병렬 VOD 서버의 확장을 위한 스트라이핑 정책

최숙영*, 최현호**, 한주희**, 유관종**
*우석대학교 컴퓨터교육과
*충남대학교 컴퓨터과학과

A Striping Policy for Extension of Parallel VOD Server

Sook-Young Choi*, Hyun-Ho, Choi**,
Joo-Hee, Han**, Kwan-Jong Yoo**
*Dept. of Computer Education, Woosuk University
**Dept. of Computer Science, Chungnam University

요 약

본 논문에서는 확장 가능한 병렬 VOD서버 모델을 제시하고, 이 병렬 VOD 서버에서 기존의 디스크 공간이 부족하여 새로운 병렬 서버를 추가할 경우, 데이터 분배 문제를 고려한다. 새로운 서버에 미디어 파일을 추가할 경우, 특정 서버에 부하가 몰리는 것을 방지하기 위해 기존의 서버에 저장되어 있는 일부 데이터들을 이동 시켜서 각 서버의 사용가능한 디스크 공간을 조정한 뒤, 각 디스크 부하를 최소화 하도록 고려하여 데이터를 저장하는 스트라이핑 방법을 제시한다.

1. 서 론

컴퓨터와 통신망의 급속한 발달은 네트워크를 이용한 다양한 멀티미디어 서비스를 가능하게 하였다. 이에 따라 멀티미디어 서비스를 전용으로 제공해주는 여러 가지 VOD(Video On Demand) 서버가 등장하였다. 현재 대부분의 모델들은 단일 VOD서버에 의존하고 있다. 하지만 단일 VOD서버는 여러 가지 문제점을 안고 있으므로 최근에는 이의 단점을 보완한 병렬 VOD서버도 제시되고 있다[1,2,3,4]. VOD서버를 개발하기 위한 최근의 연구 초점으로는 수많은 사용자에게 실시간 연속 접근을 보장하기 위한 스트라이핑(striping) 정책, 디스크 스케줄링, 버퍼 관리 등을 들 수 있다[1]. 하지만 위의 연구 이외에 데이터의 증가로 저장공간이 부족해질 경우 서버의 확장을 어떻게 할 것인가에 대한 문제도 고려해 보아야 한다.

본 논문은 확장 가능한 병렬 VOD 서버를 구성하고, 이 병렬 VOD서버에서 기존의 디스크

공간이 부족하여 새로운 병렬 서버를 추가할 경우, 이에 관한 데이터 분배 문제를 고려한다. 즉, 기존의 서버에 저장되어 있는 일부 데이터들을 새로운 서버로 이동 시켜서 각 서버의 사용 가능한 디스크 공간을 조정한 뒤 각 디스크 부하의 최소화를 고려하여 데이터를 저장하도록 하는 스트라이핑 정책에 대해 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 VOD 서버에 대한 관련 연구를 살펴보고 3장에서는 확장 가능한 병렬 VOD 서버의 구성과 각 모듈의 특징을 소개한다. 4장에서는 기존의 병렬 VOD서버에 효율적으로 새로운 서버를 추가하기 위한 방안을 제시하고 이때 필요한 스트라이핑 정책을 제시한다. 5장에서는 4장에서 제시한 스트라이핑 방법에 대한 분석 및 평가를 하고 6장에서 결론을 맺는다.

2. 관련 연구

일반적으로 VOD 서버는 실시간으로 서비스를 제공해 주기 위해서 하나 이상의 프로세서와 여러 개의 기억 장치를 기본 구조로 갖추고 있다. 또한 여러 기억 장치간의 부하를 방지하기 위해서 스트라이핑 정책을 이용하고 있다[1,2,3].

• 서버의 구조

단일 VOD 서버는 데이터를 전부 자신의 디스크에 두고 클라이언트의 요구에 따라 데이터를 전송 해 준다. 단일 VOD서버의 단점으로는 첫째, 제한된 기억장치 용량으로 인해 데이터의 급속한 증가에 제대로 대처하지 못하는 기억 용량 한계의 문제 둘째, 사용자들의 접속 횟수의 증가와 인기를 끌고있는 비디오에 대한 편중된 부하의 증가에 제대로 대처하지 못하는 부하의 편중 문제 셋째, 서버가 고장 났거나 서버에 연결된 네트워크가 다운되었을 시에는 더 이상의 서비스를 제공하지 못하는 서버의 결함 허용성 문제 등이 있다[2,3].

