스카이라인 영역 결정을 위한 효율적인 가지치기 기법

김진호° 박영배
명지대학교 컴퓨터공학과
{solbong°, parkyb}@mju.ac.kr

Efficient Pruning Method for Skyline Region Decision

Jin-Ho Kim° Young-Bae Park
Department of Computer Engineering, University of Myongji

요 약

4단계 스카이라인 영역 결정 기법[2]은 영역 결정 시간이 객체의 개수에 비례해서 현저히 증가하기 때문에 다수의 객체를 포함하는 도메인들에 적용하기 어렵다. 이러한 문제점은 스카이라인 영역이 지배 객체 집합의 부분 집합으로 이루어지는 특성을 고려하지 않았기 때문에 발생한다.

이 논문에서는 스카이라인 영역 결정에 불필요한 객체들을 제거할 수 있는 거리 기반 가지치기 기법과 영역 결정 선분의 범위 축소 기법을 제안한다. 제안한 기법들을 R*-트리와 INN(Incremental Nearest Neighbor) 알고리즘에 적용함으로써 점진적으로 스카이라인 영역을 결정할 수 있으며 영역 결정 시간을 현저하게 감소시킬 수 있다. 제안한 기법의 성능 향상을 증명하기 위해 4단계 영역 결정 기법과의 비교 실험을 수행한다.

1. 서 론

스카이라인 질의(skyline query)는 전체 객체 집합에서 대상 객체의 여러 속성을 다른 객체가 지배하지 않는 관심 있는 객체 집합을 검색한다[1]. 스카이라인 질의는 대상 객체의 다중 속성을 고려해야 하는 여러 응용에서 중요한 연산이다. 이동 객체의 위치를 파악하여 다양한 정보를 제공하는 위치 기반 서비스(Location Based Service)의 응용에서는 이동 객체의 위치 변경에 따른 연속적인 스카이라인 질의 처리 기법이 필요하게 된다.

연속적인 스카이라인 질의(continuous skyline queries)는 대상 객체의 다중 속성과 이동 객체의 동적 속성을 모두 고려해야 한다. 예를 들어, 이동 사용자는 "현재 위치에서 가깝고 숙박료가 싸며, 해변과의 거리가 가까운 호텔을 검색하라"는 질의를 할 수 있다. 이 경우, 질의 발생 시점에 호텔과의 3가지 속성을 고려하여 결과를 검색해야 한다. 그러나 이전의 질의 결과는 이동 사용자가 위치를 변경하면 유효하지 않으므로 연속적인 질의가 발생하게 된다.

단일 동적 속성인 대상 객체와의 거리만을 고려한 이동 객체의 질의 처리 기법[3-8]들과 다중 정적 속성들을 기반으로 한 스카이라인 질의 처리 기법[1][9-11]들은 연속적인 스카이라인 질의를 처리할 수 없다.

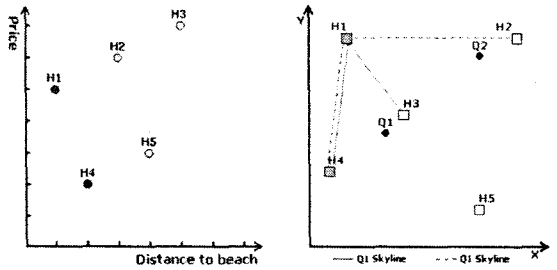
[2]에서는 이동 객체에 대한 연속적인 스카이라인 질의를 처리하기 위해 스카이라인 영역(Skyline Region)의 개념과 4단계 영역 결정 기법을 제안한다. 스카이라인 영역이란 대상 객체가 정적 속성에 대한 지배 객체보다 이동 객체에 가까운 영역을 의미한다. 스카이라인 영역은 위치에 기반하기 때문에 이동 객체의 속도와 방향과는 무관하며, 효율적으로 이동 객체에 대한 스카이라인 질의를 처리할 수 있다. 그러나 이 연구에서 제안한 4단계 스카이라인 영역 결정 기법은 영역 결정 시간이 객체의 개수에 비례해서 현저히 증가하기 때문에 다수의 객체를 포함하는 응용 도메인들에는 적용하기 어렵다. 이러한 문제점은 스카이라인 영역이 지배 객체 집합의 부분 집합으로 이루어지는 특성을 고려하지 않았기 때문에 발생한다.

