

순수 P2P 환경에서 분산된 위치 정보를 이용한 자원 검색 기법

(An Efficient Resource Discovery Mechanism based on
Distributed Location Information in Pure P2P Environments)

김 인 숙[†] 강 용 혁[†] 엄 영 익^{**}

(In-suk Kim) (Yong-hyeog Kang) (Young Ik Eom)

요약 최근 인터넷의 급속한 성장과 초고속 통신망의 구축으로 인하여 다양한 멀티미디어 서비스들이 제공되고 있다. 기존의 멀티미디어 서비스들은 클라이언트/서버 모델을 기반으로 구축되어 과부하로 인한 네트워크의 다운이나 다운로드 속도 저하 등의 문제점들이 발생하였다. 이를 해결하기 위해 P2P 네트워크의 개념이 등장하였으며 이 개념은 인터넷을 통해 급속히 확산되었다. 본 논문에서는 순수 P2P 환경으로 구축된 멀티미디어 서비스 환경에서의 자원 검색 기법을 제안한다. 제안 기법에서는 각 호스트에게 자신이 보유하고 있는 자원의 복제본에 대한 위치 정보와 다른 호스트로부터 최근 요청을 받은 자원에 대한 위치 정보를 유지하게 하여 기존 순수 P2P 환경에서의 검색 기법보다 빠른 검색 속도를 제공하며 적은 트래픽량을 발생시키는 장점을 가진다. 또한 제안 기법은 자원의 위치 정보를 분산하며 검색 결과의 반환 경로를 검색 요청 시의 경로와 다르게 하여 기존의 순수 P2P 환경에서의 검색 기법에서 네트워크가 불안정할 때 검색 결과가 유실되는 문제점을 해결한다.

키워드 : P2P 네트워크, 자원 검색, 멀티미디어 서비스

Abstract Recently, the rapid growth of Internet technology and the deployment of high speed networks make many kinds of multimedia services possible. Preexisting multimedia services are designed based on the client/server model which has the problems such as the 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 through Internet environments. In this paper, we propose a pure peer-to-peer network based resource discovery mechanism for multimedia services. In the proposed scheme, each host maintains the location information of resources which are recently requested by other hosts as well as that of the replicas of local resources. The proposed scheme has the faster response time and incurs the smaller traffic than the preexisting discovery schemes in pure peer-to-peer network environments. Also, by decentralizing the location information and differentiating the reply path, our proposed scheme can solve the search result loss problem that occurs when the network is unsettled.

Key words : P2P Network, Resource Discovery, Multimedia Service

1. 서론

최근 인터넷의 급속한 성장과 초고속 정보통신망의 구축으로 인하여 다양한 멀티미디어 서비스들이 제공되

고 있다. 이러한 멀티미디어 서비스 환경은 대부분이 몇 개의 대용량 서버를 구축하여 서비스를 제공하는 클라이언트/서버 모델로 구성되어 있다. 그러나 기존의 클라이언트/서버 모델은 서버의 과부하로 인하여 네트워크의 다운이나 다운로드 속도 저하 등의 문제점들이 발생하였다. 이러한 문제점을 해결하고 더욱 수월하게 개인이 원하는 콘텐츠를 얻을 수 있도록 하기 위해 등장한 P2P 네트워크는 고급화되어가는 PC 성능과 초고속 통신망의 확산과 함께 인터넷을 통해 확산되고 있다[1-3].

P2P 네트워크 환경이란 클라이언트 상호간 분산 및

· 이 논문은 2002년도 한국학술진흥재단의 지원에 의하여 연구되었음
(KRF-2002-041-D00420)

[†] 비회원 : 성균관대학교 정보통신공학부
easy@ece.skku.ac.kr

yhkang1@ece.skku.ac.kr

^{**} 종신회원 : 성균관대학교 정보통신공학부 교수
yieom@ece.skku.ac.kr

논문접수 : 2003년 6월 21일

심사완료 : 2004년 8월 13일

협력이라는 새로운 개념의 네트워크라 할 수 있다. P2P 네트워크의 분산 개념은 효율성의 극대화를 목적으로 P2P 네트워크가 분산 컴퓨팅을 통해 하드웨어 자원을 공유하는 것을 의미하며 협력 개념은 클라이언트 간에 상호 협력해 P2P 프로그램을 통해 공유한 자원을 검색하는 것을 의미한다. 따라서 P2P 네트워크에 연결된 사용자가 많아질수록 네트워크의 가치가 커지는 효과를 가질 수 있다. 이러한 효과를 멀티미디어 서비스에 응용한 대표적인 프로그램으로 Napster와 Gnutella 등이 있다[2,4].

P2P 모델 중 자원의 검색을 위해 브로드캐스트(broadcast) 요청 기법을 사용하는 Gnutella는 자원이 검색된 경로로 검색 결과를 라우팅한다. 따라서 검색 요청에 대한 응답 메시지가 돌아가야 할 경로 내의 노드가 연결을 끊는 경우, 해당 검색 결과를 전달할 방법이 없다. 이런 문제를 보완하기 위해 강력한 백본망을 통해 검색을 하도록 하는 KaZaA가 등장하였지만, 근본적인 문제 해결은 이루어지지 않았다[5].

