동적 버퍼 분할을 이용한 클러스터 VOD 서버의 효율적 부하 분산 방법

권 춘 자[†] · 김 영 진[††] · 최 황 규[†††]

요 약

클러스터 기반 VOD 서버는 동시에 여러 사용자에게 실시간으로 서비스하기 위해 정교한 부하 분산 기술과 버퍼 관리 기술을 요구한다. 본 논문에서는 클러스터 기반 VOD 서버에서의 동적 버퍼 분할을 이용한 새로운 부하 분산 방법을 제안한다. 제안된 방법은 사용자 요구를 처리하는 VOD 서비스 노드간의 버퍼 성능과 디스크 접근 빈도를 고려하여 전체 부하를 고르게 분산하도록 한다. 또한 동적 버퍼 분할 방법은 동일한 연속미디어에 접근하려는 여러 사용자에게 평균 대기시간을 감소시킬 수 있도록 버퍼를 동적으로 분할한다. 시뮬레이션을 통해 제안된 방법이 기존의 방법보다 부하량을 적절히 조절하면서 평균 대기시간을 감소시키고, 각 노드의 처리량도 증가시킨다. 특히, VOD 서버 클러스터가 과부하 상태일 때 제안된 방법이 Generalized Interval Caching에 기반한 부하 분산 방법 보다 약 2배의 우수한 성능을 나타냄을 시뮬레이션을 통하여 입증한다.

An Efficient Load Balancing Technique in Cluster Based VOD Servers using the Dynamic Buffer Partitioning

Chun Ja Kwon[†] · Young Jin Kim[††] · Hwang Kyu Choi[†††]

ABSTRACT

Cluster based VOD systems require elaborate load balancing and buffer management techniques in order to ensure real-time display for multiuser concurrently. In this paper, we propose a new load balancing technique based on the dynamic buffer partitioning in cluster based VOD servers. The proposed technique evenly distribute the user requests into each service node according to its available buffer capacity and disk access rate. In each node, the dynamic buffer partitioning technique dynamically partitions the buffer to minimize the average waiting time for the requests that access the same continuous media. The simulation results show that our proposed technique decreases the average waiting time by evenly distributing the user requests compared with the exiting techniques and then increases the throughput in each node. Particularly under the overloaded condition in the cluster server, the simulation probes that the performance of the proposed technique is better two times than the Generalized Interval Caching based technique.

키워드 : 멀티미디어(multimedia), VOD 서버(VOD server), 클러스터(cluster), 동적 버퍼 분할(dynamic buffer partitioning), 부하 분산(load balancing)

1. 서 론

최근 비디오, 오디오, 정지 화상 등 다양한 데이터로 이루어진 멀티미디어 정보의 인터넷을 이용한 실시간 멀티미디어 서비스는 저장 장치, 통신, 압축 기술의 발달로 크게 발전하였으며, 이에 따라 멀티미디어 정보에 대한 서비스 요구도 크게 증가하고 있다. 특히 여러 종류의 멀티미디어 정보 분야중 VOD(Video On Demand) 서비스는 기존의 TV나 유선 방송을 대체할 수 있는 획기적인 서비스 분야로 이에 대한 많은 연구가 이루어지고 있으며 여러 상용 제품들이 출현하여 널리 활용되고 있다[1]. VOD 서비스는 원하는 시간에 실시간으로 인터넷이나 전용 통신망을 통하여 검색하여 볼 수 있어야 하며, 멀티미디어 정보 특성상 용량이 매우 크므로 서로 다른 스트림들 사이의 동기화를 필요로 한다[2]. 따라서 멀티미디어 정보가 저장된 VOD 서버는 대용량 저장장치여야 하고, 사용자가 요구하는 멀티미디어 서비스를 연속으로 병행 처리하기 위한 실시간 처리 능력 구조여야 한다. 그러나 싱글 대용량 VOD 서버는 병행 처리 서비스에 한계가 있으며, 저장 장치의 대역폭 또한 멀티미디어 정보 서비스에 부적합하다. 이

※ 본 논문은 강원대학교 BK21 사업단 지원에 의한 연구 결과의 일부임.
† 준 회 원 : 강원대학교 대학원 컴퓨터정보통신공학과
†† 정 회 원 : (주)이비즈온 연구소
††† 정 회 원 : 강원대학교 전기전자정보통신공학부 교수
논문접수 : 2002년 7월 29일, 심사완료 : 2002년 9월 2일

를 위해 고가용성 및 확장성을 지원하는 클러스터 VOD 서버 구조가 제안되었다[3, 4].

클러스터 VOD 서버는 비교적 저가의 고성능 PC 또는 워크스테이션들을 고속의 네트워크로 연결하는 구조로, 기존의 강 결합 단일 서버에 비하여 확장성과 신뢰성이 높고 가격 대 성능 면에서 우수함을 나타낸다[5]. 또한 클러스터 VOD 서버에 관한 많은 연구가 이루어지고 있는데, 클러스터 VOD 서버의 대역폭을 향상시키기 위하여 스트라이핑(striping) 방법 및 복제(replication) 방법을 많이 쓰고 있다. 이 방법은 스트라이핑 및 복제를 통해 클러스터내의 각 VOD 서비스 노드에 데이터를 분산하여 재구성하는 데 초점이 맞춰졌다[6, 7]. 따라서 전체 클러스터 VOD 서버의 성능을 향상시키며 각 VOD 서버 노드의 부하를 고르게 분산시키기 위한 방법이 필요하다. 그러나 클러스터 내의 각 VOD 서버 노드에 존재하는 버퍼를 통합적으로 관리하면서 노드의 부하를 적절히 분산하는 방법에 대한 연구가 이루어지지 않고 있다.

본 논문에서는 클러스터 VOD 서버 환경에서 사용자 요구를 처리하는 서비스 노드간의 버퍼 성능과 디스크 접근 빈도를 고려하여 전체 부하를 고르게 분산할 수 있는 부하 분산 방법을 제안한다. 각 서비스 노드의 디스크 접근 빈도 수를 부하량으로 사용하여 새로운 멀티미디어 스트림 정보를 검색하고자 한다면 가장 적은 부하량을 가진 노드로 사용자의 요구를 분배한다. 즉, 각 서비스 노드의 버퍼 정보와 부하량에 의해 부하가 고르게 분산되도록 한다. 또한 멀티미디어 정보의 특성을 고려하여 각 노드의 버퍼 활용율을 크게 높일 수 있는 새로운 버퍼 관리 방법인 동적 버퍼 분할 방법을 적용한다[15, 16]. 디스크로부터 한번 읽은 데이터를 버퍼링하여 재사용하면 저장장치의 접근 부하를 줄일 수 있으므로 끊김 현상 없는 멀티미디어 정보 서비스를 위해 버퍼 정보를 활용한다. 기존의 버퍼 관리 방법은 버퍼가 필요할 때마다 교체하는 방법인 BASIC / DISTANCE, Interval Caching등을 이용하였으며[8], 예측된 데이터를 미리 버퍼에 할당하는 방법인 정적 버퍼 분할 방법, 적응 버퍼 분할 방법 등을 이용하였다. 그러나 기존의 방법은 버퍼를 최대한 활용하지 못하여 버퍼 낭비를 초래하므로 버퍼 활용도가 낮았다. 따라서 본 논문에서는 사용 인터벌이 긴 부분의 버퍼를 즉시 반환하여 사용함으로써 버퍼 활용율을 높이는 동적 버퍼 분할 방법을 사용한다. 제안된 방법을 이용함으로써 버퍼의 활용도를 높일 수 있으며 병행 사용자들의 평균 대기 시간을 크게 줄일 수 있다. 결과적으로 제안된 부하 분산 방법과 버퍼 관리 방법을 통합함으로써 전체 클러스터 VOD 서버의 성능을 크게 증가시킬 수 있다.

