Journal of Korea Multimedia Society Vol. 8, No. 10, October 2005(pp. 1383-1390)

모바일 변환 저장 서버에서 가중치 기반의 DVDS를 위한 방법 연구

조종근†, 임영환††

요 약

모바일 멀티미디어 저장 서버는 동시에 다수의 개인용 이동 단말기(PDA)에서 요청하는 데이터를 실시간으로 저장 및 전송을 해야 한다. 그러나, 대역폭이 작은 모바일 인터넷 환경에서 다수의 개인용 이동 단말기가 동시에 서버로 접속하게 되면 병목(BottleNeck) 현상이 발생할 수 있다. 본 논문에서는 카메라가 장착된 개인용 이동 단말기에서 실시간으로 캡쳐(capture)한 영상을 모바일 변환 저장서버에서 저장 및 전송을 할 때 모바일 멀티미디어 저장 서버에서 발생할 수 있는 병목 현상을 효율적으로 해결할 수 있는 새로운 방법으로 가중치 기반의 동적 가변 디스크 스케줄링을 제안하고, 기존 방법들과 비교를 통해서 제안한 방법의 우수한 성능을 증명한다.

A Method for DVDS Based on Weight Value in the Mobile Translation Storage Server

Jong-Keun Cho†, YoungHwan Lim††

ABSTRACT

Mobile Multimedia Storage Server should provide to store and transfer a data that the number of concurrent datum by Personal Digital Assistant(PDA) requests in real time. simultaneously. But, While the large number of a PDA request connect to MTS(Mobile Translation Storage) server with BottleNeck problem in mobile internet environment by many constraints such as limited bandwidth can be introduced. This paper introduces an optimal method of new DVDS(Dynamic Variable Disk Scheduling) for MTS server which is based on the Weight Value to solve the BottleNeck problems that occur when transfer and store capture images through mounted digital camera on the PDA in real time simultaneously. The comparison with previous research shows that the proposed scheme provides better.

Key words: Multimedia Storage Server(멀티미디어 저장 서버), Disk Scheduling(디스크 스케줄링)

1. 서 론

기본적으로 모바일 멀티미디어 저장 서버는 실시간으로 이동 단말기와 같은 클라이언트들의 요청이 있을 경우 데이터의 저장뿐만 아니라 전송도 가능해야 하며, PDA에 의해 요청된 멀티미디어 데이터는 종료시한의 만족과 동시에 보장 서비스 성능을 제시하여 가능한 많은 사용자를 수용해야 한다.[1]

무선 인터넷을 이용해서 멀티미디어 데이터를 사용하는 사용자가 증가함에 따라서 느린 네트워크 속도와 협소한 대역폭을 가지고 있는 무선 인터넷환경에서도 카메라가 장착된 핸드폰이나 PDA와 같은 클라이언트에서 시간과 장소에 구애받지 않고 원하는 영상들을 실시간으로 캡쳐해서 동영상으로 저장을

※ 교신저자(Corresponding Author) : 조종근, 주소 : 서울시 강남구 대치동 1005(135-280), 전화 : 02)501-2640, FAX : 02)501-2641, E-mail : jkdang@empal.com
접수일 : 2004년 8월 21일, 완료일 : 2004년 9월 29일
† 정회원, (주)GOMID/수석연구원
†† 종신회원, 숭실대학교 미디어학부 부교수
(E-mail : yhlim@computing.soongsil.ac.kr)

한 후에 저장한 동영상을 멀티미디어 저장 서버에서 저장 및 전송하기 위한 논문이다.

본 논문에서는 가중치를 기반으로 슬롯의 크기를 결정해서 클라이언트들의 영상 데이터들을 처리하는 알고리즘을 제안하고, 서버에서 발생하는 병목현상을 해결하기 위해 양쪽 클라이언트들 데이터양의 증감폭에 따라서 버퍼를 자동으로 조절함으로써 라운드당 처리할 수 있는 슬롯의 크기를 가변적으로 공유해서 사용하는 방법과 같은 효과를 낼 수 있는 동적 가변 디스크 스케줄링 알고리즘을 제안한다.

