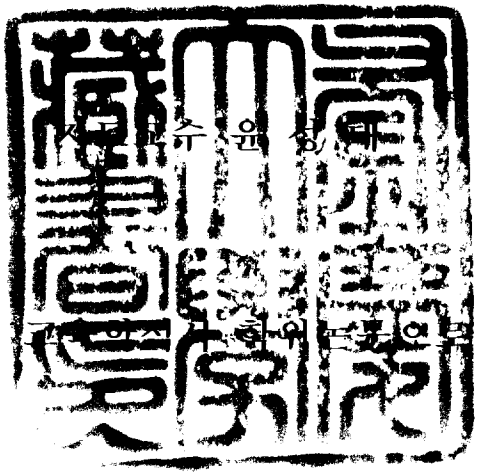
교육학석사학위논문

비구조적 P2P 네트워크에서 Shortcut을 이용한 적응적 확률 검색 기법



지도교수 윤 성 대

이 논문을 교육학 석사학위 논문으로 제출함

2005년 8월

부경대학교 교육대학원

전산교육전공

정 귀 녀

정귀녀의 교육학석사 학위논문을 인준함

2005년 6월 17일

주 심 공학박사 여 정 모 

위 원 공학박사 정 순 호 

위 원 이학박사 윤 성 대 

[차 례]

[표 차례]

[그림 차례]

The Adaptive Probabilistic Search Scheme using Shortcuts in Unstructured P2P Networks

Gwi Nyeo Jeong

Graduate School of Education
Pukyoung National University

Abstract

Preexisting internet services are designed based on the client/server model which has the problems such as network failure and the low speed communication due to the high load on the central server. In order to solve these problems, peer-to-peer networks are introduced and expanded by internet environments.

The most popular P2P applications operate on unstructured networks. An efficient search mechanism is very important in peer-to-peer networks because peers operate both clients and servers.

In this paper, we propose a resource discovery mechanism for unstructured peer-to-peer networks. In the proposed scheme, we describe an efficient algorithm for search in unstructured peer-to-peer networks, the Adaptive Probabilistic Search using Shortcuts(APSS).

APSS algorithm builds upon APS(Adaptive Probabilistic Search) which feedbacks from tne previous search to probabilistically guide future ones and shorcut list, direct links to peers that have recently proved useful in answering queries. The proposed scheme can solve the search locality problem that occurs when TTL is limited.

By the simulation, we have known that APSS achieves high success rates, increased number of discovered objects than the preexisting APS in unstructured peer-to-peer networks.

I. 서론

최근 네트워크 대역폭의 증가, 디스크 저장장치 용량의 증가, 컴퓨팅 프로세서 집적도의 증가 등의 기술 발전으로 각 PC들은 기존의 정보 소비자로서의 단말 역할을 수행하고도 남을 만큼의 잉여 자원들을 가지기 시작하였다. 이러한 기술 발전은 인터넷의 상용화 이후 수년 동안 수동적인 정보 소비자로서의 역할만을 수행하며 은둔해왔던 사용자들의 디바이스(device)들을 네트워크에서 직접 접근해서 활용 가능한 자원의 보고로 이끌어냈고, P2P 네트워크 환경을 통해서 많은 피어들의 스토리지나 프로세서들을 공동으로 활용하여 대용량의 스토리지 또는 고도의 프로세싱 능력을 제공하도록 지원할 수도 있으며, 각각의 피어에 저장되어 있는 각종 미디어 컨텐츠들이나 공동 작업에 필요한 파일 등의 공유 기능도 지원할 수 있게 되었다[17].

P2P 네트워크는 "각 피어들이 가진 자원이나 제공하는 서비스를 피어들 상호간의 직접적인 연결을 통하여 교환할 수 있도록 지원하는 네트워크 환경"이라 정의할 수 있다. P2P 네트워크는 네트워크에 연결되어 있는 모든 컴퓨터들이 서로 대등한 입장에서 데이터와 주변장치 등을 공유할 수 있는 방식을 의미한다. 이를 통해 웹 사이트에서 파일을 다운로드 받는 기존 방식 대신에 서로 연결된 타 사용자들의 컴퓨터에 직접 접근하거나 이에 필요한 주변장치로 공유할 수 있다[15].

기존의 클라이언트-서버(client-server) 네트워크 모델은 P2P 네트워크와 비교해 볼 때 서버가 중심이 되는 중앙 집중형 모델이며, 클라이언트-서버 네트워크 모델에서 동작하는 각 피어들은 서버에 접속한 후 연결 세션을 유지하면서 서비스를 제공받는 방식으로 동작하게 된다. 따라서 클라이언

트-서버 환경의 각 피어들은 서버가 운영되는 동안 언제든지 일관성 있는 서비스를 제공받을 수 있지만 각 피어들로부터 발생되는 검색 메시지들이 서버로 집중되기 때문에 높은 네트워크 대역폭과 처리 능력, 그리고 방대한 저장 공간을 보유한 서버를 필요로 하게 된다. 또한 각 피어들에게 서비스를 제공하는 서버가 더 이상 운영되지 않는 경우 사용자들이 어떠한 서비스도 제공받지 못하는 단점을 갖는다[10].

이러한 문제점들을 해결하기 위한 방법으로 P2P 네트워크 개념이 사용되기 시작하였다. 모든 자원을 서버가 관리하는 클라이언트-서버 모델과는 달리 P2P 네트워크 모델에서는 각 클라이언트, 즉 피어들에게 자원이 분산되어 있기 때문에 사용자들은 원하는 자원을 제공하는 피어에게 직접 연결하여 서비스를 제공받게 된다.

그러나 P2P 네트워크 모델은 클라이언트-서버 네트워크 모델처럼 특정 피어가 자원을 관리하지 않기 때문에 많은 피어들에게 분산되어 있는 여러 자원들을 관리하는 기법이 필요하게 되며, P2P 네트워크 구조에 맞는 피어 관리 기법 및 자원 검색 기법 등을 필요로 한다.

기존의 비구조적(unstructured) P2P 네트워크의 효율적인 자원 검색 기법의 하나인 APS(Adaptive Probabilistic Search) 기법은 기존의 플러딩(flooding) 기법이나 그 변형들이 가지는 네트워크 트래픽의 과도한 증가 문제와 여러가지 색인을 유지하는 기법들에서 나타나는 높은 색인 유지 비용문제를 해결하고 있다. 특히, Random Walks와 같이 bandwidth-efficient 가 뛰어나면서 Random Walks에 비해 검색 성능이 현저히 우수하다고 알려져 있다[2].

그러나 APS 기법은 검색 메시지의 전송시 이웃 피어들에 대한 APS 테이블(table)의 엔트리(entry)가 가지는 검색확률수치를 이용하여 검색 메시지를 전송할 피어를 선정하는데, 검색 결과를 많이 가졌던 피어의 검색확

률수치가 높더라도 그 피어가 다음 검색에서 선정될 가능성이 높아진 것을 의미하는 것이지 꼭 그 피어로 검색 메시지가 전송된다고 보장할 수는 없으며 자원검색 시마다 동일한 경로(path)를 따라갈 가능성이 크다는 단점을 가진다.

또한 검색시 TTL(Time To Live)의 한계로 인해 발생하는 검색의 지역성(locality) 문제를 해결하지 못한다. TTL의 사용으로 검색 메시지 전송이 홉 카운터(hop counter)의 제한을 받기 때문에 일정 홉 안에 없는 피어의 자원은 검색이 불가능하기 때문이다. 이 문제는 APS 기법만의 문제가 아니라 대부분의 검색 기법들의 문제라고 할 수 있다.

본 논문에서는 비구조적 P2P 네트워크를 기반으로 검색 성능을 향상시키기 위한 새로운 검색 기법인 APSS(Adaptive Probabilistic Search using Shortcuts) 기법을 제안한다. 제안한 APSS 기법은 자원이 있을 가능성이 있는 피어에게 검색 메시지를 전달하는 APS 기법에 자원을 가진 피어로 직접 연결할 수 있게 하는 Shortcut 목록(list)을 이용한 검색 기법이다. APSS 기법을 사용함으로써 APS 기법의 단점을 보완하고 검색의 지역성 문제를 해결하여 자원의 검색 성능을 높일 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구로서 P2P 컴퓨팅 환경 기반의 네트워크 구조를 비교하고, 그 네트워크 구조 중의 하나인 비구조적 네트워크에서 사용되는 검색 기법들을 분석한다. 3장에서는 APSS 기법의 구조와 알고리즘을 설명하고 시나리오를 통해 APSS 기법의 동작과정을 보인다. 4장에서는 시뮬레이션을 이용하여 성능평가를 하고, 마지막으로 5장에서는 본 논문의 결론을 맺고 향후 연구 과제를 제시한다.

II. 관련연구

본 장에서는 P2P 컴퓨팅 환경 기반의 네트워크 구조의 종류에 대해 살펴보고, 기존의 비구조적 네트워크에서 사용되는 검색 기법들을 분석한다.

2.1 P2P 컴퓨팅 환경 기반 네트워크 구조

P2P 네트워크 모델은 서버 운영의 유무에 따라 혼합형 P2P 네트워크 구조(hybrid P2P network structure)와 순수 P2P 네트워크 구조(pure P2P network structure)로 나눠지고, 순수 P2P 네트워크는 각 피어들이 형성하고 있는 네트워크 그룹의 특성에 따라 구조적(structured) 네트워크와 비구조적(unstructured) 네트워크로 세분화할 수 있다[13].