이 논문에서는 영역 결정에 불필요한 객체들을 제거할 수 있는 거리 기반 가지치기 기법과 영역 결정 선분의 범위 축소 기법을 제안한다. 제안한 기법들을 R^*-트리와 INN(Incremental Nearest Neighbor) 알고리즘[13]에 적용함으로써 점진적으로 스카이라인 영역을 결정할 수 있으며 영역 결정 시간을 현저하게 감소시킬 수 있다. 제안한 기법의 성능 향상을 증명하기 위해 4단계 영역 결정기법과의 비교 실험을 수행한다.

2. 관련 연구

2.1 이동 객체에 대한 스카이라인 질의[2]

이동 객체에 대한 스카이라인 질의 결과는 정적 속성에 대해서는 고정이지만 동적 속성에 대해서는 질의 위치에 따라 가변적이다. 즉 질의 결과 집합은 대상 객체의 정적 속성에 대한 스카이라인 객체들과 정적 속성에 대한 지배 객체보다 이동 객체와 가까운 객체들로 구성된다. 예를 들어 호텔의 정적 속성 관계가 그림 1-(a)와 같을 경우 "현재 위치(Q_1)에서 가깝고 숙박료가 싸고, 해변과의 거리가 가까운 호텔을 검색하라."는 질의의 결과는 그림 1-(b)와 같이 정적 속성(숙박료, 해변과의 거리)에 대한 스카이라인 H_1과 H_4를 질의의 위치에 관계없이 포함하고, Q_1에서 가장 가까우며 정적 속성에 대한 지배 객체들 이동 객체와 가까운 H_3을 스카이라인 결과에 포함한다.



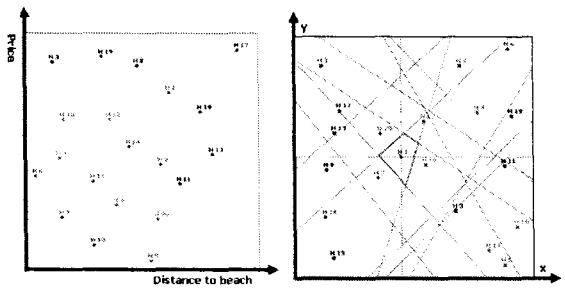
(a) 정적 속성 관계 (b) 질의 위치

그림 1 이동 객체에 대한 스카이라인 질의

스카이라인 영역 SR_i이란 객체 p_i가 정적 속성에 대한 지배 객체 p_j들보다 이동 객체에 가까운 영역으로 p_i가 질의 결과에 포함될 수 있는 영역이다.

SR_i는 그림 2와 같이 객체 p_i의 위치 좌표와 정적 속성에 대한 지배 객체 p_j들의 위치 좌표가 생성하는 수직이등분선들이 형성하는 영역으로 계산할 수 있다.

이동 객체가 위치를 변경할 때마다 대상 객체들과의 거리를 계산하여 스카이라인을 계산하는 것은 비효율적이다. 그러므로 정적 속성의 지배 관계에 따라 스카이라인 영역과 영역 간의 겹침 관계를 미리 계산하면, 이동 객체의 위치 좌표가 스카이라인 영역에 포함되는지 판별함으로써 질의 결과를 산출할 수 있다. 또한 스카이라인 영역을 이전 결과의 유효성 판단 기준으로 사용할 수 있으므로 연속적인 질의 처리가 가능하다.



