공간 데이터스트림 조인 전략과 비용 모델

Strategy and Cost Model of Spatial Data Stream Joins

유기현*, 하태석*, 남광우**
Ki Hyun Yoo*, Tae Suk Ha*, Kwang Woo Nam**
군산대학교 컴퓨터정보공학과 석사과정*
군산대학교 컴퓨터정보공학과 조교수**
{ykhid*, kwnam**}@kunsan.ac.kr

요 약

현재의 센서 네트워크 시스템은 공간적 정보를 배제한 센서 데이터스트림에 대한 저장 및 검색 방안에 대한 연구에 치중되어 있다. 하지만, 이러한 센서 네트워크가 공간적 정보와 결합하게 되면 훨씬 더 많은 응용과 의미 있는 데이터로 가공될 수 있다.

본 논문은 GeoSensor Network에서 공간적 정보와 데이터스트림이 결합된 공간 데이터스트림 정의 및 공간 데이터스트림간 조인 전략들과 그에 따른 조인 전략들 간의 비용을 추정하는 비용 모델을 제시하였다. 공간 데이터스트림간 조인 전략을 위해 Nested Loop 조인, Grid File, R-tree 알고리즘을 사용하였고, 단방향 Nested Loop 조인, 단방향 Grid 조인, 단방향 R-tree 조인 기법들을 조합하여 조인 전략들 간의 비용을 추정하였다.

1. 서론

GeoSensor Network란 지리공간상에서 발생하는 다양한 현상들을 모니터링하는 특정형태의 센서네트워크 인프라 및 관련 소프트웨어를 의미한다. 여기에는 데이터를 수집하고, 집계하며, 분석하며, 공간적 시간적 상황에 따라 적절한 반응을 주기 위한 다양한 연구들이 포함될 수 있다.

GeoSensor Network의 주요 응용들의 예를 들자면, ①환경 모니터링(동식물 생태, 건물, 홍수 탐지, 토양 및 습기, 대기/해양 모니터링), ②객체 추적(차량/동물 추적, 군사, 물류), ③객체 감시(응급 의료, 침입 탐지, 지진위험, 산림 방재) 분야 등이 있다. 이들 센서 네트워크 응용의 공통점은 사용자 및 애플리케이션 레벨에서 실시간으로 데이터를 모니터링하거나 분석하기 위해 필요한 공간에 센서 장비를 설치하고, 주기적인 시간마다 데이터를 수집하는 방식을 취한다는 것이다. 따라서 센서 스트림 데이터는 인접한 시간과 공간 사이에 강한 시공간적 의존성과 인과관계 특성을 가지며, 센서 네트워크 미들웨어에 수집되는 센서 스트림 데이터는 일련의 센서 데이터 스트림들에 대해 시공간 통합적인 측면에서 종합적으로 분석되어야 한다. 이것은 기존의 센서 네트워크 시스템이 단순한 데이터 스트림의 정제, 축소, 집계, 요약 등과 같은 단순 데이터 계산형 시스템에서 필연적으로 시간적·공간적 semantics를 포함한 시스템의 형태로 확장될 필요성이 있음을 의미한다.

최근까지 센서 네트워크에서 발생하는 데이터 스트림 처리를 위한 다양한 연구들이 있었다. 대표적인 연구들은 센서 네트워크내에서 질의를 처리하는 In-Network 질의 처리 시스템인 COUGAR[1], 데이터 스트림 서버 상에 질의를 처리하는 데이터 스트림 관리 시스템들인 관계형 DB 모델에 기반 한 STREAM[2]과 객체 DB 모델에 기반 한 Aurora[3] 및 Tribeca[4], Uncertainty를 지원하는 TelegraphCQ[5] 등

+ 본 연구는 건설교통부 첨단도시기술개발사업 - 지능형국토정보기술혁신 사업과제의 연구비지원(07국토정보C05)에 의해 수행되었습니다.

이 있다. 그러나 이러한 시스템들은 공간 데이터 스트림 질의 처리를 지원하지 않으며, GeoSensor 스트림 처리에 대한 필요성 제기[6]에도 불구하고 아직까지 데이터 스트림 관리 시스템에서 공간센서 스트림에 대한 처리에 대한 연구는 극히 제한적으로 수행되고 있다.

