논문 02-27-11C-11 한국통신학회논문지 '02-11 Vol.27 No.11C

멀티미디어 서비스를 위한 효율적인 대역폭 할당

정회원 김 정 택*, 고 인 선**

An Efficient Algorithm of Network Bandwidth Allocations for Multimedia Services

Jung-taek Kim*, Inseon Koh** *Regular Members*

요 약

멀티미디어 서비스 제공에는 네트워크의 효율적인 대역폭 할당 스케줄링이 요구된다. 비디오 데이터 스트림 같은 burstiness 특성을 갖는 정보를 전송하기 위해서는 전송률의 변화를 감소시키는 smoothing 기법을 사용하게 되는데, 현재까지 제안된 최적 smoothing 알고리즘은 peak rate를 크게 줄이고 있지만, 각각의 독립적인 데이터 스트림에 적용되었기 때문에, 네트워크 대역폭이나 프로세서 등을 공유하는 다중 스트림에 사용하기에는 적합하지 않다. 본 논문에서는 이러한 문제점의 해결을 위한 효율적인 대역폭 할당 알고리즘을 제안한다. 대역폭 할당 스케줄링을 정적 대역폭 할당과 동적 대역폭 할당으로 구분해서, 정적 대역폭 할당에서는 사용자 버퍼의 누적 데이터 양을 높게 유지시킴으로써 peak rate를 감소시켰다. 동적 대역폭 할당에서는 real demand factor를 새로운 변수를 이용하여 다중 스트림의 대역폭할당에 적합한 알고리즘을 구현하였다. 마지막으로, 페트리 네트 모의실험 도구인 ExSpect 6.41을 통해서 제안된 알고리즘의 성능을 입증하였다.

ABSTRACT

Providing a multimedia service requires efficient network bandwidth allocation scheduling. Typically, transmission of information with bursty characteristics, such as a video data stream, utilizes some type of smoothing technique that reduces the transmission rate fluctuation. While capable of a drastic reduction in the peak transmission rate, even the optimum smoothing algorithm proposed to date, however, is rather inadequate for multi-stream applications that share network bandwidth since such an algorithm has been designed and applied toward handling each independent stream. In this paper, we proposed an efficient bandwidth allocation algorithm that addresses these shortcomings. The proposed bandwidth allocation algorithm is divided into two steps, a static bandwidth allocation and a dynamic bandwidth allocation. In the former case, the peak rate reduction is achieved by maintaining the accumulated data in the user buffer at a high level, whereas the concept of real demand factor is employed to meet the needs of multi-stream bandwidth allocation in the latter case, Finally, the proposed algorithm's performance was verified with ExSpect 6.41, a Petri net simulation tool.

I. 서 론

고속 네트워크를 이용하는 사용자의 급속한 증가는 네트워크를 통한 다양하고, 양질의 정보 흐름 서비스를 요구하고 있다. 이에 따라서, VoD(Video on Demand), LoD(Lecture on Demand), 화상회의 같은 멀티미디어 정보를 제공하는 서비스의 기술에 관한 연구가 활발히 진행되고 있다 이에 관련된 여러 연구 분야에서, 서비스 제공자의 서버에 할당된 제한적인 네트워크 대역폭을 효율적으로 할당하여 정보흐름을 원활히 할 수 있는 연구도 매우 중요한 한 분야를 차지하고 있다.

* 홍익대학교 전자공학과 지능제어 연구실(lamiar@orgio.net),
** 홍익대학교 전자공학과 지능제어 연구실(inseon@wow.hongik.ac.kr)
논문번호 : 020347-0808, 접수일자 : 2002년 8월 8일

일반적으로 압축되지 않은 비디오 데이터를 전송하는 경우에는 약 100Mbps의 대역폭 할당이 필요하다. 이는 현재 설치되어 있는 네트워크의 물리적 용량을 초과하므로, 대부분의 경우 비디오 데이터를 MPEG으로 압축하여 사용자에서 전송하고 있다. MPEG 코딩 알고리즘은 세 개의 다른 타입: I-frame, P-frame, B-frame을 사용하고 있는데, 세 가지 타입의 프레임들은 GOP(Group of Pictures)라 일컬어지는 그룹을 구성한다. 각 그룹은 적어도 하나의 I-frame과 선택적으로 더 많은 P와 B프레임으로 구성된다. MPEG은 CBR (Constant Bit Rate)와 VBR(Variable Bit Rate) 유형의 코딩이 가능하다. CBR 방식으로 코딩된 비디오 데이터 흐름은, 대역폭 할당이 용이하다는 장점이 있지만, QoS가 떨어지기 때문에, 원하는 품질을 보장하기 위해서는 VBR 방식으로 코딩된 비디오 데이터를 사용하게 된다. 이러한 방식의 MPEG 비디오 데이터 흐름은 필연적으로 burstiness를 갖게 된다.

네트워크 상의 데이터 전송 속도의 변화는 위에서 언급한 특성을 지닌 VBR 비디오 데이터 전송에 큰 영향을 미치고 있다. 일반적으로 전송률의 변화는 네트워크의 부하에 부정적인 영향을 미친다고 알려져 있다. 이러한 영향을 제거하기 위해서 전송률의 변화를 완화시키는 여러 가지 smoothing 알고리즘이 개발되었다. Smoothing 알고리즘은 사용자 버퍼의 제한조건이 만족하는 상황에서 가능한 오랫동안 CBR(Constant Bit Rate) 대역폭을 할당하는 것을 목표로 하고 있다. 이러한 방식으로 부여된 데이터 흐름은 네트워크를 통해서 전송되는 동안 다른 데이터 흐름들에게 영향을 적게 미치게 되고, 네트워크에서의 대역폭 할당을 용이하게 한다. 이 알고리즘은 미리 저장되어져 있는 비디오 데이터에 대해서 일괄적으로 적용이 가능하기 때문에, 전송하기 전에 smoothing을 보장하는 데이터 흐름의 전송률을 구할 수 있다.

Smoothing 알고리즘은 크게 세 가지로 나뉘어진다. 첫째, 시간 다중화 방식에 의한 것으로 스트림의 경로에 있는 각각의 버퍼를 통해서 스트림의 최대 값을 줄이는 것이다. 둘째, 확률 다중화 방식에 의한 smoothing은 다중의 독립적인 스트림이 주어진 자원(예로, 프로세서 혹은 네트워크 대역폭)을 공유함으로써 이루어진다. 마지막으로, Work -ahead에 의한 smoothing은 미리 작업된 스케줄링에 의한 데이터 전송을 통해 이루어진다. 비디오 데이터 전송에는 특성상 Work-ahead 방식이 가장 적절하므로 본 논문에서는 마지막에 언급한 smoothing 알고리즘[1][2]에 관하여 연구하였다.

Smoothing 알고리즘이 갖는 문제점은 할당된 전송률이 다른 데이터 흐름에 대해서 독립적으로 계산되었기 때문에, 한정된 대역폭을 갖는 서버에서 적용하기에 부적절하다. 본 논문에서는 이런 문제점들을 해결하기 위한 다중 스트림에 적합한 대역폭 할당 알고리즘을 Real Demand Factor라는 새로운 변수를 이용하여 제안하였다.

