

파티션 디스크 VOD 서버를 위한 사용자 서비스 스케줄링 (User Service Scheduling for VOD Servers based on the Disk Partition policy)

최 성 욱*
(Seong-Wook Choi)

요 약

일괄 수용 서비스 스케줄링에서의 배칭(Batching) 기법은 서비스를 요청한 사용자들을 일정 시간 동안 그룹화 하여 한꺼번에 서비스하기 때문에 서비스 개시를 위한 지연시간이 발생한다. 그러나 이 지연시간을 효과적으로 제어하지 못하면 대기시간의 불규칙으로 서비스 공정성이 저하되고, 서비스 취소가 발생할 수 있다. 본 논문에서는 기존의 배칭(Batching) 기법인 선입 선출(FCFS) 및 빈도수 우선 방식(MQL)에서 문제시되던 평균 대기시간의 증가, 비 인기 비디오의 서비스 지연 문제를 해결하기 위한 리그 배칭 정책을 제안한다. 본 스케줄링 정책의 개념은 사용자의 요구와 파티션 디스크의 서비스 형태를 감안한 리그를 기초로 한다. 이를 시뮬레이션 한 결과, 기존의 방식들에 비하여 서비스 평균 지연 시간을 약 18% 정도 단축된 향상된 결과를 보임을 입증하였다.

ABSTRACT

The waiting delays of users are inevitable in this policy since the services are not taken immediately upon requests but upon every scheduling points. An inefficient management of such delays makes an unfair service to users and increases the possibility of higher renegeing rates. This paper proposes an leagued batching scheduling scheme which improves the average waiting time of users requests and reduces the starvation problem of users requesting less popular movies. The proposed scheme given intervals based on the league which reflect the disk partitions service patterns and user requests. Experimental results of simulations show that the proposed scheme improves about 18 percentage of average waiting time comparing with those of conventional methods such as FCFS and MQL.

1. 서론

VOD 서비스 환경에서 중앙 서버는 수시로 사용자의 비디오 상영 요청을 받아들이는데, 이러한 사용자의 요구는 비디오 특성상 응답 주기에 민감하고 연속적인 스트림의 전달이 필요하다.

그러므로 서버가 사용자에 대한 초기 서비스의 개시를 실시간으로 스케줄링 하는 실시간 수용 서비스(Real-time admission service)의 경우에는 서버가 동시에 수용할 수 있는 사용자의 수는 서버의 자원, 즉 사용 가능한 버퍼의 크기나 디스크 대역폭에 따

* 정희원 : 시립 인천 전문대학 전산과 교수

논문접수 : 2002. 7. 8.

심사완료 : 2002. 7. 29.

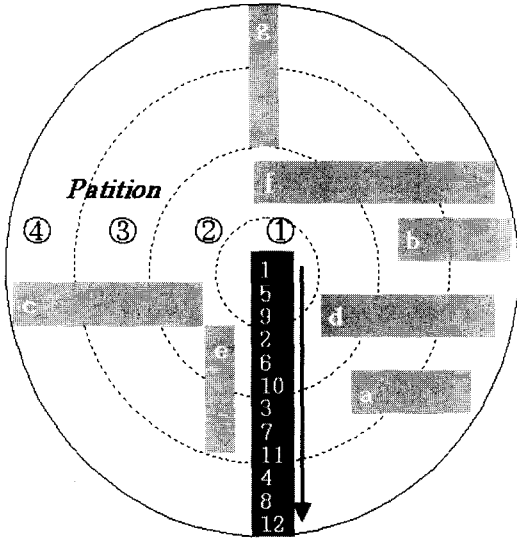
※ 본 연구는 시립 인천전문대학의 2001년도 연구비 지원에 의한 것임.

