Spatial Skyline을 계산하는 기하 알고리즘

손완빈 안희갑 황승원°
포항공과대학교 컴퓨터공학과
mnbiny@postech.ac.kr, heekap@postech.ac.kr, swhwang@postech.ac.kr

Computing The Spatial Skyline

Wan-Bin Son, Hee-Kap Ahn, Seung-won Hwang°
Department of Computer Science and Engineering, POSTECH

요 약

본 논문은 Data mining에서 선호도 분석 등에 사용되는 Skyline Query[2] 중 자료의 속성에 spatial한 성질이 있을 때 사용할 수 있는 Spatial Skyline Query[3] 문제에 대해 연구한다. 우선 Spatial Skyline 집합을 추출하기 위한 기존의 알고리즘의 문제점을 짚어보고 보다 개선된 알고리즘을 제시한다. 또한 전체 Spatial Skyline 집합이 아닌 그 중 더 의미 있을 수 있는 부분 집합을 좀 더 빠른 시간 복잡도에 구하는 방법 또한 제시한다.

1. 서 론

Skyline Query[2]는 Data mining 분야에서 선호도(preference) 분석을 위해 널리 사용되는 방법이다. 선호도 분석은 특정 자료의 집합 내에서 기준에 따라 좀 더 높은 선호도를 가지는 자료들을 추출해내는 방법이다. 다양한 속성의 자료들을 다루는 Data mining 분야에서는 자료의 속성이 spatial한 성질을 가질 때의 선호도 분석 방법이 필요하다. 예를 들면 현실 세계의 위치 정보 자료 같은 경우 자료간의 거리 관계가 선호도 판단의 기준이 될 수 있다. 이와 같은 필요에 의해 제시된 'Spatial skyline Queries'[3]은 spatial 한 속성들의 관계가 중요한 자료들의 선호도 분석에 중요하게 사용될 수 있다. 이 논문에서는 Spatial skyline을 찾는 기존 알고리즘의 문제점을 지적하고 개선된 알고리즘을 제시한다. 또한 선호도 측면에서 좀 더 좋은 평가를 받을 수 있는 Spatial skyline의 부분 집합을 전체를 추출할 때보다 빠른 시간 복잡도에 추출하는 알고리즘 을 제시한다.

2. 문제 정의

자료점의 집합 P가 있고, 같은 속성 항목을 가지는 질의점의 집합 Q가 있다고 하자. 우선 Spatial skyline을 정의하기 위해 'spatially dominate' 관계에 대해 정의한다. d(a,b)는 a와 b 점 사이의 Euclidean distance를 의미한다. 어떤 두 점 p와 p1이 있다고 하자. 만약 p가 p1을 Q에 대해 spatially dominate 한다면, 모든 $q \in Q$에 대하여 $d(p,q) \le d(p1,q)$이고, 어떤 $q1 \in Q$ 에 대하여 $d(p,q1) < d(p1,q1)$ 인 관계가 성립해야 한다. 또한 역도 성립한다. 다음으로 Spatial skyline을 정의한다. 어떤 점 $p \in P$가 Q에 대해 P의 Spatial skyline이려면 다른 모든 P의 점으로부터 spatially dominate 되지 않아야 한다. 또한 역도 성립한다. 이 때 p점을 Spatial skyline point 라고 부른다. 이 논문에서는 자료점의 집합 P에 대해, 질의점의 집합 Q가 주어질 때 P의 Spatial skyline points를 추출하는 문제를 다룬다.

논문에서 사용할 기호들을 정리하면 다음과 같다

표 1 각 기호의 의미

기호	의미
P	자료점의 집합
Q	질의점의 집합. P의 집합보다 크기가 작다.
S	Q에대한 P의 Spatial skyline 집합
CH(Q)	질의점의 convex hull

그 외에 각각의 자료점은 p을, 각각의 질의점은 q를 사용하여 나타냈다. $|\ |$는 집합의 원소 개수를 나타내는데 사용하였다.

