효율적인 P2P 기반 파일 검색 알고리즘 설계

백승재° 안진호
경기대학교 전자계산학과
dragon976@hanmail.net°, jhahn@kyonggi.ac.kr

Design of Efficient P2P Based File Search Algorithm

SeungJae Baek° JinHo Ahn
Dept. of Computer Science, Kyonggi University

요 약

최근에 파일 공유 애플리케이션을 개발하는데 있어 클라이언트-서버(client-server) 모델의 문제점을 해결하기 위해 피어-투-피어(peer-to-peer) 모델이 각광 받고 있다. 대표적인 P2P 기반 파일 공유 시스템으로 넵스터, 그누텔라등이 있다. 그러나 이 시스템들은 각각 중앙 집중적 제어, 혹은 브로드캐스팅에 의한 과도한 네트워크 트래픽 발생 등 확장성 문제를 발생시킨다. 이런 문제점을 해결하기 위해 울트라피어 및 동적 라우팅 기법을 사용하지만 여전히 rare 파일에 대한 높은 응답시간과 검색의 낮은 신뢰성 문제점들을 해결할 수 없다. 본 논문에서는 popular 파일과 rare 파일에 대한 검색을 구분하여 popular 파일을 기존의 그누텔라 검색 방법을 사용하고 rare 파일 검색을 제안하는 새로운 DHT 알고리즘을 사용하도록 한다. 특히 제안하는 DHT 알고리즘은 기존의 DHT 알고리즘들과 달리 일반 노드가 아닌 울트라피어들만으로 구성함으로서 검색 비용, 노드 조인과 리브 비용, 핑거 테이블의 엔트리 수를 매우 줄임으로서 효과적이고 확장적이라 할 수 있다.

1. 서 론

최근에 파일 공유 애플리케이션을 개발하는데 있어 클라이언트-서버(client-server) 모델의 문제점을 해결하기 위해 피어-투-피어(peer-to-peer) 모델이 각광 받고 있다[4]. 이러한 목적으로 사용되는 대표적인 P2P 기반의 파일 공유 시스템으로 넵스터(napster)[5], 그누텔라(gnutella)[3]등이 있다. 먼저, 넵스터는 C/S와 P2P 구조가 혼합된 하이브리드(hybrid) P2P 구조를 기반으로 하여 파일 검색을 위해 중앙 집중형 디렉터리(directory) 서버가 있다. 이 시스템에 참여하고자 하는 피어는 자신이 공유할 파일의 목록을 디렉터리 서버에 등록하고, 그 디렉터리 서버는 등록된 피어들의 파일 목록을 검색하기 좋은 구조로 DB에 저장하고 피어들의 주소와 전송속도 등과 같은 정보도 함께 저장한다. 이 후에 임의의 사용자가 파일을 검색하고자 할 때 그 파일을 유지하고 있는 피어의 정보를 디렉터리 서버로부터 얻은 후 사용자는 그 피어로부터 해당 파일을 다운로드 받을 수 있다. 그러나 디렉터리 서버가 직접적인 서비스 제공을 하지 않지만 피어들이 유지하고 있는 파일 목록을 관리하기 때문에 시스템 규모가 커질수록 그 디렉터리 정보를 위한 저장소의 크기가 매우 커지게 되고 이에 따라 디렉터리 서버에 과부하 및 유지비용이 매우 커질 수 있기 때문에 확장적이지 못하다. 특히, 디렉터리 서버가 고장 나는 경우 전혀 파일 공유 및 검색이 불가능하게 된다.