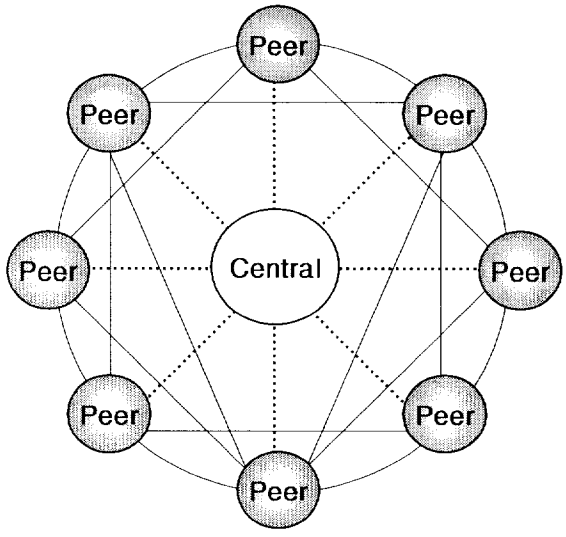
2.1.1 혼합형 P2P 네트워크 구조

혼합형 P2P 네트워크 구조는 서버를 통해 각 피어들에게 자원의 공유기능을 지원하는 모델로써 서비스를 제공할 수 있는 중앙 서버(centralized server)와 실질적인 공유자원을 가지고 있는 피어들로 이루어지는 것으로 그림 1과 같은 구조를 가진다.

중앙 서버는 각 피어들로부터의 네트워크 연결 세션을 유지함으로써 각 피어들 간의 통신에 대한 중재 역할을 한다. 피어들은 서비스를 제공하는 중앙 서버에 접속하여 원하는 자원을 보유한 피어를 검색하게 되지만 일단 다른 피어와의 네트워크 세션이 성립되면 중앙 서버의 참여 없이도 상호간

정보교환을 진행할 수 있다.

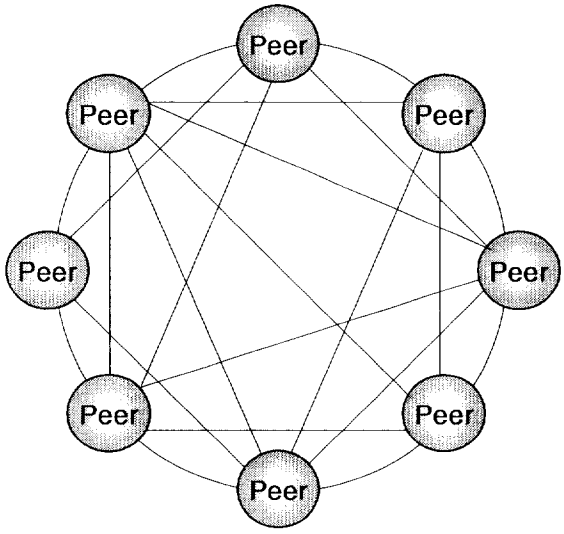
이 구조는 기존의 클라이언트-서버 네트워크 구조에서 사용자들이 특정 파일을 중앙 서버로부터 받아감으로써 발생되는 네트워크 트래픽을 일반 피어들에게 분산시키기 때문에 각 피어들의 자원을 빠른 속도로 공유할 수 있다는 장점을 가지고 있지만, 자원의 공유를 위해서는 모든 피어들이 일단 중앙 서버에 연결하므로 클라이언트-서버 모델의 중앙 서버로 인한 단점을 계속해서 일부 갖게 된다. 또한 서버가 다운되면 전체 시스템이 다운된다는 문제점을 가지고 모든 피어가 서버와 접속을 하여 정보를 얻는 방식이므로 서버 측의 병목현상으로 네트워크의 확장 가능성(scalability)이 떨어진다는 문제가 있다. 이 구조를 갖는 P2P 네트워크 시스템으로는 Napster가 있다[11].



(그림 1) 혼합형 P2P 네트워크 구조

2.1.2 순수 P2P 네트워크 구조

순수 P2P 네트워크 구조는 피어들에게 서비스를 제공하는 특정 서버를 두지 않고 모든 피어들이 서버와 클라이언트의 기능을 동시에 갖는 방식이다. 그림 2와 같이 순수 P2P 네트워크는 어떠한 서버도 존재하지 않으며 피어 간에 직접 검색 과정을 수행한다.



(그림 2) 순수형 P2P 네트워크 구조

이 방식을 사용하는 시스템에서 각 피어들은 최초 실행 시 네트워크 그룹에 이미 참여하고 있는 피어들에게 연결하게 되며, 이러한 방식으로 모든 피어들이 피라미드식으로 연결되어 무제한의 자료를 공유하게 된다. 따라서 이 구조는 특정 서버를 두지 않고도 서비스를 제공하거나 제공받을 수 있다는 장점을 갖는다. 다른 피어들에게 연결하여 자원의 위치를 검색한 후 이를 공유하는 형태로 특정 피어가 종료되어도 다른 피어들로부터 서비스를 받을 수 있기 때문에 높은 신뢰성과 확장성을 가진다. 그러나 수

많은 네트워크 연결이 이루어지기 때문에 자원을 검색하고 서비스를 받기 위해서는 많은 네트워크 트래픽이 발생하게 된다는 단점이 있다.

순수 P2P 네트워크는 각 피어들이 형성하고 있는 네트워크 그룹의 특성에 따라 구조적 네트워크와 비구조적 네트워크로 구분할 수 있다[1,11].

구조적 네트워크는 자원 검색을 용이하게 하기 위해 각 피어들을 특정 규칙을 이용하여 다른 피어들과 연결함으로써 네트워크 연결을 보다 구조화된 형태로 구성하게 된다.

구조적 네트워크는 일정한 모양의 토폴로지를 갖고 있고 파일의 배치도 규칙적이기에 검색 비용이 낮다. 그러나 구조적 네트워크에서 각 피어들은 서로 의존적이기 때문에 네트워크에 새로운 피어가 참여(join)하거나 떠나면(leave) 그것을 복구하기 위한 비용이 크고 일정한 토폴로지를 갖기 위해 일부 노드가 다른 노드에 대한 정보를 갖고 있어야 하므로 익명성이 보장되지 않는다.

비구조적 네트워크는 다른 피어들과의 연결이나 파일의 위치 등에 어떠한 정보를 관리하지 않으며 초창기의 순수 P2P 네트워크 형태를 그대로 따른다. 이 구조는 연결이 랜덤하게 이루어지고 피어들 사이의 의존성이 낮기 때문에 피어들이 네트워크를 떠나도 다른 피어들은 큰 영향을 받지 않는다. Gnutella에서는 ping 메시지를 보냈을 때 ping을 받지 못하면 그 피어와 연결이 끊겼거나 그 피어가 네트워크를 떠났다는 것을 알게 되고 다른 연결을 통하여 검색을 하고 또 새로운 연결을 맺는다.

이처럼 비구조적 네트워크는 서로에 대한 정보를 거의 갖고 있지 않고 플러딩(flooding) 방식으로 검색을 하므로 익명성이 보장된다. 그러나 검색 시 방문한 노드 수 만큼의 검색 비용이 생기므로 검색의 범위가 클수록 검색비용이 커진다. 물론 TTL을 이용하여 경과 노드수를 제한할 수 있지만 그로 인해 검색 범위도 제한된다[1,2].

2.2 비구조적 네트워크를 위한 자원 검색 기법

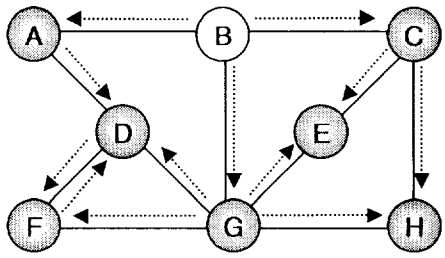
순수 P2P 네트워크 내에 공유되는 모든 자원들의 인덱스 정보를 유지하는 중앙 서버가 존재하지 않은 환경에서 사용되는 자원 검색 기법이다. 이 네트워크의 검색 기법은 검색 메시지의 전송에 대한 피어들의 참여 여부에 따라 Blind search 기법과 Informed search 기법으로 분류된다[1].

2.2.1 Blind Search 기법

각 피어들이 검색 메시지를 전송할 경우 정해진 규칙에 의해 메시지를 전달하는 방법으로 Gnutella, Modified-BFS, Random Walks 등의 기법들이 있다[1,4].

(1) Gnutella 기법

Gnutella 기법은 TTL hop 내에 있는 접근 가능한 모든 피어들에 대해서 BFS(Breath First Search) 기반의 브로드캐스트 기법을 사용한다. 이 방법은 여러 검색 기법 중 가장 간단하여 구현이 용이하나, TTL 값이 허락하는 범위의 메시지를 받은 피어들은 연결되어 있는 모든 피어들에게 메시지를 전송함으로써 많은 네트워크 트래픽이 발생한다. 그림 3은 Gnutella 기법을 이용한 검색메시지 전송과정이다[1,6].

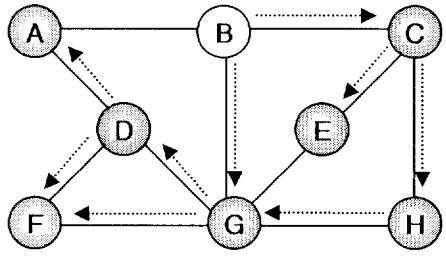


(그림 3) Gnutella 기법을 이용한 검색메시지 전송

(2) Modified-BFS 기법

Modified-BFS 기법은 브로드캐스트 기법의 변형으로 이웃 피어 중 일정 비율에 해당하는 피어들만 전송하는 방식이다. 이 기법을 이용한 검색 메시지의 전송은 그림 4와 같다.

