단일방송채널환경에서 k-최근접질의 처리를 위한 힐버트 곡선과 최소영역 사각형 기반의 분산 공간 인덱싱 기법

(A Distributed Spatial Indexing Technique based on Hilbert Curve and MBR for k-NN Query Processing in a Single Broadcast Channel Environment)

이 정 형 [†] 정 성 원 [††]
(Junghyung Yi) (Sungwon Jung)

요 약 본 논문은 단일무선방송채널환경에서 힐버트곡선과 최소영역사각형을 이용하여 공간데이터를 방송하고 이를 가지고 k-최근접질의 처리를 효과적으로 처리하는 기법에 관한 논문이다. 기존 방식은 k-최근접질의 처리시 백트랙킹문제가 발생하여 질의처리에 오랜 시간이 걸리거나 검색범위를 빠르게 줄이지 못하여 많은 정보를 수신해야 하는 단점이 존재하였다. 제안하는 방법은 공간데이터를 힐버트 곡선 순서대로 방송하되 방송중인 공간데이터를 제외한 나머지 공간데이터를 최소영역사각형으로 그룹화하고 이를 인덱스 테이블로 구성하는 방법이다. 그리고 이를 이용하여 클라이언트가 알려지지 않은 데이터의 위치를 예측하여 빠르게 검색범위를 줄여나가 불필요한 정보를 제거하여 적은 튜닝시간과 접근지연시간을 갖도록 하는 것이다.

키워드 : 위치기반서비스 질의, k-최근접질의, 힐버트곡선, 최소영역사각형

Abstract This paper deals with an efficient index scheduling technique based on Hilbert curve and MBR for k-NN query in a single wireless broadcast channel environment. Previous works have two major problems. One is that they need a long time to process queries due to the back-tracking problem. The other is that they have to download too many spatial data since they can not reduce search space rapidly. Our proposed method broadcasts spatial data based on Hilbert curve order where a distributed index table is also broadcast with each spatial data. Each entry of index table represents the MBR which groups spatial data. By predicting the unknown location of spatial data, our proposed index scheme allows mobile clients to remove unnecessary data and to reduce search space rapidly. As a result, our method gives the decreased tuning time and access latency.

Key words : LBS query, k-NN query, Hilbert curve, MBR

1. 서 론

최근에는 무선 모바일 환경에서 위치기반 데이터를 이용하는 서비스에 대한 수요가 급격히 증가하고 있다. 이러한 위치기반 데이터를 이용하는 서비스를 우리는 위치기반 서비스(Location based Service)라고 하고, 이러한 위치기반 서비스에서 질의를 요청하는 클라이언트의 위치에 근거한 질의를 위치기반 질의라고 하며 대표적인 위치기반 질의인 k-최근접질의(k-NN: k-Nearest Neighbor) 질의를 처리하기 위한 연구가 활발히 진행되어 왔다. k-최근접질의는 n개의 공간 데이터들에 대해서 질의 포인트와 가장 가까운 k개의 공간 데이터를 찾는 질의를 의미한다. 그동안 진행된 k-최근접질의 처리 연구 중에 방송 환경에 적용시켜 구현한 연구로는 공간 채움 곡선(space filling curve)을 기반으로 한 연구와 R-tree를 기반으로 한 연구가 있었다[1-4].

† 학생회원 : 서강대학교 컴퓨터공학과
mice1004@naver.com
†† 종신회원 : 서강대학교 컴퓨터공학과 교수
jungsung@sogang.ac.kr
논문접수 : 2010년 2월 3일
심사완료 : 2010년 7월 16일

정보과학회논문지: 데이타베이스 제37권 제4호(2010.8)

기존 방식은 k-최근접질의 처리시 back-tracking문제가 발생하여 질의처리에 오랜 시간이 걸리거나 검색범위를 빠르게 줄이지 못해 필요이상의 많은 인덱스 정보와 후보 공간 데이터를 수신하는 단점이 존재한다[1-4]. 이러한 문제점은 결과적으로 질의어 응답시간을 증가시키고 모바일 기기의 배터리를 빠르게 소모시키는 결과를 가져온다. 이러한 문제점에 대처하기 위해 본 논문에서는 공간 데이터간의 지역성을 1차원상에서 효과적으로 나타내는 힐버트 곡선을 활용하여 공간 데이터를 방송하고 힐버트 곡선 순서에 의해 방송되는 공간데이터의 검색범위를 빠르게 줄이고 그룹 필터링 효과를 얻기 위해 최소영역사각형(MBR)을 사용하는 분산 인덱싱 기법인 DSITM(Distributed Spatial Indexing Technique based on MBR)과 이를 활용한 k-최근접질의 처리 기법을 제안한다.