본 논문은 GeoSensor Network 및 스트림 데이터베이스 연구에서 아직 제기되지 않았으며 해결되어야하는 다양한 문제들중에서 공간적 정보를 가진 공간 데이터스트림의 정의 및 공간 데이터스트림간 조인 전략들과 그에 따른 조인 전략들 간의 비용을 추정하는 비용 모델을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 각 스트림과 공간 데이터에 대한 조인 기법들, 그리고 관련 연구들을 살펴보고, 3장에서는 공간 데이터스트림 정의와 조인 전략들을 다룬다. 4장에서는 공간 데이터스트림간 조인을 위해 각 알고리즘에 대한 단방향 조인 비용을 설명한다. 5장에서는 다양한 실험 및 성능 평가를 보이고, 마지막 6장에서 결론 및 추후 연구해야 할 방향에 대해 언급한다.

2. 관련 연구

2.1 DSMS

센서들로부터 수집되는 센서 데이터는 매우 빠른 속도로 추가되고 한번 추가된 데이터는 삭제되거나 수정되지 않는 특징이 있다. 이러한 특징을 갖는 데이터를 실시간으로 처리하기 위해 데이터 스트림(Data Stream)이라는 새로운 형태의 데이터 모델에 대한 연속 질의(Continuous Query)를 처리할 수 있는 DSMS(Data Stream Management System)가 필요하게 되었다[7,8]. 이러한 DSMS는 디스크나 메모리에 저장되어 있는 모든 데이터를 질의 처리 대상으로 하는 기존 DBMS와는 다르게, 최근 데이터만을 처리하거나 과거부터 현재까지 입력된 데이터에 대한 요약 정보만을 처리한다. 즉, 디스크 등의 저장소에 쌓여있는 데이터를 메모리에 로드하는 대신 실시간으로 입력되는 데이터를 처리 대상으로 하며, 질의는 한 번만 수행되는 것이 아니라 시스템에 등록된 후 연속적으로 수행된다. 이러한 DSMS는 센서 네트워크뿐만 아니라 네트워크 모니터링, 결제 정보 분석, 판매 내역 트랜잭션 분석 등과 같은 다양한 분야에서 활용이 가능하다.

2.2 스트림 조인

스트림 데이터를 조인하기 위해서는 스트림 데이터의 특성을 고려하여 요구되는 것들이 있다. 예를 들면, 연속적인 스트림 데이터의 생성 때문에 발생하는 메모리 요구, 스트림 데이터 환경의 non-blocking 연산자, Sliding Windows, 혹은 스트림 데이터를 처리하기 위한 Batch Processing, Sampling, Synopses 등의 기법들이다. 이러한 기법들 중에서 Sliding Windows라는 하나의 윈도우를 적용하여 스트림 데이터를 조인하는 기법들이 많이 연구되고 있다. 다수의 스트림 데이터를 조인하기 위해 Window 조인 방법이 사용되었다[9,10,11]. Telegraph project는 multi-way 조인 연산자 구현을 제안했다[11]. 이 방법에서, 윈도우가 튜플 수의 관점에서 정의되고 각 스트림에서 새로운 튜플이 오래된 튜플을 제거하도록 강요된다. [9]는 시간 윈도우 개념을 사용한다. 또한, 두 스트림의 도착율이 다른 스트림들의 윈도우 조인을 다루는 방법도 연구되었다[10]. [12]은 실행 비용을 줄이기 위해 비대칭 조인 방법을 사용한다. 예를 들면, 하나의 스트림에 대해서 중첩 루프 조인을 사용하고, 다른 스트림에서는 해시 테이블을 만들어 사용한다. Batch Processing 기법을 사용한 XJoin은 Symmetric Hash Join을 한다[13]. [14]은 연속질의를 효율적으로 처리하기위해 공간조인 기반 연속질의 처리 알고리즘을 제안했다. 이 방법은 다차원 공간상에서 연속질의를 처리한다.

2.3 공간 조인

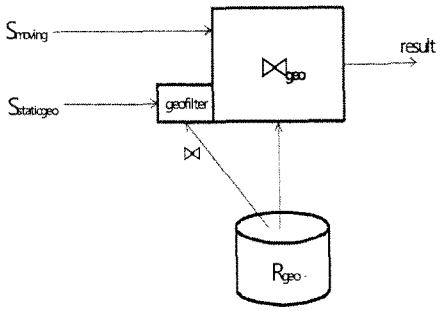
두 개의 공간 데이터들에 대해 인덱스가 존재 하는 경우 보통 이러한 인덱스들을 사용하는 것이 더욱 효과적이다. 그러나 때때로 양립할 수 없는 타입의 인덱스들이 존재 할 수 있다. Corral 등은 이러한 경우 하나의 인덱스를 무시하고 인덱스 중첩 루프 조인을 수행하는 것을 제안했다[15].

