분산 클러스터링을 이용한 네트워크 동영상 스트리밍

박동재, 성재득, 이동수, 노영주, 최진구, 한익주
한국산업기술대학교 컴퓨터공학과
{yyyssj, sjd83, ehean77, yrho, jkchey, ijhan}@kpu.ac.kr

Dynamic Streaming Network using Distributed Clustering

Dong-Jae Park, Jae-Deuk Sung, Dong-Su Lee, Young J. Rho, Jin-Goo Choi, Ik-Joo Han
Department of Computer Engineering, Korea Polytechnic University

요 약

날이 갈수록 네트워크와 클라이언트 컴퓨터의 성능이 계속 진보되고 있다. 고성능 클라이언트의 증가함에 따라 실시간으로 고품질의 동영상 서비스를 대중적으로 요구되고 있고, 서비스 제공자는 그에 맞는 고비용의 하드웨어 장비를 구축해야 하는 어려움에 직면하고 있다. 하나의 서버에 수백, 수천 개의 클라이언트를 수용할 수 있도록 설계를 하는 경우에, 그 비용은 늘어나고, 시간과 노력 또한 많이 들게 된다. 그럼에도 불구하고, 클라이언트의 수요가 늘어남에 따라 서버의 부담은 급격히 늘어나게 되어, 서비스의 양과 질이 제한을 받게 된다. 우리는 이러한 제한을 극복하고, 서비스 제공자의 부담을 줄이고자 분산 클러스터링이라는 개념을 사용하였다. 이 방법으로 우리는 서버의 부담을 클라이언트에 나누어 주어, 서비스를 받고 있는 각 클라이언트도 서버의 기능을 가지게 하였다. 클라이언트가 다른 클라이언트에게 스트리밍 서비스를 가능하도록 한 것이다. 클라이언트에게 서버의 기능이 주어져, 스트리밍 서버의 부담은 클라이언트 수에 영향을 받지 않게 된다. 그로 인해 서버의 비용 및 장비 면에서 이익이 기대되는 연구를 수행하였다.

1. 서 론

네트워크 성능의 지속적인 진보와 클라이언트 컴퓨터의 성능향상으로 인해, 고품질 스트리밍 서비스를 하는데 있어, 서비스 제공자는 많은 비용과 막대한 하드웨어를 구축해야 한다. 이러한 서비스 제공자의 한계와 서버의 부담을 줄이고, 서버 구축비용을 절약하는 방법이 대두되고 있다.

각 Client에게 Server의 기능을 부여함으로써, Client는 파일을 전송 받고, 동영상 재생을 하면서, 동시에 다른 Client에게 파일 전송을 해준다. 이로써 Client는 자체적으로 스트리밍 서비스를 할 수 있다. 이렇게 클라이언트의 증가에 따fms 서버의 부하 증가를 분산시킴으로써, 서버의 규모 절약 및 비용 절감의 효과를 얻을 수 있다.

2. 관련연구

2.1 분산처리

분산처리는 널리 산재해 있는 연산 자원들을 빠른 네트워크로 연결하여 사용자가 응용 프로그램을 처리하도록 하는 기법이다. 따라서 분산처리를 하기 위해서는 하나의 응용 프로그램을 몇 개의 소작업(task)로 분할하고, 이들 소작업들을 서로 다른 컴퓨터에 할당하여 동시에 수행하도록 함으로써, 필요한 작업을 빠르게 처리할 수 있다.

그러므로 분산처리에 따른 성능 이득은 작업을 많은 소작업으로 분할하고, 이들 소작업을 보다 많은 컴퓨터을 이용하여 얼마나 균형 있게 배치하는가와 컴퓨터 사이의 통신 비용을 얼마나 줄일 수 있는 가에 달려있다.

서버 시스템을 다중화하는 기법에는 크게 액티브/스탠바이 기법과 액티브/액티브(부하분산) 기법이 있다. 첫 번째의 액티브/스탠바이 기법은 서버의 장애를 대비한 기법으로, 서버의 메모리 채널을 이용해 클러스터링을 구성하는 방식이 대표적인 경우이다.

두 번째의 부하분산 기법이란 여러 대의 서버에 부하를 분산해 업무를 수행하는 기법으로, 장애대비와 부하의 분산을 목적으로 한다. 어떤 한 서버에 장애가 발생하면, 발생하는 모든 트래픽은 나머지 다른 서버들로 분산시킨다. 평상시에는 발생하는 트래픽을 서버의 상태 모니터링을 통해 모든 서버에 적절하게 분배해준다. 그리고 동영상 구현을 위해서 메모리 매핑파일을 사용할 수 있는데, 이는 주소 공간의 특정 영역을 확보하고 이에 대하여 메모리를 지정할 수 있다. 따라서 파일이 한번 매핑 되면 메모리에 적재된 파일처럼 접근하여 사용할 수 있는 기술을 말한다.