넵스터의 단일 고장 지점(single point of failure) 문제점을 해결하기 위해 분산형 비구조적인 P2P 구조인 그누텔라가 있다. 그누텔라는 파일 검색을 위해 검색 요청 메시지를 자신과 이웃하고 있는 다른 피어에게 플러딩(flooding) 함으로써 전체 네트워크로 브로드캐스팅(broadcasting)된다. 그러나 TTL(Time To Live)을 사용하여 브로드캐스팅 되는 범위를 제한하지만 기본적인 파일 검색 방법이 브로드캐스팅이기 때문에 네트워크 트래픽에 대한 확장성 문제를 발생시킨다. 이러한 문제점을 피하기 위해 2가지의 방법들이 사용되고 있는데, 첫 번째 방법은 피어들을 울트라피어(ultrapeer)와 리프(leaf)로 나누어 구성하는 방법이 있다[7]. 즉 그누텔라 네트워크에서 노드들을 두 계층으로 나누어서 더 높은 수행 능력을 가진 노드를 울트라피어로 하고 그렇지 못한 노드는 리프로 나눈다. 한 울트라피어는 자신이 관리하는 리프들의 파일 목록을 유지하기 때문에 다른 피어들이 이러한 울트라피어를 통해 효율적이고 빠르게 파일을 검색할 수 있다. 두 번째 방법으로는 동적 라우팅(dynamic routing)[2]이 있다. 이 모델은 그누텔라의 고정적인 TTL을 동적으로 적용하는 것으로서 TTL을 적게 하여 검색을 하고 찾아낸 파일의 개수가 만족 할 수 있다면 검색을 종료하고 그렇지 못한 경우 TTL 값을 증가시켜 계속해서 검색한다.

분산 해시 테이블(DHT : Distributed Hash Table)[6]은 구조화된 분산 P2P 알고리즘이다. DHT는 N개의 피어로 이루어진 P2P 시스템에 참여하는 모든 피어들 중 이웃하는 일부의 피어들에 대한 라우팅 정보만을 유지하고 간단한 라우팅 알고리즘에 의해 O(logN) 크기의 피어만을 검색함으로서 모든 파일에 대한 검색이 가능하다. 그러나 P2P 클라이언트는 상당히 일시적(transient)이고 각 피어들의 성능 또한 다르다. 그리고 정확한 일치 질의(exact-match query)로 검색하는 것 보다 키워드 검색이 더 일반적이고 rare 파일에 대해 정확하고 신속한 검색이 가능하지만 대부분의 질의는 popular 파일에 대해 수행되기 때문에 기존의 비구조적인 그누텔라 보다 확장성이 떨어질 수 있다.

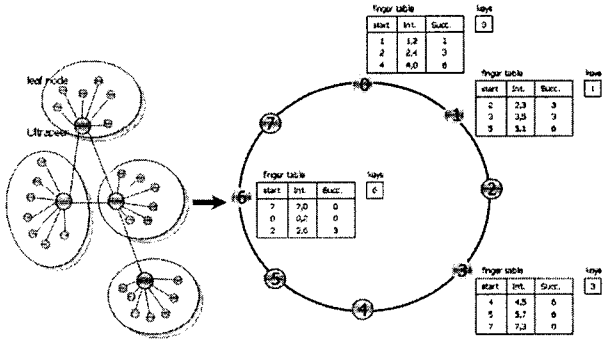
P2P 시스템 구조에서 참여하는 피어들의 이질성(heterogeneity)을 이용하는 한 연구가 있다. 이러한 연구에서는 더 높은 수용력(capacity)이 있는 노드가 더 높은 단계(degree) 노드가 되도록 한다. 또한 모든 노드는 그들의 가까운 이웃에 의해 제공되는 컨텐트의 포인터를 유지한다. 이것은 토폴로지 적합 알고리즘이 높은 수용력 노드와 높은 단계 노드 사이에서 적합성을 보증하고 1-홉 응답으로 높은 수용력 노드가 더 많은 질의에 대해 빠르게 응답할 수 있게 한다. 또한 바이어스(bias)된 랜덤 웍스(random walks)를 기반으로 보다 효율적인 검색을 할 수 있다[1].

그러나 이런 방법은 대부분의 질의가 popular 이고 rare가 아니라는 가정 하에서 설계되었기 때문에 rare 파일에 대한 검색 비용이 매우 많이 들고 어떤 경우에는 rare 파일 자체를 찾을 수 없다. 따라서 본 논문에서는 rare 파일에 대한 낮은 검