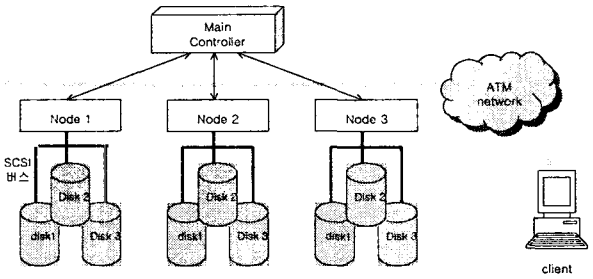
병렬 VOD 서버는 데이터를 저장하는 서버들을 여러 개 둠으로써 인기 있는 비디오에 대한 부하의 편중을 분산시킬 수 있다. 또 특정 서버가 고장 났다거나 네트워크가 다운되었을 시에도 중복성(redundancy)을 이용하여 원활한 서비스를 제공할 수 있다.

• 스트라이핑 정책

스트라이핑 정책은 여러 대의 서버에 데이터를 분산시킬 때 기억 장소를 보다 효율적으로 사용하도록 하고 또한 각 서버에 부하가 균등하게 나누어지도록 스케줄링 하는 정책을 말한다. 스트라이핑 정책은 시간 스트라이핑(time striping)정책과 공간 스트라이핑(space striping)정책으로 나눌 수 있다. 시간 스트라이핑은 비디오 스트림을 프레임 단위로 나누어 여러 대의 서버에 나누어 저장하는 방법이다. 공간 스트라이핑은 비디오 스트림을 고정된 크기(stripe unit)로 분할하여 저장하는 방법이다. 공간 스트라이핑은 같은 크기로 나누어지므로 저장 공간 활용과 버퍼 관리를 간단히 할 수 있다[1,3].

3. 병렬 VOD 서버의 구성

<그림 1>은 본 논문에서 구성한 병렬 VOD서버의 구조를 나타낸다.



<그림 1> 병렬 VOD서버의 구조

병렬 VOD서버는 크게 세 모듈로 나누어진다. 데이터를 요구하는 클라이언트, 여러 개의 저장 노드(storage node)들이 있는 저장 노드 컴포넌트(storage node component), 각 저장 노드를 관리하고 클라이언트로부터 들어온 요청을 처리 해 주는 주 제어기(main controller)가 각각의 모듈

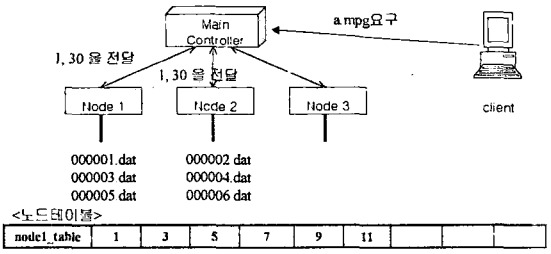
들이다.

병렬 VOD서버는 특정 저장 노드에서의 부하를 줄이기 위해 데이터를 fragment 단위로 잘라 스트라이핑 한다. 주 제어기는 각 fragment의 저장 노드 위치를 기록 해 놓는 제어기 테이블을 가진다.

클라이언트가 미디어 파일을 요구하면 주 제어기는 제어기 테이블을 참조하여 그 미디어 파일의 fragment들의 위치를 파악하고 각 fragment를 가지고 있는 저장 노드에 fragment 서비스 요구(service request) 메시지를 보낸다. 그러면 각 저장 노드는 노드 테이블을 참조하여 fragment들을 클라이언트에 전송한다. 클라이언트는 각각의 저장 노드로부터 전송 받은 fragment들을 프락시(proxy)에서 순서대로 조합한 뒤 재생(play)한다. <그림 2>는 데이터의 전송 과정과 각 테이블의 내용을 나타낸 그림이다. 다음은 각 모듈에 대한 세부 설명을 한다.

