韓國 컴퓨터情報學會 論文誌
第11卷 第5號, 2006. 11.

2006-11-5-1-4

다중 피어 분산 처리 기반 효과적 P2P 검색 알고리즘

김 분 희*

An Effective P2P Search Algorithm based on Distributed Processing of Multiple Peers

Boon-Hee Kim*

요 약

분산 시스템은 독립적인 컴퓨터 시스템에 비해서 결함 허용 측면과 성능이 우수하여 널리 이용되고 있다. 이러한 분산 시스템 가운데 P2P 기술은 자원 공유 측면에서 활용도 높은 분야이다. 그러나 자원 공유 시 자원 제공 피어들이 검색 당시와 같이 항상 온라인 상태일 것이라는 보장이 없다. 따라서 원하는 자원을 온전히 다운로드 받기 위해서는 주로 재전송의 방법을 이용하게 된다. 이는 P2P 시스템의 성능 저하의 원인이 되므로 이에 대한 해결책이 필요하다. 본 연구에서는 자원 검색에 필요한 동료 피어를 선택하여 동반 검색 작업을 수행함으로써 검색 성능 향상 및 재전송을 위한 추가적인 검색과정을 줄여 P2P 시스템의 성능 향상 효과를 얻고자 한다.

Abstract

The distributed system is used that it's features of the fault tolerance and good performances. One of the distributed systems, a peer-to-peer tech. has high availability to share resources. However when peers wanna share resources, it isn't online like the peer using resources of the search time. Therefore we usually use the re-transmission method to download all resources. It's not good at the performance of the peer-to-peer system so we need to good solution. In this study, we wanna use the peer-to-peer system with the partner peer that works same download method simultaneously. We wanna have the effective performance to reduce the search stage additionally.

▸ Keyword : 분산 시스템(Distributed System), P2P(Peer-to-Peer), 재전송(Re-transmission)

• 제1저자 : 김분희
• 접수일 : 2006.09.22, 심사일 : 2006.10.16, 심사완료일 : 2006.11.18
* 동명대학교 멀티미디어공학과 전임강사

Ⅰ. 서 론

둘 이상의 컴퓨터 간에 네트워크를 통한 통신이 이루어지는 이산적 자원에 기반한 분산 시스템은 해당 하드웨어 및 프로그래밍 언어 등이 다양한 환경으로 구성된다[1]. 이러한 분산 시스템은 일반 컴퓨터 시스템들 간의 조합에 의해 좀더 높은 성능을 발휘하고 결함에 강한 특징을 지닌다. 분산 시스템 가운데 P2P 시스템은 여러 컴퓨터 시스템에 존재하는 다양한 자원을 공유하여 해당 자원의 활용도를 높이는 방법이다. 이러한 P2P 환경에서는 해당 자원을 효과적으로 검색하는 방법에 의해 전반적인 성능향상을 기대할 수 있다. 실제로 자원 보유 피어를 선택하는 역할의 검색 알고리즘에 대한 다양한 연구가 진행되어왔다. 이러한 검색 알고리즘들은 최소 응답시간을 보장하는 피어를 선택하거나 자원과의 거리와 관련된 표를 이용하여 거리가 가장 가까운 피어를 선택하는 알고리즘, 응답시간의 평균값을 구하여 해당 피어의 가치를 측정하고 이러한 값을 이용하여 피어를 선택하는 알고리즘, 여러 가지 데이터를 수집하여 회귀분석하고 서로 다른 피어에 메시지를 전달하기 위한 시간을 예측하여 가장 낮은 전달 시간을 가지는 피어를 선택하는 알고리즘, 네트워크 지연 시간 등을 관찰하여 응답시간이 가장 작은 피어를 선택하는 알고리즘 등이 있다. 이런 알고리즘들이 이 분야에서 주요 알고리즘으로 평가되고 있는 연구이다. 이러한 연구는 전반적으로 하나의 자원 요청 피어 당 하나의 자원 지원 피어를 할당하고 요구에 대한 서비스가 일어나는 동안 전송 실패가 일어날 가능성에 대해 고려하지 않고 있다. 그러나 P2P 분산 시스템의 가장 큰 특징은 해당 피어들이 항상 온라인 상태일 것이라는 보장이 없다는 것이고, 이로 인해 해당 자원을 보유한 피어로부터 파일을 다운로드 받다가 전송이 완전히 끝나기 전에 해당 피어가 오프라인이 되는 경우가 발생하게 된다. 따라서 원하는 자원을 온전히 다운로드 받기 위해서는 주로 재전송의 방법을 이용하게 된다. 이는 P2P 시스템의 성능 저하의 원인이 되므로 이에 대한 해결책이 필요하다.

본 연구에서는 P2P 시스템의 자원 검색 과정에 동일한 작업을 수행할 동료 피어를 선택하여, 검색 작업을 동반 수행함으로써 검색 성능을 높이고자 한다. 그리고 전송 도중 작업이 중단되어 재전송이 필요한 경우를 대비하기 위해 이와같은 동반 피어를 이용함으로써 추가적인 검색과정을 줄이는 효과를 얻고자 한다. 이는 결과적으로 해당 피어의 자원 전송률을 높이고, 알고리즘 복잡도 대비 높은 검색 성능으로 의미 있는 결과를 확인 할 수 있었다. 본 논문의 구성은 다음과 같다. 2장에서 관련연구, 3장에서는 본 논문의 핵심 부분인 제안한 검색 알고리즘을 4장에서 이에 대한 평가를, 마지막으로 5장에서 결론을 맺는다.

