자기 조직적 우수 피어 링 구조를 이용한 피어-투-피어 검색기법

손 재 의[†] · 한 세 영[††] · 박 성 용[†††]

요 약

본 논문에서는 비구조적 피어-투-피어 시스템에서의 낮은 검색 성공률과 긴 검색 시간을 개선하기 위하여, 성능이 우수한 우수 피어들로 자기 조직적인 링 구조를 구성하고 광고 및 검색에 이용하는 검색 기법을 제안하였다. 우수 피어 링 구조는 동적인 환경에서 시스템의 상황에 따라 적응적으로 크기가 변하고, 에이전트를 이용하여 지속적으로 우수한 피어들이 링 구조에 참여하게 함으로써 성공률을 높이고, 빠른 검색 시간을 유지할 수 있게 하였다.

키워드 : 피어-투-피어 검색 기법, 자기 조작적인 링 구조, 우수 피어

A Peer-to-Peer Search Scheme using Self-Organizing Super Peer Ring

Jae-eui Son[†] · Saeyoung Han[††] · Sungyong Park[†††]

ABSTRACT

We propose a peer-to-peer search scheme in which super peers with high capacity constitute a ring by themselves and all peers utilize the ring for their query and publishing keys. In a dynamic peer-to-peer environment, the size of the super peer ring changes adaptively according to the changes of the system, but more powerful peers are continuously promoted to super peers. By simulations, we show that our proposed search scheme improves the query success rate of Gnutella+1hop replication search method, and maintains shorter query delay than JXTA as a static ring.

Key Words : Peer-to-peer Search Scheme, Self-organizing Ring, Super Peer

1. 서 론

피어-투-피어 시스템은 각 피어들이 가지고 있는 자원이나 서비스를 중앙 집중적인 서버의 도움 없이 이용할 수 있게 하는 시스템으로, 구조적인 피어-투-피어 시스템과 비구조적인 피어-투-피어 시스템으로 분류할 수 있다.

구조적인 피어-투-피어 시스템에서는 각 피어들이 위상을 구성하고 분산 해시 테이블(distributed hash table)을 이용하여 메시지를 라우팅 하므로, $O(\log N)$ 스텝의 빠른 검색을 하고, 찾고자 하는 대상이 시스템에 존재한다면 반드시 찾아낼 수 있음을 보장한다[11]. 그러나 특정 위상과 분산 해시 테이블을 유지하기 위한 비용이 발생하고, 특히 피어의 참여와 이탈이 빈번한 동적인 환경에서는 이 비용이 더욱 커진다. 구조적인 피어-투-피어 시스템으로는 CAN [10], Chord[11], Tapestry[12] 등이 있다.

한편, 특정 위상에 기반을 두지 않는 비구조적 피어-투-피어 시스템의 경우 위상 유지를 위한 비용은 들지 않지만, 플러딩(flooding)이나 임의 경로 검색(random walk)을 이용하여 메시지를 전송하므로 검색 속도가 $O(N)$ 스텝으로 느리고 그 성공률도 낮다[1]. 비구조적인 피어-투-피어 시스템으로는 Gnutella[2], KaZaA[13] 등이 있다.

기존의 비구조적 피어-투-피어 시스템의 긴 검색시간과 낮은 성공률을 개선하기 위하여 복제나 우수 피어를 이용하는 방법들이 연구되어 오고 있다. 그러나 복제 기법의 경우는 시스템의 규모가 크거나 동적인 상황에서 좋은 성능을 보이지 못한다. 따라서 본 논문에서는 좀 더 개선된 방법으로 우수 피어를 활용하여 비구조적 피어-투-피어 시스템의 문제점을 해결하고자 한다.

기존의 연구 결과에 따르면 비구조적 피어-투-피어 시스템의 피어들은 파워-로우 토폴로지(power-law topology)를

※ 이 논문은 2006년도 두뇌한국21사업에 의하여 지원되었음.
† 정 회 원 : 삼성전자 정보통신연구소
†† 정 회 원 : 서강대학교 컴퓨터학과 박사과정
††† 정 회 원 : 서강대학교 컴퓨터학과 부교수
논문접수 : 2006년 5월 26일, 심사완료 : 2006년 10월 26일

형성하고[5], 또한 각 피어들은 서로 다른 능력을 갖는다[9]. 본 논문에서는 이러한 연구 결과를 기반으로 하여 우수한 성능을 갖는 우수 피어들이 자기 조직적인 링 구조를 형성하고, 나머지 피어들은 자신의 키를 광고하거나 원하는 콘텐트를 검색하는데 이 우수피어 링을 이용하는 계층적 검색 기법을 제안하고자 한다.

제안된 기법에서는 각 피어들이 에이전트를 이용하여 서로의 능력에 대한 정보를 수집하고, 자신의 능력이 상대적으로 우수하다고 판단되면 자신을 우수 피어로 설정하고 링 구조에 참여한다. 이 링 구조는 피어-투-피어 시스템의 상황에 따라 적응적으로 크기가 변하므로, 링 구조를 유지하기 위한 불필요한 오버헤드를 줄이고 빠른 검색시간을 유지한다.

제안 기법의 성능을 검증하기 위하여 기존의 임의 경로 탐색 방법에 한 홉 복제기법을 이용하는 Gnutella와 고정된 우수 피어를 갖는 정적인 우수 피어 링 구조의 성능을 시뮬레이션을 통해 성공 검색 메시지 수, 평균 검색시간, 오버헤드 등의 측면에서 비교하였다. 우수피어 링을 구성하여 검색에 이용하는 경우 기존의 한홉 복제기법을 이용하는 Gnutella에 비해 링 구조를 유지하기 위한 오버헤드로 인해 약 3배정도로 메시지 수가 증가 하지만, 8배 이상 검색 성공률이 높고 검색 시간 역시 92% 이상 단축시키는 것을 볼 수 있었다. 또한 제안하는 적응적인 우수피어 링을 이용한 검색 기법이 고정된 우수피어를 이용하는 링 구조에 비하여 약 5%오버헤드가 더 많지만, 역시 90% 이상의 검색 시간 단축을 가져옴을 볼 수 있다.

본 논문은 다음과 같이 구성된다. 2장에서 비구조적 피어-투-피어 시스템의 하나인 Gnutella와 고정된 우수 피어를 이용하는 JXTA에 대해 소개하고, 3장에서 제안하고자 하는 자기 조직적 우수 피어 링 구조를 이용한 검색 기법을 자세히 소개한다. 4장에서 시뮬레이션 내용과 결과를 보이고 5장에서 결론 및 향후 과제에 대해 논의하고자 한다.

2. 관련 연구

2.1 Gnutella

Gnutella[2]는 대표적인 비구조적 피어-투-피어 시스템으로, 비록 기존의 서버/클라이언트(server/client) 검색 패러다임을 이용하지만, 모든 클라이언트들은 서버의 역할도 하고 모든 서버는 클라이언트 역할도 한다. 즉, 각 피어는 클라이언트로서 사용자의 검색을 입력 받고, 검색 결과를 보여주며, 다른 피어들에게 사용자의 질의를 전달한다. 또한 서버로서 다른 피어에서 온 질의를 넘겨받고, 질의에 해당하는 파일을 로컬 시스템에서 검색한 뒤, 질의에 대한 응답을 생성하거나 혹은 넘겨받은 질의들을 다른 피어에게 전달한다.

