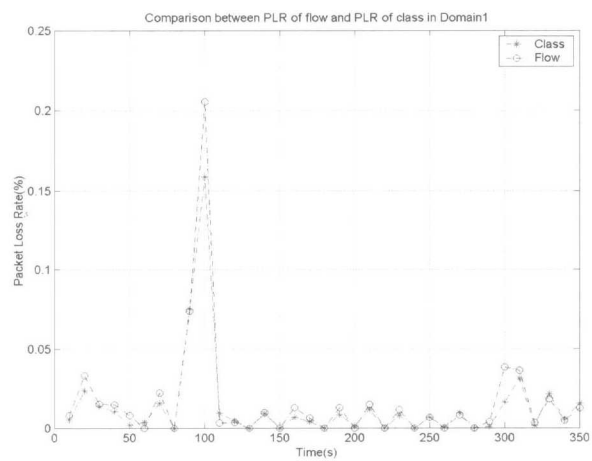


그림 10. 개개 flow와 통합된 flow의 패킷 손실률 비교

모델은 각각의 클래스의 우선순위에 따른 상대적인 QoS레벨을 유지할 수 있으므로 relative 서비스 환경의 DiffServ 모델 구현에 문제가 없다.

3.1.2 개개 flow와 통합된 flow(TAs)의 QoS 비교

그림10은 하나의 flow와 이 flow가 속해있는 클래스의 QoS 레벨이 거의 유사함을 보여주고 있다. 결과적으로 개개 flow의 QoS 레벨을 측정하기 위하여 불필요한 계산을 수행할 필요 없이 flow가 속해 있는 클래스의 QoS 레벨을 측정함으로서 우리는 개개 flow의 QoS 레벨을 유추할 수 있게 된다.

3.2 주기적인 클래스 재협상 알고리즘

주기적인 시간 간격 클래스 재협상 알고리즘의 성능을 파악하기 위한 과정으로서 위해서는 다음과 같은 QoS 요구조건을 제시한다. 이는 2.2절에 설한 것과 같이 클래스 선택에 있어서 이용되며, 사용자가 수용할 수 있는 최저의 QoS 레벨을 의미한다.

표 2. 주기적 클래스 재협상의 패킷 손실률과 전송지연

Periodic Class Change					No Class Change			
Rene. Interval	No. of Class Change	Avg. PLR	Avg. Delay	Ave. cost	Ave. cost	Used path	Avg. PLR	Avg. Delay
10s	35	0.0424	0.0941	5.314	6	2-1-3	0.0554	0.0856
						2-3-4	0.0547	0.0872
						3-2-4	0.0558	0.0895
						3-3-3	0.0578	0.0956
20s	18	0.0508	0.0966	4.971	5	2-4-4	0.0724	0.0918
						3-3-4	0.0768	0.0949
						3-4-3	0.0726	0.0957
30s	12	0.0538	0.0923	4.721	5	2-4-4	0.0724	0.0918
						3-3-4	0.0768	0.0949
						3-4-3	0.0726	0.0957
40s	9	0.0694	0.0951	4.539	5	2-4-4	0.0724	0.0918
						3-3-4	0.0768	0.0949
						3-4-3	0.0726	0.0957
50s	7	0.0898	0.0959	4	4	3-4-4	0.0898	0.0959

표 3. 주기적 클래스 재협상의 최대 패킷 손실률과 표준편차

Periodic Class Change					No Class Change			
Rene. Interval	No. of Class Change	MAX. PLR	STDEV of PLR	Ave. cost	Ave. cost	Used path	MAX. PLR	STDEV of PLR
10s	35	0.2085	0.0512	5.314	6	2-1-3	0.4186	0.0736
						2-3-4	0.4455	0.0794
						3-2-4	0.4278	0.0764
						3-3-3	0.4370	0.0751
20s	18	0.2914	0.0705	4.971	5	2-4-4	0.5314	0.0954
						3-3-4	0.5412	0.0966
						3-4-3	0.5347	0.0944
30s	12	0.3269	0.0722	4.721	5	2-4-4	0.5314	0.0954
						3-3-4	0.5412	0.0966
						3-4-3	0.5347	0.0944
40s	9	0.3364	0.0737	4.539	5	2-4-4	0.5314	0.0954
						3-3-4	0.5412	0.0966
						3-4-3	0.5347	0.0944
50s	7	0.3777	0.0784	4	4	3-4-4	0.5777	0.0984

