韓國 컴퓨터情報學會 2006동계학술발표논문집&학회지
第14卷 第2號, 2006. 12.

센서 데이터 스트림 환경에서 효율적인 질의처리 연구방향

안 동 찬*

Research Directions for Efficient Query Processing over Sensor Data Streams

Dong Chan An *

요 약

센서 네트워크는 센싱(sensing), 연산(computation), 통신(communication) 능력을 갖춘 센서 노드들의 무선 네트워크라고 할 수 있다. 각각의 센서 노드들은 노드에 있는 하나 이상의 센서들에 의해 얻어지는 데이터 아이템들을 생성한다. 이러한 특징은 센서 네트워크가 분산 데이터베이스 시스템과도 유사한 특징을 가지고 있음을 알 수 있다. 그러나 센서 네트워크에서 센서 노드의 제한된 전력(power)과 메모리 자원은 가장 큰 단점이며 주요 연구 대상이 되고 있다. 본 연구에서는 이러한 센서 네트워크 환경에서 발생되는 데이터 즉, 센서 데이터 스트림을 제한된 자원을 효율적으로 활용하면서 질의에 대해 효율적으로 응답하는 방안을 여러 가지 측면에서 모색해 보았다.

Abstract

The sensor network is a wireless network of the sensor nodes which sensing, computation and communication ability. Each sensor nodes create the data items by sensor nodes above one. Like this feature, the sensor network is similar to distributed data base system. The sensor node of the sensor network is restricted from the power and the memory resources is the biggest weak point and is becoming the important research object. In this paper, We try to see efficient sensor data stream management method and efficient query processing method under the restricted sensor network environment.

▸ Keyword : 센서 네트워크(sensor network), 데이터 스트림(data stream), 질의 처리(query processing)

• 제1저자 : 안동찬
* 안산공과대학 디지털미디어과 조교수

Ⅰ. 서 론

센서 네트워크는 불규칙적인 토폴로지(topology)를 가진 네트워크 형태로서 주로 근거리 범위를 담당하는 센서 노드들로 이루어져 있으며, 각각의 센서 노드들은 빛(light), 온도(temperature), 진동(vibration) 등의 특정한 물리적인 현상을 센싱하는 역할을 수행한다. 그 예로 스마트 빌딩 환경, 지능형 교통망시스템 그리고 스마트 홈 등을 들 수 있다[5,8,9].

센서네트워크에서 센서 노드는 하나 이상의 센싱 장치를 통해 데이터 아이템을 생성한다. 그러므로 센서 네트워크는 분산 환경에서 데이터 튜플(tuple)의 스트림을 생성하는 분산 데이터베이스 시스템과 아주 유사하다.

최근의 연구에서도 센서 데이터베이스[4]라는 용어를 많이 사용하고 있으며 전통적인 데이터베이스와 유사하게 센서 네트워크에서도 감지된 튜플 데이터의 수집(gather)/처리(process)에 대한 질의가 실행된다. SQL을 사용하는 전통적인 데이터베이스의 질의는 데이터베이스에 있어서 매우 일반적인 것이며, 전형적인 센서 네트워크에서 처리해야 할 데이터의 양이 엄청나게 많기 때문에 효율적인 SQL 질의의 구현은 아주 중요하다[2].

센서 네트워크에서 센서 노드들 간의 메시지 전송을 통하여 가장 많은 전력을 소모하게 된다[11]. 따라서 센서 네트워크에서 SQL 질의의 분산 구현은 통신비용을 최소화 하는데 중점을 두여야 한다[2].

Ⅱ. 센서 환경(Sensor Environments)

2.1 센서 네트워크(Sensor Network)

센서 네트워크분야에서 주요 연구대상 중 하나는 바로 센서 노드일 것이다. 센서노드는 최근 많은 연구를 통해 리눅스(Linux)나 윈도우즈(Windows CE) 같은 운영체제에서 사용이 가능한 단계에 이르렀다. 그림1.은 미국 버클리 대학교에서 개발한 버클리 MICA 모트(Berkely MICA Mote) 이다[10]. 상업적으로 이용이 가능한 단계의 제품이다. 그러나 이러한 센서 네트워크 환경에서 사용되는 소형의 장치들은 아래와 같은 제약사항을 가지고 있다. 첫째, 통신(communication) 제약이다. 무선 네트워크에 연결된 센서 노드는 일반적으로 제한된 대역폭(bandwidth), 데이터의 손실(drop) 그리고 가변적인 잠재주기(latency)와 같은 제한된 QoS(quality of service)를 제공한다[7].

