The Journal of GIS Association of Korea, Vol. 16, No. 2, pp. 1-18, July 2008

시공간 슬라이딩윈도우기법을 이용한 데이터스트림의 인과관계 결합질의처리방법

권 오 제* · 이 기 준**

Causality join query processing for data stream by spatio-temporal sliding window

Oje Kwon* · Ki-Joune Li**

요 약

센서로부터 획득되는 데이터 스트림은 스트림 데이터 간의 인과관계와 같은 다양한 유용한 정보를 포함한다. 센서 스트림에 대한 인과관계 조인질의는 스트림으로부터 인과관계의 (원인, 결과) 쌍을 찾아내는 것이다. 하지만 센서로부터 DSMS로 데이터가 전송될 때 발생하는 지연과 제한된 윈도우 크기로 인해 일부의 인과관계 결과 쌍이 손실될 수 있다. 본 논문에서는 먼저 데이터 스트림에서 인과관계 조인질의를 처리할 때 고려해야할 시간적, 공간적 그리고 시공간적 관점에 대해 관찰하고 이러한 관찰들을 고려한 다양한 슬라이딩 윈도우 처리 방법들을 제안한다. 제안된 방법들의 성능은 다양한 실험들을 통해 평가되어지는데 실험 결과들은 본 논문에서 제안된 방법들이 기존의 FIFO 방법에 비해 인과관계 질의 처리 결과가 더 정확함을 보여준다.

주요어 : 센서 스트림, 인과관계 조인, 시공간 슬라이딩 윈도우

ABSTRACT : Data stream collected from sensors contain a large amount of useful information including causality relationships. The causality join query for data stream is to retrieve a set of pairs (cause, effect) from streams of data. A part of causality pairs may however be lost from the query result, due to the delay from sensors to a data stream management system, and the limited size of sliding windows. In this paper, we first investigate spatial, temporal, and spatio-temporal aspects of the causality join query for data stream. Second, we propose several strategies for sliding window management based on these observations. The accuracy of the proposed strategies is studied by intensive experiments, and the result shows that

*부산대학교 컴퓨터공학과 박사과정(kwonoj@isel.cs.pusan.ac.kr)
**부산대학교 정보컴퓨터공학부 교수

we improve the accuracy of causality join query in data stream from simple FIFO strategy.

Keywords : sensor stream, causality join query, spatio-temporal sliding window

1. 서 론

센서로부터 취득되는 데이터는 인과관계 정보를 포함하여 많은 유용한 정보를 포함하고 있다. 센서의 데이터를 통하여 인과관계를 밝혀내는 것은 다양한 응용분야에서 일어나는 현상을 이해하고 그 현상에 대해 올바르게 대처하는데 매우 중요하다. 예를 들어, 온도를 감지하는 센서와 가스 파이프의 밸브의 오동작을 감지하는 센서가 있다고 가정해보자. 발화점으로부터 고온의 정보가 온도 센서에서 감지되고 가스 파이프 밸브의 오동작 정보가 밸브 감지 센서에서 감지된다. 이 경우 원인과 결과의 판단에 따라 두 가지 서로 다른 해석이 가능하다. 첫 번째, 가스 밸브의 오동작으로 인해 고온의 정보가 감지되어질 수 있다. 두 번째, 고온이 발생하여 가스 밸브의 오동작이 일어날 수 있다. 이 경우에서 사건의 원인과 결과를 어떻게 판단하느냐에 따라 화재에 대한 대처 방법은 달라져야한다.

데이터 스트림 처리 시스템(Data Stream Management System, DSMS)에서 센서 스트림 데이터 간의 원인과 결과의 쌍을 찾아내는 것은 매우 중요한 기능 중 하나이다. 본 논문에서는 이를 데이터 스트림의 인과관계 조인질의라 명한다. 시간적, 공간적 그리고 시공간적인 관계 정보는 특정 영역에 한정된 정보와 더불어 데이터 스트림의 인과관계를 분석하는데 매우 유용하다. 첫 번째로 원인은 결과에 시간적으로 앞선다. 두 번째로 원인과 결과에 대한 센서 데이터는 일정한 공간적 조건을 만족하여야 한다. 세 번째로 원인이 결과로 전달되는 것은 전달 속도와 같은 시공간적인 특성을 가진다. 이 세 가지 예제들은 각각 스트림 데이터의 원인과 결과가 맺는 시간적, 공간적 그리고 시공간적인 관계를 나타낸다.

대부분의 경우, 센서로부터 취득된 데이터가 스트림의 형태로 DSMS에 도착하기까지 다양한 이유로 지연이 존재한다. 이 지연은 제한된 슬라이딩 윈도우(sliding window)에 저장된 데이터를 가지고 질의를 처리하는데 있어서 정확도를 떨어뜨리는 중요한 요인이 된다. 예를 들어 기존에 가장 일반적으로 많이 사용되던 FIFO 방식으로 슬라이딩 윈도우에 저장된 데이터 간의 인과관계 질의를 처리하게 되면 정확도가 크게 떨어지게 된다. 이 문제를 해결하기 위하여서는 다양한 관점들이 고려되어야 하는데, 특히 앞 서 설명한 시간적, 공간적 그리고 시공간적인 관점들이 고려되어야 한다.