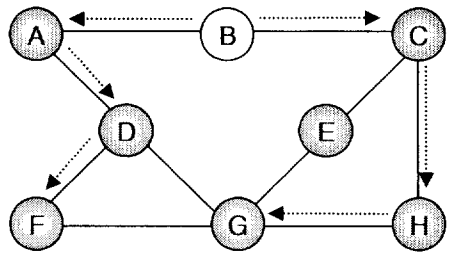
이 기법은 인접한 피어들보다 근거리에 있는 피어들의 검색에 중점을 둔 알고리즘으로써 전체 네트워크의 메시지 수가 Gnutella 기법에 비해 확실히 줄어들지만 메시지를 수신하는 피어들의 수와 검색 결과는 Gnutella 기법과 유사하다[1,8].



(그림 4) Modified-BFS 기법을 이용한 검색메시지 전송

(3) Random Walks 기법

Random Walks 기법은 검색 메시지를 이웃 피어로 전부 보내지 않고 일정한 개수만 랜덤하게 이웃 피어로 전송하는 기법이다. 이 메시지를 받은 다른 피어도 일정 개수만을 랜덤하게 다시 이웃 피어로 전송한다. 따라서 전체 네트워크에서 메시지 수는 줄어들지만, 검색 성능은 떨어진다. 그림 5는 검색메시지를 TTL=3으로 하여 2개만 전송하는 모습이다[1,7].



(그림 5) Random Walks 기법을 이용한 메시지 전송

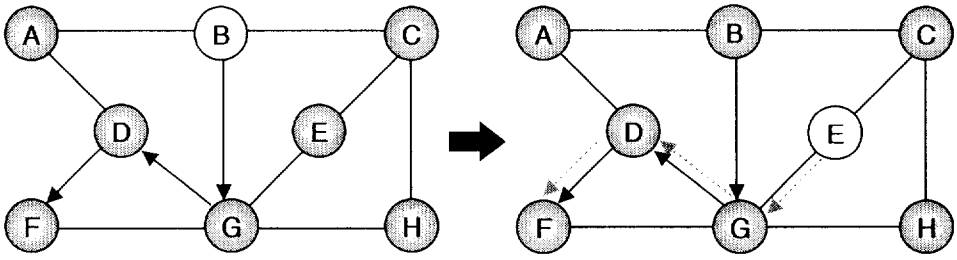
2.2.2 Informed Search 기법

각 피어들이 검색 메시지를 전송할 때, 그에 대한 반응을 저장하고 이렇게 저장된 여러 정보를 이용하여 메시지가 전송되는 피어를 결정하여 전송하는 기법으로 Intelligent-BFS, APS(Adaptive Probabilistic Search), GS(Gnutella with Shortcut) 등의 기법들이 있다[1,4].

(1) Intelligent-BFS 기법

Intelligent-BFS 기법은 Modified-BFS 알고리즘을 확장한 것이다. 검색 메시지의 전송 후 해당 자원을 가지고 있는 피어는 즉시 검색 메시지를 보낸 피어와 접속을 하는 방식이 아닌, 역경로(reverse path)를 따라 그 결과를 반환하게 된다. 따라서 검색을 시도한 피어와 자원을 가지고 있는 피어 사이에 중계를 한 피어들도 그 검색의 결과를 알 수 있게 된다. 이후 이 정보를 이용하여 검색 메시지에 따라 전송할 피어를 결정하게 된다. 검색 메시지를 받은 피어들이 저장되어 있는 데이터를 기반으로 전달할 피어를 선택하는 동작과정은 그림 6과 같다.

이 기법을 이용함으로써 네트워크에서 발생되는 검색 메시지의 양을 줄일 수는 없지만, 자원을 소유할 가능성이 있는 피어에게 검색 메시지를 전달함으로써 검색의 적중률을 높일 수 있다. 그러나 이 기법은 피어의 이탈 및 데이터의 삭제 등과 관련된 메시지를 전송하지 않기 때문에, 데이터의 삭제나 피어의 변화가 빈번하게 발생하는 네트워크 환경에는 부정확한 결과를 보일 수 있다[1,8].



(a) 자원이 있는 곳의 방향 저장 (b) Intelligent-BFS 동작모습

(그림 6) Intelligent-BFS 기법을 이용한 메시지 전송

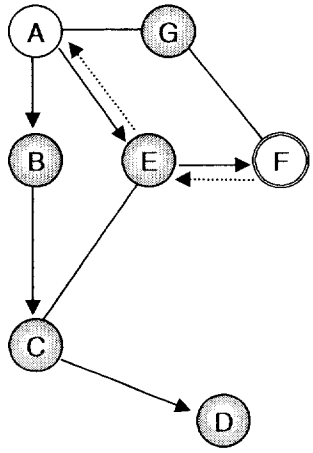
(2) APS(Adaptive Probabilistic Search) 기법

APS 기법은 각 피어로 하여금 검색 메시지의 송신과 검색에 대한 결과 메시지들의 수신이 있을 때마다 해당 자원에 대한 엔트리의 수치를 변경함으로써 이후의 검색에 대한 적중률을 높이게 하는 방법이다. 각 피어들은 표 1과 같은 이웃하는 피어들에 대한 엔트리 테이블인 APS 검색확률수치표를 가지고, 각 검색 메시지의 전송과 그 결과의 반환 시에 각 엔트리에 대한 수치를 변경한다.

이러한 수치를 토대로 Random walks와 유사하게 각 이웃하는 피어들의 수치를 반영한 성공률로서 메시지를 전송할 피어를 결정하지만 자원이 있을 가능성이 있는 피어들에게 전송하기 때문에 보다 많은 결과물을 얻을 수 있게 된다. 그러나 자원 검색의 지역성과 설정된 TTL을 초과한 노드들에 대한 검색이 어렵다는 문제점을 가지고 있다. 이 기법을 이용하여 메시지를 전송하는 과정은 그림 7과 같다[1,2].

<표 1> APS 검색확률수치 표

자원요청 경 로	피 어 초 기 값	자원 요청 후 검색확률수치	자원 반환 후 검색확률수치
A→B	30	20	20
B→C	30	20	20
C→D	30	20	20
A→E	30	20	40
E→F	30	20	40
A→G	30	30	30



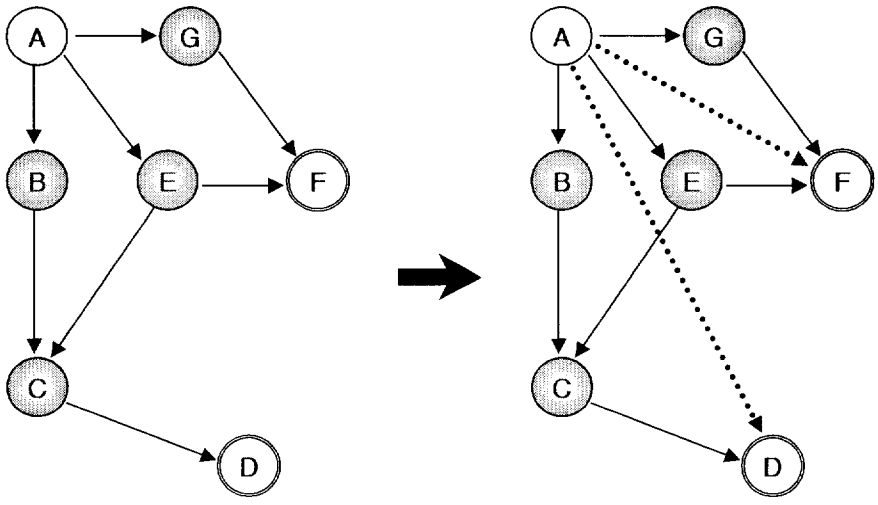
(그림 7) APS 기법을 이용한 메시지 전송

(3) GS(Gnutella with Shortcuts) 기법

GS 기법은 Gnutella 알고리즘을 기반으로 하며, 검색 이후 만족스러운 자원을 가지고 있는 피어를 Shortcut 피어로 등록하는 방법을 말한다. 검색을 원하는 피어는 브로드캐스트 기법을 이용하여 원하는 자원을 찾게 되

며, 이후 검색 결과를 받은 피어는 검색된 피어들을 Shortcut으로 사용가능하게 된다. 이러한 Shortcut 경로에 해당하는 피어들은 "Shortcut-List"라 하는 곳에 저장이 된다.

GS 기법을 구축하는 모습은 그림 8과 같이 피어 A에서 자원을 Gnutella 알고리즘으로 검색하여 피어 F와 D에서 자원을 찾게 되고, 찾은 자원이 있는 피어 F와 D를 Shorcut으로 목록에 등록하게 된다. Shortcut 목록에 저장되어 있는 Shortcut 피어에게 접속하여 원하는 자원을 찾지 못할 경우 인접한 피어에게 브로드캐스트 기법을 이용하여 재검색을 시도하게 되며, 검색조건을 만족하는 피어가 검색되었을 때 Shortcut 목록을 업데이트 한다. 이 기법은 Shortcut 목록에 있는 피어들에게 먼저 검색 메시지를 전송함으로써 보다 빠르게 자원을 검색할 수 있는 방식이다. 그러나 Shortcut 목록에 찾고자 하는 자원을 가지는 피어가 없는 경우 Gnutella 알고리즘의 검색 메시지 양과 전체 네트워크 트래픽을 효과적으로 줄이지 못한다[1,3].



(그림 8) GS 기법을 이용한 shortcuts 등록

III. 비구조적 P2P 네트워크에서 Shortcut을 이용한 적응적 확률 검색 기법

이 장에서는 본 논문에서 제안하는 APSS (The Adaptive Probabilistic Search Scheme using Shortcuts in Unstructured P2P Networks) 기법에 대해 기술하고, 알고리즘과 동작과정을 설명한다.