하나의 공간 데이터에만 인덱스가 있을 경우, 인덱스 중첩 루프 조인을 사용하여 공간 조인이 수행 될 수 있다. 다른 방법으로 인덱스가 없는 공간 데이터에 대해 인덱스를 구성하여 두 개의 인덱스들을 조인하는 방법이 있다. 만약 하나의 공간 데이터에만 인덱스가 있다면, 다른 공간 데이터는 부피-적재 기술(bulk-loading technique)을 사용하여 효율적으로 인덱스 될 수 있다[16]. 이 방법은 그 인덱스가 저장되고 나중에 재사용될 때 유용하다. 이렇게 구성되는 인덱스는 본래 가지고 있던 인덱스의 구조와 흡사하도록 만들어진다. Lo and Ravishankar는 Seeded-Tree를 제안했다[17]. 이 방법은 인덱스를 구성하기 위하여, Seed Level이라 부르는 본래 가지고 있던 인덱스의 상위 레벨을 사용한다. 이 상위 레벨을 사용하여 인덱스가 없는 공간 데이터를 분할한다. 그 데이터가 분할되면, 부피-적재 기술(bulk-loading technique)을 사용하여 하나의 R-Tree로 변환된다.

3. 공간 데이터스트림 조인 모델

3.1 GeoSensor 네트워크에서 공간 조인

GeoSensor 데이터스트림 시스템에서 S를 스트림이라 하고 R을 릴레이션이라고 할 때, 공간 조인의 대상이 될 수 있는 대상은 다음의 스트림과 릴레이션 세 가지로 구성된다.



<그림 1> GeoSensor 데이터 스트림 시스템에서의 공간 죠인

$S_{staticgeo}$: 정적 공간 데이터스트림으로서, 특정 공간상 위치에 고정된 상태로 설치되어 있는 센서로 부터의 입력과 같이 실제 들어오는 데이터에는 공간좌표가 없으나 센서가 설치된 실제 공간 좌표가 저장되어 있는 테이블과의 조인을 통해 공간 스트림으로 변환된다.

S_{moving} : 이동체 데이터스트림으로서, GPS 위치와 같이 동적으로 이동하는 센서에 의해 전송되는 공간 데이터스트림이다.

R_{geo} : 공간 릴레이션으로서, 데이터베이스 내에 지리정보를 저장하고 있는 릴레이션이다.

GeoSensor 릴레이션과 스트림이 위와 같다고 할 때 GeoSensor 데이터스트림 시스템에서 처리해야하는 공간 조인의 종류와 예는 표 1과 같다. 여기서 공간 데이터스트림은 $S_{staticgeo}$와 S_{moving}이다. 따라서, 공간 데이터스트림간 조인은 표 1에서 R_{geo}와의 조인을 제외한 나머지3가지의 경우들이다. 이 3가지의 조인에 대한 비용 모델 및 조인 전략들을 본 논문에서 제시한다.

공간 데이터스트림 질의는 그림 1과 같이 $S_{staticgeo}$와 S_{moving}, 그리고 R_{geo}의 데이터들에 대하여 수행될 수 있다. 이 때, $S_{staticgeo}$는 공간 조인에서 수행되기 위해 실제 공간 데이터스트림으로 변환되는 추가적인 연산을 필요로 한다.

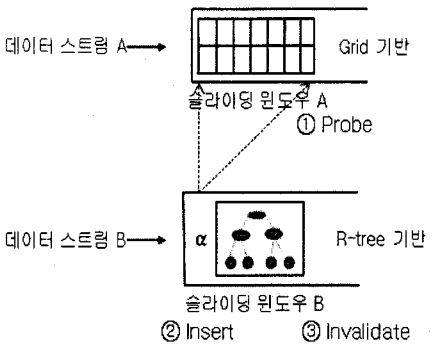
3.2 공간 데이터스트림간 조인의 비용 프레임워크

일반적인 데이터스트림 간의 조인 전략은 슬라이딩 윈도우 조인 방법을 많이 사용한다[8].

