P2P 환경에서 피어 관리 기법을 이용한 효과적인 다운로드 방법

이 인*, 황종선
*고려대학교 컴퓨터과학기술대학원
e-mail : acein@hanafos.com

Effective Downloading Method using Peer Management Technique in P2P Environments

In Lee* and Chongsun Hwang
*Graduate School of Computer Science & Technology , Korea University

요 약

최근 P2P 모델을 기반으로 한 많은 어플리케이션의 등장으로 다수의 파일서버가 필요하지 않게 되었으며 네트워크를 효율적으로 활용하여 파일 전송속도가 향상되었다. 다중 피어(Peer) 결합 방식에서는 클라이언트가 파일 전송을 요구할 경우 서버는 전송 가능한 피어들의 리스트를 클라이언트에게 보내주고 원하는 커넥션 수에 따라 연결을 맺은 후 다운로드 하게 된다. 그러나, 연결된 피어들의 네트워크 상태에 따라 전송 속도에 많은 영향을 받게 되며 어느 특정 피어에 의해 전체적인 전송속도가 저하되는 문제점이 있다. 본 논문에서는 이의 해결을 위해 교체알고리즘을 이용하여 피어들을 동적으로 접속하도록 함으로서 파일 전송 속도를 개선하였다. 제안된 교체알고리즘은 파일 전송 중 외부요인에 의해 전송속도가 저하되거나, 기 연결된 피어들 보다 더 빠른 전송속도가 예상되는 피어가 있을 경우 교체해 줌으로서 효율적으로 파일을 교환할 수 있도록 하였다.

1. 서론

최근 고품질의 서비스에 대한 수요를 만족시키기 위하여 산재 된 컴퓨터의 유휴자원을 이용하는 분산 컴퓨팅 기술을 바탕으로 하는 P2P(Peer-to-Peer)기술의 등장으로 보다 안정적이고 빠른 서비스를 즐길 수 있게 되었다. 또한 정보 제공자는 대용량의 네트워크와 다수의 중앙 파일서버가 필요하지 않게 되었다.

P2P 기술은 클라이언트가 중앙의 서버에 파일 전송을 요청하면 서버가 해당 파일에 대한 다양한 정보를 제공하는 혼합형 P2P(Hybrid P2P)와 중앙서버가 필요치 않고 피어간 직접 검색하는 순수 P2P(Pure P2P)형태로 구별된다[1]. 혼합형 P2P 에서는 중앙서버에서 파일 요청자에게 해당 파일에 대한 다양한 정보를 제공하는데 이 정보에는 해당 파일이 어떻게 분할되어 있는지, 파일 공유자의 IP 정보, 그리고 가용성 여부에 관한 정보 등이 담겨있어 이 정보를 바탕으로 클라이언트는 임의의 피어 들로부터 동시에 데이터를 전송받을 수 있다. 그리고 파일의 빠른 전송을 위해 서버는 다수의 피어에 대한 정보를 제공하는데 클라이언트는 각각의 피어들로부터 분할된 파일을 전송 받고 원래의 파일 형태로 재조립하게 된다[2,3].

그러나 최초로 피어들과 연결할 때 각 피어들의 네트워크 상태 등을 고려하지 않고 연결하기 때문에 연결되지 않은 피어들보다 전송속도가 낮을 수 있다. 또한 파일 전송도중 연결된 피어들의 네트워크 상태 및 외부 요인에 의해 전송 속도가 가변적으로 변하는 등 많은 영향을 받게 되고 특정 피어에 의해 전체적인 전송속도가 저하되는 문제점이 있다[4].

본 논문에서는 이러한 문제점을 해결하기 위해 기 연결된 피어들 중 가장 전송속도가 낮은 피어와 리스트에 존재하는 후보피어들 중 가장 빠른 피어를 일정 시간 간격으로 비교하여 비교우위에 있는 피어로 교체하여 줌으로서 전송속도의 향상을 꾀할 수 있는 기법을 제안하고자 한다. 이 제안기법은 파일전송 시 연결된 특정 피어의 네트워크 단절, 또는 급격한 네트워크 상태 변화 등으로 전송속도에 영향을 미칠 경우 주기적으로 피어들의 상태 정보를 파악하고 교체하여

일정한 전송속도를 보장하도록 하였다.

본 논문의 구성은 2 장에서 P2P 모델의 피어간 연결 과정에 대한 관련연구를 소개한다. 3 장에서는 제안된 피어 교체 알고리즘에 대한 개념을 기술하고, 4 장은 제안 기법에 대한 성능평가를 수치적으로 기술한다. 마지막으로 5 장에서는 이에 대한 결론 및 향후 연구 과제로 결론을 맺는다.

