주문형 비디오 서버의 버퍼 최소화를 위한 가변적 서비스 모드 변환

(Adaptive Service Mode Conversion to Minimize Buffer Space Requirement in VOD Server)

원 유 집 †
(Youjip Won)

요 약 CPU, 네트웍 대역폭, 디스크 속도등 컴퓨터와 통신 기반을 이루는 기반기술의 급진적인 발달에 따라, 컴퓨터 또는 단말기로 멀티미디어 데이타 서비스를 받는 것이 이제는 우리 생활의 일부분이 되었다. 이러한 급속한 서비스 저변의 확대에도 불구하고 아직도 고화질 멀티미디어 서비스를 제공하는 데 있어서 많은 기술적인 문제가 존재하는 것이 현실이라고 할 수 있겠다. 그 중의 하나로 멀티미디어 정보를 디스크로부터 읽어들여 실시간 상영하는 경우, 과도한 주기억 장치 버퍼의 요구가 문제점으로 등장한다. 주기억 장치 버퍼가 필요한 이유는 디스크는 자료를 비동기적으로 읽는데 반해 멀티미디어 데이타(프레임)를 상영하는 방법은 동기적으로 행해지므로 두가지 특성이 다른 작업간에 비동기성을 해결하기 위함이다. 사용자에게 스트리밍 데이타를 전송하는 방법에는 두 가지(디스크에서 읽어들이는 방법: 디스크 모드 와 기존에 메모리에 탑재된 데이타를 재 전송하는 방법: 메모리 모드)가 있는데, 각 방법에 따라 필요로 하는 주기억 장치 버퍼의 양이 다르다. 본 연구에서는 각 방법에 따른 주기억 장치 요구량을 계산하는 모델을 개발하고, 전체 버퍼양을 최소화하도록 자료 전송방법을 가변적으로 변환시키는 기법을 소개한다. 본 기법의 가장 큰 장점은 각 비디오 세션의 데이타 전송하는 방법이 서버의 상태에 따라서 가변적으로 변환된다는 사실이다. 본 기법은 대용량 비디오 서버에서 다수의 멀티미디어 세션을 상영하는 데 필요한 버퍼 양을 효과적으로 감소 시킬수 있으며, 특히 사용자들의 주문이 소수의 화일들에게 집중되어있는 경우 더욱 효과적으로 작동하고 있다. 제안된 기법의 근간이 되는 이론들의 구체적인 모델링이 제공되었으며, 본 기법이 항상 최적의 해를 구한다는 사실은 증명을 통해 보여진다. 주창되는 기법의 효율성과 성능을 시뮬레이션을 통해서 검증한다.

Abstract Excessive memory buffer requirement in continuous media playback is a serious impediment of wide spread usage of on-line multimedia service. Skewed access frequency of available video files provides an opportunity of re-using the data blocks which has been loaded by one session for later usage. We present novel algorithm which minimizes the buffer requirement in multiple sessions of multimedia playbacks. In continuous media playback originated from the disk, a certain amount of memory buffer is required to synchronize asynchronous disk. Read operation and synchronous playback operation. As aggregate playback bandwidth increases, larger amount of buffer needs to be allocated for this synchronization purpose. The focus of this work is to study the asymptotic behavior of the synchronization buffer requirement and to develop an algorithm coping with this excessive buffer requirement under bandwidth congestion. We argue that in a large scale continuous media server, it may not be necessary to read the blocks for each session directly from the disk. The beauty of our work lies in the fact that it dynamically adapts to disk utilization of the server and finds the optimal way of servicing the individual sessions while minimizing the overall buffer space requirement. Optimality of the proposed algorithm is shown by proof. The effectiveness and performance of the proposed scheme is examined via simulation.

• 본 연구는 2000년도 한국학술진흥재단의 신진교수 지원과제에 의해 지원을 받은 것임.

† 정 회 원 : 한양대학교 전자전기공학부 교수
yjwon@ece.hanyang.ac.kr

논문접수 : 1999년 8월 9일
심사완료 : 2001년 3월 19일

1. 서 론

1.1 연구동기

원거리의 사용자에게 멀티미디어 서비스를 제공하는 대한 주제에 관한 연구가 상당히 많이 진척이 되어왔으나, 주요 목표 응용분야(예를 들면 오락(entertainment), 교육등)에서 실용화되기에는 아직도 많은 문제점이 있는 실정이다. 그들중 가장 큰 장애 요인은 해당 분야의 전통적인 서비스 형태들(예를 들면 비디오 대여업)을 대체하기에는 아직도 온라인 멀티디미어 서비스 단가가 매우 높다는 것이다. 이의 가장 근본적인 요인은 멀티미디어 서비스가 상당히 많은 양의 시스템 자원을 소모하고 따라서, 서비스 제공단가가 전통적인 방식(비디오 대여 등)에 비하여 상대적으로 높기 때문이다. 본 연구에서는 주문형 비디오 서버를 설계하는 데 있어서 보다 많은 수의 비디오 스트림들을 최소한의 시스템 자원(주기억 장치, 하드디스크 대역폭)으로 지원하는 기법의 개발에 초점을 맞추고 있다.

디스크에 저장되어있는 동영상 화일을 화면의 떨림이나 기타 잡음없이(jitter-free)상영하기 위해서는, 일정량의 주기억 장치 버퍼가 각 서비스 세션에 할당되어야 한다. 디스크의 작동은 비 동기적인데 반하여, 화면에 동영상을 보여주는 것은 동기적인 작업이기 때문에, 서로 다른 행위적 특성을 가진 두 작업이 효과적으로 연동되기 위해서 일정량의 주기억 장치 버퍼가 필요하다. 이러한 목적으로 사용되는 주기억 장치내의 버퍼를 본 논문에서는 *동기화 버퍼*라고 부르기로 한다. 멀티미디어 자료의 실시간 상영을 지원하는 디스크 스케쥴링 기법은 90년대 초반부터 많은 논의가 되어왔는데, 요구되는 버퍼의 크기와 디스크 사용률과의 관계가 O(1/(1-utilization))형태를 가지고 있으며[24], 따라서 디스크 사용률(disk bandwidth utilization)이 100%에 접근할수록, 요구되는 동기화 버퍼의 크기가 매우 빠른 속도로 증가한다는 특징을 가지고 있다.

주문형 비디오 서비스 환경에서는 대다수의 상영 요구가 소수의 상위에 랭크되어 있는 화일들에게 집중되어 있다[22]. 다수의 멀티미디어 서비스 요청이 소수의 인기있는 비디오 화일에게 집중되어 있는 경우에, 동일한 화일이 짧은 시간간격을 두고 병렬적으로 상영될 수 있다. 이러한 경우, 나중에 시작한 서비스는 선행상영에 사용된 데이타 블럭들을 메모리에 저장해 두었다가 사용함으로써, 디스크로부터 데이타 블럭을 읽어들이는 동작을 절약할 수 있다. 후행상영은 디스크로부터 데이타를 읽어들이지 않기 때문에, 동기화 버퍼가 필요가 없어지게 되지만, 선행상영에 사용된 데이타 블럭들을 임시로 보관하기 위한 버퍼가 할당되어야 한다. 우리는 멀티미디어 데이타를 사용자에게 전송하는 데 있어서, 두 가지의 극단적인 상황을 예시할 수 있다. 첫 번째는 멀티미디어 서비스에 필요한 자료를 디스크로부터 실시간으로 읽어들이는 방법이고(디스크 모드), 두 번째는 필요한 모든 데이타를 주기억 장치에 먼저 탑재한 후, 실제 서비스는 주기억 장치로부터 지원하는 방법(메모리 모드)이다. 본 논문에서는 비디오를 상영하는 두 가지 모드(디스크 모드, 메모리 모드)의 특성을 구체적으로 분석하고 디스크의 사용정도에 따라 가변적으로 비디오의 상영모드를 변환하는 기법을 연구 개발한다.

1.2 연구배경

최근 몇 년간 멀티미디어 서버의 설계에 관한 연구 결과가 많이 등장하였다[2, 5, 12, 16, 17, 11, 19]. Anderson et al[2]은 멀티미디어 데이타의 상영을 Producer/Consumer의 개념을 이용하여 설명하였으며, 실시간 상영을 위한 시간적 제한 조건을 수학적으로 모델링 하였다. 비디오 서버 설계분야에서 초창기에 문제점으로 대두되었던 것은 비 동기적인 디스크의 작동방식으로부터 연속성을 제공하는 것이었다. Rangan[16]은 FIFO 스케쥴링 기법하에서 다중매체 상영에 필요한 연속성을 제공하는 해결책을 제시하였다. Kencham et al[12]는 SCAN 스케쥴링 기법하에서의 연속성을 제공하는 모델을 제시하였다. Chen et al.[5]은 Group Sweep Scheduling이라는 기법을 제안하였는데, 이것은 SCAN과 FIFO 스케쥴링 기법을 결합한 방식이다. 위에서 언급한 스케쥴링 기법은 각기 장점과 단점을 가지고 있는데, 차후에 좀더 자세히 설명하도록 하겠다. Reddy et al[17]는 연속성을 제공하는 데 있어서, 실시간 시스템에 입장에서 접근하는 기법을 제시하였다. 단일 디스크의 헤드 스케쥴링 기법과 더불어 서버에서 디스크 시스템을 어떻게 구현해야 하는가에 관한 보다 일반적인 해결책들이 등장하였다[8, 9, 20, 21, 13]. 다중매체의 상영을 위해서는 요구된 데이타 블럭들이 정확한 시간 조건을 만족하면서 시스템 내에서 이동하여야 한다. [14, 1]은 디스크 스케쥴링 기법을 디스크 배열(Disk Array)환경으로 확장하였다. 주문형 비디오 서버에서 후행 상영은 같은 화일을 이용하는 선행상영이 있을 경우 선행상영이 주기억 장치로 읽어들인 데이타 블럭들을 재사용 할 수가 있다. 이 경우 후행상영은 디스크 장치를 액세스할 필요가 없어진다. Dan et al.[6]은 일단 주기억 장치로 읽혀진 데이타 블럭들을 사용 후 일정 시간동안 주기억 장치 유지하는 방법을 제안하였다. 최근에는 zoning을 사용하는 디스크에

서 연속성을 보장하는 디스크 스케줄링 기법에 대한 연구가 제안되었다[10, 15].

