RDF 온톨로지로 구성한 Tapestry상의 Range Query

한종욱○, 한동윤○, 유영호†, 김경석‡
부산대학교 컴퓨터공학과○, 신라대학교 컴퓨터정보공학부†, 부산대학교 정보컴퓨터공학부‡
{jwhan○, dyhan○, yhyu†, gimgs0‡}@asadal.cs.pusan.ac.kr

Range Query on Tapestry organized by RDF Ontology

Jong-wook Han○, Dong-yun Han○, Young-ho Yu†, Kyong-sok Kim‡
Dept. of Computer Engineering. Pusan National University○
Division of Computer and Information Engineering. Silla University†
Division of Computer Science and Engineering. Pusan National University‡

요 약

현재 컴퓨팅은 하나의 개인 컴퓨터에서 이런 개인 컴퓨터들이 하나로 묶인 네트워크 컴퓨팅 형태로 발전 되었고, 앞으로 더욱 네트워크 컴퓨팅 중심으로 발전될 것은 자명한 사실이다. 그 가운데 인터넷과 더불어 P2P(Peer-to-Peer) 시스템이 발전되었다. 인터넷을 중심으로 정보통신 분야는 눈부신 발전을 하였지만, 이제 인터넷을 통한 발전도 한계를 맞고 있다. 너무 많은 정보 가운데 사용자가 원하는 정보를 어떻게 찾을 것인가란 문제를 두고 현재 시멘틱 웹[1]을 제시하여 이를 해결하고자 하는 노력들이 있다. 이러한 문제점은 P2P 시스템에서도 동일하게 나타난다. 이를 해결하고자 시멘틱 웹의 요소인 RDF(Resource Description Framework)[2]를 이용한 P2P 시스템[3][4]이 제안되었다. 하지만 DHT(Distributed Hash Table)를 이용한 P2P 시스템의 특성상 연관된 자료라도 어디에 배치될지 알 수 없다. 이러한 특성을 가진 시스템에서 Range Query를 하는 것은 P2P 시스템이 가진 문제였고, 이를 해결할 한 가지 방법을 여기에서 제안한다.

1. 서 론

초기의 P2P 시스템은 중앙에 서버를 두고 서비스를 제공하는 Napster[5] 같은 P2P 시스템도 있었지만 현재는 자료를 공유하고자하는 목적으로 하나의 노드가 자료를 가지고 서비스하는 주체이자 동시에 다른 자료를 찾아서 받는 서비스 수혜자가 되는 시스템을 주로 이야기한다.

중앙 서버가 없는 P2P 시스템은 크게 두 가지 방식이 있는데 하나는 이웃노드에 Query를 흘려보내는 방식[6]이고, 다른 하나는 DHT를 이용한 방식이다 첫 번째 방식은 시스템 전체에 많은 부하와 성능상의 문제로DHT를 이용한 P2P 시스템의 연구가 발전 되었고, CAN[7], CHORD[8], PASTRY[9] 그리고 TAPESTRY[10] 등이 대표적인 DHT를 이용한 P2P Routing 시스템이다. 이러한 시스템들은 DHT를 사용하여 Route 정보의 분산과 효율적인 검색 서비스를 제공하여 초기 P2P 시스템의 확장 가능성 문제점을 해결하고, 병목현상 등을 해결함으로 시스템의 성능과 신뢰성을 향상 시켰다

우리는 이전 연구[3]에서 P2P상에서의 자료의 모호성과 불필요한 자료의 검색을 줄이고 P2P 상에서 효율적인 자료의 검색을 위해 Tapestry상에서 파일기술을 위한 RDF 온톨로지를 제안하였다. RDF 온톨로지도 효율적인 자료 검색은 가능했지만 생성된 RDF 온톨로지의 더 효율적으로 이용하기 위해 본 논문에서 우리는 이전 시스템에 Range Query가 가능한 시스템을 제안한다

Range 검색은 keyword 검색과 함께 중요한 검색분야이지만 Index 순차적으로 되어 있어야만 Range 검색이 가능하다는 부분에서 DHT를 이용한 P2P에선 늘 이슈가 되었던 검색부분이다.

