기호공간에서 이동객체 스트림 데이터의 연속 시공간 셀프조인 질의

Continuous Spatio-Temporal Self-Join Queries over Stream Data of Moving Objects for Symbolic Space

황 병 주* 이 기 준**
Hwang Byung Ju Li Ki Joune

요 약 시공간 조인은 이동객체와 같이 시공간의 특성을 가지는 데이터를 처리할 때 요구되는 중요한 연산자로, 이동객체들의 움직임을 분석하거나 이동객체들의 시공간적 패턴을 찾는 것과 같이 다양하게 활용된다. 현재까지 실외공간에서의 시공간 조인 질의에 관한 연구는 많이 진행되어왔다. 최근에는 실내측위기술이 발전함에 따라 실외뿐만 아니라 실내에서도 다양한 위치기반 서비스가 점진적으로 제공되고 있으며, 특히 이동객체를 중심으로 다양한 응용 서비스들을 필요로 하게 된다. 하지만 실내공간에서의 시공간 조인에 관한 연구는 아직 전무하다. 본 논문에서는 실내공간에서 실시간으로 갱신되는 이동객체에 대한 연속 시공간 셀프조인 질의와 질의처리 방법론을 제안하였다. 연속 시공간 셀프조인 질의는 주어지는 특정 시간과 공간의 조건을 만족하는 모든 쌍들을 시간이 지남에 따라 지속적으로 갱신하는 질의이다. 본 논문에서는 방이나 복도와 같이 특정한 기호를 중심으로 이동객체의 위치를 표현하며 이러한 특징을 가지는 공간을 기호공간이라 한다. 그리고 방대한 스트림데이터를 효과적으로 필터링하고 관리하기 위한 후보쌍 버퍼 테이블이라는 자료구조와 이를 활용한 질의처리 방법론을 제안하였으며 실험을 통해 타당성을 검증하였다.

키워드 : 기호 공간, 연속 시공간 셀프 조인 질의

Abstract Spatio-temporal join operators are essential to the management of spatio-temporal data such as moving objects. For example, the join operators are parts of processing to analyze movement of objects and search similar patterns of moving objects. Various studies on spatio-temporal join queries in outdoor space have been done. Recently with advance of indoor positioning techniques, location based services are required in indoor space as well as outdoor space. Nevertheless there is no one about processing of spatio-temporal join query in indoor space. In this paper, we introduce continuous spatio-temporal self-join queries in indoor space and propose a method of processing of the join queries over stream data of moving objects. The continuous spatio-temporal self-join query is to update the joined result set satisfying spatio-temporal predicates continuously. We assume that positions of moving objects are represented by symbols such as a room or corridor. This paper proposes a data structure, called Candidate Pairs Buffer, to filter and maintain massive stream data efficiently and we also investigate performance of proposed method in experimental study.

Keywords : symbolic space, continuous spatio-temporal self join query

1. 서 론

GIS(Geographic Information System) 분야의 연구와 발전으로 인하여 우리 사회는 다양한 편의와 서비스를 제공받고 있다. 특히, 네비게이션 분야에서 실제 우리 생활과 밀접한 서비스가 직접적으로 제공되고 있다. 예를 들어 사람들은 비행기나 선박, 혹은 자동차를 이동수단으로 하여 서울에서 부

†본 연구는 두뇌한국21사업과 국토해양부 첨단도시기술개발사업 - 지능형국토정보기술혁신사업과제의 연구비지원(07국토정보C04)에 의해 수행되었습니다.

* 부산대학교 컴퓨터공학과 박사과정, bjhwang@krihs.re.kr
** 부산대학교 컴퓨터공학과 교수, lik@pnu.edu(교신저자)

산까지 이동하는 최단 경로나 거리, 소요시간 등의 정보를 취득하길 원하였고, 이러한 요구가 하드웨어적 발전과 함께 충분한 인프라가 구축되면서 실현되었다.

이렇듯 기존의 GIS 서비스는 비행기나 자동차가 이동하는 실외 공간 중심으로 데이터 구축이 주를 이루었으나 최근에는 대형 쇼핑몰이나 컨벤션 센터와 같이 대규모 복합 건물이 늘어나게 되고, 실내 위치기반 서비스 인프라가 구축되면서 사람들은 건물 내부에서도 다양한 서비스의 필요성을 느끼기 시작하였고, 이에 따라 실외공간과는 다른 실내공간[1,2]에 대한 연구의 필요성이 대두되었다. 실제로, 현재 국내에서 진행 중인 실내공간인지(ISA) 프로젝트에서는 실내공간에 대한 기초이론을 개발하고, 핵심기술, 시스템 구축 및 서비스를 제공하기 위한 프로젝트가 진행 중이며 세계적으로도 실내공간에서 서비스를 제공하기 위하여 이동 객체들에 대한 질의와 관련한 연구가 진행되었는데, 실내공간에서 이동객체의 궤적에 대한 인덱스[3]와 실내공간에서 이동객체에 대한 연속질의[4]가 그 예이다.

실내공간에서 제공되는 질의 중, 연속 시공간 조인은 이동객체와 같이 시공간 특성을 가지는 데이터를 처리할 때 요구되는 주요 연산자로, 이동객체들의 움직임을 분석하거나 시공간적 패턴을 찾는 것과 같이 다양하게 활용된다. 이러한 질의는 다른 질의를 위한 필터 단계로서 활용될 수도 있는데, 예를 들어 일정시간동안 인접하게 함께 이동하는 객체들(어머니-아기, 경호원-고용인, 연예인-파파라치)을 찾는데 활용될 수 있다.

하지만 이러한 질의가 비록 유용하게 활용될 수 있지만 연속 시공간 셀프조인 질의를 처리하기 위해서는 상당한 비용이 소모된다. 연속 시공간 셀프조인 질의는 주어지는 특정 시간과 공간의 조건을 만족하는 모든 객체 쌍들을 시간이 지남에 따라 지속적으로 갱신하는 질의이다. 예를 들어 스트림환경에서 현재의 이동객체들이 위치한 공간끼리 겹치는(overlap) 이동객체쌍을 찾는 질의가 연속 공간 셀프조인이며, 임의의 시간 영역 동안 겹치는 이동객체쌍을 찾는 질의가 연속 시공간 셀프조인이다. 이 때, 공간셀프 조인을 처리하기 위해서 모든 각각의 이동객체와 나머지 이동객체간의 겹치는 쌍을 찾는 것은 굉장히 많은 비용이 소모된다.