2. 관련연구

P2P 기술을 구분하는 중요한 기준은 중앙관리서버의 존재여부이다. 즉, 중앙서버가 존재하여 P2P 목록을 유지하는 것과 그렇지 않은 것의 구분이다. 전자는 중앙서버가 클라이언트의 IP 주소와 파일목록 등의 목록을 유지하면서 사용자간을 연결해 주는 역할을 하고, 후자는 모든 클라이언트가 직접 접속해 릴레이 방식으로 서로의 정보를 중개한다. 전자를 통상 하이브리드 P2P(Hybrid P2P) 라고 하고, 이에 대하여 후자를 순수 P2P(Pure P2P) 라고 한다.

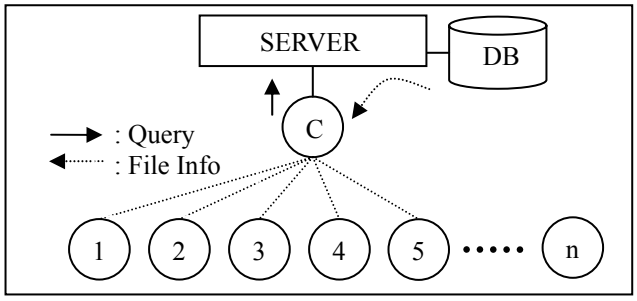
2.1 hybrid 방식

eDonkey 는 대표적인 하이브리드 P2P 방식을 사용하고 있다. eDonkey 를 실행하면 eDonkey 서버에 로그인하는 절차를 거친다. 또한 생성된 공유폴더에서 파일을 검색하여 그 목록을 중앙서버에게 보낸다. 서버는 이 목록을 검색에 용이한 구조로 데이터베이스화해서 저장하며, 이용자가 새로 접속할 때마다 목록을 갱신한다[5].

이후에 클라이언트로부터 검색요구가 들어오면, 저장된 리스트로부터 검색한 결과와 그 목록을 제공하는 피어의 접속여부, 회선속도 등을 바탕으로 원하는 파일 목록을 제공한다. 즉, 파일이 서버에는 없지만, 이용자 각각의 사이트에 어떤 파일이 있는지를 관리하면서 원하는 파일의 위치정보를 제공해 주는 것이다. 그러나 실제로 파일을 다운로드 할 때에는 중앙서버를 거치지 않고, 파일을 갖고 있는 여러 사용자의 PC 에 직접 연결하게 된다. 또한 한 개의 파일을 한곳에서만 받는 것이 아니라 서버에 등록한 특정파일을 그 서버에 속한 사람들 중에서 찾아 그 파일의 일부분을 나눠서 동시에 받을 수 있는 다중 분할 다운로드 방식을 지원하므로 전송속도를 극대화할 수 있다[2]. 그러나 피어들은 개인 PC 의 사양과 네트워크 상태가 다양하여 전송속도가 균일하지 않은 특징이 있다. 연결된 피어중 전송속도가 가장 낮은 피어로부터 전송이 완료될 때까지 기다린 후 원래의 파일 형태로 재 조립하게 된다[5].

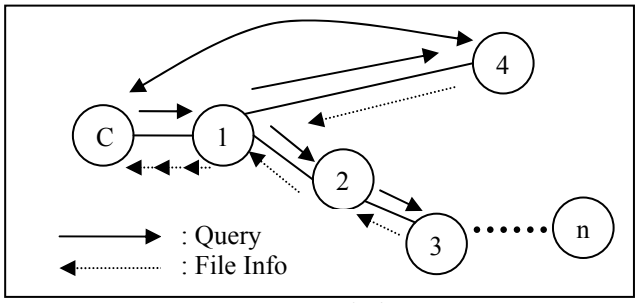
2.2 pure 방식

소리바다는 대표적인 pure 방식의 P2P 를 사용하고 있다. 중앙서버가 필요하지 않게 구현하는 방식으로 hybrid 방식과는 다르게 관리자나 운영주체는 존재하지 않는다. 따라서 검색엔진이 개별 클라이언트들이 구동한 프로그램에 설치되어 있어서 해당 컴퓨터에 직접 연결하여 자료를 찾아낸다. P2P 프로그램을 PC



