P2P 환경에서 삼각분할을 이용한 이동노드의 위치정보관리 및 질의처리

권오제[0] 문정욱 이기준
부 산 대 학 교 컴 퓨 터 공 학 과
{ojkwon, jwmoon}@isel.cs.pusan.ac.kr;lik@pnu.edu

Management and Query Processing for Moving Objects by TIN in P2P

O-Je Kwon[0] Jung-Wok Moon Ki-Joune Li
Dept. of Computer Enginnering, Pusan National University

요 약

P2P 모델은 대량으로 분산된 환경을 관리하는데 적합하다. 기존의 서버-클라이언트 모델과는 달리 P2P 모델은 서버가 존재하지 않는다. P2P에서 각 노드는 자신의 정보를 공유하고 원하는 정보를 가진 노드와 통신한다. 하지만 기존의 P2P 방법들은 공간정보를 다루고 있지 않기 때문에 본 논문에서 초점을 맞추고 있는 공간질의를 처리하지 못한다. P2P 환경에서 들로니 삼각분할 방법을 이용한 라우팅 및 공간질의처리 방법이 제안되었으나 고정노드만 다루고 있기 때문에 이동노드에 적용하는데 문제점이 있다. 이는 들로니 삼각분할 방법의 제약조건이 강해서 이동노드의 위치정보를 갱신하는데 큰 비용이 들기 때문이다. 따라서 본 논문에서는 들로니 삼각분할방법에 비해 제약조건이 약한 일반 삼각분할방법(TIN)을 이용하여 이동노드의 위치정보를 갱신하고 공간질의를 처리하기 위한 방법을 제안하고자 한다.

1. 서론

무선통신망과 GPS(Global Positioning System)를 탑재한 모바일 단말기의 발달로 사람이나 자동차의 위치파악이 용이해짐으로써 이를 이용한 위치기반 서비스(LBS : Location-Based Service)가 현재 많은 모바일 분야에서 이용되고 있다. 이러한 위치 기반 서비스는 지금까지 서버-클라이언트 환경에서 구축되었다. 이동객체(=클라이언트)들은 주기적으로 자신의 현재 위치정보를 서버에 보고하고 서버는 이 위치정보들을 관리하고 위치관련 질의를 처리한다. 하지만 다수의 이동객체들을 다루기에는 서버의 부하가 크다. 따라서 새로운 접근 방법이 필요하게 되었는데 그 중 하나가 P2P이다.

P2P(Peer to Peer)는 서버-클라이언트 모델과는 달리 모든 노드(=클라이언트)를 관리하는 서버가 따로 존재하지 않고 노드 각자가 자신의 정보를 관리하고 이를 공유하여 원하는 정보를 가진 노드와 직접 통신하는 환경이다. 각 노드는 라우팅(routing)을 통해 자신이 원하는 자료를 가진 노드를 찾아가는데 각 노드는 자신의 주변노드 정보만 가지고 있기 때문에 이들 정보를 이용하여 단계별로 찾아간다. 이 때 자신이 원하는 정보를 가진 노드까지 최적으로 찾아가는 방법이 가장 중요한 문제가 되고 이에 대한 연구가 많이 이루어졌다. 대표적인 방법으로는 분산해쉬테이블(DHT : Distributed Hash Table)을 이용한 CAN, Chord, PePeR, MAAN 등이 있다.

본 논문에서는 P2P 환경에서 이동객체의 위치정보를 관리해주고 공간질의를 처리하기 위한 방법을 제시하고자 한다. 본 논문에서 다루고자 하는 위치정보 관리 및 공간질의 처리를 위해서는 위치정보 기반의 라우팅 방법이 필요하다. 하지만 기존에 연구되었던 CAN이나 Chord등의 방법은 위치 기반이 아닌 내용 기반의 라우팅 방법이기 때문에 이동객체의 변하는 위치정보를 관리해주고 영역질의나 k-최근접질의와 같은 공간질의를 처리하는데는 적합하지 못하다. 따라서 본 논문에서는 [1]에서 제안된 위치정보 기반 라우팅 방법을 사용하였다.

[1]에서 노드들은 들로니 삼각분할 방법을 통해 네트워크를 형성한다. 들로니 삼각분할 방법을 통해 각 노드는 같은 에지를 공유하는 노드를 자신의 이웃노드로 등록하고 이 정보를 이용하여 라우팅 할 수 있다. 하지만 들로니 삼각분할 방법의 경우 제약조건이 강하기 때문에 노드가 이동했을 때 변한 위치정보를 갱신해주기 위해 들로니 삼각분할에 의해 만들어진 네트워크를 갱신해주어야 하는데 여기에 많은 비용이 든다. 따라서 본 논문에서는 들로니 삼각분할 방법에 비해 제약조건이 덜한 삼각분할 방법을 이용하여 네트워크를 형성하고 노드가 움직였을 때 각 노드간의 연결관계를 효율적으로 갱신해주기 위한 방법을 제안한다. 또 삼각분할의 특징을 이용하여 두 공간질의인 영역질의와 k-최근접질의

를 처리하기 위한 방법을 제안한다.

본 논문은 다음과 같이 구성된다. 2장에서는 관련 연구로 서버-클라이언트 모델과 P2P 모델을 비교하고 기존의 P2P에서 질의처리를 위해 연구되어온 방법들에 대해 알아본다. 3장에서는 노드가 이동할 때 노드의 위치정보를 유지해주기 위한 위치정보 갱신 방법을 제안한다. 4장에서는 삼각분할의 특징을 이용하여 영역질의와 k-최근접질의를 처리하기 위한 알고리즘을 제시한다. 5장에서는 본 논문에서 제시한 알고리즘의 성능을 보이고 마지막으로 6장에서 결론 및 향후 연구 과제에 대해 알아본다.

2. 관련 연구 및 연구 목적

2.1 관련 연구

본 절에서는 서버-클라이언트 환경과 P2P 환경에 대해 비교하고 P2P 환경에서 지금까지 연구되어 온 질의처리 방법들에 대해 간단히 알아본다.