Gnutella는 기본적으로 검색을 하려는 피어에서 검색 메시지를 생성하여 자신의 이웃 피어 모두에게 그 메시지를 전달하는 폴러딩(flooding) 방식으로 검색을 하는데, 검색 메시지를 받은 피어는 같은 방식으로 자신의 이웃 피어에게 검색메시지를 전달한다. 이때 검색 메시지는 고유의 ID와 TTL(time to live)값을 가지고 생성된다. TTL값은 한 홉을 건널 때마다 감소되며, 이 값이 0이 되면 해당 메시지는 삭제된다. 그리고 각 노드들은 메시지에 대한 고유 ID 값을 임시로 저장하여 한번 전달한 된 Query 메시지가 다시 전달되는 것을 막고 있다. 플러딩 검색 방법은 장애에 잘 대처하고, 일정 네트워크 범위 내에서 검색의 성공을 보장한다는 장점이 있지만, 검색 양에 비해 생성되는 메시지가 많아 비효율적이고 오버헤드가 크다.

검색 메시지를 자신의 이웃 피어 모두에게 전송하는 대신 임의로 k개의 이웃 피어를 선택하여 검색 메시지를 전달하는 임의 경로 검색 방법이 제안되었는데, 이 방법은 플러딩에 비해서 검색 비용을 줄일 수 있지만 검색 성공률이 떨어진다. 따라서 검색의 효율을 높이기 위하여 임의 경로 검색 방법을 보통 여러 가지 복제 기법과 같이 쓴다[1].

대표적으로 한 홉 복제 기법(one-hop replication)이 있는데, 각 피어가 모든 이웃 피어에 자신의 모든 데이터에 대한 인덱스를 복제하는 방법이다. 데이터 대신 데이터의 인덱스를 복제함으로써 복제되는 피어의 저장 공간을 절약하고 복제할 때의 통신비용을 절약한다. 다른 피어와 연결이 되었을 때 복제가 일어나고, 주기적으로 이웃 피어들에게 자신의 데이터를 업데이트하며, 복제된 데이터를 가지고 있는 피어는 복제된 후 일정시간이 지나면 복제된 데이터를 삭제한다. 이러한 복제 방법은 고용량의 노드가 많은 이웃을 갖도록 하는 알고리즘과 함께 사용된다. 즉, 이 복제 방법의 특성상 이웃이 많으면 복제되는 데이터도 많기 때문에, 많은 이웃을 가진 노드들은 복제된 데이터를 모두 저장할만한 충분한 용량을 소유해야 한다. 특히 파워-로우 토폴로지의 특성상 허브 노드들은 많은 링크를 가지고 있기 때문에 한 홉 복제 기법을 이용할 경우 허브 노드로 데이터의 복제가 많이 일어나게 된다. 따라서 임의 경로 검색을 이용할 때 아무런 정보 없이 검색을 하는 것이 아니라 허브 노드로의 임의 검색을 하게 되면 적은 통신비용으로 검색 성공률을 높일 수 있다[1]. 본 논문에서는 한 홉 복제 기법을 사용하는 임의 경로 검색 기법을, 각 피어에서 검색 메시지를 임의의 우수 피어에게 전달하기 위한 방법으로 이용하였고, 또한 제안하는 우수 피어 링 구조를 이용한 검색기법의 성능을 우수함을 보이기 위하여 한 홉 복제 기법과 비교 분석하였다.

2.2 JXTA

JXTA[3][4]는 Sun Microsystems에 의해서 만들어진 피어-투-피어 시스템 오픈 소스 프로젝트이다. JXTA 프로토콜에서는 어떻게 광고를 검색하는지에 대한 것은 명시되어 있지 않지만, 재정의 할 수 있는 기본적인 리졸버 프로토콜들을 기본적인 정책과 함께 제공하는데, 핵심 리졸버 구조에서는 메시지를 전송, 전달 그리고 응답을 받는 기능만을 제공한다. JXTA에서는 랑데부 피어(rendezvous peer)를 이용해 기본적인 리졸버 정책을 제공한다. 즉, 일반 피어인 에

지 피어(edge peer)는 랑데부 피어를 이용하여 광고를 하고, 광고에 대한 검색을 한다.

랑데부 피어들은 느슨하게 결합한 링 구조의 네트워크를 형성한다. 기존의 분산 해시 테이블(Distributed Hash Table)에서의 가정과는 달리 JXTA에서는 랑데부 피어 그룹의 크기가 크고 안정적이지 않기 때문에, 기존의 분산 해시 테이블을 쓰게 될 경우 그 유지비용이 기하급수적으로 증가하게 된다. 하지만 분산 해시 테이블을 쓰지 않는다는 것은 심각한 네트워크의 트래픽이 있더라도 네트워크의 피어들을 이동하며 검색을 해야 하는 단점이 있다. 따라서 JXTA에서는 두 가지 방법을 혼용하여 사용하였다.

각 랑데부 피어들이 다른 랑데부 피어에 대한 정보를 피어 아이디에 대하여 정렬된 순서로 가지고 있는 것을 랑데부 피어 뷰(Rendezvous Peer View)라고 한다. 각 랑데부가 가지고 있는 랑데부 피어 뷰의 일정화를 위해 느슨한 알고리즘(A loosely-coupled algorithm)을 사용한다. 즉 랑데부들은 일시적 혹은 지속적으로 일정화 되지 않은 피어 뷰를 가지고 있을 수 있다. 주기적으로 랑데부들은 자신의 피어 뷰에서 임의의 랑데부 피어를 선택하여 자신이 알고 있는 랑데부 피어 뷰를 보내고, 또한 자신의 이웃 랑데부에게 자신이 살아있음을 알리는 메시지를 주기적으로 보낸다. 각 랑데부들은 응답이 없는 랑데부를 제거한다. 또한 기본(bootstrapping seeding) 랑데부를 이용하여 다른 랑데부의 정보를 얻어 온다.

모든 랑데부 피어들이 반드시 일정화 된 분산 해시 테이블을 가지고 있을 필요는 없다. 피어 뷰가 일정화 되었다면 이를 이용하여 최적으로 검색을 하고, 각 피어의 피어 뷰가 일정화 되지 않은 상태에서는 임의 경로 검색(random walk)를 이용하여 검색한다.