본 논문은 먼저 2장에서 클러스터 및 멀티미디어 시스템에 대한 관련 연구를 기술하고, 3장에서 기존의 클러스터 VOD 서버에 대한 연구를 기술하며, 4장에서 기존의 버퍼 관리 방법에 대하여 기술한다. 5장에서 제안된 부하 분산 방법과 동적 버퍼 분할 방법에 대하여 기술하며, 6장에서 병행 사용자 수를 중심으로 제안된 알고리즘의 성능 분석을 수행하며, 마지막으로 7장에서 결론을 맺는다.

2. 관련 연구

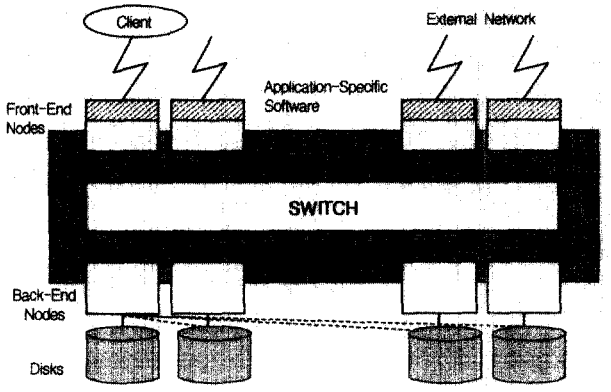
다수의 사용자에게 고품질의 서비스를 제공하기 위해서는 고성능의 멀티미디어 VOD 서버가 필수적이며 멀티미디어 서버의 처리율을 높이고 응답시간을 최소화하는 방법이 필요하다. 싱글 멀티미디어 VOD 서버의 한계를 극복한 클러스터 VOD 서버는 각 노드의 부하를 균등화하여 사용자에게 응답시간을 줄임으로 서비스 질(Quality of Service)을 높일 수 있어야 한다. 클러스터 서버에서의 일반적인 부하 분산 방법과 VOD 서버를 위한 부하 분산 방법이 있다. 또한 멀티미디어 정보는 대용량이므로 저장 시스템을 계층 구조로 구성하여 고속의 저장 장치인 버퍼를 이용하는 것이 효율적이다. 즉, 요구 데이터를 디스크로부터 읽어 온 후 버퍼에 저장하여 같은 데이터를 요구하는 사용자에게 제공하면 대용량 저장장치로의 접근 빈도 수를 줄여 응답 지연시간을 최소화할 수 있다. 버퍼를 서로 공유함으로써 병행적으로 서비스 받을 수 있는 사용자의 수가 증가하고 입출력 장치로의 접근 빈도 수가 감소된다. 멀티미디어 정보에 대한 버퍼의 관리 방법으로 버퍼 교체 방법과 버퍼 선인출 방법이 있다[9]. 다음에서 클러스터 VOD 서버 구조와 지금까지의 클러스터 VOD 서버에 대한 부하 분산 방법과 버퍼 관리 방법에 대한 연구 결과를 살펴본다.

2.1 클러스터 VOD 서버 구조 및 부하 분산

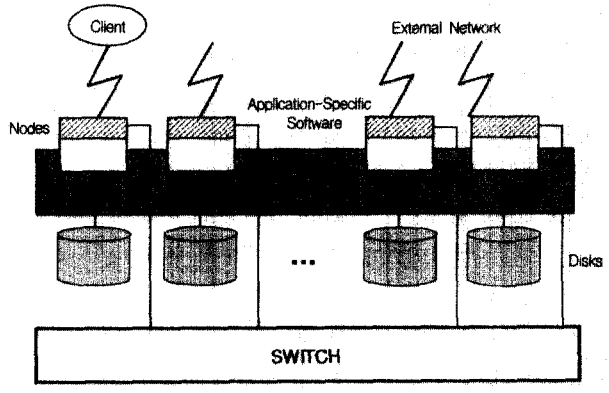
2.1.1 클러스터 VOD 서버 구조

클러스터 VOD 서버의 일반적인 구조는 스위치(interconnection network)에 의해 연결된 노드의 그룹으로 표현할 수 있다. 또 각 노드는 로컬 디스크 배열을 갖는다. 전체 클러스터의 노드는 논리적으로 front-end 노드와 back-end 노드 부분으로 나누어진다. front-end 노드는 외부 네트워크로부터 요구된 스트림을 미리 정의된 순으로 처리하거나 승인 전략(admission control strategy)에 의해 처리한다. Back-end 노드는 로컬 디스크 배열에 데이터를 저장하며, front-end로부터 데이터 요구가 왔을 때 데이터를 front-end 노드에 제공한다. 제공된 데이터는 스트림 당 하나씩 리드-어헤드(read-ahead) 버퍼에 버퍼링 되며 외부 네트워크를 통해 사용자에게 일정한 간격으로 전송된다[12, 14].

클러스터 VOD 서버는 (그림 1)과 같이 two-tier와 flat 방식 구조를 갖는다. (그림 1)의 Two-tier 구조에서 front-end와 back-end 노드는 서로 다른 물리적인 노드로 구별된다. 반면에, Flat 구조에서는 모든 노드가 front-end와 back-end 노드의 물리적인 차이 없이 존재하며 데이터 저장기능과 전송기능을 모두 수행한다[5].



(a) Two-Tier



(b) Flat

(그림 1) 일반적인 클러스터 VOD 서버 구조

2.2.2 클러스터 VOD 서버의 부하 분산 방법

클러스터 VOD 서버를 구성할 경우 가장 우선시 되는 목표는 각 노드간의 부하를 균등화하는 것이다. 각 노드의 부하를 균등화하기 위해 일반적으로 사용되는 방법은 각 노드의 접속 수에 의한 방법이다. 그러나 VOD 서버의 경우 접속 수가 노드의 부하량을 의미하지 않는다. 각 노드의 부하량은 디스크 접근 빈도 수이며, 각 노드의 버퍼 관리 방법에 따라 디스크 접근 수를 줄일 수 있으므로 디스크 접근 빈도 수와 사용자 연결 수가 비례하지 않는다. 따라서 기존의 부하 분산 방법을 클러스터 VOD 서버에 적용할 경우 부하의 불균등을 초래할 수 있다.

클러스터 서버에서 사용되는 일반적인 부하 분산 방법은 네트워크 접속을 일정 순서의 라운드 로빈으로 노드를 연결하는 라운드 로빈(Round-Robin) 방법과 각 노드의 처리 용량을 고려하여 할당되는 가중치 라운드 로빈(Weighted Round-Robin) 방법, 가장 적은 접속 수를 가진 노드로 접속하는 최소 접속(Least-Connection) 방법, 각 서비스 노드의 성능 가중치에 따라 접속을 처리하는 가중치 최소 접속(Weighted Least Connection) 방법 등이 있다[3].