서버-클라이언트 환경에서 질의처리는 클라이언트가 서버에 질의를 요청하면 서버는 그 질의를 처리하여 결과를 클라이언트에게 전달해줌으로서 수행된다. 이 때 클라이언트의 수가 증가할수록 서버의 부하는 커지게 된다. 만약 다수의 클라이언트가 동시에 서버에 질의를 요청하게 되면 서버에서는 이를 처리하는 과정에서 병목현상(bottle neck)이 발생하게 된다. 또 위치 기반 서비스를 위한 공간질의를 수행하기 위해 서버는 각 이동객체의 위치정보를 관리해주어야 한다. 이 때 매시간 변하는 이동객체의 위치정보를 갱신하기 위해 서버에 많은 부하가 든다. 이에 반해 P2P에서는 서버없이 각 노드들이 자신이 가진 자원을 공유하고 원하는 자원을 가진 노드와 직접 통신한다. 서버-클라이언트 환경에서 서버가 모든 질의를 처리하는 반면 P2P에서는 질의에 해당되는 노드들이 분담하여 처리한다. 따라서 서버에 집중되는 부하를 분산시켜 다수의 이동객체를 다루는데 용이하다. 서버-클라이언트 환경과 P2P 환경을 비교해보면 표 1과 같다.

표 1. 서버-클라이언트 환경과 P2P 환경의 비교

	서버-클라이언트	P2P
특징	서버가 모든 질의를 처리한다.	질의처리 시 각 노드들이 분산하여 처리한다.
장점	질의처리 및 정보관리가 편하다.	확장성이 좋다.
단점	서버의 의존도가 크다.	서버-클라이언트 환경에 비해 질의처리 및 정보관리가 어렵다.

P2P에서는 자신이 원하는 자원을 가진 노드까지 어떻게 찾아가느냐-여기서는 이를 라우팅(routing)이라고 한다-가 가장 큰 관심사가 된다. 따라서 P2P에서는 어떻게 하면 효율적인 라우팅을 할 수 있는가에 대한 연구가 주로 이루어졌다. 가장 대표적인 라우팅 방법에는 Gnutella[3]에서 사용하는 플러딩(flooding) 기법이 있다. 플러딩 기법이란 자신에 연결되어 있는 모든 노드에게 메시지를 흘려 보내는 방법을 말한다. 이 때 메시지가 모든 노드에 전달되는 것을 막기 위해 TTL(Time To Life)을 두어 TTL 수만큼만 메시지를 전달한다.

Gnutella에서는 자신의 이웃노드에게 모두 메시지를 전달하기 때문에 발생하는 메시지 수가 많으므로 검색비용과 네트워크의 부하가 증가한다. 따라서 이를 해결하기 위해 분산 해쉬 테이블(DHT : Distributed Hash Table)과 같은 색인을 이용하는 방법에 대한 연구가 이루어졌다.

대표적인 DHT 방법으로는 CAN[9]과 Chord[4]가 있다. CAN은 해쉬 함수를 통해 각 노드를 d 차원의 좌표 공간으로 매핑(mapping)시키고 자신의 영역에 인접한 여역의 노드를 이웃노드로 등록한다. 질의 노드와 목적 노드 역시 해쉬 함수에 의해 좌표 공간으로 매핑되고 각 노드의 이웃노드 정보를 통해 질의 노드에서 목적 노드로 메시지가 전달된다. Chord는 파일의 키 값과 노드 식별자의 키 값이 같은 주소공간을 가지도록 한다. Chord의 참여 노드들은 이 주소 공간을 분할하여 각 영역의 식별자에 해당하는 파일들을 관리한다. 노드의 개수가 n 일 때 식별자는 0에서 2^n-1 사이의 값을 지닌다. 그리고 각 노드들은 핑거테이블(finger table)이라는 경로 테이블을 유지해 노드에의 링크 정보를 유지한다.

CAN과 Chord는 해쉬 테이블을 사용하기 때문에 해쉬 테이블이 가지는 기본적인 문제점인 영역 질의를 처리할 수 없다. 따라서 이런 문제점을 해결하기 위해 CAN과 Chord를 영역 질의가 가능하도록 확장한 PePeR[2]과 MAAN[6] 방법이 제시되었다. 하지만 두 방법 역시 특정 속성에 대해서 영역질의가 가능할 뿐, 본 논문에서 다루고자 하는 공간질의를 처리하기에는 적합하지 못하다.

[1]에서는 P2P 환경에서 공간질의를 처리하는 알고리즘을 제시하였다. [1]에서 각 노드들은 들로니 삼각분할 방법을 통해 같은 에지에 연결된 노드를 자신의 이웃노드로 등록한다. 들로니 삼각분할 방법을 통해 노드들을 연결하면 하나의 노드에서 다른 노드로 가는 경로가 적어도 하나 이상 존재함을 보장할 수 있음으로 공간질의처리가 가능해 진다.

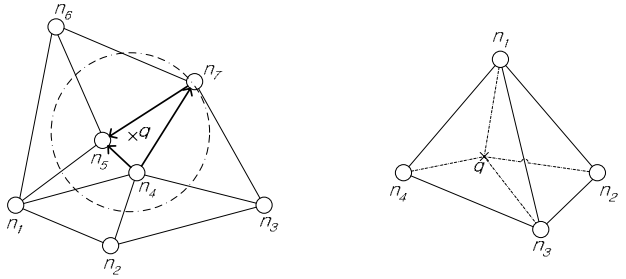


그림 1. [1]에서 공간질의 처리

2.2 연구 목적

앞 서 본 바와 같이 서버-클라이언트 모델은 다수의 이동객체에 대해서 적합하지 못하다. 또한 기존의 P2P 방법들은 위치정보를 다루고 있지 않기 때문에 본 논문에서 다루고자 하는 이동객체의 위치 갱신이나 공간질의 처리에는 적합하지 못하다. 비록

[1]에서 공간질의를 처리하기 위한 알고리즘이 제시되고 있으나 여기서 사용한 들로니 삼각분할 방 법은 제약조건이 강하기 때문에 이동노드의 위치정보를 갱신하는데 많은 비용이 든다.