2. 관련 연구

기존에는 VOD(video on demand)와 같은 상업적인 서비스를 위한 모바일 멀티미디어 저장 서버에 대한 연구가 많이 이루어졌다. 이와 같은 서버는 클라이언트에서 서버로 데이터를 요구하면 서버에 저장되어 있는 데이터를 일방적으로 다수의 클라이언트로 전송해 주는 시스템의 구조였다. 하지만, 요즘과 같은 무선 인터넷 환경에서는 카메라가 장착된 PDA나 핸드폰 같은 클라이언트에서도 자체적으로 만든 영상 데이터들을 서버에서 실시간으로 전송 받아서 저장하고, 반대로 서버에 저장된 데이터들을 언제든지 PDA나 핸드폰으로 전송을 할 수 있어야만 진정한 멀티미디어 저장 서버로써의 역할을 하게 되는 것이다.

이에 본 논문에서 제안하는 가중치 기반의 DVDS 방법과 Philips S. Yu 등이 제안한 슬롯의 사이즈를 일정하게 분할해서 사용하는 디스크 스케줄링인 GSS 방법[2] 그리고 ETRI에서 제안한 슬롯의 사이즈를 반으로 줄여가는 가변 디스크 스케줄링인 HDS 방법[3]을 비교 분석을 하여 본 논문에서 제안한 방법이 서버에서 발생할 수 있는 병목현상을 효율적으로 처리할 수 있다는 것을 실제로 시스템을 구현 한 후에 실험 데이터로 나온 그래프로 증명할 것이다.

2.1 Philip S. Yu 등이 제안한 GSS 방법

그림 1에서 보는 것처럼 GSS 방법은 서버에 요청을 하는 클라이언트들의 데이터들(Real Unit Sizes)을 컴퓨터가 처리할 수 있는 한 라운드에서 데이터를 균등분할 된 슬롯에 담아서 처리하는 방법이다.

즉, 시간당 컴퓨터가 처리할 수 있는 라운드의 처리능력을 공평하게 균등 분할된 슬롯을 만든 후에 분할된 슬롯에는 클라이언트들의 데이터들을 순서대로 담아서 처리를 한다. 이 방법에서의 장점은 모든 클라이언트들에게 동일한 슬롯의 크기가 설정되기 때문에 일정한 시간간격으로 처리가 된다는 것이다. 그러나, 라운드 시작 이후의 클라이언트 데이터는 다음 라운드에서 처리가 되기 때문에 최악의 경우에는 라운드 시간의 2배를 기다렸다가 처리가 되는 경우가 있다. 왜냐하면, 지금 들어온 클라이언트의 데이터는 현재의 라운드처리가 끝난 다음 번 슬롯에 할당되기 때문이다.

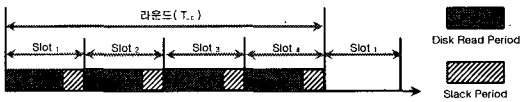


그림 1. GSS 슬롯 할당 방법

2.2 ETRI의 HDS 방법

그림 2에서의 HDS 방법은 GSS 방법의 단점을 보안하는 방법으로 새로운 클라이언트의 데이터는 크기에 상관없이 무조건적으로 다음 슬롯에 담아서 처리를 하게 함으로써 최악의 경우에는 새로운 클라이언트의 데이터는 라운드 2배의 시간을 기다렸다가 처리가 되는 것이 아니라, 슬롯 2배의 시간을 기다린 후에 처리가 되어 클라이언트 데이터들의 기다림을 많이 줄이게 된다. 그리고, 처리방법은 라운드의 시작 시간에 새로운 클라이언트의 데이터가 있는지를 검사해서 슬롯의 사이즈를 최대로 할당하고, 그 다음의 슬롯들은 이전 슬롯크기의 절반씩 줄여가며 슬롯을 할당해 가는 처리방법으로 GSS 방법보다는 좀 더 많은 클라이언트들의 요청들을 받아서 데이터들을 처리할 수 있다.

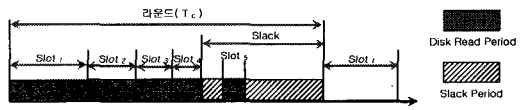


그림 2. HDS 슬롯 할당 방법

2.3 기존 방법들의 문제점 및 개선방안

GSS 방법과 HDS 방법은 모두다 멀티미디어 저장 서버에서 클라이언트의 요청에 의해서 데이터들을 전송하는 VOD와 같은 서비스에 사용하는 방법으로

이 방법들을 서버에 저장되어 있는 영상 데이터들을 클라이언트로 전송뿐만 아니라 동시에 클라이언트에서 만든 영상 데이터를 서버로 전송하는 방법에 적용해 보면 효율적이지 못 하다는 것을 알 수 있다.

