스트라이핑 정책을 이용한 병렬 VOD서버의 확장 기법

한주희* 최숙영** 유관종*
*충 남 대 학 교 컴 퓨 터 과 학 과
{jhhan, kjyoo}@cs.cnu.ac.kr
**우 석 대 학 교 컴 퓨 터 교 육 과
sychoi@core.woosuk.ac.kr

A Extension Technique of Parallel VOD server using Striping Policy

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

요 약

병렬 VOD서버를 구성하기 위해서는 서버의 확장성과 서버 결함의 복구 문제 등이 충분히 고려되어야 한다. 본 논문에서는 서버의 확장성을 고려한 병렬 VOD서버를 설계한다. 또 기존의 병렬 VOD서버에 새로운 서버가 추가 될 경우 부하 불균형 문제를 방지하기 위한 데이터 분배 방법를 고려한다. 이 문제를 해결하기 위해서 로드밸런스를 유지하면서, 서버에 저장되어 있는 최소한의 데이터만을 이동시킴으로써 각 서버의 사용 가능한 디스크 공간을 조정하는 스트라이핑 방법을 제시한다. 그리고 제시한 스트라이핑 방법을 모의 실험을 통해 분석한다.

1. 서론

VOD서버를 개발하기 위한 최근의 연구 초점으로는 수많은 사용자에게 실시간 연속 접근을 보장하기 위한 스트라이핑(striping) 정책, 디스크 스케줄링, 버퍼 관리 등을 들 수 있다[1,2,4]. 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, 로드밸런스(Load Balance)를 유지하기 위하여 최소한의 데이터 이동만을 요구하는 스트라이핑 정책을 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

• 프락시(Proxy)의 위치

복수개의 서버로부터 데이터 스트림을 받아서 각각의 스트림들을 재순서화 하고 하나의 파일로 합치는 작업을 하는 모듈을 프락시라 한다. 구현 방법에 따라 프락시를 서버에 두는 방법, 클라이언트에 두는 방법, 독립적으로 두는 방법이 있다. 프락시를 클라이언트 안에 설계하면 네트워크의 트래픽(traffic)을 줄일 수 있고 특정 서버 또는 프락시의 다운시에도 해당 사용자를 제외하고는 서비스를 받을 수 있으며, 프락시의 구현이 간단해지는 장점이 있다[3].

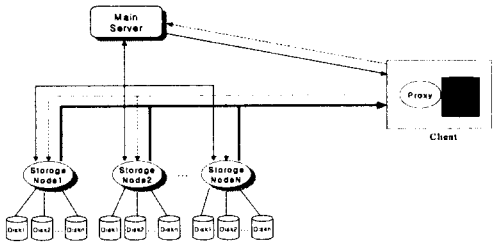
• 스트라이핑 정책

스트라이핑 정책은 여러 대의 서버에 데이터를 분산시킬 때, 기억 장소를 보다 효율적으로 사용하도록 하고 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[1,3,5].

3. 병렬 VOD 서버의 구성

본 논문에서 구성한 병렬 VOD서버의 구조는 <그림 1>

과 같다. 병렬 VOD서버는 크게 클라이언트(client), 저장 노드 컴포넌트(storage node component), 주 제어기(main controller) 모듈로 나누어진다.