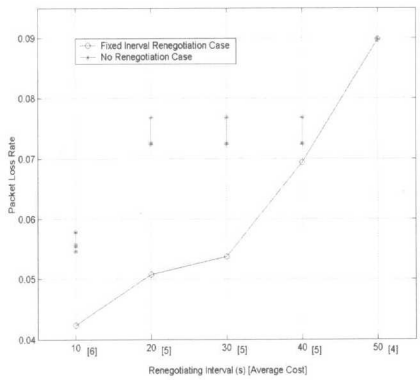


그림 11. 주기적 클래스 재협상의 패킷 손실률

$$Delay_{req} \langle 0.16s$$
$$PLR_{req} \langle 5\%$$

시간간격을 다양하게 조절하여 실험을 진행한 결과, 시간 간격이 줄어들수록 패킷 손실률과 표준편차, 그리고 최대 값이 개선되었으며, 표2와 그림11~13을 통해 확인 할 수 있다. 그러나 클래스 재협상 횟수가 증가하는 것은 상대적으로 신호 오버헤드가 증가하는 것을 의미하기도 한다.

표2와 그림11~13에서 보듯이 주기적인 시간 간격의 클래스 재협상 알고리즘은 더 낮은 네트워 cost만을 가지고, 평균 단대단 QoS 레벨을 개선시키고 있으며, 패킷 손실률과 표준편차에 있어서도

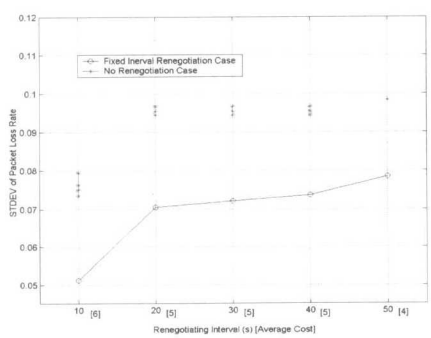


그림 12. 주기적 클래스 재협상의 패킷 손실률 표준편차

표3과 그림8에서 보듯이 향상된 결과를 보이고 있다. 또한 재협상 간격이 50초일 때의 결과에서 보듯이 동일한 cost를 갖는 재협상 없는 결과와 비교 했을 때 평균 패킷 손실률은 같으나, 표준편차와 최대 값은 감소함으로서 재협상을 통한 전송 효율의 증가를 보여 주고 있다.

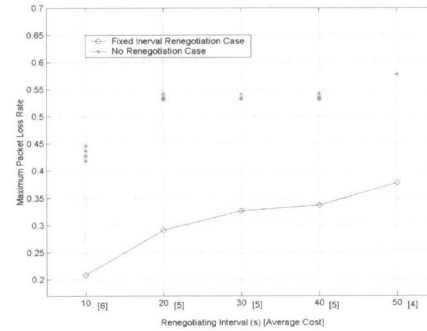


그림 13. 주기적 클래스 재협상의 최대 패킷 손실률

3.3. 비 주기적인 클래스 재협상 알고리즘

비주기적인 클래스 재협상의 경우에는 주기적인 재협상의 경우와 재협상이 없는 경우의 실험 결과를 비교 하여 성능을 평가하였다. 기본적인 QoS 제한 조건은 3.2절의 제한 조건과 같으며, 과거의 QoS 레벨을 계산하기 위한 window의 크기는 3으로 하였다. 여기서 각 window의 weight는 {2, 3, 5}로 하였으며, window는 매 초마다 현재 시간에 근접하게 한 칸씩 이동하게 된다.

QoS 레벨 전달 여부의 판정에 있어서는, window를 이용하여 QoS_{prev}를 계산 한 후, 2.1.2.2절에서 설명한 바와 같이 QoS_{prev}와 QoS_{cur}를 비교하여, 현재 시점에서의 QOS 전달 여부를 결정하게 된다. 실제 실험 과정에서는 연속적인 판단이 불가능 하므로, 이 과정은 매 초 단위로 진행 되었음을 밝혀둔다.