(a) 호텔의 정적 속성관계 (b) H_1의 스카이라인 영역

그림 2 스카이라인 영역

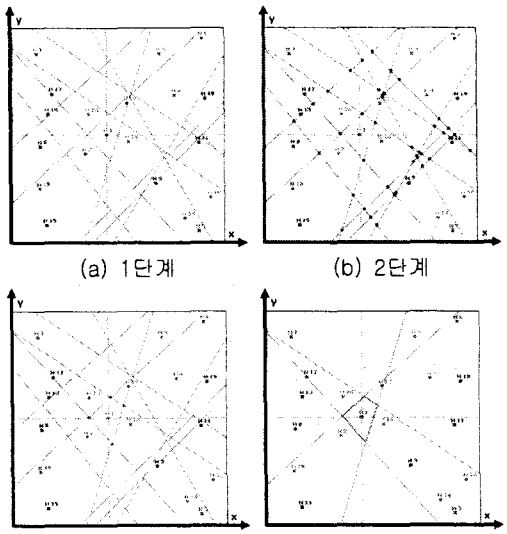
그림 2-(b)와 같이 모든 지배 객체에 대한 수직이등분선 중 일부만이 스카이라인 영역을 형성한다. 이 경우, 불필요한 선분을 제거하여 스카이라인 영역을 형성하는 최소의 선분을 결정함으로써 질의 위치와 영역과의 포함 관계 연산에 대한 불필요한 비교 횟수를 줄이고 연속 질의에 의한 정확한 유효 영역을 계산할 수 있다. 스카이라인 영역을 형성하는 최소의 선분을 결정하기 위한 4단계는 다음과 같다.

1단계: 정적 속성에 대하여 p_i를 지배하는 객체들과의 수직이등분선 및 부등식 영역 계산

2단계: SR_i를 형성하는 수직이등분선들의 모든 교차점 계산

3단계: 연립부등식을 만족하는 교차점 계산(영역의 꼭지점)

4단계: 꼭지점 2개를 지나지 않는 선분 제거



(a) 1단계 (b) 2단계

(c) 3단계 (d) 4단계

그림 4 스카이라인 영역 결정 기법(4단계)

2.2. INN (Incremental Nearest Neighbor) 알고리즘[13]

INN (Incremental Nearest Neighbor) 알고리즘은 R-트리의 최적 우선 순회(Best First Traversal) 기법으로 각 MBR 및 객체와의 최소거리(MinDist)를 기준으로 하는 우선 순위 큐(priority queue)를 이용하여 노드를 중복 방문하지 않고, 점진적으로 근접 객체를 검색하는 방법을 제안한다.

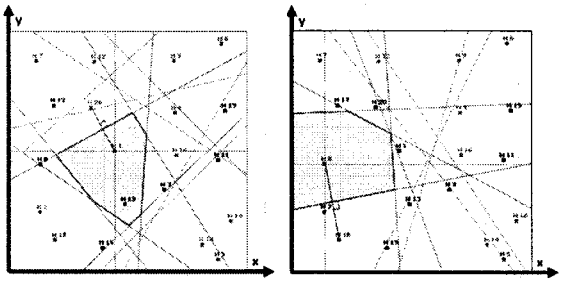
3. 스카이라인 영역 결정을 위한 효율적인 가지치기 기법

3.1. 거리 기반 가지치기(Distance Based Pruning) 기법

4단계 스카이라인 영역 결정 기법에서 연산에 소비하는 시간은 지배 객체의 개수에 비례한다. 그러나 그림 4-(b)와 같이 모든 지배 객체들이 스카이라인 영역을 형성하는데 영향을 미치지 않는다. 그러므로 연산에 소비하는 시간을 감소하기 위해서는 각 단계마다 모든 지배 객체를 연산의 대상으로 하는 것보다 스카이라인 영역을 형성하는데 불필요한 객체를 제거하여 연산을 최소화하는 기법이 필요하다. 그림 5는 스카이라인 영역과 최근접 지배 객체 간의 관계를 나타내며 스카이라인 영역

SR_i를 형성하는 영역 결정 선분 집합 S_{RDL}이라고 하면 [정리1]과 같이 정의할 수 있다.

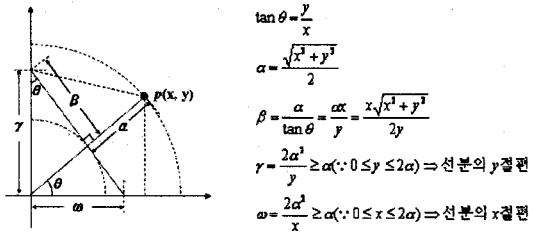
[정리1] 객체 p_i와 p_j의 최근접 지배 객체가 형성하는 수직이등분선은 반드시 S_{RDL}에 포함된다.