3. 기존의 연구 결과

현재까지 Spatial Skyline 집합을 찾는 기존에 알려진 방법 가운데 Voronoi-based Spatial Skyline(VS2) 알고리즘[3]이 있다. 이 알고리즘은 P의 보로노이 다이어그램[1]과 Q의 convex hul[1]이 주어져 있을 때 $O(|S|^2|CH(Q)| + \sqrt{|P|})$ 시간 복잡도로 모든 Spatial Skyline points를 찾을 수 있다고 알려져 있다. VS2 알고리즘의 이 시간 복잡도는 spatially dominate 여부를 검사하는 P 점의 수를 $O(|S|)$로 할 수 있다는 가정 하에 성립한다. 하지만 이 가정이 참이 아님을 아래 반례로 보일

수 있다.

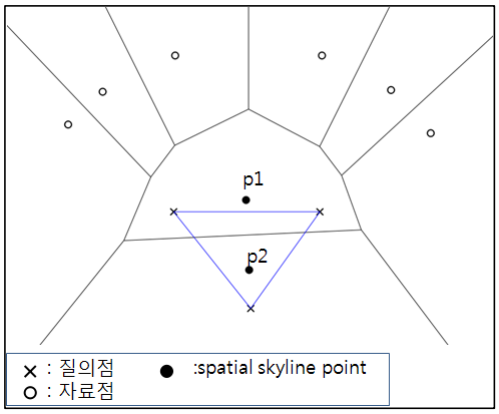

그림 1 VS2 알고리즘의 검사 한정 예

그림 1은 세 개의 질의점과 8개의 자료점으로 구성되어 있으며, 질의점의 convex hull과 자료점의 보로노이 다이어그램이 주어져 있다. p1과 p2는 Spatial skyline point이고 나머지 자료점들은 모두 Spatial skyline point가 아님에도 VS2 알고리즘은 모든 자료점에 대해 Spatial skyline 여부를 검사한다. 최악의 경우 p1,p2를 제외한 나머지 P점이 그림과 같이 모두 p1의 이웃이 될 수 있으므로 VS2 알고리즘이 검사하는 자료점의 개수는 Ω(|P|)이 될 수 있다. 이 점들을 heap에 저장하고 사용하므로 이 알고리즘의 올바른 시간 복잡도는 O(|P|(|S|+|CH(Q)|+log|P|))이다.

또한 VS2 알고리즘은 모든 Spatial skyline point를 찾아낼 수 없다. VS2 알고리즘은 P점과 CH(Q) 꼭지점의 거리 합의 순서에 따라 자료점들을 검사해 나간다. 이 때 어떤 자료점 p점에 대해 p의 이웃이 모두 Spatial skyline 이웃을 가지지 않는다면, p점은 skyline point가 아니라고 가정한다. 따라서 이 알고리즘은 p와 같은 점은 검사하지 않는다.

