QoS 적응형 플로우 기반 Active Queue Management 알고리즘 및 성능분석

QoS Adaptive Flow based Active Queue Management Algorithm and Performance Analysis

강현명, 최환석, 이우섭
국립한밭대학교 정보통신전문대학원 멀티미디어공학과

Hyun-Myoung Kang(dcracker@nate.com), Hoan-Suk Choi(rapper84@nate.com),
Woo-Seop Rhee(wsrhee@hanbat.ac.kr)

요약

방송 통신의 융합에 따라 차세대 멀티미디어 서비스인 IPTV 서비스가 각광받고 있다. IPTV 서비스는 무한한 채널수용과 미디어의 확장, 품질이 보장되는 서비스의 형태로 발전되면서 기존 네트워크의 트래픽 증가와 서비스 특성에 맞는 품질제어 기술들이 요구되고 이에 따라 플로우별 서비스 품질 제어 기술이 필요하게 되었다. 이를 위해 IETF (Internet Engineering Task Force)에서도 인터넷 상에서 IP QoS를 제공하기 위한 연구가 활발히 이루어지고 있는데 그 중 버퍼 관리 메커니즘으로 RED(Random Early Detection) 알고리즘 및 RED를 변형한 여러 가지 방법들이 제시되고 있으나 다양한 멀티미디어 서비스의 QoS 요구 사항을 충족시키는데 어려움이 있다. 이에 따라, 본 논문에서는 다양한 QoS를 요구하는 인터넷 멀티미디어 서비스를 지원하기 위해 각 서비스가 요구하는 목표 패킷 손실률을 유지시킬 수 있는 QoS 적응형 플로우 기반 AQM (Active Queue Management) 알고리즘을 제안하고 ns-2를 이용한 시뮬레이션을 통해 성능 분석을 제시하였다.

■ 중심어 : | AQM | 버퍼 메니지먼트 | IP QOS | 멀티미디어 서비스 |

Abstract

Due to the convergence of broadcasting and communications, IPTV services are spotlighted as the that next-generation multimedia services. IPTV services should have functionality such as unlimited channel capacity, extension of media, QoS awareness and are required increasing traffic and quality control technology to adapt the attributes of IPTV service. Consequently, flow based quality control techniques are needed. Therefore, many studies for providing Internet QoS are performed at IETF (Internet Engineering Task Force). As the buffer management mechanism among IP QoS methods, active queue management method such as RED(Random Early Detection) and modified RED algorithms have proposed. However, these algorithms have difficulties to satisfy the requirements of various Internet user QoS. Therefore, in this paper we propose the Flow based AQM(Active Queue Management) algorithm for the multimedia services that request various QoS requirements. The proposed algorithm can converge the packet loss ratio to the target packet loss ratio of required QoS requirements. And we present a performance evaluation by the simulations using the ns-2.

■ keyword : | AQM | Buffer Management | IP QoS | Multimedia Service |

* 본 연구는 2009년도 한밭대학교 교내 학술연구비 지원으로 수행되었습니다.
접수번호 : #091030-004
접수일자 : 2009년 10월 30일
심사완료일 : 2009년 11월 18일
교신저자 : 이우섭, e-mail : wsrhee@hanbat.ac.kr

I. 서 론

최근 초고속 인터넷의 발달과 이동통신 서비스의 성장으로 우리나라는 세계 최고 수준의 IT 강국으로 발전하였으나 기존 시장의 성장 한계와 신규 서비스의 활성화 미흡으로 인한 IT 산업의 성장이 둔화되고 있다. 이러한 성장 한계를 극복하기 위하여 방송, 통신 및 유무선 통합이 많은 주목을 받고 있다. 이러한 융합의 대세 속에서 방송과 통신의 융합을 주도하는 대표적인 서비스로 각광받고 있는 것이 IPTV 서비스이다. 사용자의 입장에서 기존의 IPTV 서비스는 인터넷 프로토콜을 이용하여 TV를 보는 정도로만 인식되었고 기존 CATV서비스와의 차별성이 거의 없어 언제 어디서나 고품질의 컨텐츠를 생성, 전달, 소비하는 서비스에 대한 필요성이 대두되고 있는 실정이다. 이를 위해 무한한 채널 수용능력, 유연한 미디어 확장성, 다양한 비즈니스 모델, 품질이 보장되는 차별화된 서비스의 형태로 발전해 나가고 있다[1]. 채널이 무한히 늘어나고 사용자의 요구가 다양해짐에 따라 네트워크상의 트래픽은 상당히 증가하고 있고 서비스의 특성에 맞는 품질 제어에 대한 요구가 증대되고 있다. 따라서 플로우 별 서비스 품질 제어 기술이 필요하게 되었다. 라우터의 입장에서도 이러한 상황을 제어하기 위하여 많은 버퍼 관리 알고리즘들이 연구되어 왔다[2][3]. IETF에서도 혼잡 발생 시 매우 큰 패킷 손실이 발생하는 것을 해결하기 위하여 RED 알고리즘과 같은 버퍼 관리 메커니즘을 제안하였으며 적절한 자원 할당을 위해 지금까지 여러 가지 버퍼 관리 및 스케줄링 기법이 사용되어 왔다. 하지만 앞서 살펴본 버퍼 관리 알고리즘들은 버퍼의 오버 플로우를 줄여서 전송률을 올리는 것에 목적이 있고 플로우별로 관리를 하더라도 다양한 멀티미디어 서비스를 위한 QoS 요구 사항을 직접적으로 반영하고 있지는 않다. 따라서 다양한 QoS를 제공하는 인터넷 망에서 효율적인 플로우 관리를 위하여 버퍼 관리 메커니즘에 이러한 요구 사항을 직접적으로 반영할 필요가 있다. 따라서, 본 논문에서는 현재 대두되고 있는 인터넷 멀티미디어 서비스들이 요구하는 QoS를 만족시킬 수 있는 능동적 버퍼 관리 알고리즘으로 QoS 적응형 플로우 기반 AQM 알고리즘을 제안한다. 이 알고리즘은 기존의 버퍼 관리 알고리즘들의 특징과 더불어 사용자의 QoS 요구 사항을 직접적으로 반영하여 각 플로우 별로 전송량과 무관하게 목표 패킷 손실률을 최대한 유지시켜준다.

이를 위해 2장에서는 RED, RIO, FRED 및 BLUE 등의 버퍼 관리 알고리즘들에 대해 설명하고 장단점들을 기술하였다. 3장에서는 본 논문에서 제안하는 QoS 적응형 플로우 기반 AQM 알고리즘에 대해 자세히 설명하고 4장에서는 ns-2를 이용한 성능 분석을 통하여 패킷 손실률, 버퍼 사용량의 관점에서 제안된 알고리즘의 우수성을 보였다. 그리고 마지막으로 5장에서 결론을 제시한다.