클러스터 VOD 서버의 중요한 부하량은 디스크 접근 빈도이다. 따라서 클러스터 VOD 서버의 부하 분산을 위해 각 노드의 디스크 접근 빈도를 일정하게 유지하는 방법이 주로 연구되었다. 클러스터 VOD 서버의 부하 분산 방법중 대표적인 방법은 스트라이핑으로 데이터를 일정량으로 분산하여 보관하며, 사용자 요구가 들어올 경우 각 노드로부터 데이터를 검색한 후 내부 네트워을 통해 하나의 노드에서 데이터를 재조합하여 사용자에게 전송하는 방법이다. 데이터가 여러 노드에 분리되어 존재하기 때문에 사용자 요구를 처리하기 위해 다수의 노드가 참여하며 모든 노드에 부하가 분산될 수 있다. 그러나 데이터의 정보가 변경될 경우 전체 노드에 재스트라이핑 하는 작업이 필요하다.

동적 복제 방법은 클러스터 내의 노드 성능을 최대한 활용하는 방법으로, 사용자 요구가 많은 데이터를 다른 노드에도 복제해 놓음으로써 부하를 분산시킬 수 있는 방법이다. 기본 알고리즘은 사용자의 요구 패턴에 따라 필요한 데이터를 동적으로 복제 혹은 삭제하는 것이다. 그러나 동적 복제의 경우 복제하는 비용이 크고 복제를 위한 디스크 여유 공간이 필요하다는 단점이 있다[7, 11].

2.2 클러스터 VOD 서버 버퍼 관리 방법

멀티미디어 정보는 대용량, 연속 접근 특성을 갖는다. 따라서 멀티미디어 VOD 서버를 위한 많은 버퍼 관리 방법들이 제안되었다. 가장 대표적인 버퍼 관리 방법으로 버퍼 교체 방법과 버퍼 선인출 방법이 있다[9].

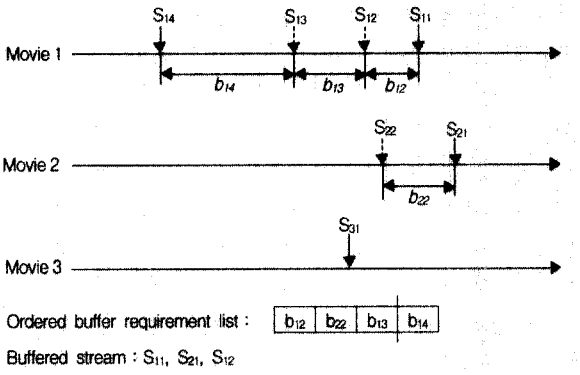
기존의 버퍼 교체 알고리즘인 LRU(Least Recently Used)나 MRU(Most Frequently Used)의 경우 데이터 접근 유형을 예측하는 것이 사실상 불가능하여 최적의 교체 방법 구현이 불가능하지만, 멀티미디어 정보 환경에서는 미래에 참조될 데이터에 대한 예측이 가능하다. 따라서 이러한 특성을 이용하여 가장 나중에 참조될 버퍼를 반환함으로써 최대의 버퍼링 효과를 얻을 수 있다. FFU(Far Future Used) 방법은 가장 미래에 참조될 버퍼를 반환하는 방법으로 LRU, MRU보다 개선된 알고리즘이다[10]. 이외에 멀티미디어 정보를 위한 대표적인 버퍼 관리 알고리즘으로 BASIC/DISTANCE, Interval Caching, Generalized Interval Caching 방법이 있다[8].

BASIC 알고리즘은 가장 오랫동안 사용되지 않은 버퍼가 반환되는 방법으로, 일정 주기마다 버퍼의 재접근 시간이 측정되어야하며 접근 시간에 따라 정렬된 리스트가 유지되어야 한다. 따라서 높은 오버헤드를 갖는다. DISTANCE 알고리즘은 각 사용자마다 선행과 후행의 DISTANCE 거리 값이 설정된다. 측정된 DISTANCE는 오름 정렬한 후 유지되며 DISTANCE 값이 가장 적은 순으로 버퍼를 해제하여 가장 최근에 해제된 버퍼를 할당하는 방법이다. BASIC/DISTANCE 방법은 오로지 적중률을 높이는 데에만 초점을 맞추고 있으므로 더 많은 스트림을 서비스 할 때 보장형 서비스를 제공하지 못한다.

Interval Caching은 멀티미디어 정보에 적합하게 고안된

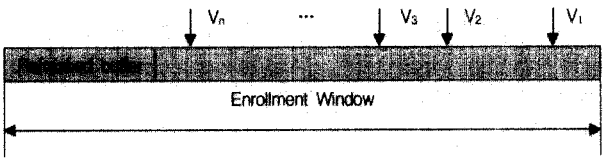
버퍼링 방법으로, 시간적 지역성(temporal locality)과 시스템 부하 변동을 이용한다. 한 쌍의 연속된 스트림들을 각각 선행 스트림(preceding stream)과 후행 스트림(following stream)이라 한다. 선행 스트림과 후행 스트림 사이의 데이터를 인터벌(interval)이라고 정의한다. 하나의 인터벌을 버퍼링 하면 그 인터벌의 후행 스트림은 디스크 접근 없이 버퍼에서 필요한 블록을 읽게 된다. Interval Caching은 (그림 2)와 같이 인터벌을 버퍼링의 단위로 사용할 것을 제안하였다. 그러나 이 방법에는 적중률을 높이기 위한 교체 방법은 포함되어 있지 않다[1]. Generalized Interval Caching 방법은 긴 분량의 데이터 이외에 짧은 Clip 데이터를 지원하기 위해 Interval Caching 방법을 확장한 것이다[8].

멀티미디어 데이터는 연속미디어 정보이므로 버퍼로 미리 읽어 오는것이 가능하다. 버퍼 선인출 방법은 일정량의 데이터를 버퍼에 미리 저장하여 서비스를 수행한다. 기존의 버퍼 선인출 방법으로는 버퍼 비분할 방법, 정적 버퍼 분할 방법, 적응 버퍼 분할 방법 등이 있다[9].



(그림 2) Interval Caching 방법

버퍼 비분할 방법은 (그림 3)과 같이 전체 버퍼를 하나의 그룹으로 보고 관리한다. 이 방법의 목적은 평균 대기시간을 최소화하기 위한 것으로 버퍼가 모두 차기 이전에 들어온 사용자는 이미 원하는 데이터가 버퍼에 있으므로 디스크 접근이 필요 없으며, 따라서 평균 대기시간이 없어지게 된다. 그러나 Enrollment Window가 닫힌 이후에 도착한 사용자는 서비스가 끝날 때까지 대기해야 한다는 단점이 있다.



(그림 3) 버퍼 비분할 방법

본 논문에서 사용되는 용어로 "Enrollment Window"는 동일한 비디오 스트림을 요구하여 디스크 접근 없이 버퍼에서 서비스 받고 있는 사용자의 그룹을 정의한다. "열려진 Enrollment Window"는 새로운 사용자가 참여했을 때 디스크 접근 없이 버퍼 재 참조만으로 서비스가 가능한 Window를 말한다. 반대의 경우를 "닫혀진 Enrollment Window"라 한다.