(그림 1) Hybrid 방식

에 설치하고 공유할 폴더를 설정하면 연결된 모든 이용자들은 상대방이 공유를 허락한 폴더를 검색할 수 있게 된다. 따라서 파일을 검색할 때마다 네트웍의 모든 컴퓨터에 직접 들어가 검색한다. 한 컴퓨터에 없으면 다시 그 컴퓨터에 연결된 다른 컴퓨터로 들어가 검색을 하는 작업을 반복하기 때문에 서버의 목록에서만 검색하는 hybrid 형 P2P 에 비해 과도한 트래픽을 유발하고 검색속도가 크게 떨어진다. 또한 파일이 존재하는 피어들에 대한 IP 정보, 네트워크 속도 등의 정보를 실시간으로 확인할 수 있으므로 가장 상태가 좋은 피어를 임의로 선택하여 분할되지 않은 파일단위로 다운로드 받을 수 있다[6].



(그림 2) Pure 방식

3. 제안기법

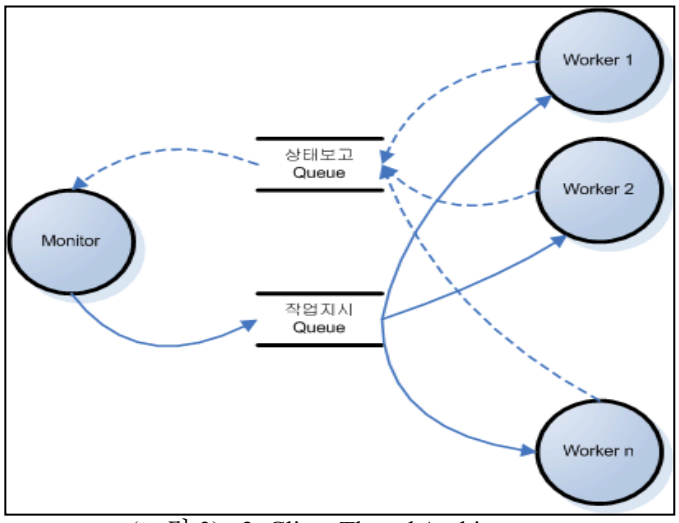
본 절에서는 기존의 eDonkey 방식과 같은 hybrid 형 P2P 네트워크 환경에서 사용자가 피어들로부터 파일을 전송 받던 중, 연결되지 않은 피어들과 속도를 비교하여 기 연결되어 전송중인 피어보다 빠를 경우 교체하여 주는 기법을 소개한다. 또한 피어 교체가 완료된 후 맨 처음으로 다운로드가 끝난 피어와 다운로드 중인 피어를 비교하여 교체하여 주는 알고리즘을 제안한다. 그리고 파일 크기와 피어의 개수에 따른 전송시간을 분석해 보았다.

3.1 기본동작

연결된 저속의 피어나 전송속도가 저하된 특정 피어로 인해 전체 다운로드 속도에 영향을 미치고 연결되지 않은 피어들 중 네트워크 속도가 빠른 피어가 있을 경우, 기 연결된 피어와의 연결을 끊고 재 접속하여 파일 이어받기를 함으로서 재 조립을 위해 부분적인 파일 전송이 끝날 때까지 기다리던 시간을 축소하여 전체적인 다운로드 시간을 단축하였다.

프로그램은 대상 파일을 상대 피어에 요청하는 클

라이언트 피어(p2pClient)와 파일을 전송해 주는 서버 피어(p2pServer)로 구성된다. 클라이언트 피어는 동시 커넥션 수 만큼 생성되어서 서버 피어에 파일을 요청하고 수신하는 워커 스레드와 이들의 상태를 감시하고 통제하는 모니터 스레드로 구성된다. 서버 피어는 클라이언트 피어의 요청에 따라 파일의 전송을 처리하는 구조로 되어있다.



(그림 3) p2pClient Thread Architecture

3.2 피어 교체 알고리즘

모니터 스레드는 피어 리스트상의 각 피어의 지연시간을 조사하여 기존 연결중인 피어보다 더 짧을경우 해당 워커 스레드에게 교체를 지시한다. 이 과정은 모든 리스트상의 피어들에 대한 조사가 끝나거나 파일 전송이 완료될 때까지 수행된다.

```
Procedure searchBestPeer {
  N = the size of compare_pool
  While1 (all total_peer_pool was not estimated) {
    Get n Recodes from total_peer_pool

    For each peer record {
      Check peer's latency
      Put it into compare_pool
    }

    while2(there is best peer to change) {
      bp = Get Best peer latency from compare_pool
      wp = Get worst peer latency from current_conn_pool
      if bp < wp
      then change wp with bp
    } // end of while2
  } // end of while1
} // end of procedure
```