<제어기 테이블>

화일이름	Start_num	End_num	node	node	node
a.mpg	1	30	1	2	
b.mpg	31	45	1	2	3



<노드테이블>

node1_table	1	3	5	7	9	11			

<그림 2> 데이터의 전송과정

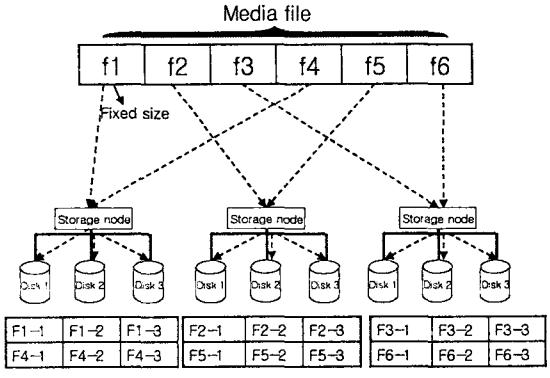
3.1 주 제어기

주 제어기는 클라이언트와 저장 노드를 이어주는 중개 역할을 한다. 먼저 클라이언트로부터 미디어 파일에 대한 요구가 들어오면 질의 프로세스(query process)는 제어기 테이블을 조사한다. 제어기 테이블에는 미디어 파일 이름, 시작 fragment 번호, 마지막 fragment 번호, 저장 노드 이름 등과 같은 정보가 들어있다. 질의가 끝나면 그 결과는 요구 프로세스(request process)에 보내어 지며, 요구 프로세스는 각 저장 노드들에게 fragment 전송 메시지를 보낸다. 여기에는 시작 fragment 번호와 마지막 fragment 번호도 함께 보낸다. 이때 클라이언트와 주 제어기, 그리고 저장 노드 사이의 메시지 전송은 신뢰성 있는 TCP/IP 프로토콜을 통해 이루어진다.

3.2 저장 노드 컴포넌트

저장 노드 컴포넌트에는 여러 개의 저장 노드가 있다. 여기에 새로운 저장 노드의 추가 또한 가능하다. 각각의 저장 노드는 SCSI 버스로 연결된 여러 개의 디스크로 이루어진 독립적인 서버이며 식별 가능한 고유 이름을 가지고 있다. 주요 기능으로는 자신에게 속해있는 디스크들의 파일 시스템 관리와 스케줄링, 파일 엑세스 컨트롤 등을 관리하는 역할을 맡는다.

저장 노드들에는 미디어 파일의 fragment들이 분산되어 저장되어 있다. 각각의 fragment는 미디어 파일을 동일한 크기로 잘라 저장하는 공간 스트라이핑 정책을 이용하여 저장 노드에 순차적으로 할당된다. <그림 3>은 fragment의 저장 과정을 나타낸다.



<그림 3> fragment의 저장과정

이때 하나 하나의 fragment에는 fragment 번호가 붙여지고 이 정보는 주 제어기에 있는 제어기 테이블에 기록된다. 각각의 저장 노드는 이 fragment를 다시 자신이 가지고 있는 디스크 수만큼 잘라 여러 개의 디스크에 스트라이핑 한다. 이때 저장 노드는 노드 테이블에 자신이 가지고 있는 fragment의 번호를 저장해 둔다.

저장 노드는 주 제어기의 요구 프로세스에서 보낸 fragment 요청 메시지를 받으면 노드 테이블을 이용하여 자신이 가지고 있는 fragment에 대해 각각의 디스크에서 sub-fragment를 가져온

뒤 노드 테이블을 참조해서 하나의 fragment단위로 조합한 뒤 RTP 프로토콜을 통해 클라이언트에 직접 전달한다.

3.3 클라이언트

클라이언트는 자신이 요청한 미디어 파일을 여러 개의 저장 노드에서 보낸 fragment들로 받는다. fragment들은 각 저장 노드마다 독자적으로 ATM 네트워크 망을 통해 클라이언트에 들어오므로 순서가 순차적이지 못하다. 따라서 클라이언트에는 fragment들을 재순서화 하고 하나의 파일로 합치는 기능을 가진 모듈인 프락시가 존재한다.

프락시를 클라이언트 안에 설계하면 네트워크 트래픽(traffic)을 줄일 수 있고 서버 또는 프락시의 다운 시에도 해당 사용자를 제외하고는 서비스를 받을 수 있으며 프락시의 구현이 간단해 진다는 장점이 있다. 또 버퍼를 프락시에 두어 버퍼가 서버에 있을 경우 별도의 메모리를 추가로 필요로 하는 문제를 해결할 수 있다[2].