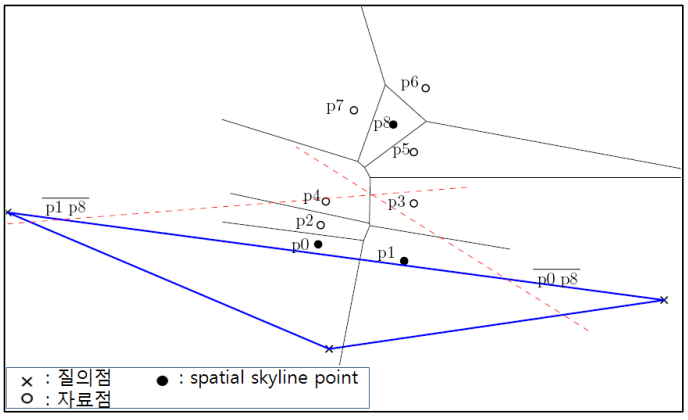

그림 2 VS2 알고리즘의 추출 누락 예

그림 2는 세 개의 질의점과 9개의 자료점으로 이루어져 있고, 질의점의 convex hull 과 자료점의 보로노이 다이어그램이 주어져 있다. 여기서 p8의 이웃인 p4,p5,p6,p7은 Spatial skyline point가 아니다. 또한 p8를 제외한 p4,p5,p6,p7의 이웃 역시 Spatial skyline point가 아니다. VS2 알고리즘은 이와 같은 경우 p8은 Spatial skyline point가 아니라고 판단하고 검사하지 않는다. 하지만 p8은 Spatial skyline point이며 VS2 알고리즘은 이를 누락하게 된다.

4. 본 론

본론에서는 Spatial skyline과 spatially dominate의 성질에 대해 증명하고, 이를 이용해 Spatial skyline을 추출하는 방법을 제시한다. 본론의 알고리즘은 두 개의 속성을 가지는 2차원의 자료로 생각하여 구현하였으며, d-차원의 문제로 확장하는 것도 가능하다.

소정리1. p1이 p2를 spatially dominate 할 수 없다면, d(p1, q)>d(p2, q) 인 어떤 점 q∈Q 가 존재하거나 또는 모든 q∈Q에 대해서 d(p1,q)≥d(p2,q) 인 관계가 성립한다. 또한 이의 역도 성립한다.

증명. spatially dominate의 정의에 대우를 적용하면 위의 명제를 얻을 수 있다.

소정리2. 임의의 자료점 p2 보다 p1이 하나의 질의점 q에 가깝다면 p2는 p1을 spatially dominate 할 수 없다.

증명. 어떤 q∈Q에 대해서 d(p2,q)>d(p1,q) 인 관계가 성립한다면 소정리1에 의해서 p2는 p1을 spatially dominate 할 수 없다.

소정리3. 임의의 자료점 p3가 p2을 spatially dominate 할 수 없다면, p3는 p2의 dominator p1도 spatially dominate 할 수 없다.

증명. p3가 p2를 dominate 하지 않기 때문에 소정리1에 의해 d(p2,q1) < d(p3,q1) 인 어떤 q1이 존재하거나 모든 q∈Q에 대해 d(p2,q) ≤ d(p3,q)가 성립한다.

경우1. d(p2,q1) < d(p3,q1) 인 어떤 q1이 존재할 때.
spatially dominate의 정의에 의해 모든 q∈Q에 대해 d(p1,q) ≤ d(p2,q)가 성립한다. q1에 대해서 d(p1,q1)≤ d(p2,q1)<d(p3,q1) 인 관계가 성립하기 때문에 소정리1에 의해서 p3는 p1를 spatially dominate 할 수 없다.

경우2. 모든 q∈Q에 대해 d(p2,q) ≤ d(p3,q) 일 때.
spatially dominate의 정의에 의해 어떤 q1∈Q에 대해 d(p1,q1)<d(p2,q1)이 성립한다. 따라서 d(p1,q1)< d(p2,q1)≤d(p3,q1)이 성립한다. 소정리1에 의해 p3는 p1를 spatially dominate 할 수 없다.

소정리4. 임의의 자료점 p1이 Spatial skyline point가 아

니라면 p1을 spatially dominate 하는 Spatial skyline point p2가 존재한다.

증명. p1이 Spatial skyline point가 아니기 때문에 p1을 spatially dominate 하는 자료점이 존재한다. p1의 dominator 중 모든 질의점과의 거리의 합이 가장 작은 자료점을 p2이라고 하자. 모든 질의점과의 거리 합이 가장 작기 때문에 다른 각각의 dominator보다 가까운 질의점이 존재한다. 따라서 소정리2에 의해서 p2는 p1의 dominator에 의해서는 dominate 되지 않는다. 만약 p2와 모든 질의점과의 거리 합이 같은 점이 있더라고 p2가 가까운 질의점이 있거나, 모든 질의점과의 거리가 같아야 하므로 p2를 dominate할 수 없다. 또한 소정리3에 의해 p1을 dominate 하지 않는 점에 의해서도 dominate 되지 않는다. 따라서 p2는 어떤 자료점에 의해서도 spatially dominate되지 않으며, Spatial skyline의 정의에 의해 p2는 Spatial skyline point이다. 그러므로 위의 명제는 성립한다.