본 논문은 다음과 같이 구성되어 있다. Ⅱ장에서는 VoD서비스의 모델과 대역폭 할당에 대해서 설명하고, Ⅲ장에서는 제시된 멀티미디어 서비스에 적합한 알고리즘을 제안한다. 제안한 대역폭 알고리즘을 대역폭 할당을 정적대역폭 할당과 동적 대역폭 할당으로 구분해서 설명하였다. Ⅳ장에서는 페트리네트 기반의 모의실험 도구인 ExSpect를 이용한 모의실험 결과와, 제안한 알고리즘의 성능 분석을 하였다. Ⅴ장에서 이 논문의 결론을 맺는다.

Ⅱ. VoD 서비스 시스템

본 논문에서 제안한 알고리즘의 이해를 돕기 위해, 알고리즘이 적용되는 시스템의 개괄적인 구조를 그림 1에 나타내었다. 본 연구의 목적은 그림 1에서 스케줄러(Scheduler)에 들어가는 알고리즘을 설계하는 것이다.

그림 1의 상단 부분은 멀티미디어 서비스 시스템 중의 하나인 VoD 서비스 시스템을 나타내고, 하단은 본 논문에서 제안한 알고리즘을 시스템에 적용하는 방법을 개괄적으로 보여주고 있다. Stored video에는 MPEG 형태의 비디오 데이터들이 저장되어 있고, 각 비디오 데이터의 스트림 정보에 의해 미리 데이터의 전송률의 변화를 어느 정도 일정하게 하는 정적 대역폭 할당(Static bandwidth allocation) 값을 스케줄러에게 넘겨준다. 스케줄저는 가상 버퍼(Virtual buffer)의 각 비디오 데이터의 누적된 상태와 네트워크의 상태에 따라서 전송 유닛의 대역폭을 할당하게 된다. 본 논문에서는, 제공되는 비디오 데이터들의 스트림 정보에 의해 미리 계산된 스케줄링을 정적 대역폭 할당이라 하였고, 가상 버퍼와 네트워크의 상태에 따라서 대역폭을 할당하는 것을 동적 대역폭 할당이라고 하였다. 그림 1에서 보는 바와 같이, 양질의 비디오 데이터를 사용자에게 제공하기 위해서는 burstiness를 어느 정도 줄일 수 있는 정적 대역폭 알고리즘과, 네트워크

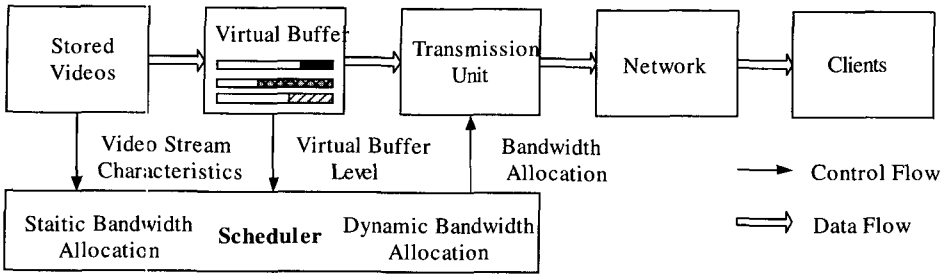


그림 1. VoD 시스템.

의 상태에 따라서 멀티미디어 데이터 스트림의 효율적인 대역폭 할당을 담당하는 동적 대역폭 알고리즘이 필요함을 알 수 있다.

2.1. 정적 대역폭 할당

정적 대역폭 할당은 smoothing 알고리즘에 의해 이루어진다. Smoothing 알고리즘[1]은 전송률의 변화를 smoothing하게 유지시킨다. 이를 통하여 저장된 비디오 데이터의 burstiness 특성을 감소시킬 수 있고, smoothing 된 비디오 데이터의 흐름은 전송 경로의 스위치에서 다른 데이터와의 혼잡(congestion) 발생 가능성을 줄일 수 있다. 전체 네트워크의 재교섭 시간이 감소하고 전송, 에러 발생률도 감소하게 된다. 이를 통해 재전송의 부하를 줄일 수 있다. 아래는 최적 smoothing 알고리즘[1]에 관한 간략한 설명이다. 이는 본 논문에서 채용하고 있는 upper limit smoothing 알고리즘 이해에 도움이 된다. 상세한 알고리즘은 참고문헌 [1][2]를 참조하시오.

서버에 N개의 프레임으로 구성된 하나의 비디오 데이터의 전체 시퀀스가 저장되어 있고, $x-byte$ 크기의 재생 버퍼를 가지고 있는 사용자에게 전송을 한다고 하자. 또한 사용자에 전송되는데 걸리는 지연은 없다고 가정한다. Low constraints는 사용자에서 재생에 필요한 데이터의 누적 량을 나타낸다. 각 프레임의 크기를 $l_1, l_2, \ldots l_n$로 표시하면, low constraint는 $L(k)=\sum_{i=1}^{k} l_i$ 가 된다. 실시간의 경우에는 초기 지연시간(T_{delay})이 존재하므로, 실시간에서의 low constraint는 $L(k-T_{delay})$가 된다. Upper constraint는 low constraint에 사용자 버퍼의 크기를 더한 것이다. Upper constraint는 사용자 버퍼에 최대로 쌓을 수 있는 누적 데이터의 양을 의미한다. Low constraint와 upper constraint를 지켜 줌으로써, 사용자 버퍼의 underflow나 overflow의 발생을 미연에 방지할 수 있다. 이 제한조건을 만족하는 대역폭의 할당을 feasible하다고 한다. 최적 smoothing 알고리즘은 feasible 상태에서, 가능한 오랫동안 Constant-Bit-Rate(CBR)를 유지하면서 대역폭을 할당하는 것이다. 이러한 조건을 만족하는 데이터 흐름은 네트워크를 통해서 전송되는 동안 다른 데이터 흐름들에게 영향을 적게 미치게 되고, 네트워크에서의 대역폭 할당을 쉽게 해준다. Feasible 상태를 유지하기 위해 전송률을 바꿔야 할 경우에는, 가능한 빨리 전송률에 변화를 줌으로서, 전송률의 변화를 작게 만드는 것을 목표로 하고 있다. 이는 전송률의 변경점(change point)에서 다음 변경점까지의 기울기를 구해 feasible 상태에서의 최대의 전송률과 최소의 전송률을 점점 줄여나감으로서 이루어진다. 다음 단위시간의 constraints point까지의 기울기가, 최대의 전송률보다 작고, 최소 전송률 보다 크면 다음 단위 시간으로 연장을 하게 된다. 그러나 최대와 최소 둘 중 하나를 만족하지 않으면, 그 전 시점의 constraints point가 변경점이 된다.

2.2 동적 대역폭 할당

정적으로 할당된 대역폭은 각각의 비디오 스트림 특성에 대해 독립적으로 계산된 것이므로 한정된 대역폭을 갖는 서버에서 다중 스트림을 전송하기 위한 대역폭을 효율적으로 할당 할 수 없다. 네트워크의 상태를 고려한 대역폭 할당을 위한 서버의 보조 공간을 가상 버퍼라고 한다. 서버에 가상 버퍼를 두고 각 스트림의 상태와 네트워크의 상태를 고려한 대역폭 할당을 동적 대역폭 할당이라고 한다.