DHT를 사용한 P2P 시스템은 Hashing의 특성상 연관된 자료라도 전혀 상관없이 위치에 Uniform하고 Random하게 재배치 되게 된다. 여기에서의 재배치는 자료의 재배치를 의미하는 것이 아니라 자료를 가지고 있는 노드의 Route 정보의 배치를 의미한다 이런 시스템에서 Range Query를 하는 것은 문제가 있다. 특정한 값 다음의 어떤 값이 어디에 나타날지 아무도 알 수 없기 때문이다. 즉 어떤 값들에 대한 배치가 전혀 순차적이지 않기 때문에 다음 값을 찾아 갈 수 없다.

우리는 이런 DHT를 이용한 P2P 시스템에서 온톨로지별로 Range Query를 지원하는 시스템을 제안한다

2장에서 본 시스템의 기반이 되는Tapestry와 파일기술을 위한 온톨로지를 살펴보고 3장에서 우리가 제안하는 시스템을 살펴볼 것이다. 그리고 마지막 4장에서 결론 및 향후 연구 과제를 알아 볼 것이다.

2. 관련연구

이번 장에선 Tapestry에서 DHT를 이용하여 노드를 찾는 방법과 파일 기술을 위한RDF 온톨로지를 알아보겠다.

2.1 Tapestry

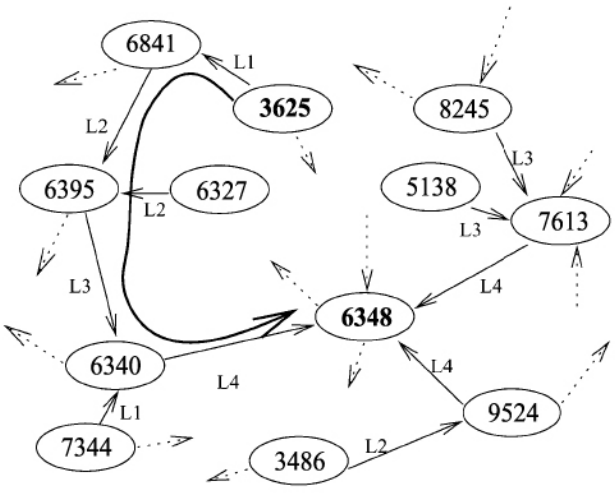
Tapestry는 Plaxton, Rajaraman과 Richa에 의해 소개된 hashed-suffix mesh[11]와 유사한 메커니즘을 사용한다. Tapestry와 같은 DHT 기반 P2P 시스템의 자원 탐색 시간은 O(logN)으로 알려져 있다.[7][8][9][10]

각 Tapestry 노드는 서버, 라우터 그리고 클라이언트 역

할을 하고 시스템에 참여할 때 임의 적이며 유일한 고정 길이의 비트를 가지고 물리적 위치와 독립된 식별자를 할당받게 된다. 이를 노드와 데이터에 따라 NodeID와 GUID(Global Unique ID)로 표현한다.

Tapestry는 네트워크 지역성(locality)을 고려하여 IP를 통해 탐색하는 시간으로 측정된 IP 탐색 기반구조에서의 가장 가까운 노드들을 이웃(neighbor)이라고 하며, 각 노드는 자신의 이웃들의 정보를 유지하여 다중 레벨들로 구성된 하나의 이웃 맵(neighbor map)을 가진다. 각 레벨은 목적지 ID인 GUID와 공유된 접두사(prefix)의 단계를 나타내며, 이 탐색 맵을 사용하여 탐색한다 이것은 요청 NodeID에서 GUID로 ID의 자릿수를 증가시키면서 메시지를 전송하는 방법으로, N번째 홉은 GUID와 적어도 N 길이의 접두사를 공유한다 GUID와 가장 많은 접두사를 공유하며 숫자상 가장 가까운 NodeID는 그 데이터의 루트 노드(root node)가 된다. 즉, Tapestry는 네트워크 내의 모든 노드가 루트가 된 트리들의 하나의 큰 집합으로 볼 수 있다.