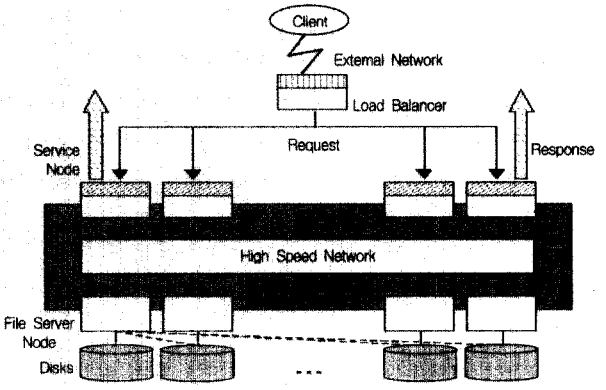
정적 버퍼 분할 방법은 전체 버퍼를 일정한 크기의 부버퍼로 분할하고 분할된 각 부버퍼는 비분할 방법을 사용한다. 이 방법은 각 부버퍼에 필요한 스트림들이 연속적으로 있지 않으므로 디스크 탐색 시간이 증가하게 된다. 적응 버퍼 분할 방법은 정적 버퍼 분할 방법에서 단점으로 나타난 버퍼 활용도를 높이기 위한 방법이다.

3. 동적 버퍼 분할을 이용한 부하 분산 방법

클러스터 VOD 서버 환경에서 연속미디어 스트림을 요구하는 사용자들에게 평균 대기시간을 최소화하면서, 클러스터 내 각 서비스 노드의 버퍼를 동적 버퍼 분할 방법을 적용하여 VOD 서버 성능을 최대로 사용할 수 있는 부하 분산 방법을 제안한다. 제안된 부하 분산 방법은 동적 버퍼 분할 방법을 기반으로 부하 분산 작업을 수행하게 된다.

3.1 클러스터 VOD 서버의 부하 분산기(Load Balancer)

본 논문에서 사용하는 클러스터 VOD 서버 구조는 기존의 two-tier 방식을 변형한 것이다. (그림 4)와 같이 일반적인 two-tier 방식에 부하 분산기를 추가한 구조이다. 이러한 구조는 리눅스 가상 서버 구조와 유사하며 부하 분산기를 통해 효율적인 부하 분산이 가능하다는 장점을 갖는다[13].



(그림 4) 사용된 클러스터 VOD 서버 구조

부하 분산기의 목적은 클러스터 내 다수의 서버간에 부하를 분산시키는 것이다. 부하가 집중된 서버를 자동 감지해 다른 서버로 트래픽을 분산시킴으로써 전체 서버의 성능을 균등화하고 빠른 응답속도를 유지하게 한다. 본 논문에서 제안한 부하 분산기는 각 VOD 서비스 노드의 부하 정보와 열려진 스트림에 대한 버퍼 정보를 수집하여 서비스 노드들 간의 부하를 적절히 유지하며 서비스 노드의 성능을 최대로 높일

수 있도록 요구를 분배한다.

제안된 부하 분산 방법은 2 단계로 나누어 처리된다. 첫 번째 단계는 사용자가 요구한 동일 스트림 데이터를 서비스하는 노드가 있는지 검색하여 해당 노드의 버퍼를 재 참조할 수 있다면 노드의 부하량에 상관없이 요구를 분배하는 단계이다. 두 번째 단계는 요구를 재 참조할 수 있는 버퍼가 존재하지 않을 경우 각 서비스 노드의 부하량을 측정하여 부하가 가장 적은 노드로 요구를 분배하는 단계이다. 이 방법의 처리 절차는 <표 1>의 알고리즘과 같다.

〈표 1〉 동적 버퍼 분할을 이용한 부하 분산 알고리즘

```
새로운 request 도착 ;
Enrollment Window Table를 검색 ;

If (request 처리 노드 존재 == TRUE)
    해당 노드로 요구 분배 ;
else
{
    노드 관리 테이블 검색 ;
    if (가용 버퍼 존재 == TRUE )
        전체 그룹 수와 현재 그룹 수의 차이가 가장 큰 곳으로 요구
        분배 ;
    else if(최대 인터벌 > 0)
        해당 노드로 요구 분배 ;
    else
        대기큐에서 대기 ;
}
```

본 논문에서 제안한 부하 분산기의 요구 분배 방법은 내용기반 요구 분배(contents-aware request distribution) 방법으로 각 서비스 노드의 버퍼 관리 방법인 동적 버퍼 분할 방법을 이용한다. 즉, 사용자가 요구하는 내용에 의해 해당하는 VOD 서버 노드로 요구 분배되어지고 해당 노드의 버퍼를 동적으로 활용하게 한다. 따라서 사용자에 의해 동일 요구 스트림이 연속적으로 들어올 경우, 버퍼링 성능이 크게 증가하여 디스크 접근없이 버퍼의 재 참조만으로 서비스하게 됨으로써 초기 지연시간을 크게 줄일 수 있다.

3.2 부하량 측정에 따른 요구 분배

멀티미디어 VOD 서버의 경우 비디오 재생시 버퍼의 재사용을 고려하지 않는다면 성능에 가장 큰 영향을 미치는 부분이 디스크 접근 횟수이다. 최대 디스크 접근 횟수 이상이 될 경우 비디오 재생시 끊김 현상이 발생하게 되어 비디오의 원활한 재생이 불가능하게 된다. 따라서 각 VOD 서비스 노드의 디스크 접근 횟수는 각 서비스 노드 부하에 직접적인 영향을 미치게 되므로 본 논문에서는 서비스 노드의 디스크 접근 빈도 수를 부하량으로 사용한다.

사용자의 요구 파일이 있는 버퍼의 서비스 노드가 존재하지 않을 경우, 즉 각 서비스 노드 내의 Enrollment Window 그룹이 모두 닫혀 있을 경우, 부하 분산기는 각 서비스 노드의 부하량을 측정하여 부하가 가장 적은 노드로 요구를 분배하게 된다.

각 서비스 노드의 버퍼 정보와 부하량 정보를 유지하기 위해 부하 분산기는 <표 2>와 같은 정보를 서비스 노드로부터 수집하여 관리한다.

〈표 2〉 부하 분산기에서 유지하는 각 노드의 정보

유지 정보	설 명
Node ID	각 노드를 식별할 수 있는 식별자
각 노드의 그룹 수	각 서비스 노드에서 실질적으로 사용하고 있는 그룹 수
최대 그룹 수	각 노드에서 제공될 수 있는 최대 그룹 수
가용 버퍼량	각 노드에 남아있는 버퍼의 양
최대 버퍼 인터벌	각 노드의 서비스 그룹들 내에서 사용되는 버퍼의 최대 참조 간격
열려진 Enrollment Window	서비스 노드의 버퍼를 재 참조 할 수 있을 경우 재 참조가 가능한 비디오에 대한 이름과 상태 정보

<표 2>의 정보에 의해 구성한 실제 데이터 구조는 <표 3>과 <표 4>에 나타낸다. <표 3>은 각 서비스 노드의 정보 테이블이다. 각 서비스 노드의 현재 그룹 수와 최대 그룹 수의 차이가 노드의 부하량이며 차이가 클수록 부하량이 적음을 나타낸다. 따라서 부하 분산기는 부하량이 적은 노드로 요구를 분배시킨다.

〈표 3〉 노드 정보 테이블

노드 아이디	현재 그룹 수	최대 그룹 수	가용 버퍼량	최대 버퍼 인터벌
1	34	100	TRUE	102
2	35	100	FALSE	378

동적 버퍼 분할 방법에 의해 현재 열려진 Enrollment Window 정보 테이블이 <표 4>에 나타나 있다. 그러므로 부하 분산기에 의해 분배된 노드의 현재 열려진 Enrollment Window 정보 테이블에 요구하는 비디오 파일이 존재하면 디스크 접근 없이 버퍼 재 참조만으로 서비스가 가능하게 된다.