동적인 환경에서는 피어-투-피어 시스템에 피어의 참여가 많아져 검색의 요청이 많아지면 랑데부들에 부하가 커지게 되므로 많은 개수의 랑데부 피어가 필요하게 되고, 피어의 이탈이 많아져 검색의 요청이 줄어들면 검색에 비해 랑데부 피어끼리의 통신비용이 커지게 되므로 적은 개수의 랑데부 피어가 필요하게 된다. 그러나 JXTA의 경우 수동적인 랑데부 피어를 이용하기 때문에 피어-투-피어 시스템의 환경에 적응적이지 못하다. 또한, 서로 다른 능력을 가진 피어 중에서 좋은 능력을 가진 피어를 랑데부 피어로 이용하기 위한 동적인 랑데부 피어 선택 기법이 필요하다.

3. 자기 조직적 우수 피어 링 구조를 이용한 검색기법

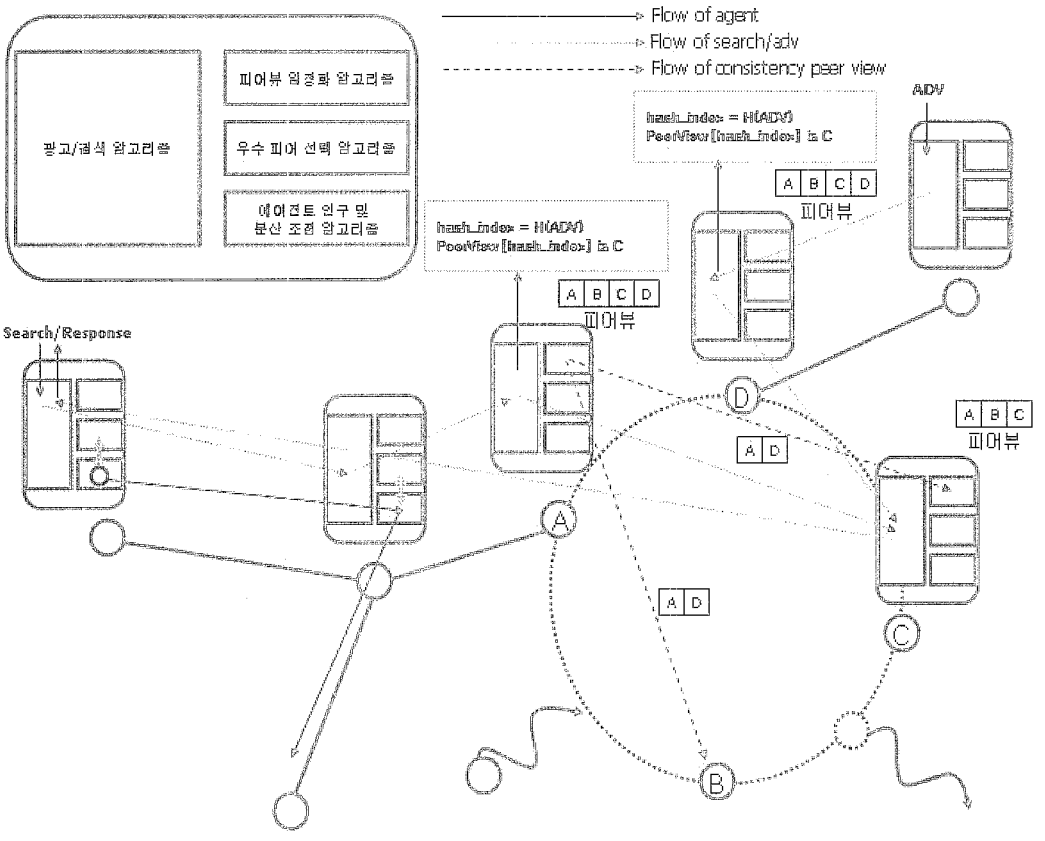
기존의 비구조적 피어-투-피어 시스템에서는 검색 메시지가 모든 피어를 대상으로 임의 경로 검색을 하기 때문에 검색시간이 길고 낮은 성공률을 보인다. 따라서 본 비구조적인 피어-투-피어 시스템에 참여하는 피어들의 능력이 다르다는 점을 착안하여 본 논문에서는 능력이 높은 피어들이 자기 조직적인 링을 구성하고 검색에 이용되도록 한다.

각 피어는 지역적 정보만을 가지고 있으므로 에이전트를 이용하여 다른 피어의 정보를 수집하고, 수집해온 정보를 이용하여 자신을 우수 피어인지를 결정하며, 우수 피어로 승격되면 우수 피어들이 형성한 링 구조에 참여하게 된다. 시스템에 계속 새로운 피어가 참여하거나 떠남이 일어남에 따라 계속 상대적으로 성능이 우수한 피어가 우수 피어로서 링을 구성하게 되므로, 우수 피어 링 구조는 시스템의 크기와 피어들의 능력 분포에 따라 적응적으로 그 크기가 변한다.

한편, 각 우수 피어는 다른 우수 피어에 대한 정보인 피어 뷰(peer view)를 다른 우수 피어들과 동기화(synchronization)하기 위해 가십 프로토콜(Gossip protocol)을 이용한다. 자신의 데이터를 광고하고자 하는 경우, 광고 메시지가 생성하고 임의 경로 검색 통해 임의의 우수 피어에게 그 메시지를 전달한다. 그 메시지를 받은 우수 피어는 해시 함수와 피어 뷰를 이용하여 해당 광고가 저장되어야 될 우수 피어를 찾아 메시지를 전달하면, 해당 우수 피어에서는 광고를 저장하게 된다. 검색 요청에 대해서도 비슷한 과정을 거치게 된다. 즉, 검색 요청을 할 피어는 검색 메시지 생성하여 임의 경로 검색을 통해 임의의 우수 피어에게 전달하고, 그 우수 피어는 역시 해시 함수와 피어 뷰를 이용하여 그 광고를 저장하고 있을 것으로 추정되는 우수 피어를 찾아 검색 메시지를 전달한다. 검색 메시지를 받은 해당 우수 피어는 자신이 저장하고 있는 정보들 중 요청한 정보가 있으면 검색을 요청한 피어에게 응답 한다. 만일 동작인 상황에 의해 피어 뷰가 일시적인 불일치 상태에 있어서 검색 요청을 받은 우수 피어가 해당 정보를 가지고 있지 않다면, 링 구조상에서 양방향의 이웃 우수 피어에게 검색 요청을 전달하는 우수 피어 경로 검색(super peer walk)이 우수 피어 TTL이 0이 될 때까지 진행된다.

위와 같은 자기 조직적인 우수 피어 링 구조를 이용한 검색을 위하여 다음과 같은 네 가지의 알고리즘이 필요하다.