수천 개 이상의 스트림데이터를 연속질의[5]로 처리하기 위해서는 데이터스트림 관리 시스템이 효과적이지만, 데이터를 실시간으로 처리하기 위하여 질의처리를 위한 전체 데이터공간을 윈도우라는 제한된 영역만으로 한정하는 데이터 스트림 관리 시스템의 구조적 제약조건[6]으로 인하여, 질의처리를 위해 필요한 시간동안의 모든 데이터를 저장할 수 없기 때문에 조인 질의를 실시간으로 처리하는 것은 기존의 방법론으로는 해결되지 않는다. 윈도우의 크기를 무제한 늘리게 될 경우 엄청난 양의 데이터에 대하여 소모적인 연산인 연속 시공간 조인 질의를 처리하게 되면, 엄청난 질의 비용으로 인하여 결국, 실시간으로 처리하지 못하게 된다.

본 논문에서는 실내공간에서 활용도가 높을 것이라 예상되는 조인질의를 제시하고, 데이터 스트림 관리 시스템의 구조적 한계로 인하여 기존 공간모델에서 사용하는 방법론으로는 실시간으로 처리할 수 없었던 데이터 스트림에서의 조인질의를 기호공간에서 처리하기 위한 질의 처리 방법론을 제안한다.

2장에서는 실내공간모델인 기호공간과 데이터 스트림 관리 시스템의 일반적인 구조를 기술하고 이동객체들의 궤적과 관련된 연구들에 대해 기술한다. 3장에서는 실내공간에서 요구되는 질의 정의 및 방법론에 대해 기술한다. 4장에서는 논문에서 제안한 시스템을 구현하고 분석한다. 마지막으로 5장에서는 결론 및 향후 연구 방향을 기술한다.

2. 관련 연구

2.1 기호 공간

실내 공간은 지금까지 GIS의 주된 대상이 되었던 실외 공간 모델들과는 다른 특성을 가지고 있다.

일반적으로 실외공간은 그림 1과 같이 비행기가 이동하는 유클리드 공간과 자동차가 이동하는 도로 네트워크 공간으로 모델화된다. 반면, 실내공간은 Symbolic Location 모델[7]과 Topology 모델[8]과 같이 기호 공간(Symbolic Space)으로 모델화 한다.

기호공간은 x,y,z로 위치를 표현[9]하는 실외공간들과는 다르게 방 번호나 좌석번호 등의 셀 식별자를 사용하여 위치를 식별하며, 식별된 공간들은 그림 2와 같이 그래프로 표현된다.

실내공간을 기호 공간으로 모델화 하는 주된 이유 중 하나는 현재의 기술적 한계와 비용문제로 인하여 완벽한 위치 식별을 위한 충분한 하드웨어적

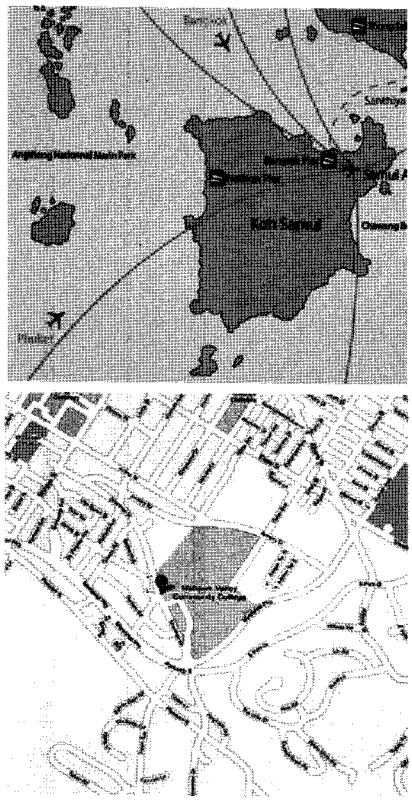

그림 1. 항로(유클리드 공간)와 도로(도로 네트워크 공간)

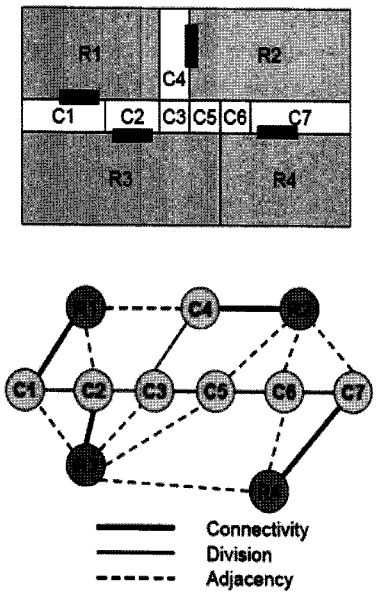

그림 2. 기호공간의 예제

설비를 갖추기가 힘들기 때문이다. 이러한 문제로 인하여 실내공간에서 이동하는 객체의 정확한 위치 추적이 불가능하지만, 실내공간을 기호공간으로 모델화함으로서 이동객체가 어느 셀에 존재하는지에 대한 데이터를 획득이 가능하여 다양한 서비스를 제공할 수 있다.

2.2 데이터스트림 관리 시스템

스트림 데이터와 같이 빠른 속도로 많은 양의 데이터를 실시간으로 분석하여 연속질의를 처리하기 위해서는 데이터스트림 관리 시스템이 효율적이다. 데이터스트림 관리 시스템은 질의를 먼저 등록한 후, 데이터에 의해 질의의 결과값을 반환하는 방법이다.