II. 버퍼 관리 알고리즘

현재까지 제안된 버퍼 관리 알고리즘들은 평균 버퍼 길이를 기준으로 하는 RED와 이를 변형 또는 보완한 것이다. 여기에는 RED[6]와 패킷을 In/Out으로 나누어 RED를 적용하는 RIO(RED with In/Out)[9], 여러 플로우에 대하여 RED를 적용하는 FRED 등이 있다. 또한 RED와 다르게 패킷 손실과 링크 사용률을 기준으로 하는 BLUE[11] 등이 있다.

RED는 버퍼의 길이가 임계값을 넘으면 평균 버퍼 길이 (avg)를 기준으로 계산된 확률로 패킷을 폐기하는 방식이다. 즉, RED 알고리즘은 패킷이 도착할 때마다 평균 버퍼의 길이를 구해 정해진 최소 임계값 (min_{th})과 최대 임계값 (max_{th})을 비교하여 최소 임계값 이하이면 정상적인 상태로 인식하고 두 임계값 사이이면 랜덤하게 패킷을 폐기하고 최대 임계값 이상이면 입력 패킷을 모두 폐기한다[4]. 평균 버퍼 길이는 다음과 같은 EWMA(Exponentially Weighted Moving Average)에 기반 하여 계산한다.

$$avg = avg + w \cdot (q - avg) \quad (1)$$

식 (1)에서 w는 가중치, q는 현재 버퍼 길이이다. w가 클 경우에는 현재의 버퍼 길이가 새롭게 구해지는 avg

에 미치는 영향이 커지며 작을 경우에는 그 반대이다. w가 작을수록 일시적인 폭주에도 avg가 크게 올라가지 않는다. avg가 min_{th}와 max_{th} 사이에 있으면 도착하는 모든 패킷을 확률적으로 폐기하게 되는데 avg가 min_{th}와 max_{th} 중 어느 쪽에 가까운가에 따라 확률이 결정된다.

$$P_b = Pmax \cdot (avg - min_{th}) / (max_{th} - min_{th}) \quad (2)$$

$$P_a = P_b / (1 - count \cdot P_b) \quad (3)$$

식 (2)에 의해 구해진 확률 P_b는 min_{th}부터 max_{th}까지 선형적으로 증가하며 최대값이 P_{max}인 확률이다. 하지만 실제 쓰이는 확률은 식 (3)에 의해 구해진 확률 P_a이며 이는 최근에 연속적으로 폐기되지 않은 패킷의 수 ($count$)를 세어 많을수록 확률을 조금씩 증가시키는 것이며 이는 한 패킷을 폐기하고 너무 오래 기다리는 것을 방지한다.

RED의 평균 버퍼 길이에 대한 패킷 폐기 확률은 [그림 1]과 같이 나타낼 수 있다. 그러나 RED는 플로우별 상태 (per-flow state)를 고려하지 않으며 버퍼의 상태만으로 동작하기 때문에 무분별한 패킷 폐기로 인해 플로우 간의 불공정성이 나타나게 된다[7][8].

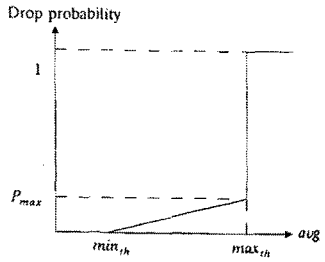


그림 1. RED의 패킷 폐기 확률

한편, RIO는 사용자 별 차별 서비스를 제공한다. SLA(Service Level Agreement)를 준수하는 패킷의 경우는 In-profile 패킷으로 구분하고 그렇지 않은 경우는 Out-profile 패킷으로 구분하여 각각 다른 RED를 적용한다. 즉, 하나의 버퍼에 In패킷과 Out패킷을 위한 두 가지 RED가 공존하게 된다. In 패킷의 min_{th}보다 Out 패킷의 min_{th}를 낮게 하여 In 패킷보다 Out 패킷이 먼저 랜덤하게 폐기된다. 또한 Out 패킷의 P_{max}가 In 패킷의 P_{max}보다 높게 하여 랜덤하게 폐기될 때 Out-profile 패킷이 폐기될 확률이 더 높게 된다. RIO 알고리즘은 대역을 지나치게 남용하는 사용자로부터 규약을 준수하는 사용자를 보호하여 공정한 대역의 사용을 보장하는 방법이다[9][10]. [그림 2]는 In과 Out에 따라 다른 패킷 폐기 확률을 보여준다.

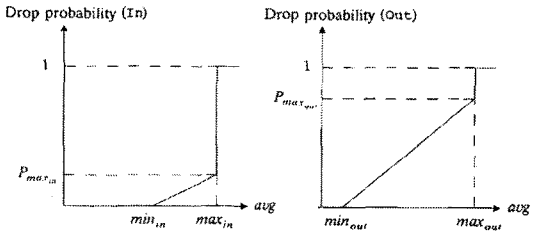


그림 2. RIO의 패킷 폐기 확률

앞서 기술한 바와 같이 RED에서는 플로우별 상태 관리가 되지 않고 여러 플로우가 한 개의 버퍼에서 관리되어 폐기되는 패킷이 어느 플로우의 패킷인지 고려되지 않는다. 따라서 상대적으로 느린 속도를 가지면서 안정적인 플로우의 패킷도 무분별하게 폐기되는 불공정성이 존재하게 되며 이는 RIO에서도 발생한다. 이와 같은 RED의 단점을 보완하기 위해 FRED는 패킷을 많이 보내는 플로우의 폐기율을 더 높게 하여 상대적으로 안정적인 플로우를 보호할 수 있는 알고리즘이다[8].

FRED 알고리즘은 플로우별로 관리하는 부분과 RED와 같은 랜덤 폐기 부분으로 이루어지는데 그 중 플로우를 관리하기 위하여 다음과 같은 부분이 추가되었다.

- 현재 버퍼에 있는 $flow_i$의 패킷이 max_q 이상이면 폐기한다.
- avg가 max_{th} 이상인데 $flow_i$ 패킷 수가 $2 * avgcq$ 보다 크면 패널티를 준다.
- 패널티를 2번 이상 받은 $flow_i$의 패킷 수가 $avgcq$ 이상일 때 폐기한다.