삼각분할을 이용할 경우 이동객체가 움직였을 때 삼각분할 방법에 의해 만들어진 네트워크를 구성하는 에지간에 교차가 일어나는 경우에만 네트워크를 갱신해주면 된다. 반면 들로니 삼각분할 방법을 이용하면 에지간의 교차 유무 조건뿐만 아니라 외접원 내에 다른 노드가 포함되는지에 대한 조건도 검사해야한다. 따라서 삼각분할 방법에 비해 더 많은 범위에서의 네트워크 갱신이 일어날 수 있다. 또한 삼각분할에서의 갱신은 갱신된 노드와 그 이웃노드들 사이에서 일어나는 반면 들로니 삼각분할에서는 이웃노드들 이외의 노드도 영향을 받을 수 있기 때문에 최악의 경우 모든 노드들이 갱신에 영향을 받을 수 있게 된다. 따라서 본 논문에서는 들로니 삼각분할 방법 대신 이보다 제약조건이 덜한 삼각분할 방법을 이용하였다.

본 논문에서는 삼각분할 방법을 이용하여 노드들 간의 연결 정보를 유지해 주고 노드가 이동할 경우 이를 효율적으로 갱신해주기 위한 알고리즘을 제안한다. 또 삼각분할을 이용하여 영역질의와 k-최근접질의를 처리하기 위한 방법론을 제시한다. 라우팅 방법의 경우 노드간의 이웃정보만 가지고 있으면 적용 가능하기 때문에 본 논문에서는 [1]의 라우팅 방법을 그대로 사용하였다.

3. 이동객체 위치정보 갱신

본 장에서는 노드가 이동할 경우 위치정보를 효율적으로 갱신하기 위한 알고리즘을 제안한다. 노드가 이동하면 기존의 삼각분할 네트워크가 깨어질 수 있고, 이러한 경우 네트워크를 재구성해야 한다. 삼각분할로 구성된 네트워크의 이상 유무는 움직인 노드에 의해 판단된다. 각 노드는 자신의 이웃노드의 위치 정보를 가지고 있기 때문에 이를 이용하여 노드가 움직일 경우 네트워크 상의 에지간에 교차가 일어났는지를 판단할 수 있기 때문이다.

본 논문에서는 이동객체의 위치정보를 갱신하기 위해 다음과 같이 가정한다.

- 노드는 한번에 하나씩 움직인다.
- 노드의 속도는 일정하다.
- 각 노드는 IPv6 환경의 인프라 구조를 이용하여 P2P 방식으로 통신한다.

이동객체의 위치정보를 갱신하기위한 알고리즘을 알아보기 전에 각 노드가 가지는 정보와 본 논문에서 사용하고자 하는 삼각분할 방법 및 라우팅 방법에 대해 알아보자.

3.1 이동노드의 구조 및 삼각분할방법

각 노드는 크게 두 가지 정보를 가진다. 하나는 자신의 정보이고 다른 하나는 자신의 이웃 노드의 정보이다. 자신의 정보에는 자신의 노드 id와 위치 정보 (x, y)가 포함되고 이웃 노드의 정보에는 자신과 이웃한 노드의 수와 각 노드의 id 및 위치 정보 (x, y)가 포함된다. 여기에 질의 처리를 위해 부가적인 정보를 가지는데 자신이 받은 메시지의 번호와 자신에게 메시지를 전송한 노드의 id가 그것이다. 이 두 정보는 질의 처리시 발생하는 메시 지의 수를 줄여 성능 향상을 도모하기 위해 추가되었다.

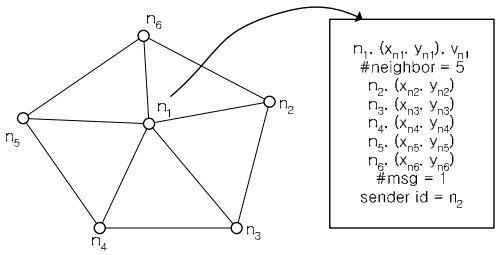


그림 2. 이동노드의 기본구조

각 노드가 자신의 이웃 노드를 경정하기 위해서 본 논문에서는 삼각분할(TIN : Triangulated Irregular Network) 방법을 사용한다. 삼각분할 방법은 노드들을 에지로 연결하는데 이때 에지들은 데이터 영역을 서로 교차하지 않는 삼각형 형태로 분할한다. 각 노드는 자신과 공통 에지를 가지는 노드를 자신의 이웃노드로 등록하게 된다. 그림 2에서 노드들은 삼각분할 방법으로 연결되어 있는데 노드 n_1은 자신과 공통 에지를 가지는 노드 n_2, n_3, n_4, n_5, n_6을 자신의 이웃노드로 등록한다.

삼각분할 방법으로 이웃노드 정보를 구성하면 각 노드는 특정 노드로의 경로를 반드시 하나 이상 가지게 되기 때문에 고립 노드가 생기지 않아 모든 노드에 이웃 노드 정보를 이용한 홉(Hop) 방식으로 메시지 전달이 가능하게 된다. 이러한 성질은 공간질의 처리를 위해서 매우 중요하다.

3.2 라우팅 방법

본 논문의 주된 목적은 이동노드간의 위치정보 갱신 및 공간질의처리이기 때문에 라우팅 방법에 대해서는 주요하게 다루지 않고 [1]에 제시된 라우팅 방법 중 부분적 최적 노드 선택 전송 기법을 사용하였다.

부분적 최적 노드 선택 전송 기법은 현재 노드의 이웃 노드 중 질의점과 거리가 가장 짧은 노드에게만 메시지를 전달하는 방법이다.

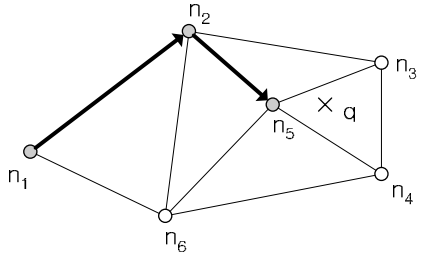


그림 3. 부분적 최적 노드 선택 전송 기법

그림 3에서 n_1은 자신의 이웃 노드 중 질의점 q에

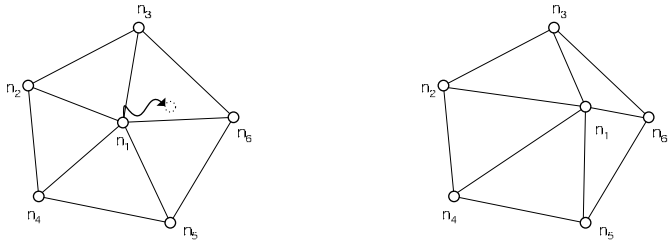
가장 가까운 n_2에 메시지를 전달하고 n_2는 자신의 이웃 노드 중 질의점에 가장 가까운 n_5에 메시지를 전달한다.