Ⅱ. 관련 연구

이 장에서는 본 논문의 P2P 자원 검색 알고리즘 관련 기반 연구에 대해 설명하고자 한다. "최근 급속히 증가하고 있는 P2P 응용프로그램의 네트워크 트래픽 양을 감소시키기 위한 연구의 필요성이 부각되고 있는데, 이는 효과적인 P2P 검색 기법의 필요성과 대응되는 주제이다. 본 장에서는 논의되고 있는 주요 검색 시스템을 분류해 보고, 각 검색 메커니즘의 특징을 살펴보겠다[2][3][4][5].

P2P 시스템에서 자원의 검색 기법은 [그림 1]에서와 같이P2P 네트워크 구조에 따라 분류를 할 수 있다. 먼저 P2P 네트워크 구조는 피어 간의 네트워킹 방법과 역할에 따라 순수한 P2P, 간단한 조회 기능 서버를 가진 P2P[6][7], 조회 서버와 룩업 서버를 가진 P2P, 조회/룩업/컨텐츠 제공 기능의 서버를 가진 P2P 모델로 분류할 수 있는데, 여기서 순수한 P2P 모델을 제외한 나머지 서버의 기능이 있는 모델을 일반적으로 하이브리드 P2P로 묶어서 분류한다. 하이브리드 P2P 구조에 속한 검색 기법은 중앙집중식 모델 (Centralized Directory Model)로써 대변되는데, 피어의 연결정보를 보유한 중앙 서버가 검색 대상 데이터의 보유 여부에 따라 중앙집중형 데이터베이스 모델(Centralized DB Model)과 분산형 데이터베이스 모델(Decentralized DB Model)로 나뉜다. 순수한 P2P 모델의 경우는 검색 기법의 특징에 따라 브로드캐스트 요청 모델(Broadcast Requests Model), 도큐먼트 라우팅 모델(Document Routing Model), 하이퍼큐브 토플로지 검색 모델(Hypercube Topology Search Model)로 나뉜다.

중앙집중식 디렉토리 모델은 하이브리드 네트워크 구조에서 사용되는 디스커버리 기법으로써 Napster[8]가 대표적인 예이다. 이는 피어의 연결정보를 보유한 중앙 서버가 검색 대상 데이터의 보유 여부에 따라 중앙집중형 데이터베이스 모델(Centralized DB Model)과 분산형 데이터베이스 모델(Decentralized DB Model)로 나뉜다. 중앙 집중형 데이터베이스 모델의 중앙 서버는 피어들의 보유 자원에 대한 메타데이터를 저장하고 관리해야 하는 역할을 하므로 중앙서버의 데이터베이스 구축을 위한 네트워크 트래픽이 많다. 이에 비해 분산형 데이터베이스 모델은 중앙 서버가

피어들의 네트워크 연결만을 관리한다. 필요한 자원을 요청하고자 하는 피어는 중앙서버에게 질의를 하게 되고, 중앙서버는 이 메시지를 현재 연결되어있는 모든 피어들에게 전달하게 된다. 전달 받은 피어는 자신의 자원 가운데 해당되는게 있다면 해당 메타데이터를 중앙 서버에게 전달하고, 전달 받은 중앙 서버가 자원 요청 피어에게 메타데이터를 전송하고, 이후 피어들 간의 통신이 일어난다. 따라서 네트워크 대역폭 요구가 많은 편이다.

<table>
<tr><td rowspan="2">하이브리드
P2P</td><td rowspan="2">중앙 집중형
디렉토리 모델</td><td>중앙 집중형
데이터베이스 모델</td></tr>
<tr><td>분산형
데이터베이스 모델</td></tr>
<tr><td rowspan="3">순수
P2P</td><td colspan="2">다큐먼트
라우팅 모델</td></tr>
<tr><td>브로드캐스트</td><td>인폼드 검색</td></tr>
<tr><td>요청 모델</td><td>블라인드 검색</td></tr>
</table>