(a) H_1's $NN = H_{20}$ (b) H_8's $NN = H_{18}$

그림 5 스카이라인 영역과 최근접 지배 객체

S_{RDL}의 첫 번째 영역 결정 선분을 [정리 1]에 의해 결정한 후 반복적으로 다음 근접 지배 객체의 수직이등분선 VBL_i와 현재 S_{RDL}의 선분 RDL_k들의 교차 조건을 비교하여 교차하는 경우, S_{RDL}에 VBL_i를 추가해간다. 그림 6의 수직이등분선의 특성 값 중에 x절편 값 ω나 y절편 값 γ을 이용하여 교차 여부를 판단할 수 있다.



$$\tan\theta = \frac{y}{x}$$

$$\alpha = \frac{\sqrt{x^2+y^2}}{2}$$

$$\beta = \frac{\alpha}{\tan\theta} = \frac{\alpha x}{y} = \frac{x\sqrt{x^2+y^2}}{2y}$$

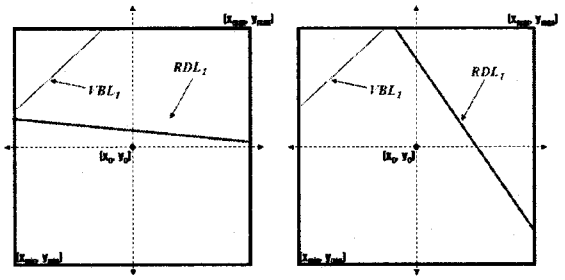
$$\gamma = \frac{2\alpha^2}{y} \geq \alpha (\because 0 \leq y \leq 2\alpha) \Rightarrow \text{선분의 } y\text{절편}$$

$$\omega = \frac{2\alpha^2}{x} \geq \alpha (\because 0 \leq x \leq 2\alpha) \Rightarrow \text{선분의 } x\text{절편}$$

(a) 수직이등분선 (b) 특성 값

그림 6 수직이등분선의 특성

객체 p_i의 위치 좌표를 원점으로 하면 RDL과 교차하지 않지만 영역 결정 선분인 VBL의 예외적인 경우가 발생한다. 그림 7은 교차하지 않는 영역 결정 선분의 예를 보여준다. 그림에서 경우 1, 2의 VBL_1은 RDL_1에 교차하지 않지만 경우 2의 VBL_1은 영역 결정 선분이므로 S_{RDL}은 VBL_1을 포함해야 한다. 예외적인 경우를 처리하기 위해서는 RDL의 형태에 따라 판별해야 한다.



(a) 경우 1: $VBL_1 \notin S_{RDL}$ (b) 경우 2: $VBL_1 \in S_{RDL}$

그림 7 RDL과 교차하지 않는 선분

그림 8과 같이 방향 벡터란 객체의 좌표 속성을 구별하기 위해 필요한 속성 값이다. 원점 좌표가 (x0, y0)이고, p(x0, y0))일 때 방향 벡터 m은 그림 8과 같이 객체의 위치에 의해 결정한다.

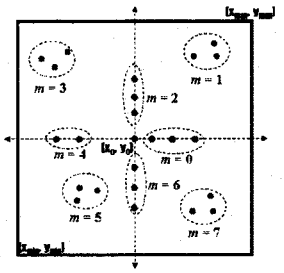


그림 8 방향 벡터

RDL에 교차하지 않는 영역 결정 선분들은 방향 벡터 m과 전역 변수 SVm 을 이용하여 판별할 수 있다. SVm의 초기 값은 모두 $false$로 설정하고 S_{RDL}에 수직이등분선을 추가할 때 SVm 의 상태 값은 변경된다. 교차하지 않는 영역 결정 선분은 m과 SVm 값으로 [정리 2]와 같이 정의할 수 있다.