2.2 메모리 매핑

대부분의 애플리케이션은 파일을 이용한 작업을 수행하는데 항상 말썽을 부리기 쉽다. 애플리케이션이 파일을 열고, 읽고, 닫아야 할 때 또는 파일을 열고 파일의 다른 부분으로부터 읽거나 쓰기 위해 버퍼링 알고리즘을 사용해야 할 때가 있다. 마이크로소프트 윈도우즈는 이를 위해 메모리 매핑 파일이라는 기능을 제공한다.

가상 메모리와 같이 메모리 매핑 파일은 주소 공간의 특정 영역을 확보하고 이에 대하여 메모리를 지정할 수 있다. 다른 점은 물리적 기억장소가 시스템 페이징 파일이 아닌 디스크에 이미 존재하는 파일이라는 것이다. 파일이 한번 매핑 되면 메모리에 적재된 파일처럼 접근하여 사용할 수 있다.

메모리 매핑 파일을 사용하는 목적은 첫째, EXE와 DLL파일을 적재하기 위해 메모리 매핑 파일을 사용한다. 이로 인해 애플리케이션이 실행되기 위해 요구되는 페이징 파일 공간과 시간을 유지할 수 있다. 둘째, 디스크의 데이터 파일에 접근하기 위해 메모리 매핑 파일을 사용할 수 있다. 이 방법으로 파일의 내용을 버퍼링하는 I/O작업으로부터 보호받을 수 있다.

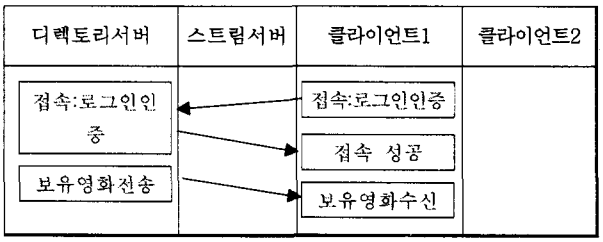
3. 시스템 구축과 구현 내용

3.1 시스템 구성

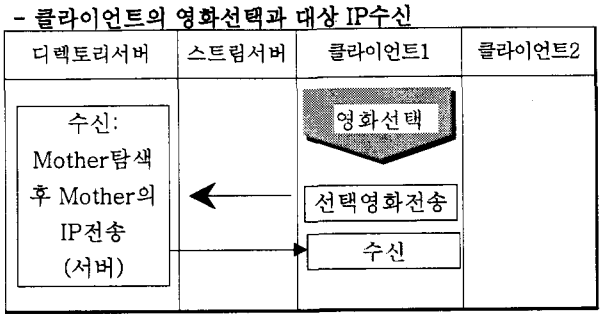
본 연구는 클라이언트 서버 환경을 기반으로 하였다. 스트림 서비스 서버, 디렉토리 컨트롤 서버와 다수의 클라이언트로 구성이 되어있다.