재협상이 없을 때의 경우와 비교한 결과가 표4와 5, 그림14~16에 나와 있다. 결과에서 보듯이 거의 동일한 네트워 cost 환경에도 불구하고 재협상이 없는 경우와 비교 했을 때, 비주기적인 재협상의 경우 평균 패킷 손실률은 30%, 표준편차는 50%, 그리고 패킷 손실률의 최대값은 70%정도의 개선을 보이고 있다.

주기적인 시간간격 재협상의 경우와 비교했을 동일한재협상 횟수에도 불구하고 비주기적인 재협상의 경우 평균 패킷 손실률은 25%, 표준편차는 50%, 그리고 패킷 손실률의 최대 값은 60%정도의 개선

표 4. 비주기적 클래스 재협상의 패킷 손실률과 전송지연 (재협상 없음과 비교)

Aperiodic Class Change					No Class Change			
Threshold of Adv.	Avg. PLR	Avg. Delay	No. of Class change	Ave. cost	Ave. cost	Used path	Avg. PLR	Avg. Delay
10%	0.0385	0.0924	15	5.7451	6	2-1-3	0.0554	0.0856
						2-3-4	0.0547	0.0872
						3-2-4	0.0558	0.0895
						3-3-3	0.0578	0.0956
20%	0.0385	0.0924	15	5.7451	6	2-1-3	0.0554	0.0856
						2-3-4	0.0547	0.0872
						3-2-4	0.0558	0.0895
						3-3-3	0.0578	0.0956
30%	0.0447	0.0929	12	5.3012	6	2-1-3	0.0554	0.0856
						2-3-4	0.0547	0.0872
						3-2-4	0.0558	0.0895
						3-3-3	0.0578	0.0956
40%	0.0520	0.0934	12	5.1942	5	2-4-4	0.0724	0.0918
						3-3-4	0.0768	0.0949
						3-4-3	0.0726	0.0957
50%	0.0651	0.0975	10	5.1428	5	2-4-4	0.0724	0.0918
						3-3-4	0.0768	0.0949
						3-4-3	0.0726	0.0957

표 5. 비주기적 클래스 재협상의 최대 패킷 손실률과 표준편차 (재협상 없음과 비교)

Aperiodic Class Change					No Class Change			
Threshold of Adv.	MAX. PLR	STDEV of PLR	No. of Class Change.	Ave. cost	Ave. cost	Used path	MAX PLR	STDEV of PLR
10%	0.1099	0.0337	15	5.7451	6	2-1-3	0.4186	0.0736
						2-3-4	0.4455	0.0794
						3-2-4	0.4278	0.0764
						3-3-3	0.4370	0.0751
20%	0.1099	0.0337	15	5.7451	6	2-1-3	0.4186	0.0736
						2-3-4	0.4455	0.0794
						3-2-4	0.4278	0.0764
						3-3-3	0.4370	0.0751
30%	0.1440	0.0356	12	5.3012	6	2-1-3	0.4186	0.0736
						2-3-4	0.4455	0.0794
						3-2-4	0.4278	0.0764
						3-3-3	0.4370	0.0751
40%	0.2000	0.0380	12	5.1942	5	2-4-4	0.5314	0.0954
						3-3-4	0.5412	0.0966
						3-4-3	0.5347	0.0944
50%	0.2920	0.0694	10	5.1428	5	2-4-4	0.5314	0.0954
						3-3-4	0.5412	0.0966
						3-4-3	0.5347	0.0944

표 6. 비주기적 클래스 재협상의 패킷 손실률과 전송지연 (주기적 재협상과 비교)

Aperiodic Class Change					Periodic Class Change			
Threshold of Adv.	Avg. PLR	Avg. Delay	No. of Class change	Ave. cost	No. of Class change	Fixed Interval	Avg. PLR	Avg. Delay
10%	0.0385	0.0924	15	5.7451	15	23s	0.0512	0.0962
20%	0.0385	0.0924	15	5.7451	15	23s	0.0512	0.0962
30%	0.0447	0.0929	12	5.3012	12	30s	0.0538	0.0923
40%	0.0520	0.0934	12	5.1942	12	30s	0.0538	0.0923
50%	0.0651	0.0975	10	5.1428	10	35s	0.0670	0.0944