1.3 공헌 내용

본 연구에서는 멀티미디어 서버에서 연속성을 만족시키는 디스크 스케줄링 기법의 공통적 특성을 찾아내었다. 개별적인 스케줄링 기법(SCAN, FIFO 등)의 특성에 관계없이, 연속성을 지원하는 디스크 스케줄링 기법들은 일련의 공통적인 특성을 가진다는 사실이며, 그것은 아래와 같이 표현된다.

> 동기화 버퍼의 크기와 디스크 사용률의 상관관계가 $O(\frac{1}{1-\alpha})$를 따른다.

위에서 α는 디스크의 최대 전송속도(Byte/sec)와 실제 디스크로부터 전송되고 있는 자료의 전송 속도와의 비율로서, 본 논문에서는 이를 *디스크 사용률*이라 부르기로 한다. 예를 들면, 디스크의 최대 전송속도가 20 MByte 이고 현재 디스크에서 전송되는 데이타의 속도가 5MByte/sec 라면, 디스크 사용률은 25%가 된다.

디스크 사용률(Disk Bandwidth Utilization)은 새로운 상영이 시작되거나, 종영됨에 따라 동적으로 변화하며, 이와 더불어 각 서비스 세션을 디스크 모드로 지원하기 위한 동기화 버퍼의 크기도 동적으로 변하게 된다. 본 연구에서는 하나의 비디오 세션을 디스크 모드 내지는 메모리 모드로 지원하는데 필요한 버퍼의 양을 구하는 구체적 모델을 개발하였으며, 이를 기반으로 총 버퍼 사용량을 최소화하는 각 세션의 서비스 모드 결정 알고리즘을 개발하였다. 디스크 사용 상황에 최적화된 서비스 모드를 결정하기 위하여, 본 논문에서 제안되는 알고리즘은 새로운 세션이 시작하거나, 상영중인 세션이 종영될 때마다 동적으로 실행된다.

본 논문은 다음과 같이 구성된다. 2장과 3장에서는 비디오 스트림을 디스크 모드로 그리고 메모리 모드로 지원하는데 있어서 필요한 버퍼의 크기를 모델링한다. 4장에서는 각각의 서비스 모드에 대한 특성을 설명한다. 5장에서는 2장, 3장, 그리고 4장에서 설명한 내용을 바탕으로 상영모드 설정문제를 구체적으로 정립하고, 해결하는 알고리즘을 제시한다. 6장에서는 문제를 각 세션이 다른 디스크 대역폭을 요구하는 일반적인 환경으로 확장하여, 제시된 알고리즘이 최적의 서비스 모드를 결정하는 사실을 증명한다. 7장과 8장에서는 각각 시뮬레이션을 통하여 개발된 알고리즘의 효율성을 시뮬레이션을 통하여 검증하고, 결론 및 향후과제를 기술한다.

2. 멀티미디어 상영을 위한 버퍼 요구량

멀티미디어 화일을 디스크에서 일정 단위씩(프레임, 블럭) 읽어 들여 상영하는 경우, 일정량의 버퍼가 서비스를 위하여 할당되어야 한다. 이 버퍼는 동기적인(synchronous) 상영 동작(화면에 주기적으로 동영상을 뿌려주는 작동, 예를 들면 1초에 30 프레임)과, 디스크로부터 데이타 블럭을 읽어들이는 비동기적(asynchronous) 작동을 서로 동기화 시켜주는 역할을 담당한다. 여러 개의 멀티미디어 세션들이 단일 디스크에서 지원되는 경우, 디스크에서는 각 세션에 필요한 데이타 블록을 주기적으로 읽어들인다. 멀티미디어 서비스의 기본요소인 연속적인 데이타 흐름을 보장하기 위하여, 각 디스크 블록들은 정해진 시간까지 주기억 장치로 탑재어야 한다. 이 조건을 만족시키기 위하여, 일정량의 시스템 자원(예를 들어, Disk bandwidth와 주기억 장치 버퍼)들이 각각의 멀티미디어 세션들에게 할당된다.

2.1 멀티미디어 상영조건

본 장에서는 다수의 멀티미디어 세션들을 단일 디스크로 지원하는 데 필요한 조건을 구체화한다. $s=\{s_1, s_2 \ldots s_n\}$을 세션들의 집합, 세션 i의 전송속도를 r_i로 정의한다. 다수의 동영상 서비스가 진행되는 경우, 운영체제(내지는 비디오 서버 소프트웨어)는 각 서비스 세션에 필요한 데이타 블록들 주기적으로 디스크로부터 읽어들이며, 각 세션에 대하여 데이타 블록을 읽어들이는 주기를 $T(s)$로 정의한다. n_i와 b를 각 주기마다 세션 s_i에 필요한 데이타 블럭의 개수, 그리고 데이타 블럭의 크기(Byte)라고 한다.

운영체제가 요청한 데이타 블록이 디스크로부터 제때 도착하지 못할 경우, 서비스 사용자 입장에서는 화면의 단기적 멈춤, 건너뜀, 흐트러짐 등이 발생하게된다. 이러한 현상을 방지하기 위하여 각 세션에 대해서, 매 주기마다 디스크로부터 읽어들이는 데이타양이 해당 주기동안 상영되는 데이타양보다 같거나 커야한다. 이러한 조건을 식 (1) 과 같이 표현할 수 있다.

$$\forall i, bn_i \geq r_i T(s) \quad (1)$$

각 세션은 식 (1)에서 나타낸 것처럼 매주기마다 bn_i Byte의 데이타를 필요로 하고, 매 주기마다 이에 해당하는 자료가 디스크로부터 탑재되어야 한다. 이를 디스크 입장에서 분석하면 또 하나의 조건식을 정립할 수 있다. 모든 세션은 매 주기마다 식 (1)을 만족할 만큼의 자료를 필요로 한다. 디스크가 모든 세션에게 이에 해당하는 분량의 멀티미디어 블록을 배분하기 위해서는 한 주기에 필요한 모든 데이타 블록을(모든 세션에 대하여)

읽는 데 소요되는 기간이 주기시간 $T(\mathbf{s})$보다 같거나 작아야 한다는 것이다. 이 성질을 관찰하면 식 (2)와 같은 관계식을 정립할 수 있다.

$$T(\mathbf{s}) \geq \sum_{i=1}^{n}\left\{\frac{bn_i}{B_{\max}}+\delta_i\right\} = \sum_{i=1}^{n}\frac{bn_i}{B_{\max}} + \sum_{i=1}^{n}\delta_i \qquad (2)$$

식 (2)에서 δ_i는 i 번째 세션이 요구하는 데이타 블록을 읽어들이는 데 있어서, 헤드를 이동하고 track의 바꾸는 등, 디스크 작동 중에서 데이타를 읽는데 직접적으로 관련되지 않은 제반 시간을 나타내다. 디스크 헤드의 작동을 모든 인자들, 예를 들면 seek time, latency, cruise time, settle down time을 고려하여 정밀하고 섬세하게 수학적으로 모델링하는 것은 가능하지만, 모델의 복잡성 때문에 실제적인 의미는 거의 없다고 할 수 있다. 디스크의 내부 구조와 작동에 대해서 관심이 있는 독자는 [18, 23]를 참조하기 바란다. $B_{\max}$는 디스크의 최대 데이타 전송속도로서(Bytes/sec) 디스크축의 회전속도와 디스크 표면의 자기 밀도에 의해서 결정된다. 디스크에서 데이타 블럭을 전송하는 경우, 전체의 소요 시간 중 디스크 헤드를 원하는 데이타 블럭이 저장된 위치(cylinder, secotr)로 이동하는 시간이 차지하는 비율은 매우 높다. 7200RPM으로 회전하고 평균 seek time이 5ms인 디스크를 고려해 보자. 디스크의 최대 전송속도를 5MByte/sec라고 가정하면, 말해서 2048Byte크기의 블럭을 읽는데 0.4msec가 걸린다. 위에서 언급한 평균 seek time과 평균 latency까지 고려하면, 실제로 2048Byte를 읽는 데 걸리는 시간은 9.6ms이다. 따라서, 2KByte의 데이타 블럭을 읽는 데 시간은 9.6ms가 되고, 전체 소요시간의 95%가 디스크 헤드 배치에 소요되는 것이다. 헤드 이동에 소요되는 시간을 줄이는 방법중의 하나로 데이타 블럭의 크기를 증가시키기도 한다[3].

디스크의 작동원리를 토대로 식 (2)의 $T(\mathbf{s})$를 크게 두 부분으로 나눌 수 있다. 하나는 데이타 전송에 소요되는 시간, $\sum_{i=1}^{n}\frac{bn_i}{B_{\max}}$이고, 나머지 한 부분은 디스크 헤드의 재배치에 관련된 소요시간이다. 디스크 헤드의 재배치에 관련된 부분을 $O(\mathbf{s})$라고 하자.

($O(\mathbf{s})=\sum_{i=1}^{n}\delta_i$). 디스크 헤드의 재배치에 소요되는 시간은 디스크 스케쥴링 알고리즘에 의해 결정되며, 데이타를 읽는데 소요되는 시간은 전송될 데이타 블럭의 개수 n_i,블럭의 크기 b 그리고 최대 전송 속도 $B_{\max}$에 의해 결정된다. $\mathbf{n}=<n_1, n_2, \ldots n_n>$을 n 개의 세션들이 한 주기 동안 읽어야 할 블럭의 개수를 나타내는 일차원 벡터라고 하자. 식 (1)과 (2)에서 표현된 조건은 식 (3)과 같이 정리 될 수 있다.

$$\mathbf{n} \geq \frac{O(\mathbf{s})\sum_{i=1}^{n} r_i}{\frac{b}{B_{\max}}(B_{\max}-\sum_{i=1}^{n} r_i)} \qquad (3)$$

$\sum_{i=1}^{n} n_i$는 한 주기 동안에 요구되는 전체 블럭의 개수이다. 따라서, 식 (3)에서 $\mathbf{n}$은 주어진 n개의 세션을 지원하는 데 필요한 버퍼 크기이다. 식 (3)의 형태를 살펴보면, 세션들의 전체 전송속도의 합이 디스크의 최대 전송속도에 근접할 수록, 요구되는 동기화 버퍼의 양이 급속히 증가함을 알 수 있다.