둘째, 전력소모(power consumption) 제약이다. 센서 노드의 전력 공급은 주로 배터리(battery)를 이용하기 때문에 시스템 설계시 전력소모에 대한 부분을 최우선으로 고려해야 한다. 예를 들면, MICA mote는 2000mAh를 제공하는 AA 배터리 2개를 사용하는데[10], 이것은 대략 주 1회 데이터 전송을 고려할 경우 약 1년간 전력공급이 가능하다.

셋째, 연산(computation) 제약이다. 센서 노드는 제한된 연산능력과 제한된 기억장치 용량을 가지고 있다. 이러한 제약은 센서노드에서 데이터 처리 알고리즘을 제약하고, 센서 노드 상에 중간 결과의 저장을 제한한다.

넷째, 센서 감지의 불확실성(uncertainty) 제약이다. 물리장치인 센서에서 신호(signal)의 감지는 본래 불확실성을 가지고 있는데 이것은 잡음(noise)등을 포함 할 수 있다. 또한 센서의 고장 또는 오동작으로 부정확한 데이터를 생성할 수 있다.

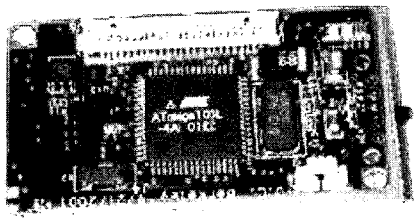

그림1. 버클리 MICA 모트
Fig1. Berkely MICA Mote

센서 네트워크는 전쟁터에서와 같이 센서 노드들의 많은 움직임과 짧은 라이프타임(life time)을 요구하는 경우와 환경 감시(monitor) 같은 과학 연구 프로젝트에서 장기간 실행되는 질의를 위한 전력 공급 등이 중요한 이슈이다. 고비용의 질의처리 기술은 짧은 실행 시간과 정확도를 향상시키지만 전력은 더 많이 소모한다. 따라서 이러한 트레이드오프(trade-off)를 잘 이용하여야 한다.

2.2 센서 데이터(Sensor Data)

센서 노드는 물리적 세계에 연결된 하나 이상의 센서들로 이루어져있다. 예를 들면 센서는 온도 감지 센서(temperature sensor), 빛 감지 센서(light sensor), 사물의 움직임을 감지하는 센서 등이다. 이들 센서의 데이터 소스는 분산되어 있고, 타임스탬프(time stamp), 센서 타입, 센싱 값 등으로 이루어진 센서의 아이디(id)와 위치

(location)들을 포함하는 몇 개의 필드(field)로 구성된 레코드(record)를 가지고 있다. 다른 노드에서 감지된 같은 센서 타입의 레코드는 같은 스키마(schema)를 가지고 있으며, 분산 테이블 형태로 이루어져 있다. 이러한 점에서 센서 네트워크는 다양한 센서 타입의 다중(multiple) 테이블로 이루어진 대형 분산 데이터베이스 시스템으로 보아도 무방할 것이다.

센서 데이터는 특성상 잡음(noise)을 포함하고 있는데 경우에 따라서는 여러 센서들로 들어온 데이터의 통합(fusion)으로 인해 더욱 정확한 결과를 얻기도 한다[12]. 그러므로 가공되지 않은(raw) 센서 데이터의 요약(summary)과 집계(aggregation)는 개개의 센서를 감지하는 것보다 센서 어플리케이션에서 더 유용할 수 도 있다[3]. 예를 들면 화학적으로 위험한 지역에서 모니터링을 한다고 할 때, 가능한 질의는 그 지역의 모든 센서의 측정값의 평균을 구하고 측정값이 평균값 이상으로 측정되었는지의 여부를 알리는 경우이다.

2.3 질의(Query)

센서 네트워크에서는 일반적으로 선언적(declarative) 질의를 사용한다. 그림2.와 같은 단순한 형태의 질의를 고려해 보자[6].

SELECT $A_1, A_2, \ldots, A_n$
FROM $r_1, r_2, \ldots, r_n$
WHERE P
GROUP BY A_m
HAVING P_m
DURATION time interval
EVERY time span

그림2. 단순한 형태의 질의
Fig2. Simple Query

SELECT 절은 센서 데이터의 속성(attribute)과 집계(aggregate)를 포함한다. FROM 절은 센서 타입의 분산 테이블을 포함한다. WHERE 절은 프레디킷(predicate)에 의한 센서 레코드 등의 필터링(filtering) 기능을 포함한다. GROUP BY 절은 센서 레코드를 어떤 특정 속성으로 분류한다. HAVING 절은 프레디킷에 의한 그룹의 제거와 FROM 절의 몇몇 테이블의 조인(join) 질의도 가능하다. 일반적인 질의와 센서 네트워크 질의의 차이점은 장기간 실행되며, 주기적으로 질의가 실행되는 것이다. 따라서, DURATION 절은 질의의 라이프타임(life time)을 기술하고, EVERY 절은 질의에 대한 응답률을 결정한다[3].

