DOI: 10.3745/KIPSTD.2010.17D.4.245

MANET에서 스카이라인 질의를 위한 효과적인 필터링 방법

박 미 라[†] · 김 민 기[††] · 민 준 기[†††]

요 약

본 연구에서는 MANET(Mobile Ad hoc NETwork) 환경에서 스카이라인 질의를 하기 위한 효과적인 필터링 방법을 제안한다. 기존의 MANET 환경에서의 스카이라인 질의 처리는 데이터가 균등하게 분포한다고 가정한다. 이러한 가정하에서 제한된 배터리 용량을 위한 에너지 소모 최소화에 중점을 두어 스카이라인 질의를 처리하는 방법을 연구한다. 그러나 실제 환경에서는 특정한 영역에 데이터가 편향되는 분포를 가진다.

배터리의 에너지 소비를 감소하기 위해서 본 논문에서는 데이터 분포를 고려한 새로운 필터링 방법을 제안한다. 그리고 기존의 필터링 방법과 본 논문에서 제안하는 필터링 방법을 비교 실험한다. 실험 결과는 본 논문에서 제안하는 방법이 기존의 방법보다 통신 오버헤드와 실행시간이 감소하는 것을 보여준다.

키워드: P2P, MANET, 스카이라인, 편향된 데이터, 데이터 필터링

An Effective Filtering Method for Skyline Queries in MANETs

Mi-Ra Park[†] · Min-Kee Kim[††] · Jun-Ki Min[†††]

ABSTRACT

In this paper, we propose an effective filtering method for skyline queries in mobile ad hoc networks (MANETs). Most existing researches assume that data is uniformly distributed. Under these assumptions, the previous works focus on optimizing the energy consumption due to the limited battery power. However, in practice, data distribution is skewed in a specific region.

In order to reduce the energy consumption, we propose a new filtering method considering the data distribution. We verify the performance of the proposed method through a comparative experiment with an existing method. The results of the experiment confirm that the proposed method reduces the communication overhead and execution time compared to an existing method.

Keywords: P2P, MANET, Skyline, Skewed Data, Data Filtering

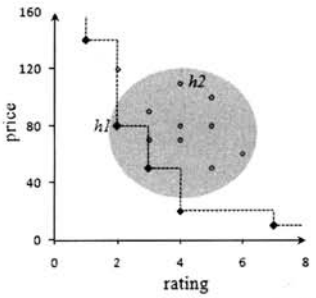
1. 서 론

P2P(peer-to-peer) 기술의 다양한 확장에 따라, MANET (mobile ad hoc networks) 환경의 분산된 데이터베이스에서 P2P 통신으로 데이터를 검색하는 문제에 대해 관심을 갖게 되었다. MANET은 유선 네트워크 망의 도움 없이 노드들 간에 서로 협력하여 다중-홉(multi-hop)으로 정보를 전달할 수 있도록 해주는 네트워크이다[2]. MANET은 이전에는 군용통신에서 사용할 목적으로 고려되었지만 최근에는 센서 네트워크나 홈 네트워크와 같은 일반적인 상용 망에서도 적용되고 있다[1].

모바일 장치는 MANET 환경에서 제한된 대역폭과 에너지 제약을 가진다. 데이터들은 모바일 장치에 분산 저장되어 있다고 가정한다. 따라서, 질의가 발생한 장치는 질의의 결과를 완성하기 위하여 주변 모바일 장치들로부터 질의의 결과에 포함될 가능성이 있는 데이터들을 가져와야 한다. 모바일 장치는 에너지 제약이 있기 때문에 통신 비용을 줄이기 위해서는 전송할 데이터의 양을 감소시켜야 한다.

스카이라인 질의는 다른 객체들에 의해 지배되지 않는 객체들 (또는 튜플)의 집합을 반환한다[3]. 객체 p가 객체 q와 모든 차원에서 속성값들이 같거나 p의 속성값이 q의 속성값보다 더 좋은 값을 가지고, 최소한 한 개의 차원에서 더 좋은 값을 가지면 객체 p는 객체 q를 지배한다(dominate)고 말한다. (그림 1)은 숙박 요금이 저렴하고 서비스의 질이 높은 호텔들의 스카이라인을 나타낸다. $h1$은 $h2$보다 숙박 요금이 더 저렴하고, $h1$의 서비스의 질은 $h2$보다 더 좋다. 그

※ 이 논문은 2010년 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No.2010-0016165).
† 준 회 원 : 한국기술교육대학교 정보미디어공학과 석사과정
†† 준 회 원 : 한국기술교육대학교 컴퓨터공학과 박사과정
††† 종신회원 : 한국기술교육대학교 컴퓨터공학부 조교수
논문접수 : 2010년 2월 4일
수 정 일 : 1차 2010년 3월 22일
심사완료 : 2010년 3월 22일



(그림 1) 편향된 데이터의 예

러므로 $h1$은 $h2$를 지배한다. 그리고 $h1$을 지배하는 속성값을 가지는 호텔은 없으므로, $h1$은 스카이라인(skyline) 객체이다.

스카이라인에 관한 이전의 연구들[3-6, 9]은 대부분 데이터베이스 시스템에서 값들이 균등하게 분포하는 데이터를 가정한다. 그러나, 실제 데이터베이스 시스템에서 데이터가 항상 균등하게 분포하는 것은 아니다. 본 논문에서는 MANET 환경에서 편향된 분포를 따르는 스카이라인 질의를 효과적으로 수행하기 위한 필터링 방법을 제안한다.

(그림 1)은 호텔들의 이용료와 서비스의 질에 따라 그래프로 나타낸 것이다. 그림에서 rating은 서비스의 질을 나타내며 작은 숫자일수록 서비스의 질이 높다. 사용자는 숙박요금이 저렴하고 서비스의 질이 높은 호텔을 선호한다. 진하게 표시된 점은 스카이라인 객체를 나타낸다. 일반적인 호텔의 데이터들은 이용료가 비쌀수록 서비스의 질도 좋다. 그리고 가격이 비싸고 서비스의 질도 매우 좋은 호텔보다는 적당한 가격에 적당한 서비스의 질을 가진 호텔의 개체수가 더 많다.