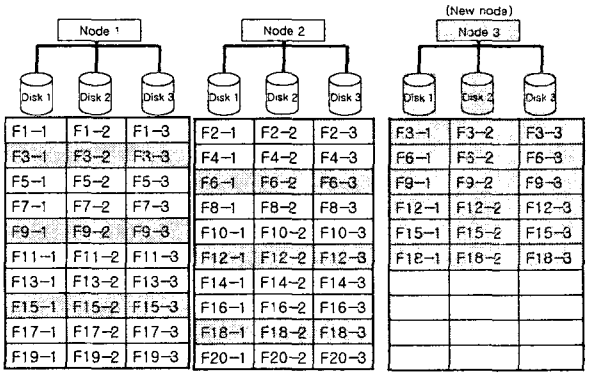
4. 새로운 저장 노드의 추가

일반적으로 보통 VOD서비스에서 새로운 미디어 파일은 클라이언트의 선호도가 높다. 병렬 VOD서버는 유한개의 디스크를 보유하고 있어 멀티미디어 파일이 증가하게 되면 공간 부족 현상이 발생하게 된다. 따라서 새로운 노드를 추가하게 된다. 그 후에 미디어 파일이 들어오는 경우 기존의 저장 노드는 이미 데이터가 꽉 차 있어 새로운 파일의 fragment들을 각각의 저장 노드에 고르게 분산시킬 수 없게 된다. 결국 새로운 파일을 fragment 단위로 나누었다 하더라도 모두 새로운 저장 노드에 쓰여지게 되므로 하나의 미디어 파일을 전부 특정 저장 노드에 저장시키는 것과 같게 된다. 이러한 기존의 노드 추가 방법은 이미 저장되어 있는 미디어 파일들의 각 저장 노드 위치를 재조정하지 않아서 특정 저장 노드에서 발생할 수 있는 부하를 방지하지 못하고 있다. 즉 새로운 미디어 파일이 저장되는 저장 노드에 부하가 발생할 수 있다. 그러므로 미디어 파일의 위치 조정이 필요하다.

이 절에서는 새로운 노드를 추가할 때 각 노드의 부하를 골고루 나눌 수 있는 방안에 대해 제시한다.

4.1 Fragment의 이동

새로운 저장 노드의 디스크에는 아무런 데이터도 들어있지 않기 때문에 다른 노드들과 부하 균등을 맞추기 위해 기존의 fragment들을 일정 부분 새로운 디스크로 옮겨주어야 한다.



<그림 4> 저장 노드의 추가 시 fragment이동 방법

이때에 연속된 각 fragment들은 가능한 서로 다른 노드에 저장하여 클라이언트가 미디어 파일을 요구할 경우 각 저장 노드는 동시에 fragment들을 클라이언트로 보내줄 수 있어야 한다. <그림 4>는 2개의 저장 노드를 가지고 있던 병렬 VOD 서버에 새로운 저장 노드를 추가한 뒤 특정 fragment들의 위치를 이동시켜 주는 방법을 나타낸다.

노드추가 횟수	시작노드	방향
1차 추가	1번 노드	오른쪽
2차 추가	2번 노드	왼 쪽
3차 추가	3번 노드	오른쪽
4차 추가	4번 노드	왼 쪽
:	:	:
:	:	:

<그림 5> 이동시킬 framgnet의 시작노드와 방향결정

세부적인 fragment이동 방법은 <그림 5>와 같다. 먼저 새 저장 노드가 추가되는 횟수에 따라 이동시킬 fragment의 시작노드와 진행 방향을 결정한다. Fragment의 시작노드와 진행방향이 결정되면 시작노드를 기준으로 하여 나머지 노드들의 이동 순번이 정해지게 된다. <그림 6>과 <그림 7>과 같이 각각의 이동 방향에 따라 한 노드 L의 순번이 정해진다.

$$\text{if}(S_n \le L \le n)$$
$$L = i - (n - 2)$$
$$\text{else}(1 \le L \le S_n - 1)$$
$$L = i + (n+1) - S_n$$