센서 네트워크 환경에서 센서의 압력을 감지해서 지역 내에 존재하는 동물들의 서식지 모니터링 시스템을 생각해 보자. 센서 네트워크 내의 각 지역 R과 S에서 동물들의 이동은 주 관심 대상이 될 수 있다. 두 지역의 각 테이블을 R과 S로 하고 그 속성은 〈nodeLocation, timeStamp, noiseValue〉로 정의하고, 같은 류의 동물들에 의해 감지되는 소리값(noise value)을 5% 범위 내에 둔다. 질의문을 작성해 보면 그림3.과 같은 조인 질의를 생각할 수 있다.

SELECT R.nodeLocation, R.timeStamp, S.noiseValue
FROM R, S
WHERE 0.95*R.noisValue 〈 S.noiseValue
AND S.noiseValue 〈 1.05*R.noiseValue

그림3. 조인 질의
Fig3. Join Query

그림3.의 질의에서와 같은 경우 "센서 네트워크의 어디(어느 노드)에서 조인 연산을 실행 할 것인가?" 하는 것은 질의의 성능과 관련해서 중요한 문제가 된다[1].

III. 질의의 구현(Implementation of Query)

3.1 집계 질의(Aggregation Query)

GROUP BY 또는 HAVING 절이 없는 집계 질의를 단순 집계 질의(simple aggregation query)라고하며, 센서 네트워크에서 가장 일반적인 질의이다[3]. 단순 집계 질의의 질의 플랜(query plan)은 통신 컴포넌트(communication component)와 연산 컴포넌트(computation component)로 구성되어있다[6]. 공간적으로 분산 되어있는 센서들로부터 필요한 데이터를 얻어 집계해야 하기 때문에 분산 노드에서 중앙 노드로 레코드를 전송하기위한 통신 전략이 필요하며, 이러한 질의 플랜을 통신 컴포넌트라고 한다. 또한, 중앙 노드에 집계된 레코드들의 연산을 담당하는 질의 플랜을 연산 컴포넌트라고 한다.

센서 네트워크에서 질의 처리 전략을 고려할 때 주 고려 대상 중 하나가 전력 문제이다. 통신컴포넌트와 연산 컴포

넌트를 적절히 조화 시킨다면, 중앙 노드로의 집계 문제를 중간 노드(intermediate node)에서 부분 집계(partial aggregate)를 통해 전송되는 메시지의 양을 줄인다면 전력 소모를 줄일 수 있을 것이다. 통신과 연산 컴포넌트를 통합한다면 다음과 같은 세 가지의 기법을 고려 해 볼 수 있을 것이다.

첫째, 직접 전송(direct delivery)인데, 가장 단순한 방법이기도 하다. 각각의 소스 센서 노드는 중앙 노드로 데이터 패킷을 보낸다. 연산은 모든 레코드가 수신되고 난후 중앙노드에서 실행된다.

둘째, 패킷 병합(packet merging)이다. 무선 환경에서 한 번에 큰 패킷을 전송하는 것 보다 작은 패킷을 여러 번 보내는 것이 통신 채널과 패킷 데이터의 헤더 부분의 로드를 고려한다면 더 많은 비용이 발생된다. 센서 레코드들의 크기는 대부분 작고, 적은 규모의 지역에서 많은 센서들이 동시에 전송하기 때문에 몇 개의 레코드를 병합(merge)해서 큰 레코드로 만들어 전송하는 것이 패킷 전송에 따른 오버헤드를 줄이는 방법이 될 것이다. 평균을 구하는 집계 연산자의 경우는 병합 방법이 전송될 데이터의 수를 줄이는 유일한 방법이기도 하다[13].

셋째, 부분집계(partial aggregation)이다. 각각의 중간 센서 노드는 최종 결과를 연산할 수 있는 충분한 능력을 갖춘 부분 결과를 연산하고, 이후에는 중앙 노드로 전송한다.

또한, 패킷 병합과 부분 집계 기법의 경우 질의 플랜의 연산 컴포넌트 실행을 위해 상호 센서 노드들 간의 동기화가 필요함을 알 수 있다. 중복 전송에 민감한 SUM과 AVG 같은 집계 연산자의 경우 오직 한번만 전송되어야 하며, MAX와 MIN같은 집계 연산자의 경우는 중복 전송 되어도 큰 문제가 발생하지는 않는다. 따라서, 이에 다른 전송 전략도 충분히 고려해야 할 것이다.