그림 2와 같이 데이터스트림 A와 B로부터 스트림데이터가 들어온다고 가정했을 때, 데이터스트림 B로부터 새로운 데이터 엘리먼트 a가 입력되었을 때의 조인 수행 과정은 일반적으로, (1) 새로 들어온 데이터 엘리먼트와 데이터스트림 A의 슬라이딩 윈도우의 데이터 엘리먼트들 간에 조인 조건을 만족하는지 검사하는 과정(probe), (2) 새로운 데이터 엘리먼트를 데이터스트림 A에 삽입하는 과정(insert), (3) 데이터 엘리먼트의 삽입과 함께 슬라이딩 윈도우의 범위를 벗어난 데이터를 데이터스트림 B의 슬라이딩 윈도우로부터 삭제하는 과정(invalidate)으로 이루어진다. 본 논문은 여기서 슬라이딩 윈도우의 구조를 Grid 기반과 R-tree 기반으로 구성하여 공간 데이터와의 조인을 효율적으로 처리할 수 있도록 하였다.

데이터스트림 A와 데이터스트림 B에 대한 슬라이딩 윈도우 조인 비용 공식은 아래와 같다[7].

$$C_{A\bowtie B} = \lambda a(probe(b) + insert(a) + invalidate(a)) + \lambda b(probe(a) + insert(b) + invalidate(b))$$



<그림 3> 슬라이딩 윈도우 조인 전략

이 공식의 첫 번째 요소가 스트림 A에 대한 처리 비용을 측정한다. 그리고 단위 시간당 도착 튜플들을 각 요소에 곱해준다. (ex. $\lambda a, \lambda b$) 위의 공식에서 조인 연산의 비용을 두 개의 독립적인 서브그룹으로 나눌 수 있다.

$$C_{A\bowtie B} = C_{A\times B} + C_{A\times B}$$
$$C_{A\times B} = \lambda a(probe(b)) + \lambda b(insert(b) + invalidate(b))$$
$$C_{A\times B} = \lambda b(probe(a)) + \lambda a(insert(a) + invalidate(a))$$

3.3 공간 데이터스트림간 조인 전략

본 논문에서는 공간 데이터스트림간 조인 전략을 위해서 Nested Loop 조인, Grid File, R-tree 알고리즘을 사용한다. 여기서 Grid File과 R-tree 알고리즘은 공간 속성을 가진 데이터를 조인하기 위해 사용하였다. 그리고 슬라이딩 윈도우 조인 비용 공식이 $C_{A\bowtie B} = C_{A\times B} + C_{A\times B}$ 으로

<표 1> GeoSensor 데이터스트림 조인의 종류와 질의 예

GeoSensor 스트림 조인	질의의 예
$S_{staticgeo} \bowtie R_{geo}$	공원 1km이내의 고정 온도센서들의 최근 30초간 온도 평균을 보이시오
$S_{moving} \bowtie R_{geo}$	공원을 지나가는 GPS 차량들의 최근 30초간 평균 속도를 보이시오
$S_{staticgeo} \bowtie S_{staticgeo}$	고정 온도 센서들 간의 거리가 1km 이내이면서 10도 이상 온도차가 나는 센서들을 보이시오
$S_{staticgeo} \bowtie S_{moving}$	온도가 30도 이상인 센서로부터 1km 이내 지역을 지나가는 GPS 차량들의 30초 평균 속도를 보이시오
$S_{moving} \bowtie S_{moving}$	120km이상의 속도로 500m이내의 거리를 두고 지나가는 GPS 차량들을 보이시오

나뉘는 점을 이용하여 각각에 대해 단방향의 NL 조인, Grid 조인, R-tree 조인 방법을 사용하였다.

위의 3개의 조인 방법들을 조합하면 표 2와 같이 9개의 조인 전략들이 발생할 수 있다.

<표 2> 단방향 조인 전략들

	단방향 NL 조인	단방향 Grid 조인	단방향 Rtree 조인
단방향 NL 조인	NL-NL	NL-Grid	NL-Rtree
단방향 Grid 조인	Grid-NL	Grid-Grid	Grid-Rtree
단방향 Rtree 조인	Rtree-NL	Rtree-Grid	Rtree-Rtree

3.4 Grid File을 사용했을 때의 장단점

Grid File은 완전히 명세된 질의에 대해 최대 두 번의 디스크 접근으로 검색 연산을 수행할 수 있는 장점이 있다. 또한, 모든 속성에 관하여 효율적인 범위 질의가 가능하다.