본 논문은 멀티미디어 서비스를 제공하는 호스트들이 순수 P2P 네트워크로 구성되는 환경에서 자원 검색 메커니즘을 제안한다. 순수 P2P 환경이므로 중앙 서버의 부하 집중 현상을 방지할 수 있으나 중앙 인덱스 서버가 존재하지 않는 상태이므로 자원 검색의 속도를 높이기 위해 각 자원의 위치 정보를 분산된 테이블로 각 호스트가 유지하게 한다. 즉 제안기법은 멀티미디어 서비스를 제공하는 각 호스트에게 보유한 자원의 복제본들에 대한 위치 정보와 다른 호스트로부터 최근에 검색 요청을 받은 자원에 대한 위치 정보를 테이블로 유지하게 하여 복수 자원의 신속한 검색을 지원한다. 또한 제안 기법은 검색 과정의 경로와 무관하게 검색 결과를 반환하여 자원 검색 정보의 유실을 방지하며 각 호스트에서 각 요청에 대한 재방문 여부를 체크하여 불필요한 트래픽량을 감소하였다.

본 논문의 2장에서는 P2P 네트워크의 개념과 P2P 네트워크에서의 기존 자원 검색 기법들에 대해 소개한다. 3장에서는 본 논문에서 제안하는 자원 검색 기법을 설명하고, 시나리오를 통해 제안 기법의 구체적인 동작 과정에 대해 알아본다. 그리고 4장에서는 제안 기법의 성능평가를 위해 수행한 시뮬레이션의 결과를 보인다. 마지막으로, 5장에서는 본 논문의 결론 및 향후 연구 과정에 대해서 기술한다.

2. 관련연구

본 장에서는 P2P 네트워크의 개념과 기존의 P2P 네트워크에서 자원 검색 기법들에 대해 기술한다.

2.1 P2P 네트워크

P2P 네트워크란 각 호스트들이 프로세싱 파워, 저장 공간, 콘텐츠 등의 하드웨어 자원들의 일부를 공유하는 네트워크를 말한다. 클라이언트/서버 네트워크에서 클라이언트는 그들의 어떠한 자원도 공유할 수 없는데 반해 P2P 네트워크에서는 임의의 호스트가 자신의 자원들을 공유할 수 있다는 차이점을 가진다. 즉 P2P 네트워크란 임의의 호스트가 필요에 따라서 서버 또는 클라이언트로 동작할 수 있는 네트워크라고 할 수 있다[6].

P2P 네트워크는 크게 네트워크 내에 중앙 서버를 갖는 하이브리드 P2P(hybrid P2P) 네트워크와 중앙 서버를 갖지 않는 순수 P2P(pure P2P) 네트워크로 구분될 수 있으며 이들의 구성 형태는 그림 1과 같다.

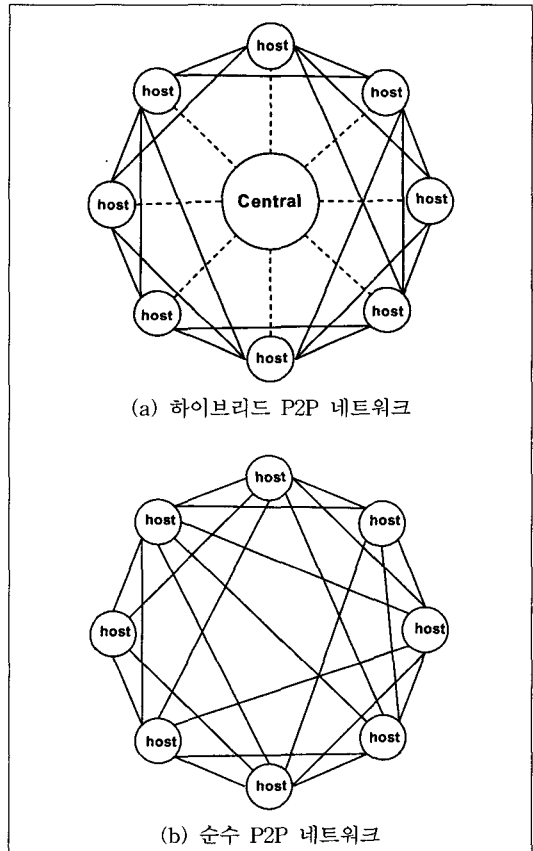


그림 1 P2P 네트워크의 구성

그림 1(a)에서 보여주는 것과 같이, 하이브리드 P2P 네트워크는 중앙에 인덱스 서버가 존재하며 이러한 인덱스 서버가 검색에 사용되는 형태의 네트워크이다. 그림 1(b)에서 보여주는 것과 같이 순수 P2P 네트워크는 어떠한 서버도 존재하지 않으며 호스트 간에 직접 검색 과정을 수행하는 네트워크이다[6,7].

2.2 P2P 네트워크에서의 검색 기법

P2P 네트워크에서 효율적인 검색을 위해 제안되고 있는 검색 기법들은 일반적으로 다음과 같이 분류될 수 있다. 이 기법들 중 본 논문에서 제안하는 기법은 브로드캐스트 요청 모델과 유사한 기법이라 할 수 있다.

(1) 중앙 집중식 디렉토리 모델(centralized directory model)

이 모델은 하이브리드 P2P 네트워크에서 사용되는 검색 모델로 P2P 네트워크 내에 공유되는 모든 콘텐츠의 인덱스 정보를 중앙 디렉토리에 유지하는 모델이다. 따라서 임의의 호스트가 콘텐츠를 검색하고자 할 때, 검색을 요청한 호스트는 먼저 반드시 중앙 디렉토리로 연결하여야 한다. 해당 호스트가 중앙 디렉토리에 연결되면 이 디렉토리로부터 적당한 콘텐츠를 가진 호스트에 대한 정보를 받은 후, 직접 콘텐츠를 가진 호스트로 접속하여 콘텐츠를 받게 된다. 그림 2는 중앙 집중식 디렉토리 모델을 보여준다.