3.2 조인 질의(Join Query)

센서 네트워크 내의 어느 한 노드가 데이터를 저장하는 정적(non-stream) 데이터베이스는 연구의 대상에서 제외하고 스트림 데이터베이스에 대해서만 고려하기로 한다. 기존의 조인 질의에서는 연산 비용의 최소화가 주 고려사항이었다고 한다면, 센서 네트워크에서는 통신 비용이 가장 중요한 요소이다.

센서 네트워크에서 효율적인 조인 연산의 실행을 위한 접근 방법을 고려해 보자[2]. 2.3절에서 언급되었던 R과 S의 조인(R⋈S)에서 가장 단순한 방법은 지역 S에서 지역 R로 즉, 테이블 S의 내용을 R 지역으로 전송 후 조인하는 방법이 가장 단순한 방법이다. 또 다른 방법은 그림4.의 중심접근법(centroid approach)이다.

Q는 질의 소스 노드이다. 센서 네트워크에서 조인 연산의 대상이 되는 지역의 어떤 한 점을 C로 잡는다. 지역 P_c는 C를 중심으로 한 가장 작은 지역을 원형으로 정한 것이다. 지역 R과 S는 지역 P_c로 전송되고 P_c에서 조인 연산이 실행된다. 그 결과를 질의 소스 노드 Q에게 알려준다. 이외에도 센서 네트워크에서 통신 비용을 줄이는 다양한 방법을 고려해 볼 수 있다.

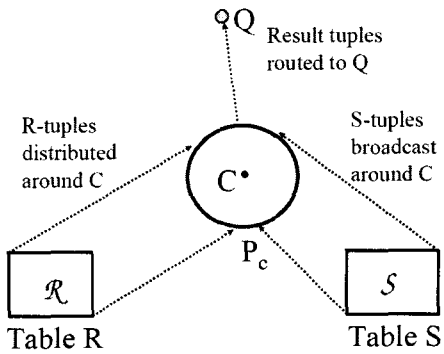

그림4. 중심접근법
Fig4. Centroid Approach

VI. 질의 최적화(Query Optimization)

질의 최적화를 고려 할 때 최선의 질의 플랜을 고려해야 한다. 그러나 임의의 질의에 대해 최선의 질의 플랜을 생성한다는 것은 쉬운 일은 아니다. 질의문에서 GROUP BY와 HAVING 절을 고려해보자. 예를 들어 센서 네트워크에서 어떤 기준 값 이하의 값은 제외하고 나머지 그룹의 평균을 구하는 연산 질의문을 그림5.과 같이 작성해 보자.

```
SELECT S.gid, AVG(S.value)
FROM SensorData S
GROUP BY S.gid
HAVING AVG(S.value)>Threshold
```

그림5. 최적화 질의 예제1
Fig5. exam1 for Query Optimization

이 질의에 대해서 두 가지 다른 질의 플랜을 생각할 수 있다. 하나는 각각 그룹에 대해서 플로루 블록(flow block)을 생성하는 것이고, 다른 하나는 다중 그룹에 의해 공유되는 플로루 블록을 생성하는 것이다[2]. 각각 플로루 블록을 생성하는 것은 같은 그룹의 센서 레코드를 집계하는 것이고, 경로 길이(path length)를 줄이는 것이기도 하다. 또한, HAVING 절에서 프레디킷을 사용하는 것은 집계 결과를 빨리 얻을 수도 있고 프레디킷의 선택도(selectivity)가 낮다면 더 많은 통신 비용을 줄일 수도 있다. 질의 최적화에서 최선의 질의 플랜을 작성하기 위해서 몇 가지 파라미터를 사용해야 한다. 한 가지는 센서의 물리적 위치의 오버랩(overlap)이다. 만약 센서가 물리적으로 인접한 단일 그룹에 속해 있다면 그 센서들을 묶어 각각의 플로우 블록을 생성하는 것이 유리하다. 왜냐하면 인접 센서들의 집계 비용은 비교적 적게 들기 때문이다. 그러나 다른 그룹들의 센서가 널리 퍼져있으면 모든 그룹의 공유를 통한 단일 플로우 블록을 생성하는 것이 더욱 효율적이다.