3.2 시스템 설계

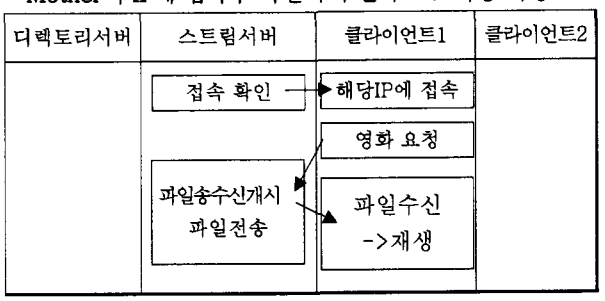
- 클라이언트 접속



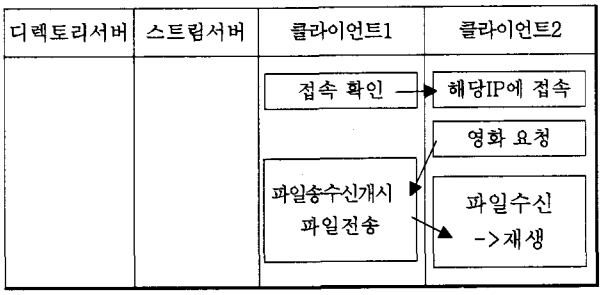
- 클라이언트의 영화선택과 대상 IP수신



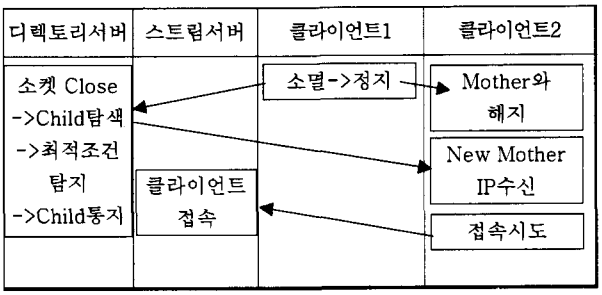
-Mother의 IP에 접속후 파일의 수신과 스트리밍 재생



- 새로운 클라이언트의 영화선택과 대상 IP 수신



- Mother 클라이언트의 소멸과 새로운 Mother부여



시스템의 기본 설계는 서버들과 클라이언트들 사이의 통신 상태의 구성이다. 디렉토리 서버, 스트림 서버, 클라이언트 1, 클라이언트2로 나누어 아래에 클라이언트의 Action을 중심으로 통신의 흐름을 나타냈다.

먼저 클라이언트1이 디렉토리 서버로부터 로그인 인증을 획득하게 되면 클라이언트1은 이용 가능한 보유 동영상의 목록을 수신한다. 클라이언트1이 특정 동영상을 더블클릭하면 해당 동영상을 보유한 최적의 스트림서버(동영상 보유서버)의 IP를 획득하고, 이 IP에서 동영상의 스트리밍 재생이 가능하다. 이 때, 또 다른 클라이언트2가 동영상의 목록에서 특정 동영상을 더블클릭했을 때도 디렉토리 서버는 최적의 Mother를 선택하여 그 IP를 클라이언트2에 넘겨준다. Mother는 스트림서버이거나 이 동영상을 전송받고 있는 클라이언트이다.

이것이 본 연구의 핵심으로, 무조건 동영상을 보유하고 있는 스트림 서버에서만 동영상을 전송하는 것이 아니라 해당 동영상을 재생하고 있는 최적의 클라이언트에 스트림 서버의 역할을 할당하여 서비스하도록 함으로써, 서버의 스트리밍 부하를 클라이언트로 온전히 분산시킬 수 있다.

만약, 클라이언트2가 클라이언트1로부터 동영상을 전송받고 있는 중에 Mother인 클라이언트1의 연결이 종료되면, 디렉토리 서버는 다시 해당 동영상의 child를 탐색하여 최적조건의 클라이언트를 클라이언트2의 Mother로 설정하게 된다. 그럼으로써, 클라이언트2는 계속해서 동영상의 스트리밍 재생이 가능하다.

3.3 접속을 관리하는 디렉토리 서버의 CTreeClass

한 개의 동영상뿐만 아니라 동영상을 상영하고 있는 클라이언트의 정보, 그리고 각 클라이언트의 접속 상태를 관리하는 객체가 디렉토리 서버이고, 핵심 클래스는 CTreeClass이다. 접속한 클라이언트에 관한 정보들 중 동영상의 상태 변화는 모두 이 CTreeClass에 접근하여 상태를 갱신받게 된다. 주요 함수로는 AddNode(), AddNode2(), ShowTree(), ShowTree3(), DeleteNode() 등이 있다.

첫째로, AddNode()함수는 CTreeClass의 동영상을 선택할 경우에 호출되는 함수로 적절한 Mother의 CClietSock 포인터를 리턴해 준다. 클라이언트의 동영상 선택시에 적절한 Mother의 IP를 찾아내는 알고리즘이 여기에 있다. CTreeClass의 맨 처음의 노드에서부터 좌, 우측의 클라이언트의 접속 상태와 각 클라이언트들의 Received Data 양의 비교로 탐색해 나가는 구조이다.

둘째로, 단순한 AddNode2() 함수는 역시 트리구조에 클라이언트를 추가해주는 함수이다. 이 함수는 Mother를 잃어버린 클라이언트가 새로운 Mother를 접속받기 위해 운영되는 함수이다.

셋째로, 클라이언트가 접속을 해지하거나 새로운 동영상을 다시 클릭할 경우, 위의 DeleteNode() 함수가 호출된다. 한 클라이언트는 자신이 서비스하고 있는 자식이 있을 수도 있고 없을 수도 있다. 위의 알고리즘상에서 CTreeClass에서 클라이언트가 삭제될 때 자신의 Child상태를 접속해 주기 위해 AddNode2() 함수를 호출해 주는 것을 알 수 있다.

넷째로, 동영상들의 접속 상태를 나타내는 표인 트리구조를 화면에 그려주는 함수는 ShowTree3()함수이다. 재귀호출함수를 이용했다. 각 트리가 화면에 투사되면 각 클라이언트의 상태를 알기위해 사용자는 마우스클릭으로 클라이언트 정보를 알 수 있다.

4. 사용 방법

4.1 주요 기능 및 실행 절차

실행절차는 일반 플레이어의 경우를 준용하고 변형하였으며, 그 절차는 다음과 같다.