그림 1. P2P 검색 알고리즘 분류
Fig 1. Classification of P2P search algorithm

순수한 P2P 모델에서 브로드캐스트 요청 모델은 검색 메시지를 전달하는데 있어, 피어들의 참여여부에 따라서 모델들 구분할 수 있다. 피어들이 검색 메시지를 전달할 때 정해져 있는 규칙에 의해 메시지 전달하는 방법 가운데, Gnutella[9]는 자원을 찾기 위해 TTL 홉 내에 있는 접근 가능한 모든 피어에 대해서 BFS(Breath First Search)[3]기반의 브로드캐스트 기법을 사용한다. 이 알고리즘의 검색 기법은 검색의 원형으로써 간단하고 구현이 용이하다. 그러나 자원을 검색할 경우 메시지 내의 TTL을 포함시키기 때문에 네트워크 그룹 내에 참여하고 있는 모든 피어에게 메시지를 전송하지 않는다. 단, TTL 값의 범위 내에서 이 메시지를 받은 피어들은 연결되어 있는 모든 피어에게 메시지를 전송함으로써 상당한 네트워크 트래픽을 유발한다. 변형된BFS(Modified-BFS)는 브로드캐스트 기법의 변형된 형태로써, 인접피어 중 일정 비율에 해당하는 피어들만을 선택하여 전송하는 방법이다. 인접한 피어들보다 근거리에 있는 피어들의 검색에 중점을 둔 알고리즘으로 메시지의 평균 발생률은 Gnutella에 비해 줄었지만, 메시지를 수신하는 피어 및 검색 결과의 수는 Gnutella와 유사한 특징을 보인다. 반복적 전개법(Iterative Deepening)은 TTL 값을 점점 증가시키면서 BFS 알고리즘을 이용하여 검색하는 방법이다. 사용자가 정의한 특정 검색 결과에 만족할 때까지 TTL의 수를 증가하면서 검색 메시지를 전송한다. 사용자가 정의한 검색 결과에 만족하지 않을 경우 TTL의 값을 증가시키면서 검색 메시지를 전송하기 때문에 표준 브로드캐스트 기법보다 더 많은 트래픽을 발생시킬 수 있다. Random Walks는 검색 메시지를 전송할 때 검색 메시지의 수를 일정한 수로 제한하여 생성하고 전달한다. 검색 하고자 하는 피어는 k개의 검색 메시지를 인접 피어들에 전송하고, 이 메시지를 받은 피어는 검색 메시지에 대한 작업을 처리하게 되며, 검색 메시지 내의 TTL 값이 1 이상이라면 1만큼 감소시킨 후 인접한 피어 중 임의의 피어에게 검색 메시지를 전송하게 된다. 전달되는 검색 메시지들에 의해 검색 수행이 이뤄진 피어는 메시지에 대한 처리를 한 후 검색 메시지를 최초 발생시킨 피어에게 현재 피어의 검색 결과가 만족스러운지에 대해 질의함으로써 메시지 전송 중단을 결정할 수도 있다. 결국 메시지 내에 지정된 TTL 홉 수만큼 전달될 수 있기 때문에 최악의 경우 검색 메시지는 k*TTL 만큼 발생한다. GUESS[10]는 울트라 피어(ultrapeer) 개념을 기반으로 하며, 각 피어들은 해당 울트라 피어에 접속하고 울트라 피어는 자식 피어들의 프락시 역할을 하게 된다. 첫 자식 피어로부터 검색 메시지를 수신한 울트라 피어는 자신의 자식 피어에게 검색 메시지를 전송하게 되고, 검색 메시지에 해당하는 자원이 검색되지 않을 경우 인접한 다른 울트라 피어에게 접속하여 검색 메시지를 전송하게 된다. 다른 울트라 피어로부터 검색 메시지를 받은 울트라 피어는 자신의 자식 피어들에게 검색 메시지를 전송하게 되고, 검색된 자원들의 메타 데이터를 자원 검색을 요청한 피어에게 전송하게 된다. 이러한 작업은 충분한 검색 결과가 나올 때까지 진행되며, 이러한 요청에 있어서 울트라 피어의 순서는 정해져 있지 않다. 순수한 P2P 모델에서 브로드캐스트 요청 모델에서 피어들이 관련 정보를 저장하고 있어서, 이 정보를 이용하여 메시지가 전송될 피어를 결정하여 전송하는 또 다른 형태가 있다. 그 예로 지능적 BFS(Intelligent-BFS) 기법은 변형된 BFS(modified-BFS) 알고리즘을 확장하여 지식기반으로 동작하도록 변형시킨 경우이다. 검색을 하고자 하는 피어는 변형된-BFS 알고리즘을 이용하여 인접 피어에게 검색 메시지를 보내게 된다. 이후 검색 메시지를 받은 피어들 중에서 해당 자원을 가지고 있는 피어는 역 경로로 결과값을 전송하게 된다. 이때 검색을 요청한 피어와 결과값을 반환하는 피어 사이에서 결과 메시지를 중재하여 주는 피어들은 결과 메시지로부터 얻게되는 자원이 검색된 피어에 대한 정보를 자신의 데이터베이스에 저장하게 된다. 그 다음 피어로 부터 검색 메시지를 수신하게 되면 해당 피