2.2 디스크 스케쥴링 기법과 헤드 이동

세션의 연속성을 보장하는 조건을 계산하기 위해서 위에서 식(1), (2)의 인자들이 구체적으로 수식화 되어야 한다. 총 버퍼의 크기를 정확하게 계산하기 위해서는, $O(\mathbf{s})$의 계산이 먼저 선행되어야 한다. $O(\mathbf{s})$의 계산은 보다 정밀한 모델링 기법을 필요로 한다. 최근 몇몇 연구 결과에서[18, 23]에서 디스크의 seek time에 대한 성질을 다음과 같이 기술하고 있다.

$$T_{seek} = \begin{cases} a_1 + b_1\sqrt{x} & \text{if } x \leq c \\ a_2 + b_2 x & otherwise \end{cases} \qquad (4)$$

식 (4)에서 x와 c 는 seek 동작 거리와 임계치를 cylinder의 개수로 나타낸 것이다. HP 97560 디스크에서, a_1, b_1, a_2, 그리고 b_2는 각각 3.24msec, 0.4msec, 8.0 msec 그리고 0.008msec 에 해당하고, c는 383 실린더이다[18]. C를 디스크에 존재하는 전체 실린더의 개수라고 가정하자. 우리는 각각의 스케쥴링 기법에 근거하여 n개의 세션, $\mathbf{s}=s_1, s_2,\ldots s_n$을 지원하는 데 필요한 헤드 이동 소요시간을 좀더 구체적으로 모델링 할수 있다. 디스크 스케쥴링 기법에 관한 것은 [7]을 참조하기 바란다.

SCAN 스케쥴링에서는 디스크의 헤드가 실린더의 가장 바깥쪽(또는 가장 안쪽)으로부터 가장 안쪽으로(또는 가장 바깥쪽)으로 이동하면서 데이타 블럭들을 읽어들인다. 따라서, 실린더들은 한 주기에 최대 한번씩 방문된다. $\frac{C}{n-1} \leq c$ 를 가정하면[12], SCAN 스케쥴링 기법에서의 이동 소요시간은 식 (5) 와 같이 표현될 수 있다. 주목할 점은 블럭들이 위치한 실린더간의 거리가 임계치보다 작을 경우는 헤드이동의 소요시간이 실린더 거리의 제곱근에 비례한다는 사실이다.

$$O_{SCAN}(\mathbf{s}) = (n-1)(a_1+b_1)\sqrt{\frac{C}{n-1}} \qquad (5)$$

FIFO 스케쥴링 기법[16]의 경우는, 데이타 블럭의 위치에 관계없이 블럭들을 일정한 순서대로 읽어들인다. 따라서, 최악의 경우에는 디스크 표면을 n-1번 왕복할

수 있다. 따라서, 이에 근거한 이동 소요시간은 식 (6)과 같이 표현 될 수 있다.

$$O_{FIFO}(\mathbf{s})=(\mathbf{n}-1)(a_2+b_2C) \quad (6)$$

[5]에 의해서 개발된 Group Sweep Scheduling 의 경우에는, 세션들을 몇 개의 그룹으로 분할해서 그룹 내에서는 SCAN 기법을 이용하고, 그룹간에는 FIFO 기법을 이용해서 헤드를 이동한다. 그룹의 개수를 g라고 가정하면, GSS 기법하에서는 최악의 경우 헤드가 디스크 표면을 g-1번 왕복한다. n개의 세션들을 g개의 그룹으로 균일하게 분할할 경우 한 그룹의 크기는 최대 $\lceil \frac{n}{g} \rceil$이다. 이에 근거한 이동 소요시간, $O_{GSS}(\mathbf{s})$는 수식 (7)과 같이 표현된다.

$$O_{GSS(s)}=\left(\lceil \frac{n}{g} \rceil -1\right)\left(a_1+b_1\sqrt{\frac{C}{\lceil \frac{n}{g} \rceil -1}}\right)+(g-1)(a_2+b_2C) \quad (7)$$

식 (5), (6), (7)에서 나열된 각각 스케줄링 기법에 대한 헤드이동 소요시간을 보면, SCAN 기법이 최소 그리고, FIFO 기법이 최대의 헤드이동 소요시간을 갖는다는 것을 알 수 있다. SCAN 기법에서 디스크 헤드가 각 블럭을 읽을 때마다 디스크 표면을 안쪽부터 바깥쪽까지 이동해야 하는 것을 고려하면, 세션 수가 적을 때에는 FIFO 기법이 좀더 유리하다고 할 수 있다.

2.3 디스크 스케줄링 기법에 따른 버퍼 요구량

멀티미디어 서버에서 주기억 장치 버퍼는 두가지의 역할을 담당하고 있다. 첫 번째는 디스크로부터 읽혀진 데이타 블럭들을 임시로 저장하는 역할을 한다. 둘째로, 동기적인 상영 동작과 비 동기적인 디스크 작동을 연동시키기 위한 완충역할을 담당한다. 식 (3)의 $O(\mathbf{s})$를 디스크 스케줄링 기법에 근거해서 계산하면, 각각의 디스크 스케줄링 기법하의 버퍼 요구량, $\mathbf{n}$,을 구할 수 있게 된다.

SCAN 기법에서는 데이타 블럭들을 실린더 순서가 증가하는(또는 감소하는) 순으로 읽는다. 따라서, 연속된 2개의 주기에서 한 세션이 데이타를 읽어들이는 시점은 최소의 경우 서로 인접할 수 있고, 최대 $2T(\mathbf{s})$만큼 떨어져 있을 수도 있다. 이러한 시간간격의 편차를 보정해 주기 위해서, 더블 버퍼링 기법이 사용되어야 하며, 따라서, 버퍼 크기는 식 (8)과 같이 표현될 수 있다.

$$\mathbf{n}_{SCAN}=2\left(\frac{O_{SCAN}(\mathbf{s})\sum_{i=1}^{n} r_i}{\frac{b}{B_{\max}}(B_{\max}-\sum_{i=1}^{n} r_i)}\right) \quad (8)$$

FIFO 스케줄링 기법에서는, 한 세션에 대해서 각 주기마다 데이타 블록을 읽는 동작은 $T(\mathbf{s})$만큼 떨어져 있다. 각 세션은 데이타 블럭이 메모리로 읽혀진 직후 해당 Data 블록을 소모할 수 있다. FIFO 스케줄링 기법에서 각 세션이 필요로 하는 버퍼의 양은 식 (9)와 같이 표현된다.

$$\mathbf{n}_{FIFO}=\frac{O_{FIFO}(\mathbf{s})\sum_{i=1}^{n} r_i}{\frac{b}{B_{\max}}(B_{\max}-\sum_{i=1}^{n} r_i)} \quad (9)$$

GSS 기법은 FIFO기법과 SCAN 기법이 결합된 형태이다. 그룹간의 액세스 순서에는 FIFO기법이 적용되고, 그룹 내에서 데이타를 읽어들이는 작동은 SCAN 기법이 적용된다. 인접한 주기들 사이에서, 한 그룹은 $T(\mathbf{s})$만큼의 시간 간격을 가지고 있다. 각 그룹내에서 읽어야 할 데이타 블럭의 양이 거의 동일하다면, 한 세션에 관련된 데이타 읽기 동작은 인접한 주기에서 최소 $T(\mathbf{s})$, 그리고 최대 $T(\mathbf{s})\left(1+\frac{1}{g}\right)$ 만큼의 시간 간격을 가지고 있다. 이에 관련된 버퍼 요구량은 식 (10)과 같이 표현될 수 있다.

$$\mathbf{n}_{GSS}=\left(1+\frac{1}{g}\right)\left(\frac{O_{GSS}(\mathbf{s})\sum_{i=1}^{n} r_i}{\frac{b}{B_{\max}}(B_{\max}-\sum_{i=1}^{n} r_i)}\right) \quad (10)$$

2.4 상영 가능조건의 일반화

식 (8), (9), (10)의 공통적인 특성은 전송속도의 총합이 디스크의 최대 전송속도에 접근함에 따라, 버퍼 요구량이 급격히 증가한다는 것이다. $M(\mathbf{s})$를 세션들의 집합 $\mathbf{s}$를 지원하는 데 필요한 총 버퍼라고 하자. 식 (8), (9), (10)을 관찰하면, 식 (11)과 같은 일반화된 형태의 식을 유추해 낼 수 있다.

$$M(\mathbf{s})=\alpha_*\chi_*\frac{\sum_{i=1}^{n} r_i}{B_{\max}-\sum_{i=1}^{n} r_i} \quad (11)$$

식 (11)에서 α_*와 χ_*는 각각 디스크 스케줄링 기법에 의해 결정되는 계수와, $\frac{O(\mathbf{s})B_{\max}}{b}$를 나타낸다. α_*는 SCAN 기법의 경우 2, FIFO 기법의 경우 1, 그리고, GSS 기법의 경우 $\left(1+\frac{1}{g}\right)$에 해당한다. 한가지 재미있는 성질은 α_*가 크기가 스케줄링 기법에 의한 이동 소요시간에 반비례한다는 사실이다.

3. 메모리 모드 상영에서의 버퍼 요구량

멀티미디어 화일 i를 현재 n 명의 사용자가 시청하고 있고, 각 사용자의 서비스는서로 다른 시간에 시작되었다고 가정하자. 해당하는 서비스 세션들의 집합을 시작 시간순으로 나열하고, 이를 $\mathbf{s}=s_i^1, s_i^2, \ldots, s_i^n$ 로 표현하기로 하겠다. $t(s_i^k)$는 s_i^k(화일 i를 이용하는 k번째 세션)의

상영 시작시간, Δ_i^k는 같은 파일을 사용하는 인접한 후행세션과의 시간 간격으로 정의한다. 따라서 Δ_i^k는 $t(s_i^{k+1})-t(s_i^k)$로 표현될 수 있다.

하나의 스트리밍 세션에 필요한 데이타 블럭들을 제공하는 데 두 가지 방법이 있다. 첫 번째는 해당 데이타 블록을 디스크로부터 주기적으로 읽어들이는 경우이고, 이를 디스크 모드 서비스라고 한다. 두 번째 방법은 현재 진행중인 세션이 동일한 파일을 사용하고 있는 경우, 선행 서비스가 사용한 데이타 블록들을 주기억 장치에 저장해 두었다가 재 사용하는 방법이다. 후자의 방법으로 지원되는 경우를 메모리 모드 서비스라 한다. 메모리 모드로 서비스를 지원하는 경우에 데이타를 디스크로부터 공급받지 않기 때문에, 동기화 버퍼와 디스크 대역폭을 절감할 수 있다. 메모리 모드로 서비스를 지원하면, 디스크 모드에서 요구되었던 주기억 장치 및 디스크 대역폭등을 절감할 수 있으나, 서비스를 지원하기 위한 주기억 장치 버퍼가 필요하다. 세션, s_i^k,가 메모리 모드로 상영되는 경우, 선행 서비스가 이미 사용한 데이타를 $\Delta_i^{(k-1)}$동안 주기억 장치에 보존하기 위한 버퍼가 요구된다. s_i^k 를 메모리 모드로 지원하고자 할 경우, s_i^k의 데이타 전송속도를 r_i라고 한다면, 필요한 버퍼의 크기는 $\Delta_i^{k-1}r_i$가 된다.