① 먼저 디렉토리 서버를 실행을 시켜 소켓을 생성한다.(그림1)

② 다음으로 스트림 서버를 실행 시킨 후, (그림2)

③ SET 버튼을 눌러 디렉토리 서버로 접속 설정을 하고,

④ ADD 버튼을 눌러 동영상을 추가한다.

⑤ START 버튼을 눌러 스트림 서버를 가동한다.

⑥ 다음으로 클라이언트를 실행 후, 좌측 하단의 Set 버튼을 눌러 디렉토리 서버로의 접속 설정을 한 후, Join 버튼을 눌러 아이디와 패스워드를 입력하여 디렉토리 서버에 접속한다. (그림3)

⑦ 접속 후 오른쪽에 나오는 동영상 목록을 클릭함으로서 동영상 상영이 시작된다.

클라이언트를 구성하고 있는 기능으로는 다른 동영상 플레이어와 같이 소리조절, 음소거기능, 반복기능 등등을 포함 하고 있다.

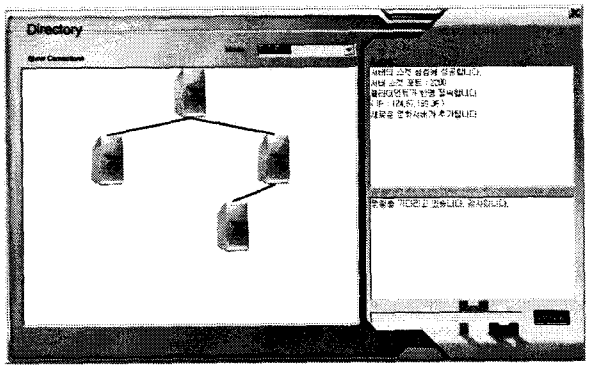


그림1. 디렉토리 서버 실행

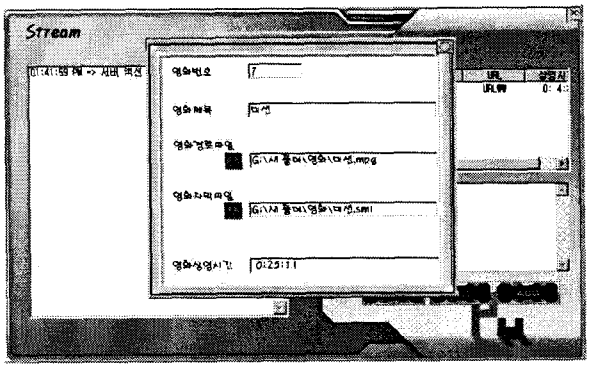


그림2. 스트림 서버 실행

그림3. 클라이언트 실행

4.2 부가기능

디렉토리 서버의 명령어 창에 아래와 같은 명령어를 입력하고 엔터를 누르면 해당 결과를 에디트 창에서 확인 할 수 있다.

- Show(show) : 접속자의 내용을 출력
- Help(help) : help에 대한 내용 출력
- Time(time) : 서버가 시작한 시간을 출력

또한 디렉토리 서버 왼쪽 화면에 보이는 클라이언트를 표현하는 컴퓨터그림을 더블클릭 하면, 각각의 클라이언트에 대한 정보를 나타내는 화면이 표시되며, 클라이언트의 ID, IP, 상영중인 동영상, CPU사용율, Ram사용율, 다운로드양 등을 표시해 준다.

클라이언트 각각의  버튼들은 재생 속도를 조절 할 수 있고, 전체화면(Full)으로도 볼 수 있는 버튼이 있다.

5. 결론

본 연구를 진행하면서 동영상 스트리밍 환경에 대한 기초연구를 진행하였고. 본 시스템을 설계하고 하나하나 구현해 나가면서 분산 클러스터링 기술의 광범위한 응용 가능성을 엿볼 수'있었다.

시스템 구현 결과 우리의 의도대로 클라이언트에서 서버의 기능을 가지게 하여서 서버의 부담을 온전히 분산할 수 있었다. 하지만 아직도 개선사항이 남아있다. 먼저 동영상 Streaming을 구현하는데 있어서 메모리 매핑파일을 사용하였는데 용량이 큰 동영상 파일일 경우에 문제가 발생할 수 있다. 그리고 우리가 테스트한 네트워크 상황은 로컬에서 이루어졌으므로 실지 인터넷 상에서 Streaming서비스가 실시된다면 클라이언트들의 통신망이 불안전할 경우 문제가 발생할 소지가 있다. 이와 같은 점을 보완하는 추가 연구개발이 이루어지면, 실용 가능한 시스템으로 현실화 시킬 수 있을 것으로 기대하고 있다.