Grid는 격자의 크기를 잘 결정해야하는 문제가 존재하고 Grid Directory를 재구성하는데 비용이 크다. Gird Array의 크기가 대개 매우 크기 때문에 분할이 발생해서 Directory의 재구성을 수행해야한다면 거기에 소요되는 비용이 매우 크게 된다. 일정한 데이터 셋의 경우는 문제가 되지 않지만 Skewed data set의 경우에 Directory 크기가 불필요하게 커진다.

3.5 R-tree를 사용했을 때의 장단점

R-tree는 공간 데이터를 인덱싱하는데 효율적인 구조이다. R-tree의 탐색은 포함(coverage)과 겹침(overlap) 관계가 모두 최소일 때 가장 효율적이 된다. 하지만, 겹침 관계를 최소로 유지하기가 어려운 단점이 있다. 또한, R-tree는 삽입 삭제에 의해 rebalancing 문제가 존재한다.

4. 공간 데이터 스트림 조인의 단위 비용

이 장에서는 각 알고리즘의 단방향 조인 비용을 예측하기 위한 모델을 제안한다. 표 3은 비용 모델을 위한 심벌을 정리한 것이다.

<표 3> 비용 모델을 위한 심벌과 정의

심벌	정의
$\dim$	차원 수(2차원 기준)
D	데이터 집합의 밀도
D_k	차원 k에서의 데이터 집합의 밀도
N	그리드 해상도
M	데이터 집합의 객체 수
f	평균 R-tree 노드 팬아웃
q_k	차원 k에서의 질의 사각형 q의 평균 범위
$D_{B_i,l}$	레벨 l에서 R-tree B_i의 데이터 집합의 밀도
λ_b	스트림 B의 도착률
B	슬라이딩 윈도우 B에서의 튜플 수
P_d	검색 가중치 요소
I_d	갱신 가중치 요소

4.1 단방향 Nested Loop 조인(NLJ)의 비용

A로 부터 B에 대한 단방향 중첩 조인의 비용은 다음과 같다.

$$C_{A\times B}(NJ) = \lambda aB \times P_N + 2\lambda b \times I_N$$

where P_N= NLJ 검색 가중치
I_N= NLJ 갱신 가중치

$\lambda aB \times P_N$은 윈도우 B를 검색하기 위해 접근하는 튜플들의 수를 나타낸다. λaB는 $\lambda a(probe(b))$와 동일하다. 여기에 NLJ에서의 검색 가중치 요소를 곱해줬다. $2\lambda b \times I_N$은 $\lambda b(insert(b)+invalidate(b))$와 동일한데, $insert(b)$와 $invalidate(b)$가 튜플 수 기반의 윈도우일 경우 각각 1이기 때문이다. 똑같이 NLJ에서의 갱신 가중치 요소가 곱해진다.

4.2 단방향 Grid 조인의 비용

A로 부터 B에 대한 단방향 Grid 조인의 비용은 다음과 같다[18].

$C_{A\times B}$(Grid)

$$= \lambda a \times \left(M_{B,i} \prod_{k=1}^{\dim} \left(\left(\frac{D_{B,k}}{B} \right)^{\frac{1}{\dim}} + N^{-1} \right) N^{\dim} \right) \times P_G$$
$$+ 2 \times \lambda b \times \left(M_{B,i} \prod_{k=1}^{\dim} \left(\left(\frac{D_{B,k}}{B} \right)^{\frac{1}{\dim}} + N^{-1} \right) N^{\dim} \right) \times I_G$$

where P_G = Grid 조인 검색 가중치

I_G = Grid 조인 갱신 가중치

$\left(M_{B,i} \prod_{k=1}^{\dim} \left(\left(\frac{D_{B,k}}{B} \right)^{\frac{1}{\dim}} + N^{-1} \right) N^{\dim} \right)$은 Grid 조인의 검색 비용을 나타낸다. 여기에 Grid 조인 검색 가중치를 곱해줬다. 삽입과 삭제 연산을 위해 삽입, 삭제 비용에 Grid 조인의 검색 비용을 곱해준다. 그러면 $2 \times \left(M_{B,i} \prod_{k=1}^{\dim} \left(\left(\frac{D_{B,k}}{B} \right)^{\frac{1}{\dim}} + N^{-1} \right) N^{\dim} \right)$이 된다. 여기서도 갱신 가중치를 곱해준다.