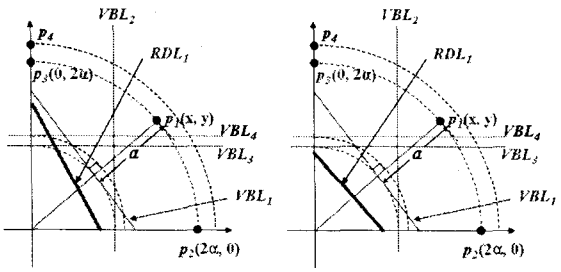
[정리2] RDL과 교차하지 않는 영역 결정 선분: S_{RDL}은 SVm 이 $false$이면 방향 벡터 m의 VBL을 RDL과 교차하지 않더라도 포함한다. 이 경우 VBL은 영역 결정 선분이며 m의 첫 번째 VBL이다.

수직이등분선의 특성에 의해 그림 9와 같이 공간 좌표에서 원점으로부터 동일 거리 2α에 있는 점 p_1, p_2, p_3의 수직이등분선 VBL_1, VBL_2, VBL_3에 대한 ω는 p_2일 때 최소이고 γ는 p_3일 때 최소이므로 [정리 3]의 거리 기반 가지치기 조건을 만족한다.

[정리3] 거리 기반 가지치기 조건: 2α거리에 있는 지배 객체 p_j의 수직이등분선 VBL_j가 다음 조건을 만족하면 S_{RDL}은 p_j보다 먼 거리(≥2α)에 있는 지배 객체들의 수직이등분선을 포함하지 않는다.

$$\alpha_j = \alpha(p_j)$$
$$\omega_{\max} = \max(\omega(RDL_k)), \gamma_{\max} = \max(\gamma(RDL_k))$$

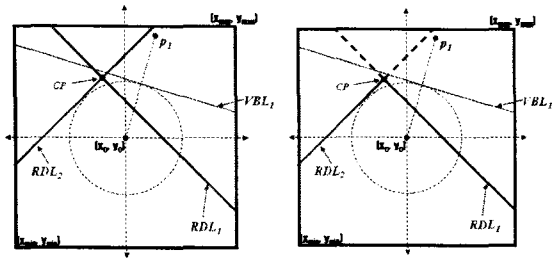
Pruning Condition :
$$\{\omega_{\max} < \omega(VBL_j)\} \wedge$$
$$\{\gamma_{\max} < \gamma(VBL_j)\} \wedge$$
$$\{\omega_{\max} < \alpha_j\} \wedge \{\gamma_{\max} < \alpha_j\}$$



(a) 경우 1: p_4 제거 불가 (b) 경우 2: p_4 제거 가능
그림 9 거리 기반 가지치기 기법

그림 9에서 경우1, 2 모두 영역 결정 선분 RDL_1은 VBL_1과 교차하지 않고 *SVm* 값도 *true*이므로 S_{RDL}은 RDL_1을 포함하지 않는다. 그러나 그림 9-(a)에서는 동일 거리에 있는 p_3의 VBL_3은 교차하므로 p_4를 제거할 수 없다. 그림 9-(b)에서는 VBL_2, VBL_3 모두 교차하지 않고 [정리 3]의 거리 기반 가지치기 조건을 만족하여 p_4를 제거할 수 있으므로 불필요한 연산을 줄일 수 있다.