다시 말해서, 한 방향 처리(서버에서 클라이언트로 영상 데이터 전송)에서는 효율적인 방법이 될 수 있을지는 모르겠지만, 양방향 처리(서버에서 클라이언트로 영상 데이터를 처리할 뿐만 아니라, 클라이언트에서도 영상 데이터를 서버로 동시에 전송하는 경우)를 동시에 할 경우에는 다수의 클라이언트들의 데이터들을 서버에서 저장 및 전송을 할 수 없다. 즉, 컴퓨터가 처리할 수 있는 능력만큼 처리를 하지 못한다. 특히, 대역폭과 처리능력이 떨어지는 PDA와 같은 클라이언트의 경우에는 더욱 더 떨어질 것이다.

왜냐하면, GSS 방법과 HDS 방법으로 양방향 처리를 할 경우에는 시간당 서버의 라운드 처리율을 고려해서 저장과 전송을 담당하는 부분의 버퍼를 각각 그림 1과 그림 2에서 제시한 방법으로 구성을 한 후에 처리해야 하기 때문에 효율적인 처리 방식이 되지 못 한다.

예를 들어서, 저장 또는 전송 부분의 클라이언트들의 요청들이 증가해서 어느 한쪽의 버퍼 사용량이 증가하게 되면 그 쪽의 버퍼에서만 병목현상이 일어나게 된다.

본 논문에서는 가중치를 기반으로 슬롯의 크기를 결정해서 클라이언트들의 영상 데이터들을 처리하는 알고리즘을 제안하고, 서버에서 발생하는 병목현상을 해결하기 위해서 양쪽의 클라이언트들의 데이터양의 증감폭에 따라서 버퍼를 자동으로 조절함으로써 라운드당 처리할 수 있는 슬롯의 크기를 가변적으로 공유해서 사용하는 방법과 같은 효과를 낼 수 있는 동적 가변 디스크 스케줄링 알고리즘을 제안한다.

3. MTS를 이용한 가중치 기반의 DVDS

MTS(mobile translation storage) 서버에서는 기본적으로 카메라가 장착된 PDA에서 실시간으로 캡쳐한 영상 데이터를 전송하면 MTS 서버에서는 영상을 받아서 자동으로 동영상 변환이 되어서 서버에 데이터들을 저장하고, 저장되어 있는 동영상을 다른 PDA로 전송하는 역할을 하고 있다.

3.1 가중치 기반의 DVDS 흐름도

그림 3은 가중치 기반의 DVDS 흐름도이다. 그림 3에서 보듯이 많은 수의 클라이언트들(PDA)이 동시에 서버로 캡쳐한 데이터들을 전송하게 되면, 클라이언트들의 데이터 영상들은 가중치 기반의 알고리즘에 적용을 받아서 이번 라운드에서 슬롯에 할당되어서 처리가 될 것인지 아니면 백업 버퍼로 넘어가서 다음의 라운드에서 처리가 될 것인지에 대한 결정을 한 후에 동적 가변 디스크 스케줄링에 의해서 서버로 저장이 되며, 그 반대로 저장된 영상 데이터 역시 가중치에 의해서 클라이언트의 처리 순서가 결정이 되면, 이 역시 동적 가변 디스크 스케줄링에 의해서 전송 및 저장이 이루어진다.