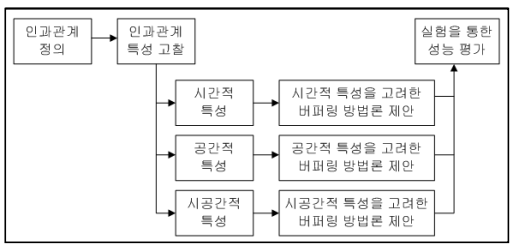
본 논문에서는 스트림 데이터에서 원인과 결과 간의 시간적, 공간적 그리고 시공간적인 특징들에 대해 관찰하고, 각각의 특성을 고려하여 인과 질의를 처리하는 방법들을 알아본다. 특히, 이 세 가지 측면을

고려한 슬라이딩 윈도우의 버퍼링(buffering) 정책을 제안하고 이를 실험을 통하여 기존의 First In First Out(FIFO) 방법과 비교하여본다. 본 논문이 기여하는 바를 요약하면 다음과 같다.

- 데이터 스트림에서 인과관계 조인질의를 소개하고
- 인과관계 조인질의에 대한 시간적, 공간적 그리고 시공간적 관계에 대해 관찰한다. 그리고,
- 인과관계 질의 결과의 정확성을 높이기 위한 슬라이딩 윈도우의 새로운 버퍼링 정책들을 소개한다.

본 논문의 전체 연구 수행 과정을 요약하면 [그림 1]과 같다. 스트림 데이터 간의 인과관계에 대해 정의를 하고 인과관계에 있는 두 스트림데이터에서 관찰되는 시간적, 공간적 그리고 시공간적 관계에 대해 고찰한다. 다음으로 이러한 관계들을 바탕으로 하여 새로운 윈도우 버퍼링 방법을 제안한다. 마지막으로 제안된 방법들의 성능을 기존의 FIFO 방법과 비교하여 본 논문에서 제안한 방법이 뛰어남을 증명한다.

본 논문은 다음과 같이 구성되어 있다. 먼저 2 장에서는 기존에 관련 연구들을 살펴보고, 3 장에서는 본 논문에서 다루고자 하는 문제를 제시하고 설명하며 스트림 데이터를 위한 인과관계 질의를 정의한다. 4 장에서는 인과관계 질의의 처리를 위하여 시간적, 공간적 그리고 시공간적 고려사항에 대해 알아보고 각각의 고려사항을 이용한 윈도우의 버퍼링 정책을 제시한다. 5 장에서는 다양한 실험을 통해 제시된 세 가지 정책과 FIFO 방법을 비교하고 마지막으로 6 장에서 결론을 내린다.



[그림 1] Total procedure of causality query process in sensor stream

2. 관련 연구

DSMS로 전달되는 센서 데이터는 무한한 스트림 형태를 나타낸다. DSMS에서 이러한 스트림을 처리하기 위해서는 몇 가지 제약 사항이 따른다. 첫 번째, DSMS는 제한된 버퍼를 가지고 있다. 따라서 부분의 데이터를 연속적으로 처리하는 비폐쇄형 연산(nonblocking operation)이 수행되어야 한다. 대표적인 슬라이딩 윈도우 방법은 STREAM(A. Arasu, 2003), Fjord(S. Madden and M. J. Franklin, 2002) 그리고 TelegraphCQ (S. Chandrasekaran, 2003)에서 제안되었다. 슬라이딩 윈도우의 성능은 윈도우의 크기에 의해 결정된다. 슬라이딩 윈도우의 크기가 클수록, 동시에 처리되는 데이터의 양이 많기 때문에 질의 처리 결과가 더 정확해진다. 하지만 질의 처리 수행 시간이 길어진다. STREAM, Fjord 그리고 TelegraphCQ에서 제안하는 슬라이딩 윈도우 방법은 데이터의 도착 시간(transaction time)만을 고려하는 FIFO 방법을 사용하였다. 하지만 스트림 데이터 간의 인과관계를 찾기 위해서 데이터의 도착 시간보다 발생 시간

(valid time)이 더 유용하다.