(알고리즘 1)후보 피어들의 교체 알고리즘

3.3 재 선택 알고리즘

대상 파일의 사이즈가 작고 피어 리스트가 많을 경우 모든 피어들의 네트워크 상태 파악이 끝나기 전에 파일 전송이 모두 완료될 수 있다. 그러나 파일 사이즈가 큰 경우 모든 워커 스레드가 최적 상태의 피어들로 모두 교체되어서 전송하더라도 각각 지연의 차이로 인해 파일의 각 분할들은 수신이 동시에 종료되지 않으며, 이는 가용한 피어들의 개수가 적고 지연시간의 편차가 심할 경우 더욱 두드러지게 나타난다. 이러한 편차를 줄이고 먼저 전송이 완료된 우수한 피어의 재활용이 가능하도록 모니터 스레드가 워커 스레드로부터 종료상태를 보고 받고, 기존 연결중에 지연이 가장 큰 워커와 비교하고 교체하여 전송 속도를 극대화하였다.

```
Procedure checkJobDone {

  While (all job does not done) {

    Wait new job_done event from worker thread
    Ep = get job_done peer's latency
    Wp = get worst peer's latency from working conn.

    If wp > ep
    Then change worst peer with job_done peer

    Send NO_MORE_JOB to job_done worker

  } // end of while

} // end of procedure
```

(알고리즘 2) 재선택 알고리즘

4. 성능평가

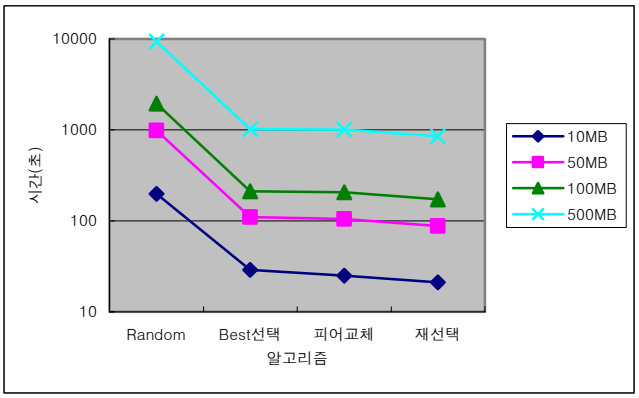
4.1 시스템 환경

파일의 분할단위는 동시에 전송하는 커넥션 수에 종속되며, 각 분할의 크기는 가변적으로 파일 크기를 커넥션 수로 나누어 결정하였다. 전송하는 피어에 대한 커넥션의 개수를 5 개, 파일을 가지고 있지만 커넥션을 맺지 않은 후보 피어의 개수를 50, 100, 200, 500 개로 설정하였고, 각 피어들의 속도는 5~100ms 의 범위에서 임의로 정하였다. 또한 전송할 파일의 크기를 각각 10MB, 50MB, 100MB, 500MB 로 설정하여 파일 크기에 따른 실험을 수행하였으며, 전송받고자 하는 파일을 보유한 피어들의 리스트를 서버로부터 받았다고 가정하였다.

4.2 구현 및 평가

본 실험에서 기준으로 삼은 2 가지 방법은 '임의의 5 개 피어로부터 전송받는 방법'과 '가장 빠르다고 판단되는 5 개 피어를 찾아낸 후에 이들로부터 전송받는 Best 선택 방법' 이며, 본 논문에서 제안하는 2 가지 방법인 '피어 교체 알고리즘만을 적용한 방법'과 '재선택 알고리즘까지 적용한 방법' 총 4 가지 방법으로 파일을 전송하는 실험을 수행하였다. 또한 각각의 방법에 따라 파일의 크기나 후보 피어의 수가 어떻게 영향을 미치는 지를 확인하기 위해 각 환경에 따른 전송시간을 측정하여 비교 평가하였다.

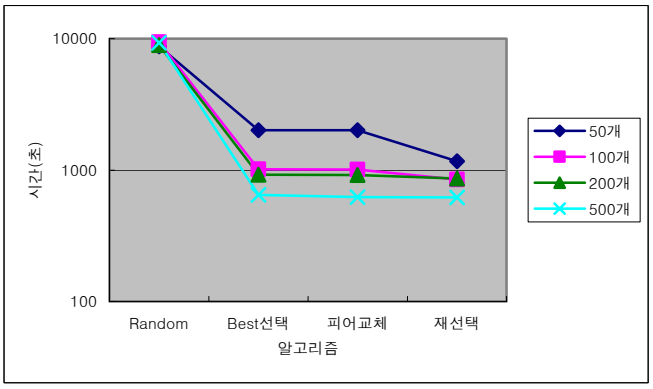
파일크기에 따른 전송속도를 실험한 결과는 표 1 에서 보이며 후보 피어 수 100 개를 기준으로 나타내었다.