이후의 본 논문의 구성은 다음과 같다. 2장에서는 힐버트곡선 및 최소영역사각형 기반의 분산 인덱싱 기법인 DSITM을 설명한다. 3장에서는 DSITM을 이용한 k-최근접질의 처리 방법에 대해서 설명한다. 4장에서는 성능 평가를 통해 제안하는 방법을 성능을 증명하고 5장에서는 성능평가에 따른 제안하는 방법에 대한 결론을 내린다.

2. 최소영역사각형 기반의 분산 인덱싱 기법

2.1 기본 아이디어

DSITM의 기본 아이디어는 그림 1에서 보여 지듯이 공간데이터를 힐버트 곡선 순서대로 방송하되 방송중인 공간데이터를 제외한 나머지 공간데이터를 최소영역사각형로 그룹화하고 이를 인덱스 테이블로 구성하는 것이다.

DSITM에서 각 방송 프레임은 공간데이터와 인덱스 테이블로 구성된다. 인덱스 테이블에는 최소영역사각형 정보가 포함되어 있다. 이 최소영역사각형 정보는 현재 공간데이터를 제외한 나머지 모든 공간데이터에 대해서 힐버트 인덱스 순서대로 만들어지며 최소영역사각형에 속한 공간데이터의 최대 개수는 r^i이다. 여기서 r은 주어진 exponential base이고 i는 이전까지 생성된 최소영역사각형의 개수이다. 즉 r이 2라면 힐버트 인덱스 첫 번째 최소영역사각형은 1개의 공간데이터만 가지고 두 번째 최소영역사각형은 2개의 공간데이터를, 세 번째 최소영역사각형은 4개의 공간데이터를 갖는다. 또한 최소영역사각형이 과도하게 커지는 것을 막기 위해서 최소영역사각형이 힐버트 곡선 상의 마지막 공간데이터를 만났다면 최소영역사각형은 마지막 공간데이터까지만 포함하고 첫 번째 공간데이터부터 새로운 최소영역사각형을 생성한다.

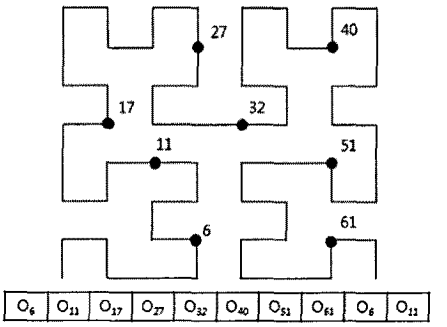


그림 1 DSITM의 방송순서

그림 2는 전체 공간데이터가 8개이고 r이 2일 때의 최소영역사각형 생성방법이다. 그림 2는 힐버트 인덱스가 6인 공간데이터 O_6의 최소영역사각형 생성방법이다. O_6의 경우 이후 방송되는 공간데이터가 순서대로 $\{O_{11}, O_{17}, O_{27}, O_{32}, O_{40}, O_{51}, O_{61}\}$이다. 첫 번째 최소영역사각형은 $\{O_{11}\}$를 갖고 두 번째 최소영역사각형은 $\{O_{17}, O_{27}\}$를 갖으며 세 번째 최소영역사각형은 $\{O_{32}, O_{40}, O_{51}, O_{61}\}$를 갖는다. 따라서 그림 2와 같은 형태로 최소영역사각형이 구성된다.