한가지 중요한 사실은 서비스를 디스크 모드로 지원하는 경우 디스크 사용률이 증가할수록(단위시간당 총 전송량이 증가), 각 세션에 필요한 동기화 버퍼의 양이 증가한다는 것이다(식 (11) 참조). 메모리 모드로 서비스를 지원하는 경우 이에 요구되는 버퍼의 크기는 디스크 사용률에 영향을 받지 않는다. 본 논문에서 집중적으로 탐구하는 것은 디스크의 사용률로 표현되는 서버의 상태에 따라, "각각의 세션을 어떤 모드로 지원해야 하는 가"라는 문제를 제기하고 이를 해결하는 방법을 제시하는 것이다. 서비스 모드를 결정하는 데 지표로 삼고 있는 것은 일련의 세션들의 집합을 지원하는 데 필요한 총 버퍼량이다.

4. 상영모드의 결정

4.1 문제 설정

멀티미디어 세션을 지원하는 데 필요한 주기억 장치 버퍼의 양은 해당 세션의 서비스 모드에 따라 다르다. 본 연구의 촛점은 일련의 멀티미디어 세션들을 최소의 버퍼를 사용하면서 지원하는 방법을 개발하는 것이다. 이 목적을 달성하기 위하여, 두 가지 상영모드의 장단점을 비교하고, 최적의 해(서비스 모드)를 결정하는 알고리즘을 개발한다.

먼저 문제를 형식화하는데 필요한 몇 가지 변수들을 나열하기로 하겠다. 전체 세션들의 집합을 S로 표시한다. 세션들의 집합 S를 요청된 비디오 화일에 따라 분류하여 화일 i와 관련된 세션들의 집합을 S_i로 표기한다. S_i에 속하는 상영들을 시작시간순으로 나열하고, $s_i^1, s_i^2, \ldots, s_i^n$으로 표기한다. S는 세션들의 서비스 모드에 따라 두 집합으로 나뉘어지며, 디스크 모드로 상영되는 세션들의 집합을 S_D, 메모리 모드로 상영되는 세션들의 집합을 S_M이라고 표시한다. 일련의 세션들을 지원하는 데 필요한 총 버퍼의 양을 표시하기 위해, 함수 $M()$을 사용한다. $M(S)=M(S_D)+M(S_M)$가 된다. $M(s_i^k)$는 세션 s_i^k를 상영하는 데 필요한 버퍼의 양이며, s_i^k가 디스크 모드 상영인 경우에는 동기화 버퍼의 크기를 나타내고, 메모리 모드 상영인 경우에는 선행 상영이 사용한 데이타 블럭을 임시로 저장하기 위한 버퍼의 크기를 나타낸다. $M(S_D)$는 식 (3)과 식 (5), (6), (7) 중의 하나를 이용해서 구체적으로 계산된다. 메모리 모드 상영과 디스크 모드 상영을 실제로 구분해서 나타내어야 할 경우, 메모리 모드 세션은 $\overline{s_i^k}$로 표시한다.

4.2 서비스 모드의 가변적 변환

디스크의 사용률이 동적으로 변화함에 따라(세션 시작과 종영에 따라), 디스크 모드의 세션이 필요로 하는 동기화 버퍼의 크기가 변화한다. 이 현상은 상당히 중요한 의미를 가진다. 디스크 사용률이 증가하면, 현재 디스크 모드로 서비스되고 있는 세션에 대하여, 이 세션을 메모리 모드로 지원하는 것이 버퍼크기 측면에서 더 효율적일 수 있다. 이 경우 서비스 모드를 디스크 모드에서 메모리 모드로 변환함으로써 총 버퍼의 양을 감소시킬 수 있다. 물론 이와 반대의 경우, 즉 서비스 모드를 메모리 모드에서 디스크 모드로 변환함으로써 버퍼의 양을 감소시키는 경우도 있을 수 있다. 버퍼의 사용을 최적화하기 위해서는 각 세션의 서비스 모드를 디스크의 사용률이 변화함에 따라 동적으로 변화시켜 주어야 한다. 서비스 모드의 변환을 표기하기 위하여 →를 사용하기로 한다.

세션 s_i^k가 현재 디스크 모드 상영되고 있다고 가정하자. s_i^k를 메모리 모드로 변환하면, 동기화 버퍼를 반납하고, 선행 세션이 사용한 데이타를 임시로 저장하기 위한 버퍼를 할당받게 된다. s_i^k가 디스크 모드에서 메모리 모드로 변환되면 디스크의 사용률이 낮아지므로, 디스크 모드로 상영되고 있는 다른 세션들의 버퍼 크기도 감소

하게된다(식 (11) 참조). 세션의 서비스 모드를 변환하는 것이 결과적으로 총 버퍼크기를 감소시키는 가를 결정하기 위하여 본 논문에서는 *수익성(Profitability)*라는 용어를 사용한다.

> **Definition.** *세션 s_i^k의 상영모드를 변환하는 것이 궁극적으로 전체 버퍼크기의 감소를 가져오게 되면, 변환은 수익성이 있다고 정의한다.*

디스크 모드 상영을 메모리 모드 상영으로 전환하는 경우에 대한 수익성 여부를 판별하는 모델을 살펴보기로 한다. 디스크 모드 상영 s_i^k를 메모리 모드로 전환하게 되면, 변환 후 디스크 모드로 상영되는 세션들의 집합은 $\mathbf{s}_D-\{s_i^k\}$가 된다. s_i^k를 메모리 모드로 변환하는 전과 후의 디스크 모드 상영에 필요한 동기화 버퍼의 크기는 각각 $M(\mathbf{s}_D)$와 $M(\mathbf{s}_D-\{s_i^k\})$이다. 그리고, s_i^k를 메모리 모드로 서비스 하는데 필요한 버퍼의 크기는 $\Delta_i^{k-1}r_i$이다. s_i^k를 메모리 모드로 서비스 함으로써 감소하는 동기화 버퍼의 크기가 $\Delta_i^{k-1}r_i$보다 크면 변환은 수익성이 있다고 판별되며 이 조건은 식 (12)와 같이 기술될 수 있다.

$$M(\mathbf{s}_D)-M(\mathbf{s}_D-\{s_i^k\}) \geq \Delta_i^{k-1}r_i \qquad (12)$$

변환으로 인한 버퍼크기의 감소를 *수익* 이라고 하며, $P_s(s_i^k \rightarrow \overline{s_i^k})$로 표시한다. 디스크에서 메모리 모드로의 변환에 의한 수익은 구체적으로 식 (13)과 같이 표기될 수 있다.

$$P_s(s_i^k \rightarrow \overline{s_i^k}) = M(\mathbf{s}_D)-M(\mathbf{s}_D-\{s_i^k\}) - \Delta_i^{k-1}r_i \qquad (13)$$

세션이 새로이 시작되거나, 종영되면 각 세션들은 현재의 서비스 모드가 최적인가에 대한 적합성을 검증받는다. 디스크 모드에서 메모리 모드로 변환시키는 것을 상영의 *합병*이라고 하고, 메모리 모들에서 디스크 모드로 모드가 변환시키는 것을 상영의 *분리*라고 한다.

이를 요약하여 각 세션의 서비스 모드를 결정하는 문제를 아래와 같이 정의한다.

> **Definition:** 서비스 모드 결정문제(Service Mode Determination Problem)
> 주어진 세션들의 집합, s,를 지원하는데 필요한 총 버퍼량, $M(\mathbf{s})=M(\mathbf{s}_M)+M(\mathbf{s}_D)$를 최소화 시키도록, 각 세션의 서비스 모드를 결정한다.

5. 가변적 서비스 모드 변환

5.1 모드 변환의 성질

먼저 모든 세션들이 동일한 전송속도를 요구하는 것으로 가정한다. 본 논문에서 제안되는 기법은 세션들의 전송속도가 다른 경우에도 각 세션의 최적화된 서비스 모드를 구할 수 있으며, 구체적인 설명은 6장에서 다루도록 하겠다.

$\mathbf{s}_D$를 디스크 모드로 상영되는 세션들의 집합, 그리고 집합 $\mathbf{v}$, ($\mathbf{v}\subseteq\mathbf{s}_D$)를 디스크 모드에서 메모리 모드로 변환될 세션들의 집합이라고 하자. $\mathbf{v}$에 속한 세션들의 서비스 모드를 메모리 모드로 변환하면 전체 동기화 버퍼의 크기가 감소하게 된다. 감소하는 버퍼의 양은 식 (14)와 같다. 식 (14)에 대한 구체적 기술은 [24]를 참조하기 바란다.

$$\Theta(M(\mathbf{s}_D))-\Theta(M(\mathbf{s}_D-\mathbf{v})) = \frac{B_{\max}\sum_{s_i\in\mathbf{v}} r_i}{(B_{\max}-\sum_{i=1}^{n} r_i)(B_{\max}-(\sum_{i=1}^{n} r_i-\sum_{s_i\in\mathbf{v}} r_i))} \qquad (14)$$

식 (14)는 다수의 세션 중에서 버퍼의 감소를 극대화하도록 서비스 모드 변환 대상 세션을 찾는 데 핵심적인 토대를 제공한다.

5.2 상영의 가변적 합병

<표 1>의 Merge 알고리즘은 새로운 서비스를 시작할 경우, 진행중인 디스크 모드 세션들 중에서 메모리 모드로 변환할 세션들을 결정한다. 새로운 서비스 요청이 도착하면, 먼저 동기화에 필요한 총 버퍼크기를 (*새로 도착한 주문을 디스크 모드로 지원한다고 가정*) 계산한다. 디스크 사용률의 증가로 인하여 각 디스크 모드 세션에 할당된 동기화 버퍼의 크기도 증가하게 된다. 경우에 따라서 디스크 모드로 서비스 되는 세션들 중에서 메모리 모드로 변환하는 변환의 수익성이 0보다 큰 세션이 존재할 수 있게된다. 기존의 디스크 모드 세션 중에서 적절한 세션들을 메모리 모드로 서비스하는 것이 총 버퍼 사용량 측면에서 더 경제적일 수도 있다는 점에 기인한다.