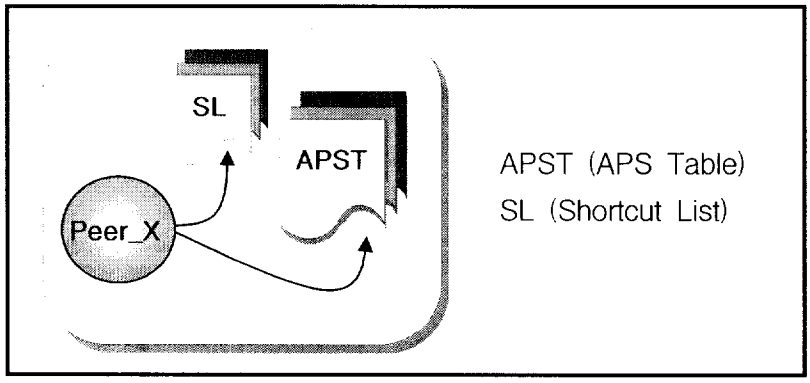
3.1 APSS 기법의 구조

본 논문에서는 비구조적 네트워크 기반에서 자원 검색 기법인 APSS 기법을 제안한다. APSS 기법은 자원을 소유할 가능성이 있는 피어에게 검색 메시지를 전달함으로써 검색의 적중률을 높이는 APS 기법을 기반으로 하여 자원을 가진 피어로 직접 연결할 수 있게 하는 Shortcut을 이용한 검색 기법이다.

자원 검색 시마다 동일한 경로를 따라갈 가능성이 크다는 APS 기법의 단점과 검색 시 TTL의 한계로 인해 발생하는 검색의 지역성 문제를 Shortcut을 이용하여 해결할 수 있고 이를 통해 자원 검색의 적중률을 높일 수 있다.

APSS 기법은 각 피어들의 이웃 피어들에 대한 검색확률수치를 가진 APS 검색확률수치 표 이외에 자원을 보유한 피어로의 Shortcut을 저장하는 Shortcut 목록을 가진다. 본 논문에서 APS 검색확률수치 표는 APST (APS Table)로 Shortcut 목록은 SL(Shortcut List)로 정의한다.

피어는 APS 검색확률수치 표와 Shortcut 목록을 관리하고, 이를 이용하여 자원을 검색한다. 본 논문에서 제안하는 APSS 기법의 피어 구조는 그림 9와 같다.



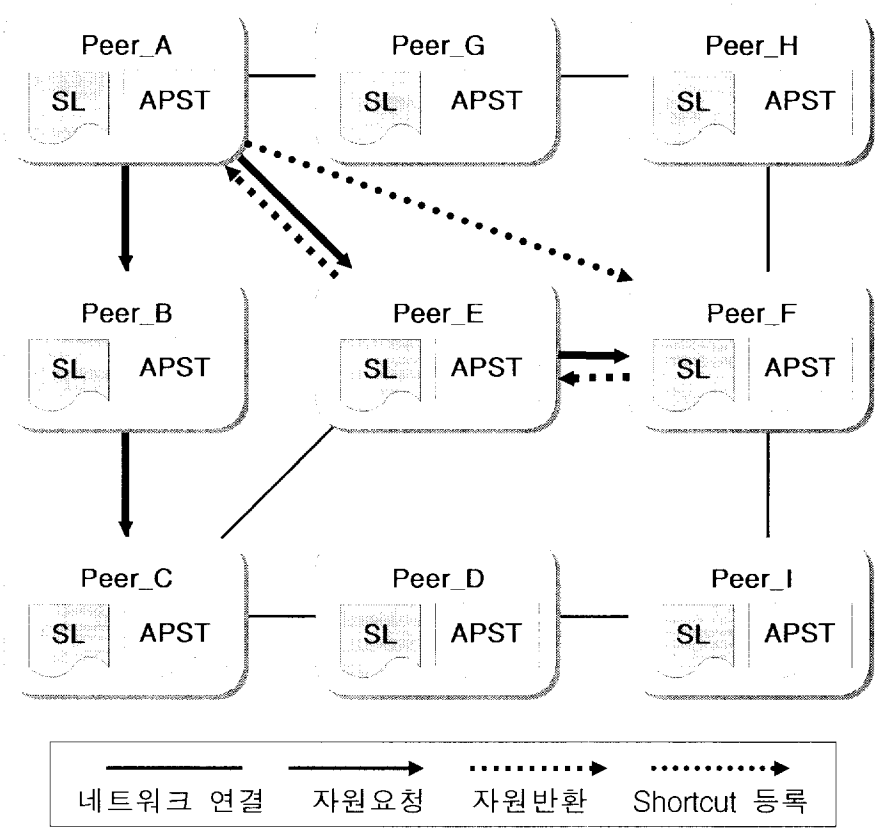
(그림 9) APSS 기법의 피어 구조

검색 메시지를 수신한 피어는 다른 피어에게 검색 메시지를 전송할 때 전송하려는 피어에 해당하는 엔트리의 값을 감소시키고, 결과 값이 피드백되는 경우에는 피드백이 발생한 피어에 해당하는 엔트리의 값을 증가 시키게 된다. 이후 동일한 검색메시지를 받는 피어는 인덱스에 저장되어 있는 해당 자원의 검색확률수치를 이용하여 검색 메시지가 전송되어질 이웃 피어를 결정하게 된다.

각 피어는 검색 메시지의 송신과 검색에 대한 결과 메시지들의 수신이 있을 때마다 해당 자원의 APS 검색확률수치 표와 Shortcut 목록의 엔트리 수치를 변경하여 이후 검색에 대한 적중률을 높인다.

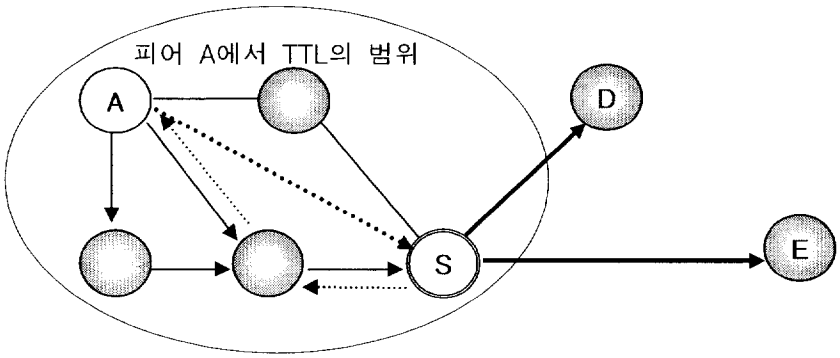
피어는 APS 검색확률수치 표에 있는 모든 피어들로 메시지를 전송하고 검색을 성공할 경우, 자원을 가진 피어는 검색 메시지를 전송한 피어의 Shortcut 목록에 추가된다. 이전에 검색을 성공한 피어는 Shortcut 목록에

저장되기 때문에 이후에 검색 메시지를 전송할 때는 다른 피어들을 거치지 않고 Shortcut을 사용하여 직접 연결할 수 있으므로 네트워크의 트래픽과 지연을 줄이고 검색 메시지도 감소된다. 또한 검색 결과에 의해서 Shortcut 목록의 엔트리 수치들이 바뀌므로 자원 검색을 할 때 마다 항상 동일한 경로를 따라가는 APS 기법의 문제가 해결된다. 그림 10은 피어 A가 네트워크 검색으로 자원을 가진 F를 Shortcut으로 등록하는 APSS 기법의 검색 구조이다.



(그림 10) APSS 기법의 검색 구조

그리고 검색시 TTL의 한계로 인해 발생하는 검색의 지역성 문제를 APSS 기법을 사용함으로써 해결할 수 있다.



(그림 11) Shortcut을 이용한 지역성 문제 극복

그림 11과 같이 A 피어에서 TTL 이내에 원하는 자원을 가진 피어가 존재 하지 않고, 해당 자원을 D나 E가 가지고 있는 경우, 기존의 APS 기법으로는 원하는 자원을 검색할 수 없다. 그러나 APSS 기법을 이용하면 A의 검색 요청이 Shortcut 피어(S)에서 실패하였을 경우, 해당 Shortcut 피어는 자신의 Shortcut 목록을 이용하여 검색 메시지를 전송한다.

Shortcut을 통한 메시지에 TTL을 부가하여 Shortcut으로 검색 메시지를 전송하면 해당 Shortcut 피어에서부터 다시 검색 메시지의 재전송이 발생하여 TTL의 감소가 일어나기 때문에 자신의 TTL 홉을 넘어서는 피어들을 검색하여 TTL의 한계로 인해 발생하는 검색의 지역성 문제를 해결할 수 있다. 또한 많은 피어들에게 검색 메시지를 전송할 수 있게 되어 다른 피어가 보유하고 있는 자원에 대한 정보가 많아지기 때문에 APS 기법보다 검색 적중률을 향상시킬 수 있다.

3.2 APSS 알고리즘

APSS 검색 알고리즘을 이용하여 피어들을 검색하고 APS 검색확률수치표와 Shortcut 목록에 자원을 가진 피어의 수치들을 갱신함으로써 자원 검색 적중률을 높인다.