스트림 데이터를 처리할 때 DSMS가 고려해야할 두 번째 제약 사항은 실시간 질의 처리이다. 인과관계 질의는 조인 연산을 수행한다. STREAM은 해쉬 테이블을 이용한 이진 조인(binary-join) 방법을 제안하였다. Fjord는 여러 입력 스트림에 대해 도착 시간을 기준으로 동일한 윈도우 내에 존재하는 데이터들 간의 조인 연산을 수행하는 지퍼 조인(zipper join) 방법을 제안하였다. T. Urhan과 M. J. Franklin(2000)은 스트림 데이터가 DSMS에 전달될 때 발생하는 지연을 고려한 X-조인 연산 방법을 제안하였다. 이 방법은 먼저 메모리 기반으로 두 스트림에 대한 조인 연산을 수행하고 지연이 발생하는 경우 디스크 기반의 조인 연산을 수행한다. 하지만 위의 방법들은 데이터의 도착 시간만을 고려하며 조인 연산 수행 시 스트림의 시공간적인 요소를 전혀 고려하고 있지 않다.

L. King 외(2003)는 데이터 스트림에 대한 메타 데이터를 이용한 M-조인 방법을 제안하였다. 이 방법은 중단점(punctuation)을 이용하는데, 중단점은 중단점 표시 이후로 들어오는 데이터는 특정 조건을 만족하지 않음을 명시하는 조건자이다. 따라서 중단점 조건을 만족하지 않는 데이터들은 윈도우에서 우선적으로 삭제된다. 하지만 중단점 조건은 발생 시간을 고려하고 있지 않고 중단점을 설정하는 단계는 전처리로 수행된다. M. A. Hammad 외(2003)는 다수의 센서 스트림의 서로 다른 지연 시간을 고려한 스트림 윈도우 조인(Stream Window Join) 방법을 제안하였다. 이 방법은 센서 스트림들 간의 최대 지연 범위를 설정하고 서로 다른 스트림 간의 조인 쌍이 맺어질 때 까지 연속적으로 조인 연산을 수행한다. 하지만 이 방법 역시 데이터의 도착 시간에 따라 순차적인 스트림만을 가정하고 있다. J. Wu 외(2007)는 조인 결과의 재현율이 떨어지는 원인을 사용자에 의해 정해진 윈도우 크기 문제로 보고, 스트림의 순서와 지연시간을 고려하여 윈도우 크기를 동적으로 설정하는 방법을 제안하였다. 하지만 이 방법 역시 스트림 데이터의 도착 시간만을 고려하여 FIFO 방식으로 수행된다.

데이터 마이닝에서 인과관계는 자료들 간의 원인과 결과 관계를 추론해 내는 방법이다(Silverstein C. et al., 2000; D. Freedman, 2004; P. W. Holland, 1986; Pearl. J., 2000). LTCCS(2007)는 실제 트럭 간에 발생한 사고 정보를 수집하여 통계적인 분석 방법을 통해 사건의 원인과 결과를 분석하였다. XinZhou Qin과 Wenke Lee(2003)는 데이터들 간의 인과관계를 통계적으로 분석하여 시스템의 위험 상황에 대해 경고하는 방법을 제안하였다. 하지만 센서 스트림에 대한 인과관계 분석은 무한한 센서 스트림에 대해 실시간으로 처리되어야 하고 제한적인 윈도우 크기로 인해 통계적인 분석이 불가능하다. 이러한 이유로 센서 스트림 데이터의 인과관계 질의를 처리하기 위한 새로운 방법이 필요하다.

3. 데이터 스트림에서 인과관계 조인질의

본 장에서는 데이터 스트림에서 인과관계 조인질의를 정의한다. 그리고 센서 네

트워크 환경에서 인과관계 조인질의를 수행하기 위해 고려해야 할 문제들에 대해 논의한다.

3.1 데이터 스트림에서 인과관계 조인질의

센서 스트림에서 인과관계 조인질의는 인과관계 조건을 만족하는 (원인, 결과)의 쌍을 찾는 질의로 정의되는데 다음과 같이 표현될 수 있다;

정의 1. 데이터 스트림에서 인과관계 조인질의

주어진 데이터 스트림에 대해, 인과관계 조인질의는 다음과 같이 정의된다.

$$R\_CQ(X) = (c,e) | c \in W(X_i), e \in W(X_j), \\ \text{and } P_{CQ}(c,e) = TRUE$$

$X = X_1, X_2, ..., X_K$는 스트림의 집합을 나타내고, c와 e는 각각 원인과 결과를 나타낸다. $W(X_i)$는 X_i 스트림에 대한 윈도우 버퍼를 나타낸다.

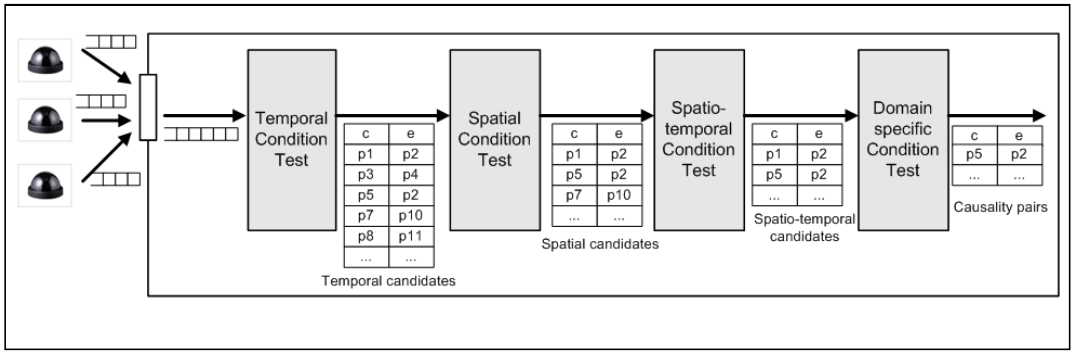
위의 정의에서 $P_{CQ}(c,e)$는 인과관계 조건을 만족하는 조건자이다. 이 조건자는 다음과 같이 세부적인 조건자들을 결합한 형태로 표현된다.