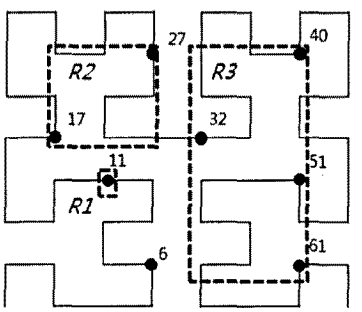


그림 2 DSITM의 최소영역사각형 정보 생성방법

2.2 인덱스 테이블의 크기 및 구조

인덱스 테이블은 여러 엔트리로 구성된다. 각 엔트리는 하나의 최소영역사각형과 대응되며 해당 최소영역사각형의 정보를 가진다. 따라서 인덱스 테이블안의 엔트리 개수는 공간데이터를 전부 포함하기 위해 필요한 최소영역사각형의 개수와 같다. 이를 n이라 하고 전체 데이터의 개수를 n_F라고 한다면 최소영역사각형에 속한 모든 공간데이터의 개수의 합은 (n_F-1) 보다 크거나 같아야 한다. 또 최소영역사각형에 속한 공간데이터의 개수는 지수 배로 증가한다. 최소영역사각형이 힐버트 곡선 상에서 마지막 공간데이터를 만났을 때만 최소영역사각형이 다 차지 않는다. 그렇기 때문에 실제로는 한

개의 최소영역사각형이 더 필요하다. 따라서 인덱스 테이블의 크기는 아래와 같다.

$$n = \lceil log_r\{(n_F-1)(r-1)+1\} \rceil + 1$$

인덱스 테이블은 n개의 엔트리로 구성되며 각 엔트리는 그림 3과 같은 제어 정보를 갖는다. 엔트리의 *MBRinfo* 필드는 최소영역사각형의 좌측 상단과 우측 하단의 위치를 나타내어 최소영역사각형의 크기 및 위치를 나타낸다. *NodeNumber* 필드는 최소영역사각형에 속한 공간 데이터의 개수로 각 최소영역사각형에 속한 데이터의 개수를 알면 질의점과 최소영역사각형간의 maxdist를 k-최근접질의 처리에 활용할 수 있고 전체 데이터의 개수를 파악할 수 있다. 또한 최소영역사각형이 검색범위 안에 속해서 살펴봐야 한다면 최소영역사각형안에 속한 데이터 중 가장 먼저 방송되는 데이터의 힐버트 곡선 인덱스 값(*HCindex*)과 방송시간(*Offset*)을 알아야 back tracking 문제를 해결할 수 있다. 그리고 이 데이터의 힐버트 곡선 인덱스를 알면 검색범위를 결정하는데 사용할 수 있다.

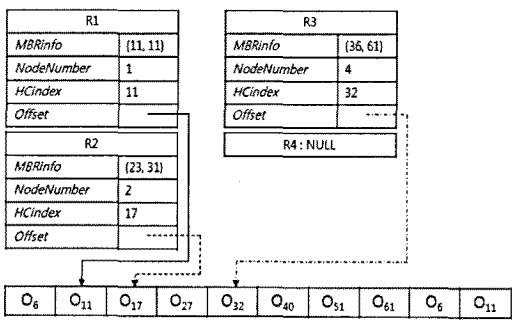


그림 3 DSITM의 O_6에 대응하는 인덱스 테이블

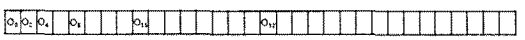
그림 3은 인덱스 테이블의 예제를 보여준다. 그림 2와 같이 8개의 공간데이터가 존재하고 r이 2일 때 O_6에 대응하는 DSITM의 인덱스 엔트리 R1, R2, R3는 그림 3과 같다.

3. k-최근접질의 처리 방법

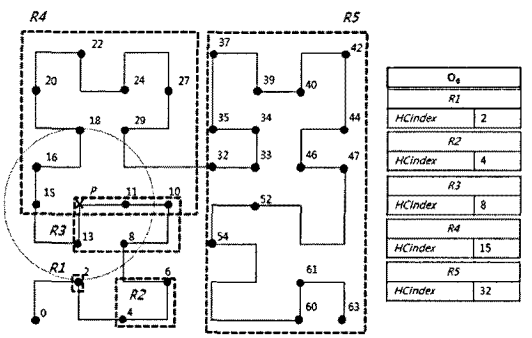
DSITM 인덱싱 기법은 k-최근접질의를 처리하기위해 크게 두 번의 동작을 수행한다. 먼저 실제로 알려진 데이터와 예측한 데이터를 통해 검색범위를 결정짓고 그 다음 검색범위 밖에 존재하는 데이터를 제거한다. 각각의 단계에서는 *HC_table* 과 *E_table* 이라는 두 개의 테이블을 사용한다. *HC_table* 은 검색범위를 결정짓기 전에 업데이트 되며 수신한 인덱스 정보를 유지하는데 사용된다. *E_table* 은 검색범위 밖에 존재하는 데이터를 결정할 때 업데이트 되고 다음에 수신할 정보를 결정짓는데 사용된다.