데이터베이스 관리 시스템에서는 스트림 데이터의 특성상 매 시간 입력되는 데이터의 양이 방대하기 때문에 모든 데이터를 검색할 수 없고, 모든 데이터가 유용하지도 않으므로, 현재의 상황에 대한 연속질의에 효과적으로 처리하기 위하여 질의의 공간 크기를 제한하는 윈도우(window)의 개념을 사용한다. 윈도우에 대하여는 다양한 연구가 진행되어 왔는데 본 논문에서는 새로운 데이터가 도착하면 윈도우의 시작점과 끝점이 나란히 앞으로 혹은 뒤로 나아가서 가장 오래된 데이터를 삭제하는 구조인 슬라이딩 윈도우를 사용한다.

2.3 궤적조인에 대한 연구들

일정한 시간동안 유사한 거리공간에 위치한 이동객체 쌍을 구하는 질의는 과거 이동객체의 궤적조인을 통한 이동객체쌍을 찾는 질의와 유사하다.

과거 일정 시간 동안의 유사한 이동 궤적을 가지는 이동객체들에 대한 조인 질의는 상당한 IO 비용이 소모되는데, 최악의 상황에서는 모든 이동객체들의 궤적에 대하여 조인 연산을 수행해야하는 상황이 발생하게 된다. 이러한 질의를 처리하기 위해서는 먼저 각 궤적들간의 유사도를 비교하기 위한 연산을 수행하기 전에 비교 대상이 되는 궤적들의 후보군을 먼저 선정하여 유사도 비교를 위한 이동객체 궤적들의 수를 줄이는 것이 필요하다. 그러한 방법들 중 하나가 기호표기를 사용하여 궤적들에 대한 조인이다[10].

지속적으로 삽입되는 스트림 데이터의 구조에서 과거 2, 3시간동안 유사한 이동궤적을 가지는 이동

객체의 쌍들을 찾는 연속 질의는 위의 연구보다 더 많은 연산 비용이 소모된다. 이러한 이동객체의 데이터는 너무나 방대하기 때문에 과거의 이동객체의 궤적은 보조 기억 장치에 저장하게 되는데, 이러한 구조에서 이동객체의 유사도에 따른 궤적조인 lower-bounding 거리 계산 비용과 이동객체의 궤적에 접근하기 위한 IO 비용의 합으로 이루어진다. Petko Bakalov는 이동객체의 궤적을 그리드하게 나누고 인덱스를 사용하여 IO비용을 줄이는 방법론을 제안하였다[11].

실내공간모델에 관한 연구와 데이터스트림에 대한 질의, 그리고 이동객체의 쌍을 찾는 궤적 질의 등의 각각에 대한 연구는 이미 진행되어 왔으며 본 논문은 실내공간과 스트림을 처리하는 환경에서 이동객체 쌍을 찾는 질의에 대한 질의를 정의하고 방법론을 제안한다.

3. 질의정의 및 방법론 제안

현재 실내공간에서 이동객체의 위치를 파악하기 위해서는 일반적으로 RFID나 블루투스 기술을 사용한다. 본 논문에서도 실내공간의 위치와 이동객체의 움직임을 RFID 리더기와 태그를 사용하여 획득한다고 가정한다. 본 논문에서 설계한 시스템에서는 이동객체가 RFID리더기의 탐색범위 안으로 들어오거나 나갈 때 데이터 스트림 관리 시스템으로 데이터를 전송하게 되면 현재의 이동객체가 탐지된 RFID 리더기를 데이터로 구축하게 된다.

본 논문의 기호공간 모델에서는 이동객체의 위치는 셀 아이디로 표시한다. 기호공간에서 셀은 하나의 방과 매핑되며, 셀과 셀과의 연결성은 문에 의해 결정된다. 하나의 방은 하나이상의 RFID 리더기들과 매핑되며, 매핑된 RFID 리더기들은 하나의 방을 완벽하게 인식할 수 있다고 가정한다. 즉, 위에서 저장된 이동객체가 탐지된 RFID 리더기 레코드는 기호공간의 셀로 매핑되어 질의처리를 위한 데이터로 구축되게 된다.

이 장에서는 기호공간에서 사용될 수 있는 이동객체 데이터의 연속 시공간 셀프조인 질의를 정의하고, 정의한 질의를 스트림데이터에서 처리할 때 발생하는 문제점과 그에 따른 방법론을 기술한다. 기호공간에서는 시간 속성과 거리 속성을 이용하여 아래와 같은 질의를 요구할 수 있다.

- 임의의 시간동안 같은 공간에 있는 이동객체 쌍들을 구하여라
- 임의의 시간동안 두 이동객체가 임의의 거리 범주 내에서 이동하는 이동객체 쌍들을 구하여라

```
Select IStream( a.moID, b.moID, cellID, updateTime, '+' )
From  ( Select moID, cellID, LASTEST(EnterTime), EnterTime
        From DataStream'
        Group by moID ) a , b [RANGE Window]
Where a.cellID = b.cellID and
        MINTIME < ( CURRENT_TIME - a. EnterTime ) and
        ( CURRENT_TIME - a. EnterTime ) <= MAXTIME
and
        MINTIME < ( CURRENT_TIME - b. recentTime )
and
        ( CURRENT_TIME - b. EnterTime ) <= MAXTIME
Union All
Select DStream( a.moID, b.moID, cellID, updateTime, '-' )
From  ( Select moID, cellID, LASTEST(LeaveTime), LeaveTime
        From DataStream'
        Group by moID ) a , b [RANGE Window]
Where a.cellID = b.cellID and
        MINTIME < ( CURRENT_TIME - a. LeaveTime ) and
        ( CURRENT_TIME - a. LeaveTime ) <= MAXTIME
and
        MINTIME < ( CURRENT_TIME - b. LeaveTime ) and
        ( CURRENT_TIME - b. LeaveTime ) <= MAXTIME
```

그림 3. CQL 문법을 이용한 질의 예제

3.1 연속 시공간 셀프조인 질의(같은 공간)

첫 번째 질의처럼 임의의 시간 영역에 대하여 같은 위치의 객체 쌍을 찾는 질의를 데이터스트림 관리 시스템에서 처리하기 위하여 [그림 3]과 같이 STREAM Project에서 제안한 CQL (Continuous Query Language) 문법의 형태로 표현하였다[12].