라 제한적일 수밖에 없다. 일괄 수용 서비스(Batched admission service)는 서버 자원의 한계를 극복하고 사용자 서비스의 수용능력을 증대하기 위하여 연구되었다. 이 방식은 미리 정의된 시간 간격 동안에 모아진 사용자의 동일한 비디오의 요구를 한번의 I/O 수행으로 처리하게 되므로 그에 해당하는 만큼 디스크의 부하를 감소시킬 수 있다. 일괄 수용 서비스 정책 중에서 대표적인 것으로는 배칭(Batching)[1,3,4] 스케줄링이 있는데, 이는 동일한 비디오를 요구하는 사용자의 서비스 개시 시간을 일정한 시간 간격(Time interval)동안 모아서 함께 처리하는 방식이다. 대표적인 배칭 스케줄링에는 선입 우선서비스(FCFS) 방식과 서비스 요구 빈도수 우선(MQL) 방식이 있다. 일괄 수용 서비스를 위한 배칭 스케줄링은 사용자의 입장에서 서비스를 요청하고 난 후 일정 시간의 지연을 감수해야 하는 불편함이 있다. 배칭 스케줄링에서 초기 서비스 지연을 방지하기 위하여 재생을 조정의 개념을 추가한 adaptive piggybacking 정책[2,7] 연구되었으나, 서비스 재생율의 변동에 의한 QOS의 저하가 발생할 수 있으며 영상 및 음성 정보를 재 동기화하기 위한 부가적인 정책이 필요하여 메커니즘 복잡해 질 수 있다[10,11]. 또한 대용량 디스크에서 복수 스트림의 연속적인 검색에 매우 효과적인 파티션 디스크 방식[5,9]은 서비스 중인 스트림의 검색에서는 매우 효율적이지만, 새로운 서비스를 위한 스트림의 초기 검색에서는 서비스 지연시간이 발생할 수가 있기 때문에, 일반적인 배칭 스케줄링(FCFS, MQL)로는 효율적인 서비스를 할 수 없다. 본 논문에서는 파티션 디스크를 사용하는 VOD 시스템에서, 새로운 서비스를 개시할 때 스트림의 초기 검색을 효율적으로 실행기 위하여 파티션 디스크 내에 저장된 비디오 정보와 사용자의 서비스 요구를 활용한 리그 배칭 스케줄링(Leagued Batching Scheduling: LBS)을 제안한다. 2장에서는 본 논문에서 다룬, VOD 서비스 스케줄링에 관한 사례들을 살펴보고, 3장에서는 본 논문에서 제안한 리그 배칭 정책에 대한 스케줄링 개념에 대하여 논의한다. 4장에서는 본 논문에서 제안한 스케줄링 정책을 시뮬레이션하고, 기존의 방식과 비교 분석한다. 끝으로 5장에서는 결론과 함께 향후 연구 방향에 대하여 간략히 기술한다.

2. 관련연구

2.1 디스크 파티션

파티션 디스크(Disk Partition)[5,9] 기법은 VOD 서비스 환경에서 효율적으로 비디오를 서비스 하기 위한 스트림의 저장 및 스캔 방식이다. 이 방식은 비디오들이 같은 주기 및 같은 추출 단위를 갖는 것으로 가정하고, 디스크를 몇 개로 분할하여 종방향 우선으로 SCAN하는 방식이며, 횡(Track)방향은 다수의 다른 스트림을 배치할 수 있기 때문에 동시에 여러개의 비디오 스트림을 SCAN하는데 유리하며, 상대적으로 적은 용량의 버퍼가 사용된다. 디스크 파티션 방식의 특징은 단일 디스크 표면에 n 개의 원형 부분들 ($1, 2, 3, \dots, n$)인 파티션으로 나누고 각 파티션은 여러 개의 트랙들로 구성되며, 각 트랙은 고정된 블록으로 나누어진다. 그림 1에서는 4개 파티션을 갖는 디스크 파티션 방식을 보여주고 있다. 각 스트림은 일련의 연속된 블록으로 표현되며, 스트림의 연속된 블록은 각각 다른 파티션에 기록되기 때문에 헤드가 특정 파티션 내의 스트림의 시작 지점을 만나면 동일 방향으로 (디스크 안쪽에서 바깥쪽으로 혹은 그 반대 방향으로) 디스크 표면을 읽어들인다. 이 방식은 동일한 주기를 갖는 비디오 스트림의 검색에 대해서는 실시간 디스크 스케줄링에 비하여 매우 우수한 검색 능력과 버퍼 활용률을 나타내었다.



[그림 1] 디스크 파티션 방식의 구조
[Fig. 1] Layout of disk partition

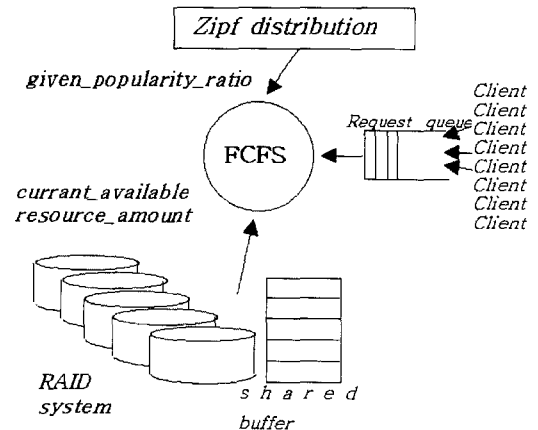
예를 들어 어느 비디오의 스트림 블록 1에서 12까지를 [그림 1]에서 표시된 것처럼 저장하였다면 스트림 블록1에서 4까지를 읽기 위해서 헤드는 디스크 안쪽에서부터 바깥쪽으로 움직이며 파티션 1, 2, 3, 4, 순으로 읽어간다. 만일 파티션 3에 위치한 스트림 블록 3을 읽을 차례에서 비디오 a, b, c, e, f, g의 파티션 3에 속해있는 어떤 스트림 블록에 대한 읽기 요청이 있다면, 디스크의 회전에 의하여 짧은 시간 동안 한꺼번에 읽어 들일 수 있다. 또한 어떤 비디오 스트림을 읽는 중에 다른 비디오의 읽기 요청이 발생하고 그 비디오의 시작 블록이 다른 파티션에 위치하고 있다면, 일단 현재 서비스 중에 있는 스트림 블록을 계속 읽어 가면서 헤드를 진행시키다가 요청된 비디오 스트림의 시작 블록이 기억되어 있는 파티션에 도착되면, 그 지점에서 새로이 요청된 스트림까지 함께 읽는 방식이다.