3.3 삼각분할의 갱신 방법

각 노드는 이동성을 가지기 때문에 한 노드가 이동 하여 다른 위치로 갈 경우 노드간의 삼각분할 관계가 깨어질 수 있다. 그럴 경우 삼각분할 관계를 갱신해 주어야 하는데 이때 각 노드가 가지는 이웃노드의 정보 역시 갱신된다. 본 장에서는 노드가 이동하여 삼각분할 관계가 깨어질 경우 이를 갱신하고 이웃노드의 정보를 갱신하는 방법을 제안한다. 이에 앞서 다음과 같이 가정한다.

- 노드는 한번에 하나씩 움직인다.

노드가 이동하는 경우는 크게 두 가지로 나뉜다. 첫 번째 경우는 노드가 현재의 삼각분할 관계를 유지하는 범위 내에서 이동하는 경우이다. 이 경우는 노드가 자신의 이웃노드들에 의해 만들어 지는 경계 내에서 움직인 것을 의미하는데 이때는 삼각분할 관계가 깨지지 않기 때문에 삼각분할 방법에 의해 만들어진 네트워크를 갱신할 필요가 없다.



(a) 갱신 전 (b) 갱신 후

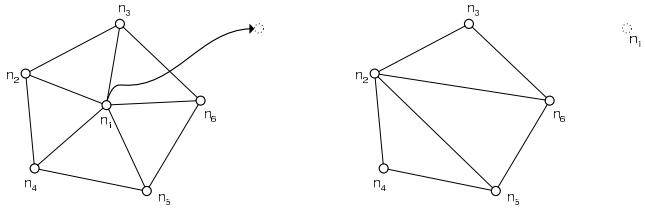
그림 4. 삼각분할이 갱신되지 않는 경우

그림 4의 경우 n_1이 원래의 위치(a)에서 이동하였지만 자신의 이웃노드들과 삼각분할 관계가 깨어지지 않기 때문에 삼각분할을 갱신해 줄 필요가 없다. 단 이 때에는 n_1의 위치 정보가 바뀌었기 때문에 자신의 변한 위치를 이웃노드들에게 알려주기만 하면 된다.

두 번째는 노드가 현재의 삼각분할 관계를 벗어나는 범위로 이동하는 경우이다. 이 경우는 노드가 자신의 이웃노드들에 의해 만들어지는 경계 범위를 벗어나는 경우로 링크간에 교차가 발생한다. 이를 갱신해주기 위한 방법은 다음의 두 단계로 나뉜다.

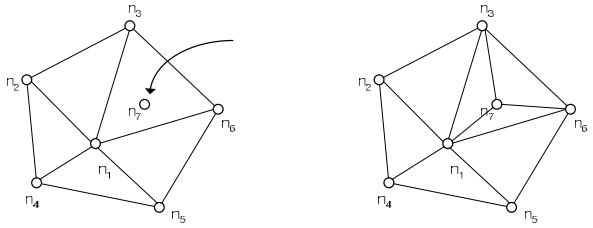
- 노드가 이전에 존재했던 위치에서 이웃노드들간의 삼각분할 관계 갱신
- 노드가 새롭게 포함된 영역에서 삼각분할 관계 갱신

그림 5에서 n_1이 이웃노드의 경계 밖으로 움직이면 이웃노드들은 자신의 이웃노드 정보에서 n_1을 삭제하고 새롭게 삼각분할을 갱신한다(b). 이 과정은 이웃노드들 간에 상호 과정을 통해 처리하기는 힘들기 때문에 움직이는 노드가 위의 과정을 처리하기 위한 메시지를 이웃노드들에게 전달함으로써 이루어진다. 이는 노드가 이웃노드의 위치정보를 가지고 있기 때문에 가능하다.



(a) 갱신 전 (b) 갱신 후

그림 5. 노드 이동 전의 위치에서 삼각분할 갱신

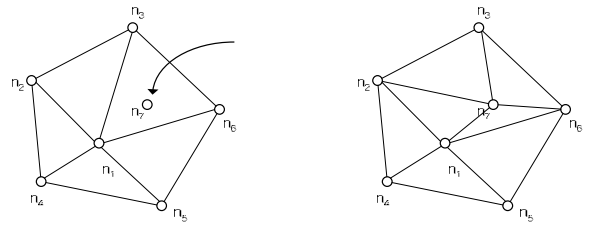


(a) 갱신 전 (b) 갱신 후

그림 6. 노드 이동 후의 위치에서 삼각분할 갱신

노드가 새롭게 포함된 영역에서의 삼각분할 갱신은 현재 위치에서 그 노드를 감싸는 세 노드를 라우팅을 통해 찾는다. 그림 6에서 n_7은 자신을 감싸는 세 노드 n_1, n_3, n_6을 찾은 후 이 세 노드의 이웃노드로 등록된다. 하지만 이 경우 노드들이 특정 위치로 집중되어 움직일 때 특정 노드에 이웃노드가 몰리는 현상이 발생하게 되고 이는 질의 처리 시 특정 노드의 부하가 커지는 문제를 야기시킨다. 따라서 이를 다음과 같이 해결하였다.

노드 n은 자신이 움직인 후의 위치(x', y')에서 가장 가까운 노드 n'를 라우팅을 통해 찾는다. n'를 찾으면 n'의 위치정보와 그 이웃노드들의 위치정보를 받아온다. 이 정보를 바탕으로 노드 n은 자신이 포함된 영역에 삼각분할을 갱신하기 위한 메시지를 만들어 전달한다. 이 때 각 노드가 일정하게 이웃노드를 가지기 위해 들로니 삼각분할 방법을 이용하였다.