데이터 D에 대한 질의 메시지는GUID를 가지고 D의 루트 쪽으로 순차적으로 레벨들을 탐색한다 각 단계에서 메시지가 D의 위치 정보를 포함하고 있는 노드를 만난다면, 그 메시지는 즉시 데이터를 포함하고 있는 서버 쪽으로 방향을 전환한다 만약 만나지 못한다면, 메시지는 루트에 도달하게 되며 D에 대한 위치 정보 탐색을 보장한다 데이터 D를 가진 서버 S는 D의 루트 노드 쪽으로 메시지를 전달하여 데이터가 네트워크에 참여하여 존재하게 되었음을 알린다. 루트 노드는 서버 S의 위치를 가리키는 간단한 포인터를 저장한다 이것은 단순한 포인터일 뿐 데이터의 복사본이 아니다.



[그림 1] GUID 6348을 찾는 노드3625의 Tapestry 탐색

[그림1]은 3625 노드에서 6348 노드를 찾아가는 모습을 보여준다. 먼저 Level 1에서 3625 이웃노드 중 접두사가 6을 찾고 다음에 Level에서 3을 찾는 방법으로 6348 노드를 찾는 모습을 보여준다 만약 이 시스템에서 6348 노드가 존재하지 않는다면 그와 가장 비슷한 노드가 대체 노드가 되어 6348 노드의 역할을 담당한다

2.2 온톨로지

RDF 형식의 온톨로지는 이미 널리 사용되며, 그 대표적인 예로는 문헌을 기술하기 위해 정의된 더블린코어 메타데이터[12]가 있다. 본 논문에선 파일을 기술하기 위한 최소한의 기술요소(성질)들에 대한 온톨로지를 제안한다.

[표1] 파일기술을 위한 온톨로지

name	파일 이름
title	자료의 제목
type	파일의 종류로 문서, 영상, 이미지 등
category	자료의 분류. 영상의 강좌, 영화 등
size	파일의 크기
format	파일의 형태
creator	파일을 만든 사람
modifier	마지막으로 파일을 변경한 사람
createDate	파일을 생성된 날짜
modifiedDate	마지막으로 변경된 날짜
description	파일에 관한 설명
source	현재 파일이 파생된 자원에 대한 참조
relation	관련 파일들에 대한 참조
keyword	검색 시 원하는 주제어
rights	파일에 대한 보유된 권리 정보

온톨로지는 특정 도메인에서 통용되는 의미 사전이다. 본문에서 제시하는 온톨로지는 특정 도메인인 “http://asadal.cs.pusan.ac.kr/~jwhan/off”에서 통용되는 온톨로지이다.

[표1]은 본 논문에서 제시하는 파일기술을 위한 온톨로지(off : Ontology For File-description)이다.

[표1]에서 Name은 파일의 이름이고, Title은 문서일 경우 제목을 의미한다. 파일명은 Tapestry에서 이미 고려된 상태이지만, Hash 값으로 찾아오기 때문에 루트노드가 관리하는 자료가 모두 동일한 검색어라고는 볼 수 없다. 그렇기 때문에 여기에선 파일이름이 명시되어야 한다. 그리고 Type은 자료의 형식이 무엇인지 나타내며, Category는 자료의 분류나 장르를 의미한다 예를 들어 Title이 “전쟁과 평화"라는 자료는 Type이 “audio/mpeg”고 Category는 “멜로”이다.

[표2] 파일에 관한 RDF

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:off="http://asadal.cs.pusan.ac.kr/~jwhan/off#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/">
   ....
   <rdf:Description rdf:about="tapestry.html">
      <off:name>tapestry.html</off:name>
      <off:title>
         RDF Ontology on the Tapestry
      </off:title>
```