〈표 4〉 열려진 Enrollment window 정보 테이블

노드 아이디	열려진 파일 명	상 태
1	1. mpg	Play
1	2. mpg	FF
2	3. mpg	FB

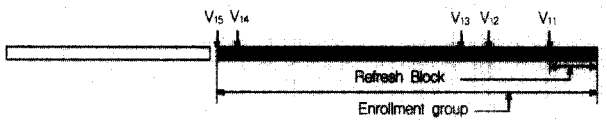
3.3 동적 버퍼 분할 방법

동적 버퍼 분할 방법은 연속미디어 스트림을 요구하는 다수의 사용자들에게 최소의 초기 지연시간과 평균 대기시간을 제공할 수 있는 버퍼 선인출 방법이다.

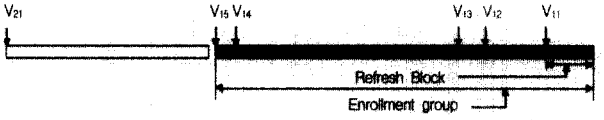
부하 분산기를 통해 분배된 요구는 각 서비스 노드에 의해 처리되며, 각 서비스 노드는 동적 버퍼 분할 방법을 적용하여 사용자의 요구를 처리한다. 기존의 버퍼 선인출 방법에 비해

적용된 알고리즘은 버퍼를 필요할 때마다 동적으로 분리하여 사용함으로써 버퍼의 활용도를 높일 수 있으며, 이에 따라 서비스를 요구하는 사용자들의 평균 대기시간을 크게 감소시킬 수 있다.

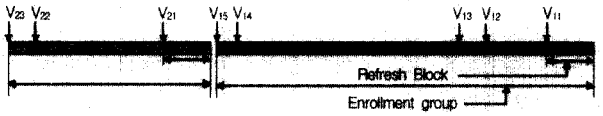
동적 버퍼 분할 방법은 처음에는 전체 버퍼를 하나의 그룹으로 관리 하다가 각 그룹에서 활용도가 가장 낮은 버퍼를 선택 재분할하여 사용하므로 저장장치의 대역폭 한계에 천천히 도달하게 된다. 따라서 버퍼의 재분할이 많이 되어도 사용자들의 평균 대기시간은 크게 늘어나지 않는다. 또한 동적 버퍼 분할 방법은 미래에 들어올 사용자를 위한 재 참조를 고려한 알고리즘으로 사용자들의 서비스 요구 간격이 길 경우 그 부분의 버퍼 재사용으로 버퍼 활용도가 높아지게 된다 [15, 16]. 이 방법의 처리 절차는 (그림 5)에 나타낸다. (그림 5)의 동적 버퍼 분할 방법의 버퍼 갱신과정에 대한 각 단계의 설명은 다음과 같다.



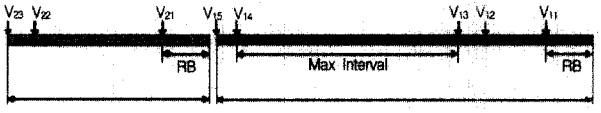
(1) Group 형성



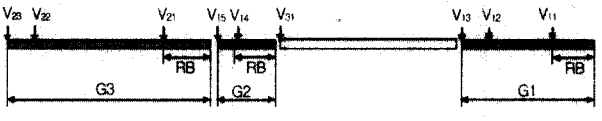
(2) 새로운 사용자 admission



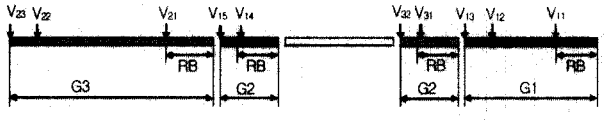
(3) 새로운 Group 형성



(4) 회수할 버퍼 선택



(5) 새로운 사용자를 위해 회수한 버퍼 할당



(6) 새로운 Group 형성

(그림 5) 동적 버퍼 분할 방법

제 1 단계 : (그룹 형성) 첫 번째 사용자가 비디오를 보기 위해 참여한 후, 첫 번째 사용자가 비디오를 보고 있는 동안 동일한 비디오를 요구한 다른 여러 사용자가 임의의 시간에 도착해 비디오를 보기 위해 참여한다. 따라서 승인이 허락된 사용자를 위한 비디오 데이터는 모두 버퍼에 있으므로 지연시간 없이 서비스 할 수 있게 된다. 여기서 첫 번째 사용자가 버퍼에 있는 프레임을 모두 서비스 받은 후 버퍼에 없는 스트림을 원할 경우, 첫 번째 사용자부터 마지막 사용자까지를 하나의 그룹으로 한다.

제 2 단계 : (새로운 사용자 참여) 첫 번째 그룹을 생성한 다음 그 이후로 처음 도착한 사용자는 두 번째 그룹의 선두 사용자로서 첫 번째 그룹에서 사용하지 않는 버퍼를 이용한다.

제 3 단계 : (새로운 그룹 형성) 그 이후로 동일한 비디오 파일을 요구하는 두 번째 그룹을 형성한다. 마찬가지로 3, 4, 5, … 번째 그룹을 형성하기 위해 위의 단계를 반복한다. 이 때 형성될 수 있는 그룹의 최대 개수는 저장장치의 대역폭 한계를 만족해야 한다. 그룹내의 사용자들이 차례로 스트림을 재 참조하고 나면, 선두 사용자는 마지막 사용자까지 참조한 버퍼를 비우고 비운 버퍼에 계속 저장장치로부터 새로운 스트림을 미리 읽어 놓으면서 진행한다.

제 4 단계 : (회수할 버퍼 선택) 형성된 그룹에서 모든 버퍼를 사용하고 있을 경우 새로운 그룹을 형성하기 위해 기존 그룹이 사용하고 있는 버퍼를 회수해야 한다. 이를 위해 각 그룹내의 사용자들 중 참조 시간 간격이 가장 큰 두 사용자를 선택하여 두 사용자가 속한 하나의 그룹을 두 개로 분리한 후 두 사용자 사이의 버퍼를 회수한다.

제 5 단계 : (회수한 버퍼 할당) 새로 들어온 사용자를 위해 이미 회수한 버퍼를 할당한다.

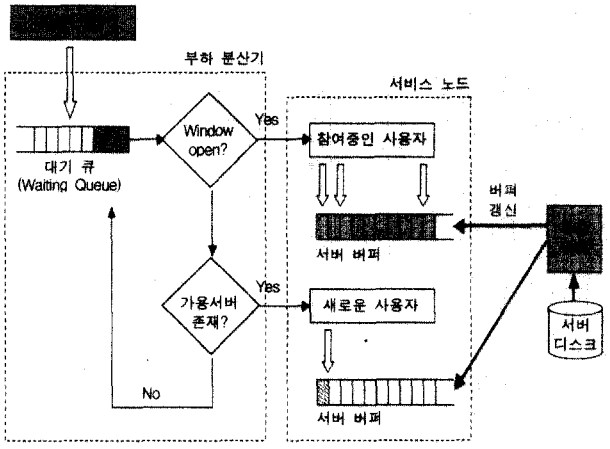
제 6 단계 : (새로운 그룹 형성) 할당된 버퍼에 위의 단계들을 반복하여 새로운 그룹을 형성한다.

4. 성능 평가

본 논문에서 제안한 클러스터 VOD 서버 상에서의 동적 버퍼 분할 방법을 적용한 부하 분산 방법에 대하여 성능 분석을 위해 시뮬레이션 하였다. 시뮬레이션 모델 및 환경 설정은 discrete-event simulation 라이브러리인 SMPL과 Visual C++ 프로그램을 이용하였다.