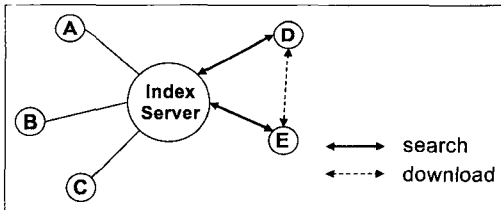


그림 2 중앙 집중식 디렉토리 모델의 검색 과정

그림 2에서 임의의 호스트 D는 콘텐츠 검색을 요청하기 위해 먼저 인덱스 서버에 접속한 후, 이 서버로부터 받은 정보를 기반으로 콘텐츠를 소유한 호스트 E에 직접 접속하여 콘텐츠를 받는다.

이 모델은 P2P 네트워크 내에 공유되는 모든 콘텐츠의 인덱스 정보를 유지해야 하는 디렉토리 서버가 필요하다. 이러한 모델은 검색 요청이 많아질수록 디렉토리 서버에 부하가 커지는 문제점이 발생한다. 또한 콘텐츠의 양이 증가할수록 보다 큰 저장 공간을 요구하게 되어 확장성에 제약을 갖는다. 대표적인 시스템으로는 Napster가 있다[7].

(2) 브로드캐스트 요청 모델(broadcast requests model)

이 모델은 순수 P2P 네트워크에서 사용되는 검색 모델로 P2P 네트워크 내에 공유되는 모든 콘텐츠의 인덱스 정보를 유지하는 중앙 서버가 존재하지 않는 모델이다. 따라서 임의의 호스트가 콘텐츠를 검색하고자 할 때에는 직접 연결된 피어들에게 검색 요청 메시지를 브로드캐스트 하게 되며 이 메시지가 해당 콘텐츠를 가진 호스트에게 도달하게 될 때까지 계속 전달되게 된다. 그림 3은 브로드캐스트 요청 모델을 보여준다.

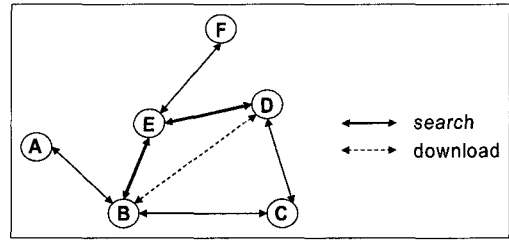


그림 3 브로드캐스트 요청 모델의 검색 과정

그림 3에서 보여주는 것과 같이 임의의 호스트 D는 콘텐츠 검색을 요청하기 위해 먼저 직접 연결된 호스트 C와 E에게 자원 검색 요청 메시지를 브로드캐스트한다. 이 메시지를 받은 호스트 C와 E는 자신이 해당 콘텐츠를 소유하지 않을 경우 다시 직접 연결된 다른 호스트들에게 자원 검색 요청 메시지를 전달하게 된다. 이 메시지를 받은 호스트 B가 해당 콘텐츠를 소유할 경우, 호스트 B는 검색 요청 메시지가 온 경로를 따라 검색 응답 메시지를 호스트 D에게 전달하게 된다. 이후 호스트 D는 직접 호스트 B에 접속하여 콘텐츠를 받게 된다.

그림 3에서 만일 호스트 B가 검색 응답 메시지를 전송하는 중에 호스트 E와의 연결에 문제가 생긴다면, 호스트 B와 D 간에는 다른 경로가 존재함에도 불구하고 응답 메시지를 유실하여 자원 검색을 실패하는 결과를 초래할 수 있다는 단점이 존재한다.

이 모델은 호스트의 존재 여부를 알기 위한 메시지들과 자원을 검색하기 위한 메시지들을 브로드캐스트하므로 네트워크 대역폭을 많이 사용하게 되며, 확장성에 제약을 가지므로 회사 네트워크 같이 작은 네트워크에서 주로 사용된다. 대표적인 시스템으로는 Gnutella가 있다 [7-9].

(3) 도큐먼트 라우팅 모델(document routing model)

이 모델은 순수 P2P 네트워크에서 사용되는 검색 모델로, 공유되는 콘텐츠의 ID를 기반으로 자원을 검색하는 모델이다. 각 호스트는 콘텐츠 ID에 가장 유사한 ID를 가진 호스트 쪽으로 콘텐츠를 라우팅하기 때문에 임의의 호스트가 콘텐츠를 검색하고자 할 때에는 요청은 해당 콘텐츠 ID와 가장 유사한 ID를 가지고 있는 피어로 간다. 이 과정은 콘텐츠의 복제본이 발견될 때까지 반복되고, 콘텐츠가 발견된 후 해당 콘텐츠는 요청을 발생한 피어로 전송되고, 라우팅에 참가한 각 호스트는 로컬에 콘텐츠의 복제본을 갖게 된다.

이 모델은 큰 규모의 네트워크에 매우 효율적이지만, 콘텐츠를 요청하기 위해서는 사용자가 미리 도큐먼트 ID들을 알아야 하며, 브로드캐스트 요청 모델에 비해 검색의 구현이 어렵다는 단점을 가지고 있다. 또한 상호간의 연결이 없을 시 커뮤니티가 독립적인 내부 그룹들

로 나누어지는 고립(islanding) 문제를 일으킬 수 있다. 대표적인 시스템으로는 Chord, CAN, Tapestry, Pastry 가 있다[7,8].

3. 순수 P2P 기반 자원 검색 기법

본 장은 논문에서 제안하는 시스템의 구성과 각각의 자료 구조, 알고리즘, 동작 시나리오에 대해 살펴보겠다.