4.1 전체 Spatial skyline points 추출

전체 Spatial skyline points를 효율적으로 찾기 위해서는 두 자료점간의 spatially dominate 여부를 빠르게 판단해야 하고, 또한 판단하는 횟수를 줄여야 한다.

4.1.1 효율적인 spatially dominate 여부 판단

효율적으로 Spatial skyline points를 찾기 위해서는 어떤 자료점이 다른 자료점을 spatially dominate 하는지를 빠르게 판단하는 방법이 필요하다. 'The spatial skyline Queries' [3]에서는 dominate 여부를 Q의 모든 점과의 비교가 아닌, CH(Q)의 꼭지점과의 거리 비교만을 통해서도 알 수 있음을 증명하였다. 따라서 두 자료점의 dominate 문제는 CH(Q)와 두 자료점의 이등분선이 만나는지 여부를 판단하는 문제로 바꿔 생각할 수 있다. convex hull과 선이 만나는지 여부는 이진 찾기를 이용하여 O(log|CH(Q)|) 시간에 구할 수 있다.

소정리 5. CH(Q)가 주어졌을 때 두 자료점이 spatially domiante 하는지 여부는 O(log|CH(Q)|) 시간에 계산 가능하다.

4.1.2 spatially dominate 판단 횟수 한정

전체 알고리즘을 빠르게 하기 위해서는 spatially dominate 판단 횟수를 줄여야 한다. 이를 위해 소정리2를 이용한다. 우선 모든 자료점 p와 CH(Q)의 한 꼭지점 q1과의 거리를 계산한다. 이 거리의 오름차순 정렬 결과를 배열 A에 (자료점 번호, 거리) 형태로 저장한다. 소정리2에 의해서 d(p,q1)<d(p1,q1) 이면 p1은 p를 dominate 할 수 없다. 따라서 배열 A에서 먼저 위치한 자료점은 뒤에 위치한 자료점에 의해 dominate 되지 않는다. (단 거리가 같은 경우는 없다고 생각한다. 이에 대해서는 뒤에 다루기로 한다.) 또한 소정리4에 의해서 어떤 자료점이 Spatial skyline points 인지 판단하기 위해서는 현재까지의 Spatial skyline points와만 dominate 여부를 판단해 봐도 된다. 결론적으로 어떤 자료점 p가 spatial skyline point 인지 여부는 A에서 p보다 앞에 위치한 자료점 중 spatial skyline point인 점과만 dominate 여부를 판단해 보면 된다. 따라서 각 자료점의 dominate 판단 횟수는 O(|S|)가 된다.

만약 q1과 거리가 같은 자료점들이 존재한다면 그 점들을 q1이 아닌 CH(Q)의 한 꼭지점 q2와의 거리로 정렬한다. A내부에서 q1값에 의한 정렬 순서는 유지한다. 또한 q1,q2에 대해 거리가 같은 점들은 q1,q2가 아닌 CH(Q)의 한 꼭지점 q3와의 거리로 정렬한다. 마찬가지로 A내부에서 q1,q2 값에 의한 정렬 순서는 유지한다. q1,q2,q3는 convex hull의 꼭지점이기 때문에 직선상에 있지 않으며 따라서 세 점에 대해서 거리가 같은 점은 하나 밖에 존재하지 않는다. 소정리2와 소정리 4에 의해 여전히 A의 순서를 따라가면 각 자료점의 dominate 판단 횟수는 O(|S|)가 된다.