2.2 배칭 스케줄링

배칭 스케줄링은 배칭 간격을 주기로 하여 서비스 요청 큐에 도착된 사용자 서비스 정보를 토대로 어떤 사용자를 서비스 개시할 것인가를 결정한다. 이러한 배칭 스케줄링 방식으로는 선입 선출 방식 (First Come First Service; FCFS)과 다중 큐를 활용한 빈도수 우선 방식(Multi Queue List; MQL)이 있다[3,4].

(1) FCFS 스케줄링

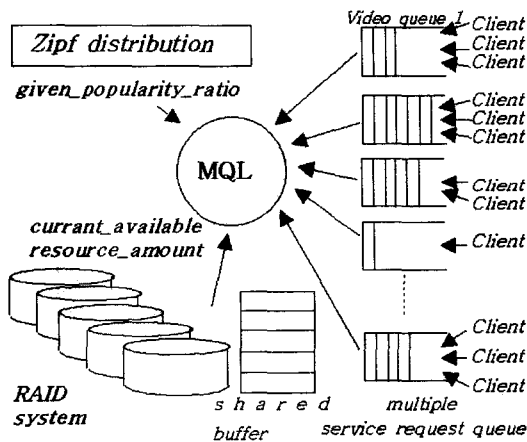
FCFS은 일괄 수용 정책을 기본으로 하면서, 먼저 들어온 요청을 먼저 서비스해 준다는 전략이며, 이를 위하여 서버는 오직 하나의 서비스 요청 큐를 가진다. 그래서 배칭 시간 간격이 경과한 후 서비스 요청 큐 안에서 먼저 도착된 요청부터 우선 서비스를 해주는데, 이때 이와 동일한 비디오표를 요청한 사용자가 서비스 요청 큐 내에 있을 경우에 이들도 함께 서비스해준다. FCFS전략은 비디오표의 인기도에 관계없이 골고루 서비스해줄 수 있다는 장점이 있지만, 서비스 요청 빈도가 높은 비디오와 서비스 요청 빈도 낮은 비디오를 동일한 우선순위로 서비스해주기 때문에 평균대기 시간이 증가한다.[그림 2]



[그림 2] FCFS 스케줄링 모델
[Fig. 2] FCFS scheduling model

(2) MQL 스케줄링

MQL은 FCFS에서 나타나는 평균 대기 시간의 증가를 개선하기 위해서 고안된 전략이며 서버가 서비스 가능한 영화의 개수만큼 서비스 요청 큐를 가지고 있으며, 각각의 비디오에 대한 사용자의 서비스 요청은 자신의 고유한 큐에 저장된다. MQL은 배칭 시간 간격이 경과한 후 각각의 서비스 요청 큐를 조사한 후, 서비스 요청이 제일 많은 큐를 우선적으로, 그 큐에 있는 사용자 모두를 동시에 서비스한다. MQL 전략은 FCFS와는 다르게 서비스 요청의 빈도가 높을수록 서비스를 자주 해주게 되어 FCFS에 비하여 상대적으로 평균대기 시간을 감소할 수 있지만, 최악의 경우에 요청 빈도가 낮은 비디오는 서비스 대기시간이 너무 길어져 버리는 문제가 발생할 수 있다. [그림 3]