```
        <off:type>Document</off:type>
        <off:format>text/html</off:format>
        <off:description>this document ...</off:description>
        <off:creator>
            <foaf:Person>
                <foaf:name>Jong-wook Han</foaf:name>
                <foaf:homepage rdf:resource=
                        "http://asadal.cs.pusan.ac.kr/~jwhan/"/>
            </foaf:Person>
        </off:creator>
        <off:createdDate>2005-03-05</off:createdDate>
        <off:keyword>
            <rdf:Bag>
                <rdf:li>tapestry</rdf:li>
                <rdf:li>rdf</rdf:li>
                <rdf:li>p2p</rdf:li>
            </rdf:Bag>
        </off:keyword>
    </rdf:Description>
    ....
</rdf:RDF>
```

[표2]는 tapestry.html이라는 파일을 파일 기술을 위한 온톨로지로 작성한 RDF 예제이다.

3. P2P 시스템에서의 Range Index 생성과 분산

P2P 시스템에서 Range Query를 위해 특정 노드에 Range Index를 생성하게 되지만, P2P 시스템에선 질의에 대한 부하보다 응답에 대한 부하가 더욱 크다.

본 시스템에선 Range Query시 하나의 노드에 그 질의가 집중되게 되고, 이 부하는 Range Query를 받는 특정 노드에 집중되게 되어 전체 시스템에 이상을 가져오게 된다.

P2P 시스템에서 Range Query를 위해 특정 노드에 Range Index를 생성하게 되고, 이 노드의 부하를 분산하기 위한 기법들을 이번 장에서 알아본다.

3.1 SearchType과 RNode

Range Query를 하기 위해 Range Index를 생성하게 된다. 하지만 이 Range Index를 모든 온톨로지에 작성하게 된다면 각각의 온톨로지마다 Range Index를 작성해야하는 부하가 생기며, 이는 네트워크 트래픽이나 리소스를 크게 소모하게 되어 전체 시스템적 효율이 떨어 질 것이다. 이를 방지하기 위해 SearchType을 두어 특정 온톨로지에만 Range Index를 생성할 수 있게 P2P 시스템의 시스템 관리자 또는 운영자가 특정 온톨로지에 Range를 지원하게 설정한다.

Tapestry는 기본적으로 Hash를 이용하지만 SearchType이 Range 경우 시스템은 먼저 Range Index를 어디에 생성할지 결정해야한다 이 부분은 각각의 온톨로지별로 따로 생성되는 부분이기 때문에 각 온톨로지를 Hashing 하여 그 결과 값에 해당하는 노드의 온톨로지 Range Index를 만든다. 이렇게 Range Index를 가지고 있는 노드를 Rnode라고 하겠다.

예를 들어 파일 기술을 위한 온톨로제3]에서의 "createDate"가 Range Query를 지원하기 위해 시스템 관리자가 SearchType을 Range로 설정 한다면 Tapestry에서 사용하는 Hash Function을 이용하여 이 "createDate"를 Hashing한다.

$$hf(\text{"off:createDate"}) = 6348$$

이 Hashing한 결과 값이 6348이라고 할 때, 이 "createDate" 온톨로지에 대해서는 Tapestry상에서 6348노드에 Range Index가 생성되며 6348노드는 Rnode가 되게 된다.

3.2 Rnode와 Histogram