본 연구에서는 모바일 장치에서 스카이라인 질의 처리를 수행하기 위해 다음을 가정한다.

1) 데이터는 균등하게 분포하지 않는다. (그림 1)에서 호텔 객체들은 회색으로 표현된 영역에 많이 나타난다.
2) 장치 M_1 에서 빈번하게 나타나는 데이터는 장치 M_2 에서도 빈번하게 나타나는 경향이 있다. 즉, 편향된 데이터의 편향되는 영역은 각 모바일 장치에서 비슷하게 나타난다.
3) 스카이라인 질의를 하는 모바일 사용자들은 오직 제한된 지리적 영역에 속해있는 장치들의 데이터에만 관심이 있다[6]. 즉, 주변의 모바일 장치에서 가져오는 데이터들의 결과들을 합치면 원하는 결과를 얻는다고 가정한다.

본 연구에서는 MANET 환경에서 통신 오버헤드를 감소시키기 위해 최빈값 (Mode value)을 사용하여 필터링 튜플을 구하는 방법을 제안한다. 필터링 튜플이란 최종 스카이라인 결과에 포함될 가능성이 없는 튜플들을 전송할 데이터에서 제외시키기 위하여 사용되는 튜플을 말한다.

최빈값을 찾기 위하여 각 차원의 값들의 빈도 수를 측정하고 최빈값을 기반으로 필터링 튜플을 선택한다. 또한, 기존의 필터링 기법과 비교하여 본 연구에서 제안하는 필터링 기법의 성능을 보여준다.

본 논문의 나머지 부분은 다음과 같이 구성된다. 2장에서는 스카이라인을 계산하는 연구들과 모바일 장치에서 스카이라인 질의를 하는 기존의 연구 기법들에 대하여 설명한다. 3장에서는 최빈값을 사용하여 필터링 튜플을 선택하는 방법을 소개한다. 4장에서는 MANET 환경에 본 연구에서 제안하는 기법의 적용 방법을 설명한다. 5장에서 실험 결과를 보여주고, 6장에서 결론을 맺는다.

2. 관련 연구

본 연구는 스카이라인 질의 처리 연구와 MANET 환경에서의 모바일 장치들 간에 스카이라인 질의에 관한 연구에 동기 부여 받았다.

2.1 스카이라인 질의

Borzsonyi 등은 최초로 스카이라인 계산에 관한 연구를 소개했다[3]. [3]에서 Block Nested Loop(BNL) 알고리즘과 Divide and Conquer(D&C)알고리즘을 통한 스카이라인 질의 처리를 제안한다. BNL 알고리즘은 순차적으로 전체 데이터를 스캔하고, 메모리에 저장된 후보 데이터와 비교한다. D&C 알고리즘은 데이터의 집합을 많은 영역으로 나누고, 각 영역에서 스카이라인을 계산한다. 그 다음, 각 영역의 스카이라인으로부터 최종 스카이라인을 결정한다. 후에, Chomicki 등은 BNL을 변형시킨 Sort Filter Skyline(SFS) 알고리즘을 제안한다[4]. SFS 알고리즘은 연산을 통해 모든 속성을 하나의 단일 값으로 만들고 정렬을 통해 비교 대상의 수를 줄인다.

중앙 서버에 스카이라인 질의를 처리하는 연구에서 벗어나서 Balke 등은 [5]에서 웹 정보 시스템을 위해서 분산된 데이터베이스에서 스카이라인 질의 처리를 다뤘다. Sun 등은 분산된 환경의 다차원 테이블에서 스카이라인 질의를 처리하는 문제를 연구했다[8]. Cui 등은 reduced minimum bounding box($rMBR_i$)를 기반으로 하는 PadSkyline (Parallel Distributed Skyline query processing)를 제안했다[9]. Yoon 등은 무선 센서 네트워크에서 분산된 공간 스카이라인을 계산하기 위해 Distributed Spatial Skyline(DSS) 알고리즘을 제안했다.

최근에 P2P와 애드혹 네트워크에서 스카이라인 질의를 처리하는 문제는 많은 관심을 끌고 있다. Wang 등은 peer-to-peer 네트워크에서 스카이라인 질의 처리를 다룬다[7]. Antony 등은 애드혹 집합 위에서 스카이라인 질의를 제안했다[11]. 관련된 논문들과 같이 본 논문에서는 애드혹과 P2P 기술이 결합된 환경을 고려한다.

2.2 모바일 장치에서 스카이라인 질의

Huang [6] 등은 MANET 환경에서 모바일 장치들 사이에서 스카이라인 질의 처리를 위한 기법을 제안하였다. 모

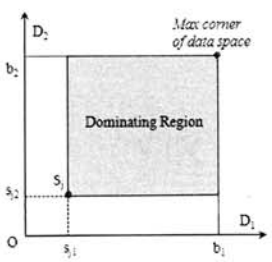
바일 장치들 사이에서 데이터를 전달할 때 최종 결과에 포함될 가능성이 있는 데이터들만 전달하기 위한 필터링 방법과 필터링을 위해 필요한 필터링 튜플을 선정하는 방법을 제시하였다.

각 장치의 모든 로컬 스카이라인 객체들이 최종 스카이라인 집합에 포함되는 것은 아니다. 그러므로 통신 오버헤드를 감소시키기 위해 각 장치의 모든 로컬 스카이라인 객체들을 전송하기 보다는 최종 스카이라인에 포함될 가능성이 있는 객체들만을 질의가 발생한 장치로 전달한다. 이를 위하여 필터링 튜플이라고 부르는 특정 튜플을 기반으로 불필요한 데이터의 전송을 막는다.