[그림 3] MQL 스케줄링 모델
[Fig. 3] MQL scheduling model

3. 리그 배칭 스케줄링

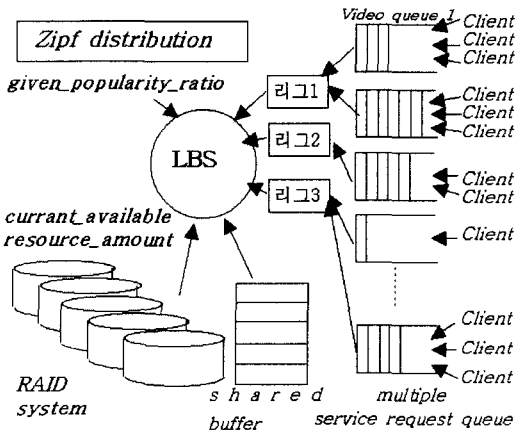
3.1 개요

리그 배칭[그림 4]에서는 배칭 스케줄링시의 초기 서비스의 지연 시간을 감소시키고, 디스크 스케줄링과의 일관성을 유지하기 위해서 파티션 디스크 저장 방식의 스캔 정책을 감안한 스케줄링을 실행한다. 일반적으로 사용자 스케줄링 시점은 디스크 스케줄링 시점과 상관없이 정해진 배칭 간격을 기준으로 미리 스케줄링 되어 진다. 그러므로 서비스 요구된 스트림을 실제로 디스크 스케줄링에 의하여 읽어 오기까지 시간적인 지연이 발생할 수 있으며, 실제로 그 지연 시간 동안, 해당 비디오 스트림에 대한 새로운 사용자의 요구가 발생하거나, 사용자의 요구 패턴이 변화 될 수 있다. 본 절에서 제안한 스케줄링 방식은, 사용자 스케줄링을 실행하는 시점을 해당 비디오 스트림이 속해있는 파티션에 도착하기 직전에 실행하므로, 최신의 사용자 요구 패턴을 스케줄링에 감안할 수 있다. 이를 위하여 해당 파티션마다, 초기 비디오 스트림이 저장되어 있는 비디오들이 함께 스케줄링 되는데, 이러한 비디오의 그룹을 리그(League)라 정하였다. 그러므로 리그란 한번의 사용자 스케줄링에 참여하는 비디오의 종류가 된다. 물론 사용자 스케줄링의 시점에서 디스크 스캔 정책의 실현까지 발생하는 초기 지연은, 파티션 디스크 뿐만 아니라, 일반적인 스캔 디스크 방식인P-Scan, FD-Scan, Scan-EDF, GSS 등의 디스크 스케줄링 방식에서도 마찬가지로 발생하므로, 리그 배칭 스케줄링은 단지 파티션 디스크 저장 방식에서만 해당되는 스케줄링 기법은 아니며, 대부분의 스캔 디스크 스케줄링 방식과 일괄 수용 사용자 스케줄링에 적용할 수 있다. 리그 배칭에서는 사용자 스케줄링에 참여하는 전체 비디오를 한 개 이상, 몇 개의 리그로 나누고, 한 리그씩 라운드 로빈(Round robin) 방식으로 배칭 스케줄링을 실행한다. 만약 전체 리그가 한 개일 경우에는 일반적인 FCFS 나 MQL 스케줄링을 실행 할 수 있으며, 리그의 수에 따라 전체적인 배칭 간격을 효과 적으로 조정할 수 있다. 이 리그는 파티션 디스크에서 1개 이상의 특정 파티션들과 동기화 되며, 리그의 수와 파티션의 개수는 서로 관계

가 없으나, 총 리그의 수는 파티션의 개수를 초과할 수는 없다. 본 논문에서 제안하는 리그 배치는 그림 5.5에서 보는바와 같이 서비스 요청 큐를 서비스 요청되는 비디오의 종류만큼 복수 개로 두고 이들을 몇 개의 리그(League)로 편성하여, 배칭 스케줄링 시에 한 리그씩 라운드 로빈(Round robin) 방식으로 스케줄링 한다. 동일한 리그에서 서비스 스케줄링은 서비스 요청 수가 많은 비디오가 선택된다. 리그의 크기는 식(2)과 같이 정의 할 수 있다.

$$L_j = \sum_{i=K}^N \Pi(V) \quad (2)$$

- L : j 번째 리그
- $P_i(V)$: i 번째 파티션에서 시작하는 비디오번호
- K : 시작 파티션 번호
- N : 마지막 파티션 번호



[그림 4] LBS 모델
[Fig. 4] LBS model

3.2 디스크 검색 시간 분석

일반적으로 복수개의 비디오 스트림을 한꺼번에 서비스할 경우, 최대 탐색시간은 FCFS, 마감시간 우선 주사(Scan-EDF)방식 및 집단 일소(GSS)방식의 경우 식(3)와 같이 임의 블록주사 (Random black scan)방식과 동일한 탐색 시간을 나타내며, 디스크 파티션 방식은 식(4)로 표현된다[5].

$$T_{sch} = k(b_i/rt + rx + se) + 2se \quad (3)$$

- k : 함께 읽는 스트림 수
- b_i : 블록 크기
- rt : 최대 전송율
- rx : 디스크 최대 회전 속도

$$T_{sch} = k(b_i/rt + rx + se) + se/p \quad (4)$$

- p : 파티션 개수

디스크 조사시간(se)은 식(5)(6)(7)과 같다.