<그림 1> 병렬 VOD 서버의 구성도

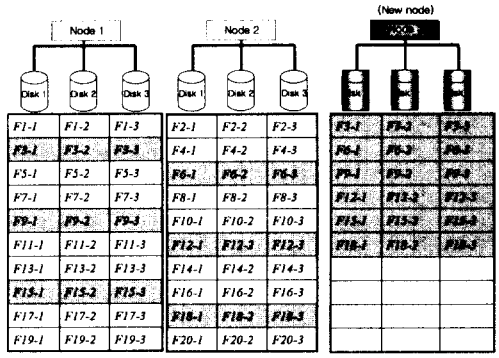
병렬 VOD서버는 특정 저장 노드에서의 부하를 줄이기 위해 데이터를 고정된 크기로 분할하여 스트라이핑 하고 이를 프래그먼트(fragment)라 정의한다.
클라이언트가 미디어 파일 정보을 요구하면 주 제어기는 제어기 테이블을 참조하여 미디어 파일의 정보를 전송한다. 클라이언트는 미디어 파일 정보를 참조하여 각 저장 노드에 서비스 요구(service request) 메시지를 보낸다. 그러면 각 저장 노드는 노드 테이블을 참조하여 해당하는 프래그먼트들을 클라이언트에 전송한다.
본 논문에서는 프락시를 클라이언트 안에 두어 저장 노드의 부하를 최소한 줄일 수 있도록 하였다. 클라이언트는 각각의 저장 노드로부터 전송 받은 데이터를 프락시(proxy)에서 순서대로 조합한 뒤에 재생(play)한다.

4. 저장 노드의 추가

연속된 각 프래그먼트들은 가능한 서로 다른 노드에 저장되어, 클라이언트가 미디어 파일을 요구할 경우, 각 저장 노드는 동시에 프래그먼트들을 클라이언트로 보내줄 수 있어야 한다. 하지만 기존의 병렬 VOD서버에 새로운 저장 노드를 추가하게 되면, 새로운 파일의 프래그먼트들은 각각의 저장 노드에 고르게 분산시킬 수 없게 된다. 따라서 로드밸런스를 유지하기 위해서는 기존의 프래그먼트들의 위치 조정이 필요하다.

4.1 프래그먼트들의 스트라이핑

대표적인 스트라이핑 방법에는 노드가 추가될 때마다 프래그먼트들을 전부 다시 스트라이핑 해 주는 방법(1)과 연속된 특정 프래그먼트들 만을 이동시키는 방법(2)이 있다. 방법(1)은 완벽한 로드밸런스를 유지할 수 있지만 매번 스트라이핑을 다시 해주어야 하는 단점이 있고, 방법(2)는 스트라이핑 방법이 간단하지만 연속된 프래그먼트들을 이동시키므로 노드가 계속 추가될수록 로드밸런스가 나빠지는 단점이 있다. 본 논문에서는 로드밸런스를 유지하면서, 최소한의 데이터만을 이동시키는 스트라이핑 방법을 제시한다.
<그림 2>는 2개의 저장 노드를 가지고 있던 병렬 VOD 서버에 새로운 저장 노드를 추가한 뒤 특정 프래그먼트들의 위치를 이동시켜 주는 스트라이핑 방법을 나타낸다.



<그림 2> 프래그먼트의 이동

프래그먼트의 스트라이핑 알고리즘은 아래와 같다.
저장 노드가 추가되는 횟수에 따라 이동시킬 프래그먼트의 시작노드와 진행 방향을 결정한다<그림 3>. 노드의 추가가 있을 때마다 프래그먼트의 이동 방향을 바꿈으로서 프래그먼트들이 최대한 고르게 선택되어 이동되도록 한다. 프래그먼트의 시작노드와 진행방향이 결정되면 시작노드를 기준으로 하여 나머지 노드들의 이동 순번이 정해지게 된다<그림 4><그림 5>. 마지막으로 n번째 새로운 노드의 추가시 한 노드가 M_i번의 순번을 가진다면 한 노드에서 꺼내는 프래그먼트의 위치는 <그림 6>의 식을 이용하여 정해진다.
프래그먼트의 이동이 끝난 뒤 새 미디어 파일이 들어오면 각 저장 노드의 빈 공간에 미디어 파일의 프래그먼트를 저장하게 된다.