3.2. 범위 축소(Extent Shrinking) 기법

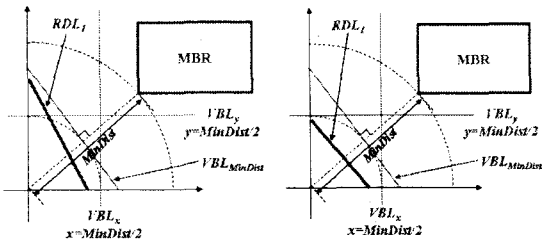


(a) 선분범위: 영역 경계선 (b) 선분범위: 교차점
그림 10 범위 축소 기법

영역 결정 선분 RDL_k의 범위를 전체 데이터 공간의 경계선과의 교점으로 정의하면 S_{RDL}은 그림 10-(a)와 같이 스카이라인 영역을 형성하는데 불필요한 선분 VBL_1을 포함하지만 10-(b)와 같이 영역 결정 선분의 범위를 다른 영역 결정 선분과의 교차점으로 축소하면 VBL_1을 포함하지 않는다. 선분의 범위를 영역 경계선으로 정의하면 최종 S_{RDL}을 결정하기 위해서는 연산의 마지막에 [2]에서 제안한 4단계 영역 결정 기법과 같이 스카이라인 영역의 교차점 및 꼭지점 등을 계산하여 불필요한 선분을 제거하는 과정이 필요하지만, 교차점으로 영역 결정 선분의 범위를 축소하면 그런 과정은 불필요하게 된다.

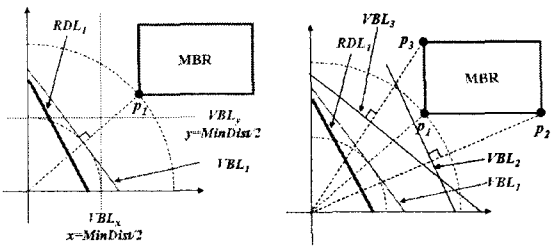
3.3. 스카이라인 영역 결정 기법

객체들의 공간 좌표를 R*-트리로 색인하고 INN알고리즘에서 제안한 거리(*MinDist*) 기반의 우선 순위 큐(priority queue)를 이용하면, 모든 지배 객체와의 거리를 계산하여 정렬하지 않아도 거리 순으로 객체를 검색할 수 있기 때문에 불필요한 객체를 점진적으로 제거할 수 있다. 그림 11은 거리 기반 가지치기 조건을 R*-트리 *MBR*에 적용한 예제이며, 그림 11-(b)와 같이 *MinDist*을 이용하여 *MBR*에 속한 객체를 정지할 수 있음을 보여준다.



(a) *MBR* 제거 가능 (b) *MBR* 제거 불가능
그림 11 *MBR*과 거리 기반 가지치기 기법

INN 알고리즘에 의해R*-트리를 *MinDist* 순으로 탐색할 때 방향 벡터 *m*에 대한 거리기반 가지치기 조건을 만족하면 동일 *m* 값을 가진 우선순위 큐의 엔트리를 제거하여 방향 벡터 *m*에 대해서는 더 이상진행하지 않는다. 그러므로 우선순위 큐의 각 엔트리는 방향 벡터 *m* 값을 유지하여야 한다. *MBR*의 방향 벡터 값은 4분면에 속한 MBR에 대해서만 결정되며 *x*나 *y*축에 걸쳐 있는 경우 *m*=-1이다.



(a) *MBR* 제거 불가능 (b) *MBR* 제거 가능
그림 12 *RDL*과 *MBR*의 교차

그림 12는 *MBR*의 3개의 꼭지점에 대해서 *RDL*과의 교차

여부를 결정하면 더욱 효율적으로 *MBR*을 제거할 수 있음을 보여주는 예제이다.

4. 실험 및 성능 평가

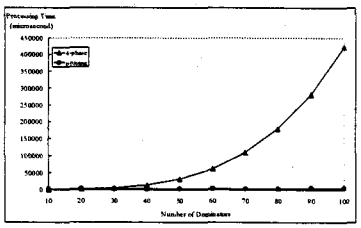
제안한 기법의 성능을 평가하기 위해 4단계 영역 결정 기법과 비교하여 실험을 수행하였다. 지배 객체 개수에 따른 단일 스카이라인 영역 결정과 대상 객체 개수에 따른 전체 스카이라인 영역 결정 성능에 대한 실험을 수행하였다. 실험 환경은 하이퍼-스레딩을 지원하는Pentium-IV 2.6GHz 프로세서와 512MB의 메인 메모리, 80GB의 하드 디스크를 가진 Windows XP 운영 체제의 PC에서 자바언어(JSDK 1.5)를 이용하여 구현하였다.