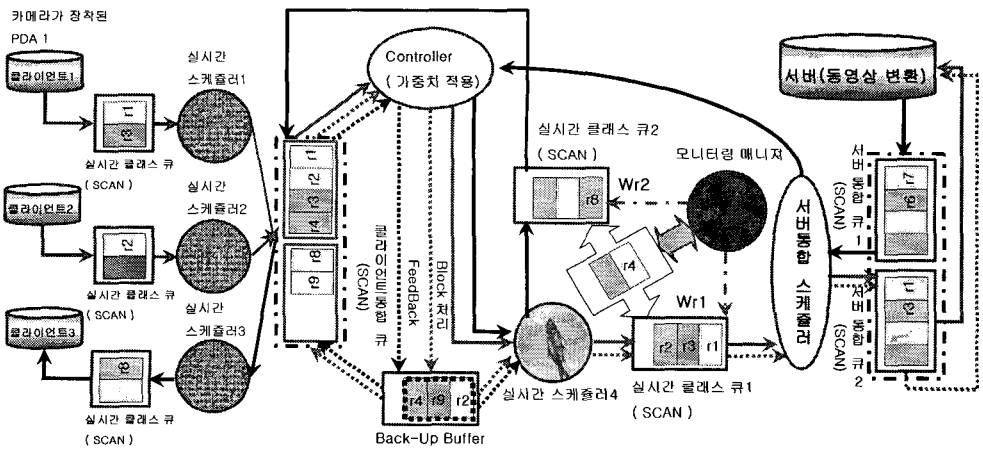


그림 3. 가중치 기반의 DVDS 흐름도

결국은 클라이언트인 PDA 요청의 종류에 따라서 서버에서는 저장 및 전송에 관련된 작업이 가중치 기반의 DVDS에 의해서 컴퓨터의 초당 라운드당 처리율을 최대한 활용하는 방법으로 처리가 되며, 제약조건은 아래와 같다.

1) 기본조건

a. MTS 서버는 다수의 클라이언트 요청에 대해 동시에 저장과 전송을 지원한다.

2) 전제조건

a. MTS 서버는 라운드 단위 방식으로 처리

b. MTS 서버는 멀티미디어 사용자의 서비스 요구에 대해서 가중치에 의한 알고리즘을 통해서 수용제어 과정을 수행한다.

→ 서비스 요청을 받으면 이미 서비스를 받고 있는 사용자의 QoS를 낮추지 않는 범위 내에서 새로운 요청을 받는다.

→ 새로운 사용자와 기존 수용된 사용자의 서비스 요청 시간이 총서비스 시간을 넘지 않는 범위 내에서 처리를 한다.

가중치 기반 DVDS 방법의 구현을 위한 변수로는 표 1과 같은 값들이 사용된다.

표 1. 논문에서 사용되는 변수

변 수	정 의
N	최대 사용자
Tround	한 라운드당 처리 시간 결정
SNi, TNj	저장,전송 라운드에 처리할 슬롯 수
Dreadsize	라운드당 처리할 총데이터 사이즈
RBtotal	라운드당 처리할 총 버퍼 사이즈
Rlength	라운드의 길이
i, j	저장과 전송을 위한 디스크 요청 i, j
si, tj	저장과 전송을 위한 디스크 요청 i, j를 위한 서비스 시간
Rsi, Rtj	라운드 Rlength 동안 디스크 요청 si, tj를 위한 서버스 시간

3.2 가중치 기반의 DVDS 방법

서버로부터 데이터를 전송받기 위해서 요청한 클라이언트들과 클라이언트들에서 만든 데이터를 서버로 저장하기 위해서 요청하는 클라이언트들의 데이터들은 현재 접속되어 있는 클라이언트들의 가중치 값들인 접속 순서, 전송 데이터의 크기 그리고 데이터의 전송 속도들에 의해서 처리순서를 결정하게 된다.

예를 들어서, PDA에서 만든 데이터를 서버로 저장을 할 경우에 대한 가중치 적용 방법은 아래의 (1), (2), (3), (4) 그리고 (5)에 의해서 적용될 수 있다.

이 연구의 실험을 위한 컴퓨터의 라운드당 처리용량을 계산한 결과가 3.9 M =< RBtotal/△Tround < 8.9 M 인 것을 알았다. 결과적으로 컴퓨터가 처리할 수 있는 라운드당 처리용량은 (1)처럼 나타낼 수 있으며,

$$\frac{RBtotal}{\Delta Tround} = (tmin + tseek + trot) * Dreadsize \quad (1)$$

이 된다. 즉,

$$\frac{RBtotal}{\Delta Tround} \leqq \sum_{i,j=1}^{N} Fsize(i,j) \quad (2)$$

(단, △Tround는 라운드당 처리시간을 의미)

(2)에 만족하는 Fsize(i,j)들을 아래의 (3), (4), (5)에서 계산되어진 가중치 값(가중치가 높은 클라이언트가 먼저 슬롯에 담겨서 처리)에 의해서 클라이언트들의 처리 순서가 결정되면 실시간 스케줄러에서 통합 스케줄러로 데이터를 보내면 서버에 저장이 된다.

$$\sum_{i,j=0}^{N} \frac{Fsize(i,j) * Nbandwidth(i,j)}{Cstart(i,j)} * 100\% = W(i,j) \quad (3)$$

$$W(v) = \sum_{i,j=1}^{N} \frac{W(v)}{W(i,j)} = \sum_{i,j=1}^{N} \frac{1}{W(i,j)} = 1 \quad (4)$$

(단, $\sum_{i,j=1}^{N} W(i,j) = 1, W(v) = 1$ 이며, W(i)와 W(j)는 저장과 전송을 위한 클라이언트들의 가중치 값)

$$v = \frac{W(v)}{W(i,j)} \quad (5)$$

(v는 접속 클라이언트들의 우선 순위 결정)

본 논문의 식들에서 사용한 상수 기호는 아래의 표 2와 같다.