색 비용을 발생시키고 그 정확도를 매우 향상 시키는 rare 파일을 위한 DHT 기반 검색 알고리즘을 설계하고자 한다. 이 프로토콜을 rare 파일만을 대상으로 하기 때문에 기존의 DHT 알고리즘에 비해 네트워크 유지비용이 매우 적다. 즉 popular 파일은 기존의 비구조적인 방식으로 검색을 하고, rare 파일은 새로운 구조적인 DHT 기반으로 검색을 하는 것이다. 또한 기존의 DHT 검색 방법은 모든 N개의 피어들을 대상으로 그 중 이웃하는 일부의 피어들을 대상으로 검색을 하였지만 본 논문에서는 그누텔라 네트워크상의 울트라피어만을 대상으로 DHT를 구성한다. 따라서 리프를 대상으로 하는 기존의 알고리즘의 한 키(key)에 대한 검색 비용이 O(logN), 리프가 조인(join)하고 리브(leave) 할 때 $O(\log^2 N)$, 핑거 테이블(finger table)의 엔트리 수가 logN인데 비해, 본 논문에서 제안한 알고리즘은 울트라피어의 수를 n이라 할 때 각각의 비용이 O(logn), $O(\log^2 n)$, logn으로 줄어든다. 따라서 제안하는 알고리즘은 기존의 DHT에 비해 매우 효과적이고 확장적이라 할 수 있다.

2. 제안하는 DHT 알고리즘

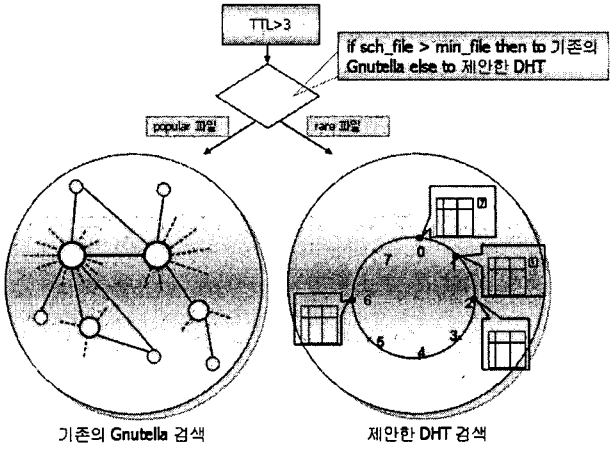
그누텔라 네트워크에서 노드의 계층을 두어 더 좋은 성능과 수용력을 가진 울트라피어는 리프가 소유한 파일들에 대한 포인터를 관리하게 되고 각 리프들은 하나의 울트라피어와 연결되어 있다. 또한 DHT에서는 모든 노드를 대상으로 구조적인 정확한 매치로 파일을 검색한다. 본 논문에서는 파일의 종류를 popular와 rare로 구분하고 그림 1과 같이 rare 파일에 대해서만 일반 노드가 아닌 울트라피어만을 대상으로 DHT를 구성한다. 이후 절에서는 이러한 특성에 따른 기존 DHT와의 변화와 리프와 울트라피어가 조인하거나 리브 할 때 각각 수행하는 프로시저에 대해 설명하고자 한다.



<그림 2> 울트라피어만으로 DHT 구성

2.1 이질적인 파일 검색

그누텔라 검색 모델의 결점 중의 하나인 popular 파일과 rare 파일에 대한 모든 검색 방법이 동일하게 취급되는 문제점을 해결하기 위해 동적 라우팅[7] 기법과 같이 단순히 TTL만을 변화하면서 검색하는 것이 아니라 본 논문에서는 rare 파일만을 대상으로 rare 파일 검색에 적합한 새로운 DHT 알고리즘으로 검색한다. 동적 라우팅에서 TTL이 3이하로 검색하였을 경우 popular나 normal 파일에 대해 충분한 검색 결과를 얻을 수 있다. 그러나 TTL이 4이상이 되면 네트워크에 너무 많은 질의 트래픽을 발생시킬 수 있다. 즉, 그림 2와 같이 TTL=3을 초과하는 경우 현재까지 검색된 파일의 수 sch_file이 요구되는 최소한의 파일의 수 min_file보다 크면 기존의 그누텔라와 같은 방법으로 검색하고, 적으면 rare 파일로 간주하여 본 논문에서 제안하는 DHT 알고리즘으로 수행하게 된다.