어는 저장된 결과값을 이용하여 검색메시지의 전달 경로를 결정할 수 있게 된다. 이 알고리즘은 네트워크에서 발생되는 검색 메시지 양을 줄일 수는 없지만, 자원이 있을 법한 피어에게 검색 메시지를 전달함으로써 검색의 적중률을 높일 수 있다. 그러나 피어 및 데이터의 삭제와 관련된 메시지는 전송하지 않아 자원의 삭제가 자주 발생하는 환경에서는 부정확한 결과를 낳을 수 있다. APS(Adaptive Probabilistic Search)[11]는 자원을 찾기 위해 TTL 홉 내에 있는 접근 가능한 모든 피어에 대해서 BFS 기반의 인접 피어 중 일정 비율에 해당하는 피어들만을 선택하여 전송하는데, 인접한 피어들보다 근거리에 있는 피어들의 검색에 중점을둔 알고리즘으로 각 피어가 검색 메시지의 송신과 검색에 대한 결과 메시지의 수신이 했을 때 해당 자원에 대한 방향성을 지역 데이터베이스에 저장하게 되고, 이후 질의 발생시 검색에 대한 적중률을 높이게 한다. LI[12]는 각 피어들이 r홉 내에 있는 모든 피어들이 보유한 자원들의 목록을 유지한다. 하나의 피어가 검색 메시지를 수신할 경우 해당 피어는 r홉 내의 모든 피어에게 보유 자원에 대한 메타데이터를 가지고 있기 때문에 인접한 피어들에게 검색 메시지를 정송한 것과 같은 결과를 얻을 수 있다. 이 방법을 이용함으로써 몇 개의 피어에게만 검색 메시지를 전송하여도 많은 피어들에게서 검색한 것과 동일한 결과값을 얻게 되어 낮은 비용으로 충분한 결과물을 획득하는 이점이 있다. RI[13]는 각 피어는 자원을 보유한 실제 피어에 대한 정보를 저장하는 것이 아니고, 그 자원이 있는 곳의 방향을 저장하는 방법이다. DRLP[14]는 LI 알고리즘과 유사하며, 이 기법을 이용하는 피어들은 처음부터 다른 피어들의 메타데이터를 저장하지 않는다. 다른 피어부터 검색이 이뤄질 경우 각 피어들은 결과값 및 메터데이터도 함께 전송하여 LI 테이블을 업데이트 한다. 각 피어는 자신이 보유한 자원들을 가리키는 지역 디렉토리와 다른 피어의 지역 디렉토리를 가리키는 디렉토리 캐쉬를 가지게 된다. 만약 검색 메시지를 받은 피어에서 자원이 발견된다면 자원을 발견한 피어는 자원을 요청한 피어에게 역경로와 자원의 메타데이터가 저장된 메시지를 전송하게 된다. 그리고 검색 결과를 전송하는 피어는 해당 디렉토리 캐쉬 정보를 이용하여 역경로로 전송하기 때문에 관련 메시지를 수신하는 피어는 특정 피어의 디렉토리 캐쉬 정보를 유지할 수 있게 되는 것이다. 그렇지 않고 자원이 발견되지 않을 경우는 같은 방법으로 이웃 피어에게 메시지를 전송한다. 이 알고리즘을 이용하게 되면 처음에는 자원의 위치를 찾기위해 BFS 알고리즘을 이용하여 많은 검색 메시지를 전송하지만, 그 다음의 검색 과정에서는 디렉토리 캐쉬를 보유하고 있는 피어에 의하여 메시지 전송 과정이 감소하게 된다. 그러나 네트워크 구조가 변경될 경우 BFS 방법을 사용하여 다시 구축하기 때문에 검색 실패율이 증가되며, 네트워크 트래픽 발생율이 늘어나게 된다. GS는 Gnutella 알고리즘을 기반으로 하며, 검색 과정이 이루어진 후 자원을 지닌 피어에 대해 단거리 피어로 등록하는 방법이다. 검색을 원하는 피어는 브로드캐스트 기법을 이용하여 원하는 자원을 찾게 되며, 이후 검색 결과를 받은 피어는 검색된 피어들을 단거리로 이용한다. 따라서 이후의 검색 과정에서 단거리에 등록된 피어와 직접 연결하여 검색 메시지를 전송함으로 검색 시간이 줄어들게 된다. 단거리 경로에 해당하는 피어를 저장하는 단거리 리스트를 이용하여 해당 피어에 접속하게 되고, 그 결과 원하는 자원을 찾지 못하게 되면, 인접 피어로 브로드캐스트 기법을 적용하여 검색하게 된다. 순수 P2P 모델에서 도큐먼트 라우팅 모델은 각 피어가 보유하고 있는 공유 자원의 ID를 기반으로 자원을 검색하는 기법인 DHT[15]는 해쉬 테이블을 이용한 검색 기법으로 모든 피어들은 공유하고 있는 자원에 대한 메타 데이터를 해쉬 값에 의해 지정된 피어에게 전송한다. 메타데이터를 수신한 피어는 다른 피어로부터 검색 요청이 들어올 경우 자원의 위치를 알려주는 방식으로 동작하게 된다. VBST[16]는 P-Grid를 이용한 구조로써 피어가 공유하고 있는 자원의 사본들을 네트워크 그룹 내에 있는 여러 피어에게 분포시키고 각 피어로 하여금 필요한 자원을 검색할 수 있도록 라우팅 테이블을 제공하는 방법이다".[17] 지금까지 P2P 모델에 따른 기존의 검색 기법들과 해당 특징에 대해 간략히 설명하였다.

III. 제안 알고리즘

본 논문에서는 기존의 TO[17] 자원 검색 알고리즘 기반하에 자원 재전송 발생률을 줄이기 위해 동료 피어 동반 검색 알고리즘을 제안한다. 본 알고리즘의 동료피어 선정 기준은 다음과 같다.

$$S = Ran(Size(Grid)) = \left(\frac{\sum_{i=1}^{n} Hi(hit)}{n} \right) \quad \cdots\cdots\cdots \text{식 1)}$$

식 1)에서 Size(Grid)는 실험 공간의 전체 네트워크 사이즈이다. H는 현재 피어로부터 떨어진 거리를 나타내는 홉 수이다. Hi(hit)는 해당 피어가 이전에 원하는 피어를 찾았