소정리 6. CH(Q)과 자료점간의 거리 관계와 소정리 4를 이용하면, 한 자료점이 spatial skyline point임을 보이기 위해 dominate를 확인하는 횟수는 O(|S|)가 된다.

4.1.3 알고리즘 및 분석

전체 알고리즘은 다음과 같다.

```
(1) 배열 A 와 리스트 S 초기화
(2) Q의 convex hull 계산
(3) CH(Q)의 한 꼭지점 q1에 대해 모든 자료점과의 거리를
계산하여 A에 (자료점 번호,거리)의 형태로 저장
(4) A를 거리의 오름차순으로 정렬. 이 때 값이 같은 경우는
4.1.2 의 방식으로 정렬
(5) for i=0 to n
(6) A[i]가 S에 의해 spatially dominate 되는지 판단
(7)     if (dominate 되지 않으면 )
(8)         A[i]의 자료점을 S에 삽입
        end
    end
(9) return S
```

시간 복잡도를 계산해보면 (2)의 convex hull 계산이 O(|Q|log|Q|), (3)의 거리 계산이 O(|P|) 이다. (4)의 정렬은 기본적으로 O(|P|log|P|) 시간이 걸리며 같은 거리의 값이 있는 경우도 최대 세 점에 대해서만 정렬해 보면 되기 때문에 O(|P|log|P|)이 된다. (6)은 소정리 6에 의해 O(|S|)만큼 dominate를 판단하게 되고 각 dominate 당 소정리 5에 의해 O(log|CH(Q)|) 시간이 걸리므로 (5)~(9)까지의 과정은 O(|P||S|log|CH(Q)|)가 된다. |Q|<|P|이기 때문에 전체 시간 복잡도는

O(|P|(|S|log|CH(Q)|+log|P|))가 된다.

4.2 Spatial skyline points의 부분집합 추출

'The Spatial Skyline Queries' [3]에서는 CH(Q) 내부의 P점과, CH(Q)와 겹치는 P의 보로노이 셀의 중심점은 spatial skyline point임을 증명하였다. 이 점들은 질의점들의 convex hull을 둘러싸는 영역에 있으며 이 범위 밖의 spatial skyline point 들에 비해 선호도면에서 더 좋은 평가를 받을 수 있다. 따라서 모든 spatial skyline points는 아니지만, 이 점들을 빠른 시간 내에 추출할 수 있다면 시간과 선호도의 측면에서 사용자의 요구를 충족시킬 수 있다.

빠른 시간 내에 위의 조건을 만족하는 P의 점들을 구하기 위해 다음과 같은 방법 사용하였다. CH(Q)의 한 꼭지점을 포함하는 보로노이 셀을 point location[1] 방법으로 구한다. 그리고 그 셀에서부터 시계 방향으로 CH(Q)의 선분을 따라가며 CH(Q)와 겹치는 보로노이 셀의 중심 자료점을 S에 저장한다. CH(Q)을 한바퀴 돌고 나면 CH(Q)을 둘러싸고 있는 보로노이 셀들은 모두 알 수 있으므로, delaunay graph를 따라가며 현재까지 구한 S가 둘러싼 영역 안의 자료점들을 모두 S에 저장하면 원하는 spatial skyline points의 부분집합을 구할 수 있다. 구체적인 알고리즘은 다음과 같다.