사용자가 모바일 장치 M_{org}(query originator)를 통해 질의를 하면, M_{org}는 우선 로컬 데이터들로부터 스카이라인 객체들을 구한다. 그리고 M_{org}는 로컬 스카이라인 객체들 중에 다른 객체들을 가장 많이 지배할 가능성이 있는 튜플을 선택(여기서 선택된 필터링 튜플을 T_{org}라고 하겠다)하여 주변의 장치 M_1에게 보낸다. 질의를 받은 모바일 장치 M_1은 로컬 스카이라인 객체들을 구한다. 그리고 나서, M_{org}로부터 받은 필터링 튜플 T_{org}에 의해 지배되는 M_1의 로컬 스타이라인 객체들은 최종 스카이라인 집합에 포함될 가능성이 없으므로 M_{org}로 전송되는 결과 집합에서 제외시킨다. 이 과정을 통해서 로컬 스카이라인 객체들을 전송할 때 드는 통신 오버헤드를 감소시킨다.

모바일 장치 M_1은 또한 남아있는 로컬 스카이라인 객체들과 T_{org} 사이에서 필터링 튜플 선택 방법에 따라서 필터링 튜플 T_1을 선택하고, 주변의 노드로 T_1을 질의와 함께 보낸다. 이러한 단계를 거쳐서 MANET 환경에서 스카이라인 질의가 수행된다. 필터링 튜플에 따라서 전송하지 않아도 되는 로컬 스카이라인 객체를 제거하는 양이 달라지기 때문에 필터링 튜플의 효율적인 선택은 통신 오버헤드를 감소시키기 위해 중요하다.

[6]에서는 VDR(Volume of Dominating)이 필터링 튜플을 선택하는 기준으로 사용된다. (그림 2)는 VDR의 개념을 설명한다. 각 차원 D_k는 가상 전역 관계 R에서 $[0, b_k]$ 범위의 값들을 가진다. 튜플 S_j의 지배 영역의 크기는 $VDR_j = \Pi_{k=1}^{l}(b_k - s_{jk})$이다. (그림 2)를 보면, 회색 박스로 표현된 지배영역에 있는 객체들은 튜플 S_j에 의해 지배된다. 그러므로 데이터가 균등 분포한다는 가정하에서 S_j의 지배영역의 크기가 클수록 S_j에 의해 지배되는 객체의 수는 증가한다. 그러므로 스카이라인 객체들 중에 최대 VDR값을 가지는 튜플을 필터링 튜플로 선택한다. 필터링 튜플은 질의가 수행되는 동안에 동적으로 갱신된다. 모바일 장치에서 로컬 스카이라인 질의를 수행한 후에, 이전의 필터링 튜플보다 더 큰 최대 VDR 값을 가지는 튜플이 있으면 필터링 튜플을 변경하여 다른 모바일 장치로 전송한다.



(그림 2) 지배 영역의 예

그러나 데이터가 특정 영역에 편향되어 있는 분포에서는 VDR이 효과적이지 않다. 그래서 우리는 편향된 분포를 고려하여 필터링 튜플을 선정하는 방법을 제안한다.

3. 최빈값을 이용한 필터링 튜플

이 절에서는 각 단말 장치에서 필터링에 의한 스카이라인 질의 처리를 소개한다.

3.1 문제 정의

n개의 모바일 장치 $M = \{M_1, M_2, \ldots M_n\}$이 있다고 가정하자. 각 장치 M_i는 릴레이션 R_i를 가진다.

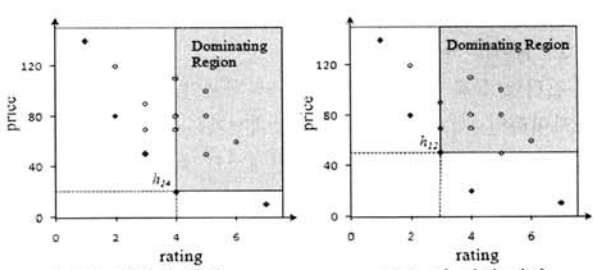
<표 1>에서, 모바일 장치 M_1은 호텔 릴레이션 R_1을 가진다. 사용자는 싸고 좋은 호텔의 정보를 원한다. 각 호텔 객체는 숙박 요금과 서비스 질을 기반으로 한 등급을 속성으로 갖는다.

M_1의 스카이라인 객체는 $\{h_1, h_6, h_{12}, h_{14}, h_{15}\}$이다. 가격과 등급이 가질 수 있는 최대값을 200과 10이라고 가정하자. M_1의 로컬 스카이라인들로부터 필터링 튜플을 선택할 필요가 있다. 2.2에서 설명한 VDR의 정의를 사용하면, VDR_1 = (200 - 140) * (10 - 1) = 540, VDR_6 = (200 - 80) * (10 - 2) = 960, VDR_{12} = (200 - 50) * (10 - 3) = 1050, VDR_{14} = (200 - 20) * (10 - 4) = 1080, VDR_{15} = (200 - 10) * (10 - 7) = 570의 값을 가진다. h_{14}가 가장 큰 VDR 값을 가지므로 h_{14}가 필터링 튜플로 선정된다.

(그림 3)은 h_{12}와 h_{14}의 지배영역을 나타낸다. h_{14}의 지배

〈표 1〉 릴레이션 R_1

hotel	price	rating	hotel	price	rating
h_1	140	1	h_9	70	3
h_2	120	2	h_{10}	70	4
h_3	110	4	h_{11}	60	6
h_4	110	5	h_{12}	50	3
h_5	90	3	h_{13}	50	5
h_6	80	2	h_{14}	20	4
h_7	80	4	h_{15}	10	7
h_8	80	5			



(a) h_{14}의 지배 영역 (b) h_{12}의 지배 영역

(그림 3) 지배 영역의 모순

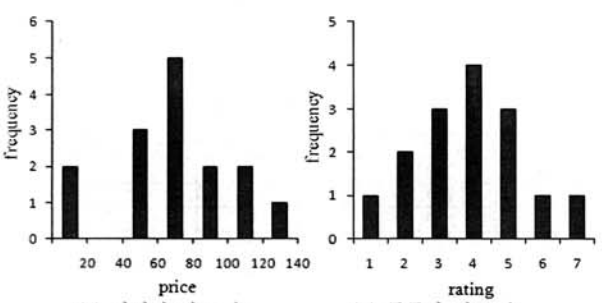
영역은 h_{12}의 지배 영역보다 크다. 그러나 h_{14}는 7개의 객체를 지배하는 반면에 h_{12}는 9개의 객체를 지배한다. 이처럼 *VDR*이 정확히 지배되는 객체들의 수를 반영하는 것은 아니다. 이런 모순은 데이터의 편향되는 비율이 높을수록 더 심하게 나타난다.