3.1 검색범위 결정

k-최근접질의의 검색범위는 질의 점 p를 중심으로 원을 그렸을 때 최소한 k개 이상의 노드가 있음을 보장하는 범위이다. 따라서 검색범위를 결정짓는 데는 두 가지 요소를 통해서 할 수 있다. 수신한 데이터나 인덱스 테이블의 *HCindex* 에 명시된 각 최소영역사각형에 속한 노드 중 가장 먼저 방송되는 노드의 힐버트 인덱스와 같이 정확하게 그 위치를 파악할 수 있는 정보를 가지고 검색범위를 결정지을 수 있다. 또 *minmaxdist* 와 *maxdist* 같이 실제 공간데이터의 위치는 알지 못해도 그 위치를 예측하게 하는 정보를 가지고 검색범위를 결정지을 수 있다. 최소영역사각형안에 존재하는 여러 공간데이터 중 *minmaxdist* 보다 가까운 데이터가 적어도 한 개 존재하고 모든 공간데이터는 *maxdist* 보다 가깝게 존재하기 때문이다.



(a) O_0 을 수신한 직후 *HC_table*



(b) O_0 을 가지고 검색범위 개략 결정

그림 4 알려진 공간데이터를 가지고 검색범위 개략 결정하기

그림 4(a)는 32개의 공간데이터가 존재할 때 모바일 클라이언트가 최초로 O_0 을 수신하였을 때 *HC_table* 을 나타낸다. 이 때 알려진 정보는 각각의 최소영역사각형 안에 존재하는 공간데이터 중 가장 먼저 방송되는 공간데이터이므로 이는 $\{O_2, O_4, O_8, O_{15}, O_{32}\}$가 된다. 따라서 모바일클라이언트가 3NN을 수행하고자 할 때 이중 가장 가까운 $\{O_2, O_8, O_{15}\}$가 후보가 된다. 검색범위는 그림 4(b)에서와 같이 질의 점에서 부터 가장 먼 O_2까지의 거리가 된다.

알려진 공간데이터를 통해 검색범위를 개략 검색범위를 결정지은 다음에 현재 수신한 방송 프레임의 인덱스

테이블의 최소영역사각형 정보를 통해 검색범위를 확정짓는다. R_1의 경우 포함하는 공간데이터가 1개이고 이는 이미 알려졌으므로 예측할만한 남은 공간데이터가 존재하지 않는다. 또 R_2와 R_5의 경우 개략적으로 판단한 검색범위 바깥에 존재하기 때문에 이 안에 존재하는 어떠한 공간데이터도 검색범위에 포함되지 않는다. 따라서 R_3와 R_4만이 공간데이터의 예측을 통해 검색범위를 확정짓는데 사용된다. 알려진 공간데이터를 통해 검색범위를 개략 검색범위를 결정지은 다음에 현재 수신한 방송 프레임의 인덱스 테이블의 최소영역사각형 정보를 통해 검색범위를 확정짓는다. R_3의 경우 질의점 p를 중심으로 minmaxdist(R_3, p)를 반지름으로 원을 그렸을 때 R_3에 속한 공간데이터 중 알려진 것은 오직 O_8인데 이 공간데이터는 이원에 포함되지 않는다. 따라서 R_3 안에 minmaxdist(R_3, p) 보다 가깝거나 적어도 동일한 거리를 갖는 공간데이터가 적어도 1개 존재함을 예측할 수 있다. 따라서 이 정보를 추가해서 새로운 검색범위를 확정지을 수 있다. 알려진 공간데이터와 예측한 검색범위를 질의 점에서 가까운 순서대로 정렬하면 {O_{15}, minmaxdist(R_3, p), O_8, O_2, O_4, O_0, O_{32}}가 된다. 따라서 여기서 질의 점 p에 3번째로 가까운 O_8까지의 거리가 검색범위가 된다. 그림 5는 그림 4의 개략적인 검색범위를 공간데이터의 예측을 통해 확정지은 것을 보인다.