그림 4는 질의의 예로서 10분내지 30분 이내로 같은 공간에 머무르는 쌍들을 찾는 질의이다. 그림 5는 각각의 이동객체들에 대한 미래궤적의 예제이며 위의 질의의 조건을 만족하는 상황을 표현한 그림이다.

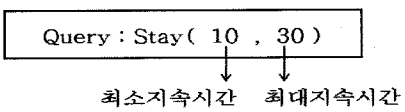

그림 4. 조인질의의 예

그림 5와 같이 0초에 연속질의가 등록이 되면, MO2와 MO3는 20초가 되는 시점부터 질의의 조건

을 만족함으로 <표 1>과 같이 시간에 따라 결과 값을 사용자에게 전송하게 된다. 표의 "상태"는 현재 두 이동객체간의 상태를 나타내는 것으로, '+'는 두 이동객체가 질의의 조건을 만족했음을 의미하고, '-'는 한 개 이상의 이동객체가 다른 공간으로 이동하였음을 의미한다. 타임스탬프 0초에 질의를 등록하면 20초가 되는 시점에 MO2와 MO3가 질의의 조건을 만족하여 B셀에서 함께 머무르고 있음을 결과 값으로 받게 되고, 22초가 되는 시점에서 MO3의 위치가 B셀에서 A셀로 바뀌게 되어, 더 이상 질의의 조건에 만족하지 않으므로 변화된 상태 결과 값을 전송한다.

연속 조인 질의를 등록한 사용자는 조건을 만족하는 두 이동객체의 쌍과, 더 이상 결과값을 만족하지 않는 이동객체 쌍에 대한 결과를 받게 된다.

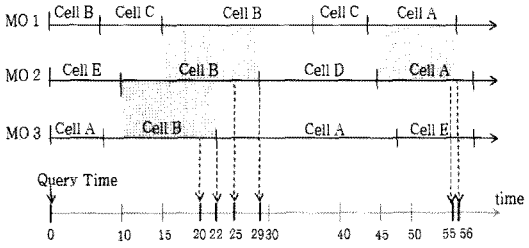

그림 5. 이동객체들의 미래궤적에 대한 예제와 결과값

표 1. 질의의 결과값

이벤트 시간	이동객체 ID_1	이동객체 ID_2	셀ID	상태
t=20	MO_2	MO_3	B	+
t=22	MO_2	MO_3	B	-
t=25	MO_1	MO_2	B	+
t=29	MO_1	MO_2	B	-
t=55	MO_1	MO_2	A	+
t=56	MO_1	MO_2	A	-

3.2 연속 시공간 셀프 조인 질의(인접한 공간)

두 번째 질의는 첫 번째 질의의 확장으로서 두 이동객체의 거리가 N이하를 유지하며 이동하고 있는 객체들의 쌍들을 찾는 질의이다. 그림 4의 질의와는 다르게 거리 값을 인수로 추가하여 질의하여야 한다. 첫번째 질의는 두 이동객체의 거리가 0인 특수한 형태라고 할 수 있다. 그림 7은 구체적인 질의 예로서 10분 내지 30분 동안 두 이동객체의 거리가 1을 넘지 않는 쌍들을 찾는 질의이다.

그림 7의 질의를 처리하기위해서는 기호공간에서 노드들의 거리를 알기 위하여 그림 8과 같은 그래프 정보가 필요하다. 각 에지의 가중치는 두 이동객체의 거리를 표현하기 위해 사용한다.

그림 6은 각각의 이동객체들의 미래궤적의 예제이며, <표 2>은 그에 따른 결과 값의 예제이다.

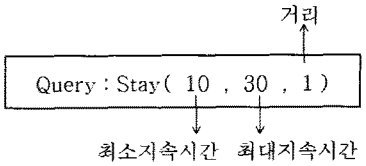

그림 6. 질의 예제

그림 6의 질의 예를 살펴보자. 타임스탬프 0초에 연속질의가 등록이 되면, MO1과 MO2는 20초가 되는 시점부터 질의의 조건을 만족함으로 그림 <표 2>과 같이 시간에 따라 결과 값을 사용자에게 전송하게 된다. 29초가 되는 시점에 MO2의 위치가 바뀌게 되지만 두 객체의 거리가 1이므로 질의조건을 계속 만족하게 된다.36초가 되는 시점에 MO1와 MO2의 위치가 각각 C셀과 D셀로 바뀌어 두 이동객체의 거리가 2가 되는 순간 질의의 조건을 더이상 만족하지 않으므로 사용자에게 결과 값을 보내게 된다.

표의 "상태"는 현재 두 이동객체간의 상태를 나타내는 것으로, '+'는 두 이동객체가 질의의 조건을 만족하였음을 의미하고 '-'는 더 이상 두 이동객체의 거리가 질의 조건을 만족하지 않음을 의미한다. '*'는 질의 조건은 계속 만족하지만 이동객체의 위치가 변경되었음을 의미한다.

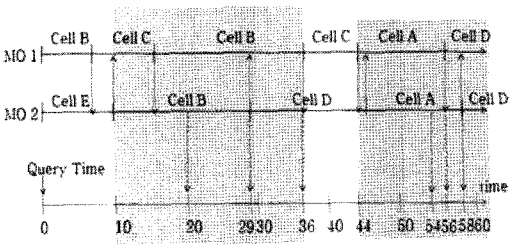

그림 7. 이동 객체들의 미래궤적에 대한 예제와 결과값

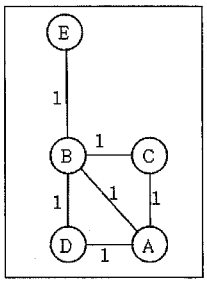

그림 8. 거리 가중치 그래프

표 2. 질의에 대한 결과값의 예제

이벤트 시간	이동객체 ID_1	이동객체 ID_2	셀 ID1	셀 ID2	상태
t=20	MO_1	MO_2	B	B	+
t=29	MO_1	MO_2	B	D	*
t=36	MO_1	MO_2	C	D	-
t=54	MO_1	MO_2	A	A	+
t=56	MO_1	MO_2	A	D	*
t=58	MO_1	MO_2	D	D	*

3.3 데이터스트림 관리 시스템의 구조적 문제

그림 3의 CQL 문법을 보면 질의를 처리하기 위하여 대상이 되는 범위를 윈도우로 정의하는 것을 볼 수 있다. 그러나 데이터 스트림 관리 시스템에서는 시스템 성능상의 문제로 윈도우를 무한히 크게 설정할 수가 없다.