(1) 리스트 S 초기화
(2) Q의 convex hull 계산
(3) P의 보로노이 다이어그램 계산
(4) q1<-CH(Q)의 한 꼭지점
  q2<-CH(Q)에서 q1의 시계 방향 다음 꼭지점
(5) p<- q1을 포함하는 보로노이 셀의 중심
(6) while(true)
(7) $\overline{q1q2}$와 p를 중심으로 하는 보로노이 셀 교점 계산
(8) if(교점이 두 개이면)
(9) p<-q2에 가까운 교점을 공유하는 보로노이 셀 중심
  S에 p 삽입.
(10) else if(교점이 하나, q1이 p의 보로노이 셀 안에 있을 때)
(11) p<-교점을 공유하는 보로노이 셀의 중심
  S에 p 삽입.
(12) else
(13) q1<-q2
  q2<-CH(Q)에서 q1의 시계 방향 다음 꼭지점
  if(q1==q) break (CH(Q)를 한 바퀴 다 돈 경우)
(14) end
(15) end
(16) P의 delaunay graph를 따라가며 S에 의해 둘러싸인 그래프 내부의 점을 S에 삽입
return S
(CH(Q)의 꼭지점과 보로노이 셀이 만나는 점은 교점에서 제외)

위의 알고리즘은 우선 Q의 convex hull 계산에 O(|Q|log|Q|), P의 보로노이 다이어그램 계산에 O(|P|log|P|) 시간이 걸린다. (5)는 보로노이 다이어그램의 point location[1] 방법으로 O(log|P|)에 구할 수 있다. (6)~(15)는 해당 범위의 보로노이 셀과 CH(Q)의 선분의 교점 계산 부분이다. 보로노이 셀들은 각각 convex hull 이므로 이진검색을 이용하면 교점을 계산할 수 있다. CH(Q)의 선분과 만나는 모든 보로노이 셀들은 교점이 계산되어 지고, (12) 부분에 해당하는 보로노이 셀에 완전히 포함된 CH(Q)의 선분들도 보로노이 셀과 교점을 계산하게 된다. 따라서 시간 복잡도는 O((|S|+|CH(Q)|)log|P|)가 된다. (16)은 delaunay graph 상에서 이미 구해진 S에 의해 둘러싸인 점들만 구하면 되므로 O(|S|)에 구할 수 있다. 따라서 P의 보로노이 다이어그램과, Q의 convex hull이 주어져 있다면 전체 시간 복잡도는 O((|S|+|CH(Q)|)log|P|) 이 된다.

5. 결 론

Spatial 한 속성을 가지는 자료의 선호도 분석 방법으로서 Spatial skyline은 의미가 있다. 이 논문에서는 기존의 Spatial skyline 추출 방법의 문제점을 기술하고 개선된 알고리즘을 제시하였다. 기존에 O(|P|(|S||CH(Q)|+log|P|)) 시간 복잡도를 가지지만 모든 Spatial skyline points를 찾지 못하는 알고리즘 대신 O(|P|(|S|log|CH(Q)|+log|P|)) 만에 모든 Spatial skyline points를 찾을 수 있게 되었다. 또한 모든 Spatial skyline points를 찾지는 않지만 선호도가 높을 수 있는 부분 집합을 구할 수 있는 방법을 제시하였다. 사용자가 모든 Spatial skyline points가 아닌 부분 집합을 구하고 싶을 때 전체를 구할 때 보다 더 낮은 시간 복잡도인 O((|S|+|CH(Q)|)log|P|)에 구할 수 있다. 이 부분 집합은 부분 집합에 포함되지 않은 점보다도 대체로 Q와 밀집한 위치에 위치하기 때문에 선호도에서 더 좋은 평가를 받을 가능성이 높다. 이와 같이 기존의 방법보다 더 빠른 Spatial skyline 추출 방법을 제시하였으며, 이후 좀 더 개선된 Spatial skyline 추출 방법을 연구해 볼 예정이다.

6. 참고 문헌

[1] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. Computational Geometry : Algorithms and Applications. Springer Verlag, 2nd edition, 2000

[2] S. Börzsönyi, D. Kossmann, and K. Stocker. The Skyline Operator. In Proceedings of ICDE'01, pages 421.430, 2001.

[3] M. Sharifzadeh and C. Shahabi, "The Spatial Skyline Queries," in VLDB, 2006.