$$\text{장거리 탐색일 경우 } se = a + bc \quad (5)$$

$$\text{단거리 탐색일 경우 } se = a + b\sqrt{c} \quad (6)$$

$$\text{최단거리 탐색일 경우 } se \approx 2ms \quad (7)$$

- a : 헤드 안정 계수
- b : 가속 안정 계수

마감 시간 우선 방식 및 집단 일소 방식은 단거리 탐색이나 장거리 탐색이고, 일반적인 디스크 파티션 방식은 최단거리 탐색이거나 단거리 탐색이며, 만일 새로 요청된 비디오 스트림이 현재 헤드의 위치와 동일 트랙 상에 존재할 경우는 조사시간이 거의 없다. 그러므로 디스크 파티션 방식은 다른 방식에 비하여 디스크 검색 시간이 짧고, 실시간 읽기 작업을 수행하기 때문에 미리 읽기를 위한 중간 버퍼가 필요 없으며, 일단 검색이 시작되면 동시에 여러 개의 비디오를 읽어 들일 수 있어 VOD 서비스에 매우 유리한 방식이 될 수 있다.

3.3 파티션 디스크에서 스케줄링 별 서비스 대기 시간 분석

[그림 5]에서는 파티션 디스크 방식과 리그 배칭 스케줄링을 연계한 예를 나타내었다. 먼저 그림 5에서는 디스크 표면을 논리적인 4개의 파티션으로 분할한 디스크 시스템에 10개 비디오의 시작 스트림을 저장한 상태를 나타내고 있다. 각각의 비디오에 대한 시작 스트림 블록의 위치는 파티션1에서 비디오 7, 비디오 4가 시작되고 파티션2에서는 비디오 8, 비디오 2, 파티션 3에서는 비디오 1, 비디오 3, 파티션 4에서는 비디오 5, 비디오 6, 비디오 9가 각각 저장되어 있다. 모든 비디오의 시작 스트림을 하나의 특정한 파티션에 함께 저장하거나 파티션 전체를 하나의 리그로 간주한 경우는 FCFS나 MQL 방식과 동일한 배칭 스케줄링을 실행할 수 있다.

현재 헤드가 서버가 비디오 7의 1번째 스트림 블록 읽기를 완료하고 계속 다음 스트림 블록을 읽기 위하여 헤드를 파티션 2로 이동하려 할 경우, 먼저 다음 파티션에 속해 있는 비디오 8과 비디오 4에 대한 서비스 요청이 도착되어 있는가를 확인하고, 만일 비디오 8이 서비스 요구 되었다면 비디오 8은 함께 읽을 수 있도록 스케줄링 된다. 또한 파티션 3의 경우 비디오 1, 비디오 3, 비디오 10의 시작 스트림 블록이 위치하여 있고 세 개의 비디오가 모두 서버

스가 요청되었을 경우에는 리그내에 서비스 요청이 많은 비디오가 스케줄링 된다.

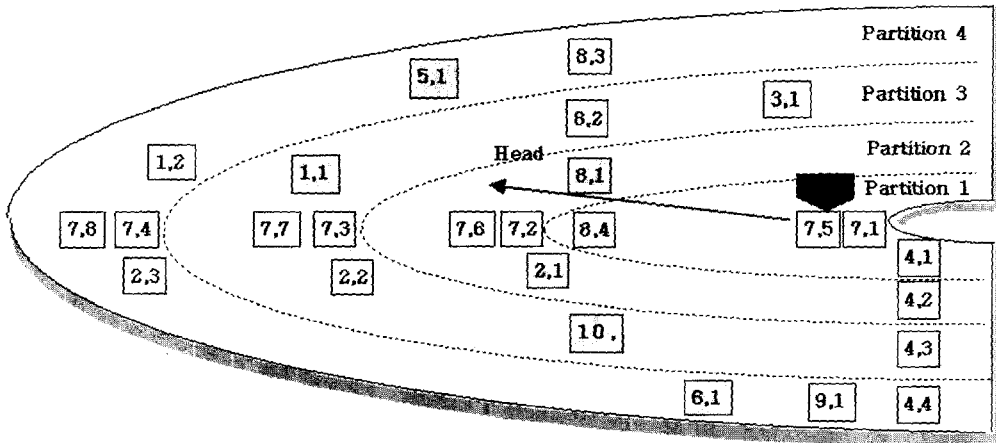
파티션 디스크에서 사용자 스케줄링 방식을 FCFS 스케줄링이나 MQL 스케줄링으로 할 경우, 총 지연 시간은 배칭 대기시간 및 현재의 파티션에서 새로 스케줄링 된 비디오의 시작 스트림 블록이 위치한 파티션까지 헤드가 순차적으로 움직일 시간의 합으로 나타난다.