일반적인 동적 대역폭 할당은 가상 버퍼에 쌓여 있는 데이터의 양에 비례하게 선형의 비율로 대역폭을 할당하는 것이다. 이 개념은 주변의 상황에 따라서 버퍼 레벨의 증감을 통해 시스템의 안정성을 유지하고, 서버에 할당된 총 대역폭을 각 비디오 스트림에 효율적으로 분배 할 수 있다. 이것은 매우 단순한 계산을 요구하므로 적용하기가 쉽고, 또한

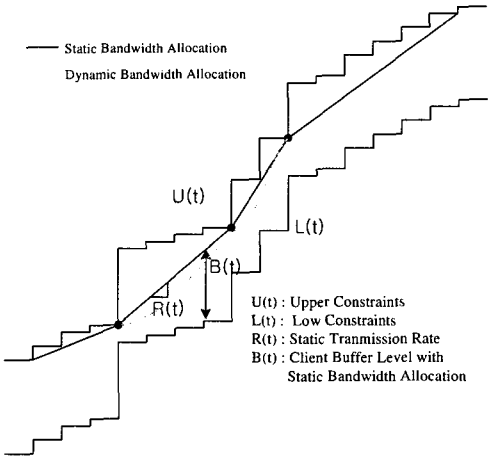


그림 2. 정적 대역폭 할당과 동적 대역폭 할당.

비교적 효율적인 개념이다. 그러나 가상 버퍼의 레벨에 의해 단순히 선형으로 할당해 주고 있기 때문에, 순간적인 네트워크 혼잡상황이 발생 할 때, 각 스트림의 버퍼 레벨을 같게 유지하지 못하고, 부등성을 유지하게 되므로 QoS의 저하를 야기 시킨다[3]. 이러한 문제점을 수정하기 위해서 고려된 것이 Min-max 알고리즘[3]이다. 이 알고리즘의 기본 개념은 혼잡상황이 발생할 경우, 각 스트림의 버퍼의 레벨을 같게 해주기 위해서, 가장 버퍼의 길이가 큰 것을 되도록 작게 해주도록 대역폭을 할당한다. 하지만 이 알고리즘은 계산 량이 많기 때문에 실시간을 요구하는 대역폭 할당에 적용하기가 어렵다.

Ⅲ. 제안하는 알고리즘

앞에서 서술한 최적 smoothing 기법[2]은 Peak cell rate를 줄이는 것에 주안점을 두고 있기 때문에, 사용자 버퍼의 효율은 떨어지게 된다. 다중 스트림의 경우, 각 스트림이 독립적으로 미리 계산된 대역폭만큼 할당받지 못 할 수 있고, 이런 경우 사용자에게 제공되는 비디오 서비스 품질이 떨어지게 된다. 특히 low constraint에서 변경점이 이루어지는 경우에는 할당받은 대역폭은 변경점 주변 시점에서는 반드시 실제로 할당받아야 한다. 이를 위해 한정된 서버의 총 대역폭 중 일부를 우선적으로 할당해야 되는 문제를 야기 시킨다.

본 논문에서 제안하는 알고리즘은 정적 대역폭 할당(Static bandwidth allocation)과 동적 대역폭 할당(Dynamic bandwidth allocation)의 두 단계로 이루어진다. 그림 2에서, 정적 대역폭 할당은 실선으로, 동적 대역폭 할당은 점선으로 표시되어 있다. 정적 대역폭 할당에서는 가능한 오랫동안의 CBR 특성을 유지하면서, 사용자 버퍼의 오버플로오가 발생하지 않는 범위 내에서 대역폭을 할당한다. 최대한 사용자 버퍼에 많은 양의 데이터를 저장할 수 있도록 할당하는 것이다. 이를 통해서 버퍼의 이용도를 높이고, 버퍼에 남게 되는 데이터의 량을 높일 수 있다. 두 번째 단계인 동적 할당에서는, 정적할당에서 지정된 대역폭과 사용자 버퍼에 남아있는 데이터 양 사이의 상관관계를 나타내는 real demand factor라는 새로운 변수를 이용, 사용자 버퍼가 언더플로오 되지 않는 범위 내에서, 서버에 할당된 대역폭을 효율적으로 이용할 수 있게 한다. 그림 2에서 보는 봐와 같이 실제적인 대역폭 할당인 동적 대역폭 할당은 각 데이터 스트림의 정적 대역폭 할당이 주어진 상황에서, 네트워크의 상황에 맞는 다중 데이터 스트림을 위한 대역폭 할당이므로, 정적 대역폭 할당을 따라 가려는 경향을 보인다. 아래에서는 본 논문에서 제안한 두 단계의 알고리즘에 관해서 설명하였다.

3.1 Upper Limit Smoothing 알고리즘

정적 대역폭 할당은 전송률의 변경이 upper constraints에서만 이루어지도록 하고, 전송률의 smoothness 조건을 만족하기 위해, 전송률 변경점까지 CBR 특성이 유지되도록 한다. 이를 본 논문에서는 upper limit smoothing 알고리즘이라고 하였다. Upper limit smoothing 알고리즘은 사용자 버퍼의 크기만을 고려하므로 실제로 데이터 스트림의 대역폭 할당 전에 미리 결정되어진다. 이러한 방식의 할당을 하게 되면, 변경점에서 사용자 버퍼가 가득 찰 확률이 높아진다.

아래는 제안한 upper limit smoothing 알고리즘이다. t_s는 가장 최신의 전송률 변경점이고, $C_{\max}$는 feasible 상태에서의 전송률을 나타낸다. 현재의 검사점(t_c)와 t_s를 연결한 직선의 기울기가 $C_{\max}$보다 크면, 현재의 전송률을 연장 할 수 있다. 그렇지 않은 경우는 변경점에서 검사점의 바로 전, 즉, 최대 feasible point(t_f)까지의 구간에서 $C_{\max}$의 전송률을 갖는 대역폭을 할당하게 된다.

- Upper limit smoothing Algorithm

  $t_c = 1;\ t_s = 0;\ t_f = t_c;$

  $$C_{\max} = \frac{U(t_c) - U(t_s)}{t_c - t_s}$$

Repeat

$t_c = t_c + 1;$

if $C_{max} \geq \frac{U(t_c) - U(t_s)}{t_c - t_s}$ *then*

$C_{max} = \frac{U(t_c) - U(t_s)}{t_c - t_s};$

$t_f = t_c$

else

Output(t_s, t_f, C_{max});

$t_s = t_f$;

$t_f = t_f + 1;$

$t_c = t_f$;

$C_{max} = \frac{U(t_f) - U(t_s)}{t_f - t_s}$

end

Until $t_c = N$

t_s : *the lastest change point,*

t_f : *the most feasible point,*

t_c : *the latest check point,*

C_{max} : *the feasible rate.*

Output(t_s, t_f, C_{max})는 t_s와 t_f 구간에서 정적 대역폭 할당 C_{max}를 출력하고, 이는 그 구간의 정적 대역폭 할당에 의한 전송률 $R(t)$가 된다. 또한 *Output*(t_s, t_f, C_{max})는 $B(t)$ 값을 계산한다. $B(t)$는 upper limit smoothing 알고리즘으로 대역폭을 할당하였을 때 사용자 버퍼에 남아있을 데이터의 양이고, 이 값은 동적 대역폭 할당에서 사용되어 진다. $B(t)$에 관한 자세한 사항은 다음절에서 다룬다.