3.1 시스템 구성

본 논문에서는 멀티미디어 호스트들로 구성된 순수 P2P 네트워크 환경을 가정하며, 방화벽(firewall)과 NAT에 대한 시스템 고려는 배제하고 멀티미디어 호스트간의 자원 검색을 위해 에이전트를 사용하는 기법을 제안한다.

본 논문에서 제안하는 기법의 시스템 구성은 그림 4에서 보인다.

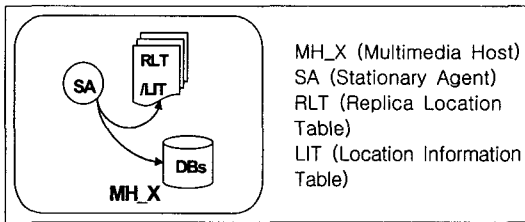


그림 4 멀티미디어 호스트 구성

그림 4에서 멀티미디어 호스트(MH)는 자신이 소유한 콘텐츠 정보를 유지하는 DB를 가지며, DB 및 자원 관리 테이블(RLT, LIT)을 관리하는 고정 에이전트(SA)를 가진다. RLT는 자신이 소유한 자원에 대한 정보와 자신이 소유한 자원의 복제본을 소유한 다른 MH들에 대한 정보를 유지한다.

LIT는 자신이 소유하지 않지만 최근에 검색 요청이 들어온 자원을 소유한 다른 MH들에 대한 정보를 유지한다. SA는 사용자의 UI로부터 수신한 검색 요청에 근거하여 자원 질의 메시지(QM:Query Message)를 생성하고, 자원 질의 메시지를 통해 해당 자원을 요청한다. 자원 질의 메시지에 메시지의 경로(M_Path:Message Path)가 쌓여서 메시지가 루프에 빠지는 것을 방지하고 자원을 신속하게 찾을 수 있도록 돕는다.

3.2 자료 구조

본 논문의 제안 기법에서 사용되는 자원 관리 테이블들과 메시지의 구조는 그림 5와 그림 6에서 보인다.

R_Id(Resource Identifier), R_Type(Resource Type), R_Attr(Resource Attribute)는 RLT, LIT에서 사용되며, 자원을 구분하는 키워드를 구성하여 사용자가 자원 검색을 요청했을 때 사용하게 된다. 그림 5(a)에서 필드

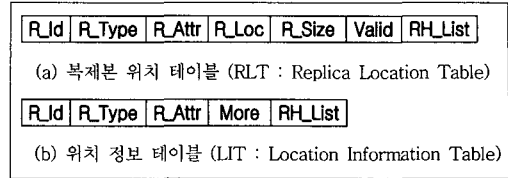


그림 5 테이블의 구조

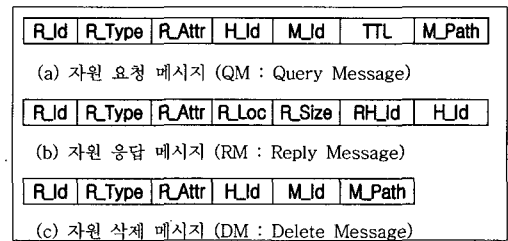


그림 6 메시지 구조

R_Loc(Resource Location)에는 자원의 물리적인 주소가 저장되고, 필드 Valid에는 자원 정보의 생성/삭제를 나타내는 값이 들어간다. 필드 RH_List(Resource Host List)는 해당 자원의 복제본을 보유한 MH들의 Id 목록이다. 그림 5(b)의 필드 More에는 RH_List에 MH들의 추가 여부를 나타내는 값이 들어간다.

그림 6(a)는 자원을 검색할 때 사용되는 요청 메시지인데, H_Id(Home Identifier)에는 사용자에게 검색 결과를 돌려주기 위해, 검색을 발생시킨 home의 Id가 저장되고 M_Id(Message Identifier)에는 각 메시지를 구분해주는 Id가 들어간다. 필드 TTL(Time To Live)과 M_Path (Message Path)는 QM이 P2P 네트워크 상에서 루프가 생성되거나 무한히 반복되는 것을 방지하도록 한다. 그림 6(b)는 자원을 발견했을 때 home에게 검색 결과를 돌려주는 응답 메시지이며 RH_Id에는 자원이 발견된 MH의 Id가 들어간다. 그림 6(c)는 자원이 삭제됐을 때 다른 MH들에게 해당 자원이 삭제되었음을 알리는 메시지이다.

3.3 알고리즘

본 절에서는 자원의 검색과 자원의 관리를 위한 SA의 알고리즘들을 소개한다.

알고리즘 1은 SA에게 사용자가 자원을 요청한 후부터 검색된 자원 위치 정보가 home으로 전달되어 RLT에 반영될 때까지 home에 위치한 SA의 동작 과정을 보인다. 사용자가 검색된 자원의 다운로드를 완료하더라도 SA의 작업은 종료되지 않는다. 이는 SA가 돌아오는 검색 결과에 대해 자원 위치 정보인 RH_List를 작성하여 해당 자원의 다음 번 검색 속도를 향상시키기 위한 것이다.

```

when a SA receives a request from a user
  find an entry E in RLT with the specified keywords;
  if (there exists E) {
    send E to the user;
    send QM to each MH in RH_List;
  }
  else {
    create and insert a new entry E' into RLT with the
    specified keywords;
    find an entry E in LIT with the specified keywords;
    if (there exists E) {
      send QM to each MH in RH_List;
      E' in RLT = E in LIT ;
    }
    else
      send QM to the neighbor MHs;
  }
when the SA receives RM from another SA
  update the RH_List information in RLT;
    
```