던 기록이다. 따라서 이전 기록을 기준으로 히트가 일어난 홉수를 합산하여 전체 횟수 n 만큼 나누어 평균값을 구한다. 이 평균값을 기준으로 하고 전체 그리드 사이즈를 넘지 않는 범위로 해당 동료 피어를 임의로 선정하는 Rand 함수를 적용할 수 있겠다. 또는 다음과 같이 동료 피어 선정 기준을 정할 수도 있다.

```
The_Peer_Partner_Search( ) {
Initialization : pure P2P, random graph model
Definition : Qinode, Rinode, i, Niindex, K=1
Pre_Processor : S from an expression 1) or an
expression 2)
Start Parallel_Section(S).
   For (i = 0 ; i <= K; i++) {
    If (Niindex == null) {
      For ( i = 0 ; i <= neighbor; i++ ) {
        If ( Probabilistic.value(i) > Max ) {
          Max = Probabilistic.value(i) ;
      } }
    Neighbor_Select(Max);
   }
   else {
    Top_List(all_of_Niindex);
    Select_Max(Top_List.Probabilistic.value);
    Neighbor_Select;
   }
   If (Qinode == Rinode) {
          Backward_Forwarding(update_index);
          For (All neighbors of the hit node) {
            TO_hit_ratio_update
            constantV+recentTOHitRatio(0..1) }
   }
    When Broadcast_Flooding(Kwalker_success);
End Parallel_Section.
}
```

그림 2. TO 기반 동료 피어 동반 검색 알고리즘
Fig 2. Peer search algorithm with partner based on TO

$$S = Ran\left(\left(\frac{\sum_{i=1}^{n} Hi(hit)}{n} \right) > H(1) \right) \quad \cdots\cdots \text{식 2)}$$

식 1)의 구성과 세부 요소는 같다. 그러나 식 2)는 임의의 피어를 선정하는데 있어서 최대 홉 수를 이전 히트 기록의 평균수로 하고, 최저 홉 수를 바로 이웃 피어로 하여 검색 피어와의 평균값 보다 가까운 거리의 동료 피어를 선정하는 방법이다. 이 외에도 다양한 기준으로 동료 피어를 선정할 수 있겠으나 본 실험환경은 네트워크 사이즈가 17 X 15인 마이크로 사이즈로써 다양한 동료 피어 선택 기준에 따라 해당 성능에 미치는 영향을 파악하기 어려워 식 1)과 식 2)의 기준만으로 적용하고자 한다.

[그림 2]에서 제안한 알고리즘을 보여주고 있다. 이는 기본적으로 TO 알고리즘에서처럼 순수 P2P 모델을 기반으로 하며, P2P 오버레이 네트워크의 토플로지는 대부분의 P2P 검색 논문에서 적용한 랜덤 그래프 모델을 바탕으로 한다. 또한 질의와 자원의 분포는 Zipfian 이외의 실제 P2P 시스템에서 보여 지는 분포와 유사한 다양한 분포 법칙을 적용한다. 기본 입력값, 인덱스, 자원 보유 노드의 수에 따른 네트워크 구조 설정, 노드의 평균 차수 등에 대한 내용은 TO를 따른다. Walker는 질의 발생 노드의 K 수만큼 질의가 발생되어 K 수에 따라 전체 메시지 발생량에 큰 영향을 줌으로 적절한 수의 K를 결정하는데 있어 주요한 역할을 하고, 이웃노드에 관한 연관성 확률값 보유 인덱스인 Ni 인덱스의 값이 초기값 null을 가지고 있는 경우 이웃 노드를 선택하는 기준은 오로지 온라인 상태에 대한 확률값이다. 동료 피어는 기준식에 의해 임의의 값 S로 결정되고, 해당 TO의 검색 작업이 원래의 자원 검색 노드와 S 노드가 동시에 병렬로 동작되게 된다. 이러한 과정에서 원하는 자원을 발견하였을 시 질의 진행 방향과 역방향으로 연관성 확률 값을 갱신하면서 질의 발생 피어에 도달하게 된다. 즉 질의를 발생한 노드 Qj 노드와 부합하는 자원을 지닌 노드 Rj 노드를 찾은 경우 역방향으로 연관성 확률 값을 갱신하면서 질의 발생 피어에 도달하게 되는 것이다. 이러한 과정은 TO와 동일하다. 동료 피어 선택 기준인 식 1)과 식 2)에 따라 해당 성능의 차이는 성능 평가 절에서 다루고자 한다.

다음은 기존의 Peersim 시뮬레이터[18]의 전체 네트워크 구성 및 자원 보유 피어를 가시화하기 위해 그래픽화 한 모습이다. 먼저 [그림 3]은 전체 네트워크 사이즈를 너비와 높이를 기준으로 입력한 결과를 가시화한 장면이다. 해당 네트워크의 기본 설정으로 최대 링크 수는 8, 최소 링크 수는 1로 정하여 링크가 없는 피어인 섬(island)이 발생하지 않도록 정하였다.

[그림 4]은 자원 보유 피어의 수를 임의로 결정하여 실제 전체 피어 네트워크에 반영된 모습으로 검은색 점은 해당 자원을 보유한 피어를 표시하고, 진한 실선은 검색 작업을 수행 하고자 하는 피어를 표현한다. 해당 네트워크에 자원 을 배치하는데 있어 네트워크 자체를 17 X 15 매트릭스로 설정하였고, (9, 8) 지점이 중심된다. 이 중심을 기준으로 4사분면을 구성하고, 중심에서 떨어진 정도를 기준으로 (1사분면: 8,7) , (2사분면: 7,6) , (3사분면: 6,5) , (4사분면: 5,4) , (1사분면: 4,3) , (2사분면: 3,2) , (3사분면: 2,1) 노드에 자원이 배치되었다.