(그림 4) 파일크기에 따른 전송속도 비교

위의 시뮬레이션 결과에서 보듯이 교체 알고리즘을 이용할 경우 임의의 피어로부터 연결하여 전송 받을 때 보다 전송속도에서 많은 효율을 얻었다. 파일의 크기가 작을수록 피어의 교체를 위한 오버헤드가 크기 때문에 큰 파일을 전송받을 때보다 효율이 낮음을 알 수 있다.

또한 후보 피어 개수에 따른 전송속도 비교는 표 2 에서 보이며 파일의 크기는 500MB 를 기준으로 나타내었다.



(그림 5) 피어 개수에 따른 전송속도 비교

적은 수의 후보 피어가 존재할 때 피어교체 알고리즘에 비해 재선택 알고리즘이 상대적으로 빠른 이유는 피어의 개수가 적을수록 빠른 피어의 개수가 적고, 피어 교체 알고리즘은 우수한 피어를 재활용 하지 않는 반면 재선택 알고리즘은 재사용 함으로서 속도가 향상되었는데 최고 80%의 이득을 얻었다. 그러나 피어의 개수가 많아짐에 따라 피어교체 알고리즘 과 재선택 알고리즘의 전송속도의 편차가 좁혀지는 이유는 빠른 피어의 개수가 많아지고 피어교체 완료시 동등한 레벨의 우수한 피어들과 연결되어 전송중이기 때문에 재선택을 하여도 큰 이득이 없기 때문이다.

Best 선택 알고리즘과 피어교체 알고리즘의 차이가 적은 이유는 우수한 피어를 선택하기 위하여 피어들의 상태를 파악하고 비교하는 시간이 지나면 두 알고리즘 모두 최적의 피어들로부터 전송받기 때문이다. Best 선택 알고리즘은 전체 후보피어들의 상태를 먼저 파악하고 선택하므로 그 시간만큼의 시간이 지연되고, 피어교체 알고리즘은 우선 임의의 후보피어들로부터 전송받는 동시에 나머지 후보 피어들의 상태를 파악하여 교체하여 줌으로 두 알고리즘은 그 시간차만큼의 차이를 보이게 된다.

5. 결론 및 향후 연구과제

본 논문에서는 Hybrid 방식의 P2P 에서 피어 교체기법을 제안하고 알고리즘과 동작 시나리오를 통해 이를 살펴보았다. 파일의 빠른 전송을 위하여 후보 피어들에 대한 네트워크 상태를 실시간 체크하고, 기 연결된 피어와 비교하여 보다 더 빠른 피어와 교체하여 줌으로서 전송속도를 단축하였다. 또한 후보피어들의 개수와 파일 크기에 따른 전송속도의 효율성을 비교하여 분석을 하였는데, 후보피어 집단의 크기가 크면 초기 교체과정에서 연결 수만큼 최적 피어들이 선택되기 때문에 피어교체 알고리즘과 재선택 알고리즘의 차이가 거의 없고, 피어 집단이 작으면 최적의 피어 집단이 상대적으로 적기 때문에 재선택 알고리즘의 성능이 우수함을 보였다.

향후 연구로는 전송속도가 빠른 피어에게 다중 으로 세션을 연결하고, 세션 별 전송 속도에 대한 제한을 둠으로 서 QoS(Quality of Service)를 보장할 수 있는 기법에 대하여 연구 하고자 한다.

참고문헌

[1] Andy Oram, “Peer-to-Peer:Harnessing the Power of Disruptive Technology”, O’Relly, March 2001.
[2] Sang Li, “Early Adopter JXTA”, Wrox Press Ltd., 2002
[3] 김영진,엄영익, “순수 P2P 네트워크 환경을 위한 효율적인 피어 연결 기법”, 한국정보과학회 학술발표 논문집 (I) 제 31 권, 제 1 호, p11-19, 2004
[4] 정의현,김성진, “P2P 네트워크에서의 오디오 스트리밍 시스템”, 한국정보과학회 춘계학술대회논문집(I) 제 31 권, 제 1 호, p694-696, 2004
[5] eDonkey Homepage, http://www.edonkey2000.com/documentation
[6] 소리바다 Homepage, http://www.soribada.com