메모리 모드로 세션이 변환하게되면 해당 세션은 동일한 파일을 사용하는 선행 세션이 디스크로부터 읽어들인 데이타블록을 사용하게 된다. 디스크 측면에서는 하나의 블록을 읽어들여서 다수의 세션을 지원할 수 있게 되는 것이고, 이러한 특성으로 인하여 디스크 모드에서 메모리 모드로 서비스 모드를 변환하는 것을 세션의 합병(session merge)이라고 하겠다.

세션 합병에서의 핵심사안은 변환의 수익성이 0보다 큰 세션이 여러 개 존재할 경우, 버퍼의 감소량이 극대화 되도록 서비스 모드를 변환할 세션들을 선택하는 것

표 1 Merge 알고리즘

```
Merge(s){
1.    Set of Intervals: I1, I2 ;
2.    Set of Requests: s' ;
3.    s' ← { s_i^k ∈ s   such that  s_i^k  is in disk mode} ;
4.    Boolean: DONE ;
5.    I1 ← intervals in s' ;
6.    I2 ← φ  ;
7.    DONE ← False ;
   %Initializing profitable interval sets
8.    For  ∀Δ_i^k ∈  I1{
9.      If( P_s'(s_i^{k+1} → s_i^{k+1}) ≥ 0      ) then
10.         I2 ← I2 ∪ {Δ_i^k}      ;
11.     end if
12.   end for
13.   I2 ← sort(I2)     ; % With the increasing order of interval length
14.   while( not DONE){
15.       Δ_i^k ← ShortestInterval(I2)
16.     if( P_s'(s_i^{k+1} → s_i^{k+1}) ≤ 0      ) then
17.       DONE ← TRUE ;
18.     else
          I2 ← I2 − {Δ_i^k} ;
19.       s_i^{k+1} => s_i^{k+1}  ;
          s' ← s' − {s_i^{k+1}} ;
          s ← s − {s_i^{k+1}} ∪ {s_i^{k+1}}
20.     end if
21.   end while
22.   return(s) ;
}
```

이다. 이를 위하여 각 디스크 모드 세션의 변환 수익성을 계산하고 수익성이 큰 순서로 정렬을 한다. 수익성이 가장 큰 세션을 메모리 모드로 서비스 모드를 변환하고, 각 세션의 변환 수익성을 다시 계산한다. 다시 계산을 하는 이유는 디스크 모드에서 메모리 모드로 세션을 변환함에 따라 디스크의 사용률이 감소하고 따라서 각 세션에 대하여 서비스 모드 변환에 따른 수익성이 변화하기 때문이다. 모드 변환의 수익성은 식 (14)에 근거해서 계산된다. 이러한 과정을 디스크 모드 세션들의 변환 수익성이 모두 0보다 작을 때까지 반복한다.

표 1에서 라인 8-12 의 for 루프는 집합 I2를 수익성이 있는 세션들의 집합으로 초기화시킨다. 모든 세션들은 동일한 전송속도, r_t를 가지고 있다고 가정하기 때문에, 동일한 양의 동기화 버퍼를 가지고 있다. 따라서, 모드의 변환을 선행 상영과의 시간간격이 작은 세션부터 차례로 하게되면, 결과적으로 전체 수익성이 극대화 된다. 이에 근거해서 라인 14-24에 있는 while loop는 선행 시간과의 시간간격, Δ_i^k가 작은 세션부터 메모리 모드로 전환한다. 세션들이 차례로 메모리 모드로 변환됨에 따라, 디스크로부터의 총 전송속도가 감소하고, 각 디스크 모드 세션에게 할당된 버퍼의 크기가 작아진다. 세션들이 차례로 메모리 모드로 전환됨에 따라, 남아있는 디스크 모드 세션들의 변환 수익성이 점차로 감소하고, 마지막에는 0보다 작아진다. 시간간격, Δ_i^k,가 가장

작은 세션의 수익성이 0보다 작으면, 알고리즘은 수행을 종료하게 된다.

5.3 알고리즘의 최적성

Merge 알고리즘은 주어진 세션들의 집합에서 이들을 서비스 하기위한 총 버퍼량이 최소가 되도록 각 세션의 서비스 모드를 결정하며 본 장에서는 이에 대한 증명을 보이기로 한다.

Theorem 1.(최적성의 증명) $\mathbf{s}_D$와 $\mathbf{v}(\mathbf{v}\subset\mathbf{s}_D)$를 각각 디스크 모드로 지원되는 세션의 집합, 그리고, $\mathbf{s}_D$의 원소 중에서 메모리 모드로 전환될 세션들의 집합이라고 하자. 표 1의 Merge 알고리즘은 항상 수익성을 최대화시키는 집합 $\mathbf{v}$를 찾는다.

증명: 증명은 모순을 이용한다. Merge 알고리즘은 선행 상영과의 시간 간격이 짧은 세션부터 메모리 모드로 변환한다. 가장 작은 Δ_i^k를 가지는 세션의 수익성이 0보다 작으면 알고리즘은 수행을 종료한다. $\mathbf{v}$를 최적의 해, 다시 말해서 변환의 수익성이 극대화되는 세션집합이라고 하고, 세션 $s_i^k\notin\mathbf{v}$에 관해서 $\mathbf{v}$에 속하는 어떤 세션, s',의 선행상영과의 시간간격, Δ',이, s_i^k의 선행상영과의 시간간격, Δ_i^{k-1}보다 크다고 가정하자. $\mathbf{v}$를 변환시키는 총 수익은 아래의 식과 같이 계산될 수 있다.

$$P_s(\mathbf{v}\rightarrow\overline{\mathbf{v}}) = \frac{B_{\max}\sum_{s_i'\in\mathbf{v}} r_i}{(B_{\max}-\sum_{s_i'\in\mathbf{s}_D} r_i)(B_{\max}-\sum_{s_i'\in\mathbf{s}_D} r_i-\sum_{s_i'\in\mathbf{v}} r_i)} - \sum_{s_i'\in\mathbf{v}}\Delta_i^{k-1}r_i$$

집합 $\mathbf{u}$를 $\mathbf{v}-\{s_i^k\}\cup\{s'\}$ 라고 지정하자. 모든 세션들은 동일한 전송속도를 요구하고, $|\mathbf{v}|=|\mathbf{u}|$이다. 따라서, $\mathbf{u}$에 속한 세션들을 메모리 모드로 변환함으로써 감소되는 동기화 버퍼의 양은 $\mathbf{v}$와 $\mathbf{u}$의 경우 동일하고, 수 식 (15)와 같이 표현된다.

$$M(\mathbf{s}_D)-M(\mathbf{s}_D-\mathbf{v})=M(\mathbf{s}_D)-M(\mathbf{s}_D-\mathbf{u}) \quad (15)$$

$\mathbf{v}$와 $\mathbf{u}$를 변환하는 수익성의 차이는 아래와 같이 표현된다.

$$P_s(\mathbf{v}\rightarrow\overline{\mathbf{v}})-P_s(\mathbf{u}\rightarrow\overline{\mathbf{u}})=r(\Delta_i^{k-1}-\Delta') \quad (16)$$

본 증명의 초기 가정에서, $\Delta_i^{k-1}<\Delta'$. 따라서, 식 (16)은 0보다 작게 되며, $\mathbf{v}$의 수익성보다 $\mathbf{u}$의 수익성이 더 크다. 이 결과는 $\mathbf{v}$가 수익성을 최대화시키는 가정에 모순된다. 따라서, Merge 알고리즘은 항상 수익성을 최대화시키는 집합을 구한다. ■

5.4 상영의 가변적 분리

디스크 모드로 지원되는 세션이 종료하면, 디스크의 사용률이 감소하고, 이에 따라 각 디스크 모드 세션이 필요로 하는 동기화 버퍼 크기가 감소한다. 세션을 메모리 모드로 지원하는 데 필요한 버퍼양은 디스크 사용률에 영향을 받지 않으므로 메모리 모드로 지원되는 모든 세션들은 디스크 모드로 변환하는 것이 수익성이 있는지에 대한 수익성 여부를 검증받아야 한다. 이 검증과정은 디스크 모드세션이 종영될 경우에 실행된다. 각 메모리 모드 세션, s_i^k,가 사용하는 버퍼크기가 해당 세션이 디스크 모드로 변환될 경우 증가하는 동기화 버퍼의 양보다 클 경우, 다시 말해서 $\Delta_i^{k-1}r_i$이 $M(\mathbf{s}_D\cup\{s_i^k\})-M(s_n)-M(\mathbf{s}_D)$보다 크면, s_i^k의 서비스 모드는 디스크 모드로 변환된다. 변환으로부터 생기는 수익은 식 (17)과 같이 계산된다.

$$P_s(\overline{s_i^k}\rightarrow s_i^k)=\Delta_i^{k-1}r_i-\{M(\mathbf{s}_D\cup\{s_i^k\})-M(\mathbf{s}_D)\} \quad (17)$$

세션이 메모리 모드에서 디스크 모드로 변환되는 경우, 선행상영의 데이타 블럭을 재 사용하는 상태에서 디스크로부터 데이타를 읽어들이는 상태로 변화되므로, 세션이 분리된다고 호칭한다.

세션 종영시 수행되는 가변적 분리 알고리즘, *Split*,는 표2에서 보는 바와 같다. 라인 8-12 사이의 for loop는 분리의 결과로 총 버퍼량이 감소하는 세션들을 추출하여 집합 *I2*를 초기화한다. 버퍼량의 감소여부는 식 (17)에 근거한다. 모든 세션들은 동일한 전송속도를 가진다는 가정에 의해, 분리에 의한 수익성은 선행사영과의 시간간격에 비례한다. 따라서, 모드의 변환(메모리 모드로부터 디스크 모드로)은 선행상영과의 시간간격이 큰 세션부터 변환을 한다. 라인 14-21의 while loop를 보기로 하자. 여기에서는 I2에 속한 세션들을 차례로 디스크 모드로 변환한다. 디스크 모드로 지원되는 총 세션의 개수가 증가함에 따라, 디스크 모드의 각 세션에 할당되는 동기화 버퍼가 증가하게 되고, 라인 16이 언젠가는 0보다 작아지게 된다.