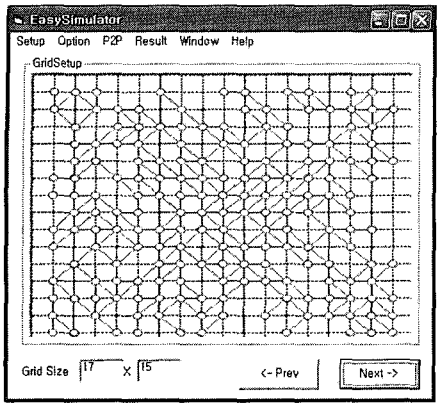

그림 3. 전체 피어 네트워크 사이즈 결정
Fig 3. Decision of network size for all peers

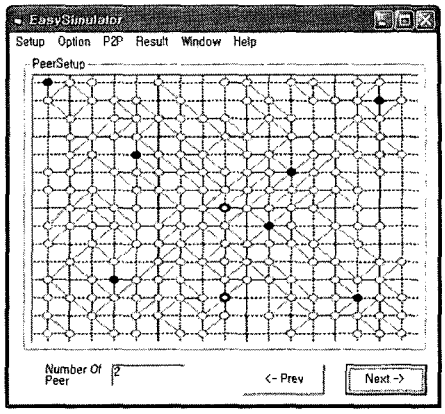

그림 5. 동반 피어 수 결정
Fig 5. Decision of numbers of the partner peer

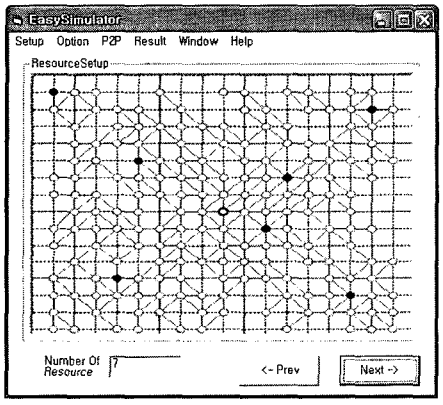

그림 4. 자원 보유 피어 결정
Fig 4. Decision of the peer to have resource

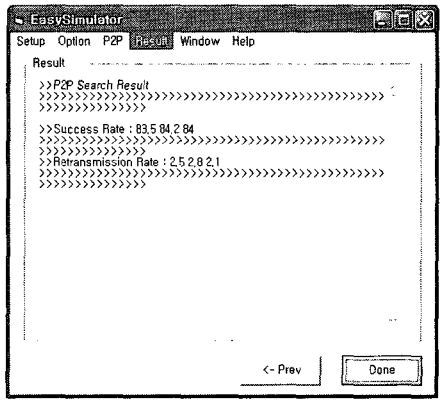

그림 6. 성능 평가 산출
Fig 6. Performance result

〔그림 5〕는 해당 피어 네트워크에 자원을 배치 한 이후에 동반 피어 선택 과정에 의해 해당 알고리즘을 적용한 결과로써 동시에 해당 검색 알고리즘을 수행할 피어 또한 진한 실선으로 표현하고 있다. 2개의 노드에서 해당 검색 작업을 분산처리하는 과정이다. 검색이 시작되는 노드에서 떨어진 거리를 결정하면 동시에 동일한 검색 작업에 대해 분산 처리할 노드가 임의로 결정된다. 〔그림 5〕는 5홉 거리에 위치한 피어가 원 검색 피어와 함께 동일한 검색 작업에 대해 분산 처리할 피어이다.

〔그림 6〕은 이러한 실험 환경에서 본 논문에서 제안한 TO 기반 동료 피어 동반 검색 알고리즘을 적용한 결과를 간략히 보여주고 있다.

IV. 평가

이 장에서는 참고문헌〔1〕에서와 같이 확장한 PeerSim P2P 시뮬레이터를 이용한 일련의 실험들을 통해 제안한 TO 기반 동료 피어 동반 검색 알고리즘의 성능을 평가한다. 제안한 검색 알고리즘의 성공적인 검색 연산을 확인하기 위해 관련 알고리즘 분석을 통해 조사된 Random Walks, TO와의 성능 비교 작업을 수행한다. 본 논문에서는 Window Server 2000 운영체제 하에 JDK 개발 환경을 기반으로 소프트웨어 플랫폼을 구성했고, 시스템 사양은 Intel Pentium III 871MHz, 하드디스크 40 Gb, 메인 메모리 256 Mb 환경에서 실험하였다. 해당 실험 환경 하에서 중요 시뮬레이션 인수 가운데 P2P 모델은 순수한 P2P 모델, 그래프 모델은 랜덤 그래프 모델을 기본으로 하

며, 노드 수, 평균 노드의 차수(degree), TTL, 객체 수는 3장의 설정을 따르고, K Walker의 수는 1로 평가하였다. 본 실험에서는 K Walker의 수에 따른 성능 평가를 목적으로 하지 않으므로 해당 수를 1로 정한 것이다. 따라서 평가 결과치는 기존의 평균 Walker 수에 따른 값에 대응 될 수 있도록 비교 대상 식으로 전이한 값으로 한다.