3.2.1 APSS 검색 알고리즘

피어는 Shortcut 목록의 피어들에게 검색 메시지를 전송하고 검색이 끝나면 Shortcut 목록의 엔트리 수치를 갱신한다. 만약 Shortcut으로 검색이 실패하면, APS 검색확률수치 표에서 높은 검색확률수치를 가지는 피어들에게 검색 메시지를 전송한다. 이후 응답 메시지를 받으면 자원을 가진 피어는 Shortcut 목록에 삽입하고 Shortcut 목록의 엔트리 수치를 갱신하여 다음 검색에서 사용한다. APSS 검색 알고리즘은 그림 12와 같다.

```
PROCEDURE : APSS_Search
begin
    If find Message is entry of the SL
        SL_update( );
        // Update entry information in the SL;
    Else APST_update( );
        // Search Message broadcast to entry of the APST;
    Shortcut_update( );
    // Insert entry information in the SL;
end.
```

(그림 12) APSS 검색 알고리즘

3.2.2 SL 검색확률수치 갱신 알고리즘

APSS 기법은 Shortcut 목록의 엔트리에 새로운 Shortcut을 삽입할 때, 일정한 초기값을 부여한다. 그림 13의 알고리즘을 이용하여 검색 메시지의 전송 시마다 Shortcut 목록 엔트리의 수치를 감소시키고 검색에 성공한 엔트리에 대해서만 수치를 증가시킨다. 이때 그 수치가 0 이하로 내려가는 엔트리에 대해서는 Shortcut 목록에서 삭제하여 검색될 가능성이 낮은 Shortcut으로 가는 불필요한 메시지의 전송 가능성을 줄인다.

```
PROCEDURE : SL_update
  // Si : Shortcut which in the SL
begin
    Select Shortcut with Calculating Si's value;
    If send message to Si
        Si = Si - 1;
        If Si ==0
                Delete Si in the SL;
    If Si returns result
        Si = Si + 2;
end.
```

(그림 13) SL 검색확률수치 갱신 알고리즘

3.2.3 APST 검색확률수치 갱신 알고리즘

Shortcut 목록을 이용한 자원 검색이 실패하면, APS 검색확률수치 표를 이용하여 이웃 피어를 통한 검색을 수행한다. 만약 이 검색이 성공한다면, 자원을 가진 피어는 자원 검색 메시지를 전송한 피어의 Shortcut 목록에 삽입한다. 이때 Shortcut 목록에 삽입할 엔트리를 위한 공간이 없으면, 가장 낮은 수치를 가지는 엔트리를 삭제를 하고 그 자리에 삽입된다. 이 동작 알고리즘이 그림 14와 같다.

```
PROCEDURE : APST_update
  // Ni : neighbor node which is in the APST
begin
    While Ni != N's neighbor
        Ni = initial value;
    Select node with Calculating Ni's value;
    If send message to Ni
        Ni = Ni - 10;
    If Ni returns result
        Ni = Ni + 20;
        Insert Ni to SL of node N;
end.
```

(그림 14) APST 검색확률수치 갱신 알고리즘

3.2.4 Shortcut의 삽입 및 삭제 알고리즘

APS 검색확률수치 표의 수치로 이웃 피어를 통한 검색이 성공하면 자원을 가진 피어는 검색 메시지를 전송한 피어의 Shortcut 목록에 삽입한다. 이 과정에서 Shortcut 목록 엔트리 공간의 제한 때문에 그림 15의 알고리즘을 수행한다. Shortcut 목록에 엔트리 공간이 존재하면 자원을 가진 피어를 Shortcut으로 등록하고 초기값을 부여한다. 그러나 Shortcut 목록의 엔트리 공간이 부족하면 가장 낮은 수치를 가진 엔트리를 삭제하고 새로운 피어를 엔트리에 삽입하는 동작과정을 수행한다. 삽입된 피어의 초기값은 검색이 성공하면 높아지고 아니면 낮아지는 갱신과정을 거친다.

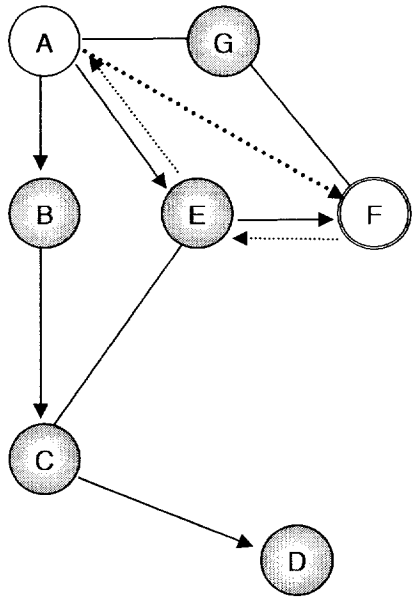
```
PROCEDURE : Shortcut_update
 // Emax : max # of the SL entry that node N can accommodate.
 // Ecur : # of entry in the SL in node N.
begin
    If Ecur + 1 ≤ Emax // if there's room
        Insert the peer which has the values we find to SL
        in node N;
        Initialize the value of tne inserted peer;
    Else // if there's no room
        Delete the Shortcut which has the lowest value in the
        SL;
        Insert the peer which has the values we find to SL
        in node N;
        Initialize the value of the inserted peer;
end.
```

(그림 15) Shortcut의 삽입 / 삭제 알고리즘

3.3 APSS 동작 시나리오

시나리오에서 Shortcut 목록의 엔트리 수를 2개, 초기값을 5로 가정한다. 그림 16은 APS 검색확률수치 표를 이용하여 검색 메시지를 전송할 피어를 B, E로 정하고, 피어 A에서 검색을 하는 과정을 보여준다. 이때 자원을 가진 F를 찾은 경우, F는 A의 Shortcut 목록 엔트리로 삽입된다.

APSS 알고리즘을 수행한 후 검색확률수치 변화는 표 2와 같고, 표 3은 검색확률 초기값으로 5가 설정된 F가 피어 A의 Shortcut 목록에 삽입되는 것을 나타낸다.



(그림 16) APSS 기법을 이용한 SL 삽입과정(Ⅰ)

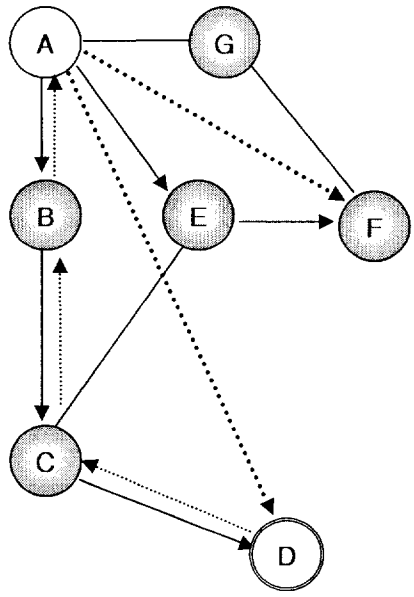
<표 2> 자원 F 검색의 APST 수치변화

APST			
자원요청 경 로	피 어 초 기 값	자원 요청 후 검색확률수치	자원 반환 후 검색확률수치
A→B	30	20	20
B→C	30	20	20
C→D	30	20	20
A→E	30	20	40
E→F	30	20	40
A→G	30	30	30

<표 3> 자원 F의 SL 삽입

SL			
Shortcut 경 로	Shortcut 초 기 값	자원 요청 후 Shortcut 값	자원 반환 후 Shortcut 값
A→F	5		

그림 17을 보면 자원 D를 검색하기 위해서 피어 A의 Shortcut 목록의 엔트리인 F로 검색 메시지를 전송했지만 그 검색이 실패한 경우 F에 대한 수치는 4로 떨어진다. 이후 APS 검색확률수치 표의 엔트리를 이용하여 원하는 자원이 있는 D를 찾게 된다. 자원을 검색하고 찾는 과정에서 APS 검색확률수치의 변화는 표 4와 같다. 검색된 자원 D는 피어 A의 Shortcut 목록의 엔트리로 삽입되며 이에 따르는 Shortcut 목록의 엔트리 변화는 표 5와 같다.



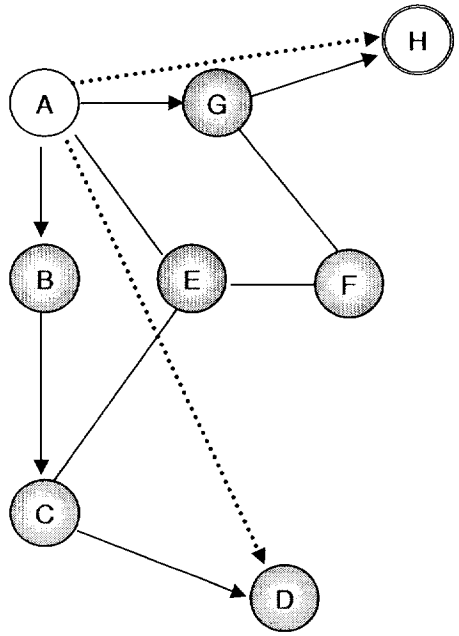
(그림 17) APSS 기법을 이용한 SL 삽입과정(II)

<표 4> 자원 D 검색의 APST 수치변화

APST			
자원요청 경 로	피 어 초 기 값	자원 요청 후 검색확률수치	자원 반환 후 검색확률수치
A→B	20	10	30
B→C	20	10	30
C→D	20	10	30
A→E	40	30	30
E→F	40	30	30
A→G	30	30	30

<표 5> 자원 D의 SL 삽입과 자원 F의 수치 변화

SL			
Shortcut 경 로	Shortcut 초 기 값	자원 요청 후 Shortcut 값	자원 반환 후 Shortcut 값
A→F	5	4	4
A→D	5		



(그림 18) APSS 기법을 이용한 SL 삽입과정(III)

자원 H를 찾기 위해서 피어 A의 Shortcut 목록 엔트리인 F와 D에게 검색 메시지를 전송하였으나 실패했을 경우 APS 검색확률수치 표를 이용하여 H를 찾게 된다. 그림 18은 위의 동작과정을 보여준다. 이 동작과정의 검색확률수치 변화는 표 6과 같다. 자원의 검색 결과를 Shortcut 목록에

삽입시에 A의 Shortcut 목록에 여유 공간이 없는 경우, Shortcut 목록 엔트리 중에서 가장 작은 수치를 가지는 자원을 삭제한다. 따라서 표 7과 같이 Shortcut 목록에서 가장 작은 수치인 4를 가지는 F를 삭제하고 H를 삽입하게 된다.