4.3 단방향 R-tree 조인의 비용

A로 부터 B에 대한 단방향 R-tree 조인의 비용은 다음과 같다[19].

$C_{A\times B}$(R-tree)

$$= \lambda a \times \sum_{i=1}^{1+\lceil \log_f \frac{data_{B_i}}{f} \rceil} \left\{ \frac{data_{B_i}}{f^i} \cdot \prod_{k=1}^{\dim} \left(\left(D_{B_i,i} \cdot \frac{f^i}{data_{B_i}} \right)^{\frac{1}{\dim}} + q_k \right) \right\} \times P_R$$
$$+ 2 \times \lambda b \times \sum_{i=1}^{1+\lceil \log_f \frac{data_{B_i}}{f} \rceil} \left\{ \frac{data_{B_i}}{f^i} \cdot \prod_{k=1}^{\dim} \left(\left(D_{B_i,i} \cdot \frac{f^i}{data_{B_i}} \right)^{\frac{1}{\dim}} + q_k \right) \right\} \times I_R$$

where P_R = R-tree 조인 검색 가중치

I_R = R-tree 조인 갱신 가중치

$\sum_{i=1}^{1+\lceil \log_f \frac{data_{B_i}}{f} \rceil} \left\{ \frac{data_{B_i}}{f^i} \cdot \prod_{k=1}^{\dim} \left(\left(D_{B_i,i} \cdot \frac{f^i}{data_{B_i}} \right)^{\frac{1}{\dim}} + q_k \right) \right\}$은 R-tree에서의 검색 비용을 나타내다. 여기에 R-tree에서의 검색 가중치 P_R를 곱해줬다. 삽입과 삭제 연산을 위해 삽입, 삭제 비용에 R-tree의 검색 비용을 곱해주면 $2 \times \sum_{i=1}^{1+\lceil \log_f \frac{data_{B_i}}{f} \rceil} \left\{ \frac{data_{B_i}}{f^i} \cdot \prod_{k=1}^{\dim} \left(\left(D_{B_i,i} \cdot \frac{f^i}{data_{B_i}} \right)^{\frac{1}{\dim}} + q_k \right) \right\}$이다. 여기에 갱신 가중치 I_R이 곱해진다.

4.4 가중치 요소 평가

초당 100개의 튜플 도착률을 가지는 검색 작업에 대한 CPU 시간을 측정하기 위해, 우리는 일괄적으로 6000개의 튜플을 처리하고 전체 실행 시간을 측정했다. 측정된 가중치 요소를 가진 비용 공식은 아래와 같다.

$C_{A\times B}$(NJ) = $\lambda a B \times 3.5 \times 10^{-4} + 2\lambda b \times 10^{-4}$

$C_{A\times B}$(Grid)

$$= \lambda a \times \left(M_{B,i} \prod_{k=1}^{\dim} \left(\left(\frac{D_{B,k}}{M_B} \right)^{\frac{1}{\dim}} + N^{-1} \right) N^{\dim} \right) \times 6.2 \times 10^{-4}$$
$$+ 2 \times \lambda b \times \left(M_{B,i} \prod_{k=1}^{\dim} \left(\left(\frac{D_{B,k}}{M_B} \right)^{\frac{1}{\dim}} + N^{-1} \right) N^{\dim} \right) \times 6.5 \times 10^{-4}$$

$C_{A\times B}$(R-tree)

$$= \lambda a \times \sum_{i=1}^{1+\lceil \log_f \frac{data_{B_i}}{f} \rceil} \left\{ \frac{data_{B_i}}{f^i} \cdot \prod_{k=1}^{\dim} \left(\left(D_{B_i,i} \cdot \frac{f^i}{data_{B_i}} \right)^{\frac{1}{\dim}} + q_k \right) \right\} \times 4.7 \times 10^{-4}$$
$$+ 2 \times \lambda b \times \sum_{i=1}^{1+\lceil \log_f \frac{data_{B_i}}{f} \rceil} \left\{ \frac{data_{B_i}}{f^i} \cdot \prod_{k=1}^{\dim} \left(\left(D_{B_i,i} \cdot \frac{f^i}{data_{B_i}} \right)^{\frac{1}{\dim}} + q_k \right) \right\} \times 4.7 \times 10^{-4}$$