(알고리즘 1) Home에서 자원 요청을 받았을 때 SA의 작업 절차

```

when SA receives a QM from another SA
  if (M Id of QM has already been registered in current MH)
    discard the QM;
  else {
    find an entry E in RLT with the specified keywords;
    if (there exists E) {
      send RM to the home;
      send QM to each MH in RH_List;
      if (RH_List is not fully occupied)
        insert H_Id into RH_List;
    }
    else {
      find an entry E in LIT with the specified keywords;
      if (there exists E) {
        send QM to each MH in RH_List;
        if (RH_List not fully occupied)
          insert H_Id into RH_List;
      }
      else {
        send QM to neighbor MHs;
        create a new entry E' into LIT with the specified
        keywords;
        insert H_Id into E'. RH_List;
        E'. More in LIT = 1;
      }
    }
  }
    
```

(알고리즘 2) Home이 아닌 다른 MH 상에서의 SA 작업 절차

알고리즘 2는 home이 아닌 다른 MH 상에 위치하는 SA들이 자원의 검색에 참여할 때의 동작 과정을 보인다. home에서 다른 MH 상으로 보내진 QM들은 각

MH 상의 SA에 의해서 처리가 된다. SA는 도착한 QM의 M_Id를 이용해 이전에 해당 QM이 현 MH에 전달되었는지 여부를 검사한다. 이것은 불필요한 QM이 계속적으로 발생하는 것을 막기 위함이다. SA는 해당 QM이 중복되어 전달되었으면 폐기하고, 현 MH에 처음으로 전달되었으면 다음 과정을 진행한다. SA는 해당 자원의 위치 정보가 RLT, LIT에 있는지 검사하고 없으면 가까이 이웃한 MH로 QM을 보낸다. 만약 현 MH의 RLT나 LIT에 자원의 위치 정보가 있으면 위치 정보가 발견된 테이블의 RH_List에 명시된 MH들로 QM을 발생시킨다. 전자의 경우에는 차후 해당 자원이 home에 추가가 되면 이에 대한 자원 위치 정보를 유지하기 위해 미리 LIT를 작성하고 RH_List에 home의 Id만을 적어놓는다.

RLT나 LIT의 RH_List 상 위치로 QM을 전송할 때에는 각 MH마다 QM 하나를 보내는데 이때 QM에는 메시지가 지나간 경로 정보가 포함된다.

```

when the states of RLT or LIT are changed
  if (E.Valid in RLT == 1) // resource is added
    send QM to local MH;
  else if (E.Valid in RLT == 2) { // resource is deleted
    send DM to each MH in RH_List;
    delete E;
  }
  if (E.More in LIT == 1) // more RH_List is needed
    send QM to each MH in RH_List;
    
```

(알고리즘 3) 자원의 생성/삭제에 대한 SA의 작업 절차

알고리즘 3은 사용자가 직접 MH에 새로운 자원을 추가하는 경우와 자원을 삭제하는 경우에 이를 반영하기 위한 SA의 동작 과정을 보인다. 사용자가 다른 MH로부터 자원을 다운로드하는 경우 또한 새로운 자원이 MH의 DB에 추가가 되지만 자원 검색 과정에서 이에 대한 자원 관리 테이블 상의 추가 작업은 처리되지 못하여 이 알고리즘에서 다루지 않는다.

SA는 알고리즘 2에서 다른 MH로부터 QM을 받았으나 해당 자원의 위치 정보가 RLT나 LIT에 없었을 경우, 차후 해당 자원이 home에 추가될 것을 감안하여 해당 자원에 대해 LIT를 작성하고 RH_List에 H_Id를 기록했다. 이 자원에 대해 RH_List의 채워지지 않은 부분에 대한 처리는 알고리즘 3에서 보인다. SA가 주기적으로 RLT와 LIT를 모니터링하면서 이를 발견하였을 때 LIT의 RH_List에 기록된 home으로 QM을 보낸 후, 돌아오는 결과로 나머지 RH_List를 작성한다.

3.4 동작 시나리오

다음 시나리오는 한 M11이 요청하는 자원이 M14,

MH_7, MH_8, MH_15, MH_16에 있으며 이 중 MH_15에서 해당 자원을 다운로드하는 경우를 보인다.

그림 7과 같이 MH_1에서 SA는 자신의 RLT, LIT를 검색했으나 해당 자원에 대한 위치 정보를 얻지 못해 이웃한 MH_5로 QM을 보낸다. MH_5 또한 자원에 대한 정보가 없으므로 MH_5 상의 SA는 다시 이웃한 MH_10으로 QM을 보내는데, MH_10의 LIT에서 해당 자원에 대한 위치 정보가 발견된다. MH_10의 SA는 LIT의 RH_List 상 MH들(MH_4, MH_15)에게 QM을 보낸다.

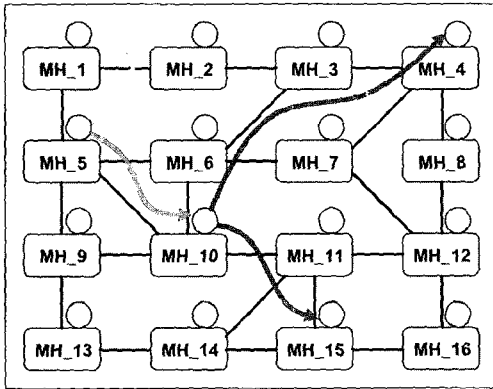


그림 7 자원의 검색

MH_4와 MH_15 상의 SA는 QM을 받고 이 메시지의 home인 MH_1에게 RM을 통해 각각 자신의 RH_Id를 보낸다. MH_4와 MH_15는 RLT 상 RH_List의 MH들에게 QM을 보내게 되는데, MH_4는 MH_7과 MH_8로 QM을 보내고, MH_15는 MH_8과 MH_16으로 QM을 보낸다. 그 결과 MH_8은 MH_4와 MH_15로부터 QM을 중복해서 받게 된다. 그러나 SA가 중복 메시지에 대해 검사를 하므로 늦게 도착하는 메시지가 무시된다. 이 과정을 그림 8에서 보인다.