$$P_{CQ}(c,e) = P_{CQ1}(c,e) \wedge P_{CQ2}(c,e) ... \wedge P_{CQn}(c,e) \quad (1)$$

인과관계 조인질의를 수행하기 위해서는 응용 분야에 따라 정의되는 위의 인과관계 조건자들의 각 조건들을 조사하여야 한다. 수식 (1)에 나타나있는 조건자들은 다음과 같이 네 가지로 분류된다.

- **시간적 조건** : 원인과 결과는 항상 원인이 결과보다 우선적으로 발생해야 한다는 시간적 조건을 만족해야 한다.
- **공간적 조건** : 원인이 발생한 지점과 결과가 발생한 지점은 일정한 공간적 조건을 만족하여야 한다. 예를 들어 결과는 원인과 일정한 거리 이내에 있어야 한다.
- **시공간적 조건** : 인과관계는 원인과 결과의 동적인 관계로 시공간적인 특성을 가지고 있다. 한 가지 분명한 시공간적 특성은 원인에서 결과로 전달되는 속도 정보이다. 예를 들어, 화재는 발화 지점으로부터 속도를 가지고 퍼진다. 이러한 사실은 발화점으로부터 화재의 결과까지 전달되는 시간을 거리와 속도의 함수로 판단할 수 있음을 나타낸다.
- **응용 분야에 한정되어 있는 조건** : 위의 세 가지 조건은 응용의 종류와 관계없이 정의될 수 있는 조건이라고 한다면, 응용에 따라서 정의될 수 있는 조건이 있다. 이 조건은 본 논문의 범위에 포함되지 않는다.

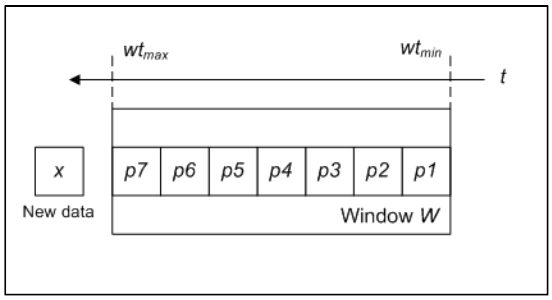
인과관계 조인질의에 대한 위의 네 가지 조건을 조사하는 과정은 [그림 2]와 같다. 먼저 시간적 조건을 조사하고, 다음 단계에서 공간적 조건을 조사하고, 세 번째 단계에서 시공간 조건을 조사하면 일정한 수의 후보 쌍의 집합을 얻을 수 있다. 그렇게 되면 마지막 단계에서 응용 분야에 한정된 조건만을 확인하여 최종 결과를 얻을 수 있다.



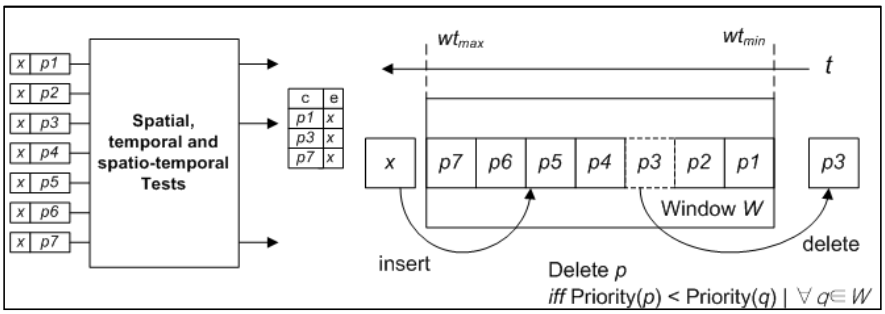
[그림 2] Conceptual evaluation procedure of causality join query in data stream

3.2 인과관계 조인질의를 처리하기 위한 버퍼링 방법

데이터 스트림을 위한 인과관계 조인질의는 데이터베이스 관리 시스템의 조인질의 처리와는 달리 [그림 3]과 같이 제한된 슬라이딩 윈도우에 저장된 데이터만 가지고 수행한다. 본 논문에서는 편의상 모든



(a) Initial data stream in window



(b) Casuality join query process and deletion data from window

[그림 3] Causality join query processing procedure with data stream

센서의 데이터가 하나의 데이터 스트림으로 들어와 하나의 슬라이딩 윈도우를 통해 인과관계 질의가 수행된다고 가정하였다.