5.5 알고리즘의 복잡도(Time Complexity)

알고리즘의 우수성을 결정하는 척도로 최적의 해에 어느 정도 근접한 해를 찾을 수 있는가와 그 실행속도를 들수 있다. 본 논문에서 제안하는 알고리즘은 최적의 해를 찾을 뿐만 아니라, 수행시간도 세션의 개수에 선형 비례한다. 디스크 모드 세션들의 집합, s_D의 크기가 n이라 가정하자. 제안된 합병 알고리즘은 먼저 집합 s_D의 세션들을 수익성에 따라 정렬한다. 이때 요구되는 계산량은 $O(n\log n)$이다. 그리고, 표 1, 라인 14-24의 while

표 2 Split 알고리즘

```
Split(s){
    Set of Intervals: I1, I2 ;
    Set of Requests: s' ;
    s' ← { s_i^k ∈ s   such that  s_i^k  is in memory mode} ;
    Boolean: DONE ;
    I1 ← intervals in s' ;
    I2 ← φ ;
    DONE ← False ;
  %Initializing profitable interval sets
    For ∀Δ_i^k ∈ I1{
      If( P_s'( \overline{s_i^{k+1}} → s_i^{k+1} ) ≥ 0     then
          I2 ← I2 ∪ {Δ_i^k}      ;
      end if
    end for
    I2 ← sort(I2)     ; % With the increasing order of interval length
    while( not DONE){
      Δ_i^k ← LongestInterval(I2)        ;
      if( P_s'( \overline{s_i^{k+1}} → s_i^{k+1} ) ≤ 0      ) then
         DONE ← TRUE ;
      else
         I2 ← I2 − {Δ_i^k} ;
         \overline{s_i^{k+1}} => s_i^{k+1} ;
         s' ← s' − {s_i^{k+1}} ;
         s ← s − {s_i^{k+1}} ∪ {\overline{s_i^{k+1}}}
      end if
    end while
    return(s) ;
}
```

loop를 수행하는 데 필요한 시간은 $O(n)$이다. 따라서, 집합 s_D가 주어진 경우의 계산량은 $O(n\log n)$이 된다.

실제 상황에서는 가변적 합병/분리 알고리즘이 새로운 세션이 시작할 때, 그리고, 기존의 상영이 종료될 때마다 시행된다. 또 하나 중요한 의미를 가지고 있는 알고리즘의 복잡도는 n개의 주문이 순차적으로 도착할 경우, 총 계산량(amortized time complexity)이 얼마인가 하는 것이다. 새로운 주문이 도착하는 경우를 고려해 보자. 기존의 변환의 수익성에 기반하여 정렬된 세션들의 집합(수익성이 0보다 큰 변환들)에 새로운 원소를 추가할 경우에는 이진 탐색을 이용하여 새 주문의 위치를 결정할 수 있다. 그 다음은 집합에 속한 상영들에 대하여 수익성을 다시 계산하고, 계산된 수익성에 근거하여 상영모드를 변환한다. 수익성을 재 계산하는 작업의 계산량은 원소의 개수에 비례한다. 이런 성질을 고려하면, 기존의 집합에 i개의 원소가 있는 경우, 새로운 주문의 도착은 $O(\log i+i)$가 된다. 따라서, n개의 작업이 순차적으로 도착하는 경우의 총 계산량, $T(n)$은 아래와 같이 표현될 수 있다.

$$T(n)=\sum_{i=1}^{n} O(i+\log i)=O(n^2) \tag{18}$$

식 (18)에서와 같이 본 논문에서 제안하는 합병/분리 기법은 매우 효율적으로 수행된다 할 수 있겠다.

6. 동일 전송속도 요구조건의 일반화

대용량 주문형 비디오 서버에서는 각 사용자들이 각기 다른 품질의 서비스를 원할 수 있다. 예를 들면, 멀티미디어를 이용한 원격진료 같은 경우 실 시간성과 더불어 고 선명도의 화질이 반드시 보장되어야 하는 반면, 오락이나 교육분야의 멀티미디어 서비스의 경우에는 화질이나 음질에 대한 사용자의 요구가 비교적 완화되어있다. 음질이나 화질은 데이타의 전송속도가 직접적으로 연관

되어 있으며, 고화질, 고음질의 서비스일수록, 단위시간당 전송해야할 자료의 양이 많다. 본 장에서는 세션들이 각기 다른 전송속도를 갖는 경우에도, *Merge/Split* 알고리즘이 최적의 해를 찾는다는 성질을 증명한다. 증명에 앞서 디스크에 새로운 세션이 시작될 때의 버퍼량의 증가에 관한 중요한 특성을 언급한다. 새로운 세션이 추가됨으로써 증가하는 총 동기화 버퍼의 양은, 새로운 세션의 전송속도에 초 선형적(hyper linearly)으로 비례한다는 것이다[24]. 이해를 돕기 위하여 간단한 예를 들도록 하겠다. 두 개의 세션 s_x와 s_y는 각각 2Mbits/sec 그리고 4 Mbits/sec 의 전송속도를 가진다고 가정하자. 어떤 세션 s_A를 디스크 모드로 추가함으로써 전체적으로 동기화 버퍼가 3 MByte라면 증가한다면, s_x대신 s_y를 추가할 경우 증가하는 총 동기화 버퍼의 양은 6 MByte 보다 같거나 크다. 일반화된 *Merge* 알고리즘은 세션들을 수익성이 큰 것부터 변환시키는 것을 골자로 한다. *Merge*(표 1의 13째 줄)에서는 세션을 시간간격이 작은 것을 우선으로 나열했다는 것이 다르다.

Theorem 2. 세션들의 집합 **s**, 디스크 모드 세션들의 집합 $\mathbf{s}_D$, $\mathbf{s}_D \subset \mathbf{s}$에서, 일반화된 Merge 알고리즘에 의해서 구한 변환 세션들의 집합 **v**, ($\mathbf{v} \subset \mathbf{s}_D$)는 총 버퍼의 크기, $M(\mathbf{s})$를 최소화한다.

증명: **v**를 버퍼크기를 최소화시키는 서비스 모드들의 집합이라하고, 다음 조건을 만족하는 세션 $s_i^k \notin \mathbf{v}$가 존재한다고 가정하자.

$$P_{s_D}(s_i^k \rightarrow \overline{}_{s_i^k}) > P_{s_D}(s' \rightarrow \overline{}_{s'}),\ \exists s' \in \mathbf{v}$$

위의 식이 의미하는 것은, **v**에 속한 세션보다 변환의 수익성이 큰 세션이 존재한다는 것이다. 상영의 집합 **u**와 **w**를 각각 $\mathbf{u} = \mathbf{v} - \{s'\} \cup \{s_i^k\}$ 그리고, $\mathbf{w} = \mathbf{v} \cap \mathbf{u}$로 정의하자. **v**에 속한 세션들을 메모리 모드로 변환하는 수익성을 식 (19) 과 같이 계산할 수 있다.

$$\mathbf{n}_{FIFO} = \frac{O_{FIFO}(\mathbf{s}) \sum_{i=1}^{n} r_i}{\frac{b}{B_{max}}(B_{max} - \sum_{i=1}^{n} r_i)} \qquad (19)$$

v의 변환에 관한 수익성을 **w**를 이용하면 아래의 식과 같이 정리된다.

$$P_{s_D}(\mathbf{v} \rightarrow \overline{}_{\mathbf{v}}) = P_{s_D}(\mathbf{w} \rightarrow \overline{}_{\mathbf{w}}) + P_{s_D - \mathbf{w}}(s_i^k \rightarrow \overline{}_{s_i^k})$$

따라서, 식 (20)의 오른편이 0보다 작다. 따라서, **v**에 있는세션들을 메모리 모드로 변환하는 것 보다, **u**에 있는 세션들을 메모리 모드로 변환하는 것이 더 높은 수익성을 가진다. 이것은 **v**가 최적의 해를 가진다는 초기 가정에 모순된다. 따라서, 최적의 해, 수익성을 최대화시키는 세션들의 집합은 변환의 수익성이 큰 순으로 선택하여 구할 수 있다. ■

$$P_{s_D}(\mathbf{v} \rightarrow \overline{}_{\mathbf{v}}) - P_{s_D}(\mathbf{u} \rightarrow \overline{}_{\mathbf{u}})$$
$$= P_{s_D - \mathbf{w}}(s' \rightarrow \overline{}_{s'}) - P_{s_D - \mathbf{w}}(s_i^k \rightarrow \overline{}_{s_i^k}) \qquad (20)$$

7. 성능평가

이번 장에서는 개발된 알고리즘이 어느 정도의 주기억장치 사용량을 절약할 수 있는지 여부를 시뮬레이션을 통하여 검증한다. 시뮬레이션 환경에서는 100개의 서로 다른 비디오 화일들이 존재한다. 비디오 대여형태는 소수의 인기비디오에 대다수의 요청이 집중되는 형태를 가지고 있다. 이를 효과적으로 표현하기 위하여, 화일들의 신청빈도는 Zipf분포를 이용하여 모델되었다. Zipf 분포에서 i번째로 자주 신청되는 비디오의 신청빈도는 식 (21)와 같이 표현된다.

$$P_i = \sum_{j=1}^{i} \frac{C}{j^{(1-\alpha)}},\quad C = \frac{1}{\sum_{j=1}^{n} \frac{1}{j^{(1-\alpha)}}} \qquad (21)$$

식 (21)의 α는 요구빈도가 어느 정도 편향되는가를 나타내는 지표이다. α가 1인 경우는 요구빈도가 균등하게 분포되어 있는 경우를 나타내고, 그와 반대로 α가 0인 경우는 대부분의 요구가 상위에 랭크된 소수의 파일에 집중되어 있는 분포를 표현한다. Zipf 분포의 구체적인 특성에 대한 언급은 본 논문의 범위를 벗어나므로, 생략하겠다. 서비스 요청이 도착하는 형태는 Poisson 프로세스를 이용하여 모델링하였다. 비디오 파일은 MPEG-1(1.5Mbits/sec)로 압축되어있으며, 따라서, 하나의 비디오 스트림을 서비스하는 데 필요한 전송 대역폭은 1.5Mbits/sec(187KBytes/sec)이다.

시뮬레이션에 사용된 디스크는 다음과 같은 특성을 가진다. 디스크의 회전속도는 7200RPM이다. 디스크의 블럭 사이즈는 4096바이트이고, 한 블럭을 읽는 데 0.2 msec가 걸린다. 따라서, 디스크 블럭사이즈가 4KByte로 포맷되었을 경우 최대 전송속도는 20MByte/sec가 된다[1]. 디스크는 1964개의 실린더를 가지고 있고, seek time은 [18]의 연구결과를 바탕으로 모델링 되었다(식 4 참조). 디스크 헤드는 SCAN 스케쥴링 기법으로 제어되며, 버퍼관리에 있어서, double buffering 기법을 사

1) Fibre-Channel을 사용하는 High-End server용 고성능 디스크 (Seagate Cheeta ST136403FC, http://www.sea gate.com)의 경우 평균전송 속도(포맷 후) 25.4 MByte/ sec 이다.