윈도우내에 원하는 데이터가 모두 존재 한다면 정확하게 질의가 처리되겠지만 필요한 데이터가 윈도우의 사이즈보다 클 경우 정확한 결과 값을 도출하지 못하게 된다. 스트림 데이터는 매초마다 수 천개 이상씩 삽입되기 때문에 윈도우의 크기를 과도하게 늘리면 시스템의 과부하를 초래하고, 시스템 정지라는 심각한 문제를 발생시킬 수 있으므로 데이터를 처리하는 윈도우의 크기는 작게 설정하여야 하지만, 상대적으로 필요한 데이터는 너무 크기 때문에 정확한 결과 값을 도출하지 못하는 것이다.

3.4 업데이트 버퍼와 후보쌍 버퍼를 이용한 방법론

데이터 스트림 관리 시스템의 목적이 현재 이동객체들에 대한 연속 질의를 효과적으로 처리하기 위하여 설계된 것으로, 시간영역에 대하여 영역 질의를 지속적으로 조인하는 것은 데이터스트림 관리 시스템의 기본 목적과는 다르게 일정 시간동안 이동객체들의 궤적을 저장해야만 정확한 결과 값을 도출할 수 있으므로 일반적인 데이터 스트림 관리 시스템에서는 효율적으로 처리할 수가 없었다. 시간영역의 조인 질의를 처리하기 위해서는 일반적인 데이터 스트림 관리 시스템에서는 무한한 질의의 시간 영역 동안 모든 이동객체의 궤적을 메모리에 유지해야 하는데 이러한 문제를 해결하기 위하여 업데이트 버퍼 테이블과 후보쌍 버퍼 테이블을 이용한 질의 처리 방법을 제안한다.

3.4.1 질의 처리를 위한 전체 구조

그림 9는 질의처리를 위한 전체 구조를 설명한 그림이다. 제안한 방법에서는 모든 이동객체의 궤적을 저장하여 매 타임스탬프마다 질의시간영역들을 조인하는 것이 아니라, 연속 질의가 등록된 순간부터 질의의 결과 값이 될 수 있는 이동객체의 쌍들

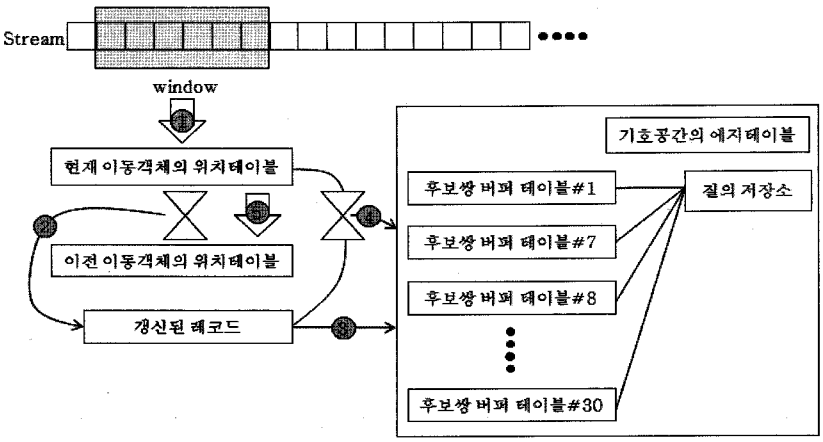

그림 9. 윈도우 사이즈에 따른 질의 처리의 문제 비교

을 데이터 스트림 관리 시스템에서 지속적으로 관리하여 처리한다. 즉, 결과 값이 될 수 있는 이동객체의 쌍들을 매 타임스탬프마다 버퍼 테이블에 등록을 하고, 등록된 후보 쌍 중 상태가 변화하여 더 이상 질의의 조건을 만족하지 않는 객체 쌍들을 탈락시키면서, 최종적으로 질의의 시간 영역동안 탈락되지 않은 이동객체 쌍들을 결과 값으로 반환하게 된다.

3.4.2 질의에 대한 구체적인 처리과정

본 논문에서 제안하고 있는 방법론으로 데이터스트림에 대하여 질의의 시간영역에 따라 무한히 커지는 데이터들의 엄청난 조인비용으로 인하여 실시간으로 처리하지 못하는 문제를 해결하고자 한다. 데이터 스트림 관리 시스템에 추가적인 버퍼테이블을 관리하여 비용을 줄이고, 결과적으로 시간영역에 무관하게 실시간으로 질의처리가 가능하도록 한다. 질의 처리를 위하여 메모리에 지속적으로 관리하여야 하는 테이블은 크게 2가지로, 이전 타임스탬프의 이동객체의 위치를 저장하고 있는 버퍼 테이블과 질의의 결과 값이 될 수 있는 후보쌍들을 저장하는 버퍼 테이블이다.

이 버퍼테이블을 이용하여 아래의 알고리즘을 활용하면 조인질의의 결과값을 도출할 수 있는데, 후보쌍 버퍼에는 질의를 만족할 가능성이 있는 쌍들을 저장하고 더이상 질의를 만족할 가능성이 사라졌을 때 후보쌍에서 삭제하는 구조를 지닌다.