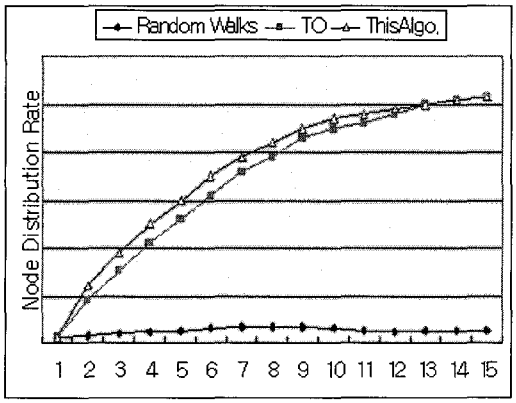

그림 7. 실험 단계에 따른 노드 분산율
Fig 7. Node distribution rates at the simulation steps

본 검색 알고리즘에 대해 복잡도를 구하여 Gnutella, TO와 성능 비교한 결과는 다음과 같다. 알고리즘 복잡도 계산 시 n에 대해 가장 빨리 증가하는 항만을 남기고 다른 항들은 무시해도 되는 점근적 복잡도 평가 방법을 적용하였다. 참고문헌 [17]과 동일한 방법으로 설정한 결과 TO 알고리즘과 동일한 O(log n)의 결과를 보여, Gnutella의 $O(n^n)$에 비해 확연히 낮은 복잡도를 보였다. 따라서 점근적 복잡도 평가에 의해 해당 알고리즘의 우수한 성능을 확인할 수 있었다.

[그림 7]에서 X축은 실험의 횟수를 Y축은 노드 분산율을 의미하는데, 노드 분산율은 피어들 간의 해당 질의가 전파되면서 특정 노드에 부하가 분산되는 정도를 의미한다. 실험 결과는 [그림 7]에서와 같이 Random Walks의 경우 시뮬레이션 단계의 변화에 따른 분산율의 변화에 영향이 거의 없었고, TO와 본 알고리즘은 실험 단계가 증가 될수록 급격한 증가를 보인다. 특히 본 알고리즘의 경우 초기 실험 단계에서 TO에 비해 분산율이 다소 높았고, 전반적으로 TO와 유사한 노드 분산율을 보이고 있다.

[그림 8]은 검색 성공률을 해당 실험 단계에 따라 Random Walks와 TO 그리고 본 알고리즘을 비교한 결과이다. Random Walks의 경우 실험 초기의 검색 성공률과 실험 단계가 증가함에 따른 변화의 추이가 거의 없어서 실험의 횟수가 미치는 영향이 미미한 것으로 판단된다. 그에 비해 TO와 본 알고리즘의 경우 어느 정도 실험의 횟수가 미치는 영향이 의미 있는 수치로 나타났다. 또한 해당 검색 성공률의 차이가 TO와 본 알고리즘이 2~3%에 지나지 않았으나 고른 성공률 증가 추이를 보이고 있으므로 의미있는 결과라 할 수 있다.

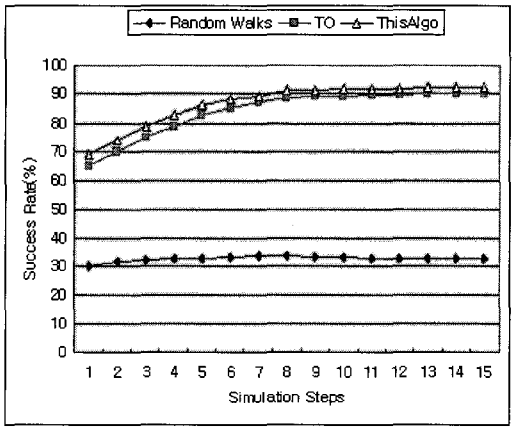

그림 8. 실험 단계에 따른 검색 성공률
Fig 8. Success rates at the simulation steps

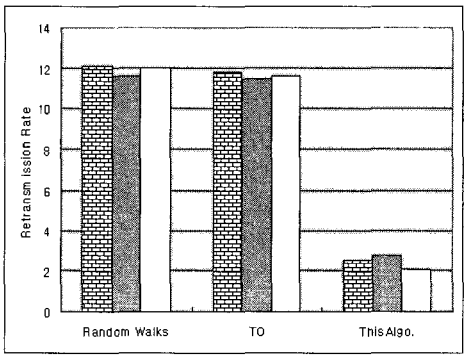

그림 9. 알고리즘에 따른 재전송율
Fig 9. Success rates at algorithms

해당 검색 작업을 진행중인 피어가 검색 알고리즘에 의해 처리되어진 결과로써 해당 자원을 보유한 피어를 검색하여 원래의 목적인 자원 다운로드 과정 중에 해당 피어가 여러 요인에 의해 자원 전송 중단의 결론에 이를 수 있다. 이 때 해당 자원을 지닌 또 다른 피어를 찾는 작업을 다시 시작하여야 하고, 원래의 목적인 자원 전송 작업을 다시금 시행하게 되는데, 이를 자원 재전송이라 한다. [그림 9]는 해당 알고리즘들의 재전송율을 나타낸 것이다. 이러한 자원 재전송율은 해당 값이 낮을수록 좋은 결과치를 낳게 되는데,

TO 기반 동료 피어 동반 검색 알고리즘과 기존의 TO 알고리즘의 재전송율을 살펴보면 본 알고리즘은 2.5%에 준하는 재전송율을 나타내었고, Random Walks는 12% 정도의 결과를 나타내었고, TO 알고리즘은 Random Walks와 마찬가지로 12%에 육박하는 재전송율을 보였다.