(a) 갱신 전 (b) 갱신 후

그림 7. 부분적 들로니 삼각분할 방법을 통한 갱신

그림 7에서 n_7이 이동하여 새로운 영역에 들어오면 자신과 가장 가까운 노드인 n_1을 라우팅을 통해 찾는다. n_7은 n_1로부터 n_1의 위치정보 및 이웃노드인

n_2, n_3, n_4, n_5, n_6의 위치정보를 받아서 삼각분할을 갱신한다.

4. 공간질의처리

본 장에서는 본 논문에서 다루고자 하는 공간질의인 영역질의와 k-최근접질의를 처리하기 위한 방법을 제시한다. 영역질의는 질의점으로부터 질의 반경 범위 내에 포함되는 노드를 찾는 질의로 정의할 수 있고 k-최근접질의는 질의점으로부터 가장 가까운 노드 k개를 찾는 것으로 정의할 수 있다. 예를 들어 사용자는 " 부산시청에서 반경 100m 이내에 존재하는 이동객체를 찾아달라." 라고 영역질의를 요청할 수 있고 " 부산대학교 정문에서 가장 가까운 k개의 이동객체를 찾아달라." 라고 k-최근접질의를 요청할 수 있다.

본 논문에서는 공간질의처리를 위해 다음과 같이 가정한다.

- 공간 질의 처리 시 노드는 움직이지 않는다.

4.1 영역질의처리

영역 질의를 처리하기 위해서는 먼저 질의를 시작할 최초의 노드를 선택하여야 하는데 본 논문에서는 이 최초의 노드를 질의점을 둘러싸는 세 노드 중 라우팅 방법에 의해 가장 먼저 도달하는 노드로 정한다. 이 노드를 시작 노드로 하여 이웃 노드의 정보를 보고 질의 범위에 포함되는 노드들에게 메시지를 전송하면서 질의를 처리해 나간다.

메시지 전송은 공간적으로 질의 범위에 포함되는 이웃 노드에게만 전달한다. 하지만 이 방법으로 하면 찾지 못하는 노드가 생기게 되는데 그림 8은 이러한 경우를 보여 주는 예이다.

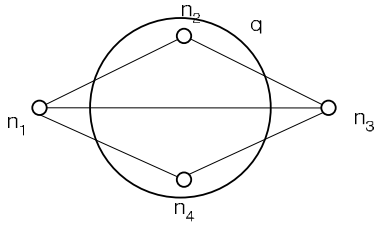


그림 8. 노드가 질의영역에 포함되나 찾지 못하는 경우

그림 8에서 현재 메시지를 받은 노드를 n_2이라 할 때 n_4는 n_1와 n_3이 질의 영역에 포함되지 않기 때문에 메시지를 전달받지 못한다. 따라서 비록 질의 영역에 포함되지 않지만 메시지를 전달 받아야 하는 노드를 선택하여야 한다. 이런 문제가 발생하게 되는 이유는 노드가 삼각분할 방법에 의해 연결되어 있기 때문이다. 이 문제를 해결하기 위해 본 논문에서는 질의 영역 밖에 있는 노드를 두가지로 분류하였다.

첫 번째 노드는 질의영역 밖에 존재하면서 메시지를 받지 않아도 되는 노드이다. 이 노드는 질의영역 밖에 존재하고 자신의 이웃노드 중 질의영역 내에 포함되는 노드가 존재할 가능성이 없기 때문에 메시지를 전달받지 않아도 되는 노드를 의미한다.

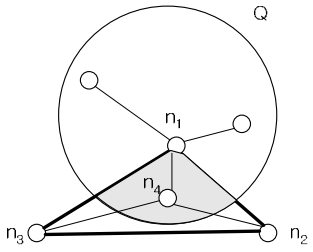


그림 9. 질의영역 밖에 있는 노드에게 메시지를 전송하지 않아도 되는 경우

그림 9에서 n_1의 이웃 노드인 n_2, n_3에 대해 n_1이 메시지를 날릴 것인지 결정해야 한다. n_2나 n_3가 메시지를받을지 말지 결정하는 근거는 n_1, n_2, n_3가 만들어 내는 삼각형의 영역이 n_1을 중심으로 벡터 $v(n_1, n_2)$와 벡터 $v(n_1, n_3)$를 무한히 확장했을 때, 잘리는 질의 영역(그림 9에서 색칠된 영역)을 포함하느냐 못하느냐 유무이다. 여기서는 삼각형 영역이 질의 영역을 포함하기 때문에 n_4는 삼각분할의 특징에 의해 n_1로부터 메시지를 전달 받을 수 있다. 따라서 n_2, n_3에 메시지를 전달할 필요가 없다.

두 번째 노드는 질의영역 밖에 존재하지만 메시지를 받아야 하는 노드이다. 이 노드는 질의영역 밖에 존재하지만 자신의 이웃노드 중 질의영역에 포함되는 노드가 존재할 가능성이 있기 때문에 메시지를 전달받아야 하는 노드를 의미한다.

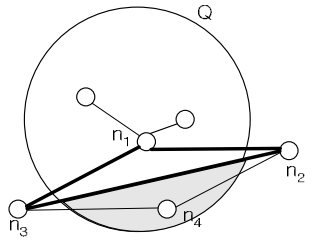


그림 10. 질의영역 밖에 있는 노드에게 메시지를 전송해야 하는 경우

그림 10에서 n_4는 질의 영역에 포함되지만 n_2와 n_3가 질의 영역에 포함되지 않아 메시지를 전달받지 못했기 때문에 메시지를 받을 수 없다. 그러므로 이 경우에는 n_2와 n_3에 메시지를 전달해야 한다. 이런 경우는 n_1, n_2, n_3로 이루어지는 삼각형 영역이 n_1을 중심으로 벡터 $v(n_1, n_2)$와 벡터 $v(n_1, n_3)$를 무한히 확장했을 때 잘리어지는 질의 영역을 완벽히 포함하지 못하여 그 포함하지 못한 영역에 노드가 존재할 경우 발생하게 된다. 그림 10에서는 색칠된 영역을 삼각형 영역이 포함하지 못하기 때문에 색칠된 영역에 존재하는 n_4에 메시지를 전달하기 위해 질의 영역 밖에 존재하는 이웃 노드에게도 메시지를 전송해 주어야 한다. 이 때 두 이웃 노드 중 하나의 노드에게만 메시지를 전달해 주면 되는데 질의 영역에 포함되는 노드는 삼각분할 방법의 특성 상 두 이웃 노드 중 한 노드에 의해 접근되어질

수 있기 때문이다.