3.2 데이터의 최빈값

편향되게 분포되어 있는 데이터를 다루는 많은 연구들이 있다. MANET 환경에서 각 노드의 전력은 제한되어 있기 때문에 복잡한 기법은 사용할 수 없다. 편향된 분포를 한 데이터 구조를 가질 때, 필터링 튜플의 효율성을 향상시키기 위해 최빈값을 사용한다.

본 연구에서 필터링 기법의 기본 아이디어는 모든 차원에서 최빈값을 커버하는 영역을 지배하는 필터링 튜플을 선택하는 것이다. 각 차원에서 데이터들은 최빈값 주변에 많이 나타나기 때문에 최빈값을 커버하는 튜플이 결국 많은 객체들을 지배하게 된다. 값의 빈도를 얻을 때 추가적인 비용이 들지 않도록, 각 장치가 스카이라인 질의를 처리할 때 모든 차원에서 값의 빈도를 함께 계산한다. 그리고 나서 값들의 빈도 수를 이용하여 각 차원에서 최빈값을 얻는다. 값의 빈도를 나타내기 위해서 본 연구에서는 equi-width 히스토그램을 사용한다.

(그림 4)는 <표 1>에 있는 데이터의 값의 빈도 수를 보여준다. 이 예에서 각 차원의 값들의 10개의 구간으로 나누어 계산된다. 그림에서는 편의상 7개의 구간만 나타내었다. (그림 4)(a)에서 가격의 최빈값은 60-80의 구간이고 그것의 평균은 70이다. (그림 4)(b)에서 등급의 최빈값은 4이다. 각 차원의 최빈값은 *MP*(Mode Point)로 표기한다. 이 예에서 *MP*는 (70, 4) 이다.



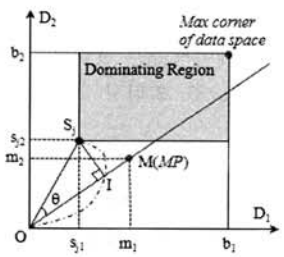
(a) 가격의 빈도 수 (b) 등급의 빈도 수

(그림 4) 값의 빈도

3.3 필터링 튜플 선정 방법

필터링 튜플은 최종 스카이라인 객체가 될 가능성이 없는 스카이라인 객체들을 전송 대상에서제외시킨다. 이에 따라서 전송 비용이 감소된다. 이 절에서는 *MP*를 사용하여 필터링 튜플을 선택하는 방법을 설명한다. (그림 5)는 본 논문에서 제안한 방법에 대한 기본 개념을 2-차원 공간상에서 설명한다.

우선 *O*와 *MP*를 지나는 선을 긋는다. (그림 5)에서 *OM*



(그림 5) DPL 계산을 위한 예

직선에 가장 가까운 스카이라인 객체는 최빈값에 가장 근접한 값이므로 가장 많은 객체를 지배할 가능성이 있다. 그러므로 각 스카이라인 객체들과 직선 사이의 거리를 계산하여 최소가 되는 값을 필터링 튜플로 선택한다.

벡터의 내적과 피타고라스 정리를 사용하여 점과 직선 사이의 거리를 계산한다. 차원의 수를 *d*라 하고 D_k 차원 (1 ≤ k ≤ d) 의 값의 범위를 [0, b_k] 라고 가정하자. *M*은 *MP*(Mode Point)를 나타낸다. S_j (= $<s_{j1},\cdots,s_{jd}>$) 는 스카이라인 객체이다. 그리고 각 차원들의 값의 범위에 따른 영역의 크기를 정규화(normalization)하기 위해서 각 차원 D_k에 있는 값을 각 차원의 최대값인 b_k로 각각 나누어 사용하였다.

먼저 점 *O*와 점 S_j사이에 거리 $|\overline{OS_j}|$를 계산한다.

$$|\overline{OS_j}| = \sqrt{\sum_{k=1}^{d}\left(\frac{s_{jk}}{b_k}\right)^2}$$

그리고 나서 $\overrightarrow{OS_j}$와 $\overrightarrow{OM}$의 벡터의 내적을 이용하여 cos θ를 계산한다.

$$\cos\theta = \frac{\sum_{k=1}^{d}\frac{m_k}{b_k}\cdot\frac{s_{jk}}{b_k}}{\sqrt{\sum_{k=1}^{d}\left(\frac{m_k}{b_k}\right)^2\cdot\sum_{k=1}^{d}\left(\frac{s_{jk}}{b_k}\right)^2}}$$

그 다음 $|\overline{OS_j}|$와 cosθ를 이용하여 $|\overline{OI}|$를 구한다.

그러면 (그림 5)에 표시된 삼각형의 양변인 $|\overline{OS_j}|$와 $|\overline{OI}|$를 알기 때문에, 피타고라스 정리를 사용하여 나머지 한 변인 $|\overline{IS_j}|$를 계산할 수 있다. 점 S_j부터 $\overrightarrow{OM}$ 까지 거리를 DPL_j(Distance between the Points and Line)로 나타내고 다음과 같이 공식화한다.

$$\begin{aligned} DPL_j^2 &= |\overline{IS_j}| = |\overline{OS_j}|^2 - |\overline{OI}|^2 \\ &= |\overline{OS_j}|^2 - |\overline{OS_j}|^2\cdot(\cos\theta)^2 \qquad (1) \\ &= \sum_{k=1}^{d}\left(\frac{s_{jk}}{b_k}\right)^2 - \frac{\left(\sum_{k=1}^{d}\frac{m_k\cdot s_{jk}}{b_k^2}\right)^2}{\sum_{k=1}^{d}\left(\frac{m_k}{b_k}\right)^2} \end{aligned}$$

여기서 *d*는 차원의 수를 나타낸다.