이는 FRED 알고리즘에서 플로우별 관리를 위하여 플로우의 특성에 따라 패킷의 폐기율을 달리하는 부분이다. 이를 위하여 파라미터 max_q, $avgcq$가 추가되었

다. 또한 플로우별 파라미터로 현재 버퍼에 있는 $flow_i$의 패킷 개수인 $qlen_i$와 패널티의 정도를 나타내는 $strike_i$ 변수가 추가되었다. 그러나 FRED는 플로우 간의 공정성은 향상시키지만 그 방법이 패킷을 많이 보내는 플로우에게 패널티를 주며 패널티를 많이 받은 플로우에게 또 패널티를 주는 것이므로 항상 빠른 속도의 플로우가 패널티를 받게 된다. 따라서 허용도를 플로우마다 개별적으로 적용할 수 없다.

BLUE는 RED와 다르게 패킷 손실과 링크 이용률(utilization)을 직접적으로 이용하며 임계값 없이 항상 단일 확률 P_m에 의해 폐기된다. 오버플로우로 인한 패킷 손실이 발생하면 P_m은 일정량($d1$)씩 증가하고 버퍼가 비거나 link idle상태에서 일정량($d2$)만큼 감소한다. P_m의 변화는 P_m의 최소 업데이트 시간(freeze_time)에 의해 이루어지며 최소 업데이트 시간 동안에는 패킷 손실이나 link idle의 상황이 발생해도 P_m은 변화하지 않는다. BLUE 알고리즘은 [그림 3]과 같이 간단하지만 RED보다 적은 폐기율을 갖는다. RED는 실제적으로 오버플로우가 나지 않을 플로우에 대해서도 확률적으로 폐기한다. 따라서 이런 경우에는 오버플로우가 날 때까지 폐기하지 않는 BLUE의 폐기율이 더 적게 된다[11]. 그러나 BLUE도 RED와 마찬가지로 라우터의 링크를 공유하고 있는 다양한 플로우들을 구분하지 않고 모두 동일한 기준에 의해 패킷 폐기를 결정하게 된다. 따라서 RIO와 유사하게 차별 서비스 제공을 위한 BIO (BLUE with In/Out)가 제안되었지만 RIO와 비교해서 뛰어난 성능이 나오지는 않았다[12].

```
Upon packet loss (or Qlen>L) event:
  if ( (now - last_update) > freeze_time ) then
    Pm = Pm + d1
    last_update = now
Upon link idle event:
  if ( (now - last_update) > freeze_time ) then
    Pm = Pm - d2
    last_update = now
```

그림 3. BLUE 알고리즘

ORANGE 알고리즘은 트래픽을 real-time 서비스 등의 높은 우선순위 서비스인 Class I 과 그 밖의 non real-time 서비스로 우선순위가 낮은 서비스인 Class II로 나누어 서비스를 제공하는 환경에서 두 클래스의 트래픽이 하나의 큐를 공유해서 쓰는 경우 real-time 서비스인 Class I 서비스의 품질을 보장하기 위한 버퍼 관리 알고리즘이다[14]. ORANGE는 버퍼를 congestion 상태, potential-congestion 상태, potentially-safe 상태, safe 상태 등 네 가지 상태로 구분하여 관리한다. 각 상태에 따라 확률적으로 Class II 서비스의 패킷을 폐기하며 Class I 서비스의 패킷은 어떤 상태에서도 임의로 폐기하지 않는다. 따라서 ORANGE는 클래스에 따른 QoS 제공이 가능하지만 real-time 서비스의 품질을 보장하는 것을 최우선으로 하고 있으므로 non real-time 서비스는 사용자의 요구 사항과 무관하게 real-time 서비스를 위해 희생되게 된다. 자세한 ORANGE 알고리즘은 [그림 4]와 같다. Class II 패킷은 M에 의해 확률적으로 폐기되게 된다. safe 상태에서는 M=0이 되어 Class II 패킷도 임의로 폐기되지 않으며 congestion 상태에서는 M=1이 되어 버퍼에 도착하는 모든 Class II 패킷이 폐기되게 된다. potential-safe 상태에서는 M=$M1$이 되어 비교적 낮은 확률로 Class II 패킷이 폐기되며, potential-congestion 상태에서는 M=$M2$가 되어 비교적 높은 확률로 Class II 패킷이 폐기되게 된다.

Parameters:
L : current queue length
Q : a threshold for traffic buffering
I : packet queuing rate
T : total buffer size
M : probability of drop class II data packet
Ib : number of incoming bits
Ob : number of outgoing bits
ML : maximum queue length

$$M1 = \frac{Q}{ML} \qquad M2 = \frac{L-Q}{ML-Q}$$

for each packet arrival:
if ($L >= T$) {
congestion 상태로 정의, $M = 1$ }
if ($Q < L < T$ and $I > 0$) {
potential-congestion 상태로 정의, $M = M2$ }
if ($Q < L < T$ and $((Ob - Ib) > (L - Q))$) {
safe 상태로 정의, $M = 0$ }
if ($0 < L < Q$ and $((T - Q) > I)$) {
safe 상태로 정의, $M = 0$ }
if ($0 < L < Q$ and $((T - Q) < I)$) {
potential-safe 상태로 정의, $M = M1$ }

그림 4. ORANGE 알고리즘

III. 제안된 알고리즘

본 논문에서 제안된 FQA(Flow based QoS Adaptive) 알고리즘은 RED 알고리즘처럼 두 임계값을 기준으로 확률적으로 패킷을 폐기하되 패킷 손실률이 임의로 정해진 목표 패킷 손실률에 수렴하도록 한다. 목표 패킷 손실률을 손실 허용치로 보아 평소에도 확률적인 패킷 폐기를 통하여 패킷 손실률을 유지시켜주며 폭주 상태에서도 목표 패킷 손실률을 초과하는 손실률이 발생하지 않도록 한다. 단, 버퍼의 상태가 최소 임계치 이하를 유지할 경우에는 패킷을 폐기하지 않는다. FQA 알고리즘은 각 플로우의 전송 속도와 관계없이 모두 동일한 패킷 손실률을 유지할 수도 있고 해당 플로우의 QoS 요구 사항에 따른 패킷 손실률을 만족시켜 줄 수도 있는 장점이 있다. FQA 알고리즘에서는 RED 알고리즘에서 사용되는 파라미터 중 avg, min_{th}, max_{th} 등 일부 파라미터를 동일하게 사용한다. 또한, 플로우별 변수로서 플로우별 패킷 손실률을 유지하기 위하여 해당 플로우의 입력 패킷수를 나타내는 *count*가 사용되며 플로우들을 구분하기 위한 상태 테이블 (State table)을 동적으로 관리하기 위하여 각 플로우의 버퍼 내에 존재하는 패킷 수(*qlen*)가 사용된다. 또한 목표 패킷 손실률(*LR*)은 모든 플로우가 동일하게 한 개의 값만 가질 수도 있고 플로우별 변수로 하여 각 플로우마다 다른 목표 패킷 손실률을 갖게 할 수도 있다. 패킷이 버퍼에 도착하면 RED 알고리즘과 같이 EWMA에 기반 하여 식 (1)을 통해 avg를 계산한다. 그리고 패킷의 폐기 여부를 결정하기 위하여 avg와 두 임계값을 비교하며 다음과 같이세 가지 상태로 정의할 수 있다.