S_n : 시작노드 번호

n : 노드 갯수

<그림6> i차 추가가 오른쪽 방향일 때 한 노드 L의 순번

$$\text{if}(1 \le L \le S_n)$$
$$L = (S_n + 1) - i$$
$$\text{else}(S_{n+1} \le L \le n)$$
$$L = (n - i) + (S_n + 1)$$

S_n : 시작노드 번호

n : 노드 갯수

<그림 7> i차 추가가 왼쪽 방향일 때 한 노드 L의 순번

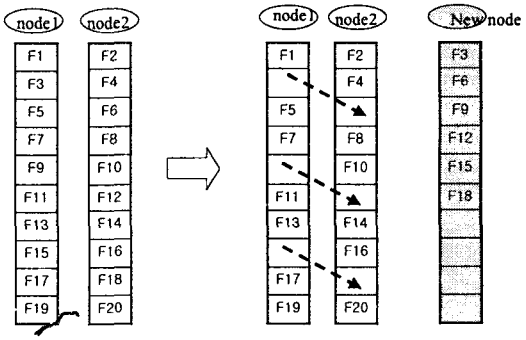
마지막으로 I번째 새로운 노드의 추가시 한 노드가 k번의 순번을 가진다면 한 노드에서 꺼내는 fragment의 위치는 <그림 8>의 식을 이용하여 정해진다.

$$(k + 1) + (n + 1)j \quad (j = 0 \ldots m)$$

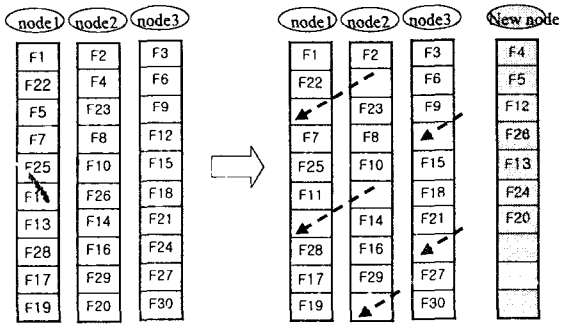
n : 노드의 갯수

<그림 8> 한 노드 k에서 꺼내어지는 fragment의 위치

주 제어기는 위의 수식을 이용하여 이동시킬 fragment의 위치를 결정한 뒤 각 저장 노드로부터 fragment를 가져와서 새 저장 노드에 써 준다. <그림 9>와 <그림 10>은 각각의 진행 방향에 따른 fragment의 이동 과정을 나타내 주고 있다. fragment의 이동과 동시에 제어기 테이블과 노드 테이블의 정보가 수정되고 새로운 저장 노드의 노드 테이블이 생겨난다. 새로운 미디어 파일이 들어오면 각 저장 노드의 빈 공간에 처음과 같은 방법으로 스트라이핑 한다.



<그림 9> 홀수번째 저장 노드의 추가 시 fragment의 이동과정



<그림 10> 짝수번째 저장 노드의 추가 시 fragment의 이동과정

5. 분석 및 평가

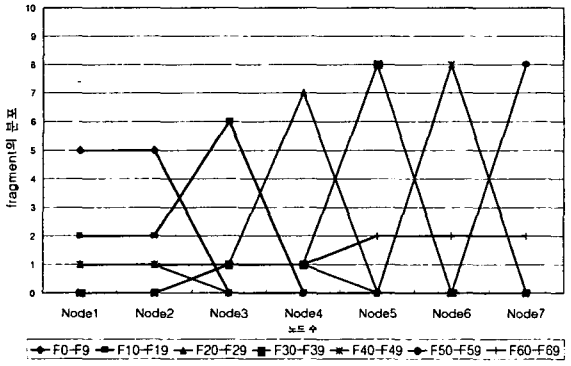
병렬 VOD 서버에서 처음 2개의 저장 노드에서 새로운 저장 노드를 추가하는 방법을 반복하여 5개 까지의 저장 노드를 추가로 확장하였다.

<표 1>은 앞에서 제시한 부하균등을 고려한 스트라이핑 정책을 이용하여 초기 2개의 저장노드에서 5개의 저장 노드를 추가한 뒤 총 7개의 저장 노드에서의 최종적인 fragment들의 저장 내용을 담고 있는 테이블이다.