구간 t_s와 t_f 사이의 R은, $R = \sum_{t_s}^{t_f - 1} l_i / (t_f - t_s) = C_{max}$, 그 구간에서의 비디오 데이터의 평균 프레임 크기로 해석 할 수 있다. 새로운 변경점 형성은 전 구간에서의 평균 프레임 크기가 변경점에서 소모되는 프레임 크기(변경점 시점에서 연속적인 재생을 위해서 필요한 데이터의 량)보다 작은 경우에 해당한다. 이는 앞으로의 전송률이 전 구간에서의 전송률보다 크다는 것을 의미하는 것은 아니다. 단지 현시점에서 필요한 프레임 크기가 과거의 평균 프레임 크기보다 크기 때문에 feasible 구간을 연장하게 되면 upper constraint를 가로지르게 되어서 위반을 하기 때문이다. 일반적으로 변경점 형성은 I-frame에서 많이 형성되게 된다.

제안한 upper limit smoothing 알고리즘의 최대 목적은 데이터를 전송할 때 burstiness를 줄이는 것이다. MPEG 데이터의 흐름은 MPEG 알고리즘 특성상 네트워크 혼잡의 원인이 되는 burstiness를 갖고 있기 때문에, 효율적인 대역폭의 할당이 필수적이다. 제안된 알고리즘은 사용자 버퍼에 누적된 데이터 크기가 높은 상태에서 CBR 특성이 나타난다. 사용자 버퍼의 레벨을 높이기 위해서는 변경점은 upper constraint 지점에서만 형성 될 수밖에 없다. 제안된 알고리즘은 최적 smoothing 알고리즘[2]과 유사하지만, low constraints에서 변경점의 형성이 이루어지지 않기 때문에 peak rate를 감소하는 점에서는 최적이지는 않다. 그러나 low constraints에 대한 정보를 $B(t)$를 통해서 동적 대역폭 할당에서 처리하게 해줌으로써 다중 비디오 스트림의 대역폭 할당에 적합하게 하였고, 대부분의 burstiness를 줄여주고 있다. 이는 모의실험 결과에서도 잘 나타나고 있다. 제안된 알고리즘은 버퍼의 크기와 관계없이 수행될 수 있다는 장점이 있다.

3.2 동적대역폭 할당과 Real Demand Factor

그림 3은 K개의 스트림을 동시에 지원하는 동적 대역폭 할당 시스템을 나타낸다. 본 논문에서는 단순하고 효율적인 proportional 선형 알고리즘을 사용한다. 선형 알고리즘은 입력에 비례해서 대역폭을 할당해 줌으로써, 버퍼의 안정을 유지한다. 그러나 선형 알고리즘은 단순히 대역폭 요구량의 비교만으로 대역폭을 할당하기 때문에, 네트워크 혼잡 상황에서 각 스트림의 버퍼의 크기를 불균등하게 하는 문제점이 있다[3]. 본 논문에서는 이러한 문제점을 보완하기 위해서, 대역폭에 대한 요구량과 다음 단위 시간에서의 사용자 버퍼의 데이터 량의 비를 의미하는 real demand factor ($P_k(t)$)라는 변수를 이

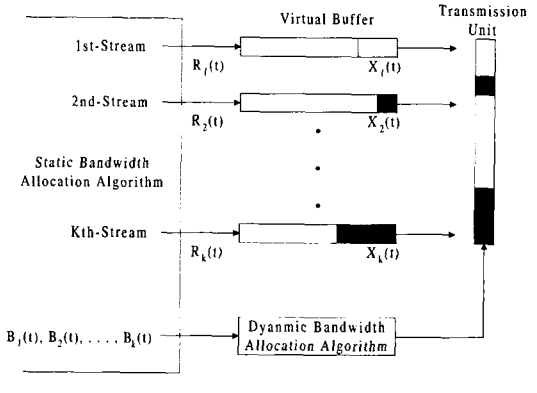


그림 3. 동적 대역폭 할당 시스템.

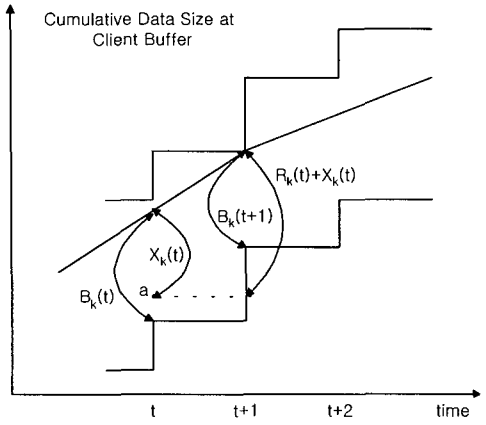


그림 4. Real Demand Factor.

용하여, 선형 알고리즘 입력에 가중치로 사용하였다.

그림 4에서 나타난 봐와 같이, k번째 스트림에서 정적 대역폭 할당에서 요구한 대역폭($R_k(t)$)과 동적 대역폭 할당($A_k(t)$)에 의해 실제 할당받은 대역폭의 차이에 의해서 서버의 가상 버퍼에 남게 되는 데이터의 량을 $X_k(t)$라 하면, 다음 단위 시간에 가상 버퍼에 남게 되는 데이터의 량($X_k(t+1)$)은 $X_k(t)+R_k(t)-A_k(t)$이 된다.

k번째 스트림이 다음 단위 시간에 정적 대역폭 할당과 일치하기 위한 필요한 대역폭은 $X_k(t)+R_k(t)$가 되고, $X_k(t)+R_k(t)$의 할당이 k번째 스트림에 주어질 경우, 다음 단위 시간에 사용자 버퍼에 남아 있을 수 있는 최대 데이터의 크기는 $B_k(t+1)$이 된다. $B_k(t)$는 정적 대역폭 할당이 이루어 질 때 low constraint에 따라 소모되고 남게 되는 사용자 버퍼의 데이터 양이다. 본 논문에서는 이 둘의 비율을 스트림의 real demand factor로 정의하였다. 그림 4에서 포인트 a는 동적 대역폭 할당에 의해 전송되어 사용자 버퍼에 쌓인 전송된 스트림의 누적 값을 나타낸다. 많은 경우 정적 대역폭 할당이 실제 대역폭 할당보다 크기 때문에, 포인트 a의 위치는 정적 대역폭 할당의 최고점과 low constraint 사이에 위치하게 된다.

정의) Real Demand Factor:

$$P_k(t)=\frac{X_k(t)+R_k(t)}{B_k(t+1)}.$$

위에서 정의한 real demand factor, $P_k(t)$는, 현재의 상태에서 다음 단위 시간 동안 사용자에게 전송을 하지 않게 되었을 때의 사용자 버퍼 상태를 의미한다. real demand factor가 0 과 1 사이의 값을 갖게 때 사용자 버퍼는 안정 상태에 있고, 1 이상의 값을 갖게 될 때 사용자 버퍼는 언더플로오가 발생하게 된다. 동적 대역폭 할당은 real demand factor의 가중치를 두고 서버의 할당된 총 대역폭을 선형적으로 할당해 준다. 이러한 방식의 동적 대역폭 할당은 정적 대역폭 할당에서 정해진 대역폭을 따라 가려고 노력한다.