[그림 3]에서와 같이 현재 슬라이딩 윈도우 *W*에 있는 데이터들 $p1 \sim p7$ 간에 이미 모든 인과관계가 조사되어 결과가 보고되었다. 이 때 새로운 하나의 데이터 *x*가 들어오면 다음의 두 가지 작업을 수행하여야 한다.

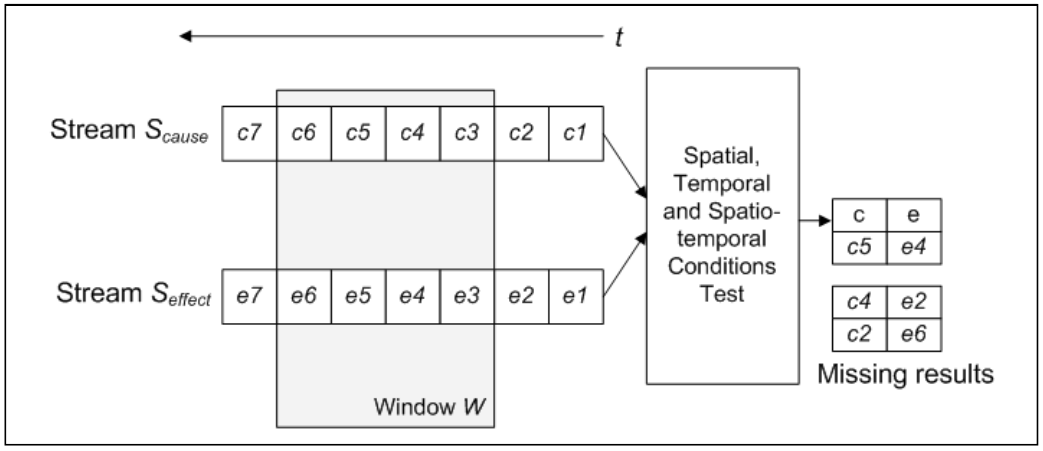
- *W*의 데이터들과 조인 : *W*에 저장된 모든 데이터들이 새로 들어온 x와 인과관계 조건자 $P_{CQ}(p,x)$를 만족하는지 조사되어야 한다.
- 하나의 데이터를 윈도우에서 제거 : *W*의 크기는 제한되어 있으므로 하나의 데이터를 *W*에서 제거하여야 한다. 제거되는 데이터는 앞으로 들어올 데이터와 인과관계의 확률이 가장 작은 것이어야 한다.

3.3 스트림의 인과관계 조인질의 처리 시 고려 사항

본 논문에서 가정하는 컴퓨팅 환경은 센서 네트워크와 DSMS가 설치되어 운영되는 베이스 스테이션으로 구성되어 있다. 센서는 멀티홉(multi-hop)으로 DSMS 에 데이터를 전달하면 DSMS 에서 적절한 처리를 하게 된다. 이러한 환경에서 스트림 데이터의 인과관계 질의를 수행하기 위해서는 다음과 같은 중요한 문제가 존재한다.

- 제한된 윈도우의 크기 : 윈도우의 크기가 제한되어 있기 때문에 DSMS에서 모든 스트림 데이터에 대해 인과관계 질의를 수행할 수 없고 윈도우에 저장된 데이터에 대해서만 인과관계 질의 처리가 가능하다. 따라서 일부의 인과관계 질의 결과의 손실이 일어날 수 있다.
- 지연 시간 : 본 논문에서 가정하는 환경은 멀티홉으로 연결되는 센서 네트워크이다. 이 환경에서는 다양한 이유로 센서에서 취득된 정보가 DSMS가 있는 베이스 스테이션까지 전달되는 과정에서 지연이 발생한다. 그런데 이 지연은 랜덤하게 발생하여 먼저 센서에서 취득된 원인의 데이터가 오히려 결과의 데이터보다 늦게 DSMS에 도착할 수 있게 된다.

위의 두 가지 문제는 서로 독립적인 것이 아니라 연관되어 인과관계 질의의 결과를 부정확하게 만든다. [그림 4]는 부정확한 질의 결과가 일어나는 현상을 보여주는 예이다. 그림에서 $(c5,e4),(c4,e2),(c2,e6)$ 이 인과관계 질의 결과라고 가정하자. 지연 시간으로 인해 $c6,c5,c4$ 그리고 $c3$이 늦게 DSMS에 도착하였다. 윈도우의 크기가 4이고 슬라이딩 윈도우는 FIFO로 수행된다고 가정할 때, $c2$는 이미 슬라이딩 윈도우에서 삭제되었기 때문에 $(c4,e2)$ 는 질의 결과에 포함될 수 없다. 같은 이유로 $(c2,e6)$ 역시 질의 결과에 포함되지 않고 $(c5,e4)$ 만이 질의 결과에 포함된다. 이것은 기존의 FIFO 방법이 데이터의 발생 시간(t_v) 이 아닌 DSMS에 도착한 시간(t_t)만을 고려하고 있기 때문이다. 하지만 스트림 데이터의 인과관계 질의를 처리하기 위해서는 도착 시간 보다 발생 시간이 더욱 중요하다.