따라서, 모바일 장치에서 각 스카이라인 객체의 DPL을 계산한다. 그리고 DPL의 최소값을 가지는 튜플을 필터링 튜플로 선택한다. <표 1>의 예에서는 DPL의 최소값을 가지는 h_{12}가 필터링 튜플로 선택된다.

4. MANET 환경에의 적용

4.1 통신 절차

모바일 장치는 메모리의 제약 때문에 사용자가 검색하기를 원하는 정보의 일부분을 가진다. 그래서 MANET 환경에서는 질의를 보내는 모바일 장치가 보다 정확한 정보를 얻기 위하여 다중 홉 라우팅을 기반으로 주변의 모바일 장치들과 통신한다고 가정한다.

사용자가 스카이라인 질의를 요청하면, 먼저 질의를 수행하는 장치인 M_{org}는 로컬 스카이라인 질의를 수행한다. 이때 M_{org}는 각 차원에서 값의 빈도수를 히스토그램을 이용하여 계산한다. 그리고 나서 M_{org}는 빈도 수를 사용하여 최빈값을 구한다. 각 차원의 최빈값들을 조합하여 MP를 생성한다. M_{org}는 MP를 이용하여 DPL의 최소값을 가지는 로컬 스카이라인 객체를 필터링 튜플로 선택한다. M_{org}는 주변의 장치에게 질의를 보낼 때 필터링 튜플과 값의 빈도 수도 함께 전송한다. 후에 주변의 장치들로부터 최종 스카이라인이 될 가능성이 있는 객체들을 전송받으면 M_{org}는 최종적으로 로컬 스카이라인 객체와 주변의 장치들로부터 받은 스카이라인 객체들을 합병하여 최종 스카이라인 객체를 구한다. 이 합병하는 과정에는 중복을 제거하고 지배 관계에 있는 객체들을 제거한다.

주변의 장치 M_1은 질의 요청과 함께 필터링 튜플과 값의 빈도수를 전송받는다. M_1은 먼저 로컬 객체들과 필터링 튜플을 비교한다. 필터링 튜플에 의해 지배되는 로컬 객체들은 최종 스카이라인 객체가 될 수 없기 때문에 로컬 스카이라인을 처리하는 연산에서 제외시킨다. 이 과정을 통해 스카이라인 질의를 수행하기 위해 비교될 로컬 객체들의 수가 감소됨으로 실행시간이 축소된다. 그리고 이 과정을 실행하는 동안에 M_{org}로부터 온 값의 빈도수에 로컬 데이터의 빈도 수를 누적시킨다. 그리고 나서 M_1은 남아있는 객체들을 사용하여 스카이라인 질의를 처리한다. 로컬 스카이라인 객체들과 M_{org}에서 전송된 필터링 튜플 중 DPL을 계산하여 새로운 MP와 필터링 튜플을 계산하고 새로운 필터링 튜플을 이웃 모바일 장치로 전송한다.

4.2 필터링 알고리즘

(그림 6)은 처음 질의가 수행된 장치 M_{org}에서 로컬 스카이라인 질의 처리를 하는 orginate_skyline() 알고리즘을 보여준다.

우선 알고리즘의 설명에 앞서, 필요한 함수와 변수들을 간략히 설명한다. 라인 (3)의 updateFrequency(S, F)는 빈도수를 갱신하는 함수이다. F는 모든 차원의 데이터의 빈도수를 가지고 있는 히스토그램이다. getMP(F)는 히스토그램 F를 가지고 MP를 구하는 함수이다. SK_i는 모바일 장치 M_i의 로컬 스카이라인 집합이다. 함수 minDPL(SK_i,MP)는 로컬 스카이라인 객체들의 집합 SK_i 에서 DPL의 최소값을 가지는 스카이라인 객체(즉, 필터링 튜플)을 리턴한다.

[Algorithm 1] originator_skyline()

```
Input
S : dataset of local
Output
  Z : final skyline
Procedure
1:  F, SK_i  // frequency histogram of data, local skyline set
2:  for each tuple S_j in S
3:    F = updateFrequency(S_j ,F)
4:    if ( S_j is skyline ) then  // compute skyline
5:  insert S_j into SK_org
6:  end for
7:  MP ← getMP(F)          // find a mode point
8:  T ← minDPL(SK_org ,MP)  //find a filtering tuple
9:  SK_1,SK_2,···, SK_n ← surrounding_skyline(T , F)
10: TEMP = ⋃_{i=1}^{n} SK_i ∪ SK_org
11: for each tuple z_i in TEMP
12:    if (z_i is skyline) then
13:      insert z_i into Z
14: endfor
15: return Z
```

(그림 6) 질의가 수행된 장치의 로컬 스카이라인 질의

먼저, 스카이라인 집합을 찾기 위해 로컬 데이터 집합 S를 스캔한다 (라인(2)-(6)). 이때, 스카이라인 계산을 위해 S를 스캔하는 동안 F를 갱신한다(라인 (3)). 또한, 스카이라인 객체들은 SK_{org}로 삽입된다 (라인 (4)-(5)).

F에 저장된 빈도수를 사용하여 각 차원의 최빈값을 계산하여 MP를 구한다 (라인 (7)). 각 스카이라인 객체의 DPL 값을 계산하여 최소값을 가지는 튜플을 필터링 튜플로 선택한다 (라인 (8)). 그리고 나서 주변의 다른 장치들에게 필터링 튜플 T와 히스토그램 F를 전송한다. 다른 장치 M_j가 로컬 스카이라인 객체의 집합인 SK_j를 리턴하면 (라인 (9)), SK_j와 SK_{org}를 병합하여 중복을 제거한다(라인 (10)). 이때, 전송 받은 스카이라인 객체들 사이에 지배관계가 존재할 수 있음으로 병합된 결과에 대하여 다시 스카이라인 계산하여 Z를 생성한다. (라인 (11)-(14)). 최종적으로 사용자는 최종 스카이라인 결과인 Z를 받는다.