- 동적 대역폭 할당

$$A_k(t)=C\frac{(X_k(t)+R_k(t))*P_k(t)}{\sum_{k=1}^{K}((X_k(t)+R_k(t))*P_k(t))}$$

$$X_k(t+1)=X_k(t)+R_k(t)-A_k(t)$$

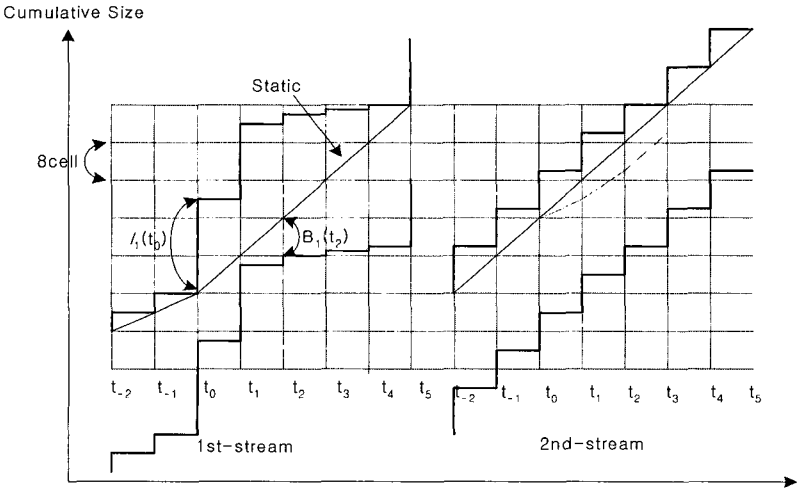


그림 5. Real demand factor를 이용한 2 스트림의 대역폭 할당.

표 1. 데이터 스트림의 프레임 크기

	t_{-2}	t_{-1}	t_0	t_1	t_2	t_3	t_4	t_5
l_1	4	4	20	16	2	1	1	13
l_2	10	8	8	8	6	8	8	15
B_1	26	26	10	2	8	15	22	
B_2	20	20	20	20	22	20	20	

위 식은 real demand factor를 사용한 동적 대역폭 할당 알고리즘을 보여준다. $A_k(t)$는 k번째 비디오 스트림의 비례 선형 알고리즘에 의해서 할당받은 대역폭이고, C는 서버가 지닌 최대 전송 대역폭을 표현한다. 선형 알고리즘의 입력 변수로 real demand factor를 사용하기 때문에, 동적 대역폭 할당 알고리즘은 정적 대역폭 할당을 유지하기 위한 대역폭이 변화와 다음 단위 시간에 사용자 버퍼의 량의 변화에 따라서 총 대역폭을 할당하게 된다. 정적 대역폭 할당을 유지하기 위한 대역폭이 많아지는 경우, 즉, 서버의 가상 버퍼에 데이터가 많이 있을 경우 정적 대역폭 할당을 유지하기 위해서 많은 대역폭이 필요하다. 정적 대역폭 할당에서 할당받은 대역폭이 일정하더라도, 다음 단위 시간에 사용자 버퍼의 레벨이 감소되는 것은 bursty 프레임을 의미하고, 이 경우에도 많은 대역폭이 할당되게 된다.

앞에서 언급했듯이 정적 대역폭 할당은 각 비디오의 프레임 크기에만 의존해서 계산이 된다. 사용자 버퍼의 레벨을 높이고 burstiness를 줄이는 것에만 초점을 두고 있기 때문에 다중 비디오 스트림을 지원하기에는 적합하지가 않다. 각 스트림 사이에 정적 대역폭 할당을 유지하기 위해 필요한 대역폭의 합은 서버의 전송대역폭 보다 클 수가 있다. 이런 상황은 네트워크 혼잡 상황을 유발하고, 각 사용자의 QoS를 보장하기 위해서 각 스트림 사이에 대역폭 조절이 필요하게 된다. 이러한 부분을 조정하는 것이 동적 대역폭 할당의 역할이다. 아래에서는 본 논문에서 제안한 대역폭 할당의 예제를 제시하였다.

- 예제: 2개의 비디오 스트림의 대역폭 할당

그림 5에서 보는 봐와 같이, 서버는 두 종류의 비디오 스트림을 사용자에게 제공하고 있다고 가정하였다. 서버의 단위 시간당 전송할 수 있는 최대 cell의 수는 12 cell(서버에 할당된 최대 대역폭), 각 사용자의 최대 버퍼링 능력은 30 cell이라고 가정한다. 표에서 l_1, l_2는 각 스트림의 해당 시간에서의 프레임 크기를 나타낸다.

첫째 스트림의 t_{-2}에서 t_0 전까지의 정적 대역폭 할당이 4 cell 이라고 하면, t_0에서 변경점을 형성하게 된다. 이것을 upper limit smoothing 알고리즘을 이용해서 정적 대역폭 할당을 구해보면 $t_s = t_0$, $C_{\max} = 20$ cell, $t_f = t_1$이 된다. 검사점을 t_2로 하면 조건문에서 $20 \geq 36/2$ 가 되므로 $C_{\max} = 18$ cell, $t_f = t_2$가 된다. 계속해서 검사점을 증가시키면 $t_f = t_5$ 일 때 $C_{\max} = 8$ cell 이 된다. 여기서 검사점을 하나 더 증가시키면, $t_c = t_6$가 되고 조건문에서 $C_{\max} = 8 < (20 + 16 + 2 + 1 + 1 + 13)/6$ 이 되므로 시작점에서부터 현재까지의 가장 feasible한 점인 t_5까지의 구간에 정적 대역폭 할당으로 8 cell을 할당하게 된다. 그리고 t_5는 새로운 시작점이 된다.

t_0 전까지는 두 스트림의 정적 대역폭 할당에 의한 대역폭의 합이 12 cell 보다 작아서 정적 대역폭 할당에 따라 대역폭이 할당되었다고 가정하면 두 비디오 스트림의 정적 대역폭 할당 대역폭과 동적 대역폭 할당의 차이, $X_1(t_0) = X_2(t_0) = 0$이 된다. t_0 시점에서 첫째 스트림이 변경점을 형성하고 두 스트림의 정적 대역폭 할당이 8 cell로 같게 되면서 두 스트림의 대역폭의 합이 12셀 보다 크기 때문에 네트워크의 상태가 혼잡 상황으로 변화하게 된다. 이 시점에서 real demand factor를 이용하여 동적 대역폭을 할당해 보자. 위의 real demand factor 식에 의하여,

$$P_1(t_0) = \frac{(X_1(t_0) + R_1(t_0))}{B_1(t_1)} = \frac{0+8}{2} = 4,$$

$$P_2(t_0) = \frac{(X_2(t_0) + R_2(t_0))}{B_2(t_1)} = \frac{0+8}{20} = \frac{2}{5}$$

가 되고 $A_1(t_0) = 10.91$, $A_2(t_0) = 1.09$, $X_1(t_1) = -2.91$, $X_2(t_1) = 6.91$가 되게 된다.