이러한 방법을 이용하여 전달하면 한 노드는 다른 노드로부터 같은 메시지를 여러 번 받게 된다. 이렇게 중복해서 발생하는 메시지 수를 줄이기 위해서 메시지를 프루닝(pruning)시켜 줘야 한다. 프루닝 작업은 메시지를 이웃 노드에 전달할 때 발생한다. 한 노드에서 질의에 해당하는 이웃 노드로 메시지를 전달할 때 바로 직전에 메시지를 전달받은 노드의 이웃 노드는 메시지를 언젠가는 받기 때문에 제외시킨다.

그림 11에서 현재 노드가 n_4이고 이 노드는 n_1로부터 메시지를 전달 받았다고 할 때 n_4는 자신의 이웃 노드들 중 질의 영역에 포함되는 노드 n_2, n_3, n_5, n_6에 메시지를 전달해야 한다. 이 때 n_1의 이웃 노드가 되는 n_2와 n5는 n_1에 의해 메시지를 전달받기 때문에 n_4는 이 두 노드에게 메시지를 전달할 필요가 없다. 따라서 n_2와 n_5로 가는 메시지는 프루닝 되고 이 두 노드를 제외한 n_3와 n_6에게만 메시지를 전달하면 된다.

이렇게 메시지를 프루닝 해주면 영역질의 처리 시 발생하는 전체 메시지 수를 많이 줄일 수 있다. 하지만 중복된 메시지는 여전히 발생하게 되는데 각 노드는 이전에 받은 메시지 정보를 알고 있어 이를
통해 자신이 이미 받은 메시지 이면 그 메시지를 거부(discard)한다.

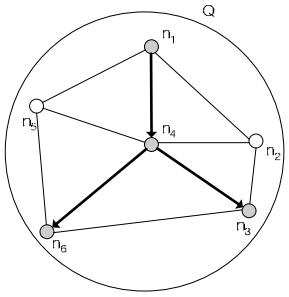


그림 11. 메시지 전달 시 프루닝 정책

4.2 k-최근접질의처리

k-최근접 질의 수행을 위해서는 메시지를 언제까지 전송해야 하는지 결정하는 문제가 가장 중요하다. 물론 여기서는 질의 결과가 k개가 될 때까지 메시지를 전송해야 한다. 서버-클라이언트 구조에서는 서버가 모든 클라이언트의 정보를 알고 있기 때문에 이를 쉽게 처리할 수 있지만 P2P에서 각 노드는 자신의 이웃노드 정보만 알고 있기 때문에 메시지를 받은 노드는 자신이 k개의 결과에 포함되는지 판단하기가 어렵다. 따라서 본 논문에서는 노드가 질의 결과에 포함되는지 유무를 판단하는 것을 질의 노드에게 맡긴다. 즉 질의 노드가 일종의 서버 역할을 담당하는 것이다. 질의 노드는 자신의 이웃 노드에 메시지를 전달하면서 그 결과를 다시 자신이 받고 질의 결과가 k 개가 넘지 않으면 다음 메시지를 전달할 노드를 선택하여 메시지 전송을 다시 수행한다.

k-최근접 질의 처리를 위해서 노드는 두 종류로 나뉜다. 하나는 k-최근접 질의를 처리하기 위해 연산을 하는 노드로 질의노드가 이 경웨 해당된다. 다른 하나는 자신의 이웃노드 정보를 보내주는 노드인데 이 이웃노드 정보를 질의노드로 보내주면 질의노드는 이 정보를 최근접 질의처리에 사용한다.

질의 노드에서의 연산 과정은 경계 사각형을 만드는 단계와 이를 이용하여 경계 사각형에 내접하는 원을 만드는 단계, 메시지를 전송할 다음 노드를 선택하는 단계로 나뉜다.

- 경계 사각형을 만드는 단계 : 첫 번째 단계에서는 다른 노드들로부터 받은 위치 정보들을 통해 경계 사각형(BR : Boundary Rectangle)을 만드는 것이다. 이 경계 사각형은 위치 정보들이 추가될 때 마다 갱신되어 점점 커지는데 이 사각형에 포함되는 노드들은 질의 결과의 후보(candidate) 노드가 된다.
- 최대 내접원을 만드는 단계 : 두 번째 단계에서는 이전 단계에서 만들어진 경계 사각형에 포함되는 질의점을 중심으로 한 최대 내접원을 구한다. 이 최대 내접원에 포함되는 노드는 질의 결과의 최종 후보 노드가 된다. 여기에 포함되는 노드의 수가 k 개보다 크면 질의 처리는 끝나게 되고 만약 k 개보다 작으면 질의 노드는 세 번째 단계를 수행한다.
- 메시지를 전송할 다음 노드를 선택하는 단계 : 마지막 세 번째 단계는 메시지를 전송할 다음 노드를 구하는 단계인데 이 노드는 경계 사각형을 확장시켜 최대 내접원을 확장시키는 역할을 한다. 이 노드는 질의점에서 가장 가까이 있는 노드로 선택하는데 이는 경계 사각형을 노드 분포에 균형적으로 확장시키기 위해서이다.

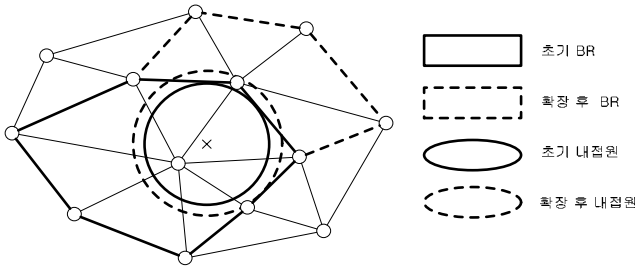


그림 12. k-최근접질의처리 시 BR과 내접원 확장

그림 12에서 초기 경계 사각형에 k 개의 결과가 포함되지 않으면 가장 가까운 노드에 메시지를 전달하고 메시지를 받은 노드는 자신의 이웃 노드의 위치 정보를 질의 노드에 전달하고 질의 노드는 이 정보를 바탕으로 경계 사각형과 내접원을 갱신한다.