- max_{th} <= avg 이면 모든 패킷을 폐기하며 이를 all-drop 상태로 정의한다.
- avg < min_{th} 이면 모든 패킷을 받아들이며 이를 non-drop 상태로 정의한다.
- min_{th} <= avg < max_{th}일 때 목표 패킷 손실률을 유지하면서 패킷을 확률적으로 폐기하며 이를 random-drop 상태로 정의한다.

이 때 사용되는 폐기 확률 P는 *count*가 0 이하 일 때 0이고 $1/LR$일 때 1까지 선형적으로 증가한다. 즉 P는 *count*가 0 이하 일 때 0이며 1 이상일 때 $LR * count$로 정해진다.

$$P = LR * count \tag{4}$$

확률 P로 인해 패킷이 폐기되었을 경우 *count*를 $1/LR$만큼 감소시켜 목표 패킷 손실률을 유지할 수 있도록 한다. 이때 *count* 값이 $1/LR$ 일 경우 폐기 확률이 1이 되어 무조건 패킷이 폐기되고 *count* 값은 $1/LR$을 감소시켜 0이 된다. 그러나, *count*값이 $1/LR$보다 작을 경우 패킷이 식 (4)에 의해 확률 P에 의해 폐기되면 식 (5)에 의해 *count* 값이 음수가 된다. 이때 *count* 값이 음수가 된다는 것은 *count* 값이 0이 될 때 까지 해당 플로우에서 입력되는 패킷은 폐기되지 않는다는 의미이다.

with probability P:
  drop the arriving packet; (5)
  count = count - 1/LR;

따라서 *count* 변수와 패킷 폐기 확률 P의 관계는 [그림 5]와 같이 표현할 수 있다. 즉 해당 플로우의 입력 패킷은 *count* 값이 음수일 경우 폐기되지 않으며 양수일 경우에만 확률 P로 폐기된다.

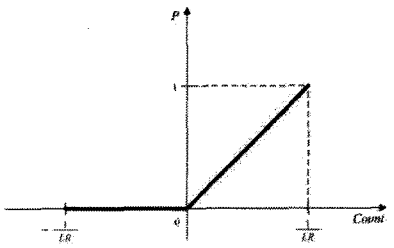


그림 5. count 변수에 따른 패킷 폐기 확률 P의 변화

또한 FQA 알고리즘에서는 버퍼의 이동 방향(drift) 개념을 통하여 패킷 폐기를 좀 더 효율적으로 할 수 있

도록 하였다. 버퍼 이동 방향은 버퍼의 현재 상태만 보지 않고 과거 상태와 비교하여 동향을 적용하는 방법이다. 사용된 버퍼 이동 방향 방법은 [10]의 "Buffer states and feedback mechanism"과 유사하지만 피드백을 하지 않으며 두 번에 걸친 변화량 측정을 통해 최종적인 변화량을 계산하게 된다. avg를 기준 값으로 사용했으며 일정 시간마다 측정된 변화량을 적용하도록 하였다. 따라서 버퍼의 상태 변화는 [그림 6]과 같이 두 임계값과 avg와의 비교뿐만 아니라 버퍼의 동향을 나타내는 버퍼 이동 방향 벡터(v)도 적용되어 나타나게 된다.

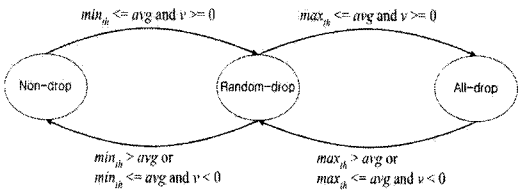


그림 6. 버퍼 이동 방향에 따른 버퍼 상태 변화

다음 [그림 7]은 FQA 알고리즘에서 사용되는 파라미터들을 기술하였고 [그림 8]은 자세한 알고리즘을 보여준다.

Per-flow variables:
$count_i$: 플로우 i 가 보낸 입력 패킷 수
$qlen_i$: 버퍼내의 플로우 i 패킷 수
LR_i : 플로우 i의 target loss ratio
p_i : 플로우 i의 패킷 drop 확률

Global variables:
q : 현재 버퍼 길이
min_{th} : 최소 임계값
max_{th} : 최대 임계값
w : 가중치
avg : 평균 길이

Buffer drift variables:
v : 최종 변화 벡터
v_{t-1} : 이전 벡터
v_t : 현재 벡터
avg_{t-1} : 이전 avg 값
Δt : 단위 시간

그림 7. FQA 알고리즘의 파라미터

for each packet arrival:
if the packet is new $flow_i$ {
initialization;
$count_i = 0$;
$qlen_i = 0$; }
Calculate new average buffer length avg ;
$avg = (1 - w) * avg + w * q$;
Increment $count_i$;
if ($count_i$>0 && ((v>=0 && min_{th} <= avg <max_{th}) || (v < 0 && max_{th} <= avg))){
calculate probability p_i :
$p_i = LR_i * count_i$;
with probability p_i :
Drop the arriving packet;
$count_i = count_i - 1/LR_i$;
return; }
else if (v >= 0 && max_{th} <= avg){
Drop the arriving packet;
return; }
Insert the arriving packet into the buffer;
Increment $qlen_i$;
for each packet departure:
Calculate new average buffer length avg ;
$avg = (1 - w) * avg + w * q$
Decrement $qlen_i$;
Calculate Buffer drift (Δt):
$v_t = avg - avg_{t-1}$
$v = v_t - v_{t-1}$
$v_{t-1} = v_t$
$avg_{t-1} = avg$

그림 8. FQA 알고리즘

FQA 알고리즘에서는 어느 플로우든지 패킷 손실률이 목표 패킷 손실률(LR)에 가깝게 유지될 수 있도록 해준다. 만약 플로우의 LR이 10^{-3}으로 설정되어 있다면 그 플로우가 보내는 패킷들은 $1 / LR$개, 즉 1000개 중에 1개가 폐기되게 된다. 하지만 버퍼가 안정된

non-drop 상태에서는 패킷을 폐기하지 않으므로 실제로는 1000개 중에 1개 이하가 폐기되게 된다. 이는 플로우의 전송 속도와 관계없이 원하는 패킷 손실률을 유지할 수 있음을 의미한다. 즉, 전송 속도가 높지만 패킷 손실률이 아주 낮아야 하는 실시간의 고화질 영상 서비스나 반대의 경우인 플로우도 모두 적용이 가능하며 다양한 서비스들을 위해 모든 플로우마다 목표 패킷 손실률을 다르게 적용하거나 서비스 클래스별로 그룹화 시켜서 적용하는 것도 가능한 장점이 있다.