- 에이전트 인구 및 분산 조절 알고리즘
  피어-투-피어 시스템에서의 각 피어는 로컬 정보만을 얻을 수 있기 때문에 에이전트를 이용하여 다른 피어들의 정보를 수집한다. 이 때 에이전트의 수가 많으면 정확한 정보를 얻어 오지만 시스템의 오버헤드가 커지고, 반대로 에이전트의 수가 적을 경우는 다른 피어의 정확한 정보를 얻기가 힘들다. 한편, 에이전트가 전체 시스템에 골고루 퍼져 있어야 전체적인 시스템의 정보를 얻을 수 있다. 따라서 에이전트의 인구 및 분산 조절 알고리즘은 피어-투-피어 시스템의 오버헤드를 줄이면서 편중되지 않은 정확한 다른 피어의 정보를 얻어오는데 목적이 있다.
- 우수 피어 선택 알고리즘
  에이전트가 수집해온 임의의 다른 피어들의 정보를 이용하여 전체 시스템의 평균을 추정하고 그 추정된 값을 이용하여 자기 자신이 우수 피어인지를 판단한다. 하지만, 추정된 평균이 정확한 전체 피어의 평균은 아니므로, 각 피어가 추정한 평균을 수렴시키는 작업도 필요하다.



(그림 1) 제안 시스템 아키텍처

◦ 피어 뷰 일정화 알고리즘
제안한 검색 기법에서는 우수 피어들이 시스템의 환경에 맞게 동적으로 변하게 된다. 따라서 각 우수 피어가 가지고 있는 피어 뷰를 일정화 시킬 필요가 있는데, 이를 위하여 기존에 연구 되어있는 가십 알고리즘을 이용하였다. 우수 피어들 간에 피어 뷰가 일정화 되어 있지 않다면, 검색 메시지가 요청된 정보를 갖는 우수 피어에게 바로 전달되지 못하고 우수 피어 경로 검색 과정을 거쳐야 하므로 검색 시간이 지연된다.

◦ 광고 및 검색 알고리즘
비구조적 피어-투-피어 시스템과 링 구조를 복합적으로 이용 할 수 있는 검색 알고리즘이 필요하다. 즉, 우수 피어를 찾을 때 까지는 임의 경로 검색을 이용하고 우수 피어에서는 해시 함수를 이용한다. 그러나 해당 우수 피어에 요청한 정보가 없으면 다시 이웃 피어에게 요청을 전달하는 우수 피어 경로 검색을 이용한다.

3.1 에이전트 인구 및 분산 조절 알고리즘

본 논문에서 제안한 우수 피어 링 구조를 이용한 검색 기법에서 에이전트는 피어-투-피어 시스템 내의 피어들의 상태를 수집하는 도구로 사용된다. 즉, 에이전트들은 피어들 사이에 이전(migration) 되면서 피어들의 정보를 수집하고, 또한 수집한 피어들의 정보를 현재 이전되어 진 피어에게 알려주는 역할을 한다. 이 피어 정보는 뒤에서 나올 우수 피어 선택의 성능에 큰 영향을 미친다. 즉, 에이전트들이 편중되어 있거나 오래된 정보만을 가지고 있다면 피어들이 정확한 판단을 하지 못한다.

이 알고리즘에서 중요한 점은 두 가지로 볼 수 있다. 에이전트의 인구 조절과 에이전트의 분산 조절이다. 즉, 에이전트의 인구가 많을 경우 피어들은 정보를 신속하게 업데이트 할 수 있지만 오버헤드가 증가 하게 되고, 인구가 적을 경우 피어들이 정보를 업데이트 할 기회가 적어지게 되어 정확한 판단을 하기 힘들다. 그리고 에이전트의 분산 조절은 같은 에이전트의 인구라도 에이전트들이 전체 시스템에 고르게 분포되게 이전되지 않고 편중 된다면, 에이전트가 가진 정보의 신빙성도 떨어지고, 에이전트가 오랫동안 이전되지 않은 피어들은 정확한 정보를 가지지 못하여, 역시 정확한 판단을 하기 어렵다.

에이전트 인구 및 분산 조절 알고리즘은 [7]에서 제안된 방법을 개선하였다. 기존의 방법에서는 한 피어에 에이전트가 이전된 시간과 이전의 에이전트가 이전된 시간의 차가 일정 시간 이내 일 경우에는 이 에이전트를 종료시키고, 일정 시간 간격 이상일 경우는 새로운 에이전트를 생성하여 또 다른 피어로 이전 시킨다. 그러나 이 알고리즘에서는 각 시간 마다 에이전트의 인구의 변화가 급격하여 인구 조절을 정교하게 할 수 없고, 또한 에이전트의 분산은 고려되지 않았다.

본 논문에서 제안 하는 에이전트 인구 및 분산 조절 알고

리즘은 크게 두 부분으로 구성되어 있다. 에이전트를 생성, 소멸하는 부분과 에이전트의 이전을 결정하는 부분이다.

에이전트를 생성, 소멸을 하는 알고리듬은 다음과 같이 동작한다. 한 피어에 에이전트가 이전되면, 이전에 에이전트가 이전되었던 시간과 현재 시간의 차를 구한다.

$$\Delta t = T_{current\ Agent} - T_{before\ Agent} \qquad \text{(식 1)}$$

이 시간 간격 Δt가 상수 α보다 작다면, 현재 이전된 에이전트를 소멸시키고, Δt가 상수 β보다 클 경우 γ의 확률로 새로운 에이전트를 생성하여 현재의 에이전트에 추가하여 서로 다른 피어로 이전시킨다. 확률을 사용하여 에이전트를 생성하는 이유는 에이전트의 인구라는 것이 전역적인 파라미터이기 때문에 각 피어의 현재의 행동이 어느 정도의 시간이 지난 후에 결과를 미치므로, 바로 에이전트를 생성하는 것이 아니라 확률적으로 생성함으로서 에이전트의 초과 생성을 방지한다.

에이전트를 이전시킬 때 어느 피어로 이전 시킬지를 결정하는 것은 다음과 같다. 각 피어에서 자신의 이웃 피어들의 정보를 주기적으로 업데이트하고, 이웃 피어들 중에서 에이전트 이전 시간 차이가 크고, 이웃의 피어의 개수가 적은 피어일수록 에이전트가 이전될 확률을 높게 한다. 에이전트 이전 시간 차이가 클수록 에이전트가 이전될 필요가 높은 피어이고, 이웃 피어의 수가 적을수록 에이전트가 이전될 기회가 적은 피어이기 때문이다.

3.2 우수 피어 선택 알고리즘

에이전트들이 이전되면서 수집해온 정보를 기반으로 각 피어들은 시스템의 평균 능력치를 추정하고 자신의 능력치와 비교하여 자신이 우수하다고 판단했을 경우는 자신을 우수 피어로 승격시킨다. 각 피어가 전체 피어의 정보를 알고 있다면 시스템 평균 능력치를 쉽게 결정할 수 있겠지만, 피어-투-피어 시스템에서는 각 피어는 지역적인 정보만을 가지므로 에이전트들이 이전되면서 수집해온 부분적인 정보들을 이용하여 평균치를 결정해야한다. 이를 위해서 다음과 같이 표본평균 $\overline{X}$을 사용하였다 한다.

$$\overline{X} = \frac{X_1 + X_2 + \ldots + X_n}{n} \qquad \text{(식 2)}$$

각 $X_1, \ldots, X_n$의 값은 전체 인구에서 임의적으로 선택된 값이어야 하고, n은 전체 인구에서 임의적으로 몇 개를 선택했는지에 대한 값이다. 이럴 경우 $\overline{X}$를 미지에 모평균 μ에 대한 표본평균이라고 한다. 이 추정은 모평균 μ에 대한 편의추정(unbiased estimator)이라고 알려져 있다.