4.1. 지배 객체 개수에 따른 단일 스카이라인 영역 결정 성능 비교

지배 객체의 개수를 10에서 100까지 변화하면서 단일 스카이라인 영역 결정 시간을 μs 단위로 측정하였다. 그림 13은 지배객체 개수에 따른 단일 영역 결정 성능의 비교 결과를 나타낸 것이다. 4단계 기법은 지배 객체가 2배 증가했을 때 결정 시간이 약 5~14배 이상 증가하지만 제안한 기법은 단지 1~2배 정도만 증가하였고, 지배 객체가 90개 증가했을 때 4단계 기법의 결정 시간이 약 1,640배 증가한 반면 제안한 기법은 약 4배 정도만 증가하였다.

Number of Dominators	Processing Time (microsecond)		Improvement ratio
	4-phase	pruning	
10	257.59	1,205.06	-
20	1,344.61	2,512.22	-
30	4,639.76	1,544.06	66.7%
40	14,003.25	1,992.68	85.8%
50	31,007.90	2,641.84	91.5%
60	61,571.75	2,535.76	95.9%
70	108,758.24	2,716.87	97.5%
80	179,208.24	2,965.20	98.3%
90	281,639.35	3,937.38	98.6%
100	423,547.58	4,594.45	98.9%

(a) 성능 비교 데이터



(b) 성능 비교 그래프

그림 13 단일 영역 결정 성능 비교

그림 14는 지배객체 개수에 따른 단일 영역 결정 시간의 변화를 측정하기 위해 지배 객체의 개수를 100에서 1000까지 100씩 증가하면서 실험한 결과이며 지배 객체가 900개 증가했지만 단일 영역 결정 시간은 약 2배 정도만 증가했다. 이 실험에서는4단계 기법은 현저하게 결정 시간이 증가하기 때문에 제외하였다.

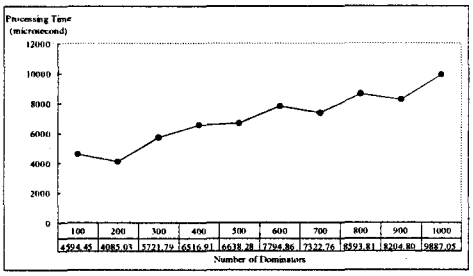


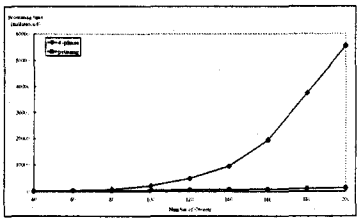
그림 14 지배 객체 개수에 따른 성능 변화

4.2. 대상 객체 개수에 따른 전체 스카이라인 영역 결정 성능 비교

대상 객체의 개수를 40에서 200까지 20씩 증가하면서 전체 스카이라인 영역 결정 시간을 ms 단위로 측정하였다. 그림 15는 대상 객체 개수에 따른 전체 스카이라인 영역 결정 성능 비교 결과를 나타낸 것이다. 4단계 기법은 대상 객체가 2배 증가하면 결정 시간이 약 20~27배 증가했지만 제안한 기법은 약 3배 정도만 증가하였고, 대상 객체가 160개 증가했을 때 4단계 기법은 결정 시간이 약 1,500배 증가를 보였지만 제안한 기법은 약 14배 정도만 증가하였다.

Number of Objects	Processing Time (millisecond)		Improvement ratio
	4-phase	pruning	
40	37	91	-
60	222	173	22.2%
80	751	277	63.0%
100	2,092	383	81.7%
120	4,854	540	88.9%
140	9,503	660	93.1%
160	19,476	797	95.9%
180	37,575	1,043	97.2%
200	55,596	1,291	97.7%

(a) 성능 비교 데이터



(b) 성능 비교 그래프

그림 15 전체 영역 결정 성능 비교

그림 16은 대상객체 개수에 따른 전체 영역 결정 시간의 변화를 측정하기 위해 대상 객체의 개수를 100에서 1000까지 100씩 증가하면서 실험한 결과이며 대상 객체가 900개 증가했을 때 전체 영역 결정 시간은 약 34배 정도 증가했다. 이 실험에서도 그림 14와 같은 이유로 4단계 결정 기

법은 제외하였다.