5. 실험 및 성능 평가

본 장에서는 앞서 제안한 이동객체의 삼각분할 관계를 갱신하기 위한 알고리즘과 두 공간질의인 영역질의와 k-최근접질의처리 알고리즘의 성능을 실험을 통해 알아본다.

본 논문에서는 평가 기준으로 한 노드에서 발생하는 평균 메시지 수를 측정하였다. 이를 통해 서버

-클라이언트 환경에서 이동객체의 위치정보 갱신 및 공간질의처리 시 서버에 집중되는 부하가 P2P에서는 얼마나 줄어들 수 있는지를 알 수 있었다.

실험은 균등분포, 사선분포, 집단분포에 대한 합성 데이터를 사용하였고 각 이동객체들은 삼각분할 방법으로 연결되어있다고 가정한다. 이동객체들의 삼각분할 연결 방법은 [1]의 방법을 활용하였다.

5.1 이동객체 위치정보 갱신 실험

이동객체 위치정보 갱신 실험에서는 전체 이동노드 중 임의의 100개의 노드가 순차적으로 움직인다고 가정하였다. 즉, 한 노드가 움직인 후 갱신이 일어나고 갱신 후 다음 노드가 움직인다. 이 때 각 노드의 이동 범위를 다양하게 변화시키면서 한 노드가 발생하는 평균 노드수와 최대 노드수를 측정하였다. 먼저 서버-클라이언트 환경과 P2P 환경에서 100개의 이동노드의 위치갱신 시 발생하는 메시지 총 수를 측정해 보았다. 결과는 표2와 같다.

표 2. 갱신처리 시 발생하는 메시지 총 수

	0.001	0.003	0.005	0.01	0.02
서버-클라이언트	100	100	100	100	100
P2P(집단분포)	624	930	1005	1253	1360
P2P(균등분포)	608	734	834	1131	1443
P2P(사선분포)	634	823	1245	1307	1348

표 2를 보면 서버-클라이언트 환경이 P2P 환경에 비해 발생하는 메시지 수가 작음을 알 수 있다. 이는 P2P 환경에서는 노드들이 분산하여 처리를 하는 반면 서버-클라이언트 환경에서는 서버가 도맡아서 처리하기 때문이다.

다음으로 균등분포에서 갱신처리 시 노드 하나가 발생하는 메시지 수를 측정해 보았다. 그 결과는 그림 13과 같다.

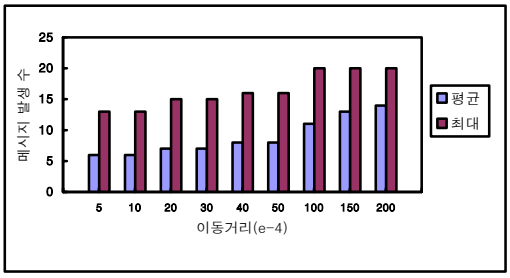


그림 13. 균등분포 데이터에서 위치정보 갱신 실험

서버-클라이언트의 경우 서버가 모든 처리를 도맡아 하기 때문에 서버에 100개의 메시지가 발생한다. 반면, 본 논문에서 제시한 방법에 의하면 한 노드가 발생하는 메시지 최대 수가 20개 정도로 서버-클라이언트에 비해 작음을 알 수 있었다. 이는 갱신 처리 시 갱신에 직접 영향을 받는 노드에게만 메시지를 전달하기 때문이다. 집단분포 및 사선분포에서도 유사한 결과를 보였다.

5.2 공간질의처리 실험

영역질의는 질의영역을 데이터 영역의 0.1%, 0.5%, 1%의 원(circle)으로 설정하고 각 질의영역에 대해 1000개의 질의점을 임의로 선택하여 수행하였다. 여기서 질의점은 삼각분할의 네트워크 내에 존재한다고 가정하였다. 결과로 발생하는 메시지 수를 측정하였는데 본 논문에서 제시한 프루닝 방법을 적용시키기 전과 후의 메시지 수를 비교해 보았다. k-최근접질의의 경우 k값을 1, 10, 50으로 변화를 주고 영역질의와 마찬가지로 각 k값에 대해 1000개의 질의점을 임의로 선택하여 수행하였다. 이 때, 질의를 하는 노드의 최대 발생 메시지 수와 평균 발생 메시지 수를 측정하였는데 k-최근접질의에서는 질의를 하는 노드가 임시 서버의 역할을 하여 질의를 처리하기 때문이다.

먼저 영역질의처리 시 발생하는 총 메시지 수를 측정하였다. P2P 환경에서의 실험은 프루닝 정책을 적용시켰다. 그 결과는 표 3과 같다.

표 3. 영역질의처리 시 발생하는 메시지 총 수

	0.1%	0.5%	1%
서버-클라이언트	1000	1000	1000
P2P(집단분포)	52008	281757	581913
P2P(균등분포)	34157	216081	461519
P2P(사선분포)	296706	1438938	2642397

실험결과 P2P 환경에서 발생하는 메시지 수가 상당히 많음을 알 수 있었다. 이는 영역질의를 처리하기 위해 포함되는 참여 노드들이 많기 때문이다. 데이터의 특성상 균등분포 데이터에 비해 집단분포와 사선분포 데이터에서 더 많은 메시지가 발생하였다. 이는 균등분포에 비해 집단분포 및 사선분포에서 참여하는 노드 수가 더 많기 때문이다.

다음으로 균등분포에서 영역질의처리 시 한 노드가 발생하는 메시지 수를 측정해 보았다. 그 결과는 그림 14와 같다.