[그림 4] Example : Inaccurate query result due to the limited size of window and transfer delay

본 논문에서는 인과관계 질의의 성능을 향상시키기 위한 여러 방법들을 제안한다. 본 논문에서 제안하는 슬라이딩 윈도우의 버퍼링 방법의 목적은 원인과 결과가 동시에 윈도우 내에 존재하도록 하는 것이다. 이를 위하여 데이터 스트림의 원인과 결과 사이의 시간적, 공간적 그리고 시공간적 관계를 살펴보고 이를 반영한 여러 가지 버퍼링 방법을 제안한다.

4. 원인과 결과의 시공간적 관계

본 장에서는 센서 스트림에서 원인과 결과 사이에 존재하는 일반적인 시간적, 공간적 그리고 시공간적인 특성에 대해 관찰한다. 그리고 이러한 특성들을 기반으로 슬라이딩 윈도우의 새로운 버퍼링 정책들을 제한한다.

4.1 인과관계의 시간적 연관성

원인과 결과 사이의 가장 직관적이고 확실한 관계는 시간적 선후 관계로 원인은 항상 결과보다 시간적으로 앞선다는 것이다. 이를 조건자로 표현을 하면,

$$P_{CQ.T}(c,e)=\begin{cases}TRUE & \text{if } t_V(c)<t_v(e)<t_v(c)+\delta_T,\\ FALSE & \text{otherwise}\end{cases}$$

$t_v(c)$ 와 $t_v(e)$는 각각 원인 c와 결과 e의 발생 시간을 나타낸다. 인과관계의 시간적 특성을 요약하면 다음과 같다.

특성 1.
인과관계의 시간적 특성은

c는 원인이고 e가 c의 결과이면 $P_{CQ.T}(c,e)$ = TRUE 를 항상 만족한다.

결과적으로 인과관계 질의 처리 시 인과관계의 시간적 특성을 만족하는 (c,e)

쌍은 질의 결과로 선택되어야 한다.

슬라이딩 윈도우에서 버퍼링을 수행할 때 가장 중요한 요구 사항은 나중에 들어올 데이터와 인과관계를 만족할 확률이 가장 작은 데이터를 삭제하는 것이다. 인과관계의 시간적인 특성은 원인과 결과의 발생 시간 차이가 클수록 원인과 결과가 인과관계를 만족할 확률이 낮아짐을 의미한다. 이는 다음의 가정으로 표현될 수 있다.

가정 1.

슬라이딩 윈도우 W에 포함되어 있는 두 데이터 p, q 에 대해,

if $t_v(p) < t_v(q)$ then,
Prob($P_{CQ,T}(p,x)$ = TRUE)
< Prob($P_{CQ,T}(q,x)$ = TRUE),

여기서 x는 앞으로 DSMS 에 도착할 데이터이다. 물론 위의 가정 1은 증명하기는 어렵지만 직관적으로 받아들일 수 있다. 그리고 가정 1은 슬라이딩 윈도우에서 FIFO 방식이 아니라 데이터의 발생 시간 순서로 데이터를 삭제해야 함을 의미한다. 이러한 관찰을 이용하여 FHFO(First Happens First Out) 버퍼링 방법을 제안한다.

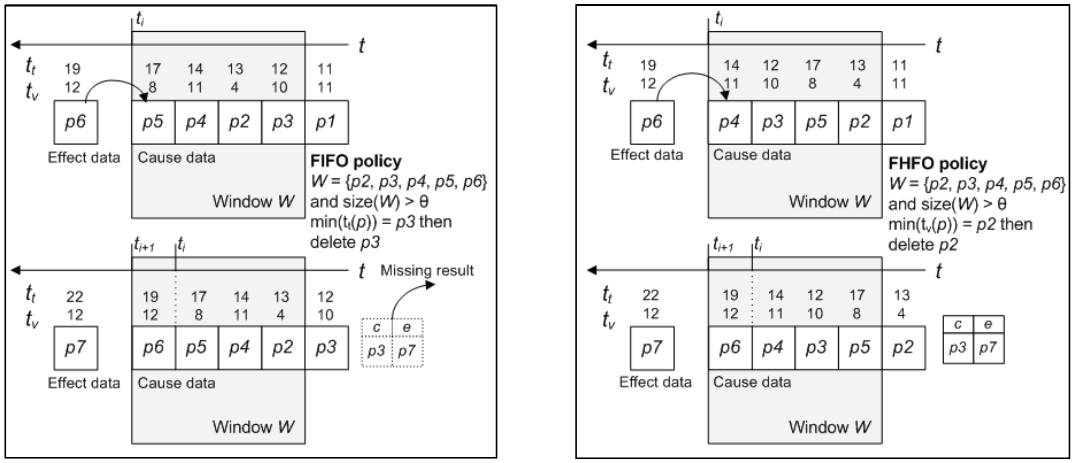
정의 2. FHFO(First Happens First Out)

슬라이딩 윈도우 W에서 FHFO 버퍼링 방법은 다음과 같이 정의된다.