식(4)인 $Tsch = k(b_i/rt + rx + se) + se/p$ 은 해당 파티션 내의 디스크 검색 시간이므로, 현재의 파티션과 새로 요청된 비디오가 저장된 파티션사이의 거리(파티션 수)를 d 라 하면 서비스 대기시간은 (8)으로 표시할 수 있다.

$$W_i = Bin_v + dk(b_{i+1} + rx + se) + se/p \quad (8)$$

서비스 대기시간 : W_i
 배칭 대기 시간 : Bin_v

d 는 현재의 파티션 p_i 로부터 새로 스케줄링 되어 서비스 개시할 스트림 블록의 파티션 p_j 와의 거리(파티션의 수)이며, 이때 총 파티션의 수를 n 이라하고 하면, d 는 $p_j - p_i$ ($p_j \geq p_i$), $p_j + n - p_i$ ($p_j < p_i$)가 된다.



[그림 5] 파티션 디스크 방식에서 비디오 스트림 저장
 [Fig. 5] Stored Video streams with disk partition

본 논문에서 제안한 리그 배칭에서의 서비스 대기 시간은 식(9)으로 표시할 수 있다.

$$W1 = Binv + k(bitt + rx + se) + se \tag{9}$$

리그 배칭은 사용자 스케줄링시에 다음 파티션에 속해 있는 리그를 중심으로 스케줄링하므로 원칙적으로 *d*가 발생하지 않는다.

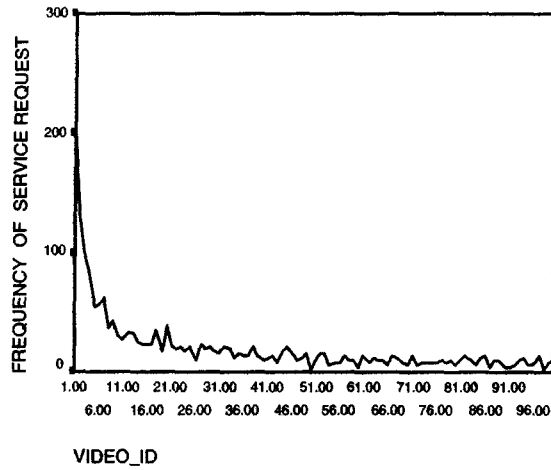
4. 시뮬레이션 및 분석

본 논문에서 제안한 효율적 서비스를 위한 리그 배칭 정책의 주요 관심은 주어진 서버의 능력 하에서 서비스 평균대기 시간의 단축에 있으며, 이를 시뮬레이션 하기 위하여 사용자가 요청한 비디오를 서비스 받기까지 평균대기 시간 및 최대 서비스 대기 시간을 대표적인 배칭 기법인 FCFS, MQL 특성을 비교하였다. 시뮬레이션을 위한 파라미터의 설정은 디스크 시스템 HP9756016을 기준 하였으며 동시에 서비스해야 할 비디오의 수는 최대 100개로 하였고, 파티션 개수는 16개로 하였다. 블록길이는 1028Byte로 정하였다. 그 외의 자세한 특징은 <표 1>에 나타내었다. [그림 6]에서 시뮬레이션에 사용할 자료의 비디오 요구패턴은 Zif Distribution 을 기본으로 하였고, 서비스 도착율은 λ는 1sec당 한 개의 평균 도착률을 갖는 Poisson Distribution으로 하였다. [그림 6]의 X축은 시뮬레이션에 사용된 100개의 비디오 번호이며 Y 축은 비디오에 대한 요구 빈도이다. 마감 시간 우선 주사 방식(SCAN-EDF)과 파티션 디스크의 검색시간의 차이를 [그림 7]에 나타내었다. 우선 그래프의 X축은 검색하는 비디오의 수이며, Y 축은 검색시간을 나타내었다. 결과에서 보는바 와 같이 스트림 수의 증가에 따른 디스크 검색 시간은 파티션 디스크 방식이 마감 시간 우선 주사 방식(Scan-EDF)에 비하여 매우 유리한 결과를 보이고 있다. <표 2>에서는 각 스케줄링 별 최대 대기시간, 평균 대기시간, 최소 대기시간을 정리하여 보았다. 시뮬레이션 데이터는 2000개로 하였으며, 그래프에서 표시된 평균대기 시간의 단위는 100(ms)이다. 평

균대기 시간은 FCFS가 9550, MQL이 9201, LBS가 7753으로 LBS가 FCFS나 MQL에 비하여 평균대기 시간이 약 18% 정도 단축됨을 보이고 있다. 평균대기 시간의 차이를 알기 쉽게 표현한 것이 [그림 7]에 있다. 또한 스케줄링이 경과함에 따라 평균대기 시간의 변화 패턴을 선형 그래프로 나타낸 결과가 그림9에 있다. [그림 9]에서 X축의 state는 배칭 단계를 순차적으로 표시한 것이며, 그 단위는 1이다. Y축의 값은 스케줄링 별 배칭 간격 동안에 서비스 받지 못하고 대기 중에 있는 사용자의 총 대기 시간을 배칭 단계 내에서 평균한 수치이다. 각 스케줄링의 사용자 평균대기 시간은 시간이 지남에 따라 늘어나는데, 비디오 서비스 신청이 1초 간격으로 계속 누적되기 때문이다. 평균대기 시간은 FCFS, MQL에 비하여 LBT가 좋은 결과를 보여 주고 있으며, 그래프의 특성도 특정 대역에서 큰 변화가 없이 다른 스케줄링과 동일한 특성을 보여준다.