<표 6> 자원 H 검색의 APST 수치변화

APST			
자원요청 경 로	피 어 초 기 값	자원 요청 후 검색확률수치	자원 반환 후 검색확률수치
A→B	30	20	30
B→C	30	20	30
C→D	30	20	30
A→E	30	30	30
E→F	30	30	30
A→G	30	20	40
G→H	30	20	40

<표 7> 자원 H의 SL 삽입과 자원 F의 삭제

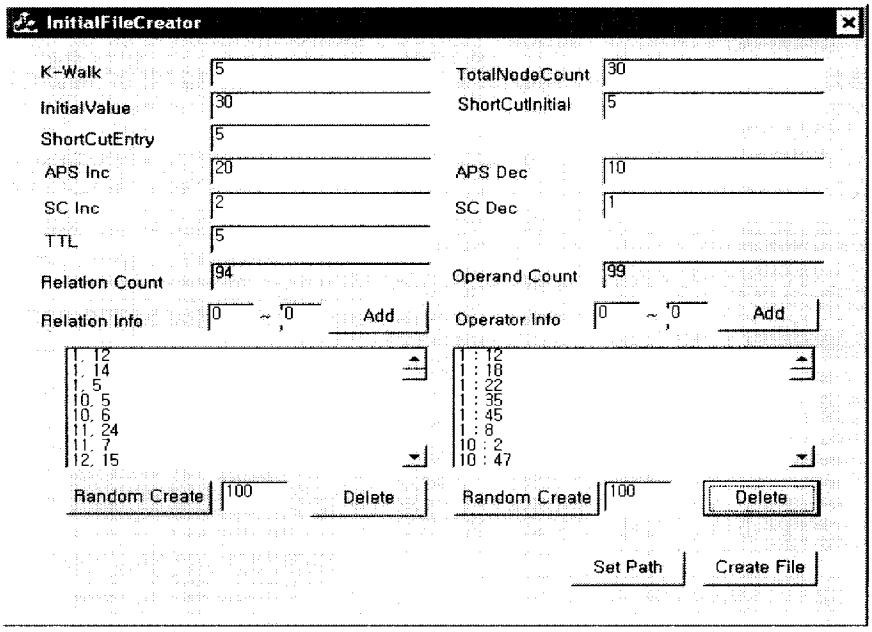
SL			
Shortcut 경 로	Shortcut 초 기 값	자원 요청 후 Shortcut 값	자원 반환 후 Shortcut 값
A→D	5	4	4
A→H	5		

IV. 성능 평가

본장에서는 제안한 APSS 기법의 성능 분석을 위해 시뮬레이션 환경과 비교인자들을 설명하고, 시뮬레이션 결과를 분석한다.

4.1 시뮬레이션 환경

본 논문에서 제안한 기법의 시뮬레이션을 위해 Windows 2000 플랫폼을 사용하였으며, 툴은 Microsoft사의 Visual C++ 6.0을 사용하였다. 시뮬레이션을 위한 실험 파일 생성기의 화면구성은 그림 19와 같다.



(그림 19) 실험파일 생성기

<표 8> 실험 파일 생성기의 파라미터

파라미터	기능
k-walk	k-walk의 k 수치
TotalNodeCount	총 노드 수
InitialValue	APS 검색확률수치 표의 초기 수치
ShortCutEntry	Shortcut 목록의 엔트리 개수
ShortCutInitial	Shortcut 목록의 초기 수치
APS Inc	APS 검색확률수치 표의 수치 증가값
APS Dec	APS 검색확률수치 표의 수치 감소값
SC Inc	Shortcut 목록의 수치 증가값
SC Dec	Shortcut 목록의 수치 감소값
TTL	Time To Live
Relation Count	Relation의 개수
Relation Data	Relation 실제 정보(개수만큼)
Operation Count	Operation의 개수
Operation Data	Operation 실제 정보(개수만큼)

초기화 파일 생성기에 사용된 파라미터는 표 8과 같다. 시뮬레이션을 위해 파일 생성기에 수치를 입력한 후, 그림 19의 [Set Path] 버튼을 눌러 파일 경로를 지정하고 [Create File] 버튼을 누르면 해당 경로에 초기 파일 노드의 네트워크 관계(Relation)와 검색 명령(Operation)이 랜덤하게 설정되어 파일이 생성된다.

본 시뮬레이션에서는 비교모델로 APS 기법을 사용하여 노드 수 대비 자원 검색 시간과 노드 수 대비 자원 검색 성공률에 대한 성능 평가를 수행한다.

또한 Shortcut 목록의 엔트리 수의 제한이 없다면, 검색 메시지의 개수가 엔트리 수만큼 늘어나서 과도한 트래픽을 초래하므로 이를 제한할 필요가 있다. [3]의 연구 결과인 "10개 이상의 Shortcut을 유지할 때는 더 이상의 성능향상이 없다"는 사실에 기인하여 본 논문에서 Shortcut 목록의 적절한 엔트리 크기를 찾기 위해서 시뮬레이션 하는 Shortcut 목록의 엔트리 개수는 1~10까지로 제한하여 실험한다.

APS 검색확률수치 표와 Shortcut 목록의 초기값은 30과 5로 설정하고, 각 결과는 20회의 시뮬레이션 수행 결과들의 평균값들을 사용한다. 노드 수 변화에 따르는 성공률과 자원검색시간을 얻기 위해서 노드 수를 30개~300개 까지 30 간격으로 나누어서 성능평가를 하고 Shortcut 엔트리 수는 3, 검색요청 수는 100, 자원종류 수는 50으로 한다. Shortcut 엔트리 개수에 따른 성공률을 찾기 위하여 노드 수는 150개로 고정하고 Shortcut 엔트리 수를 1부터 10까지 변화하면서 시뮬레이션을 수행한다. 본 시뮬레이션에서 사용한 성능 평가 인자 값들은 표 9와 표 10에 나타낸 것과 같다.

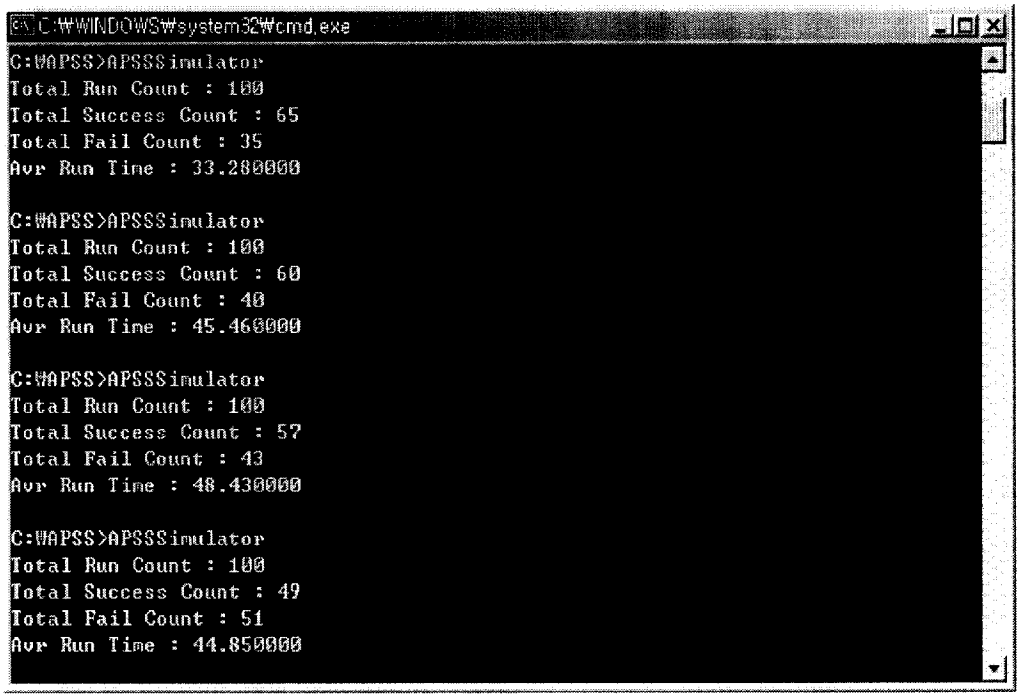
<표 9> 노드 수 변경에서 사용한 성능평가 인자

	노드 수	Shortcut 엔트리 수	검색요청 수	자원종류 수
노드 수 변경	30~300	3	100	50

<표 10> Shortcut 엔트리 수 변경에서 사용한 성능평가 인자

	노드 수	Shortcut 엔트리 수	검색요청 수	자원종류 수
Shortcut 엔트리 수 변경	150	1~10	100	50

실제 시뮬레이션 수행 화면은 그림 20과 같다. 수행한 후의 결과는 명령어 수, 성공률과 실패율 그리고 명령들을 수행한 평균 실행시간을 ms 단위로 보여준다.