- 윈도우에서 현재이동객체의 위치를 뷰 테이블로 생성한다.
- 직전 타임스탬프에서 저장된 이동객체의 이전 위치 버퍼 테이블과 현재위치를 조인하여 갱신된 이동객체 레코드를 선별한다.
- 후보쌍에서 선별된 이동객체를 조인하여 업데이트를 한다. 업데이트 된 후보쌍들은 갱신된 위치와 지속 시간에 따라 상태를 변화시킨다.
- 2번에서 선별된 이동객체들과 현재위치를 조인하여 새로운 후보쌍들을 추가한다.
- 직전 이동객체의 위치를 현재 이동객체의 위치로 바꾼다.

<표 3>는 후보쌍 버퍼 테이블에 있는 이동객체 쌍과 상태에 대한 예제이다. 후보로 등록된 레코드들은 레코드에 포함된 이동객체의 위치가 갱신될 때마다 두 이동객체간의 거리와 질의 시간을 비교하여 후보에서 탈락되어 삭제할지, 혹은 질의조건을 만족하여 사용자에게 갱신된 결과값을 전송할지를 결정한다. 또한 질의조건을 만족한 레코드 역시 이동객체 위치가 바뀔 때마다 삭제될지, 갱신될지를 결정한다.

Algorithm 1: 후보쌍 버퍼 테이블의 갱신을 위한 알고리즘

```
1  while isAliveQuery do
2      List currentMOList = getCurrentMOList(DATASTREAM);
3      if pastMOList != null then
4          updatedRecords =
           getUpdatedRecords(currentMOList,pastMOList);
5      else
6          updatedRecords = currentMOList;
7      end
8      if updatedRecords != null then
9          updateCandidatePairList(updatedRecords);
10         List newCandidatePair =
           createNewCandidatePair(updatedRecords, currentMOList);
11     else
12         if newCandidatePair != null then
13             updateCandidatePairList(updatedRecords);
14         end
15     end
16     pastMOList = currentMOList;;
17 end
```

그림 10. 후보쌍 버퍼를 활용한 알고리즘

표 3. 질의에 대한 결과값의 예제

이동객체 ID1	이동객체 ID2	셀 ID1	셀 ID2	상태	입력 시간
MO1	MO42	CE1	CE1	+	06:40
MO1	MO32	CE1	CE1	-	06:43
MO3	MO56	CE89	CE89	-	06:44
MO7	MO34	CE36	CE36	-	06:43
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
MO99	MO423	CE25	CE25	+	06:40

4. 실험 평가

4.1 실험 설정 및 실험 방법

이 장에서는 실내공간에서 인접한 거리를 유지하고 있는 두 이동객체에 대하여 임의의 시간 영역동안 조인 질의를 처리하기 위하여 실내 공간을 기호공간으로 구축하여 질의에 대한 비교 실험을 하였

다. 실험에 사용한 실내 공간 데이터는 가상으로 생성한 그림 11(a)의 가상 건물 데이터와 실제 컨벤션 센터인 킨텍스 데이터로 생성한 그림[그림 11](b)의 실제 건물 데이터를 사용하였다. 각 건물 데이터는 위치를 셀 식별자로 나타내는 기호공간으로 모델화 하였다. 기호공간은 문으로 연결된 각 셀(방)들의 그래프를 가지는데, 그래프의 노드는 셀을, 에지는 셀과 셀 사이를 연결하는 문을 가리키며, 실험에 사용된 노드와 에지의 수는 <표 4>와 같다.

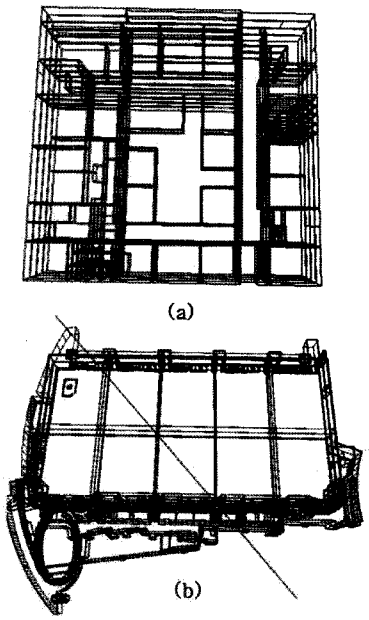

그림 11. (a) 가상건물모델
(b) 실제킨텍스건물모델

표 4. 실험에 사용한 건물 데이터

	가상공간	킨텍스
노드	71	799
에지	83	1167
각 에지의 가중치	1	1

기호 공간 안에서 이동하는 객체에 대하여 다음의 제약사항을 설정하였다.

- 이동 객체는 에지를 통하여 다른 노드로 이동이 가능하다
- 이동객체는 이동할 때 자유롭게 가감속을 한다.
- 전체 이동객체 중 10퍼센트는 인접한 거리를 유지하며 이동한다.

실험 방법은 본 연구에서 제안한 질의 처리방법에 따라 데이터 스트림 관리 시스템의 일부 모듈을 구현하여 위에서 지정한 기호공간과 이동객체의 움직임에 따라 질의의 조건을 달리하여 <표 5>와 같이 세 가지 실험 군으로 분류하여 실험을 하였다.

표 5. 실험군 정의

#	실험군
1	기호공간에 따라 이동객체의 수를 달리하여 실행시간을 비교하여 본다.
2	이동객체의 수에 따라 질의의 거리를 달리하여 실행시간을 비교하여 본다.
3	이동객체의 수에 따라 질의의 최소 지속 시간을 달리하여 실행시간을 비교하여 본다.

4.2 실험 결과

4.2.1 실험1. 기호공간에 따른 질의 처리의 수행 시간 비교

그림 12는 킨텍스공간과 가상공간을 이동하는 이동객체의 수를 달리하여 질의에 대한 실행 시간을 비교한 실험결과이다. 질의에서 사용한 거리는 0이며, 최소지속시간은 30초로 설정하였다. 실험결과 노드 갯수가 적은 가상공간 모델이 노드 갯수가 많은 킨텍스 모델에서의 질의처리를 위한 실행 시간보다 비교적 짧은 것을 알 수 있다.