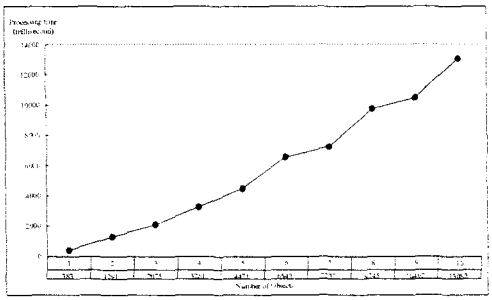


그림 16 대상 객체 개수에 따른 성능 변화

5. 결 론

4단계 스카이라인 영역 결정 기법은 스카이라인 영역이 지배 객체 집합의 부분 집합으로 이루어지는 특성을 고려하지 않았기 때문에 대상 객체의 개수에 비례하여 영역 결정 시간이 현저하게 증가하는 문제점이 있었다.

이 논문에서는 스카이라인 영역 결정에 불필요한 객체들을 제거할 수 있는 거리 기반 가지치기 기법과 영역 결정 선분의 범위 축소 기법을 제안하였다. 제안한 기법은 영역 결정 시간을 현저하게 감소시켜 4단계 영역 결정 기법보다 훨씬 뛰어난 성능을 보임을 실험을 통해 증명하였다.

제안한 기법은 4단계 영역 결정 기법과의 성능 비교 결과, 단일 스카이라인 영역 결정 시간은 지배 객체 개수에 따라 평균 91.7%, 전체 스카이라인 영역 결정 시간은 대상 객체 개수에 따라 평균 80% 감소시켰다. 대상 객체가 200개인 경우, 제안한 기법의 전체 스카이라인 영역 결정 성능은 4단계 기법에 비해 약 43배 차이를 보였으며, 500개의 이하의 대상 객체를 가진 경우 5초 이내에 전체 스카이라인 영역을 결정할 수 있어 온라인에도 적용할 수 있음을 보였다.

참고문헌

[1] Borzsonyi, S., Kossmann, D., Stocker, K. "The Skyline Operator" In ICDE, p.421-430, 2001.

[2] 나경석, 김진호, 안대혁, 박영배 "연속적인 스카이라인 질의를 위한 효율적인 영역 결정기법", 정보과학회 데이터베이스 연구회지, 제22권 2호, 2006

[3] Y. Theodoridis, J. R.O. Silva, and Mario A. Nascimento, On the Generation of Spatiotemporal Datasets, In Proceedings of the 6th Int'l Symposium on Large Spatial Database(SSD), 1999.

[4] Zheng B., Lee, D. "Semantic Caching in Location- Dependent Query Processing" SSTD, p.97-116, 2001.

[5] Song, Z., Roussopoulos, N. K-Nearest Neighbor Search for Moving Query Point. SSTD, 2001.

[6] Benetis, R., Jensen, C., Karciauskas, G., Saltenis, S. Nearest Neighbor and Reverse Nearest Neighbor Queries for Moving Objects. IDEAS, 2002.

[7] Tao, Y., Papadias, D., Shen, Q. Continuous Nearest Neighbor Search. VLDB, 2002.

[8] J. Zhang, M. Zhu, D. Papadias, Y. Tao, and D. L. Lee. Location-based spatial queries. In SIGMOD, p.443-454, 2003

[9] Tan, K., Eng, P. Ooi, B. "Efficient Progressive Skyline Computation." In VLDB, p.301-310, 2001.

[10] D. Kossmann, F. Ramsak, S. Rost, "Shooting Stars in the Sky: an Online Algorithm for Skyline Queries." In VLDB, p.275-286, 2002

[11] Papadias, D., Tao, Y., Fu, G., Seeger, B. "An Optimal and Progressive Algorithm for Skyline Queries." In SIGMOD, p.443-454, 2003.

[12] Yufei Tao, Xiaokui Xiao, Jian Pei, "SUBSKY: Efficient Computation of Skylines in Subspaces," In ICDE, p. 65, 2006.

[13] G. R. Hjaltson, H. Samet, "Distance Browsing in Spatial Databases", ACM Transaction on Database Systems, 24(2), 265-318, 1999.