4.1 시뮬레이션 모델

본 논문의 성능 분석을 위한 시뮬레이션 모델은 (그림 6)과 같다. 우선 클러스터 VOD 서버의 서비스를 받기 위한 사용자는 임의의 시간 간격으로 도착한다.



(그림 6) 시뮬레이션 모델

도착 시간은 평균 도착 시간의 지수 분포를 사용하였다. 또 요구 파일에 대한 분포는 Zipf 분포를 이용하였다. 서비스를 요구한 사용자는 먼저 부하 분산기의 대기 큐에서 처리를 기다린다. 부하 분산기는 사용자 요구를 분석하여 열려진 Enrollment Window의 요구일 경우 해당 비디오를 서비스하고 있는 노드로 요구를 분배하고, 그렇지 않을 경우 최소 부하량을 가진 노드로 분배한다. 그러나 서비스 가능한 노드가 존재하지 않을 경우 부하 분산기의 대기 큐에서 대기한다.

파일 서버의 디스크에는 MPEG 블록으로 압축된 데이터가 저장되어 있으며, 서비스에 참여중인 사용자를 위한 MPEG 블록이 서버의 버퍼에 선인출 된다.

4.2 부하 분산 방법의 성능 평가

〈표 5〉 시뮬레이션 파라미터

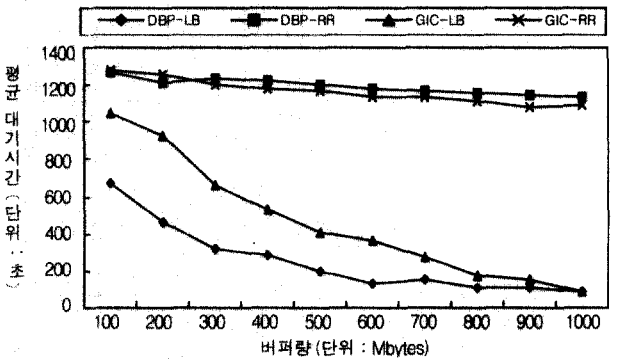
파 라 미 터	값	단 위
전체버퍼크기	500(100 … 1000)	Mbytes
압축된 블록의 크기	0.06	Mbytes
재생률	4	Bblocs/sec
디스크속도	20(20, 30, … 100)	Mbytes/sec
비디오 재생 시간	6000	sec
평균서비스요구간격	10(5, 10, … 30)	sec
Refresh Slack	0.012	block

본 논문에서 사용되는 파라미터는 <표 5>와 같이 정의하였다. 성능 평가 시뮬레이션은 각 서비스 노드에 동적 버퍼 분할 방법과 Generalized Interval Caching 방법을 본 논문에서 제안한 부하 분산 방법과 라운드 로빈 방법에 각각 적용하였을 때 발생하는 지연 시간을 측정하였다.

시뮬레이션 결과 그래프에 표시되는 DBP-LB는 동적 버퍼 분할 방법에 부하 분산 방법을 적용한 경우를, DBP-RR은 동적 버퍼 분할 방법에 라운드 로빈 분배 방법을, GIC-RR은 Generalized Interval Caching 방법에 라운드 로빈 분배 방법을 적용했음을 나타낸다. 또한 GIC-LB는 Generalized Interval Caching 방법에 부하 분산 방법을 적용한 결과이다. GIC에 적용한 부하 분산 방법은 다음과 같다. 첫 번째, 새로운 요구가 같은 파일에 대해 인터벌을 형성할 때 새로 형성된 인터벌이 서비스 노드의 최대 인터벌보다 작을 경우 버퍼링 가능한 노드로 요구를 분산한다. 두 번째, 새로 형성된 인터벌이 각 노드에서 처리하고 있는 인터벌보다 클 경우 디스크 접근 빈도가 가장 작은 노드로 요구를 분산한다.

(그림 7)은 VOD 서비스 노드의 수를 5개로 유지하고 각 서비스 노드의 버퍼량을 증가시켰을 때 각 적용방법의 평균 대기 시간을 측정한 그래프이다. 그래프에서 버퍼량이 증가함에 따라 평균 대기시간이 점차 줄어드는 것을 볼 수 있다. 특히 제안된 부하 분산 방법이 라운드 로빈 방법에 동적 버퍼 분할 방법과 인터벌 캐싱 버퍼 관리방법을 사용하는 것보다 성능이 훨씬 우수함을 볼 수 있다. Generalized Interval Caching 방법과 동적 버퍼 분할 방법에 각각 라운드 로빈 방법을 사용한 경우, 이전 요구와의 관계를 고려하지 않고 요구 분배가 이루어짐으로 버퍼량 증가는 평균 대기시간에 큰 영향을 미치지 못함을 알 수 있다. 그러나, 부하 분산 방법을 적용한 경우 같은 비디오에 대해 이전 요구와의 관계를 고려하여 버퍼 재 참조가 가능한 쪽으로 요구 분배가 이루어지므로 버퍼량이 커질수록 평균 대기시간이 줄어든다.

시뮬레이션 환경은 사용자들의 평균 서비스 요구 시간 간격을 9초로 하였으며, 디스크 속도는 27Mbytes/sec로, skew factor 0.271의 Zipf 분포를 이용하여 비디오 파일을 선택하도록 하였다. 또, 총 비디오 파일의 수는 500개로 하였으며 시뮬레이션 시간은 8시간으로 하였다.

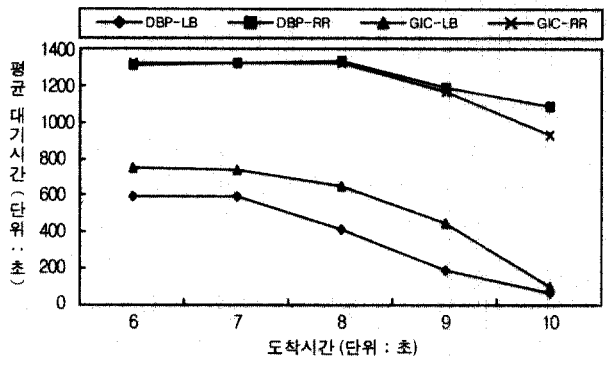


(그림 7) 버퍼량 변화에 따른 평균 대기시간 측정

(그림 8)은 사용자들의 평균 서비스 요구 시간 간격을 6초

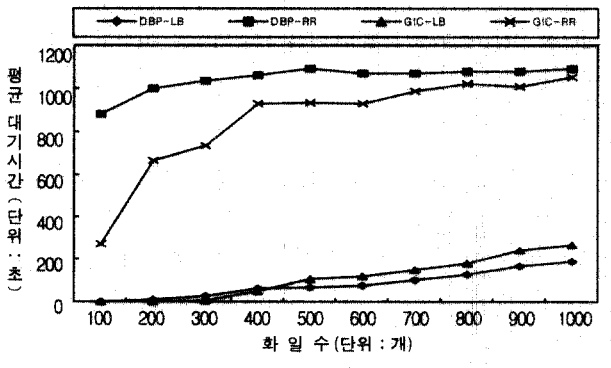
에서 10초까지 변경하면서 위의 네 가지 적용 방법에 대해 평균 대기시간을 측정한 그래프이다. 그래프에서 사용자들의 평균 도착 시간이 증가함에 따라 각 방법들의 평균 대기 시간이 크게 감소함을 알 수 있다. 도착 시간이 10초 이상이 되면 전체 클러스터 VOD 서버는 사용자들의 요구를 지연시간 없이 바로 처리할 수 있게 된다. 특히, 제안된 DBP-LB 방법의 경우 DBP-RR보다 평균 6배 이상 성능 향상을 보였으며, GIC-LB와 비교하면 1.5배정도 우수한 성능을 보이는 것으로 나타났다.