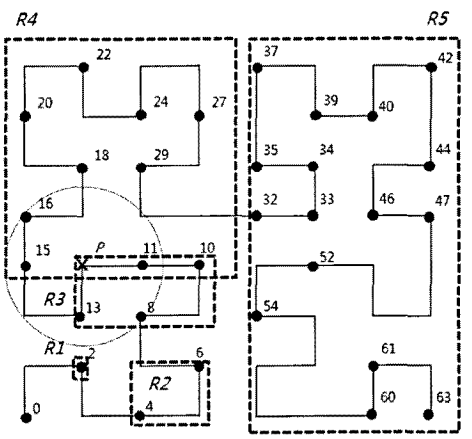
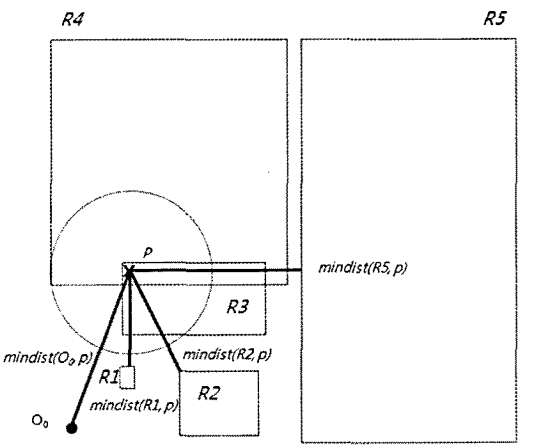


그림 5 공간데이터의 예측을 통해 검색범위 확정하기

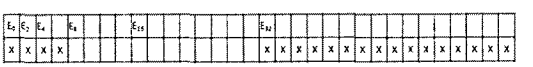
3.2 검색범위 밖의 데이터 제거

검색범위가 결정되면 검색범위 밖에 존재하는 데이터를 질의 후보에서 제거해서 수신하지 않도록 해야 튜닝시간을 줄일 수 있다. DSITM은 어떤 공간데이터가 제거되었는지 기억하기 위해서 E_table을 유지한다. 만약 공간데이터가 제거되었다면 E_table에 해당 공간데이터가 추가된다.

DSITM은 두 가지 데이터 제거 방식을 제안한다. 첫 번째는 최소영역사각형의 특성을 이용한 것이고 두 번째는 힐버트 곡선의 인덱스 값이 나올 수 있는 범위를 예측하는 것이다. 최소영역사각형안의 모든 공간데이터는 질의 점으로부터 mindist보다 더 멀거나 같은 위치에 존재한다. 따라서 질의 점과 최소영역사각형의 mindist를 구했을 때 검색범위보다 더 멀리 떨어져 있다면 해당 최소영역사각형과 최소영역사각형내의 모든 공간데이터는 제거가능하다. 그림 5은 O_0을 수신한 이후 결정된 검색범위와 O_0의 최소영역사각형을 나타낸다. 검색범위가 정해졌으므로 질의 점과 최소영역사각형 사이의 mindist를 구해서 k-최근접질의 처리에 불필요한 데이터를 제거할 수 있다. 그림 6(a)와 같이 O_0, R_1, R_2, R_5는 질의 점 p에서부터 mindist를 구하였을 때 모두 질의 점 p에서 검색범위를 결정짓는 O_8까지의 거리보다 멀다. 즉 O_0, R_1, R_2, R_5는 검색범위 밖에 존재하므로 제거 가능하다. R_1은 O_2부터 1개의 공간데이터로 구성되고 R_2은 O_4부터 2개의 공간데이터로 구성되며 R_5은 O_{32}부터 16개의 공간데이터로 구성된다. 따라서 그림 6(b)와 같이 E_table을 갱신할 수 있다.