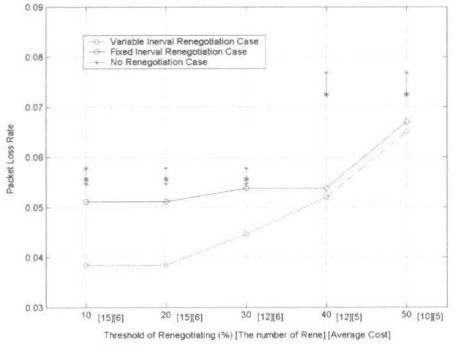


그림 14. 비주기적 클래스 재협상의 패킷 손실률

을 보이고 있으며, 표6과 7, 그림14~16을 통해 확인 할 수 있다. 결과적으로 비주기적인 시간 간격의 클래스 재협상 알고리즘은 동일한 계산 량에도 불구하고 주기적인 재협상의 경우보다 효율적이라고 할 수 있다.

Ⅳ. 결 론

지금까지, 본 논문에서는 다중 DiffServ 도메인에서 네트웍 cost를 최소화 하는 가운데 멀티미디어 데이터의 단대단 QoS를 제공할 수 있는 동적 클래스 재협상 알고리즘을 제안하였다. 제안된 알고리즘은 DiffServ의 relative 서비스 모델을 효과적으로 구현할 수 있는 방법과 QoS 레벨 전달 메카니즘, 그리고 동적 클래스 재협상 방법으로 구성되어 있으며, 실험 결과를 통해 제안된 알고리즘의 성능을 측정하였다.

실험결과, 동적 클래스 재협상 알고리즘은 동일한 cost를 갖는 재협상 없는 경우에 비해 QoS 레벨의 평균과 표준편차에 있어서 개선된 결과를 보이고 있으며, 비주기적인 재협상 알고리즘은 동일한 재협상 횟수를 갖는 주기적인 재협상 알고리즘보다 여러 가지 QoS 측면에서 효율성이 증대된 것을 볼 수 있다.

표 7. 비주기적 클래스 재협상의 최대 패킷 손실률과 표준편차 (주기적 재협상과 비교)

Variable Interval Case					Periodic Class Change			
Threshold of Adv.	MAX PLR	STDEV of PLR	No. of Class change	Ave. cost	No. of Class change	Fixed Interval	MAX PLR	STDEV of PLR
10%	0.1099	0.0337	15	5.7451	15	23s	0.3145	0.0714
20%	0.1099	0.0337	15	5.7451	15	23s	0.3145	0.0714
30%	0.1440	0.0356	12	5.3012	12	30s	0.3269	0.0722
40%	0.2000	0.0380	12	5.1942	12	30s	0.3269	0.0722
50%	0.2920	0.0694	10	5.1428	10	35s	0.3352	0.0729

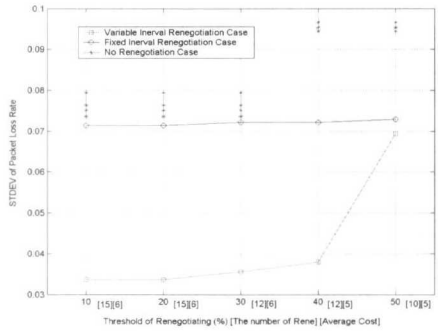


그림 15. 비주기적 클래스 재협상의 패킷 손실률의 표준편차

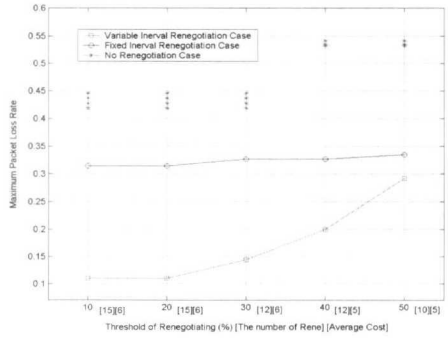


그림 16. 비주기적 클래스 재협상의 최대 패킷 손실률

지금까지의 연구 결과를 바탕으로 향후에는 재협상 제어 신호 및 오버 헤드의 문제를 포함한 전체 시스템의 구성 및 이에 따른 성능분석의 과제가 남아 있으며, 실제 시스템 구성 및 적용도 향후 연구 과제로 남아 있다.

참 고 문 헌

[1] S. Blake, D. Blake, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services," RFC2475, IETF, Dec. 1998.