가중치에 의한 수용 제어 알고리즘을 통과한 저장과 전송을 위한 데이터들의 라운드당 총 처리시간(tround)은 (6)과 (7)처럼 나타낼 수 있다.

표 2. 논문에서 사용되는 상수

상 수	정 의
Cstart	접속 순서
Nbandwidth	패킷 전송 속도
Fsize	데이터 전송 사이즈
D	디스크 수
tmin	디스크의 초기 대기 시간
trot	디스크의 최대 지연 시간
tseek	디스크의 최대 탐색 시간
tlatency	허용 가능한 초기 대기 시간
Tdiskrate	디스크의 평균 disk rate
Fp	PDA에서 캡쳐한 평균데이터 사이즈

$$Rsi = \lim_{N\to\infty} \frac{\sum_{i=1}^{N} si}{\sum_{i=1}^{N} ai} => \frac{\sum_{i=1}^{N} si/N}{\sum_{i=1}^{N} ai/N} \quad (6)$$

$$Rtj = \lim_{N\to\infty} \frac{\sum_{j=1}^{N} tj}{\sum_{j=1}^{N} aj} => \frac{\sum_{j=1}^{N} tj/N}{\sum_{j=1}^{N} aj/N} \quad (7)$$

(여기서, ai, aj는 저장과 전송을 위한 출발과 도착 시간)

그런데, 그림 3에서 보는 것처럼 양방향(서버에서 데이터를 저장과 전송)의 데이터양을 실시간으로 파악하는 모니터닝 매니저에 의해서 데이터 처리가 많지 않은 쪽의 버퍼를 공유해서 사용할 수 있도록 익스체인지 버퍼(Exchange Buffer:EB)를 만들어서 어느쪽이든 사용량이 많은 곳에서는 사용량이 적은 쪽의 메모리를 동적으로 사용을 할 수 있도록 하였다. 그러므로, 서버의 라운드당 처리시간은 (8)처럼 나타낼 수 있다.

$$\frac{RBtotal}{\Delta Tround} < \rho = \lambda_{\overline{Rsi}} + \lambda_{\overline{Rtj}} + \lambda_{\overline{EBij}}$$

$$=> \frac{\lambda_{\overline{Rsi}}}{N} + \frac{\lambda_{\overline{Rtj}}}{N} + \frac{\lambda_{\overline{EBij}}}{N} \quad (8)$$

(여기서, Lambda는 데이터의 도착 시간이다.)

또한 (6)과 (7)를 좀 더 자세하게 실질적으로 멀티미디어 저장 서버의 하드디스크에서 데이터를 처리한다고 가정을 하고 표현을 해 보면 (9)와 (10)처럼 나타낼 수 있다.

$$Rsi = N*tmin + \sum_{i=1}^{N} SNi*tseek + \sum_{i=1}^{N} N* \frac{\lceil \frac{si}{RBtotal} \rceil * RBtotal}{Tdiskrate} + trot \quad (9)$$

$$Rtj = N*tmin + \sum_{j=1}^{N} TNj*tseek + \sum_{j=1}^{N} N* \frac{\lceil \frac{tj}{RBtotal} \rceil * RBtotal}{Tdiskrate} + trot \quad (10)$$

위의 (9)와 (10)의 si와 tj는 (11)에 의해서 계산이 되며,

$$\sum_{i,j=1}^{N} \frac{Dreadsize(i,j)}{RBtotal} / Tround \quad (11)$$

Tround는 클라이언트 데이터들이 기다려야 하는 초기 대기시간보다 작아야 하므로, (12)를 이용해서 (13)을 유도해 낼 수 있다.

$$2\frac{Tround}{(SN+TN)} \leq tlatency \quad (12)$$

$$Tround \leq \frac{(SN+TN)*tlatency}{2} \quad (13)$$

(13)에서 SN과 TN의 값들은 각각 개별적으로 계산을 해서 값을 얻을 수 있다. 그러면, Tround 값을 알아낼 수 있다. 그리고, 이 논문에서는 카메라가 장착된 PDA에서 캡쳐된 데이터 즉, QCIF(144*176) 사이즈의 절반((144*176)/2) 데이터만 서버로 전송을 하면 서버에서는 그 데이터를 받아서 원래 데이터로 복원을 한 후에 동영상으로 자동 변환하는 방법을 이용했다. Dread size 값을 얻어내기 위한 (14)는 아래와 같이 나타낼 수 있다.