한편, RED나 FRED 알고리즘과 같이 서서히 변화하는 확률을 통해 패킷을 폐기할 경우에는 한번 높아진 확률이 다시 낮아지는 동안 높은 확률에 의해 패킷이 집중적으로 폐기될 가능성이 있다. 하지만 FQA 알고리즘에서는 random-drop 상태에서의 집중적인 패킷 폐기가 나타나지 않으며 n번째 패킷이 폐기되었을 때 그 다음 $(1 / LR) - n$개까지의 패킷을 수용하여 패킷 손실률의 유지와 함께 확률에 의한 집중적인 패킷 폐기를 막아준다. 따라서 버퍼 오버플로우로 인한 무조건적인 폐기를 제외하면 $2 / LR$개의 패킷 중 최대 2개까지만 연속적으로 폐기될 수 있으며 발생 확률도 낮으므로 확률에 의한 집중적인 패킷 폐기가 일어나지 않는 장점이 있다. 또한, FQA 알고리즘에서는 버퍼 이동 방향 벡터를 사용하여 불필요한 패킷 폐기와 all-drop 상태에서의 무조건적인 패킷 폐기를 줄였다. 즉, avg가 max_{th}를 넘었을 때도 버퍼 이동 방향 벡터 값을 통해 측정된 버퍼의 동향이 하향세이면 지속적으로 버퍼에 쌓이는 패킷이 없는 것으로 판단하여 무조건적인 폐기를 피하게 된다.

IV. 성능 분석

1. 시뮬레이션 환경

본 논문에서 제안된 FQA 알고리즘의 성능 분석을 위해 네트워크 시뮬레이터 ns-2를 이용하였다[13]. 비교 대상으로는 유사하게 최소, 최대 임계값을 사용하는 RED, FRED 알고리즘과 클래스별 QoS 제공이 가능한 ORANGE 알고리즘을 택했다. 사용된 프로토콜은 UDP이고 입력 트래픽은 지수 분포 (Exponential distribution) 트래픽을 사용하였으며 시뮬레이션 파라미터는 [표 1]에 기술되었다.

표 1. 시뮬레이션 파라미터

파라미터	값
Buffer size	100
min_{th}	25
max_{th}	70
min_q	4
P_{max}	0.02
w	0.002
LR	0.001

여기서 P_{max}는 RED, min_q는 FRED, LR은 FQA 알고리즘 전용 파라미터이다. 한편, 본 논문에서 사용된 시뮬레이션 모델은 [그림 9]와 같다. 이 모델은 지수 분포 트래픽을 발생시키는 소스들이 있으며 전송률에 따라 256 Kbps, 512 Kbps, 1 Mbps, 2 Mbps 네 가지 그룹으로 나눌 수 있다. 각각의 소스들은 성능 분석용 버퍼가 있는 에지 라우터를 거쳐서 목적지 노드까지 패킷을 전송한다. 이 때 각각의 소스들과 라우터간의 링크 대역폭은 이 구간에서 패킷 손실이 없을 정도로 충분하다고 가정하고 에지 라우터와 목적 노드와의 링크 대역폭을 10 Mbps로 하여 병목현상이 발생될 수 있도록 하였다. 부하는 출력 링크 (10 Mbps) 대비 입력 트래픽의 비율로 정의하였고 입력 부하를 증가시켜야 할 경우에는 각 소스의 입력 플로우 개수를 늘려 정의하였다.

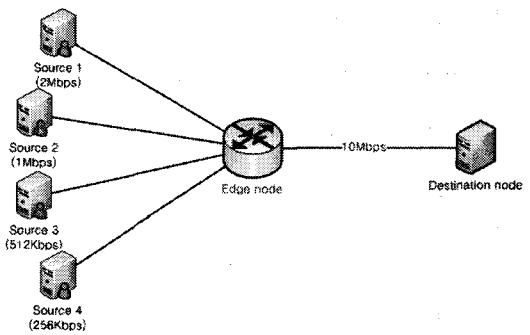


그림 9. 시뮬레이션 모델

2. 시뮬레이션 결과

시뮬레이션 결과는 부하 별로 각 플로우의 패킷 손실률을 측정하였으며 모든 패킷이 폐기되지 않는 non-drop과 random-drop 상태를 대상으로 측정하였다.

[그림 10]은 RED에 대한 패킷 손실률을 보여주고 있다. RED는 각 플로우별로 상태 관리를 하지 않기 때문에 많이 보내면 많이 폐기되고 적게 보내면 적게 패킷이 폐기된다. 또한 각 플로우별로 측정된 패킷 손실률도 비슷하게 폐기되며 부하가 커질수록 패킷 손실률도 커지게 되는 것을 알 수 있다.

[그림 11]은 FRED에 대한 패킷 손실률을 보여주고 있는데 FRED는 기본적으로 RED의 특성과 유사하게 부하가 커질수록 패킷 손실률도 커지는 결과를 나타낸다. 또한 전송 속도가 높은 플로우에게 패널티를 주어 더 많이 폐기하고 반대로 전송 속도가 낮은 소스는 거의 폐기하지 않는 FRED의 특징도 잘 반영되어 있음을 알 수 있다.

[그림 12]는 BLUE에 대한 패킷 손실률을 보여주고 있는데 BLUE는 RED/FRED와 달리 min과 max를 사용하지 않고 전 구간에 대하여 확률적인 폐기를 한다. 따라서 손실률을 측정하는 구간이 달라 절대적인 비교가 되지 않는다. BLUE는 부하가 커질수록 손실률이 매우 높아지는 것을 확인할 수 있으며 이것은 손실이 발생하면 폐기확률을 높여주는 방식이기 때문이다. 또한 플로우별 관리를 하지 않기 때문에 소스의 특성에 구분없이 모든 플로우의 손실률이 비슷하게 나타난다.