(그림 7)에서 surrounding_skyline() 알고리즘은 다른 장치로부터 질의를 받은 장치가 필터링 튜플을 이용하여 스카이라인 질의를 처리하는 절차를 보여준다.

모바일 장치 M_j는 다른 장치로부터 질의와 함께 필터링 튜플 T와 히스토그램 F를 받는다. 먼저 M_j는 로컬 데이터

[Algorithm 2] surrounding_skyline(T , F)

Input
S : local dataset
T : the filtering tuple
F : frequency of data of previous devices.
Output
SK: local skyline set
Procedure
1: for each tuple S_j in S
2: F = updateFrequency(S_j ,F)
3: if (T dominate S_j) then // *filtering course*
4: do nothing
5: else if (S_j is skyline) then // *compute skyline*
6: insert S_j into SK
7: end for
8: MP ← getMP(F) // *find a mode point*
9: T ← minDPL(SK ∪ {T},MP)
10: return SK

(그림 7) 주변의 장치들에서의 로컬 스카이라인 질의

를 스캔하고 빈도 수 F를 누적시킨다 (라인 (2)). 그리고 T에 의해 지배되는 튜플들을 스카이라인 질의를 계산할 데이터에서 제외시킨다 (라인(3,4)). M_j는 남아있는 데이터의 스카이라인을 계산한다 (라인 (5,6)). 누적된 F를 사용하여 새로운 데이터의 최빈값 MP를 찾는다 (라인 (8)). 그리고 M_j는 각 로컬 스카이라인 객체들과 T중에서 최소 DPL값을 가지는 필터링 튜플을 구한다 (라인 (9)). 마지막으로 질의가 처음 수행된 장치로 SK를 리턴한다 (라인 (10)).

5. 실험 및 성능평가

이 절에서는 실험을 통하여 제안된 방법의 효율성을 검증한다. MANET 실험 환경에서 로컬 스카이라인을 계산할 때 필터링 튜플의 효율성을 확인한다.

비교 실험을 통한 성능 평가를 위해 모든 기법을 동일하게 Intel® CoreTM2 Duo CPU E6750 2.66GHz 프로세서와 2GB의 메인 메모리, 300GB의 하드 디스크를 가진 Windows XP 운영체제의 PC에서 자바 가상 메모리를 256MB로 고정한 환경에서 자바언어(Java SDK 1.6)를 이용하여 구현하고 실험을 수행하였다. 실험에서 사용한 변수들의 정보는 <표 2>에 나타나 있다.

〈표 2〉 실험 환경 변수

파라미터	값
모바일 장치의 수	2^2, 3^2, 4^2, … , 10^2
전체 릴레이션의 데이터 수	50K, 100K, … , 500K
로컬 릴레이션의 데이터 수	500, 1000, … , 5000
차원 수	2, 3, 4, 5, … , 10
데이터의 편향 비율	10%, 20%, … , 70%, 80%, 90%

모의실험을 위한 시뮬레이터에는 100개의 무선 이동 장치가 애드혹 네트워크를 구성하도록 설정하였다. 한 개의 장치는 500개에서 5000개의 정보를 가진다. 그리고 차원의 수에 따른 효과를 측정하기 위해 차원을 2차원에서 10차원까지 변화시키면서 실험하였다. 차원의 기본적인 실험값은 5이다.

비슷한 데이터의 값이 많이 나타나는 부분을 표현하기 위해서 전체 영역 중 25% 크기의 영역에 전체 데이터 객체 수의 10%에서 90%까지 객체들을 배치시켰다. 여기서 추가로 삽입한 데이터의 비율을 앞으로 편향된 데이터의 비율이라고 표현하겠다. 편향된 데이터의 비율은 기본적으로 70%로 고정하고 실험하였다.

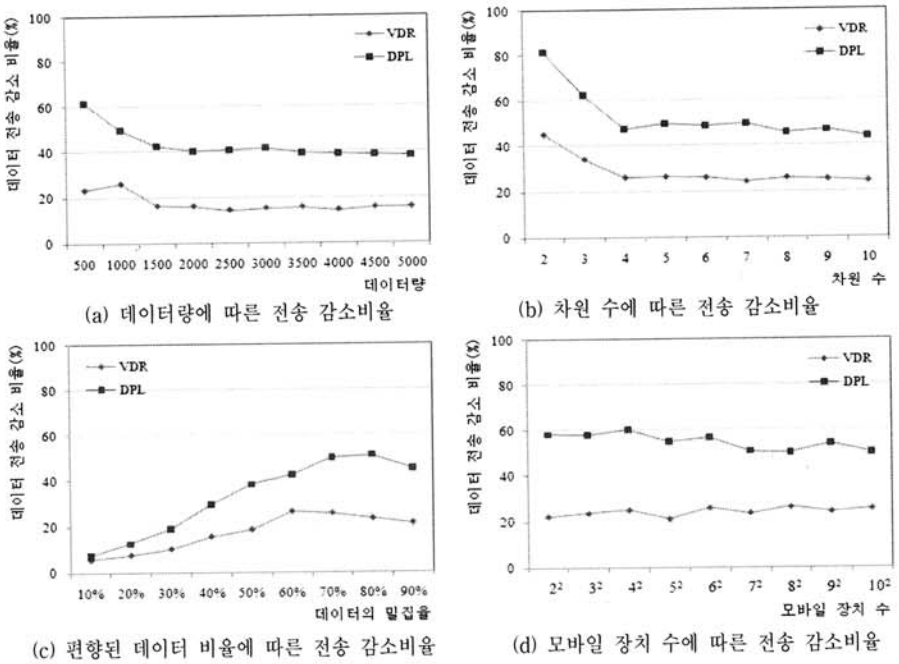
본 실험에서는 두 가지의 방향으로 성능을 측정하였다. 첫 번째 실험은 모바일 장치들 사이의 통신 비용이 감소되는 비율을 비교한다. 두 번째 실험은 각 모바일 장치에서 스카이라인 질의를 할 때 비교횟수의 감소를 측정한다. 본 연구에서 제안된 방법을 이 연구의 동기가 된 [6] 과 비교 평가한다.