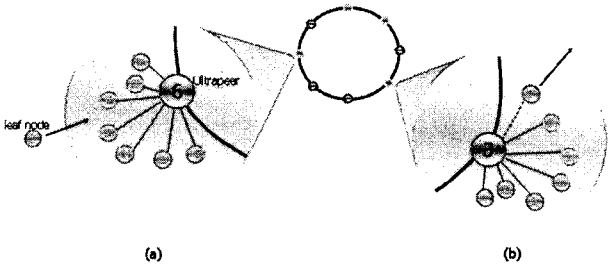


<그림 1> popular 파일과 rare 파일 구분

2.2 노드 조인/리브

2.2.1 리프

동적인 네트워크에서는 노드가 언제라도 조인하거나 리브하게 된다. 기존의 DHT에서는 노드가 조인하거나 리브 할 때 각 노드의 상속자(successor)는 유지되어야 하고, 모든 키 k를 위해 노드 successor(k)는 k에 대해 책임이 있다. 예를 들어 노드가 조인 할 때 새로운 노드의 IP 주소를 SHA-1과 같은 해쉬 함수로 해쉬하여 만들어진 m-bit의 값을 modulo 2^x 연산을 수행하여 원형 식별자 공간에 위치시킨다. 그러나 본 논문에서는 리프가 아닌 울트라피어만으로 구성되어 있고 울트라피어는 모든 리프에 인덱싱(indexing) 질의를 주기적으로 보내고 리프들은 모든 공유 파일 속성들을 전달하여, 울트라피어는 자신이 관리하는 리프에 대한 모든 인덱스 정보를 유지한다. 또한 그림 3(a)와 같이 리프가 조인 할 때 리프는 자신의 파일정보를 울트라피어에게 보낼 것이고 울트라피어는 새로운 리프의 정보에 대한 인덱스를 업데이트 할뿐 다른 전체적인 구조에는 전혀 영향을 주지 않는다. 그리고 그림 3(b)와 같이 리프가 리브 할 때에도 리브한 리프의 파일이 이웃하는 리프로 옮겨지거나 리프에 대한 정보만을 삭제하게 된다.

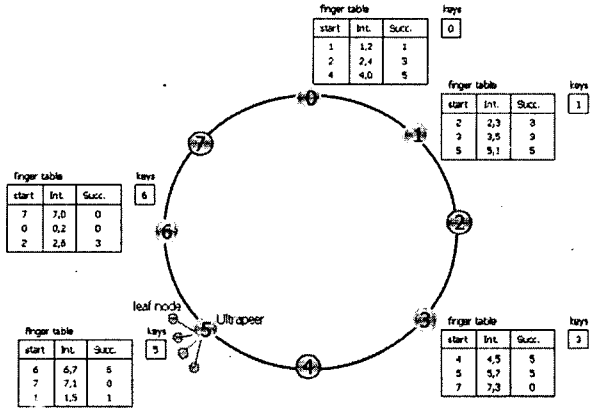


<그림 3> 리프의 조인과 리브

2.2.2 울트라피어

현재의 그누텔라 네트워크에서는 울트라피어는 어느 정도 체류를 전제로 구성된다. 그러나 울트라피어도 동적으로 조인하

거나 리브 할 수 있다. 이런 울트라피어가 조인하거나 리브 할 때의 구성을 보면 기존의 DHT방식과 다르다. 그림 4와 같이 네트워크에 새로운 울트라피어 5가 조인할 때 Chord와 같이 각 울트라피어는 전임자(predecessor) 포인터를 유지하고 전임자 포인터는 그 울트라피어의 인접한 전임자의 식별자(identifier)와 IP 어드레스를 포함하는 불변량을 보존한다. 새로운 울트라피어 n의 핑거 테이블을 초기화하고, 기존의 울트라피어에게 새로운 울트라피어 n의 정보를 전송한 후, 각 울트라피어는 책임 있는 엔트리의 정보의 포인터와 관련되는 상태를 업데이트하는 일을 수행하게 된다.



<그림 4> 울트라피어 조인

또한 울트라피어가 네트워크에서 리브할 때 기존의 DHT에서는 그 노드가 가지고 있던 키를 시계 방향의 이웃하는 노드가 그 키를 관리하도록 전송하였다. 그러나 그림 5(a)와 같이 울트라피어에는 많은 리프들이 있기 때문에 울트라피어가 리브할 경우 리프들 중 울트라피어 능력을 가진 리프가 리브하는 울트라피어를 대신하여 그 기능을 수행할 수 있다. 또한, 그림 5(b)와 같이 울트라피어의 능력을 가진 리프가 존재하지 않을 경우, 이웃하는 울트라피어로 단순히 키만을 전송하는 것이 아니라 리브하는 울트라피어가 관리하고 있던 리프까지 모두 인계하게 된다.