여기서 보듯이 두 스트림은 같은 대역폭을 요구하지만, $B(t_0)$의 차이에 의해서, 두 스트림에 할당받게 되는 대역폭이 다르게 된다. 위의 예의 경우, 첫 번째 스트림의 대역폭의 요구가 두 번째 스트림의 대역폭 요구보다 더 크다는 것을 나타내고 있다. 알고리즘을 계속 적용하게 되면, $P_1(t_1) = 5.09/8 = 0.64$, $P_2(t_1) = 14.91/22 = 0.68$, $A_2(t_1) = 9.08$, $X_1(t_1) = 2.17$ $A_1(t_1) = 2.92$, $X_2(t_2) = 5.83$가 되게 된다. 이렇게 정적 대역폭 할당을 미처 따라가지 못하고 서버의 가상 버퍼에 쌓이게 되는 데이터들은 네트워크의 혼잡 상황이 해소되면서 사용자에 전송

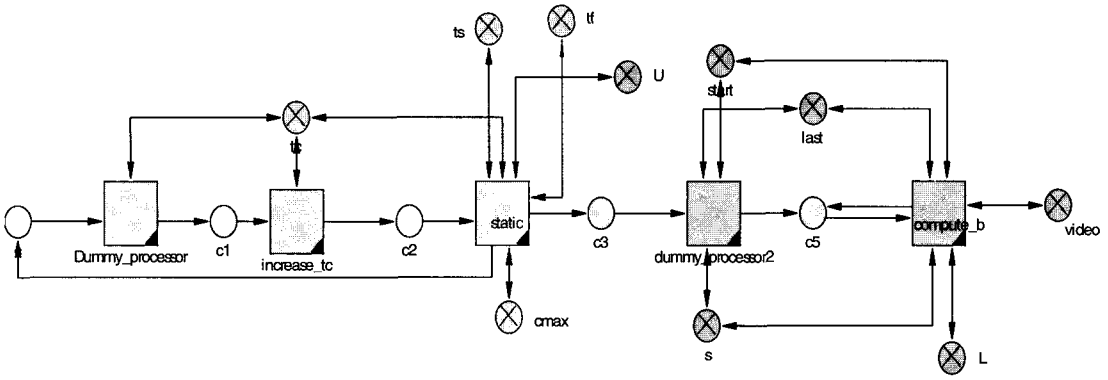


그림 6. 정적 대역폭 할당 모듈.

할 수 있게 된다. 즉 real demand factor는 $B(t)$에 의해서 사용자에 의해서 소모되는 특성을 반영하게 되고, $X(t)$와 $R(t)$에 의해서 정적 대역폭 할당을 유지하기 위해서 필요한 대역폭을 서버에 요구하게 된다.

Ⅳ. 모델링 및 모의실험 분석

제안한 알고리즘은 페트리 네트에 기반을 둔 모의 실험 도구인 ExSpect 6.41을 이용하여 모델링 하였고, MPEG_1 Trace 데이터를 이용하여 성능을 분석하였다. ExSpect는 Channel(작은 원), Processor(네모 상자), Store(작은 원에 X마크), Arc (화살표)로 구성된다. Channel은 페트리 네트의 place의 기능이고 토큰(정보와 제어권)을 Processor로 넘겨준다. Processor는 transition과 같은 기능이고 토큰으로 받은 정보를 계산 할 수 있다. Store는 항상 하나의 토큰이 있는 channel이 되고 정보를 저장하는 곳이 된다. Arc는 토큰이 흐르는 방향을 나타내게 된다.

그림 6은 정적 대역폭 할당을 계산하는 모듈이다. 좌측 세 개의 processor (Dummy_processor, increase_tc, static)가 upper limit smoothing 알고리즘의 Output(t_s, t_f, C_{max})를 계산해서 c3에 정보를 넘겨주게 된다. 우측 두 개의 processor (dummy_processor2, compute_b)는 upper limit smoothing 알고리즘에 의해서 요구한 대역폭을 할당할 때, 사용자 버퍼에 남아있을 데이터의 양인 $B(t)$를 계산해서 video store에 저장한다. 각 trace에서 얻은 정보를 이용하여 U와 L store에 constraint에 대한 정보를 입력해 주게 된다.

그림 7은 동적 대역폭 할당을 계산하는 모듈이다. 4개의 비디오가 정적 대역폭 할당과 $B(t)$가 계산되

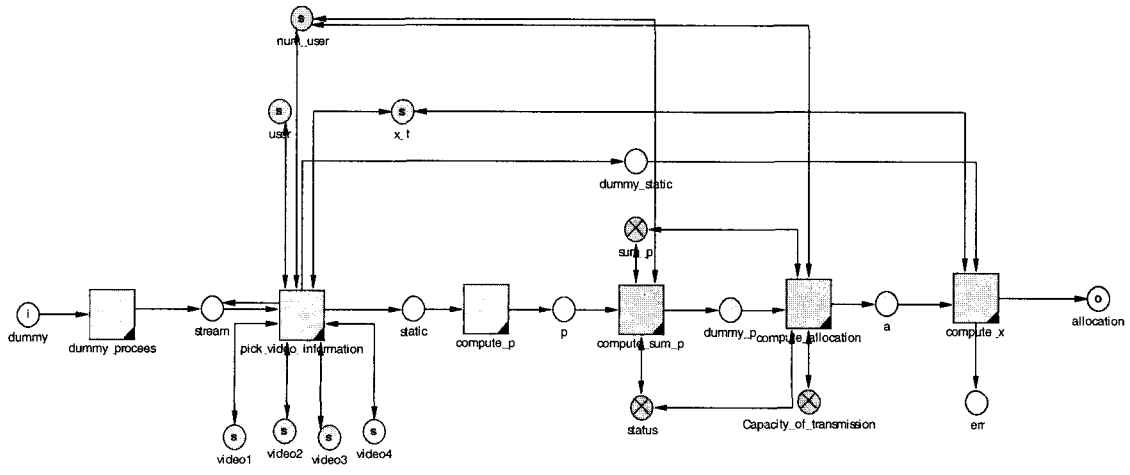


그림 7. 동적 대역폭 할당 모듈.

어 video1~4에 저장되어 있다. Pick video information processor는 사용자가 원하는 데이터 스트림의 프레임 정보를 알려준다. real demand factor($P_k(t)$)를 계산하여, 동적 대역폭 할당을 계산하게 된다.

- U: upper constraints
- L: low constraints
- t_s: change point
- t_f: feasible point from change point
- R: transmission rate from t_s to t_f = C_{max}
- Static: compute algorithm with static constraints and Output(t_s, t_f, C_{max})
- Compute_b : $B(t_s+k)=kR+b-\sum_{t_s}^{t_s+k} l_i$
- X_t: 가상 버퍼에 있는 데이터의 량
- User: 사용자가 원하는 비디오와 현재 재생중인 프레임 시간.
- Video1~4: 각 비디오에 대한 프레임 시간에 따른 정적 대역폭 할당 정보와 $B(t)$
- Pick_video_information: 스트림의 playback: 프레임 시간에 대한 정적 대역폭 할당 정보와 $B(t)$를 얻는다.
- Compute_p, Compute_sum_p: real demand factor 계산
- Compute_allocation : 스트림의 동적 대역 폭 할당 계산
- Compute_x: 스트림의 가상 버퍼의 크기 계산

모의실험에서 사용한 trace는 University of Wurzburg Institute of Computer Science에서 MPEG1 방식으로 코딩된 비디오를 분석한 결과를 사용하였다[4]. 표 2는 모의실험에서 사용한 trace의 통계적인 특성과 정적대역폭 할당을 적용한 결과를 보여준다. Dino, Bond, Lambs 는 영화이고 Mtv는 음악채널 방송을 MPEG으로 코딩한 것이다. 사용되어진 trace는 초당 24 프레임으로 구성되어 있다.