[그림 13]은 FQA 알고리즘에 대한 패킷 손실률을 보여주고 있는데 모든 플로우의 목표 패킷 손실률을 10^{-3}으로 정하였다. [그림 13]에서 보는 바와 같이 모든 플로우들에 대해 측정된 패킷 손실률이 목표 패킷 손실률로 설정한 10^{-3}을 넘지 않도록 유지되고 있음을 알 수 있다. 이는 모든 플로우의 패킷 손실률은 목표 패킷 손실률에 맞춰지도록 폐기 확률이 결정되기 때문이다.

한편, [그림 14]에서는 FQA 알고리즘에서 각 플로우별 목표 패킷 손실률을 다르게 설정하여 플로우별 패킷 손실률을 조절할 수 있는 결과를 보여 주고 있다. 각 플로우들의 목표 패킷 손실률은 2 Mbps 플로우의 경우 10^{-4}, 1 Mbps 플로우는 10^{-3}, 512 Kbps와 256 Kbps 플로우는 10^{-2} 으로 설정하였다. 이 시뮬레이션 결과에서 각 플로우들의 전송 속도와는 관계없이 플로우별 목표 패킷 손실률이 잘 지켜지고 있음을 볼 수 있다.

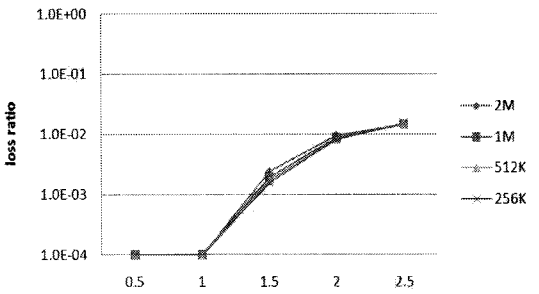


그림 10. 부하에 따른 RED의 패킷 손실률

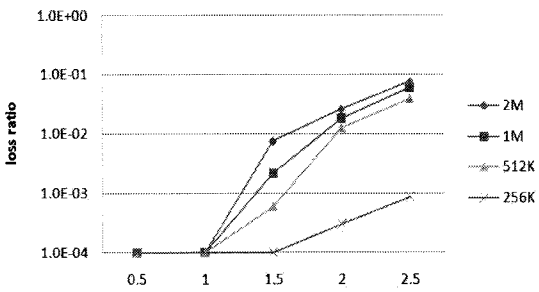


그림 11. 부하에 따른 FRED의 패킷 손실률

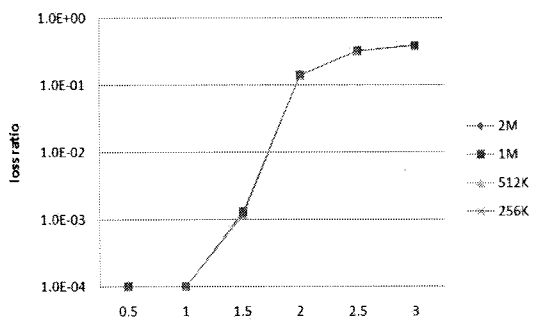


그림 12. 부하에 따른 BLUE의 패킷 손실률

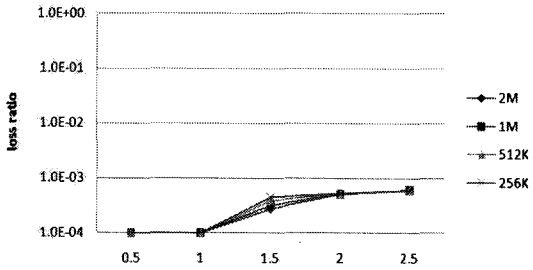


그림 13. 부하에 따른 FQA의 패킷 손실률

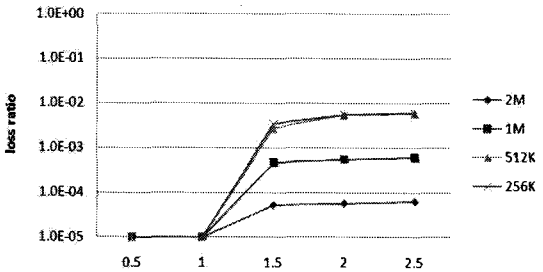


그림 14. 플로우별 목표 패킷 손실률에 따른 패킷 손실률

다음은 각 플로우들을 클래스별로 나누어 QoS를 제공할 때, 클래스별 패킷 손실률을 측정한 결과이다. 이때, Class I 은 Controlled load 서비스에서 어느 정도 품질이 보장되는 real-time 서비스이며 Class II 는 Controlled load 서비스에서 Class I 서비스를 제외한 non real-time 서비스로 분류 된다.

[그림 15]는 클래스별 RED의 패킷 손실률을 보여준다. RED는 플로우별 관리는 물론 클래스별 QoS도 제공하지 않으므로 클래스와 관계없이 균일하게 패킷 손실이 발생했다.

[그림 16]은 클래스별 FRED의 패킷 손실률을 보여준다. FRED는 클래스의 특성과는 관계없이 플로우별 트래픽 사용량에 관계하여 차등 관리를 하게 된다. 따라서 스트리밍 서비스 등 real-time 서비스를 제공하는 Class I 서비스가 오히려 패킷 손실률이 높아지는 것을 볼 수 있다. 이것은 클래스와 관계없이 단지 Class II 서비스보다 Class I 서비스가 트래픽을 더 많이 사용하게 되어 발생하는 결과이다.

[그림 17]은 클래스별 ORANGE의 패킷 손실률을 보여준다. ORANGE는 클래스별 QoS 제공이 가능한 알고리즘이며 특히 real-time 서비스의 품질을 보장해주기 위한 알고리즘이다. 따라서 [그림 17]에서 보이는 바와 같이 real-time 서비스인 Class I 의 패킷 손실률은 0에 가깝게 나타난다. 하지만 non real-time 서비스인 Class II 에 대해서는 품질을 보장해주지 않으므로 다른 알고리즘에 비하여 다소 높은 패킷 손실률이 나타나는 것을 볼 수 있다.