용한다.

시뮬레이션의 전반적인 구성은 그림 1과 같다. 서비스 요청이 도착하면 CACM(Call Admission Control Module)이 서비스에 필요한 대역폭, 요청된 파일, 서비스 품질에 대한 정보를 서비스 모드를 결정하는 모듈에 전달한다. SMDM(Service Mode Determination Module)은 전달받은 정보를 기반으로 서비스 요청의 가능여부와 서비스 모드를 결정한다.

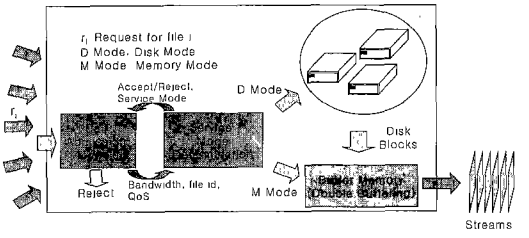

그림 1 시뮬레이션 환경구성도

CACM이 작동하는 구체적인 논리의 흐름은 그림 2와 같다. 먼저 디스크 모드로 서비스하는 것이 가능한지 여부를 판명하고, 메모리 모드로 서비스하는 것이 가능한지를 판명한다. 두 가지 방법이 모두 가능할 경우, 추가로 요구되는 메모리의 양을 비교하여 더 작은 추가 메모리를 요구하는 모드를 서비스 모드로 선택한다.

```
if( BDW_avail > r_i &&
   Buffer_avail > Increase in Synchronization Buffer Size)
   DiskMode = Yes ;
else
   DiskMode = No ;
if( Buffer_avail > Window Buffer)
   MemoryMode = Yes ;
else
   MemoryMode = No ;
if(DiskMode = No && MemoryMode = No)
   service_mode = reject ;
else{
   if( Window Buffer >
      Inncrease in Synchronization Buffer Size)
      service_mode = DiskMode ;
   else
      service_mode = MemoryMode
}
return(service_mode) ;
```

그림 2 CACM의 논리 흐름도

7.1 버퍼사용량의 감소

20MByte/sec의 최대 전송속도를 가지는 디스크는 동기화 버퍼로 사용되는 버퍼 크기의 제한이 없다면 최대 106개의 MPEG-1 stream(187KBytes/sec)를 이론적으로 지원할 수 있다. 모든 스트림을 디스크로부터 전송하는 경우 이에 요구되는 총 버퍼량을 살펴보기로 한다. <그림 1> 에서, x축은 스트림의 개수, y축은 해당 개수의 스트림을 지원하는 데 필요한 버퍼량을 보여주고 있다. 스트림의 개수가 최대치에 근접할수록 버퍼크기가 급격하게 증가하는 것을 관찰할 수 있다. <그림 1>에서 보는 바와 같이 디스크의 한계용량인 106개의 스트림을 지원하기 위해서는 약 6.2GByte의 주기억 장치 버퍼가 동기화를 위해 할당되어야 한다.

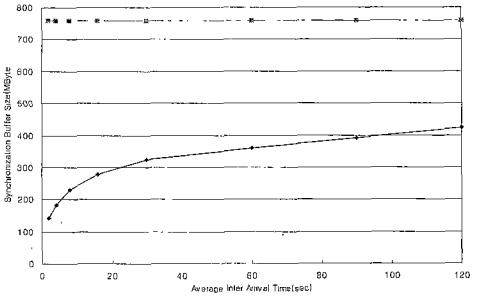

그림 3 100개의 스트림을 지원하기위한 버퍼 요구량

본 논문에서는 스트리밍 서비스 모드를 메모리 모드와 디스크 모드로 구분하고 이를 효과적으로 병행함으로써 주기억 장치를 보다 효율적으로 사용할 수 있다는 주장을 하고 있다. 이번 장에서는 시뮬레이션을 통하여 이의 효용성을 알아보도록 하겠다. 본 논문에서 주창하는 기법은 디스크 모드 서비스와 메모리 모드 서비스를 디스크의 사용률에 따라 동적으로 변환시켜 서비스에 필요한 버퍼 요구량을 최소화 한다는 것을 주지하기 바란다.

<그림 4>에서는 100개의 스트림을 지원하는 데 필요한 총 버퍼의 양을 나타내고 있다. 사용자의 요구는 Zipf 분포(α=9.271)를 따라서 분포되어 있다. 모든 세션을 디스크에서 서비스하는 경우와 가변적으로 서비스 모드를 변환하며 세션을 서비스하는 경우로 나누어

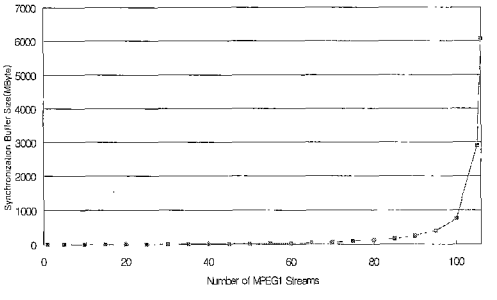

그림 4 스트림의 개수와 버퍼 요구량

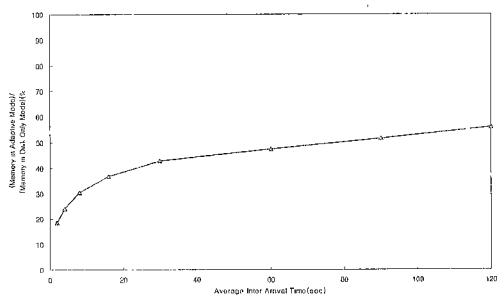

그림 5 100개의 스트림을 지원하기 위한 버퍼 요구량

보여주고 있으며, x축은 사용자 주문 요청의 평균 도착 간격을 나타내고 있다.

100개의 MPEG-1 스트림을 디스크 모드만으로 지원할 경우 <그림 4>에서 보는 바와 같이 760MByte의 동기화 버퍼가 필요하다. 하지만, 각각의 주문이 Poisson 프로세스로 도착하고, 사용자들의 주문이 어느 정도 대중적인 파일에 편향되어 있는 경우 상영의 합병을 이용하여 총 버퍼 요구량을 최소화시킬 수 있다. <그림 4>는 사용자 주문의 편향도가 Zipf 분포(α=0.271)를 따를 경우 필요한 총 버퍼량을 나타내고 있다. <그림 4>에서 가변적 서비스 모드 방법을 이용할 경우 평균 도착간격(inter-arrival time)이 클수록 총 버퍼의 양이 증가하는 것을 발견할 수 있다. 도착하는 서비스 요청간의 간격이 작을수록, 하나의 세션이 같은 비디오 파일을 사용하는 선행 세션과의 거리가 평균적으로 작아지고, 따라서 이를 메모리 모드로 지원하는 데 필요한 버퍼의 양이 작아진다. 서비스 요구가 빈번하게 도착하는 비디오 서버에서는, 세션들을 메모리 모드로 서비스함으로써 총 버퍼량을 감소시킬 수 있게 된다. 100개의 세션을 디스크에서(Disk Only Mode) 상영하는데 760MByte 의 주기억장치 버퍼가 필요하다. 가변적 서비스 모드 결정기법을 이용하면 평균 도착 간격이 2초인 경우 총 버퍼양이 140 MByte, 120초인 경우는 약 435MByte로 디스크에서 100개의 스트림을 지원하는 것에 비해 각각 18% 그리고 57%의 버퍼만을 사용한다. <그림 5>는 Disk Only Mode와 Adaptive Service Mode 기법을 사용한 경우 이들 간의 비율을 보이고 있다.

7.2 합병의 효율성

가변적 서비스 모드 변환기법을 사용하지 않는 경우 본 실험에서 사용된 디스크에서 최대 지원 가능한 스트림의 개수가 106개이며, 스트림의 개수가 디스크 용량의 한계에 근접함에 따라 버퍼의 양이 급격하게 증가함을 관찰할 수 있었다<그림 3>. <그림 7>에서는 스트림의 서비스 모드가 Merge/Split을 이용하여 가변적으로 변환되는 경우, 총 버퍼의 크기가 매우 효과적으로 감소함을 볼 수 있다. 또 한가지 관찰할 수 있는 사실은, 가변적으로 서비스 모드를 변환함으로써, 단일 디스크에서 지원할 수 있는 스트림의 개수가 디스크 용량을 능가할 수 있다는 사실이다.

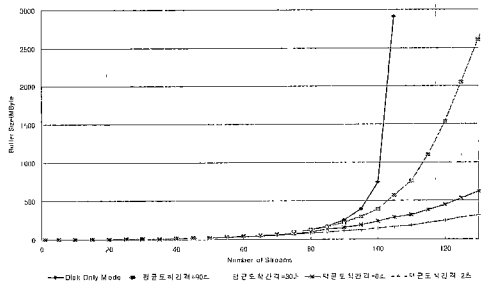

그림 6 스트림의 개수와 버퍼 요구량(요구 편향도 =0.271)

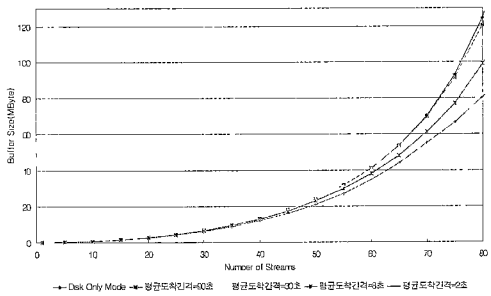

그림 7 스트림의 개수와 버퍼 요구량(요구 편향도 =0.271)

그림 6과 그림 7은 서로 다른 평균 도착간격(2초, 8초, 32초, 90초)하에서 요구 편향도 0.271로 서비스 요청이 도착하는 경우, 상영의 합병을 이용하였을 때 버퍼 요구량을 보여주고 있다. 그림 6은 스트림의 개수가 140개까지 증가할 때의 결과이고, 그림 7는 스트림의 개수가 80개까지 증가할 때의 결과이다. 두 개의 그래프를 분리한 이유는 디스크 모드만을 사용하는 경우에 버퍼의 요구량이 디스크의 용량에 근접함에 따라서 급격히 증가하기 때문에, 스트림의 개수가 비교적 적은 범위에서 버퍼 요구량의 변화형태를 자세히 볼 수 없기 때문이다.