표 2에 나와 있는 바와 같이, 본 논문에서 제안한 upper limit smoothing 알고리즘은 원래 데이터의 peak rate를 70~50% 감소시킨다. 그러나 300K bits의 버퍼를 가진 최적 smoothing 알고리즘에 비해 20%~40% 감소 효과가 떨어짐을 알 수 있다. 이것은 upper limit smoothing은 사용자 버퍼의 크기를 고려하지 않고, 버퍼의 데이터 레벨을 높이는 것을 목적으로 하고 있기 때문이다. 그러나 전송률 변경점이 low constraint와 접하지 않기 때문에, 동적 대역폭 할당 단계에서 peak rate를 감소시켜 줄 수 있었다.

그림 8은 Lambs의 원래 데이터의 특성을 나타내고, 그림 9는 upper limit smoothing 알고리즘에 의한 대역폭 할당을 보여주고 있다. 가로축은 단위 시간을 나타내고, 세로축은 할당받아야 하는 대역폭을 나타낸다. 그림 8에서 길게 나온 부분들은 대부분 I-frame에 의한 것이고, 밑에 검게 나타나는 부분들은 B-frame, I-frame에 의한 것이다.

Upper limit smoothing 알고리즘에 의해 대부분의 burstiness가 없어진 모양을 그림 9에서 보여주고 있다. Upper limit smoothing 알고리즘은 최적이지는 않지만, 그림 8에 비해서 smoothing 됨으로써 충분히 burstiness를 줄여주고 있고, 이는 전체적인 대역폭 할당을 용이하게 한다.

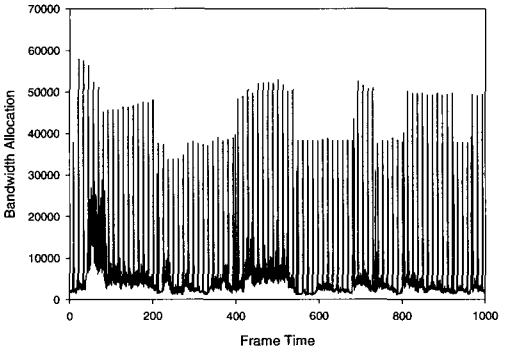


그림 8. Lambs의 데이터 특성.

표 2. Trace 데이터의 통계적 특성 및 smoothing 알고리즘을 적용 할 경우의 peak rate.

	Dino	Bond	Mtv	Lambs
	Original Statistical Property			
Mean(Kbits/frame)	14.8	14.6	19.4	8.6
Peak Rate(Kbits/frame)	107.3	106.7	89.4	88.0
	Upper Limit Smoothing Algorithm			
Peak Rate(Kbits/frame)	41.9	45.3	46.6	26.7
	Optimal Smoothing Algorithm (300Kbits)			
Peak Rate(Kbits/frame)	25.5	35.5	37.4	21.2

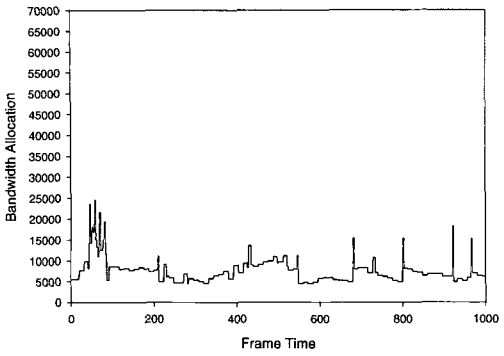


그림 9. Upper Limit Smoothing 알고리즘에 의한 정적 대역폭 할당 (Lambs).

그림 10과 11은 4개의 비디오 스트림을 지원하는 서버의 전체의 대역폭의 변화량을 나타내고 있다. 그림 10은 최적 smoothing (buffer size= 300K) 알고리즘과 선형 알고리즘을 적용한 경우의 필요한 대역폭을 보여준다. 각 스트림의 peak cell rate는 줄었지만, 전체 대역폭은 80Kbits/frame (1.92Mbps)를 초과하고, peak cell rate는 98Kbits/frame이다. 높은 peak cell rate는 비디오 스트림 QoS를 떨어뜨리는 원인이 된다.

그림 11은 본 논문에서 제안한 알고리즘을 적용한 결과를 보여준다. Peak cell rate가 80K bits/frame (1.92Mbps)로 감소함을 알 수 있다. 그러나, 최적 smoothing 알고리즘에 비해, 결과 값을 얻는데 걸리는 시간은 약간 증가한다.

K개의 비디오 스트림에 대한 네트워크 다중송신 이득은 $G=(1-p_i/\sum_{i=1}^{K}P_i)*100$로 정의된다[5]. p_i는 비디오 스트림을 다중송신을 통해 주어진 QoS를 만족하기 위한 필요한 대역폭을 나타내고, P_i

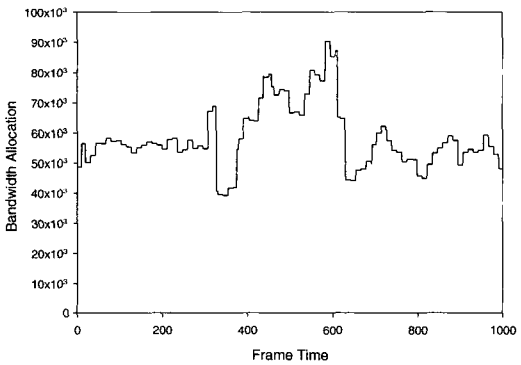


그림 10. 최적 smoothing 알고리즘에 의한 4개의 비디오 데이터의 대역폭 할당.

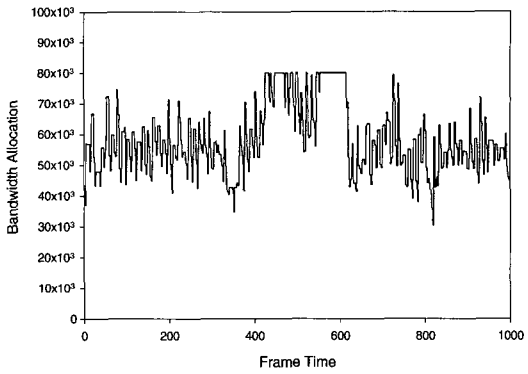


그림 11. 제안한 알고리즘에 의한 4개의 비디오 데이터의 대역폭 할당.

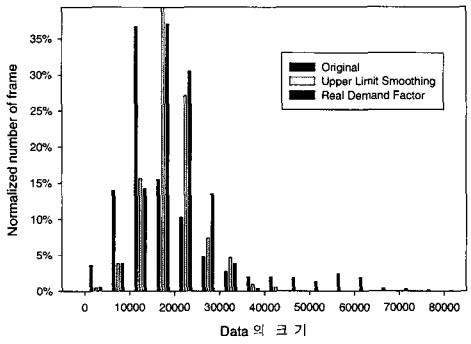


그림 12. 제안한 알고리즘에 의한 단계별 Mtv 프레임 데이터의 크기.