Rnode에는 Range 정보가 수집되게 되고, 이 정보를 바탕으로 Range Index를 생성하게 된다. 이는 하나의 노드에 전체 P2P 시스템에서 많은 정보가 집중되게 되지만 Character형태의 정보는 P2P 시스템에 영향을 줄 만큼 큰 부분이 아니라서 무시할 수 있다. 하지만 여기에선 한 노드에 모든 Range Query가 집중되어 응답의 부하가 생겨 이를 해결하기 위해 Range Index를 이웃 노드에 분산하게 된다. Tapestry에서 이웃 노드는 가장 가까운 물리적 위치를 가진다. 이로 인해 이웃 노드에 Range Index를 분산하는 것이 접근 시간 면에서 유리하다. 이런 Range Index 분산을 이웃 노드 사이에 uniform하게 분산하기 위해서 본 논문에선 Histogram을 사용하는 시스템을 제안한다.

Histogram은 전체 정보에 대한 특정 값의 빈도를 쉽게 알 수 있게 하지만 위치 정보를 가지고 있진 않다.

본 시스템에선 Rnode는 하나의 Histogram을 가지고 차후 자신의 Range Index를 분산할 때 Histogram을 이용하여 Uniform하게 이웃 노드에게 전파한다.

3.3 Rnode의 Range Index 분산

Rnode는 하나의 Histogram을 가지고 계속적인 정보를 수집하게 된다. 하지만 이 노드는 전체 시스템의 특정 온톨로지의 Range Query를 모두 혼자 담당하게 된다. 이는 전체 시스템의 노드가 늘어남에 따라 Range 정보와 그에 상응하는 Query에 대한 응답 등도 늘어나 시스템의 부하가 더욱 커지게 된다. 이때의 부하는 메모리, cpu 사용률, 네트워크 트래픽 등으로 볼 수가 있다.

이렇게 하나의 노드에 부하가 특정 기준치를 넘어가게 되면 그 Rnode는 자신의 정보를 이웃 노드에 분산하게 된다.

처음 Range 정보를 분산하는 노드가 Root Rnode가 되고, 이는 처음 온톨로지를 Hashing한 값을 가진 노드가 된다. Root Rnode는 Range 정보를 분산할 때 자신의 Range 정보를 이웃노드에 모두 나눠주게 되고, 자신은 Range 값과 그 Range 값에 따른 담당 노드 번호만 가지고 Routing 서비스만 하게 된다.

Root Rnode를 포함한 모든 Rnode는 Range 정보를 이웃 노드에게 분산할 때 자신이 가지고 있는 Histogram을 이용하여 uniform하게 이웃 노드에게 분산하게 되지만, 만약 분산하는 노드가 Root Rnode가 아니면 이웃 노드에 Range 정보 분산시 자신을 포함하여 Range 정보를 이웃 노드에 분산하고, 자신이 분산한 Range 값과 그 Range를 관리하게 된 Rnode에 대한 Route 정보를 Root Rnode에 알리려 Root Rnode의 Route 정보를 갱신하게 된다.

3.4 시나리오

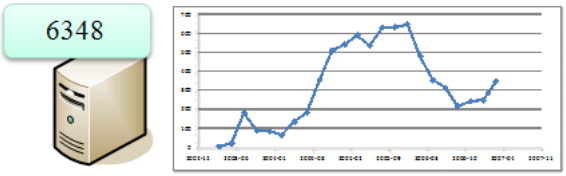
특정 온톨로지에 대하여 Range Index를 만들고 이를 분산하는 시나리오를 보겠다.

앞에서 설명한 파일기술을 위한 온톨로지의 createDate에 대하여 Range Index를 만들면, 먼저 createDate에 대해 Range Index를 수집할 Rnode를 선택해야한다. 이는 Hash Function을 이용해 특정노드를 선택한다.

$$hf("off:createDate") = 6348$$

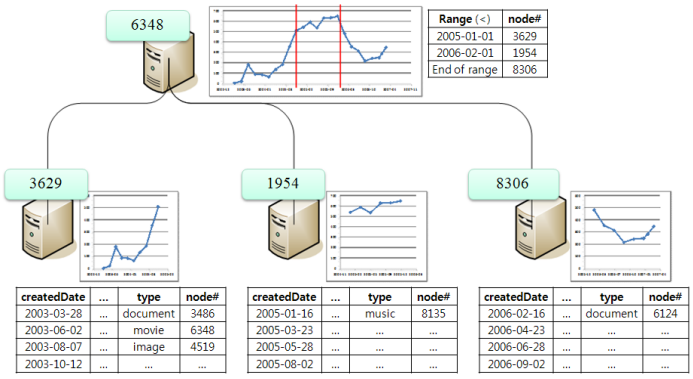
이렇게 createDate의 Range 정보를 관리할 노드가 6348이라고 하면, createDate에 관한 정보는 모두 6348노드에 모이게 된다.

Tapestry 시스템에서 6348이 createDate에 대한 Rnode가 되면 먼저 Histogram을 생성하게 되고, 시스템 리소스 등의 사용량이 임계치에 이를 때까지 Range 정보를 모으고, 동시에 특정 값의 빈도를 측정할 Histogram을 만들어 간다.



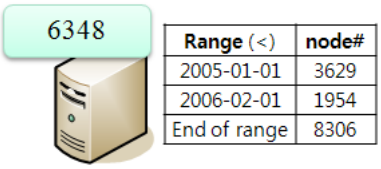
[그림2] 6348 노드와 Histogram

6348 노드에 Range 정보와 Query가 집중됨에 따라 그 노드의 부하가 임계치에 이르면 6348 노드는 이웃 노드에 Range 정보를 나누어 주게 된다.



[그림3] 이웃 노드에 분산된 Range 정보