5. 공간 조인의 비용 실험 및 평가

5.1 실험 데이터 및 실험 환경

이 논문에서 사용하는 실험 데이터는 S_{moving}과 $S_{staticgeo}$ 두 개의 스트림데이터들이다. 앞에서 정의한 것처럼, S_{moving}은 GPS 위치와 같이 동적으로 이동하는 센서에 의해 전송되는 공간 데이터스트림이고, $S_{staticgeo}$는 고정된 위치에 있는 공간 데이터스트림이다. $S_{staticgeo}$는 그리드 형태로 센서들이 배치되어 있다고 가정하고, 이것에 대한 공간 좌표는 저장되어 있는 테이블과의 조인을 통해 얻을 수 있다.

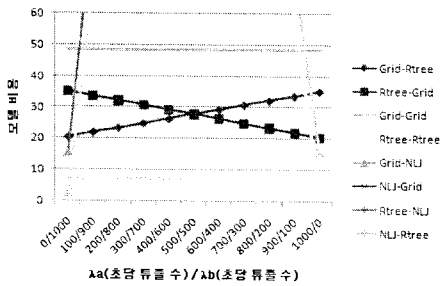
실험은 AMD Athlon 64X2 Dual 2.02GHz의 CPU와 1GB의 메모리, 그리고 윈도우 XP에서 수행했다.

5.2 실험평가

5.2.1 조인 전략들의 비용 비교

본 논문에서 제시하는 단방향 조인 알고리즘들을 조합하면 9가지의 공간 데이터스트림간 조인 전략들을 만들 수 있다.

그림 3은 본 논문이 제안하는 조인 전략들의 비용을 추정하는 그래프이다. 데이터스트림 A, B로부터 입력되는 튜플들의 수가 다를 때, 각각의 조인 전략들의 비용을 보이고 있다. NLJ-NLJ 조인 전략은 비용이 너무 커서 그래프에서 제외했다.



<그림 4> 단방향 조인 조합 비용
(슬라이딩 윈도우 A, B = 5000)

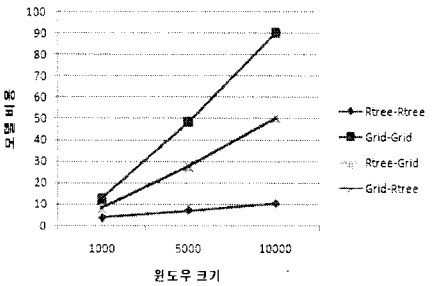
5.2.2 슬라이딩 윈도우 크기에 따른 비용

그림 4는 서로 다른 슬라이딩 윈도우 크기에 따른 처리 비용을 나타내는 그래프이다. Rtree-Rtree 조인은 윈도우의 크기에 영향을 적게 받는 반면, Grid-Grid 조인의 경우 윈도우의 크기에 비용이 다른 조인들에 비해서 많이 증가하는 것을 볼 수 있다.

6. 결 론

본 논문에서는 공간 데이터스트림간 조인 전략과 그에 따른 조인 전략들의 비용을 추정하는 비용 모델을 제시하였다. 그리고 실험을 통해 제안하는 비용 모델의 성능을 검증하였다.

본 논문은 슬라이딩 윈도우 조인 비용 공식이 $C_{A\times B} = C_{A\times B} + C_{A\times B}$ 으로 나뉘는



<그림 5> 슬라이딩 윈도우 크기에 따른 비용

점을 이용하여 각각에 대해 단방향의 NL 조인, Grid 조인, R-tree 조인으로 조합한 조인 전략들을 제안한다.

각 조인 전략에 대한 공간 데이터스트림간 조인 비용을 실험을 통하여 비교 할 수 있었다. 이 실험에서, Rtree-Rtree 조인이 가장 좋은 성능을 보인다는 것을 알 수 있었다. 또한 슬라이딩 윈도우의 크기에 따른 비용 변화를 실험을 통하여 확인 할 수 있었다.

본 논문의 향후 연구는 공간 데이터스트림간의 조인 개념을 공간 데이터스트림과 공간 릴레이션과의 조인으로 확장하는 것이다.