V. 결론 및 향후 연구

P2P 검색 알고리즘은 주로 해당 검색 알고리즘의 검색 성공률은 높이고 자원의 재 전송률을 줄임으로써 사용자 편의성 및 자원의 효율적인 사용의 결과를 얻고자 한다. 본 연구에서는 P2P 시스템의 자원 검색 과정에서 해당 검색 작업을 동시에 수행할 동료 피어를 선택하여, 해당 검색 작업을 동반 수행하도록 함으로써 검색 성능을 높이고자 하였다. 또한 전송 도중 작업이 중단되어 재전송이 필요한 경우를 대비하기 위한 본 알고리즘의 목적에서 동반 피어를 함께 이용함으로써 재전송에 의한 추가적인 검색과정을 줄이는 효과를 얻고자 하였다. 실험 및 평가 결과 알고리즘 복잡도 대비 높은 검색 성능을 보이고 있다. 또한 자원 재전송율 측면에서 이전의 TO 알고리즘에 비해 의미있는 차이를 보임으로써 그 유용성을 확인할 수 있었다.

향후 연구과제로 본 검색 알고리즘에서 동료 피어 선정 기준에 휴리스틱 기법을 적용하여 좀더 명확한 결과값을 제시하고자 하며, 마이크로 실험 환경을 확장하여 좀 더 정확한 결론에 도달하고자 한다. 동료 피어 선정 기준식의 휴리스틱 기법을 적용하여 최적의 식을 도출하는데 에는 많은 실험이 필요하며, 다양한 시도에 해당 인공지능 방법론의 충분한 실험을 거치는 작업이 필요하겠다. 또한 P2P 시뮬레이터의 시각화 작업을 시도하였으나 초기 단계로 사용의 편의성 측면에서 보완이 필요하여 지속적인 갱신이 요구된다.

참고문헌

[1] 김분희, "표준 미들웨어 기반 분산 객체 리플리카를 위한 효과적 푸쉬 결함 관리 알고리즘", 한국컴퓨터정보학회 논문지, 10권 6호, pp.37-45, 2005. 12.

[2] K. Aberer, et al., "Improving Data Access in P2P Systems," IEEE, vol. 5, no. 1, pp. 58-67, 2002.

[3] D. Tsoumakos and N. Roussopolulos, "Analysis and Comparison of P2P Search Methods," Technical Report(CT-TR-4451), University of Maryland, Dept. of Computer Science, 2003.

[4] B. Yang and H. Garcia-Molina, "Improving search in peer-to-peer networks," ICDCS'02, pp.103-113, 2002.

[5] Boon-Hee Kim, Young-Chan Kim, "Automatically Generated Ontology System for Peer-to-Peer Networks", Lecture Notes in Computer Science 3320, Springer-Verlag, pp. 493-496, 2004.12.

[6] Boon-Hee Kim, Young-Chan Kim, "Ptops Index Server for Advanced Search Performance of P2P System with a Simple Discovery Server," LNCS 3032, Springer-Verlag, pp. 285-291, 2004. 4.

[7] P. Ganesan, et al., "YAPPERS: A Peer-to-Peer Lookup Service over Arbitrary Topology", INFOCOM'03, pp.1250-1260, 2003.

[8] Napster website: http://www.napster.com

[9] Gnutella website: http://gnutella.wego.com

[10] S. Daswani and A. Fisk, Gnutella UDP extension for scalable searches (GUESS) v0.1, White Paper, 2002.

[11] J. Chu, K. Labonte, and B. Levine, "Availability and locality measurements of peer-to-peer file systems," SPIE, pp.82-94, 2002.

[12] B. Yang and H. Garcia-Molina, "Improving search in peer-to-peer networks," ICDCS, pp.103-113, 2002.

[13] D. Tsoumakos and N. Roussopoulos, "A Comparison of Peer-to-Peer Search Methods," WebDB, pp.61-66, 2003.

[14] Daniel A. Menasce and Lavanya Kanchanapalli, "Probabilistic Scalable P2P Resource Location Services," ACM SIGCOMM, pp.48-58, 2002.

[15] Sean C. Rhea and John Kubiatowics, "Probabilistic Location and Routing," INFOCOM02, pp.336-346, 2002.

[16] K. Aberer and Z. Despotovic, "Managing Trust in a Peer-2-Peer Information System," Proc.

10th Int'l Conf. Information and Knowledge Management (2001 ACM CIKM), ACM Press, New York, pp.310-317, 2001.

[17] 김분희, "분산 객체의 확률적 비례 검색 기반 전송률 향상 검색 알고리즘", 한국컴퓨터정보학회 논문지, 11권 3호, pp.49-56, 2006. 7.

[18] 김분희, 김영찬, "P2P 오버레이 네트워크의 효과적 평가를 위한 시뮬레이터 설계" 한국인터넷정보학회 2004 추계학술발표논문집 제5권 제 2호 pp.65-68, 2004.11.

저 자 소 개

김 분 희

2005년 2월 : 중앙대학교 컴퓨터공학과 공학박사

2005년~현재 : 동명대학교 멀티미디어공학과 전임강사

〈관심분야〉 분산 시스템, P2P 검색 기법, 웹 표준 응용