[그림3]과 같이 6348 노드의 정보를 이웃 노드인 3629, 1954, 8306 노드가 나누어 가지게 되고, 그 범위는 6348 노드의 Histogram을 기준으로 uniform하게 배분하게 된다. 이렇게 6348 노드에서 이웃 노드로 Range 정보를 분산하게 되면, 6348 노드는 Root Rnode가 되어 더 이상의 Range 정보를 관리하지 않고 해당 Range 값과 그에 해당하는 Route(노드) 정보만 가지고 Routing 서비스만 담당하게 된다.



[그림4] Root Rnode와 Route 정보

Root Rnode는 [그림3]에선 Histogram이 보이지만 이는 배분하는 모습을 보여주기 위해 둔 것이고, Root Rnode는 [그림4]와 같이 더 이상 Histogram을 관리하지 않는다.

[그림4]는 Root Rnode와 Route 정보이다. Range 값은 미만을 의미하고, Node Number는 해당 Range를 담당하는 노드의 번호이다. 여기에선 노드번호를 나타내지만 실제로 구성하면 노드에 대한 pointer가 될 것이다. 이렇게 Root Rnode는 응답에 대한 부하가 없고, 다만 Routing만 담당하게 된다.

Root가 아닌 Rnode는 해당 Range에 대한 정보를 가지고 응답을 하게 되며, 이를 위해 Finger Table을 가지게 된다. 이 노드들이 분산할 때에는 Root Rnode와는 달리 자신도 일부의 Range 정보를 가지고 이웃 노드에게 분산하게 되며, 분산된 Range 정보와 담당하는 노드 번호를 Root Rnode에 알리게 되고, Root Rnode는 Route Table을 갱신함으로 Rnode의 Range 정보 분산을 완성하게 된다.

3.5 Range 정보의 중첩

P2P 시스템은 노드의 잦은 접속과 이탈이 특징이다. 이중에서 이탈은 문제가 될 경우가 있어 시스템 설계시 고려해야하는 중요한 부분이다.

P2P 시스템에서 이탈은 두 가지 형태가 있다. 정상적인 이탈과 비정상적인 이탈이다.

정상적인 이탈은 시스템에 자신의 이탈 사실을 알리고, 모든 서비스를 정상 종료한 이후 P2P시스템에서 이탈하는 것이고, 비정상적인 이탈은 정전, 시스템 Crash, 네트워크 마비 등으로 인한 이탈로 모든 서비스의 비정상적인 종료가 일어나 문제가 된다.

본 시스템에서 정상적인 이탈시 이탈하는 Rnode는 Tapestry상 자신의 대체 노드에 모든 정보를 넘겨주고 Root Rnode에 이 사실을 알리고 종료를 한다. 하지만 Rnode의 비정상적인 이탈시 해당 Rnode가 가지고 있는 Range 정보를 모두 잃게 된다. 이 잃을 Range 정보는 다시 생성하기 위해선 많은 시스템의 부하를 요구하여 새로 생성하기가 매우 힘들다. 이런 이유로 본 시스템은 Range 정보를 서로 중첩되게 관리를 하여 Rnode 비정상적인 이탈에 대비한다.

Range 정보의 중첩은 하위의 Range 값을 지닌 노드가 상위의 Range 값을 중첩하는 형식으로 하여, 최하위 Range 값의 정보는 최상위 Range 값의 정보를 가지고 있는 노드가 가지게 된다. [그림3]에서 보면 3629 노드는 1954 노드의 정보를 중첩하고, 8306 노드는 1954 노드가 지닌 정보를, 8306 노드는 3629 노드가 지닌 Range 정보를 중첩하게 된다. 만약 [그림3]으로 이루어진 Rnode중 8306 노드가 비정상적인 이탈을 하였을 경우 1954 노드는 자신이 가진 8306 노드의 Range 정보를 이웃 노드에 넘기게 되고, 이 사실을 Root Rnode인 6348 노드에 알림으로 정상적인 서비스가 가능하게 한다. 그리고 Root Rnode의 경우 자신의 대체 노드에 자신의 정보를 중첩되게 관리함으로 Tapestry에서 6348 노드가 없을 때 바로 서비스가 가능하게 된다.

4. 결론 및 향후 연구 과제