5.1 통신 비용 실험

통신 비용의 효율을 실험하기 위하여 본 절에서는 모바일 장치 사이에 데이터를 전송할 때 데이터가 감소되는 비율을 측정한다. (그림 8)은 데이터량, 차원 수, 편향된 데이터의 비율, 모바일 장치 수를 변화시키면서 데이터 전송 감소 비율을 측정한 결과를 나타낸다.

데이터 전송 감소 비율은 로컬 스카이라인 집합에서 필터링 튜플에 의해 필터된 튜플들의 비율을 나타낸다. VDR은 [6]에서 사용하는 필터링 튜플을 선택하는 방법을 나타내고, DPL은 본 연구에서 제안한 필터링 튜플을 선택하는 방법을 나타낸다.

(그림 8)(a)는 로컬 모바일 장치의 데이터의 양을 늘려가면서 데이터 전송 감소 비율을 측정한 결과를 보여준다. 본 연구에서 제안한 필터링 튜플 선택 기준인 DPL을 이용한 방법이 VDR을 이용한 방법보다 더 큰 데이터 전송 감소 비율을 가지는 것을 볼 수 있다. 데이터량이 많아져도 비슷한 비율의 데이터 전송 감소 비율을 가지는 것도 확인할 수 있었다. (그림 8(b))는 차원의 수를 변화시키면서 데이터 전송 감소 비율을 측정한 결과를 보여준다. 저차원에서는 대부분의 데이터가 필터되지만 고차원으로 갈수록 필터링 되는 스카이라인 객체가 적어지는 것을 볼 수 있다. (그림 8)(c)는 편향된 데이터의 비율(데이터의 밀집율)을 변화시키면서 데이터의 전송 감소 비율을 측정한 결과를 나타낸다. 데이터의 밀집율이 높을수록 필터링 효율이 향상되며 DPL 방법이 더 큰 폭으로 효율이 증가되는 것을 볼 수 있다. DPL 방법은 데이터의 특성을 고려하여 필터링 튜플을 선택하기 때문에 데이터의 편향 비율이 높을수록 VDR방법보다 더 효율이 좋다. (그림 8)(d)는 모바일 장치의 수를 2^2에서 10^2까지 변화시키면서 데이터 전송 감소 비율을 측정한 결과를 보여준다. 통신을 하는 모바일 장치의 수가 많아



(a) 데이터량에 따른 전송 감소비율 (b) 차원 수에 따른 전송 감소비율

(c) 편향된 데이터 비율에 따른 전송 감소비율 (d) 모바일 장치 수에 따른 전송 감소비율

(그림 8) 로컬 릴레이션에서 데이터 감소 비율

져도 데이터 감소비율은 비슷한 양상을 나타냈다. 본 연구에서 제안한 필터링 튜플 선택 기준을 이용한 필터링 방법은 기존의 연구보다 모든 실험의 경우에서 더 우수함을 확인하였다.

5.2 로컬 모바일 장치의 계산

두 번째 실험으로는 단일 모바일 장치에서 질의가 실행되는 시간을 추정하기 위해 스카이라인 객체를 구하는데 비교되는 수와 필터링 튜플에 의해 필터링 되는 과정에서의 비교횟수의 합을 측정한다. 스카이라인 질의를 계산할 때의 실행 시간 중 가장 많은 비율을 차지하는 연산은 비교 연산이므로 비교횟수를 측정하였다. 비교 횟수를 측정한 실험의 결과는 (그림 9)에서 보여준다. (그림 9)의 y축은 단일 모바일 장치 한대 당 평균 비교 횟수를 나타내었다.

(그림 9(a))는 데이터량의 변화에 따른 각 단말 장치에서의 비교횟수의 평균을 나타낸다. *DPL*은 전체 데이터 중 필터링 튜플에 의해 지배되는 데이터를 제외하고 스카이라인 집합을 구하므로 *VDL* 보다 데이터의 비교횟수가 적다. (그림 9(b))는 차원의 수의 변화에 따른 스카이라인 질의를 할 때 각 장치에서의 비교횟수의 평균을 나타낸다. (그림 9(a))와 같은 이유로 *DPL* 방법으로 스카이라인 질의 수행할 때 비교횟수가 더 적음을 볼 수 있다. (그림 9(c))는 편향된 데이터의 비율에 따른 스카이라인 질의의 비교횟수를 나타낸다. 편향된 데이터의 비율이 높을수록 필터링 튜플에 의해 지배되는 양이 많으므로 비교횟수가 감소한다. (그림 9(d))는 모바일 장치의 수를 2^2에서 10^2까지 변화시키면서 스카이라인 질의의 비교횟수를 측정한 결과를 보여준다. 장치의 수가 증가할수록 필터링 튜플의 효율이 좋아지므로 비교횟수가 감소한다. 그러나 일정 수 이상으로 장치의 수가 증가하면 더 좋은 효율을 가진 필터링 튜플로 바뀔 가능성이 줄어드므로 비교횟수가 일정하게 유지된다.