위와 동일한 시뮬레이션 환경으로 버퍼의 크기는 500MBytes로 하였다.



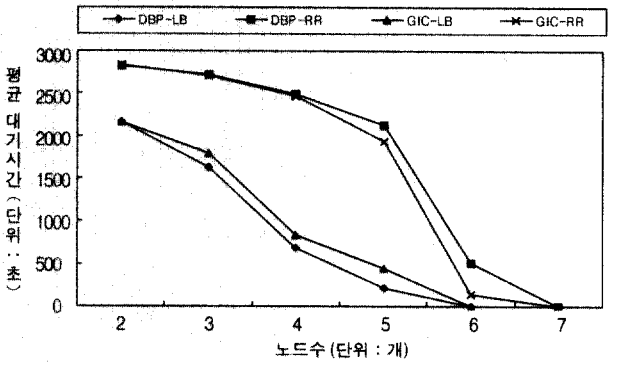
(그림 8) 도착 시간 변화에 따른 평균 대기 시간 측정

(그림 9)는 클러스터 VOD 서버에서 제공하는 비디오의 수가 100개에서 총 1000개까지 변화할 때 각 방법의 평균 대기 시간을 측정한 것이다. 같은 시뮬레이션 환경에서 적용한 결과, 본 논문에서 제안한 DBP-LB 방법이 파일 수에 다소 영향을 받으나 평균 대기 시간의 증가폭은 크지 않음을 알 수 있다.



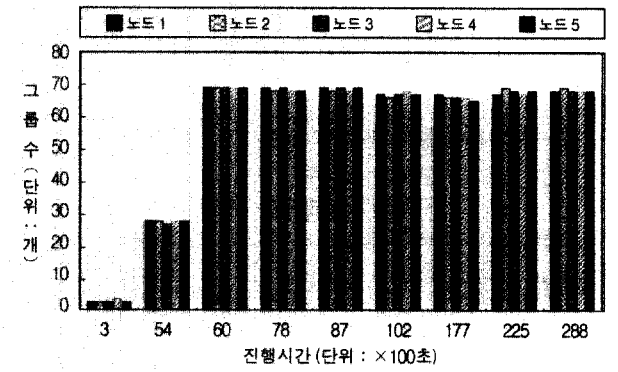
(그림 9) 파일 수 변화에 따른 평균 대기 시간 측정

(그림 10)은 VOD 노드 수 변화에 따른 각 방법의 평균 대기 시간 변화를 조사하였다. VOD 노드 수를 2개에서 7개까지 증가시키면서 각 노드 수에 따른 평균 대기시간을 측정하였다. 동일한 환경의 시뮬레이션 결과, 노드 수가 증가함에 따라 각 방법의 평균 대기 시간이 줄어들었으며 본 논문에서 제안한 부하 분산 방법을 적용한 방법이 다른 두 방법 보다 평균 대기 시간이 크게 낮은 것으로 나타났다. 즉, 노드 수가 5개일 때 DBP-LB 방법은 GIC-LB보다 약 2배, GIC-RR보다 약 9배 성능이 우수한 것으로 나타났다. 특히 노드 수가 7개 이상일 때는 노드의 처리량이 증가하여 평균 대기시간이 없는 것으로 나타났다.



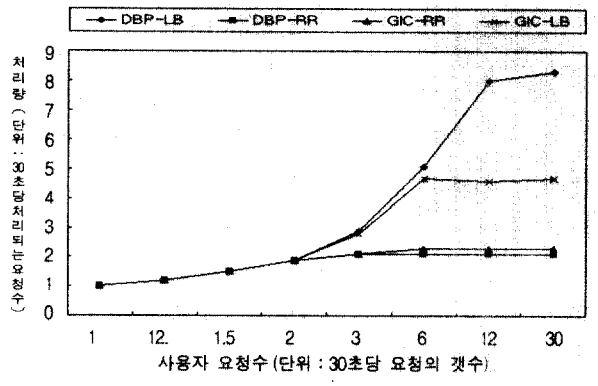
(그림 10) 노드 수 변화에 따른 평균 대기 시간 측정

(그림 11)은 제안된 부하 분산 방법이 각 노드의 부하를 얼마나 일정하게 유지할 수 있는지 나타낸다. 300초에서 28800초까지 서비스 시간이 경과함에 따라 각 VOD 서비스 노드의 부하량(디스크 접근 빈도)을 측정하였다. 동일한 시뮬레이션 환경에서 노드의 수는 5개, 사용자의 서비스 요구 평균 시간 간격은 10초로 시뮬레이션 하였다. 그래프에서 보는 바와 같이 서비스 시간이 증가하여도 제안된 부하 분산 방법을 적용할 경우 처리되는 그룹의 수가 일정하게 유지되는 것을 볼 수 있다.



(그림 11) 서비스 진행 시간에 따른 각 노드의 부하량 측정

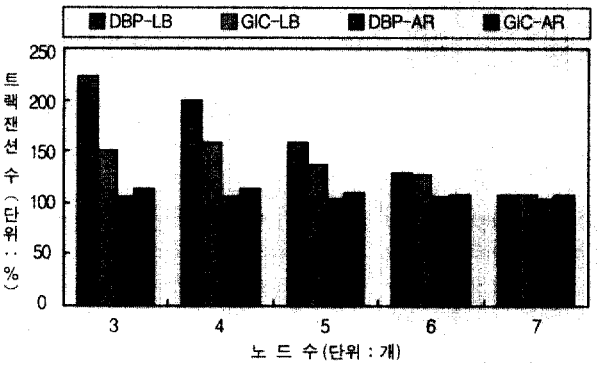
(그림 12)는 30초 사이에 들어오는 사용자의 서비스 요구에 대한 평균 도착 수를 정량화하여 구한 처리량(Throughput)이다. 즉, 1초에서 30초 사이의 사용자 요구수에 대한 처리량을 구했다. 동일한 시뮬레이션 환경에서 제안된 DBP-LB 방법의 처리량은 DBP-RR, GIC-RR보다 약 4.2배 뛰어남을 보였으며, GIC-LB보다 1.6배 우수함을 보였다.



(그림 12) 30초당 요청 빈도 수에 따른 처리량

마지막으로 (그림 13)은 VOD 서비스 노드 수가 증가함에 따라 각 방법이 지원하는 병행 사용자 수를 보여준다. 동일한 시뮬레이션 환경으로 노드 수가 작을때 DBP-LB와 GIC-LB의 경우, 부하 분산기 대기 큐에서 서비스 노드로 분배 시 참조 시간간격이 짧아져 더 많은 병행 사용자를 지원하는 특성이 나타났다.

결과적으로 DBP-RR, GIC-RR의 경우 노드 수에 큰 영향을 받지 않으나, DBP-LB의 경우 버퍼 관리를 하지 않을 경우보다 평균적으로 177%의 사용자를 더 처리할 수 있었다. 또한 GIC-LB의 경우는 143% 정도이며, 이에 비해 제안된 방법이 약 30% 더 많은 병행 사용자를 지원할 수 있는 것으로 나타났다.