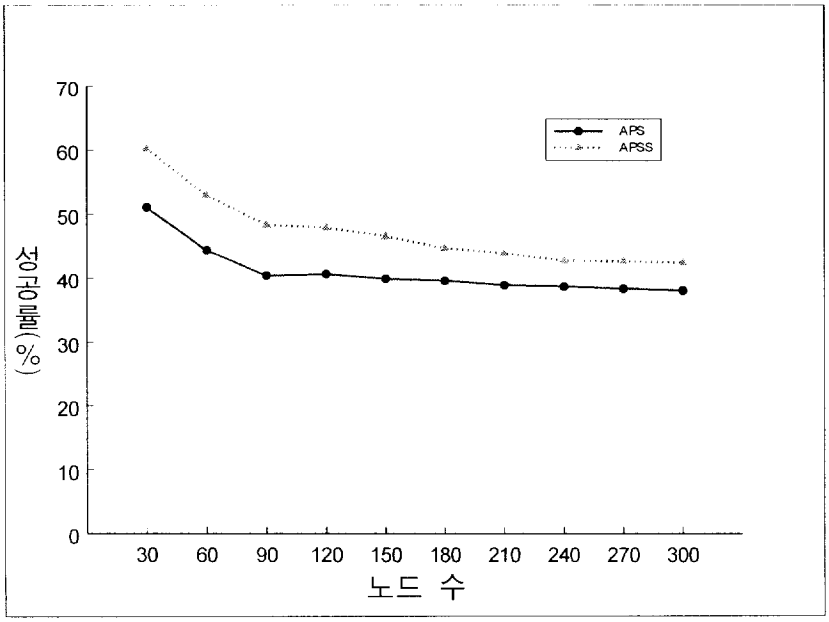
(그림 20) 시뮬레이션 수행 화면

4.2 시뮬레이션 결과 및 분석

표 11과 그림 21은 기존의 APS 기법과 제안한 APSS 기법의 자원 검색 성공률을 노드 수를 30개씩 증가시키면서 노드수 300개 까지 실험한 결과를 나타낸다.

<표 11> 노드 수 변화에 따른 자원 검색 성공률

노드 수	30	60	90	120	150	180	210	240	270	300
APS 기 법	51.05	44.40	40.45	40.70	39.95	39.65	38.95	38.75	38.40	38.10
APSS 기 법	60.25	52.95	48.35	47.90	46.55	44.70	43.90	42.75	42.65	42.45



(그림 21) 노드 수 변화에 따른 자원 검색 성공률

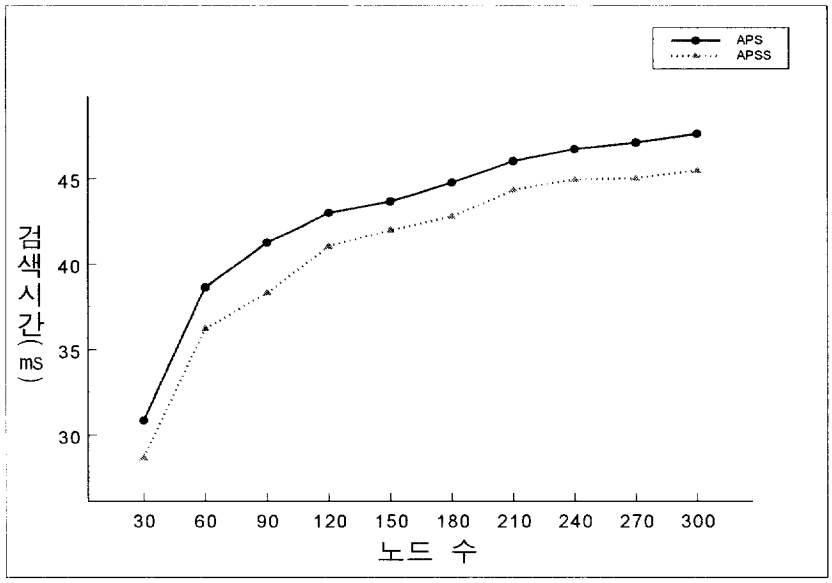
시뮬레이션 결과를 통해 APSS 기법이 APS 기법에 비해 자원 검색 성공률이 평균 6%정도 향상됨을 알 수 있다. 그림에서 노드 수가 30개에서 90개 일때 약 9%정도 APS 기법 보다 더 높은 자원 검색의 성공률을 보이며 그 이상의 노드 수를 가질 때는 노드 수가 증가하더라도 APS 기법 보다 평균 5% 높은 자원 검색 성공률을 유지함을 알 수 있다.

APSS 기법은 Shortcut을 이용함으로써 기존 APS 기법이 가졌던 단점들을 해결하여 APS 기법 보다 많은 피어들에게 검색 메시지를 전송할 수 있게 되고, 다른 피어가 보유하고 있는 자원에 대한 정보가 많아져 검색 적중률을 향상시킬 수 있음을 알 수 있다. 특히 노드 수가 적을수록 보유한 자원에 대한 정보를 많이 가질 확률이 높기 때문에, 노드 수가 적을 때의 성공률이 노드 수가 많을 때 보다 비교적 높은 것을 알 수 있다.

표 12와 그림 22는 APS 기법과 APSS 기법의 자원 검색 시간을 비교한 것으로 평균적으로 APSS 기법의 자원 검색 시간이 APS 기법보다 평균 2ms 빠름을 알 수 있다.

<표 12> 노드 수 변화에 따른 자원 검색 시간

노드 수	30	60	90	120	150	180	210	240	270	300
APS 기 법	30.85	38.65	41.25	43.02	43.71	44.78	46.04	46.77	47.17	47.68
APSS 기 법	28.67	36.20	38.32	41.04	42.01	42.81	44.36	44.96	45.04	45.50



(그림 22) 노드 수 변화에 따른 자원 검색 시간

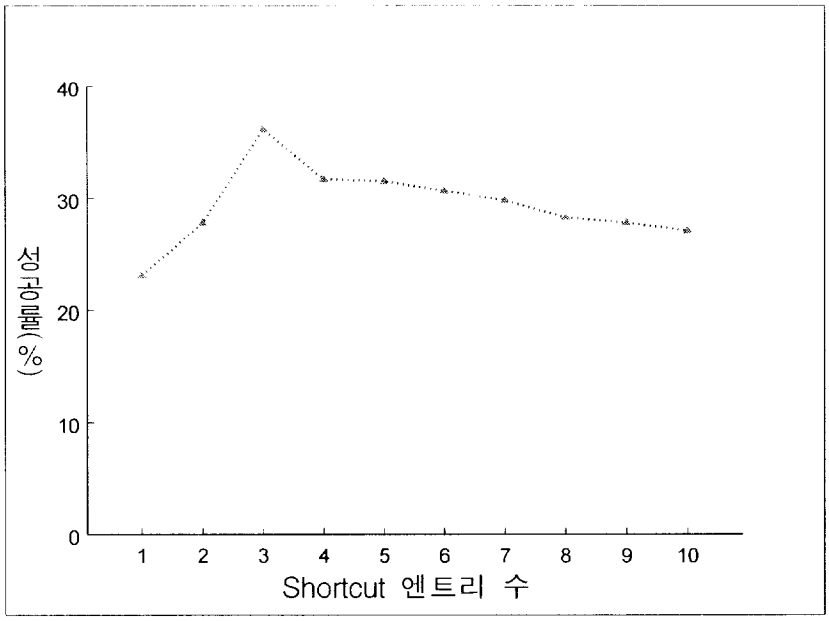
자원 검색 시간이 평균 2ms 정도 빠르기는 하지만 실험환경이 아닌 실제 네트워크에서 검색 시간을 고려해 본다면 APS 기법의 자원 검색 시간과 차이가 거의 없다고 할 수 있다.

그러나 기존의 APS 기법보다 Shortcut 목록을 추가로 가지는 APSS 기법에서 Shortcut을 검색하고 갱신하는 Shortcut 목록 자체의 오버헤드를 고려한다면 APSS 기법이 자료를 좀 더 빠르게 검색할 수 있기 때문에 APS 기법과 비슷한 시간을 기록하게 되는 것으로 유추해 볼 수 있다.

표 13과 그림 23은 APSS 기법에서 가지는 Shortcut 목록의 엔트리 수 변경에 따르는 자원 검색 성공률을 보여준다. Shortcut 엔트리 수를 1~10까지 변경하면서 실험한 결과로 Shortcut 엔트리 수가 3일 때 가장 좋은 자원 검색 성공률을 가지며 그 이상은 엔트리 수치를 증가시키더라도 자원 검색 성공률을 일정하게 유지하게 된다.

<표 13> Shortcut 엔트리 수에 따른 자원 검색 성공률

Shortcut 엔트리 수	1	2	3	4	5	6	7	8	9	10
성 공 률	23.1	27.8	36.2	31.7	31.5	30.7	29.8	28.3	27.8	27.1



(그림 23) Shortcut 엔트리 수에 따른 자원 검색 성공률

Shortcut 목록의 엔트리 수가 많이 늘어나면 자원을 찾을 성공률도 지속적으로 올라갈 것으로 기대하였으나 실험 데이터의 경우 명령의 최대 개수가 100개이므로 하나의 노드에서 검색 명령이 자주 있지 않다. 그래서 Shortcut 목록의 엔트리 수를 계속 증가시켜도 최대 자원 검색 성공률을 가지는 엔트리 수 이상에서는 자원 검색 성능의 차이가 크지 않다.

Shortcut 목록의 엔트리 수가 많아도 검색 명령의 개수가 엔트리 수만큼 되지 않으면 나머지 공간이 사용되지 않으므로 자원 검색 성공률에 영향을 끼치지 못하게 되지만 만약 한 노드에서 잦은 검색이 있다면 엔트리 수가

많은 것이 성공률에 도움이 될 것이다. Shortcut 엔트리 수는 노드 수나 명령어 등의 영향을 받으므로 Shortcut 엔트리 수를 고정시키는 어렵다.

시뮬레이션 결과처럼 Shortcut 목록의 엔트리 수는 노드의 수나 명령에 따라 가변적이지만 동일한 환경에서 최대의 자원 검색 성공률을 가지는 엔트리가 존재하므로 그 수치를 적절하게 검색에 이용한다면 자원 검색의 적중률을 높일 수 있다.

이상의 시뮬레이션 결과를 분석해 보면 본 논문에서 제안한 APSS 기법이 APS 기법 보다 자원 검색 시간 측면에서는 평균 2ms, 자원 검색 성공률 측면에서는 평균 6%가 높음을 알 수 있다. 그리고 실험 환경에서 APSS 기법의 Shortcut 목록 엔트리 수는 3일 때 자원 검색의 성공률이 가장 높음을 알 수 있다 .