본 논문에서는 파일의 종류를 popular와 rare로 구분하여 각각 기존의 그누텔라 알고리즘과 새로운 DHT 알고리즘을 사용하여 검색하도록 하였다. 또한 rare 파일 검색에 대한 응답시간을 줄이고 검색에 대한 정확성을 매우 향상시키기 위해 기존의 DHT 알고리즘과 달리 제안한 새로운 DHT 알고리즘은 일반 노드가 아닌 울트라피어만으로 구성함으로서 설계 하였다. 따라서 리프를 대상으로 하는 기존의 알고리즘의 한 키)에 대한 검색 비용이 O(logN), 리프가 조인하고 리브 할 때 $O(\log^2 N)$, 핑거 테이블의 엔트리 수가 logN인데 비해, 본 논문에서 제안한 알고리즘은 울트라피어의 수를 n이라 할 때 각각의 비용이 O(logn), $O(\log^2 n)$, logn으로 줄어든다. 따라서 제안하는 알고리즘은 기존의 DHT에 비해 매우 효과적이고 확장적이라 할 수 있다.

향후, 본 논문에서 제안한 알고리즘과 기존의 그누텔라 및 DHT 알고리즘들을 시뮬레이션을 통해 성능 평가하여 본 논문에서 언급한 제안된 알고리즘의 특성을 입증하고자 한다.

참고문헌

[1] Y. Chawathe, S. Ratnasamy, L. Breslay, N. Lanham, and S. Shenker, "Making gnutella-like p2p systems scalable", In ACM SIGCOMM, Germany, IRB-TR-03-014, Aug. 2003.

[2] Adam Fisk, "Gnutella dynamic query protocol. v0.1", Gnutella Developer's Forum, May 2003.

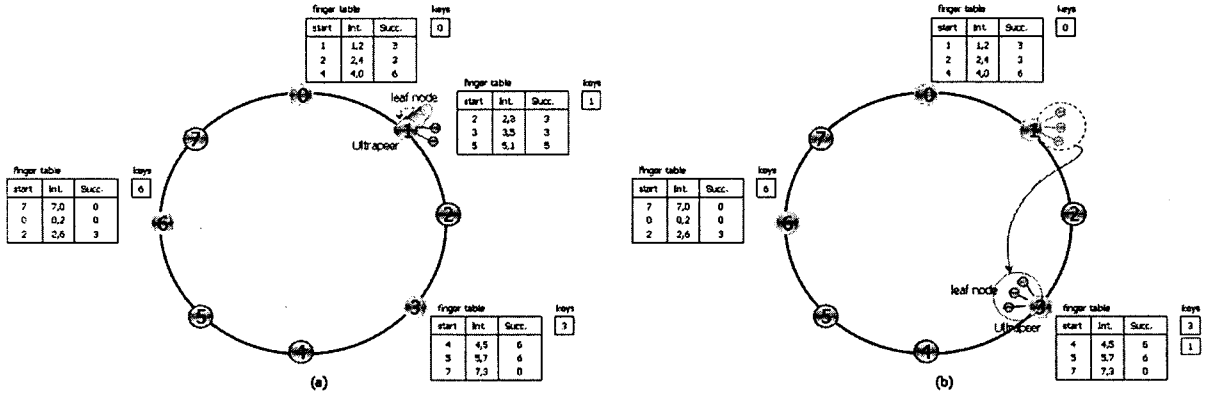
[3] Gnutella, http://rfc-gnutella.sourceforge.net/

[4] D. Milohicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, and B. Richard, "Peer-to-peer computing", Technical Report HPL-2002-57, HP Laboratories, Match 2002.

[5] Napster, http://www.napster.com/

[6] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan. "Chord: a scalable peer-to-peer lookup protocol for internet applications", IEEE/ACM Trans. Netw., 11(1):17-32, 2003.

[7] Singla, A., Rohrs, C. "Ultrapeers: Another Step Towards Gnutella Scalability", Gnutella Developer's Forum, 2002.



<그림 5> 울트라피어 리브 (a) 울트라피어를 대신할 리프가 있을 때. (b) 울트라이어를 대신할 리프가 없을 때.

3. 결 론