$$Dreadsize = \lceil \frac{(SN+TN)Fp*Tround}{d} \rceil * d \quad (14)$$

즉, (13)을 (14)에 대입을 하면 Dreadsize의 크기를 알아낼 수 있다. 또한 저장과 전송을 위한 최대 사용자는 (9)와 (10)에 의해서 아래 (15)와 (16)으로 계산할 수 있다.

$$N1 = \sum_{i=1}^{N} \frac{(Rsi - SNi*tseek)}{tmin(\frac{si*RBtotal}{RBtotal*Tdiskrate} + trot)} \quad (15)$$

$$N2 = \sum_{j=1}^{N} \frac{(Rtj - TNj*tseek)}{tmin(\frac{tj*RBtotal}{RBtotal*Tdiskrate} + trot)} \quad (16)$$

4. 실험방법 및 결과분석

4.1 실험방법

실험을 위해서 사용된 PC는 여건상 IBM Note Book R30 컴퓨터를 사용하였으며, 세부 Bus Architecture로는 Host bus 64bits/100 MHz, PCI 32 bits/33 MHz, ISA 16 bits/8.33 MHz, PCMCIA 16 bits/33MHz, Cardbus 32bits/32 MHz의 시스템을 사용하였다.

구현에 사용된 HDD는 Hitachi Travelstar 4K80 2.5-inch hard disk drive를 사용하였고, 서버의 OS는 Windows 2000을 사용하였으며, 메모리는 382 Ram을 장착하였다. 그리고, 현실적으로 10대 이상의 카메라가 장착된 PDA에서 들어오는 데이터를 받아서 처리를 할 수 없기 때문에 자체 클라이언트와 서버 프로그램을 작성을 한 후에 데이터를 주고받는 상황을 실시간으로 컴퓨터화면에 그래프로 디스플레이하게 한 후에 결과를 캡쳐했다.

실제 구현에 사용된 데이터의 사이즈는 1~800K 사이의 데이터들을 Random하게 라운드당 처리를 한다.

표 3. 논문의 구현을 위한 실제 상수값들

상수	정 의
Cstart	접속순서={1, 2, 3, 4.. }
Nbandwidt (MBit)	패킷전송속도{1=1024, 2=2048 3=3072, 4=4096}
Fsize (KByte)	데이터전송사이즈{1=(0<Fsize<40), 2=(40<Fsize<80),... 20=<760<Fsize<800)}
D	1
tmin	3ms
trot	14ms
tseek	13ms
tlatency	7.1s
Tdiskrate	43.4MB(43.4/7.1=1 초당 6.1MB)
Fp	36.09KB/s

위에 나오는 표 3은 본 연구에서 사용된 디스크의 특성을 포함한 상수들의 실제 값들이다. Hitachi Travelstar 4K80 2.5-inch hard disk drive의 1개 저장 용량은 30GB이며, 사용되는 데이터는 HDD의 여유 공간을 고려하면 26,214명의 데이터를 다룰 수 있다.

4.2 성능분석

본 논문에서는 제안한 가중치 기반 DVDS 방법의 성능을 기존 GSS/HDS 방법과 비교 분석하였다.

그림 4에서 구현한 프로그램의 결과화면은 MTS 서버의 1sec 라운드 시간에 클라이언트들의 프레임 데이터들을 7:3과 3:7 비율, 8:2와 2:8 비율 그리고, 5:5 비율의 저장/전송 프레임 데이터들을 Random하게 발생시켰을 때 처리되는 처리율을 표시하였다. 그림 4를 보면 알 수 있듯이, 5:5의 비율로 데이터를 발생시켰을 경우에는 입력과 출력에 따른 버퍼를 저장과 전송에 따른 프레임의 데이터들을 Process에서 점유를 하기 때문에 GSS 방법, HDS 방법 그리고 가중치 기반의 DVDS 방법들 모두가 프레임 데이터들의 처리량에는 변화가 없지만, 데이터의 발생 비율이 7:3과 8:2 비율로 프레임의 데이터를 발생시켰을 경우에는 GSS 방법과 HDS 방법은 7:3으로 발생하지만 실제 프레임 데이터들 중에 5:3의 데이터 즉, 발생되는 80%의 데이터를 처리하게 된다. 반면에 가중치 기반의 DVDS 방법의 경우에는 저장이나 전송에서 사용하지 않는 여유 버퍼가 존재하는 Process에 데이터를 처리 할 수 있도록 구현이 되어 있어 GSS 방법과 HDS 방법에 비해 처리 할 수 있는 데이터의 양이 현저하게 증가함을 그림 4에서 볼 수 있다.