표본평균을 구하기 위해서는 임의로 선택된 값이 필요한데, [6]에 의하면, 임의 경로 검색으로 임의의 값을 선택할 수 있다고 되어있다. 즉 에이전트들이 임의 경로 검색을 하기 때문에 에이전트에 의해서 수집된 값은 임의적으로 선택된 값이라고 할 수 있다.

따라서 각 피어는 에이전트들이 이전되면서 수집한 다른 피어들의 능력치를 이용하여 표본평균을 구하고, 자신이 구한 표본평균을 이용하여 피어 자신의 능력치가 표본평균에 비해서 상수 ρ배 이상 크다면 자신을 우수 피어로 승격시킨다.

하지만, 여기서의 문제점은 표본평균의 경우 오로지 모평균을 근사할 뿐 모평균과는 동일하지 않다는 점이다. 즉 각 피어에서 구한 표본평균의 값이 다르며 모집단의 분포에 상관없이 표본크기 n이 충분히 크면 표본평균의 분포는 정규분포 $N(\mu, \sigma^2/n)$이 된다. 따라서 우수한 능력을 갖고 있음에도 불구하고 표본평균의 값의 오차로 인해 우수 피어가 될 수 없는 경우나 우수한 능력이 아님에도 불구하고 표준평균의 오차로 인해 우수 피어로 승격되는 경우가 발생할 수 있다.

위와 같은 문제를 해결하기 위해서 각 피어는 두 종류의 업데이트를 수행한다. 첫째는 다른 피어들의 능력치로 표준평균을 구하는 것이고, 둘째는 이웃 피어와 자신의 표본평균을 이용해서 표준평균을 일정하게 맞추는 일이다. 이를 위해서 에이전트가 수집하는 정보는 각 피어의 능력치뿐만 아니라 각 피어가 계산한 표준평균도 포함한다. 즉, 피어의 표준평균을 새로운 값으로 업데이트하기 위해서는 에이전트들이 수집한 다른 피어의 능력치를 이용해 표준평균을 구하여 업데이트하고, 다른 피어들과 표준평균을 일정하게 맞추기 위해서는 이웃피어의 표준평균과 자신의 표준평균의 평균을 새로운 표준평균 값으로 사용한다.

이와 같이 구한 표준 평균을 기반으로 각 피어는 자신의 능력치와 비교하여 자신이 우수피어로 선택될지, 일반 피어로 강등되어야 하는 지를 판단하게 된다. 이를 위해 capacity_threshold_varaible을을 사용하는데, 이는 계산한 전체 시스템의 표준 평균값의 상수 ρ배에 해당하는 값으로, ρ는 우수 피어 비율에 의해 구해진다. 일반 피어의 경우 그 능력치가 지속적으로 capacity_threshold_variable 보다 커서 그 selected_count가 PROMOTION_THRESHOLD를 넘게 되면 우수 피어가 되기 위하여 참여 메시지를 보낸다. 이 메시지

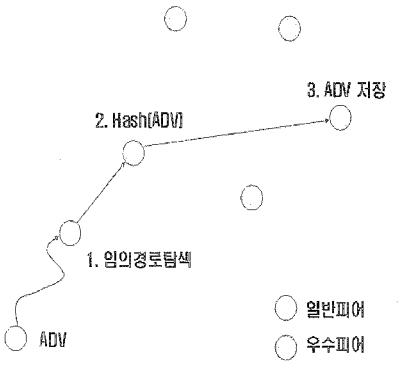
```
//At peer
if( node_situation > capacity_threshold_variable )
        selected_count++;
else
        selected_count--;
if ( node.RDV == TRUE )
        if( selected_count < DEMOTION_THRESHOLD )
                demotion();
else
        if( selected_count > PROMOTION_THRESHOLD )
                selection();
```

(그림 2) 우수피어 선택 및 강등 알고리듬

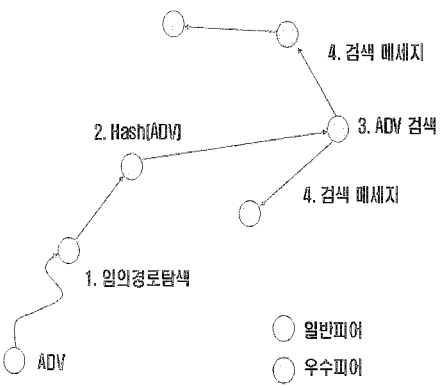
는 임의 경로 검색을 통해 우수 피어에게 전달되고, 이 메시지를 받은 우수피어는 해당 피어를 자신의 피어뷰에 추가하고 새로운 우수피어에게 자신의 피어뷰를 전달한다. 우수피어들은 가십을 통해 새로운 우수피어의 존재를 피어뷰에 추가하게 된다. 한편, 우수피어의 능력치가 지속적으로 capacity_threshold_variable보다 작아서 그 selected_count가DEMOTION_THRESHOLD보다 작아지면, 우수피어뷰와 자신이 저장하고 있는 광고메시지를 모두 삭제하고 일반 피어로 강등된다. 이때 기존의 간선을 유지하므로 새로운 간선을 만들지 않아도 된다. (그림 2)에서 알고리즘을 나타내었다.

3.3 키 광고 및 검색 알고리즘

본 논문에서 제안한 광고 프로세스는 (그림 3)과 같다. 자신의 키를 광고 하고자 하는 피어는 광고 메시지를 생성하여 임의의 이웃 피어에게 그 메시지를 전달한다. 광고 메시지를 받은 이웃 피어가 일반 피어인 경우 그 메시지를 전송한 피어를 제외하고 임의의 이웃 피어에게 다시 메시지를 전송한다. 이렇게 우수 피어에 도달할 때 까지 임의 경로 검색 방식으로 메시지를 전달한다. 광고 메시지가 일단 우수 피어에 도착하면, 모든 우수 피어가 동일하게 가지고 있는 해시 함수를 이용하여 검색 메시지가 저장될 우수 피어를 찾아 광고 메시지를 전송한다. 위의 해시 함수를 이용하는 방법은 [4]를 이용하였다. 동적인 환경에서 우수 피어들이 동적으로 생성됐다가 소멸되어 광고된 메시지가 사라질 수 있으므로 주적으로 광고 메시지를 다시 생성하여 광고해야 한다.