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 3> 이동시킬 framgnet의 시작노드와 방향결정

기호	설명
S_n	시작노드 번호
M_i	프래그먼트 이동을 위한 노드 L의 순번
N_i	노드 L의 위치 번호
n	노드 갯수
F_i	노드 L에서의 프래그먼트 위치

<표 1> 기호 설명

$$\text{if}(S_n \le N_L \le n)\quad M_L = N_L - (S_n + 1)$$
$$\text{else}(1 \le N_L \le S_n - 1)\quad M_L = n - S_n + N_L + 1$$

<그림 4>i차 추가가 오른쪽 방향일 때
이동시킬 한 노드 M_L의 순번

$$\text{if}(1 \le N_L \le S_n)\quad M_L = (S_n + 1) - N_L$$
$$\text{else}(S_n+1 \le N_L \le n)\quad M_L = n - (N_L - S_n) + 1$$

<그림 5> i차 추가가 왼쪽 방향일 때
이동시킬 한 노드 M_L의 순번

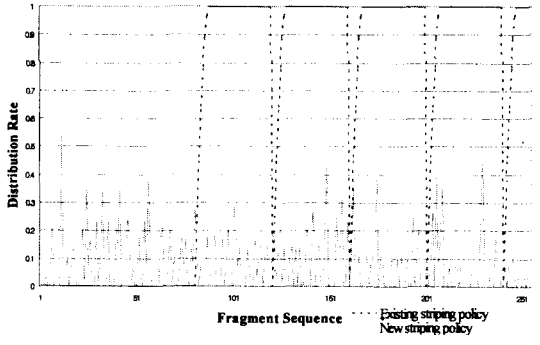
$$F_L = (\ + 1) + (n + 1)j \quad (j = 0,1, \ldots, m)$$

<그림 6> i차 추가가 왼쪽 방향일 때
이동시킬 한 노드 M_L의 순번

5. 분석 및 평가

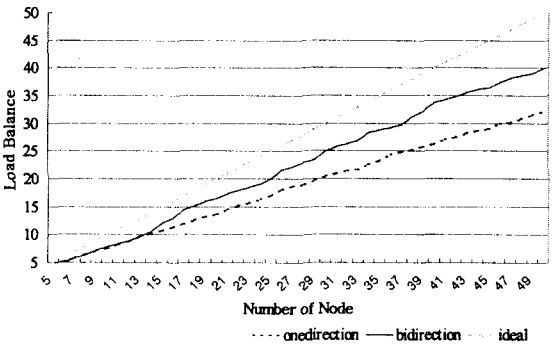
본 논문에서 제시한 스트라이핑 기법의 분석을 위하여 다음 두가지 방법의 실험을 하였다.

첫째, 논문에서 제시한 스트라이핑 기법을 이용할 때 프래그먼트들의 분산정도를 알아보았다. 2개의 저장 노드를 가진 병렬 VOD 서버에서 새로운 저장 노드를 추가하는 방법을 반복하여 5개 까지의 저장 노드를 추가로 확장하고, 이때 연속된 특정 부분만을 이동시키는 스트라이핑 방법(2)과 논문에서 제시한 스트라이핑 방법을 각각 적용하여 보았다. <그림 7>을 보면 제시한 스트라이핑 방법을 이용하면 프래그먼트들의 분포율이 기존의 방법(2)보다 고르게 퍼져있음을 알 수 있다.



<그림 7> 스트라이핑 방법에 따른 프래그먼트의 분포도

둘째, 노드의 추가에 따른 스트라이핑 방향을 한쪽으로만 수행하였을 경우와 양쪽으로 수행하였을 경우에 대한 로드밸런스 정도를 비교하였다. <그림 8>을 보면 노드가 추가될 때 마다 진행 방향을 바꾸어 스트라이핑 기법을 적용한 경우가 이상적인 로드밸런스에 가까운 것을 알 수 있다.



<그림 8> 스트라이핑 진행 방향에 따른 서버의 로드 밸런스 정도

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 로드밸런스를 유지하기 위해 기존 노드의 데이터를 새로운 노드로 고루 분산 배치시키는 스트라이핑 기술에 대해 언급하였다. 이와 같은 기술을 통해 새로운 노드의 증가에 보다 효과적으로 대처할 수 있게 되었다. 향후 과제로는 서버의 결함복구 문제를 포함한 스트라이핑 방법을 연구해야 할 것이다.

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