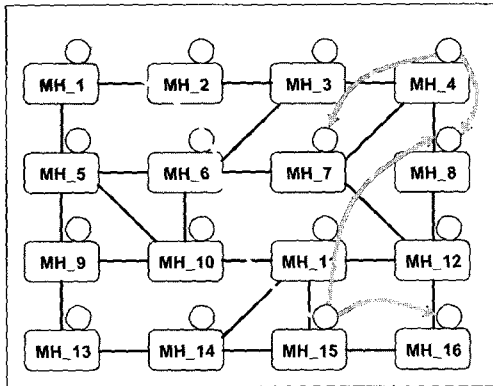


그림 8 자원 위치 정보의 전달

MH_1은 검색된 MH의 목록 중 하나(MH_15)를 선택해서 파일을 다운받을 수 있다. 각 MH 상의 SA들은 일정 기간마다 RLT와 LIT를 점검해 RH_List 값을 갱신한다. MH_5의 경우 그림 7의 자원 검색 과정에서 해당 파일에 대한 정보가 전혀 없었으므로 LIT 상 RH_List에 MH_1을 추가했다. 이를 MH_5의 SA가 발견하여 MH_1에게 RH_List 정보를 요구하고 이 정보를 MH_1이 제공하게 된다. 이 과정을 그림 9에서 보인다.

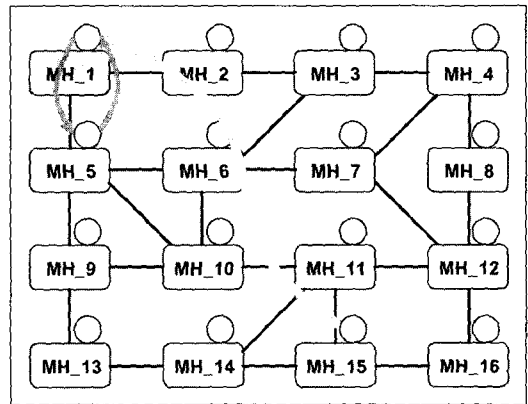


그림 9 자원의 다운로드 및 LIT에 자원 위치 정보 추가

4. 성능 평가

4.1 시뮬레이션 환경

본 논문에서 제안된 기법을 시뮬레이션하기 위해 Simlib[10]을 사용하였으며, 비교 모델로는 브로드캐스트 요청 모델을 사용하였다. 본 시뮬레이션에서는 노드 수 대비 평균 자원 검색 시간, 노드 수 대비 트래픽 발생량과 자원 관리 테이블 크기 대비 평균 자원 검색 시간에 대한 성능 평가를 수행하였다. 각 결과는 5회의 시뮬레이션 수행 결과들의 평균값을 사용하였다.

본 논문에서 제안하는 기법에서는 사용자의 자원 검색 요청이 발생한 MH의 SA가 다른 MH들에게 QM을 전송한 후 자원 관리 테이블들을 검색한다.

본 시뮬레이션에서 사용한 성능 평가 인자 값들은 표 1에서 보인다.

본 논문에서는 3가지의 경우를 시뮬레이션하였으며, 첫 번째 시뮬레이션에서는 노드 수 대비 평균 자원 검색 시간, 노드 수 대비 발생 트래픽량을 시뮬레이션하기 위해 노드 수를 성능 평가의 매개변수로 사용하였다. 비교 모델인 브로드캐스트 요청 모델은 소규모 네트워크에 적합하기 때문에 일반적으로 P2P 네트워크의 구성이 수백 개의 노드를 사이에서 이루어지므로, 노드 수의 범

표 1 시뮬레이션에서 사용한 성능 평가 인자

	노드 수	RLT의 RH_List	LIT의 엔트리 크기	검색 요청 수	자원 종류 수
노드 수 변경	30~430	25	35	1000	200
RLT의 RH_List 크기 변경	300	4~28	35	1000	200
LIT의 엔트리 크기 변경	300	25	5~105	1000	200

위를 500개 미만인 30~430으로 하였다. 두 번째 시뮬레이션에서는 RLT의 RH_List 크기 대비 평균 자원 검색 시간을 시뮬레이션하기 위해 RLT의 RH_List 크기의 범위를 4~28로 변경하여 시뮬레이션하였다. 세 번째 시뮬레이션에서는 LIT의 엔트리 크기 대비 평균 자원 검색 시간을 시뮬레이션하기 위해 LIT의 엔트리 크기를 5~105로 설정하였다.

4.2 시뮬레이션 결과

본 논문에서 제안하는 모델은 RLT 테이블과 LIT 테이블의 내용을 기준으로 검색을 하게 된다. 비교 모델은 어떤 캐쉬 테이블도 가지고 있지 않은 브로드캐스트 요청 모델을 가정하였다.

(1) 노드 수 변경

그림 10은 노드 수가 증가하는 경우에 기존의 브로드캐스트 요청 기법과 본 논문의 제안 기법에서의 평균 자원 검색 시간을 비교한 것이다. 평균 자원 검색 시간은 자원의 검색 요청이 발생한 후 자원이 발견된 각 MH에서 보낸 QM이 home에 도착한 시간을 평균한 값이다.