참고 문헌

1. Y. Yao, J. Gehrke, "The Cougar Approach to In-Network Query Processing in Sensor Networks," ACM SIGMOD Record, Vol.31 No.3, 2002, pp. 9-1.
2. A. Arasu, B. Babcock, S. Babu, M. Datar, K. Ito, I. Nishizawa, J. Rosenstein, and J. Widom, "STREAM: The Stanford Stream Data Manager," IEEE Data Engineering Bulletin, Vol.26 No.1, 2003, pp. 19-26.
3. D. Abadi, D. Carney, U. Cetintemel, M. Cherniack, C. Convery, S. Lee, M. Stonebraker, N. Tatbul, and S. Zdonik, "Aurora: A New Model and Architecture for Data Stream Management," VLDB Journal, 2003.

4. M. Sullivan, and A. Heybey, "Tribeca: A System for Managing Large Databases of Network Traffic," In Proc. of USENIX Annual Technical, 1998.
5. S. Chandrasekaran, O. Cooper, A. Deshpande, M. J. Franklin, J. M. Hellerstein, W. Hong, S. Krishnamurthy, S. R. Madden, V. Raman, F. Reiss, and M. A. Shah, "TelegraphCQ: Continuous Dataflow Procesing for an Uncertain World," Proceedings of the CIDR conference, 2003.
6. S. Nittel, A. Stefanidis, I. Cruz, M. Egenhofer, D. Goldin, A. Howard, A. Labrinidis, S. Madden, A. Voisard, and M. Worboys, "Report from the First Workshop on GeoSensor Networks", SIGMOD record, Special Issue on "Sensor Network Technology Infrastructure, Security, Data processing, and Deployment", Ed.Vijay Kumar, March, 2004.
7. L. Golab, and M. T. Ozsu, "Issues in Data Stream Management," ACM SIGMOD Record, Vol.32 No.2, 2003, pp. 5-14.
8. B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and Issues in Data Stream Systems," In Proc. of the ACM Symposium on Principles of Database Systems, 2002, pp. 1-16.
9. S. Chandrasekaran and M. J. Franklin, "Streaming queries over streaming data," In *Proc. of the VLDB Conference,* 2002.
10. J. Kang, J. F. Naughton, and S. D. Viglas, "Evaluating window joins over unbounded streams," In *ICDE, Feb*, 2003.
11. S. Madden, M. Shah, J. Hellerstein, and V. Raman, "Continuously adaptive continuous queries over streams," In *Proc. of the SIGMOD Conference, June*, 2002.
12. M. A. Hammad, W. G. Aref, and A. K. Elmagarmid, "Stream window join: Tracking moving objects in sensornetwork databases," In *SSDBM*, 2003.
13. T. Urhan, and M. Franklin, "XJoin: A reactively scheduled pipelined join operator," IEEE Data Engineering Bulletin, 2000.
14. 임효상, 이재길, 이민재, 황규영, "데이터와 질의의 이원성을 이용한 데이터스트림에서의 연속질의 처리," 정보과학회논문지, 제33권 제3호, 2006.
15. A. CORRAL, M. VASSILAKOPOULOS, AND Y. MANOLOPOULOS, "Algorithms for joining R-trees and linear region quadtrees," In *Advances in Spatial Databases—-6th International Symposium, (SSD)* (Hong Kong, China), R. H. G" uting et al. Eds. Lecture Notes in Computer Science, vol. 1651, Springer Verlag, g 251-269.
16. G. R. HJALTASON, AND H. SAMET, "Improved bulk-loading algorithms for quadtrees" In *Proceedings of the 7th ACM International Symposium on Advances in Geographic Information Systems,* (Kansas City, MO), C. B. Medeiros, Ed. 110-115.
17. M.-L. LO, AND C. V. RAVISHANKAR, "Spatial joins using seeded trees," In *Proceedings of the ACM SIGMOD Conference,*(Minneapolis, MN). 209-220.
18. 김진덕, 홍봉희, "고정 그리드를 이용한 병렬 공간 조인을 위한 비용 모델," 정보과학회논문지, 제28권 제4호, 2001.
19. Y. Theodoridis, E. Stefanakis, T. Sellis, "Cost Models for Join Queries in Spatial Databases," Proc. of Int. Conf. on Data Engineering, 1998, pp. 476-483.