[2] K. Nichols and B. Carpenter, "Definition of Differentiated Services Per Domain Behaviors and Rules for their Specification," RFC3086, IETF, April 2001.

[3] K. Nichos, V. Jacobson, and L. Zhang, "A two-bit differentiated service architecture for the Internet, " RFC2638, IETF, July 1999.

[4] Z. Zhang, Z. Duan , L. Gao , Y. T. Hou, "Decoupling QoS control from core routers: a novel bandwidth broker architecture for scalable support of guaranteed services," Proc. ACM SIGCOMM, Aug. 2000.

[5] E. Nikolouzou et al., "Network Services Definition and deployment in a differentiated services architecture," Proc. IEEE ICC, April 2002.

[6] N. Christin, J. Liebeherr, and T. F. Abdelzaher, "A quantitative assured forwarding service,"Proc. IEEE INFOCOM, June 2002.

[7] Y. Xu and R. Guerin, "Individual QoS versus Aggregate QoS: A Loss Performance Study," Proc. IEEE INFOCOM, June 2002.

[8] C. Dovrolis and P. Ramanathan, "Dynamic class selection:from relative differentiation to absolute QoS," Proc. ICNP 2001.

[9] J. Heinanen, F. Baker, W. Weiss and J. Wroclawski, "Assured Forwarding PHB Group", RFC2597, IETF, June 1999.

[10] Constantinos Dovrolis and Parameswaran Ramanathan, "A case for Relative Differentiated Services and the Proportional differentiation Model," IEEE Network, September/October 1999.

[11] Fang, W., N. Seddigh, and B. Nandy. "A Time Sliding Window Three Color Marker," RFC2859, IETF, June, 2000.

[12] Heinanen, J., T. Finland, and R. Guerin. "A Single Rate Three Color Marker," RFC2697, IETF, September, 1999.

[13] Heinanen, J., T. Finland, and R. Guerin. "A Two Rate Three Color Marker," RFC2698, IETF, September, 1999.

[14] A. J. Viterbi and J. K. Omura, Principle of Digital Communication and Coding, New York, McGraw-Hill, 1979.

[15] A. Ortega, K. Ramchandran and M. Vetterli, "Optimal trellis-based buffered compression and fast approximations," IEEE Trans. On Image Processing, Vol. 3, No. 1, Jan. 1994.

[16] D. P. Bertsekas, Nonlinear Programming, Massachusetts, Athena Scientific, 1995.

[17] A. Papoulis, Probability, Random ariables,

and Stochastic Process, McGraw-Hill Inc., 1991.

[18] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," IEEE Trans. Signal Processing, Vol. 36, pp. 1445-1453, Sept. 1988.

[19] D. G. Lueberger, Linear and Nonlinear Programming, Addison-Wesley, 1984.

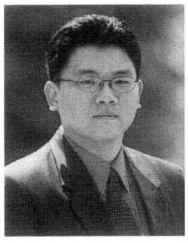
[20] UCB/LBNL/VINT, "Network Simulator ns (ver. 2)," http://www-mesh.cs.berkeley.edu/ns, 1998.

[21] Berkeley Multimedia Research Center, ftp://mm-ftp.cs.berkeley.edu/pub/

[22] KAIST, http://viscom.kaist.ac.kr/

[23] http://nile.wpi.edu/NS/

이 대 붕(Dai-boong Lee)



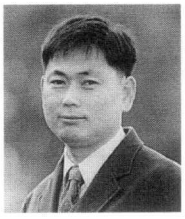
2002. 2 : 홍익대학교 공과대학 전자전기제어공학과 학사

2004. 2 : 홍익대학교 공과대학 전파통신공학과 석사

2004. 3 ~ 현재 : 홍익대학교 공과대학 전파통신공학과 박사과정

<관심분야> Multimedia Communication System, Network QoS, Diff/Int-Serv, Real time video/image transmission

송 황 준(Hwang-jun Song) 정회원



1990. 2 : 서울대학교 공과대학 제어계측과 학사

1992. 2 : 서울대학교 공과대학원 제어계측과 석사

1999. 5 : EE-System, University of Southern California, Los Angeles, USA (공학박사)

2000. 9 ~ 현재 : 홍익대학교 공과대학 전자전기공학부 조교수

<관심분야> Multimedia communication/ signal processing, Packet video, Network protocols necessary to implement a functional real-time image/video application