FHFO(W) : remove $p \in W$ such that
$t_v(p)$ = min($\{t_v(q) | q \in W\}$)

FHFO 방법을 이용하여 인과관계 질의 결과의 손실을 줄여 결과적으로 인과관계 질의 처리의 성능을 높일 수 있다. 이를 FIFO 와 비교하여 설명하면 [그림 5]와 같다.

[그림 5]에서 인과관계 질의 결과를 $(p3, p7)$ 라고 가정하자. 새로운 데이터 $p6$ 이 도착했을 때, FIFO 방법의 경우 $p3$이 윈도우에서 삭제된다. 이 후 $p7$이 도착하면 이미 $p3$이



(a) FIFO buffering policy (b) FHFO buffering policy

[그림 5] Comparison of FIFO and FHFO

윈도우에서 삭제되었기 때문에 결과적으로 $(p3, p7)$은 결과 집합에 포함되지 않는다. 반면에 FHFO 방법의 경우 $t_v(p2) < t_v(p3)$이기 때문에 $p2$가 삭제되고 $p3$는 윈도우에 남기 때문에 $(p3, p7)$가 결과 집합에 포함된다.

4.2 인과관계의 공간적 연관성

공간적 관계도 시간적인 관계와 더불어 원인과 결과 사이의 중요한 관계이다. 하지만 일반적으로 공간적 관계는 시간적 관계와 달리 기하학적 관계, 위상적 관계와 같이 다양하고 복잡하다. 본 논문에서는 인과관계의 공간적 관계를 일반적이고 분명한 관계인 원인과 결과의 거리로 초점을 맞춘다.

$$P_{CQ.S}(c,e) = \begin{cases} TRUE & \text{if } dist(p(c), p(e)) < \delta_S, \\ FALSE & \text{otherwise} \end{cases}$$

$p(c)$와 $p(e)$는 각각 원인과 결과의 지리적 위치를 나타낸다. 만약 원인과 결과의 거리가 멀다면, 두 데이터는 인과관계를 만족하지 않는다. 여기서 거리는 응용분야와 원인, 결과의 종류에 따라 결정된다.

공간적 특성은 원인과 결과 사이의 공간적 거리가 멀어지면 두 데이터가 인과관계를 만족할 확률이 낮아짐을 의미한다. 이는 다음의 가정과 같이 표현된다.

가정 2.

슬라이딩 윈도우 W에 포함되는 두 데이터 p, q에 대해,

if $dist(p(p), p(x)) < dist(p(q), p(x))$ then,

$$\text{Prob}(P_{CQ.S}(p,x) = \text{TRUE}) > \text{Prob}(P_{CQ.S}(q,x) = \text{TRUE}),$$

x는 센서 네트워크에서 DSMS에 도착할 데이터를 나타낸다. 윈도우 내의 데이터를 위치에 대해 정렬할 수 없기 때문에, 위의 가정을 적용하기 위해 거리를 DSMS와 윈도우 내의 데이터 간의 거리로 정의한다. 물론 이러한 거리는 스트림 데이터의 공간적인 특성을 완벽하게 반영하지 못할 수 있다. 본 논문에서는 이러한 공간적 특성을 반영한 FCFO(First Closely located First Out) 버퍼링 방법을 제안한다. FCFO는 베이스 스테이션 b까지의 거리가 가장 짧은 데이터 p를 버퍼에서 삭제한다. 이 방법을 요약하면 정의 3과 같다.