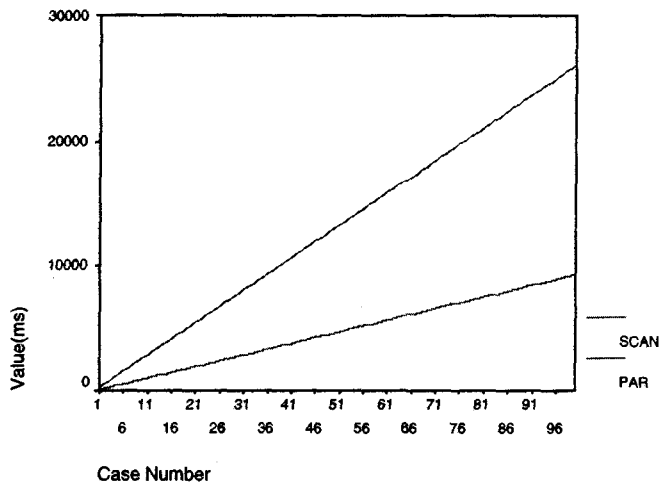
<표 1> 디스크 파라미터 테이블
<Table 1> disk parameter table

디스크	
파라미터	값
sector size	512 byte
cylinders	1,962
tracks per cylinder	19
data sector per tracks	72
number of zones	1
track skew	8sectors
revolution speed	4,002 rpm
controller reads	2.2ms
overhead writes	2.2ms
seek time sort(ms)	$3.24 + 0.4 \sqrt{C}$
long(ms)	8.0 +0.88c
boundary	c=383
maximum transfer rate	10m bps



[그림 6] 비디오 요구 패턴

[Fig. 6] Request video pattern



[그림 7] 디스크 검색 시간 비교

[Fig. 7] A comparative disk search time

<표 2> 스케줄링 별 대기 시간 비교표

<Table 2> A comparative table of waiting time by scheduling polices
Statistics

		FCFS	MQL	LBS
N	Valid	2000	2000	2000
	Missing	0	0	0
Mean		9550.7231	9201.2231	7753.3446
Minimum		66.80	50.19	53.44
Maximum		18923.04	25657.09	21766.69

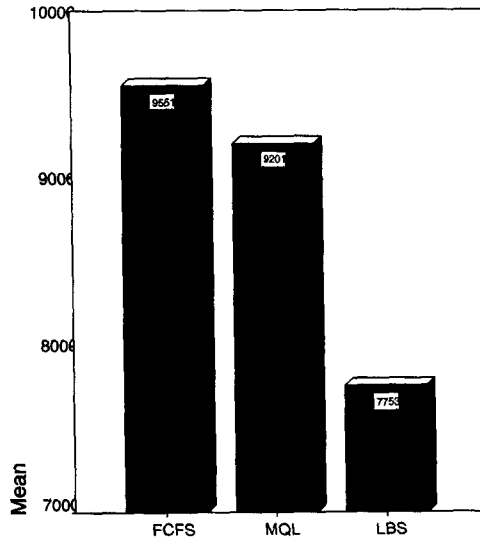


그림 8] 스케줄링별 평균 대기 시간

[Fig. 8] Scheduling policies and mean of waiting time

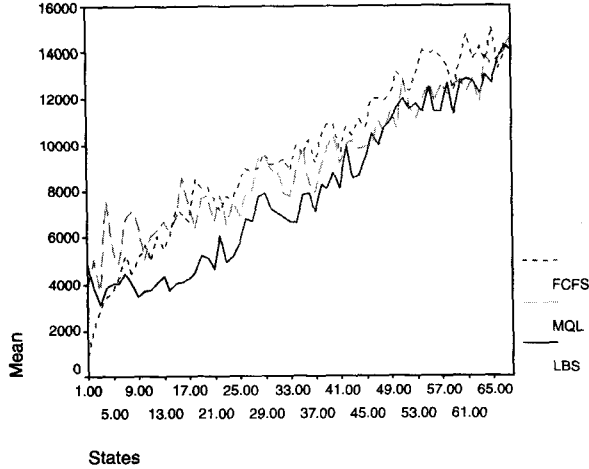


그림 9] 배치 단계별 평균 대기 시간

[Fig. 9] Batching sequence and mean of waiting time

5. 결론

VOD 서비스 환경에서 복수개의 비디오 스트림을 효율적으로 검색할 수 있는 파티션 디스크는 현재 서비스 중인 스트림 검색에는 유리하지만 새로 허락된 사용자의 서비스 개시에는 상대적으로 지연시간