우리는 두 가지의 주목할 만한 현상을 언급하고자 한

다. 첫 번째는 필요한 총 버퍼의 양이 주문의 평균도착 간격이 작을수록 감소한다는 사실이다. 도착 간격이 작을수록 디스크 모드를 메모리 모드로 변환하는 수익성이 증가하고, 서비스 모드 변환으로 인하여 절약할 수 있는 버퍼량이 증가하게 된다. 따라서, 본 연구에서 제안하는 *Merge/Split* 알고리즘은 서버의 사용량이 증가할수록 더 효율적으로 작동하는 것을 알 수 있다.

두 번째로 주목해야할 것은 디스크의 전송용량에 관한 문제이다. 디스크에서 모든 세션을 지원하는 경우 이론적으로 지원 가능한 세션의 개수는 앞에서 언급한 바와 같이 106개다. 이에 반하여, 가변적 서비스 모드 변환기법을 이용하면, 보다 적은 양의 버퍼를 이용하여 훨씬 많은 수의 세션을 지원할 수 있을 뿐 아니라, 지원할 수 있는 세션의 개수가 디스크의 전송용량을 능가할 수 있게된다.

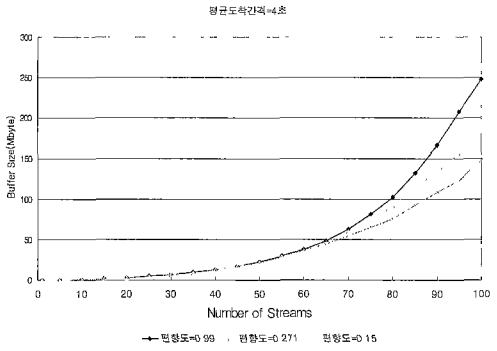

그림 8 요구 편향정도와 버퍼요구량

그림 8에서는 비디오들의 상영 빈도와 버퍼크기의 상관관계를 보여주고 있다. 평균 도착간격은 4초이다. 서로 다른 주문 편향도를 모델링하기 위하여, Zipf 함수에서 α의 값을 0.15, 0.271 그리고 0.99로 놓고 실험을 하였다. α=0.99는 비디오들의 상영빈도가 거의 균등한 (Uniform Distribution) 경우를 나타낸다. 그림 8에서는 사용자의 요청이 소수의 비디오 파일에 집중되어 있을수록, 이를 서비스하는데 필요한 총 버퍼의 크기가 줄어드는 것을 볼 수 있다. 그 이유는 본 논문에서 제안하고 있는 가변적 서비스 모드 변환기법이 사용자 요청의 편향적 성향을 효율적으로 반영하고 있음을 시사하고 있다.

8. 결 론

멀티미디어 서비스를 실행하는 데 필요한 과도한 시스템 자원(CPU 싸이클, 주기억 장치, 디스크 대역폭)의 요구는 서비스를 보다 경제적으로 최종 사용자에게 공급하는데 심각한 장애요인이 되어왔다. 본 연구에서는 사용자들의 서비스 요청이 상위에 랭크되어 있는 화일들에게 집중되어있는 현상을 이용하여 보다 적은 시스템 자원을 사용하여 비디오를 서비스하는 기법을 연구 개발하였다.

사용자들이 자주 요청하는 비디오 파일들은 인접한 세션간의 상영간격이 비교적 작으므로, 하나의 세션이 디스크에서 읽어들인 데이타 블럭들을 차후에 상영될 세션에 재 사용하기 위하여 메모리에 탑재해 두는 방법을 생각할 수 있다. 이 방법을 사용하면 디스크에서 자료를 읽어들이는 대신 메모리로부터 자료를 사용자에게 전송할 수 있으므로 디스크의 대역폭과 디스크로부터 자료를 읽어들이는 데 필요한 동기화 버퍼를 절약할 수 있다. 그러나, 언제 도착할지 모르는 상영요청을 위하여 사용된 멀티미디어 블록들을 주기억 장치에 저장하는 것 때문에 후행 상영 요청의 도착 시점에 따라 주기억 장치에 대한 부담이 상당히 클 수도 있게된다.

본 연구에서는 사용자의 서비스 요청을 실행하는 방법을 디스크 모드와 메모리 모드로 분류하고 각 방법의 장단점을 정교하게 분석할 수 있는 모델을 제시하였다. 이를 기반으로 새로운 세션이 시작하거나, 기존의 세션이 종료함에 따라 변화하는 가용 디스크 대역폭, 주기억 장치의 사용 현황등 서버의 상황에 따라 가변적으로 서비스 모드를 변환하는 알고리즘을 개발하였다. 본 연구 결과는 멀티미디어 서비스에 필요한 가장 중요한 시스템 자원 중의 하나인 주기억 장치 버퍼의 요구량을 현저히 감소시킴으로써, 보다 경제적으로 서비스를 제공하는 데 공헌하고 있다. 개발된 기법은 사용자의 주문이 상위 소수 화일에 집중되고, 그리고 서버가 많은 요구를 처리해야하는 상황에서 가장 효율적으로 작동함을 시뮬레이션을 통하여 검증하였다. 개발된 기법은 기존에 개발된 다양한 멀티미디어 디스크 스케쥴링 기법을 포용할 수 있다.

참 고 문 헌

[1] Antine Mourad. Issues in the design of a storage server for video-on-demand. *Multimedia Systems*, 1996(4):70-86, 1996.

[2] D. P. Anderson, Yoshitomo Osawa, and Ramesh Govindan. A File System for Continuous Media. *ACM Trans. Comput. Syst.*, 10(4):311-337, Nov 1992.

[3] David W. Brubeck and Lawrence A. Rowe. Hierarchical storage management in a distributed

VOD system. *IEEE Multimedia Magazine*, 3(3):37-47, Fall 1996.

[4] Thomas H. Cormen and Charles E. Leiserson adn Ronald L. Rivest. *Introduction to Algorithms*, chapter 2. MIT Press, 1 edition, 1990.

[5] Mon-Song Chen, Dilip D. Kandlur, and Philip S. Yu. Optimization of the grouped sweeping scheduling(gss) with heterogeneous multimedia streams. In *ACM Multimedia '93*, pages 235 - 242, 1993.

[6] Asit Dan and Dinkar Sitaram. Buffer Management Policy for an On-Demand Video Server. Technical Report IBM Research Report RC 19347, IBM Research Division, T.J. Watson Research Center, Yorktown Heights, NY 10598, 1993.

[7] Harvey M. 7, An Introduction to Operating Systems, Addison-Wesley Pub Co; ISBN: 0201180383

[8] J. Gemmell and S. Christodoulakis. Principles of Delay Sensitive Multi-media Data Storage and Retrieval. *ACM Trans. on Information System*, 10(1):51-90, January 1992.

[9] J. Gemmell. Multimedia Network File Servers: Multi-Channel Delay Sensitive Data Retrieval. In *Proc. of 1st ACM Multimedia Conf.* ACM, Oct. 1993.

[10] Ghandeharizadeh, Shahram and Kim, S. and Shahabi, C. Continuous Display of Video OBjects Using Multi-Zoned Disks. Technical report, University of Southern California, 1995.

[11] S. Ghandeharizadeh, S.H. Kim, and C. Shahabi. "on disk scheduling and data placement for video servers". In *Proceedings of ACM Multimedia Systems*, 1996.

[12] D.R. Kenchammana-Hosekote and J. Srivastava. Scheduling Continuous Media on a Video-On-Demand Server. In *Proc. of International Conference on Multi-media Computing and Systems*, Boston, MA, May 1994. IEEE.

[13] B Ozden, A. Biliris, R. Rastogi, and Avi Silberschatz. A Low-Cost Storage Server for Movie on Demand Databases. In *Proc. of VLDB '94*, 1994.

[14] Lougher P. and Shepherd D. The design of a storage server for continuous media. *The Computer Journal*, 36(1):32-42, 1993.

[15] Renu Tewari and Richard King and Dilip Kandlur and Daniel M. Dias. Placement of Multimedia Blocks on Zoned Disks. In *Proceedings of SPIE West '96*, 1996.

[16] P. Rangan, H. Vin, and S. Ramanathan. Designing an on-demand multimedia service. *IEEE Communication Magazine*, 30(7):56-65, July 1992.

[17] A. L. N. Reddy and J. Wyllie. Disk Scheduling in a Multimedia I/O system. In *Proc. ACM Multimedia Conf.*, pages 225--233. ACM Press, New York, 1992.

[18] Chris Ruemmler and John Wilkes. An Introduction to Disk Drive Modeling. *IEEE Computer*, 27(3):17-29, March 1994.

[19] P. Triantafillou and C. Faloutsos. "overlay striping and optimal parallel i/o in modern applications". *Parallel Computing*, (24):21-43, March 1998.

[20] Harrick M. Vin, Pawan Goayl, Alok Goyal, and Anshuman Goyal. A Statistical Admission Control Algorithm for Multimedia Servers. In *Proc. of ACM Multimedia Conf.*, pages 33-40, San Francisco, CA, Oct. 1994.

[21] Harrick M. Vin, Alok Goyal, Anshuman Goyal, and Pawan Goyal. An Observation-Based Admission Control Algorithm for Multimedia Servers. In *Proc. of 1th IEEE International Conf. on Multimedia Computing and Systems*, pages 234--243, Boston, MA, May 1994.

[22] Video Store Magazine, December 13, 1992.

[23] B. Worthington, G. Ganger, and Y. Patt. Scheduling Algorithms for Modern Disk Drives. In *Proc. of ACM SIGMetircs*, May 1994.

[24] Youjip Won, Issues in Distributed Hierarchical Storage Based Continuous Media Server, *Ph.D Dissertation, University of Minnesota*, Aug., 1997

원 유 집

1990년 서울대학교 자연과학대학 계산통계학과 학사. 1992년 서울대학교 자연과학대학 계산통계학과 전산학 석사. 1997년 University of Minnesota 전산학 박사. 1997년 ~ 1999년 Server Performance Analyst, Intel Corp. 1999년 3월 ~ 현재 한양대학교 공과대학 전자전기컴퓨터 공학부 교수. 관심분야는 멀티미디어 시스템, 멀티미디어 네트웍, 인터넷 프로토콜, 성능평가이론, 데이타베이스, 고성능 연결구조(High Performance Interconnect), 클러스터 컴퓨팅.