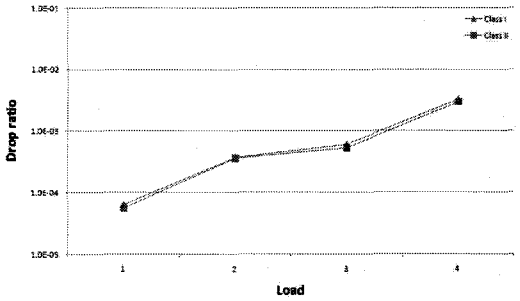


그림 15. 클래스에 따른 RED의 패킷 손실률

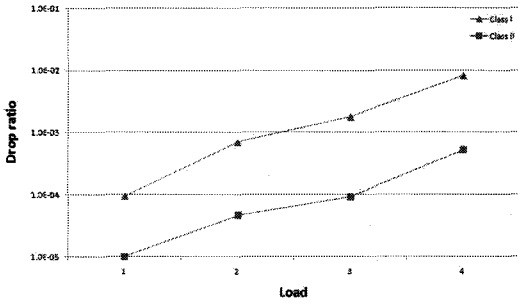


그림 16. 클래스에 따른 FRED의 패킷 손실률

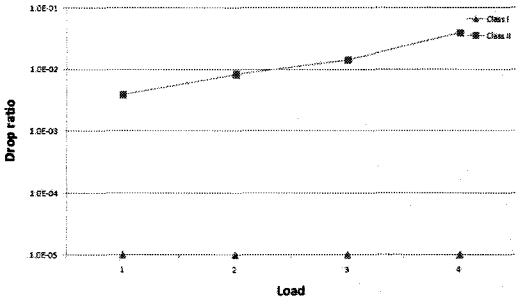
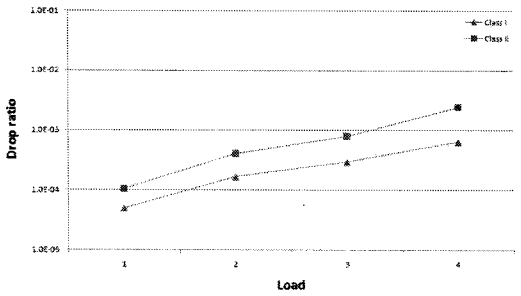
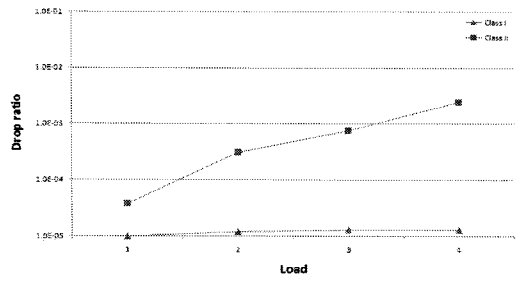


그림 17. 클래스에 따른 ORANGE의 패킷 손실률

[그림 18]은 클래스별 FQA의 패킷 손실률을 보여준다. [그림 18](a)는 Class I 의 목표 패킷 손실률은 10^{-3}, Class II 의 목표 패킷 손실률은 10^{-2}으로 설정한 결과이며 클래스별 패킷 손실률이 목표 패킷 손실률을 넘지 않도록 유지되는 것을 볼 수 있다. [그림 18](b)는 [그림 17]의 ORANGE 알고리즘과 같이 Class I 의 낮은 패킷 손실률 보장을 위해 Class I 의 목표 패킷 손실률을 10^{-5}, Class II 의 목표 패킷 손실률은 $5*10^{-3}$로 설정한 결과이다. 그림에서와 같이 Class I 의 목표 패킷 손실률을 크게 낮춤으로써 ORANGE 알고리즘을 사용했을 때와 비슷하게 품질이 보장되는 것을 볼 수 있다. Class II 또한 ORANGE 알고리즘을 사용했을 때에 비하여 품질이 보장되기 때문에 더 낮은 패킷 손실률이 나타나는 것을 볼 수 있다. 이와 같이 제안된 알고리즘에서는 IPTV 와 같이 실시간 성과 높은 품질을 요구하는 서비스를 지원할 수 있으며 동시에 데이터 서비스에 대해서도 ORANGE 알고리즘에 비해 서비스가 요구하는 최소한의 품질을 보장할 수 있는 알고리즘이다.



(a) Class I 의 목표 패킷 손실률 : 10^{-3}, Class II 의 목표 패킷 손실률 : 10^{-2}



(b) Class I 의 목표 패킷 손실률 : 10^{-5}, Class II 의 목표 패킷 손실률 : $5*10^{-3}$

그림 18. 클래스에 따른 FQA의 패킷 손실률

본 논문에서 제안하는 FQA 알고리즘은 전송 데이터의 특성이나 전송 속도에 관계없이 플로우별 목표 패킷 손실률을 설정 할 수 있다. [그림 19]와 [그림 20]은 Class II에서 1Mbps로 전송속도가 동일한 플로우 세 개에 대하여 목표 패킷 손실률을 다르게 하고 측정한 결과이다.

[그림 19]는 ORANGE를 사용했을 때의 패킷 손실률이다. ORANGE는 non real-time 서비스에 해당하는 Class II 플로우에 대하여 어떤 플로우든 관계없이 품질 보장을 하지 않아 높은 패킷 손실률을 보이며 또한 플로우의 특성과 무관하게 서로 비슷한 패킷 손실률을 보이는 것을 알 수 있다. [그림 20]은 FQA를 사용했을 때의 패킷 손실률이다. 1번 플로우의 목표 패킷 손실률은 10^{-4}로 설정하였고 나머지 2,3번 플로우의 목표 패킷 손실률은 $5*10^{-3}$로 설정한 결과이다. FQA는 같은 대역폭을 갖은 플로우라도 서로 다른 목표 패킷 손실률을 가질 수 있게 된다. 따라서 [그림 20]에서 보이는 바와 같이 동일한 non-real time 서비스인 Class II 서비스 중에서도 특정한 플로우에 대해서 품질 보장 정도를 조절할 수 있는 것을 알 수 있다.