또한 Read Unit의 증가에 따른 최대 초기 대기 시간을 비교 해보면 GSS 방법의 경우 Round 시간의 2배의 초기 대기 시간이 필요한 반면, HDS 방법의 경우에는 라운드 시간이 클 경우에는 여러 개의 Slot으로 나누어 수행되고, 새로운 사용자의 디스크 Read 요구는 우선적으로 가까운 Slot에서 처리가 되

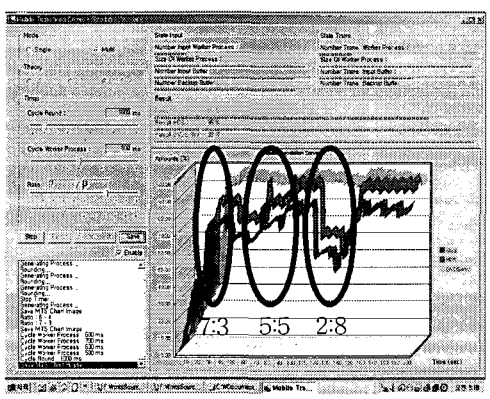


그림 4. GSS/HDS/DVDS의 처리 프레임 수

어 초기 대기 시간이 클라이언트들의 수가 증가함에 따라서 일정한 비율로 증가한다. 본 논문의 방법 또한 HDS 방법과 동일하게 적용했음에도 불구하고 클라이언트들의 가중치 값을 계산하여 가중치가 높은 클라이언트들을 먼저 처리하는 과정에서 대기시간이 HDS 방법보다 늘어나는 결과를 그림 5에서 볼 수 있다.

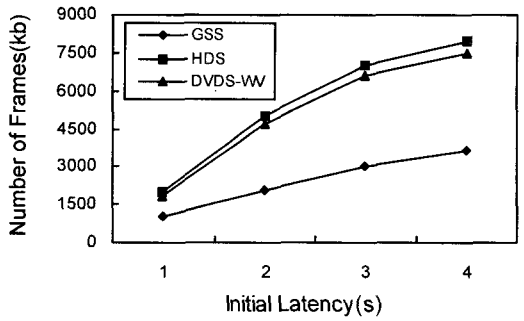


그림 5. GSS/HDS/DVDS의 초기 대기 시간

5. 결 론

본 논문에서 가중치 기반의 DVDS 방법은 기존 논문들에 비해서 동시에 지원할 수 있는 사용자의 수의 증대와 초기 대기시간의 감소는 물론 사용되지 않고 낭비될 수 있는 여유 자원의 활용을 통한 처리율의 극대화를 이루려고 하였으며, 서버의 이런 능력을 향상시키는 것이 모바일 멀티미디어 저장 서버의 필수적인 요소임이 분명하다는 것을 알 수 있다. 성능 비교에서 나타난 바와 같이 모바일 멀티미디어 서버에서는 슬롯을 교체하는데 사용되는 디스크의 seek time보다 사용자의 데이터들을 바꾸어주는 Rotation Time이 라운드시간의 대부분을 차지하므로 처리 할 수 있는 데이터 사이즈의 증가와 동시에 지원할 수 있는 사용자 수의 최대화, 또한 낭비되는 자원의 효율적인 사용이 반드시 함께 해결되어야 하는 것임을 확인 할 수 있었다.

요컨대 기존 방식에서 자원들의 효율성을 높이는 기법을 추가한 것만으로도 모바일 데이터 처리 서버의 효율이 상당히 높아질 수 있음을 알 수 있었다.

참 고 문 헌

[1] E.Park, N.Kim, S.Park, J.Kim, and H. Shin, "Dynamic Disk Scheduling for Multimedia Storage Server," *In Proceedings of 1999 IEEE region conference (TENCON'99)*, Vol. 2, pp. 1483-1486, (Cheju, Korea), Sep. 1999.