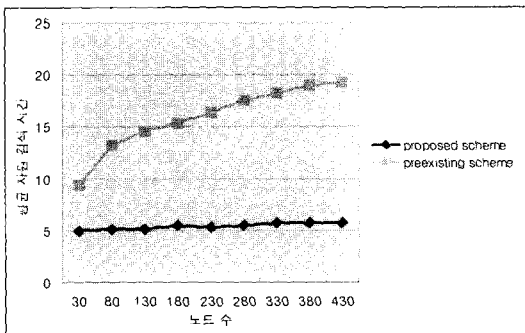


그림 10 노드 수 변경에 따른 평균 자원 검색 시간

그림 10에서 제안 기법과 기존 브로드캐스트 요청 기법 모두 노드 수가 증가함에 따라 자원 검색 시간이 증가하는 것을 볼 수 있으나, 제안 검색 기법은 기존 브로드캐스트 요청 기법에 비해 자원 검색 시간이 크게 증

가하지 않는 것을 볼 수 있다. 이는 검색이 진행됨에 따라 제안 기법의 RLT 및 LIT가 작성되면서 제안 기법의 자원 검색 속도를 향상시키기 때문이다. 결국 제안 검색 기법은 노드 수가 증가할수록 기존 브로드캐스트 요청 기법에 비해 자원 검색 속도에 있어서 좋은 성능을 보인다.

그림 11은 노드 수가 증가하는 경우에 기존의 브로드캐스트 요청 기법과 본 논문의 제안 기법에서의 자원 검색을 위해 발생하는 트래픽량을 비교한 것이다.

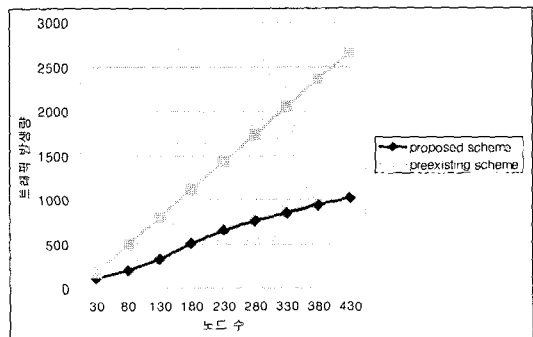


그림 11 노드 수 변경에 따른 트래픽 발생량

그림 11에서 두 모델 모두 노드 수가 증가함에 따라 자원 검색을 위한 트래픽량이 늘어나는 것을 볼 수 있으나, 시간이 지남에 따라 제안 모델의 트래픽량의 증가분이 감소하는 것을 볼 수 있다. 이는 노드 수가 증가할수록 검색이 진행되면서 제안 기법의 RLT와 LIT에 작성되는 자원의 위치 정보가 증가함에 따라 자원의 위치를 검색하기 위해 이웃 노드로 전송되는 메시지의 양이 줄기 때문이다. 결국 제안 검색 기법은 노드 수가 증가하면 기존 브로드캐스트 요청 기법에 비해 적은 트래픽을 발생시킨다.

따라서 제안 기법은 노드 수를 증가시킬 때 검색 속도나 트래픽 발생량에 더 좋은 성능을 보이므로 기존 브로드캐스트 요청 기법보다 확장성에 덜 제약을 받는다.

(2) RLT의 RH_List의 크기 변경

그림 12는 제안 기법에서 RLT의 RH_List 크기가 증가하는 경우에 기존의 브로드캐스트 요청 기법과 본 논문의 제안 기법에서의 평균 자원 검색 시간을 비교한 것이다. RH_List는 다른 MH 상에 존재하는 동일 자원에 대한 위치 정보를 유지하고 있으므로 자원의 검색 속도를 단축시킨다.

그림 12에서 제안 기법의 평균 자원 검색 시간이 기존 브로드캐스트 요청 기법에 비해 적게 걸리는 것을 볼 수 있으며 제안 모델에서 RLT의 RH_List 크기가

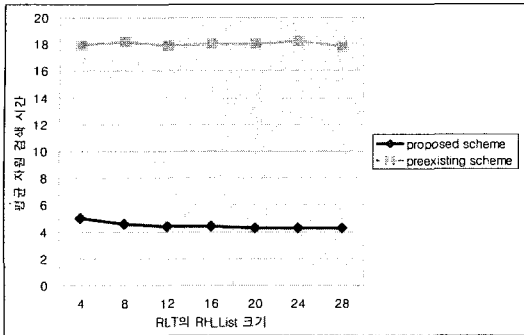


그림 12 RLT의 RH_List 크기 변화에 따른 평균 자원 검색 시간

증가함에 따라 평균 자원 검색 시간이 줄어드는 것을 볼 수 있다. 이는 RLT의 RH_List 크기가 증가할수록 각 MH에서 보유한 자원의 복사본들에 대한 위치 정보가 증가하므로 자원 검색 속도를 높일 수 있기 때문이다. 결국 제안 검색 기법은 RLT의 RH_List가 증가할수록 기존 브로드캐스트 요청 기법에 비해 자원 검색 속도에 있어서 좋은 성능을 보인다.

(3) LIT의 엔트리 크기 변경

그림 13은 제안 기법에서 LIT의 엔트리 크기가 증가하는 경우에 기존의 브로드캐스트 요청 검색 기법과 본 논문의 제안 기법에서의 평균 자원 검색 시간을 비교한 것이다. LIT의 엔트리는 로컬에 존재하지는 않으나 최근 자원 검색 요청이 들어온 자원에 대해 위치 정보를 유지하고 있으므로 자원의 검색 속도를 빠르게 한다.