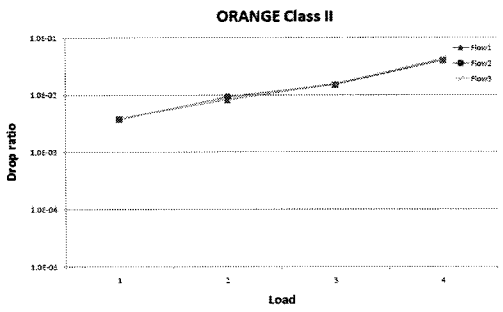


그림 19. ORANGE에서 Class II 의 각 플로우별 패킷 손실률

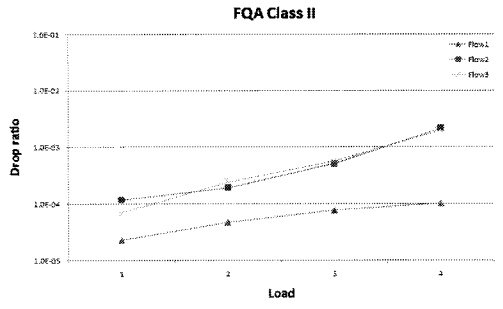


그림 20. FQA에서 Class II 의 각 플로우별 패킷 손실률

V. 결 론

방송통신 융합의 대세 속에서 IPTV 서비스는 차세대 멀티미디어 서비스로 각광받고 있다. IPTV 서비스는 무한한 채널수용과, 미디어의 확장성, 품질이 보장되는 서비스의 형태로 발전되어 나가고 있으며 이는 네트워크의 트래픽 증가와 서비스의 특성에 맞는 품질제어 기술을 요구한다. 따라서 플로우 별 서비스 품질 제어 기술이 필요하다. 본 논문에서는 효율 적인 혼잡 제어를 위하여 IETF에서 권고한 RED 알고리즘과 이 RED 알고리즘의 문제점을 해결하기 위해 제안된 FRED, RIO, BLUE 및 ORANGE 알고리즘들을 간단히 살펴보았다. 그리고 기존 버퍼 관리 알고리즘들의 장점을 유지함과 동시에 다양한 멀티미디어 서비스의 QoS 요구 사항을 직접적으로 반영할 수 있는 FQA 알고리즘을 제안하였다. FQA알고리즘은 기존 버퍼 관리의 목적을 유지하는 동시에 다양한 멀티미디어 서비스의 QoS 요구 사항을 직접적으로 반영하여 각 플로우별 버퍼량과 무관하게 목표 패킷 손실률을 최대한 유지시켜준다. 또한 본 논문에서는 ns-2를 이용한 시뮬레이션을 통하여 RED 알고리즘, FRED 알고리즘, BLUE 알고리즘, ORANGE 알고리즘 및 FQA 알고리즘 간의 성능을 비교 분석 해보았다. FQA 알고리즘은 전송량과 무관하게 각 플로우의 손실률을 목표 손실률 이하로 유지할 수 있다는 것을 제시하였다. 따라서 IPTV 서비스와 같은 차세대 멀티미디어 서비스의 다양한 요구사항을 만족시킬 수 있음을 확인하였다.

참 고 문 헌

[1] 심진보, 김유진, "IPTV 2.0 진화에 기초한 IPTV 수용-확산 영향요인 연구," Telecommunications Review, Vol.19, No.2, 2009(4),

[2] J, H. Koo, W. C. Choi, and K. S. Chung, "A New Queue Management Algorithm Improving Fairness of the Internet Congestion Control," 한국정보과학회논문지, 정보통신, Vol.30, No.3, pp.437-447, 2003(6).

[3] P. J. Lee and J. Y. Yang, "A Packet Dropping Algorithm based on Queue Management for Congestion Avoidance," 한국인터넷정보학회논문지, Vol.3, No.6, pp.43-51, 2002(12),

[4] A. K. Min and J. B. Suk, "A Traffic Management Scheme for the Scalability of IP QoS," 한국정보과학회논문지, 정보통신, Vol.29, No.4, pp.375-385, 2002(10).

[5] S. C. Park, "DiffServ QoS Support in DSL Broadband Access Networks," 한국정보처리학회 논문지, Vol.13, No.5, pp.613-620, 2006(10).

[6] B. Braden, "Recommendations on Queue Management and Congestion Avoidance in the Internet," RFC 2309, 1998(4).

[7] J. H. Nam, "Improve ARED Algorithm in TCP/IP Network," 한국컴퓨터정보학회논문지, Vol.12, No.3, pp.177-183, 2007(7).

[8] L. Dong and M. Robert, "Dynamics of Random Early Detection," Proc. ACM SIGCOMM, pp.127-137, 1997(9).

[9] David D. Clark and WenjiaFang, "Explicit Allocation of Best-Effort Packet Delivery Service," IEEE/ACM Transactions on Networking, Vol.6, No.4, pp.362-373, 1998(8).

[10] H. Jonathan Chao and Xiaolei Guo, *QUALITY OF SERVICE CONTROL IN HIGH-SPEED NETWORKS*, Wiley-Interscience Pub, pp.214-297, 2002.

[11] W. C. Feng, Kang G. Shin, Dilip D. Kandlur and Debanjan Saha, "The BLUE Active Queue Management Algorithms," IEEE/ACM Trans. of Networking, Vol.10, No.4, pp.513-528, 2002(8).

[12] O. Bahri and O. Sema, "BIO: An Alternative to RIO," SPIEIT Com: Int'l Symp. Convergence of IT and Comm., Denver, USA, pp.4524-4532, 2001(8).

[13] http://www.isi.edu/nsnam/ns/.

[14] S. W. Kim and S. C. Kim, "An Online Buffer Management Algorithm for QoS-Sensitive Multimedia Networks," ETRI Journal, Vol.29, No.5, pp.685-687, 2007(10).

저 자 소 개

강 현 명(Hyun-Myung Kang) 준회원

- 2008년 2월 : 국립한밭대학교 멀티미디어공학전공(공학사)
- 2008년 3월 ~ 현재 : 국립한밭대학교 멀티미디어공학과 석사과정

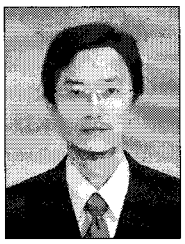
<관심분야> : 게임엔진, IP QoS, Mobility management, Mobile 멀티캐스팅

최 환 석(Hoan-Suk Choi) 준회원

- 2009년 2월 : 국립한밭대학교 멀티미디어공학전공(공학사)
- 2009년 3월 ~ 현재 : 국립한밭대학교 멀티미디어공학과 석사과정

<관심분야> : IP QoS, Mobility management, Mobile 멀티캐스팅

이 우 섭(Woo-Seop Rhee) 종신회원



- 1983년 2월 : 홍익대학교 전자계산학과(공학사)
- 1995년 8월 : 충남대학교 전산과학과(공학석사)
- 2003년 8월 : 충남대학교 컴퓨터과학과(공학박사)
- 1983년 3월 ~ 2005년 3월 : 한국전자통신연구원 광대역통합망연구부 팀장/책임연구원
- 2005년 3월 ~ 현재 : 국립한밭대학교 정보통신컴퓨터공학부 멀티미디어공학전공 교수
- 정통부 지정 국제표준 전문가, ITU-T SG13 한국대표, TTA 광인터넷 PG 의장 역임

<관심분야> : 유무선통합망구조, 멀티캐스팅, IP QoS, Mobility management, Mobile 멀티캐스팅