[2] P.S. YU, M.S Chen, and D.D Kandlur, "Grouped Sweeping Scheduling for DASD-based Multimedia Storage Management," *Multimedia System*, Vol. 1, No. 1, pp. 99-109, Jan. 1993.

[3] P.S. Mah, C.S. Cho, Y.S. Jin, and G.S. Shin, "Design of a Video Storage Server that Maximizes Concurrent Streams and Minimizes Initial Latency," *Korea Information Science Society*, Oct. 1999.

[4] Jinsung Cho and Heonshik Shin, "A Multi-resolution Video Scheme for Multimedia Servers in Mobile Computing Environment," *Proc. of International Conference on Telecommunications*, Vol. 2, Cheju, Korea, Jun. 1999.

[5] J.R. Santos, R.R. Muntz, and B.R. Neto, "Comparing Random Data Allocation and Data Stripping in Multimedia Servers," *SIGMETRIC*, 2000.

[6] S.W. Lau, John C.S, and Lui, L. Golubchik, "Merging Video Streams in a Multimedia Storage Server: Complexity and Heuristic," *Journal of Multimedia Systems*, Vol. 6(1), pp. 29-42, Jan. 1998.

[7] Hridesh. Rajan, "Distributed Mobile Servers: Vehicle to Increase Effective Bandwidth and End to End Connectivity in an Ad Hoc Mobile Network," *In the proceedings of SCI 2001/ ISAS 2001*, Vol. XII, July. 2001, Orlando, USA.

[8] B.R. Badrinath and Shirish Phatak, "On Clustering in Database servers for Supporting mobile clients," *To appear in Cluster Computing (Special Issue on Mobile Computing)*, Fall. 1998.

[9] J.R. Santos and R. Muntz, "Design of the RIO (Randomized I/O) Storage Server," *Technical Report, UCLA Computer Science Department*, May. 1997.

[10] R. Muntz, J.R. Santos, and S. Berson, "RIO: A Real-Time Multimedia Object Server," *Performance Evaluation Review*, ACM Press, Vol. 25, No. 2, pp. 29-35, Sep. 1997.

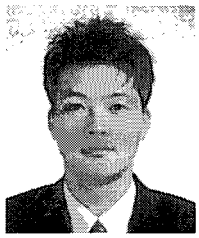
[11] B. Ozden, R. Rastogi, A. Silberschatz and P.S. Narayan, "The Fellini multimedia storage system." *In Journal of Digital Libraries*, 1997.

[12] J.R. Santos and R. Muntz. "Comparing Random Data Allocation and Data Striping in Multimedia Servers," *Technical Report*, UCLA Computer Science Department, Nov. 2000.

[13] B. Li, C. Lin, and S. Chanson, "An Efficient and Adaptive Bandwidth Allocation Scheme for Mobile Wireless Networks Based on Intelligent On-Line Parameter Estimations," *ACM/Kluwer J. Wireless Networks*, Vol. 7, No. 2, pp. 107-116, Mar./Apr. 2001.

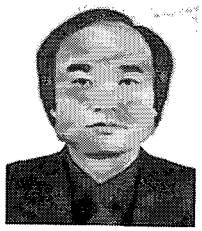
[14] Si. Wu, K. Y. Michael Wong, and Bo. Li, Se, "A Dynamic Call Admission Policy With Presision Qos Guarantee Using Stochastic Control for Mobile Wireless Networks," *IEEE TRANSACTIONS ON NETWORKING*, Vol. 10, No. 2, April 2002.

[15] D. Zhu, R. Melhem, and B.R. Childers, "Scheduling with dynamic voltage/speed adjustment using slack reclamation in multiprocessor real-time systems," *IEEE Transactions on pds*, Vol. 14, No. 7, pp. 686-700, July 2003.



조 종 근

1998년 성결대학교 컴퓨터공학(공학사)
2001년 숭실대학교 컴퓨터학과(공학석사)
2004년 숭실대학교 컴퓨터학과(공학박사)
2004년~현재 모바일 3D 표준화 포럼/전문위원
2004년~현재 (주)GOMID/수석연구원
관심분야 : 모바일 컴퓨팅, 멀티미디어, 컴퓨터 그래픽스



임 영 환

1977년 경북대학교 수학과(이학사)
1979년 한국과학원 전산학과(공학석사)
1985년 Northwestern University 전산학과(공학박사)
1979년~1996년 한국전자통신연구소 책임연구원
1996년~현재 숭실대학교 미디어학부 부교수
관심분야 : 멀티미디어