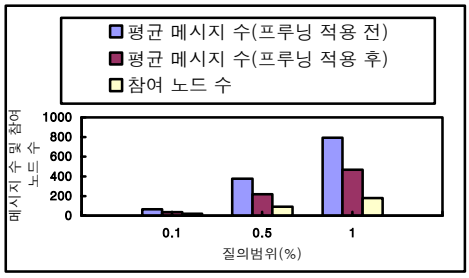


그림 14. 균등분포 데이터에서 영역질의처리 실험

서버-클라이언트 환경은 서버가 질의처리를 수행하기 때문에 1000개의 영역질의에 대해 1000개의 메시지가 발생한다. 반면 본 논문에서 제시한 알고리즘에 따르면 질의영역에 해당하는 노드들이 분산하여 수행하기 때문에 그보다 작은 메시지가 발생한다. 실험 결과 한 노드가 발생하는 메시지는 3~5개 정도로 아주 작았다. 집단분포 및 사선분포에서도 결과가 3~5개 정도로 균등분포 실험과 유사하였다. 프루닝을 적용시켰을 때 프루닝을 적용하지 않았을 때에 비해 40% 이상 메시지 수가 줄어듦을 알 수 있었다.

k-최근접실험의 경우 k값이 증가함에 따라 발생하는 메시지 수는 증가하지만 서버-클라이언트 환경에서 서버가 발생하는 메시지 수 보다는 상당히 작음을 알 수 있었다.

6. 결론 및 향후 연구

본 논문에서는 P2P 환경에서 이동노드에 대한 위치정보를 효율적으로 관리하기 위해 삼각분할 방 법을 사용하였다. 이 때 노드가 이동할 경우 현재 삼각분할 관계가 깨어질 수 있기 때문에 이를 갱신해주기 위한 알고리즘을 제시하였다. 이때 움직이는 노드는 갱신에 관련된 노드들의 정보를 가지고 그들에게 갱신 메시지를 전달함으로서 삼각분할 갱신을 처리하였다. 공간질의처리는 목적노드로 질의 메시지를 전달하는 라우팅 단계와 질의 조건에 만족하는 노드를 찾는 공간질의수행 단계로 나뉘는데 라우팅 단계에서는 [1]의 라우팅 알고리즘을 적용시켰다. 영역질의는 질의 범위에 포함되는 모든 노드를 찾는 질의로 질의 범위에 포함되는 모든 노드에 질의 메시지를 전달하기 위한 알고리즘을 제시하였다. k-최근접질의는 질의점에 가장 가까운 k개의 노드를 찾는 질의로 분산된 환경에서 하나의 노드가 자신이 질의 범위에 포함되는지 유무를 판단하기 어렵기 때문에 질의 노드에 임시 서버 역할을 맡겨 처리하였다.

삼각분할 갱신 및 공간질의처리는 기존의 서버-클라이언트 모델과 같이 한 노드에 집중되어 처리되는 것이 아니라 그것에 관여한 노드가 분산하여 처리하기 때문에 전체 발생하는 메시지 수가 서버-클라이언트 모델에 비해 본 논문에서 제안한 방법이 적음을 실험을 통해 알 수 있었다.

본 논문의 향후 과제는 다음과 같다. 먼저 들로니 삼각분할 방법에 대한 갱신 방법을 제안하는 것이다. 들로니 삼각분할 방법은 삼각분할 방법과는 달리 제약조건이 더 강하기 때문에 이 제약조건을 만족시켜주기 위해서는 삼각분할에 비해 더 많은 검사가 필요하다. 하지만 들로니 삼각분할 방법을 이용하면 질의처리를 삼각분할에 비해 쉽게 처리할 수 있다는 장점이 있다. 따라서 들로니 삼각분할을 이용한 갱신 방법을 만들어내면 보다 효율적으로 이동객체의 공간질의처리를 다룰 수 있을 것이다. 다음으로 본 논문에서는 노드는 한번에 하나씩만 이동한다고 가정하고 위치정보를 갱신해주는 알고리즘을 제시하였는데 현실 세계에 적용시켜 생각해 볼 때 노드는 동시에 하나가 아닌 여러 개가 동시에 움직일 수 있으므로 이를 처리해주기 위한 알고리즘 제시가 필요하다. 마지막으로 질의 처리 시 노드의 움직임을 고려한 알고리즘을 제시해야 한다. 본 논문에서는 질의 처리 시 노드는 이동하지 않는다고 가정하고 처리하였다. 이는 완벽히 이동 노드에 대한 질의 처리 방법을 제시했다고 볼 수 없다. 따라서 노드의 이동성을 고려한 접근 방법이 필요하다.

Acknowledge

본 논문은 과학기술부 한국과학재단 지정 한국항공대학교 부설 인터넷정보검색연구센터의 지원에 의함.
본 연구는 산업자원부의 지역혁신 인력양성사업의 연구결과로 수행되었음.

참고 문헌

[1] Lim Bog-Ja, P2P Spatial Query Processing by Delaunay Triangulation. *Master's thesis, Department of Computer Science, Pusan National University*, 2004
[2] Shahram Ghandeharizadeh, Antonios Daskos and Xinghua An, PePeR: A Distributed Range Addressing Space for Peer-to-Peer Systems. *In Proceedings of Databases, Information Systems, and Peer-to-Peer Computing*, pages 200-218, 2003
[3] Gnutella. http://gnutella.wego.com/
[4] David Karger, M. Frans Kaashoek, Ion Stoica, Robert Morris and Hari Balakrishnan, Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. *In ACM SIGCOMM Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149-160, 2001
[5] Leda. http://www.algorithmic-solutions.com/
[6] Jinbo Chen, Min Cai, Martin Frank and Pedro Szekely, MAAN: A Multi-Attribute Addressable Network for Grid Information Services. *In Proceedings of Grid computing*, pgae 184, 2003
[7] Napster. http://napster.com/
[8] Soribada. http://www.soribada.com/
[9] Mark Handley, Richard Karp, Sylvia Ratnasamy, Paul Francis and Scoot Shenker, A Scalable Content-Addressable Network. *In ACM SIGCOMM Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communications*, pages 161-172, 2001.
[10] Charles Perkins and Pravin Bhagwat, Highly Dynamic Destination-Sequenced Distance-Vector Routing for Mobile Computers. *In ACM SIGCOMM Proceedings of the Conference on Communications Architectures, protocols and applications*, pages 234-244, 1994.
[11] Brad Karp and H. T. Kung, *GPSR: Greedy Perimeter Stateless Routing for Wireless Netwokrs. In Proceedings of Mobile computing and networking*, pages 243-254, 2000.