(그림 3) 광고 알고리즘



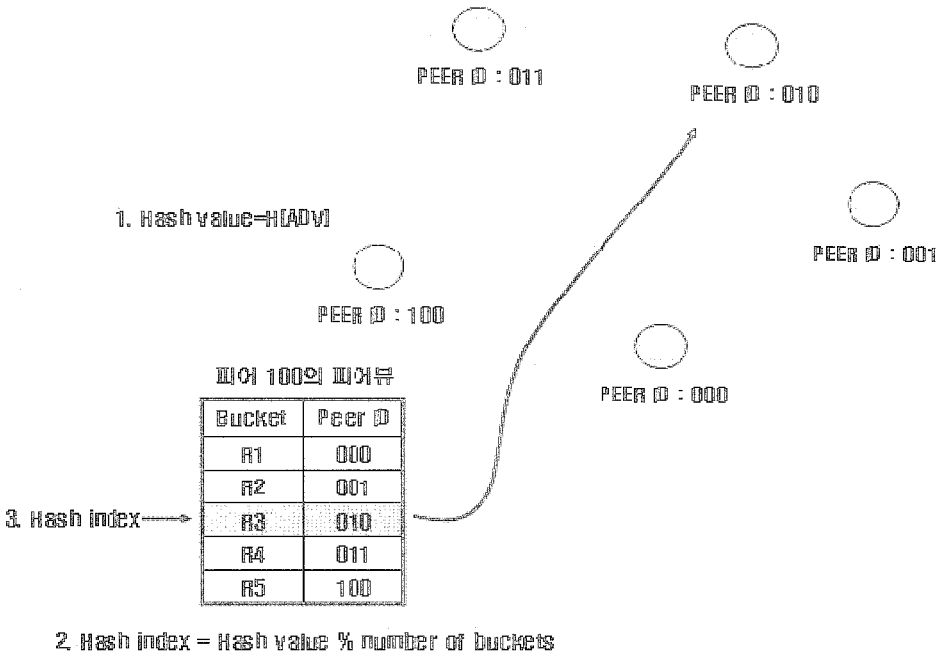
(그림 4) 검색 알고리즘

검색을 하고자 하는 피어 역시 (그림 4)와 같이 검색 메시지를 생성해서 임의의 이웃 피어에게 전송한다. 광고 메시지를 전달하는 과정과 마찬가지로, 이웃 피어가 일반 피어인 경우 임의 경로 검색 방법으로 우수피어를 찾을 때 까지 검색 메시지를 임의의 이웃 피어에게 전달한다. 검색 메시지가 우수 피어에 도착하면, 우수 피어는 역시 해시 함수를 이용하여 광고가 저장 되어있을 것으로 예측되는 우수 피어를 찾는다. 해당 우수피어가 요청된 키의 광고를 저장하고 있다면 검색 메시지가 전달되어 온 반대의 경로를 통해 키의 광고를 포함하는 응답 메시지를 전송한다. 그러나 동적인 환경에서 일시적인 피어 뷰의 불일치가 일어난 경우나 광고를 저장했던 우수 피어가 시스템을 떠나고 아직 해당 광고가 업데이트되기 전인 경우 등 해시 함수로 찾은 우수 피어에 요청된 키의 광고가 저장되어 있지 않을 수 있다. 이런 경우, 해당 광고를 저장하고 있을 것으로 추정되었던 그 우수 피어는 링 구조상에서 이웃 우수 피어들, 즉 피어뷰에서 자신의 아이디보다 큰 아이디를 갖는 우수 피어 중 가장 작은 아이디를 갖는 우수 피어와 자신보다 작은 아이디를 갖는 우수 피어 중 가장 큰 아이디를 갖는 우수피어에게 검색 메시지를 전달한다. 이런 검색 메시지를 전달 받은 우수 피어는 자신이 요청된 키의 광고를 저장하고 있는지 확인하고, 저장하고 있을 경우 응답 메시지를 보내고, 아니면 역시 링 구조상에서 자신의 이웃 우수 피어 중 메시지를 전송한 피어를 제외한 나머지 피어에게 같은 메시지를 전달한다. 이와 같은 방식으로 요청된 키의 광고를 저장한 우수 피어를 찾거나 링 경로 검색 TTL 값이 0이 될 때 까지 검색 메시지가 전송된다.

3.4 피어 뷰 일정화 알고리즘

우수 피어들이 해시 함수를 이용할 때 중요하게 사용되는 것이 다른 우수 피어에 대한 정보인 피어 뷰이고, (그림 5)에서 나타내었다. 한 우수 피어에서 피어 뷰가 정확한 정보를 가지고 있지 않을 경우 광고 및 검색 메시지를 잘못된 우수 피어로 전달하게 되어 검색 시간이 길어지고 통신비용이 증가하므로, 정확한 피어 뷰를 갖는 것이 검색 및 광고에서 중요한 역할을 한다.

각 우수 피어들의 피어 뷰의 일관성을 맞추기 위해서, 본 논문에서는 가십 알고리즘을 이용한다. 가십에 참여하는 프로세스의 개수가 N개 일 때 팬 아웃 될 대상은 최소한 $log(N)+c$ (c는 상수)개이다[8]. 따라서 각 우수 피어는 저장한 피어 뷰의 개수를 N이라 할 때, 주기적으로 자신의 피어 뷰에서 임의의 $log(N)+c$개의 피어 정보를 선택하여 가십 메시지를 만들고, 임의의 $log(N)+c$ 개의 피어를 선택하여 가십 메시지를 전달한다. 가십 메시지를 받은



(그림 5) 피어 뷰를 이용한 해싱

우수 피어는 가십 메시지에 있는 모든 정보를 자신의 피어 뷰로 업데이트 한다. (그림 6)에서 가십 알고리즘을 나타내었다.

```
size_gossip = log(size of peer view) + c;
LOOP(size_gossip) {
    select peer randomly at own peer view;
    if( selected peer is alive ) {
        make gossip message with selected peer information;}
    else {
        delete this peer information from own peer view;}
}

LOOP(size_gossip) {
    select peer randomly at own peer view;
    send gossip message to selected peer;
}
```

(그림 6) 가십 알고리듬

4. 성능평가

본 장에서는 제안한 알고리즘의 성능을 성공한 검색 메시지 개수, 검색 시간, 그리고 오버헤드의 측면에서 비조직적인 피어-투-피어 시스템인 Gnutella에 한 홉 복제 기법을 이용한 방법, 그리고 정적인 우수 피어 링 구조를 이용하는 방법과 비교하였다. 이를 위하여 이벤트-구동(event-driven) 시뮬레이터를 이용하였다.

4.1 시뮬레이션 환경

◦ 이벤트-구동 시뮬레이터

본 실험에서는 이벤트-구동 시뮬레이터를 구현하여 사용하였다. 구현한 이벤트-구동 시뮬레이터는 기존에 나와 있는 이벤트-구동 시뮬레이터들이 동작하는 방식을 따랐다. 즉, 각 시뮬레이션 타임에 시뮬레이션 되는 모든 피어들은 자신의 이벤트 큐를 확인하고 발생된 이벤트에 맞는 동작을 수행한 후 다음 이벤트를 생성하고 다음 시뮬레이션 타임으로 진행된다.