$(i=1,2,..,K)$는 각 스트림의 peak rate이다. 위의 두 경우의 다중송신 이득은 $G_{optimal}=67.4$ $G_{algorithm}=76.9$이다. 제안한 알고리즘은 다중송신 이득이 향상되었음을 보여준다.

그림 12는 Mtv의 프레임 데이터의 크기와 알고리즘을 적용했을 때의 대역폭을 나타내고 있다. 그림에서 보듯이, 원래 스트림 분포는 넓게 퍼져있다. 그래프의 꼬리 부분은 burstiness가 존재함을 보여주고, 이는 대역폭 할당의 문제를 야기 시킨다. Upper limit smoothing 알고리즘에 의해서 Mtv는 10000~25000 byte/frame, Lambs는 0~20000 byte/frame에 분포가 집중되고 있다. 이는 burstiness가 줄어들고 CBR의 특성을 나타내게 되었음을 의미한다. 동적 대역폭 할당(그림에서는 real demand factor라고 표현되어 있음)에 의한 실제적인 대역폭 할당의 분포는 정적 대역폭 할당과 유사하지만, 조금 더 넓게 분포하고 고대역폭 부분이 감소 한 것이 특징이다. 이는 다중 스트림의 대역폭 할당 과정에서 원하는 대역폭만큼 할당받지 못했기 때문이다.

표 3. 데이터 Loss 율.

	Optimal Smoothing and Linear Algorithm		Upper Limit Smoothing and Proportional Linear Algorithm	
	1.68Mbps (70kbits/frame)	1.92Mbps (80kbits/frame)	1.68Mbps (70kbits/frame)	1.92Mbps (80kbits/frame)
200kbits	3.6%	1.4%	2.8%	0%
300kbits	3.5%	1.3%	0.78%	0%
500kbits	1.3%	0%	0.16%	0%
700kbits	0.13%	0%	0%	0%
1000kbits	0%	0%	0%	0%

표 3에서 사용자 버퍼의 크기와 대역폭의 변화에 따른, 최적 smoothing 알고리즘에 선형 알고리즘을 사용한 경우와 본 논문에서 제안한 알고리즘의 데이터 loss rate를 비교해 보았다. 좌측은 사용자 버퍼의 크기를 나타내고, 서버에 할당된 총 대역폭이 1.68M와 1.92M인 경우를 비교하였다. 표에서 보듯이 사용자 버퍼의 크기가 작을수록 본 논문에서 제안한 알고리즘 성능이 최적 smoothing 알고리즘에 비해 좋아짐을 알 수 있다.

V. 결 론

멀티미디어 서비스를 사용자에게 원활히 제공하기 위해서는 효율적인 대역폭 할당 스케줄링이 필수적이다. 비디오 데이터와 같은 burstiness 특성을 지닌 다중 스트림을 전송하기 위해서는 smoothing 기법을 사용하게 되는데, 현재까지 제안되어진 최적 smoothing 알고리즘은 peak rate를 크게 줄일 수 있지만, 각각의 독립적인 스트림에 계산되었기 때문에, 서버에서의 네트워크 대역폭할당이나 프로세서 등을 공유하는 다중 스트림의 대역폭 할당에 적용하기에 적합하지 않다. 본 논문에서는 이러한 문제점 극복을 위해, 정적 대역폭 할당에서는 Upper limit smoothing 알고리즘을, 동적 대역폭 할당에서는 real demand factor를 이용한 proportional 선형 알고리즘을 제안하였다. Upper limit smoothing 알고리즘은 최적 smoothing 알고리즘과 유사하지만, low constraints에 대해서 적용하지 않고 있기 때문에 peak cell rate를 감소하는 관점에서는 최적은 아니다. 그러나 low constraints에 대한 정보를 $B(t)$를 통해서 동적 대역폭 할당에서 처리하게 해줌으로써 다중 스트림의 대역폭 할당을 효율적으로 수행하게 하였다.

본 논문에서는, 요구되는 대역폭과 다음 단위 시간에서의 사용자 버퍼의 양의 비를 의미하는 real demand factor($P_k(t)$)를 제안하였고, 이를 통하여 동적 대역폭 할당에서 비디오 스트림들 간의 대역폭 할당의 불균형을 극복 할 수 있도록 하였다. Real demand factor는 각 스트림이 사용자 버퍼를 능동적으로 사용하게 만들어 준다.

제안한 알고리즘은 페트리 네트에 기반을 둔 모의실험 도구인 ExSpect 6.41을 이용하여 모델링 하였고, 4 종류의 MPEG_1 Trace 데이터를 이용하여 성능을 분석하였다. 모의실험 결과에 따르면, 제안한 알고리즘은 비디오 스트림의 peak rate를 50%~70% 감소시킴을 알 수 있고, 다중 송신 이득의 향상을 얻을 수 있었다.

참 고 문 헌

[1] Jennifer Rexford and Don Towsley, "Smoothing Variable-Bit-Rate Video in an Internetwork," *IEEE/ACM Trans. On Networking*, Vol. 7, No. 2, April 1999.

[2] James D. Salehi and Zhi-Li Zhang, "Supporting Stored Video:Reducing Rate Variability and End-to-End Resource Requirements Through Optimal Smoothing," *IEEE/ACM Trans. On Networking*, Vol. 6, No. 4 August 1998.

[3] Subir K. Biswas and Rauf Izmailov, "Design of Fair Bandwidth Allocation Policy for VBR Traffic in ATM Networks," *IEEE/ACM Trans. On Networking*, Vol. 8, No. 2, April 2000.

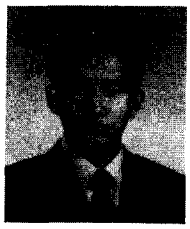
[4] O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems," University of Wurzburg Institute of Computer Science research report No. 101.

[5] L.Zhang and H.Fu, "A novel scheme of transporting pre-stored MPEG video to support video-on-demand (VoD) services," ELSEVIER Computer communications 23, pp. 133-148, 2000.

[6] Marco Furini and D. F. Towsley, "Real-time transmissions over Internet," *IEEE Transactions on Multimedia*, Vol. 3, No. 1, March 2001.

[7] Derek Eager, Mary Vernon, and John Zahorijan, "Minimizing bandwidth requirements for on-demand data delivery," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 13, No 5, September/October 2001.

김 정 택(Jung-Taek Kim) 정회원



2000년 2월 : 홍익대학교 전자전기공학부 졸업,
2002년 2월 : 홍익대학교 전자공학과 석사.
<주관심 분야> 멀티미디어 전송, 네트워크 프로토콜

고 인 선(Inseon Koh) 정회원

1979년 2월 : 서울대학교 전자공학과 졸업,
1989년 5월 : Marquette University 석사,
1991년 12월 : Rensselaer Polytechnic Institute 박사,
1992년 3월~현재 : 홍익대학교 전자공학과 교수
<주관심 분야> 멀티미디어 전송 및 동기, Network Analysis, 페트리 네트, Discrete Event System, 지능제어 시스템.