따라서 APSS 기법으로 비구조적 네트워크에서 검색을 한다면 기존의 APS 기법보다 비슷한 검색시간을 사용하면서 자원 검색의 적중률을 향상시킬 수 있다.

또한 노드와 명령의 수에 적절한 Shortcut 목록 엔트리 수를 설정하여 자원을 검색한다면 보다 높은 자원 검색 성능을 얻을 수 있을 것이다.

V. 결론 및 향후 과제

최근 P2P에 대한 관심이 많아지면서 여러 분야에서 다양한 연구가 이루어지고 있다. P2P 네트워크 모델이란 각 피어들이 가지고 있는 자원 또는 제공하는 서비스를 피어들 상호간의 직접적인 연결을 통하여 교환할 수 있도록 지원하는 네트워크 모델을 의미한다. 이러한 P2P 네트워크 모델은 클라이언트-서버 네트워크 모델처럼 특정 피어가 자원을 관리하여 주고 있지 않기 때문에 많은 피어들에게 분산되어 있는 여러 자원들을 관리하는 기법이 필요하게 되며, 더불어 P2P 네트워크 구조에 맞는 피어 관리 기법 및 자원 검색 기법 등을 필요로 한다.

본 논문에서는 비구조적 P2P 네트워크 기반에서 사용되는 자원 검색 기법들을 분석하고 새로운 검색 기법인 APSS 기법을 제안하였다.

기존에 제시되어 있는 비구조적 네트워크에서의 효율적인 자원 검색 기법 중 하나인 APS 기법은 네트워크 트래픽의 과도한 증가문제와 여러 가지 색인을 유지하는 기법들에서 나타나는 높은 색인 유지비용을 해결하고 있으며 검색 성능이 뛰어나다고 알려져 있지만 자원검색 시마다 동일한 경로를 따라갈 가능성이 크다는 단점과 검색시 TTL의 한계로 인해 발생하는 검색의 지역성 문제를 해결하지 못한다.

제안하는 APSS 기법은 이웃 피어들에 대한 정보만을 가지고 자원이 있을 가능성이 있는 피어에게 검색 메시지를 전달함으로써 검색의 적중률을 높이는 APS 기법에 자원을 가진 피어로 직접 연결할 수 있게 하는 Shortcut을 이용한 검색 기법이다. APSS 기법을 사용함으로써 APS 기법의 단점을 보완하고 검색시 TTL의 한계로 인해 발생하는 검색의 지역성 문제를 해결하였다.

이와 같이 검색을 위해 추가적으로 Shortcut을 사용함으로써 자원의 검색 성능을 높일 수 있었다. 시뮬레이션을 이용한 성능 측정을 통해 제안한 APSS 기법이 기존의 APS 기법보다 평균 자원 검색 속도가 약 2ms 향상되고, 자원 검색의 성공률이 대략 6% 정도 향상됨을 알 수 있다.

이 결과로 비구조적 네트워크에서 APSS 기법을 사용하여 자원을 검색한다면 APS 기법과 비슷한 자원 검색 시간을 사용하면서 자원 검색의 적중률을 높일 수 있음을 알 수 있다. 그러므로 동일한 네트워크 환경이라면 APS 기법보다 APSS 기법이 자원 검색 성능이 우수할 것이다.

향후 연구과제는 Shortcut 목록의 엔트리 수를 어떻게 설정하는가에 따라 자원 검색의 정확도와 성능이 더 향상될 수 있으므로 많은 연구 및 실험을 통해서 네트워크의 노드와 명령의 수에 적절한 Shortcut 목록의 엔트리 수를 찾는 것이다.

참고문헌

[1] D.Tsoumakos and N. Roussopolulos, "Analysis and Comparison of P2P Search Methods," Technical Report, University of Maryland, Dept. of Computer science, Nov. pp.1-17, 2003.

[2] D.Tsoumakos and N. Roussopolulos, "Adaptive Probabilistic Search for Peer-to-Peer Networks," Proc. Of the 3rd IEEE International Conference on P2P Computing, Sep. pp.1-9, 2003.

[3] K.sripanidkulchai and Bruce Maggs, Huio Zhang, "Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems", Proc. Of the IEEE INFOCOM, Mar. pp.2166-2176, 2003.

[4] D.Tsoumakos and N. Roussopolulos, "A Comparison of Peer to Peer Search Methods", International Workshop on the Web and Databases (WebDB), June. pp.1-6, 2003.

[5] Q. Lv, S. Ratnasamy, and S. Shenker, "Can Heterogeneity Make Gnutella Scalable? ", 1st International Workshop on Peer-to-Peer Systems, pp.1-6, 2002.

[6] Eytan Adar and Bernardo A. Huberman, "Free Riding on Gnutella", Internet Ecologies Area Xerox Palo Alto Research Center Palo Alto, CA 94304. pp.1-14, 2000.

[7] Q.Lv, P.Cao, E. Cohen, K. Li, and S. Shenker, "Search and Replication in Unstructured Peer-to-Peer Networks", ICS, pp.258-269, 2002.

[8] V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-Yazti, "A Local Search Mechanism for Peer-to-Peer Network," Proc. Of the 11th International Conference on Information and Knowledge Management, Nov. pp.1-8, 2002.

[9] E. Choen, A. Fiat, H. Kaplan, "Associative Search in Peer to Peer Networks:Harnessing Latent Semantics", Proc. Of the 3rd IEEE International Conference on P2P Computing, Jan. pp.1261-1271, 2003.

[10] D. S. Milojicic. et. al., "Peer-to-Peer Computing", HP Technical Report, HP Labora-to ries, Mar, pp.1-51, 2002.

[11] K. Aberer and Hauswirth, "An Overview on Peer-to-Peer Information Systems." Workshop on Distributed Data and Structures(WDAS-2002), pp.1-14, 2002.

[12] Y. Guo, K. Suh, J. Kurose, and D. Towsley "A Peer-to-Peer On-Demand Streaming Service and Its Performance Evaluation", Proc. Of the IEEE INFOCOM, Nov. pp.649-652, 2003.

[13] D. Barkai, "An Introduction to Peer to-Peer Computing," Developer Update Magazine, Intel Coporation, Feb, pp.1-7, 2000.

[14] 김경인, 김영진, 엄영익 "순수 P2P 네트워크 환경에서 에이전트 이주를 위한 자원 예약 기반 동적 부하 균형기법", 정보처리학회 논문지 A 제 11-4권 제 4호, pp.257-266, 2004.

[15] 김인숙, 강용혁, 엄영익. "순수 P2P 환경에서 분산된 위치 정보를 이용한 자원 검색 기법", 정보처리과학회 논문지: 정보통신 제 31권 제 6호, pp.573-581, 2004.

[16] 김영진, 엄영익. "순수 P2P 네트워크 환경을 위한 효율적인 피어 연결 기법", 정보과학회 논문지 : 정보통신 제 31권 제 1호, pp.11-19, 2004.

[17] 김명섭, 강훈정, 홍원기, "Flow Grouping을 통한 P2P 트래픽 분석 방법에 관현 연구", Proc. of KNOM 2003 Conference, pp.210-218, 2003.

[18] 김영진, 김문정, 김응모, 엄영익, "순수 P2P 네트워크 환경에서 프락시-서버 할당을 위한 동적 피어 선정 기법 설계 및 구현", 정보처리학회 논문지, VOL. 10-D NO. 01. pp.153- 016. 2003.

[19] 이승은, 진명희, 김경석. "효율적인 키워드 검색을 지원하기 위한 계층적 확장 Chord", 한국정보과학회 가을학술발표 논문집, Vol. 31, No. 2, pp.574-576, 2004.

[20] 김영진, 엄영익. "순수 P2P 네트워크 환경 유지를 위한 피어 목록 관리 기법" 한국정보처리학회 춘계학술발표 논문집 제 10권 제 1호, pp.999-1002, 2003.

[21] Napster. http://www.napster.com/.

[22] Gnutella. http://www.gnutellaforums.com/.

감사의 글

대학원을 입학할 때 멀게만 느껴지던 졸업이었는데 벌써 그 문턱에 서 있습니다. 길고도 짧았던 대학원 생활을 통해 제가 조금은 성장했음을 느낍니다. 부족한 저를 따뜻함 맘으로 채워주신 모든 분들께 진심으로 감사드립니다.

이 논문이 완성되기까지 격려와 가르침으로 저의 부족한 면을 채워 주시고, 학업의 길로 인도해주신 지도교수님 윤성대 교수님께 진심으로 감사드립니다. 논문 심사를 위해 바쁘신 와중에도 세심한 조언을 해주신 여정모 교수님, 정순호 교수님께도 깊은 감사드립니다. 또한 학부부터 석사과정까지 많은 가르침을 주신 박만곤 교수님, 김영봉 교수님, 김창수 교수님, 박승섭 교수님, 이경현 교수님, 박홍복 교수님, 신상욱 교수님께도 감사의 마음을 전합니다.

늘 힘이 되는 친구들, 격려를 아끼지 않는 선배님들과 대학원 생활하는 동안 많은 도움을 주셨던 순환 선배, 현호 선배, 석범 선배, 상일 선배, 윤종찬 선생님, 이경미 선생님, 소영이 언니, 은아 선배, 정애, 지영, 미나에게도 고마움을 전하며, 그 외 연구실의 모든 식구들에게도 감사드립니다.

삶의 기쁨과 행복을 함께 해주는 사랑하는 가족에게 진심으로 감사하며 부족한 저의 논문을 바칩니다.

2005. 6. 21.

정 귀 녀 드림