(a) O_0 수신 후 최소영역사각형단위의 제거



(b) O_0 수신 후 갱신된 E_table

그림 6 mindist를 사용해 검색범위 밖의 데이터 제거

3.3 DSITM의 k-최근접질의 처리 알고리즘

DSITM은 그림 7에 나타나있는 k-최근접질의 처리 알고리즘을 따라 k-최근접질의를 처리한다.

```
Input : a query point, p, and the number of nearest neighbors, k;
Output : k nearest neighbors to p;
Function: insert (result, dist, point, k, num) - insert point or NULL which
          distance from p is dist into result as much as num and maintain
          only the k nearest candidates in result; return the maximal
          distance between query point and the candidates.
          mindist(MBR, p) - return the minimal distance between MBR
          and query point p.
          minmaxdist (MBR, p) - return the minimum of the maximum
          possible distance between MBR and query point p.
          maxdist(MBR, p) - return the maximum distance between MBR
          and query point p.
          dist (point, p) - return the distance between point and query
          point p.

radius = ∞; new_r = ∞; cnt = 0; next_index = 0;
result = ∅; new_result = ∅; HC_table = ∅; E_table = ∅;
begin the initial probe and retrieve a frame Fc;
create and initialize HC_table and E_table;
while Fc do
    new_result = ∅; new_r = ∞;
    HC_table[i] = Oi, data object Oi in the Fc;
    for all the pointers Pi in the index table of Fc do
        HC_table[i] = Pi;
    for all the MBR Rj in the index table of Fc do
        for all data object Oi in the MBR Rj do
            cnt = 0;
            if HC_table[i] is not empty and E_table[i] is 0 than
                new_r = insert(new_result, dist(Oi, p), Oi, k, 1);
                cnt++;
            end if
        end for
        if there is not known object on boundary which within minmaxdist than
                new_r = insert(new_result, minmaxdist(Rj, p), NULL, k,1);
                cnt++;
        end if
        if (cnt != Rj.NodeNumber) than
                new_r = insert(new_result, maxdist(Rj, p), NULL, k.
                Rj.NodeNumber-cnt);
        endif
    end for
    if radius <= new_r than
        radius = new_r;
        result = new_result;
    end if
    for all the MBR Rj in the index table of Fc do
        if (mindist(Rj, p) > radius) than
            for all data object Oi in the MBR Rj do
                E_table[i] = 1;
            end for
        end if
    end for
    for all known object Oi do
        if next known object Oi exist and dist(all object in [Oi, Oi), p) > radius
        than
            for all data object Oi in the [Oi, Oi) do
                E_table[i] = 1;
            end for
        endif
    end for
    next_index = i witch currently accessed data object Oi is in frame Fc.
    for check j from j = next_index to end of table size do
        if HC_table[j] is not empty and E_table[j] is 0 than
            next_index = j;
            break;
        end if
    end for
    if next_index = i witch currently accessed data object Oi is in frame Fc than
        Fc = NULL;
    else
    endif
end while
return result;
```

그림 7 DSITM의 k-최근접질의 처리 알고리즘

4. 성능 평가

이 장에서는 본 논문에서 제안한 DSITM의 인덱스 기법의 성능을 확인하기 위해서 최근 관련연구인 DSI conservative[3], DSI aggressive[3] 그리고 w-disk[4]와 함께 비교 성능 실험을 수행하였다. 실험에 사용된 공간데이터는 [5]에서 사용된 점으로 표현된 그리스의 도시와 마을에 대한 5,922개의 데이터들로 구성된다. 각 실험에서는 1,000번의 질의를 발생시켜 처리하였고 질의 점으로부터 각 알고리즘이 k-최근접질의를 완료하기까지 바이트 단위의 평균 접근 지연시간과 평균 튜닝 시간을 측정했다. 또한 모바일 클라이언트가 채널을 수신하기 시작하는 시점은 임의로 결정하였으며 모든 테스트에 동일하게 적용하였다. 본 실험에서는 k값의 변화에 따라 제안하는 기법의 성능을 알아본다.

그림 8에서 DSITM이 가장 적은 평균 접근지연시간을 보였다. DSITM은 DSI의 Conservative 방식보다 평균적으로 약 73%의 접근시간을 보였고 Aggressive 방식보다 약 70% 그리고 w-disk보다 약 74%의 접근시간을 보였다. 그림 9에서는 DSI Aggressive가 가장 적은 튜닝시간을 보였고 그 다음은 DSITM이 적은 튜닝시간을 보였다. DSI Aggressive 방식은 질의 점에 가장 가까운 공간 데이터를 우선 찾고 이후 k-최근접질의를 수행한다. 힐버트 곡선은 공간 채움 곡선 중 최적