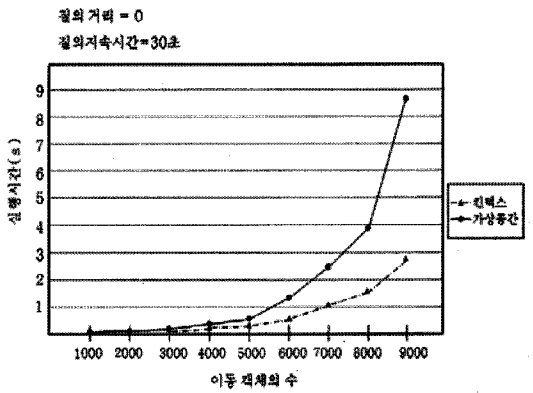

그림 12. 기호공간에 따른 질의 처리의 실행 시간 비교

이것은 노드가 많을수록 하나의 노드에 머무르는 이동객체의 수가 적어져서 후보쌍으로 만들어지는

이동객체의 쌍들이 적어지기 때문에 그만큼 실행 속도가 빨리 지기 때문이다.

4.2.2 실험2. 질의의 거리에 따른 질의 처리의 실행 시간 비교

그림 13은 이동객체의 수에 따라 질의의 거리를 달리하여 질의에 대한 실행 시간을 비교한 실험결과이다. 질의에서 사용한 공간은 가상공간에 대한 기호공간이며, 최소 지속 시간은 30초로 설정하였다. 실험결과 질의의 거리를 크게 설정할수록 실행시간이 늘어남을 알 수 있었다. 이러한 결과는 질의의 거리를 크게 설정하게 되었을 때, 새로운 이동객체의 후보쌍들을 선정하는 과정에서 더 많은 이동객체쌍들에 대하여 질의의 조건을 만족하는지를 처리해야하기 때문에 발생한다. 그렇기 때문에 다른 공간으로의 접근성이 높은 큰 홀이나 복도를 가지는 기호공간에서는 질의의 거리를 작은 값으로 설정하여 질의하여도 실행시간이 커짐을 유추해 볼 수 있다.

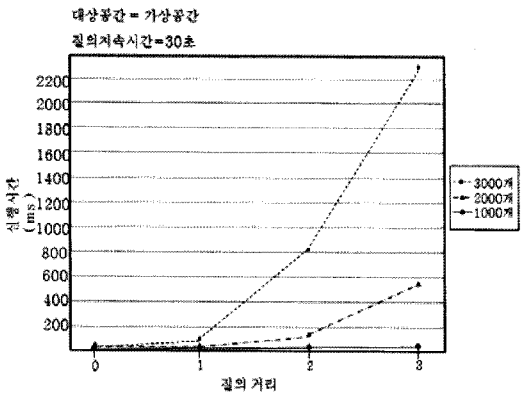

그림 13. 질의의 거리에 따른 질의 처리의 실행 시간 비교

4.2.3 실험3. 질의의 지속시간에 따른 질의 처리의 실행 시간 비교

그림 14는 이동객체의 수에 따라 질의의 지속시간을 달리하여 질의에 대한 실행 시간을 비교한 실험결과이다. 질의에서 사용한 공간은 가상공간에 대한 기호공간이며, 질의에서 사용한 거리는 0로 설정하였다. 실험결과 질의의 지속시간과 무관하게 실행시간이 크게 변화하지 않는 것을 알 수 있었다. 본 논문에서 제안한 질의 처리 방법론은 과거 전체 이동객체의 궤적에 따른 조인이 아니라 현재에 시점에서 바로 이전의 이동객체들 간의 비교를 통하여 지속적으로 질의의 결과값이 될 수 있는 후보쌍들을 선정하고, 탈락시키기 때문에 질의의 지속시간과는 무관하게 질의를 처리 할 수 있었다. 즉, 데이터 스트림 관리 시스템의 윈도우 안에 질의 시간동안의 모든 데이터가 존재하지 않더라도 질의를 처리할 수 있음을 의미한다.

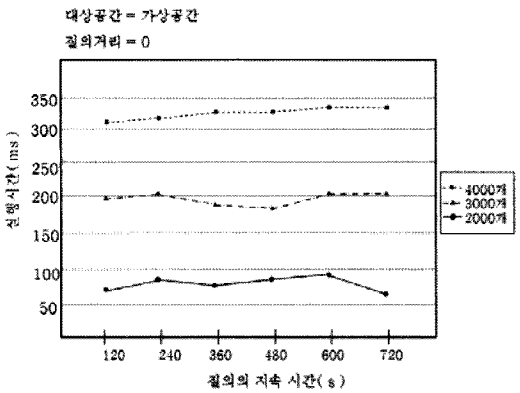

그림 14. 질의의 지속시간에 따른 질의 처리의 실행 시간 비교

4.3 실험 결과 분석

위의 세 가지 실험은 충분한 크기의 타임스탬프에 의하여 질의를 처리한 것으로, 모든 질의 결과에 대하여 100퍼센트의 정확도를 확인하였다.

하지만 매 타임스탬프마다 질의를 처리해야하는 스트림 환경에서 질의를 처리하기 위한 시간이 오래 걸린다는 것은, 그만큼 다음의 데이터 스트림에 대한 처리를 생략하거나 일부 데이터는 버려지게 되어 매 타임스탬프마다 질의를 처리하기 위한 후보쌍 선별의 누락시키게 된다. 결과적으로 질의의 정확도를 저해하는 요인이 된다. 궤적의 유사도에 의한 질의처리 방법[13,14]이 아닌 매 시간마다 정확하게 질의 조건에 매치되는지를 판단하여 처리하는 구조에서 데이터 스트림의 누락은 심각한 오결과를 도출하게 된다.

위의 세 가지 실험으로 본 논문에서 제안한 질의 처리 방법이 기호 공간의 모델과 질의의 거리, 이동객체의 수에 따라 실행시간이 달라짐을 알 수 있었다. 조인질의가 상당한 비용이 소모되는 질의이기 때문에, 이동객체의 수가 과도하게 늘어나거나 질의의 거리를 늘리게 될 때에는, 실행시간이 오래

걸려 정확한 결과 값을 도출하지 못함을 유추할 수 있다.