◦ 네트워크 토폴로지

비구조적인 피어-투-피어 네트워크의 토폴로지는 파워-로우 토폴로지로 알려져 있다[9]. 본 시뮬레이션에서는 동적인 환경에서 파워-로우 토폴로지를 구성하기 위하여 새로운 피어가 생성 되었을 때, 전체 피어 중에서 임의의 피어를 선택하고 한 피어가 생성할 수 있는 간선의 개수만큼 간선을 생성한다. 또한, 피어의 떠남을 위해 임의의 피어를 선택하고 그 피어에 연결된 모든 간선과 그 피어를 제거한다.

◦ 피어의 능력 분포

기존의 연구[1]에 따르면 비구조적인 피어-투-피어 시스템에 참여하는 피어들의 성능은 서로 다르다. 본 시뮬레이션에서는 중간 성능을 갖고 있는 피어가 가장 많은 부분을 차지하고, 높은 성능과 낮은 성능을 갖고 피어는 중간 성능을 갖는 피어에 비해 적은 부분을 차지하는 정상 분포를 가정하였다. 또한 피어의 능력에 따라 검색 메시지를 처리하는 시간을 다르게 두었다.

자세한 시뮬레이션을 위한 파라미터는 다음의 <표 1>과 같다.

〈표 1〉 시뮬이션 파라미터

Topology	Power-Law Topology
Alpha	2.4 - 2.6
# of node	약 30,000
Node arrival	Poisson distribution (λ=0.5)
Node departure	Poisson distribution (λ=0.5)
Node capacity distribution	Normal distribution (1000, 30)
Contents Distribution	Uniform Distribution
Agent population alpha (α)	2
Agent population beta (β)	80
Agent population gamma (γ)	0.1
Super Peer selection row (ρ)	0.01
Query generation	10 queries per every tick (> 8000)
Query processing time	0.001*exp(8000/node_capacity)
Random walk TTL	32
Super peers walk TTL	5

4.2 성능 측정 및 분석

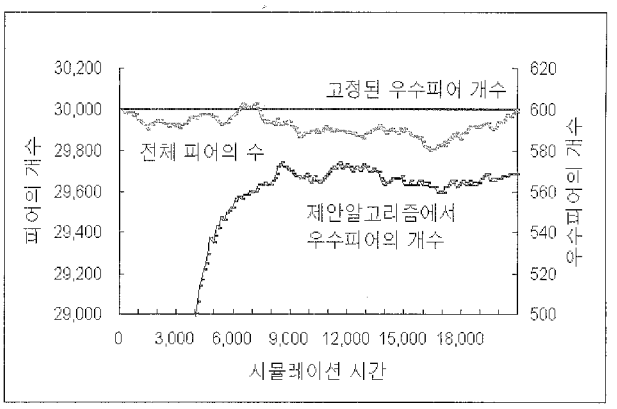
본 절에서는 동적인 환경에서 제안하는 알고리즘과 고정된 우수피어를 갖는 경우, 그리고 Gnutell +한홉복제 기법의 성능 측정 결과를 나타내었다. 성능의 측정기준은 성공한 검색 메시지 개수, 평균 검색시간 그리고 전체 시스템에서 발생한 메시지 수인 오버헤드이다.

먼저 <그림 7>에서 동적인 시스템의 피어 개수의 변화와 그에 따르는 제안 알고리듬에서의 우수 피어의 개수의 변화를 나타내었다. 피어 개수의 변화에 따라 우수 피어의 개수가 적응적으로 변함을 알 수 있다.

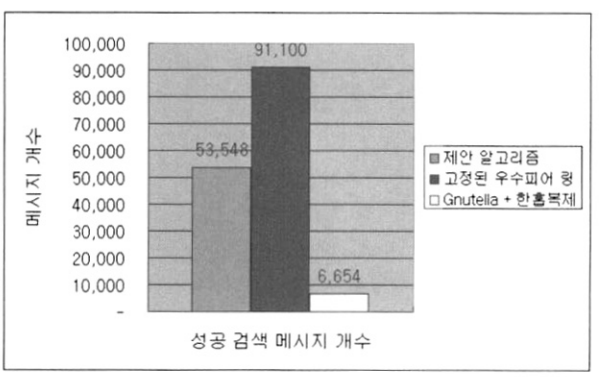
<그림 8>에서 세가지 알고리즘의 성공한 검색 메시지 개수를 비교하였는데, 제안한 알고리즘이 53,543개로 Gnutella+한홉복제의 6,654개 비해 8배 정도 많지만, 고정된 우수피어의 91,100개 보다는 40% 이상 적은 것을 볼 수 있다. 이는 고정된 링 구성의 경우 우수 피어의 변화가 전혀 없는 이상적인 상황을 시뮬레이션 환경으로 구현하여 안정적인 피어 뷰를 유지하도록 하였기 때문이다. 한편 제안하는 적응적인 우수피어 링을 갖는 구성의 경우 우수 피어도 일반 피어와 같은 비율로 시스템을 떠나도록 구현하였으므로, 실제 상황에서는 좀 더 안정정인 피어뷰를 유지하고 성공률이 높아져 두 구성 사이의 검색 성공률의 차이는 줄어들 것으로 기대할 수 있다.

한편, (그림 9)에서 보는바와 같이 평균 검색시간은 제안하는 알고리듬이 62tick으로 Gnutell+한홉복제의 786tick에 비해 92%이상 단축시키고, 고정된 우수피어링의 662 tik에 비해서도 90%이상 단축시킨다. 이는 제안하는 알고리즘에 지속적으로 상대적으로 우수한 성능을 가진 피어가 우수 피어 링을 구성하도록 하고, 이들 우수 피어들을 이용하여 메시지를 전달하므로, 검색에 걸리는 홉 수가 같더라도 그 처리 시간이 빨라, 제안하는 알고리즘에서 검색 시간이 상당히 빠르다. 한편 그러나 빠른 검색 시간을 얻기 위하여, 고정적인 링 구조와 달리 에이전트를 사용하여 시스템 내의 피어들의 능력치 정보를 수집하여 평균을 추정하는 등의 오버헤드가 가중되므로, (그림 10)에서 보는바와 같이 발생되는 메시지의 개수는 제안하는 알고리듬이 69,925,395개로 고정된 우수피어의 65,953,181개보다 5%정도 많아 오버헤드가 상대적으로 큼을 알 수 있다. 한편, Gnutella + 한홉복제 기법에서는 우수피어 링 구조를 유지하지 않으므로 오버헤드가 24,179,026개로 제안 알고리듬보다 65%정도 적은데, 그 대신 검색의 성공률과 검색 시간 측면 모드에서 링 구조를 유지하는 알고리즘들에 비해 현저히 성능이 떨어짐을 볼 수 있다.