우리는 P2P 시스템에서 시멘틱 웹 요소인 RDF를 접목한 Tapestry에서 온톨로지별로 Range Query가 가능한 시스템을 제안하였다. 본 시스템은 Range 정보를 한곳으로 집중시키기 위해 온톨로지를 해싱 하였고, 그 결과 값으로 찾은 노드를 Rnode로 삼았다. 하지만 하나의 Rnode에 전체 시스템의 Range Query가 집중되게 되고, 이에 응답하는 부하가 커짐으로 Range 정보를 이웃 노드에 분산시킨다. 여기에서 Range 정보를 uniform 하게 분산하기 위해 Histogram을 이용하였다. 처음 Range 정보를 분산한 노드는 Root Rnode가 되고, 이 노드가 관리하는 Routing Table에는 Range 값과 그 Range를 담당하는 노드 번호가 있어 이후로는 Routing만을 담당하게 된다. 여기에 P2P 특성인 비정상적인 이탈시를 대비해 Rnode의 정보를 중첩되게 관리하여 이를 대비하였다.

하지만 여전히 DHT의 특성인 uniform한 분산을 이루지 못 하고, 특정노드를 중심으로 Range 정보가 집중되는 문제가 있다. 우리가 제안한 시스템에서의 uniform한 분산은 Root Rnode의 이웃에 한한 분산이다. Tapestry 전체에 해당하는 uniform이 아니다. 그리고 Range 정보를 분산할 때를 정확하게 파악하지 못했다 기준이 될 항목들은 언급하였지만, 그 항목의 특정 임계치를 제안하진 못하였다. 이 부분은 정확한 실험을 통해 제안할 예정이다.

참고문헌

[1] Semantic Web Road map, http://www.w3.org/DesignIssues/Semantic.html

[2] World-Wide Web Consortium: Resource Description Framework, http://www.w3.org/RDF

[3] 효율적인 P2P 파일 검색을 위한 RDF 파일 온톨로지 구조, 2006년도 한국정보과학회 가을 학술발표논문집 (D), 33권 2호, 637-341페이지

[4] EDUTELLA : A P2P Networking Infrastructure Based on RDF, http://edutella.jxta.org/

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

[6] Gnutella, http://www.gnutella.com/

[7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, "A Scalable Content-Addressable Network", ACM SIGCOMM'01, Aug. 2001

[8] I. Stoica, R. Morris, D. Karger, M.F Kaashoek, and H. Balakrishman, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications", ACM SIGCOMM'01, Aug, 2001

[9] A. Rowstron and P. Druschel, Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. IFIP/ACM International Conference on Distributed Systems Platforms, Nov. 2001

[10] B. Y. Zhao, J. Kubiatowicz, and A. D. Joseph,"Tapestry: An infrastructure for faulttolerant wide-area location and routing", Tech. Rep. UCB/CSD-01-1141, Computer Science Division, University of California, Berkeley, Apr 2001

[11] PLAXTON, C. G., RAJARAMAN, R.,AND RICHA, A. W."Accessing nearby copies of replicated objects in a distributed environment", In Proceedings of the Ninth Annual ACM Symposium on Parallel Algorithms and Architectures(SPAA) (June 1977)

[12] http://www.dublincore.org/about/overview/. Dublin Core Metadata Initiative