5. 결론 및 향후 연구과제

본 논문에서는 새로운 연구분야인 실내공간에서 서비스될 수 있는 기호공간에서 이동객체 스트림 데이터의 연속 시공간 셀프조인 질의의 필요성을 제시하고, 정의하였다. 이 때, 연속질의를 효과적으로 처리하는 데이터 스트림 관리 시스템에서 윈도우라는 구조적 제약 조건으로 인하여 전체 질의 데이터에 대한 저장이 용이하지 않기 때문에 이를 해결하기 위하여 후보쌍 버퍼테이블을 활용한 질의 처리 방법론을 제안하였다. 기존의 유클리드 공간에서 이동객체들의 궤적 조인에 대한 연구들과는 다르게 기호공간의 그래프적 특성을 활용하여 보조기억장치에 궤적을 저장하지 않더라도 간단한 자료구조를 활용하여 이동객체의 인접성 검사를 가능하도록 하였다.

실험을 통하여 질의의 지속시간과 무관하게 질의 처리가 가능했으며, 이것은 윈도우의 크기와 무관하게 항상 같은 성능을 내는 방법임을 확인하였다.

본 논문에서는 이동객체의 움직임을 RFID리더기와 태그를 사용하여 획득하였으며, 이 때, RFID는 모든 방에 대한 인식이 가능하다는 가정 하에 실험을 진행하였지만, 실제로 RFID의 특성과 외부의 여러 방해 요소로 인하여 태그를 인식하는 RFID의 커버리지가 일정한 형태를 지니지 않고 태그의 인식률이 항상 100퍼센트가 아니기 때문에 전체공간에 대한 정확한 인식이 힘들 수 밖에 없다. 즉, RFID 리더기가 탐지 할 수 없는 공간에 이동객체가 존재하는 경우가 발생할 수 있는데, 향후 과제로 이러한 이동객체의 위치를 인식을 위하여 이전 타임스탬프의 이동객체의 위치와 기호공간의 그래프를 활용하여 현재 이동객체의 위치를 추적하는 연구 등이 이루어져야 한다. 그리고 인덱스를 구축하여 조인처리에 대한 수행 속도를 증가시키고, 본 논문에서 정의한 질의 처리를 위한 방법에 대한 비용 모델의 비교 역시 향후 연구 과제이다.

향후 실내공간인지 프로젝트와 함께 실내공간에 대하여 유비쿼터스 서비스가 실현될 때, 본 논문에서 제안한 조인 질의와 방법론이 유용하게 활용되기를 기대한다.

참 고 문 헌

[1] 이기준, 2008, Awareness Project and Indoor Spatial Data Model, 한국GIS학회지, 제 16권 4호

[2] Li, Ki-Joune, 2008, Indoor Space: A New Notion of Space, Springer-Verlag, 1-3.

[3] Jensen, Christian S. and Lu, Hua and Yang, Bin, 2009, Indexing the Trajectories of Moving Objects in Symbolic Indoor Space, Springer-Verlag 208-227.

[4] Yang, Bin and Lu, Hua and Jensen, Christian S.,2009, Scalable continuous range monitoring of moving objects in symbolic indoor space, ACM 671-680.

[5] Babu, Shivnath and Widom, Jennifer, 2001, Continuous queries over data streams, ACM 109-120.

[6] Babcock, Brian and Babu, Shivnath and Datar, Mayur and Motwani, Rajeev and Widom, Jennifer, 2002, Models and issues in data stream systems, ACM 1-16.

[7] Junichi Sakamoto and Hirokazu Miura and Noriyuki Matsuda and Hirokazu Taki and Noriyuki Abe and Satoshi Hori, 2005, Indoor Location Determination Using a Topological Model, DBLP143-149.

[8] Mamei, Marco and Zambonelli, Franco, 2005, Field-Based Coordination for Pervasive Multiagent Systems, Springer-Verlag.

[9] Hornsby, Kathleen and Egenhofer, Max J., 2002, Modeling Moving Objects over Multiple Granularities, Kluwer Academic Publishers, 177-194.

[10] Bakalov, Petko and Hadjieleftheriou, Marios and Keogh, Eamonn and Tsotras, Vassilis J., 2005, Efficient trajectory joins using symbolic representations, ACM, 86-93.

[11] Bakalov, Petko and Tsotras, Vassilis J., 2008, Continuous Spatiotemporal Trajectory Joins, pringer-Verlag, 109-128.

[12] Arasu, Arvind and Babu, Shivnath and Widom, Jennifer, 2006, The CQL continuous query language: semantic foundations and query ex-

ecution, Springer-Verlag, 121-142.
[13] Tiakas, Eleftherios and Papadopoulos, Apostolos N. and Nanopoulos, Alexandros and Manolopoulos, Yannis and Stojanovic, Dragan and Djordjevic-Kajan, Slobodanka, 2006, Trajectory Similarity Search in Spatial Networks, IEEE Computer Society.
[14] 강혜영, 2008 셀룰러 공간에 존재하는 이동객체 궤적의 유사성 측정, 한국 GIS학회지, 제16권 3호

논문접수 : 2010.03.04
수 정 일 : 1차 2010.04.26 / 2차 2010.04.30
심사완료 : 2010.04.30

황 병 주
2008년 신라대학교 컴퓨터공학과 졸업(학사)
2010년 부산대학교 컴퓨터정보공학과 졸업(석사)
2010년~현재 부산대학교 컴퓨터정보공학과 (박사 과정)
2010년~현재 국토연구원 위촉연구원
관심분야는 시공간DB, GIS, GIS표준

이 기 준
1984년 서울대학교 계산통계학과 졸업(학사)
1986년 서울대학교 계산통계학과(전자계산학 전공) 졸업(석사)
1992년 프랑스 응용과학원 전자계산학과 전산학박사
1993년~현재 부산대학교 정보컴퓨터공학부 교수
관심분야는 공간DB, 시공간DB, GIS, Telematics 및 Ubiquitous Computing, Indoor space databases