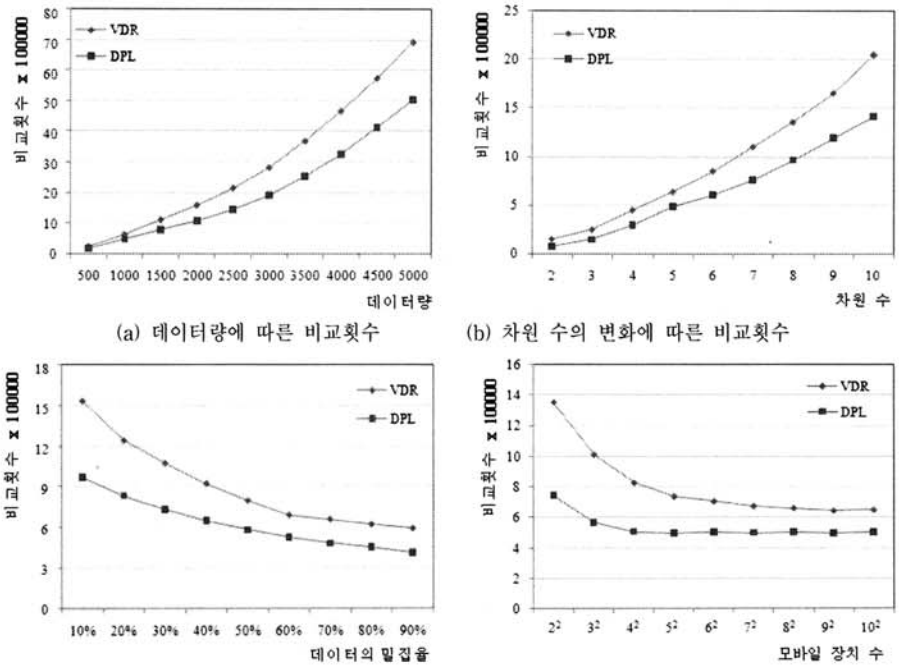
실험을 통하여 제안된 방법이 모든 실험에서 기존 연구보다 더 우수함을 확인하였다. *VDR* 방법은 스카이라인 질의를 계산하기 전에 필터링 하는 과정이 없다. 그러나 *DPL* 방법은 스카이라인을 계산하기 전에 필터링 단계를 가진다. 그러므로 *DPL* 방법으로 스카이라인 질의할 때 비교횟수가 *VDR* 방법과 비교하여 상당히 감소되는 것이다.

6. 결 론

본 연구는 MANET 환경에서 스카이라인 질의를 처리하는 문제를 다룬다. 각 장치는 전체 데이터의 일부분을 가지고 있다. 질의를 수행하는 장치는 전체 데이터를 검색하기 위해서 다른 주변의 장치들과 통신한다. 모바일 장치들 사이의 통신 비용과 각 단말 장치에서의 실행시간을 감소시키기 위해 필터링 튜플을 사용한다.

본 연구에서는 데이터의 최빈값을 기반으로 필터링 튜플을 선택하는 방법을 제안한다. 스카이라인 질의를 계산하기 전에 필터링 튜플에 의해 지배되는 객체들을 스카이라인을 계산할 데이터들 사이에서 제외시킨다. 이 과정을 거치면서 스카이라인을 계산하는 비교 횟수를 감소시킬 수 있으므로 전체 실행 시간이 줄어든다. 또한 필터링 튜플에 의해 지배되는 객체들 중에는 스카이라인 객체로써 질의가 수행된 장치로 전송될 객체도 포함되어 있으므로 제거된 스카이라인 객체의 양만큼 통신 비용이 감소된다.

제안된 방법의 효율성은 다양한 환경에서의 실험을 통해 확인하였다. 향후 연구 방향으로는 P2P 환경에서 스트림 데이터의 스카이라인 질의 처리를 고려한다.



(a) 데이터량에 따른 비교횟수 (b) 차원 수의 변화에 따른 비교횟수

(c) 편향된 데이터 비율에 따른 비교횟수 (d) 모바일 장치 수에 따른 비교횟수

(그림 9) 로컬 릴레이션에서 비교횟수

참 고 문 헌

[1] 이재훈, "Mobile Ad-hoc Network (MANET) 표준화 동향," 한국정보기술학회지, 제6권 제1호, pp.49-56, 2008.

[2] C. Perkins, Ad Hoc Networking, Addison Wesley, 2001.

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

[4] J. Chomicki, P. Godfery, J. Gryz, and D. Liang, "Skyline with presorting," In Proc. IEEE ICDE, p. 717-719, 2002.

[5] W. T. Balke, U. Guntzer, and J. X. Zheng. "Efficient distributed skylining for web information systems," In EDBT, p.256-273, 2004.

[6] Z. Huang, C. S. Jensen, H. Lu, and B. C. Ooi, "Skyline Queries Against Mobile Lightweight Devices in MANETs," In Proc. IEEE ICDE, p. 66, 2006.

[7] S. Wang, B. C. Ooi, A. K. H. Tung, and L. Xu, "Efficient skyline query processing on peer-to-peer networks," In Proc. IEEE ICDE, p.1126-1135, 2007.

[8] D. Sun, S. Wu, J. Li, and A. K. H. Tung, "Skyline-Join Distributed Databases," IEEE ICDE, 2008.

[9] B. Cui, H. Lu, Q. Xu, L. Chen, Y. Dai and Y. Zhou, "Parallel distributed processing of constrained skyline queries by filtering," IEEE ICDE, p.546-555, 2008.

[10] S. H. Yoon and C. Shahabi, "Distributed Spatial Skyline Query Processing in Wireless Sensor Networks," In IPSN, 2009.

[11] S. Antony, P. Wu, D. Agrawal, and A. E. Abbadi, "Efficient Skyline Computation over ad-hoc Aggregations," IEEE ICDE, 2008.

박 미 라

e-mail : happypmr@kut.ac.kr

2007년 한국기술교육대학교 인터넷미디어공학부 정보보호(학사)

2007년~현 재 한국기술교육대학교 정보미디어공학과 석사과정

관심분야 : 스카이라인 질의, 스트림 데이터 등

김 민 기

e-mail : kintelk@kut.ac.kr

2007년 한국기술교육대학교 인터넷미디어공학부 정보보호(학사)

2010년 한국기술교육대학교 정보미디어공학과(석사)

2010년~현 재 한국기술교육대학교 컴퓨터공학과 박사과정

관심분야 : 센서 네트워크, 스트림 데이터 등

민 준 기

e-mail : jkmin@kut.ac.kr

1995년 숭실대학교 전자계산학과(학사)

1997년 한국과학기술원 전산학과(석사)

2002년 한국과학기술원 전산학전공(박사)

2003년~2004년 한국과학기술원 Post-Doc 및 초빙교수

2004년 한국전자통신연구원 선임연구원

2005년~현 재 한국기술교육대학교 컴퓨터공학부 조교수

관심분야 : XML, 시공간DB, 스트림 데이터, 센서네트워크