이번에는 조인 질의를 고려해보자. 플로우 블록의 연산부분은 집계 연산에서 고려할 필요가 없다. 왜냐하면 기존 질의 템플릿에 추가하는 것이 가능하기 때문이다. 조인 질의는 센서 네트워크에서 어떤 현상 등의 감시에 가장 일반적인 어플리케이션이다. 예를 들어 두 지역 R과 S에서 감지되는 모든 객체를 찾는 질의를 그림6.과 같이 작성해 보자.

```
SELECT oid
FROM SensorData S1, SensorData S2
WHERE S1.loc IN R AND S2.loc IN S
      AND S1.oid=S2.oid
```

그림6. 최적화 질의 예제2
Fig6. exam2 for Query Optimization

조인 질의는 데이터 감소가 가능한 광범위의 데이터를 다룬다. 따라서 조인 질의의 선택도(selectivity)에 의해 질의 결과로 표현되는 데이터의 크기가 작아질 수 도 커 질수도 있다. 만일 조인 질의가 결과 데이터를 증가시킨다면, 튜플을 전송하는 것보다 중앙노드에서 연산하는 비용이 더 많이 소모 될 수 도 있다. 또한, 적절한 카탈로그 데이터는 조인 질의의 선택도와 전송 노드를 결정하는 중요한 역할을 할 것이다.

V. 결론 및 향후 연구방안

센서 네트워크 환경의 센서들은 일반적으로 작은 크기의 대용량의 연속적인 스트림 데이터를 발생시킨다. 연속적인 데이터 스트림의 처리를 위해 효율적인 질의처리는 매우 중요하다. 이러한 상황에서 센서 데이터 스트림 처리를 위해서 센서 노드의 특성에 의한 기억장치와 같은 자원의 제약과 통신 즉, 데이터 전송에 따른 전력의 제약을 극복해야 한다. 특히, 센서 데이터베이스 질의에 대한 효율적인 구현을 위해 고려되어야 하는 중요한 이슈이다.

본 연구에서는 센서 네트워크 환경에서 발생되는 센서 데이터 스트림의 집계 질의와 조인 질의에 대하여 살펴보았고, 질의 최적화를 위한 몇 가지 방안도 살펴보았다. 질의 및 질의 최적화 방안은 센서 데이터 스트림을 효율적으로 다루어야 하는데 결국 전력 소모의 최소화라는 문제를 해결하는 것으로 초점이 맞추어 지게 됨을 알 수 있었다.

향후 연구방안은 효율적인 전력소모 관점에서 센서 데이터 스트림의 제한적인 상황을 최대한 고려해 효율적인 질의 처리 및 질의 처리의 최적화를 위한 연구를 진행하고자 한다.

참고문헌

[1] Wei Hong, Samuel Madden. Implementation and Research Issues in Query Processing for Wireless Sensor Networks. ICDE, 2004.

[2] Vishal Chowdhary, Himanshu Gupta, Communication-Efficient Implementation of Join in Sensor Networks. DASFAA, 2005.

[3] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: A tiny aggregation service for ad-hoc sensor networks. OSDI, 2002.

[4] R. Govindan, J. Hellerstein, W. Hong, S. Madden, M. Franklin, and S. Shenker. The Sensor Network as a Database. Technical Report, University of Southern California,

Computer Science Department, 2002.
[5] B. Badrinath, M. Srivastava, K. Mills, J. Scholtz, and K. Sollins, editors. Special Issue on Smart Spaces and Environments, IEEE Personal Communications, 2000.
[6] Y.Yao and J.Gehrke, Query Processing for Sensor Networks, CIDR, 2003.
[7] G. J. Pottie and W. J. Kaiser. Wireless integrated network sensors. Communications of the ACM, 2000.
[8] D. Estrin, R. Govindan, and J. Heidemann, editors. Special Issue on Embedding the Internet, Communications of the ACM, volume 43, 2000.
[9] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar. Next century challenges: Scalable coordination in sensor networks. MobiCom, 1999.
[10] J. Hill and D. Culler. A Wireless Embedded Sensor Architecture for System-level Optimization. Submitted for publication, 2002.
[11] R. Ramanathan and R. Rosales-Hain. Topology Control in Multihop Wireless Networks Using Transmit Power Adjustment. In Proceedings of the IEEE INFOCOM, 2000.
[12] D. L. Hall and J. Llinas, editors. Handbook of Multisensor Data Fusion. CRC Press, 2001.
[13] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh. Data Cube: A Relational Aggregation Operator Generalizing group-by, cross-tab, and sub-totals. Data Mining and Knowledge Discovery, 1997.

저 자 소 개

안 동 찬

2004년 8월 : 서강대학교 컴퓨터학과 공학박사 수료
1996년 ~ 1999년 : SK텔레콤(주) 정보기술연구원
2002년 ~ 현재 : 안산공과대학 디지털미디어과 교수
관심분야 : XML 데이터베이스, 데이터스트림처리