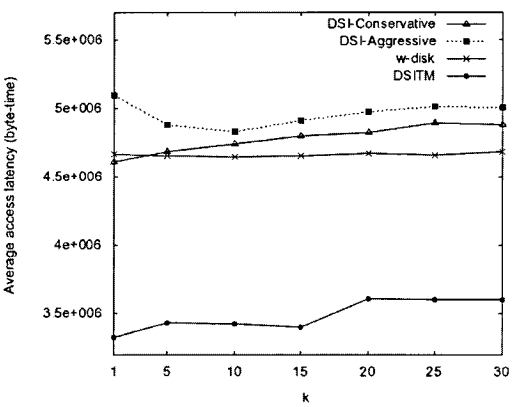


그림 8 k의 변화에 따른 평균 접근 지연시간

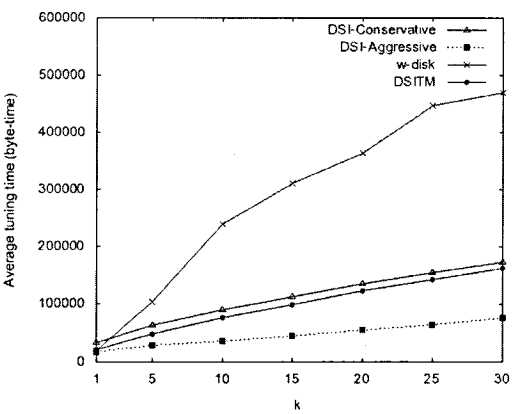


그림 9 k의 변화에 따른 평균 튜닝 시간

의 지역성을 가지므로 이렇게 질의 점에서 가장 가까운 공간데이터를 찾으면 상대적으로 인접한 지역에 k-최근접질의의 답이 되는 공간데이터들이 존재하게 된다. 따라서 가장 적은 평균 튜닝시간을 갖게 된다. DSITM은 DSI Conservative 방식 대비 약 88%의 튜닝시간을 가졌고 w-disk 방식 대비 약 34%의 튜닝시간을 가졌다.

5. 결 론

본 논문에서 우리는 단일 무선방송채널 환경에서 k-최근접질의 처리를 위해 최소영역사각형을 활용하면서 힐버트 곡선을 기반으로 하는 새로운 공간 인덱싱 기법인 DSITM 과 이를 활용한 k-최근접질의 처리 기법을 제안하였다. 이는 최소영역사각형의 특성을 활용해서 검색범위를 빠르게 줄이고 불필요한 공간데이터를 제거하여 전체적으로 빠른 접근 지연시간과 적은 튜닝시간을 가지고 정확한 k-최근접질의를 수행하여 좋은 성능을 보여주었다.

참 고 문 헌

[1] Jun Zhang, Manli Zhu, Dimitris Papadias, "Location-based Spatial Queries," *Proceedings of the ACM SIGMOD international conference on Management of Data (SIGMOD'03)*, pp.443-454, June 2003.

[2] B. Zheng, W.C. Lee, and D.L. Lee, "Spatial Queries in Wireless Broadcast Systems," *Wireless Network*, 10(6), pp.723-736, December, 2004.

[3] W.C. Lee, B. Zheng, "DSI: A Fully Distributed Spatial Index for Location-based Wireless Broadcast Services," *Proceedings of the 21st International Conference on Data Engineering*, pp.349-358, 2005.

[4] Chuan-Ming Liu, Shu-Yu Fu, "Effective Protocols for kNN Search on Broadcast Multi-Dimensional Index Trees," *Information Systems*, 33, pp.18-35, 2008.

[5] Spatial Datasets in 2D Space, http://www.rtreeportal.org/datasets/spatial/ greece/cities_loc.zip, 2006.

이 정 형

2000년 3월~2008년 2월 서강대학교 컴퓨터공학과 학사. 2008년 3월~2010년 2월 서강대학교 컴퓨터공학과 석사. 2010년 2월~현재 삼성증권. 관심분야는 이동통신, 모파일 컴퓨팅 시스템, 모바일 데이터 베이스, 모바일 트랜잭션, 모바일 커머스

정 성 원

1988년 서강대학교 전자계산학 학사. 1990년 M.S. in Computer Science at Michigan State University. 1995년 Ph.D. in Computer Science at Michigan State University. 1997년~2000년 한국전산원 선임연구원. 2000년~현재 서강대학교 컴퓨터학과 교수. 관심분야는 Mobile Databases, LBS, Mobile Computing Systems, Spatial Databases, Telematics, GIS