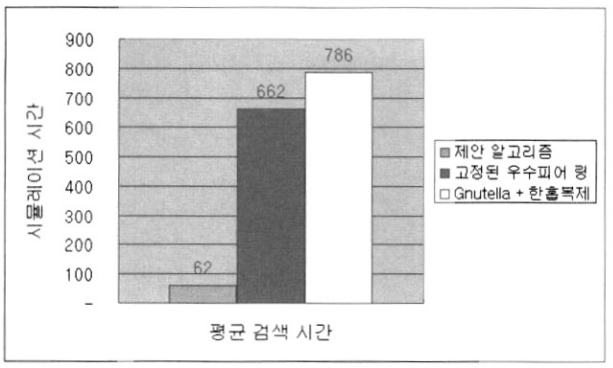
따라서 우수피어의 링 구조를 이용하여 검색을 하는 경우, 기존의 Gnutella + 한홉복제 기법에 비해 링 구조 유지를 위한 오버헤드는 증가하지만, 그 검색 성공률이나 검색 속도 면에서 월등히 성능이 향상됨을 볼 수 있었다. 그리고 우수피어의 개수를 동적인 시스템의 변화에 적응적으로 변화시키는 제안 알고리즘의 경우 지속적으로 우수한 성능을 가진 피어를 우수 피어로 유지하게 함으로써 검색 시간을 월등히 단축시킬 수 있는데, 이를 위하여 검색 성공률의 저하와 약 6% 정도의 오버헤드가 증가함을 알 수 있었다. 이때 검색 성공률의 차이 부분에 대해서는 현실적인 환경을 반영하는 좀 더 정교한 시뮬레이션이 필요하고, 그런 시뮬레이션 상에서는 차이가 상당히 적어질 것으로 예상된다.



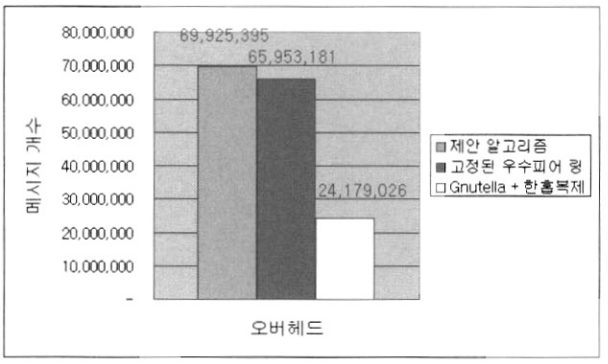
(그림 7) 피어 및 우수피어의 개수 추이



(그림 8) 성공 검색 메시지 개수



(그림 9) 평균 검색 시간



(그림 10) 오버헤드

5. 결론 및 향후 과제

본 논문에서는 비구조적 피어-투-피어 시스템에서 성능이 우수한 피어들이 스스로 자기 조직적으로 우수 피어 링을 구성하고 광고 및 검색에 참여하게 함으로써 시스템의 검색 성공률 및 검색 시간을 개선하고자 하였다.

제안 기법에서는 각 피어가 에이전트를 이용하여 지속적으로 다른 피어들의 정보를 수집하여, 그것을 기반으로 스스로가 우수 피어인지 아닌지를 판단하게 하므로, 지속적으로 피어들이 시스템을 떠나고 참여하는 동적인 상황에서 항상 성능이 우수한 피어가 우수 피어 링을 구성하게 하여 빠른 검색시간을 유지할 수 있었다. 또한 가십을 이용하여 우수 피어들의 피어 뷰를 일정화 함으로써 동적인 상황에서 우수 피어의 피어 뷰 불일치에 따르는 성능 저하를 최소화 하였다.

본 논문에서는 시스템의 평균 성능에 비해 일정 비율(ρ) 이상의 성능을 가진 피어를 우수피어로 선택하도록 하였는데, 시스템 변화의 정도에 따라 우수 피어의 비율 자체를 적응적으로 변화시킴으로써 시스템의 성능을 유지시키는 알고리즘의 개발이 더 필요하다.

참 고 문 헌

[1] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making Gnutella-like P2P Systems Scalable," Proceedings of ACM, SIGCOMM, 2003.

[2] The Gnutella Protocol Specification v0.4. http://www9.limewire.com/developer/gnutella_protocol-0.4.pdf

[3] B. Traversat, A. Arora, M. Abdelaziz, M. Duigou, C. Haywood, J.C. Hugly, E. Pouyoul, and B. Yeager, "Project JXTA 2.0 Super-Peer Virtual Network," http://www.jxta.org/project/www/docs/JXTA2.0protocols1.pdf

[4] B. Traversat, M. Abdelaziz, E. Pouyoul, "Project JXTA: A Loosely-Consistent DHT Rendezvous Walker," http://www.jxta.org/project/www/docs/jxta-dht.pdf

[5] L.A. Adamic, R.M. Lukose, A.R. Puniyani, and B.A. Huberman, "Search in Power-Law Networks," Physical Rev. E, Vol. 64, No. 4, Oct. 2001.

[6] M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani, "Estimating Aggregates on a Peer-to-Peer Network," www.cc.gatech.edu/~mihail/ D.8802readings/garciamolina2.pdf

[7] K. A. Amin and A. R. Mikler, "Dynamic Agent Population in Agent-Based Distance Vector Routing," In Proc. Second International Workshop on Intelligent Systems Design and Application (ISDA), 2002.

[8] A. J. Ganesh, A. M. Kermarrec, and L. Massoulie, "Peer-to-Peer Membership Management for Gossip-Based Protocols," IEEE Transactions on Computers. 2003.

[9] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "Measurement Study of Peer-to-Peer File Sharing Systems," Proceedings of Multimedia Computing and Networks, 2002.

[10] S. Ratnasamy, et. al, "A Scalable Content-Addressable Network," Proceedings of the 2001 Conference on Applications, Technologies, Architectures, Protocols for Computer Communications, Vol. 31, Issue 4. Aug. 2001.

[11] I. Stoica, et. al, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Vol.31, Issue 4, pp.149-160. August 2001.

[12] B. Y Zhao, J. D. Kubiatowicz, and A.D. Joseph, "Tapastry: An Infrastructure for Fault-Tolerant Wide-Area Location and Routing," U.C. Berkerley Technical Report UCB/CDS-01-1141, April, 2001.

[13] KaZaA file sharing network Homepag. http://www.kazaa.com/

손 재 의

e-mail : arison@gmail.com
2003년 서강대학교 컴퓨터학과 (공학사)
2005년 서강대학교 대학원 컴퓨터학과 (공학석사)
2005년~현재 삼성전자 재직중
관심분야 : 피어투피어 컴퓨팅, 분산처리 시스템

한 세 영

e-mail : syhan@sogang.ac.kr
1991년 포항공과대학교 수학과 (이학사)
2003년 서강대학교 정보통신대학원 (공학석사)
2004년~현재 서강대학교 컴퓨터학과 박사과정
1996년~2006년 (주)이엔지 기술본부
관심분야 : 피어투피어 컴퓨팅, 분산처리 시스템

박 성 용

e-mail : parksy@sogang.ac.kr
1987년 서강대학교 컴퓨터학과(공학사)
1994년 미국 Syracuse University 대학원 (공학석사)
1998년 미국 Syracuse University (공학박사)
1998~1999 미국 Bell Communication Research 연구원
1999~현재 서강대학교 컴퓨터학과 부교수
관심분야 : Autonomic Computing, Peer to Peer Computing, High Performance Cluster Computing and System