(그림 13) 노드 수 변화에 따른 병행 사용자 수

5. 결 론

클러스터 VOD 서버는 서버간의 부하를 효율적으로 분산할 수 있어야 한다. 이를 위해 각 서비스 노드의 부하 정보를 적절히 파악할 수 있어야 하며, 어떤 요소가 서버의 부하에 가장 큰 영향을 미치는 가를 파악해야 한다.

클러스터 VOD 서버에서 가장 크게 부하를 일으키는 부분은 디스크의 접근 빈도 수이다. 따라서 각 서비스 노드의 부하를 적절히 분산하기 위해서는 서비스 노드의 접근 빈도 수를 이용한 부하 분산 방법이 필요하다. 또한 연속미디어 정보를 제공하는 VOD 서버는 사용자의 서비스 요구에 대해 연속미디어를 끊김 현상 없이 실시간으로 서비스 할 수 있어야 한다. 이를위해 시간 제한 조건 내에 저장 장치에 저장된 연속미디어를 접근할 수 있어야 하므로 저장 장치의 접근 빈도 수에 크게 좌우된다. 따라서 한번 읽은 연속미디어 데이터를 버퍼링하여 재사용하면 저장 장치의 접근 빈도수를 줄일 수 있다.

본 논문은 클러스터 VOD 서버 환경에서 효율적인 VOD 서비스를 제공하기 위하여 동적 버퍼 분할 방법을 이용한 부하 분산 방법을 제안하였다. 멀티미디어 정보는 주로 순차적으로 접근되고 연산의 지속 시간이 긴 특징을 가진다는 점에 착안하여, 본 논문에서 제안한 방법은 연속미디어 저장 시스템을 위해 동적으로 버퍼를 관리함으로써 버퍼 성능을 최대로 활용할 수 있는 부하 분산 방법이다.

시뮬레이션 성능 분석을 통하여 클러스터 VOD 서버 환경에서 제안된 방법이 기존의 다른 버퍼 관리 방법과 부하 분산 방법을 적용하였을 경우에 비해 부하량을 적절히 조절하면서, 평균 대기시간을 감소시키고 처리량을 높일 수 있음을 보였다. 또한 동일 환경 하에서 처리할 수 있는 병행 사용자 수도 30% 이상 월등함을 보였다. 따라서 성능 분석 결과, 전체 클러스터가 과부하 상태일 때 본 논문에서 제안한 DBP-LB 방법이 GIC-LB보다 약 1.5~2배 성능이 우수한 것으로 나타났으며, 여러 시뮬레이션을 통해 제안 방법의 우수함을 입증하였다.

참 고 문 헌

[1] D. P. Wu, Y. W. T. Hou, and W. W. Zhu, "Streaming Video over the Internet : Approaches and Directions," IEEE Transactions on Circuits & Systems for Video Technology, Vol.11 No.3, 2001.

[2] D. Jadav and A. Houdhary, "Design Issues in High Performance Media-on-Demand Servers," IEEE Parallel and Distributed Technology Systems and Applications, Summer, 1995.

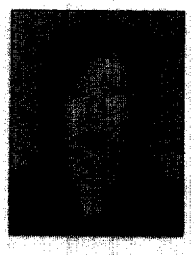
[3] W. Zhang, "Linux Virtual Server for Scalable Network Service," Linux Virtual Server Project, 1998.

[4] 최재영, 최종명, "고가용성 리눅스", 정보처리학회논문지, Vol. 6, 1999.

[5] R. Tewari and R. Mukherjee, "Design and Performance Tradeoffs in Clustered Video Servers," Proceedings of the International Conference on Multimedia Computing and Systems, Los Alamitos, CA, 1996.

[6] A. Garica-Martinez and J. Fernandez-Conde, "Efficient

Memory Management in Video on Demand Servers," Computer Communications 23, 2000.
[7] P. W. K. Lie and J. C. S. Lui, "Threshold-based Dynamic Replication in Large-Scale Video-on-Demand Systems," Multimedia Tools and Applications 11, 2000.
[8] A. Dan and D. Sitaram, "A Generalized Interval Caching Policy for Mixed Interactive and Long Video Workloads," Proceedings of Multimedia Computing and Networking, SPIE, San Jose, CA, Jan., 1996.
[9] D. Rotem and J. L. Zhao, "Buffer Management for Video Database Systems," Proceedings of International Conference on Data Engineering, Taipei, Taiwan, March, 1995.
[10] 권택근, 이석호, "연속매체를 위한 FFU 버퍼 재배치 알고리즘", 정보과학회논문지, 제22권 제10호, Oct., 1995.
[11] A. Dan, M. G. Kienzle, and D. Sitaram. "Dynamic Segment Replication Policy for Load-Balancing in Video-on-Demand Servers," ACM Multimedia Systems, Vol.3, No.3, 1995.
[12] O. Sandst, S. Lang ϕ rgen, and R. Midtstraum, "Video Server on an ATM Connected Cluster of Workstations," Proceedings of XVII International Conference of the Chilean Computer Science Society, Valparaso, Chile, November, 1997.
[13] M. Aron, D. Sanders, P. Druschel, and W. Zwaenepoel. "Scalable Content-Aware Request Distribution in Cluster-Based Network Servers," Proceedings of the USENIX, 2000.
[14] L. Aversa and A. Bestavros, "Load Balancing a Cluster of Web Servers using Distributed Packet Rewriting," Proceedings of the 2000 IEEE International Performance, Computing, and Communications Conference, February, 2000.
[15] 권춘자, 최창열, 최황규, "VOD 서버에서 효율적인 연속미디어 서비스를 위한 동적 버퍼 분할 기법", 정보처리학회논문지A, 제9-A권 제2호, June, 2002.
[16] 권춘자, 김영진, 최황규, "클러스터 기반의 VOD 서버 상에서 동적 버퍼 분할을 이용한 효율적인 부하 분산 기법", 정보과학회 봄 학술발표논문집(A), 제29권 제1호, April, 2002.



권 춘 자

e-mail : kwoncj@mail.kangwon.ac.kr
1986년 한양대학교 전자공학과(학사)
1991년 한양대학교 전자계산학과(석사)
1991년~1994년 산업기술정보원 데이터베이스 운영실 연구원
2000년~현재 강원대학교 컴퓨터정보통신공학과 박사과정
2002년~현재 한림정보산업대학 전산정보처리과 초빙교수
관심분야 : 멀티미디어 VOD 시스템, 데이터베이스 시스템, 클러스터 웹 서버 시스템 등

김 영 진

e-mail : yjkim@mail.kangwon.ac.kr
1999년 강원대학교 컴퓨터공학과(학사)
1999년~2000년 (주)에피온
2002년 강원대학교 대학원 컴퓨터정보통신공학과(석사)
2002년~현재 (주)이비즈온 연구소
관심분야 : 멀티미디어 시스템, 데이터베이스 시스템, 소프트웨어 공학 등

최 황 규

e-mail : hkchoi@kangwon.ac.kr
1984년 경북대학교 전자공학과(학사).
1986년 한국과학기술원 전기및전자공학과 (석사)
1989년 한국과학기술원 전기및전자공학과 (박사)
1994년~1995년 Univ. of Florida Database R&D Center 방문교수
1999년~2001년 강원대학교 전자계산소 소장
1990년~현재 강원대학교 전기전자정보통신공학부 교수
2002년~현재 Univ. of Minnesota 방문교수
관심분야 : 멀티미디어 시스템, 데이터베이스 시스템, Intelligent Storage System 등