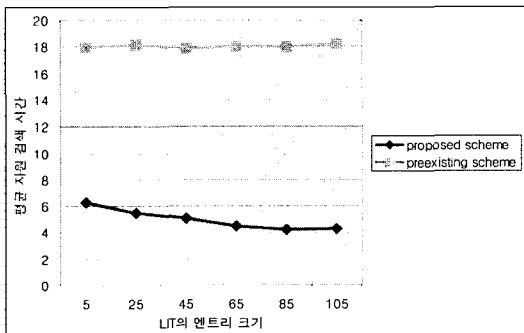


그림 13 LIT의 엔트리 크기 변화에 따른 평균 자원 검색 시간

그림 13에서 제안 기법의 평균 자원 검색 시간이 기존 브로드캐스트 요청 기법에 비해 적게 걸리는 것을 볼 수 있으며 제안 모델에서 LIT의 엔트리 크기가 증가함에 따라 평균 자원 검색 시간이 줄어드는 것을 볼 수 있다. 이는 LIT의 엔트리 크기가 증가할수록 해당 MH

를 방문한 QM들에 대해 유지하는 위치 정보가 증가하므로 자원 검색 속도를 높일 수 있기 때문이다. 결국 제안 검색 기법은 LIT의 엔트리가 증가할수록 기존 브로드캐스트 요청 기법에 비해 자원 검색 속도에 있어서 좋은 성능을 보인다.

5. 결론

현재 대부분의 멀티미디어 서버들에서 나타나는 트래픽의 집중 현상에 대한 대안으로 P2P 컴퓨팅을 이용한 해결 방안이 모색되고 있다. P2P 컴퓨팅 환경에서의 효율적인 멀티미디어 서비스 제공을 위해서는 자원 검색 기법이 매우 중요하다. 순수 P2P 네트워크 환경에서의 일반적인 자원 검색 기법으로는 브로드캐스트 요청 기법과 도큐먼트 라우팅 기법이 있다. 그러나 브로드캐스트 요청 기법은 많은 네트워크 대역폭을 요구하는 문제점이 있으며, 도큐먼트 라우팅 기법은 사전에 콘텐츠의 ID를 알아야 하고 네트워크가 고립되는 문제점이 있다.

본 논문에서는 순수 P2P로 구성된 멀티미디어 서비스 환경에서의 검색기법을 제안하고 알고리즘과 간단한 동작 시나리오, 성능 평가 결과를 통해 이를 살펴보았다. 제안 기법은 자원 검색을 위해 발생하는 트래픽과 자원 검색 시간을 줄이기 위해, 로컬에 있는 자원들의 복제본들의 위치 정보와 최근 다른 호스트로부터 자원 요청이 들어왔던 자원들의 위치 정보를 각 호스트에 분산된 테이블로 유지하였다. 제안 기법은 네트워크가 불안정할 때 유실되는 검색 결과를 최소화하기 위해 검색된 결과의 반환 경로를 검색 요청 시의 경로와 다르게 하였다. 또한 자원의 검색을 위해 발생하는 메시지가 불필요하게 증가하지 못하도록 하기 위해 자원 검색을 위한 메시지의 호스트 재방문 여부를 확인하여 기존의 브로드캐스트 요청 기법보다 메시지의 양을 감소시켰다.

참고 문헌

- [1] IBM, "Peer to Peer Networking: Addressing Scaling Issues in Peer to Peer Architecture," <http://www.research.ibm.com/smartnetwork/peer.html>
- [2] "개발자 세계의 화두 P2P", http://network.hanbitbook.co.kr/view.php?bi_id=138&pg=5
- [3] White Paper: Intel Corp., "Peer to Peer Computing: P2P File-Sharing at work in the Enterprise," http://www.intel.com/ebusiness/pdf/prod/peertopeer/p2p_edgesvc.pdf, 2001.
- [4] Oram, A., "Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology," O'Reilly, 2001.
- [5] Aitken, D. et al, "Peer-to-Peer Technologies and Protocols," <http://ntrg.cs.tcd.ie/undergrad/4ba2.02/p2p/>
- [6] Schollmeier, R., "A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer

- Architectures and Applications," Proceedings of the IEEE 2001 International Conference on Peer-to-Peer Computing (P2P2001), 2001.
- [7] Milojicic, D. et al. "Peer-to-Peer Computing," HP Labs Technical Report HPL-2002-57, 2002.
- [8] Yang, B. and Garcia-Molina, H., "Improving Search in Peer-to-Peer Networks," In 22nd International Conference on Distributed Computing Systems (ICDCS 2002), 2002.
- [9] Jovanovic, M., Annexstein, F. and Berman, K., "Scalability Issues in Large Peer-to-Peer Networks - A Case Study of Gnutella," University of Cincinnati Technical Report, 2001.
- [10] Law, A. and Kelton, W., Simulation Modeling and Analysis, Mc Graw Hill, 2001.



김 인 속

2002년 성균관대학교 전기전자 및 컴퓨터 공학부 학사. 2004년 성균관대학교 전기전자 및 컴퓨터 공학과 석사. 현재 LG전자 연구원. 관심분야는 P2P Computing, 이동 에이전트 시스템



강 용 혁

1996년 성균관대학교 정보공학과 학사
1998년 성균관대학교 정보공학과 컴퓨터 공학전공 석사. 2003년 성균관대학교 정보통신공학부 박사. 현재 극동대학교 경영학부 전자상거래 전공 교수. 관심분야는 분산 시스템, 이동 컴퓨팅 시스템

Mobile Ad-hoc networks

엄 영 익

정보과학회논문지 : 정보통신
제 31 권 제 1 호 참조