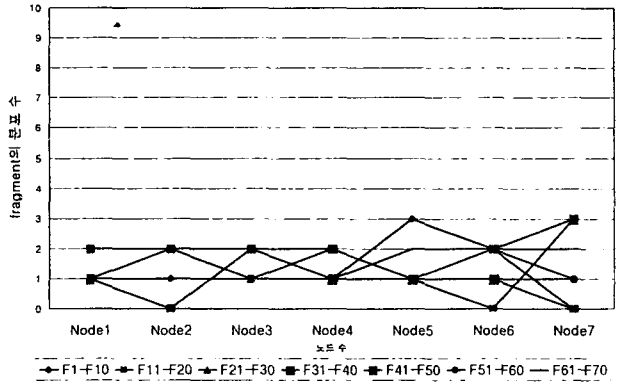
<표 1> 7개의 저장 노드상에서 각 fragment의 위치

Node 1	Node 2	Node 3	Node 4	Node 5	Node 6	Node 7
F64	F2	F3	F50	F5	F6	F7
F22	F65	F55	F9	F8	F25	F10
F42	F23	F34	F56	F15	F17	F18
F67	F43	F12	F26	F51	F11	F20
F32	F68	F44	F21	F28	F14	F4
F53	F37	F69	F45	F29	F24	F52
F13	F54	F38	F70	F30	F35	F1
F47	F16	F59	F31	F61	F46	F33
F36	F48	F27	F60	F41	F62	F63
F19	F40	F49	F39	F57	F58	F66

fragment들이 고루 분산되어 있는지 알아보기 위해, 기존의 스트라이핑 방법을 이용하여 데이터를 이동시켰을 때 fragment의 분포 그래프와 본 논문에서 제시한 개선된 방법으로 스트라이핑 시킨 fragment의 분포 수를 각각 <그림 11> 과 <그림 12>과 같은 그래프로 나타내었다.



<그림 11> 기존의 fragment 분포 그래프



<그림 12> 개선된 스트라이핑 방법을 이용한 fragment의 분포 그래프

두 개의 그래프를 비교해 보면 본 논문에서 제시한 스트라이핑 방법을 사용하여 이동시킨 fragment들은 기존의 fragment들 보다 특정 저장 노드에 집중되지 않고 대부분 고르게 분산되어 있음을 알 수 있다.

6. 결 론

VOD서버는 계속되는 미디어 파일의 증가에 대처하기 위해서 새로운 노드의 추가가 필요하게 된다. 이때에 기존 노드와 추가된 노드 사이의 부하 불균형이 발생하면 VOD서버의 기능을 제대로 수행할 수 없게 된다. 따라서 본 연구에서는 이와 같은 부하 불균형을 줄이기 위해 기존 노드의 데이터를 새로운 노드로 고루 분산 배치시키는 기술에 대해 언급하였다. 이와 같은 기술을 통해 새로운 노드의 증가에 보다 효과적으로 대처할 수 있게 된다.

참고문헌

[1] Freedman, C.S., David J.DeWitt, "The SPIFFI Scalable Video On Demand Systems," Proc. of 1995 ACM SIGMOD, pp.352-263, 1995

[2] M. M. Buddhikot and G.M. Parulkar, "Efficient Data Layout, Scheduling and Playout Control in MARS," Proc. of NOSSDAV 95, pp.318-329, 1995

[3] Jack Y.B.Lee, " Parallel Video Servers:A Tutorial," IEEE Multimedia, Vol. 5, No. 2, April June 1998

[4] W.J Boloscy, et. al., "The Tiger Video File Server," Proc. of the 6th International Workshop on Network and Operating System Support for Digital Audio and Video, pp. 97-104, 1996

[5] F. A. Tobagi, J.Pang.R. Baird, and M. Gang, "Streaming RAID: A Disk Array Management System for Video Files," Proc. of the ACM Multimedia93, pp.

393-400, 1993

[6] M. Stonbraker, et al. "Mariposa: A Wide-Area Distributed Database System," VLDB Journal, Heidelberg, Germany, Springer-Verlag, pp. Vol.5, No.1, pp.48-63, 1996.

[7] 조진성, 신현식, MPEG-1 스트림의 재구성을 통한 시간적 다중해상도 비디오 재생 기법, 한국정보과학괴논문지(C) 제 4권 제 4호, pp.439-488, 1998