이 길어질 수 있다는 단점이 있다. 본 논문에서는 파티션 디스크의 장점을 그대로 살리면서 새로운 사용자에게 대한 서비스 개시를 단축할 수 있는 리그 배치 스케줄링을 제안 하였다. 리그 배치 스케줄링은 디스크 스케줄링을 감안한 사용자 서비스 스케줄링을 행하는 것으로, 파티션 디스크의 파티션과 유사한 개념의 리그를 사용한다. 이 리그는 서비스되는

비디오를 몇 개의 그룹으로 묶는 것으로 이러한 그룹들은 파티션내 시작 비디오들과 논리적으로 동기화 된다. 이 스케줄링은 기존 방식에 비하여 18% 정도의 평균 대기 시간이 감소되는 성과가 있었다. 또한 리그 배칭은 스케줄링은 파티션 디스크를 사용함으로써 다수의 사용자와 다중 미디어 스트림 등 다양한 조건에서 서비스해야하는 VOD서버일수록 그 효과가 클 수 있다고 하겠다. 향후 연구 방향은 먼저 사용자 스케줄링과 디스크 스케줄링을 통합할 수 있는 VOD 모델을 확정하는 것이며, 이어 이 논문에서 상세히 취급되지 않은, 버퍼 계층내의 효율적인 스케줄링에 관한 방안을 마련한다. 마지막 단계는 이들 세 개의 스케줄링 계층을 다시 통합화하여, 모든 계층의 스케줄링과의 연계성을 충실히 할 필요가 있다.

※ 참고문헌

[1] Heek-Young Woo, Chong-Kwon Kim, "Multicast Scheduling for VOD Services". *Multimedia Tools and Applications*, pp. 157-171, 1996

[2] Leana Golubchik, John C.S. Lui, "Adaptive piggybacking: a novel technique for data sharing in video-on-demand storage servers". *Multimedia Systems*, pp 140-15, 1996

[3] Asit Dan, Dinker Sitaram, "Dynamic batching policies for an on-demand video server". *Multimedia Systems*, pp 112-121, 1996

[4] Asit Dan, Dinkar Sitaram and Perwez Shahabuddin, "Scheduling Policies for an On-Demand Video Server with Batching". *ACM Multimedia*, pp 15-23, 1994

[5] P. Bocheck, H. Meadows, S. Chang, "Disk Partition Technique for Reducing Multimedia Access Delay", *Preceedings of the IASTED/ISMM International Conference*, pp.27-30, 1994.

[6] C. Ruemmer and J. Wilkes, "An Introduction to Disk Drive Modeling," *IEEE Computer*, Vol.27. No. 3. pp.17-28, March 1994.

[7] L.Goubchik, J.C.S. Lui. and R. Muntz, "Reducing I/O demand in video-on-demand storage servers", in *proc. of Intl. Conf. on Measurement and Modeling of Comp. Sy st. (SIGMETRICS'95)*, pp 25-36, 1995

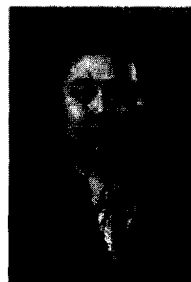
[8] Ozden B, Biliris A, Rastogi R, Silberschatz A, "A Low-cost storage server for movie-on-demand data bases". *Proceedings of the 20th international Conference on Vary Large Databases*, pp 594-605, 1994

[9] Seung-Kyu Park, Sung-Wook Choi. "Storing Technique of Multiple streams on Disk with a fixed size Buffer for Playout", *ITC-CSCC '96*, PP.649-652, Korea, 1996.

[10] Kurt Rothermel, Tobias Helbig, "An adaptive protocol for synchronizing media streams", *Multimedia Systems*, pp 324-336, 1997

[11] 박승철, 최양희, "QOS를 고려한 적응형 멀티미디어 동기화 기법" *정보과학회 논문지 (A) 제 22권 제9호*, PP 1307-1318, 1995

최 성 욱



1983년 광운 대학교
전자계산과(이학사)
1987년 경희대학교 대학원
전자공학과(공학 석사)
2001년 아주대학교 대학원
컴퓨터공학과(공학 박사)
1992년~현재
시립 인천 전문대학 전산과
교수
관심분야 :
멀티미디어 시스템 및 응용,
소프트웨어 시스템 디자인,
이동 컴퓨팅 시스템 등