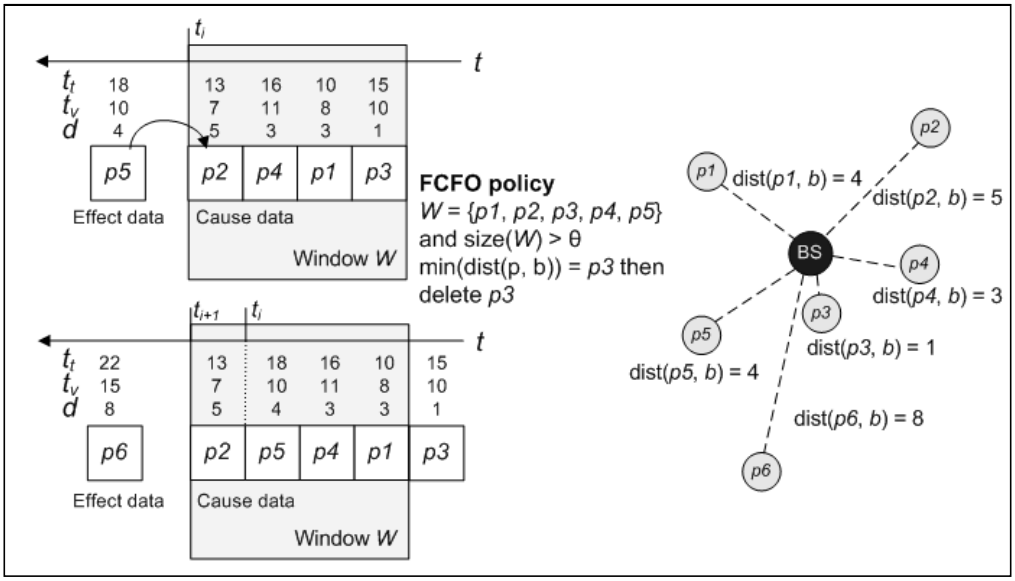
정의 3. FCFO(First Closely located First Out) 슬라이딩 윈도우 W에서 FCFO 버퍼링 방법은 다음과 같이 정의된다.

FCFO(W) : remove $p \in W$ such that
$dist(p,b) = \min(\{dist(q,b) | q \in W\})$
or $t_{cur} - t_t(p) > t_{max.stay}$

t_{cur}과 $t_t(p)$는 각각 현재 시간과 p의 도착 시간을 나타낸다.

FCFO의 방법에서 두 번째 조건은 데이터가 윈도우 내에 오랫동안 남아있는 것을 방지하기 위한 것이다. 만약 데이터 a가 $t_{max.stay}$ 시간보다 오랫동안 윈도우 내에 존재하게 되면 이 데이터를 삭제한다. FCFO 방법의 예는 [그림 6]에 나타나 있다.

[그림 6]의 우측 그림과 같이 센서들이 배치되어 있다고 가정하자. 새로운 데이터



[그림 6] Example of FCFO

$p5$가 도착하면 먼저 윈도우 내에 $t_{max.stay}$ 시간 보다 오래 남아있는 데이터가 있는지 찾는다. 만약 $t_{max.stay}$보다 오래 남아있는 데이터가 없다면, 베이스 스테이션과의 거리가 가장 짧은 $p3$이 버퍼에서 삭제된다.

4.3 인과관계의 시공간적 연관성

원인과 결과 사이의 세 번째 관계는 시공간적인 관계이다. 본 논문에서는 원인에서 결과로 전송되는 속도와 연관된 시공간적 관계에 초점을 맞춘다. 원인에서 결과로 데이터가 전송되는 속도를 v라 하고 원인과 결과 사이의 거리가 s라고 할 때, 결과는 원인보다 적어도 $t_p = s/v$ 시간 이후에 발생한다. 이러한 관계를 시공간 조건자인 $P_{CQ.ST}(c,e)$ 로 표현하면 다음과 같다.

$$P_{CQ.ST}(c,e) = \begin{cases} TRUE & \text{if } t_p < t_v(e) - t_v(c) < t_p + \delta_P, \\ FALSE & \text{otherwise} \end{cases}$$

δ_P는 원인에서 결과로 전달될 때 발생하는 전송 시간의 최대 허용 오차를 나타낸다. 만약 원인에서 결과로 전달되는 시간이 $t_p + \delta_P$ 보다 크면 두 데이터는 인과관계를 만족하지 않는다. $t_p = 0$인 경우 시공간 조건자는 시간적 조건자와 같다.

이러한 시공간 조건자를 바탕으로 FHCFO (First Happen and Closely located First Out) 버퍼링 방법을 제안한다. 새로운 데이터 x가 DSMS에 도착하였다고 가정해 보자. 윈도우 W에 존재하는 데이터 p에 대해, 만약 $t_v(p) - t_v(x) < t_p + \delta_P$의 조건을 만족한다면 p와 x는 인과 조건을 만족할 확률이 높다. 다시 말해 p는 $t_p + \delta_P$ 시간이 만료되기 전까지 윈도우 W에 남아있어야 한다. 이 시간이 만료되면, 시간이 흐를수록 인과관계를 만족할 확률이 낮아진다. 이러한 관점을 기초로 하여 FHCFO 방법을 정의하면 다음과 같다.

정의 4.

FHCFO(First Happen and Closely located First Out)

슬라이딩 윈도우 W에 대해 FHCFO 버퍼링 방법은 다음과 같이 정의된다.

FHCFO(W) : If $(t_v(p) - t_v(x) < t_p + \delta_P)$, then remove $p \in W$ such that *dist* (p, b) = *min*($\{\text{dist}(q, b) | q \in W\}$) Else, remove $p \in W$ such that $t_v(p)$ = min($\{t_v(q) | q \in W\}$)

이 방법은 FCFO와 FHFO 방법의 합성 형태이다. [그림 7]은 FHCFO 방법의 예를 보여준다.

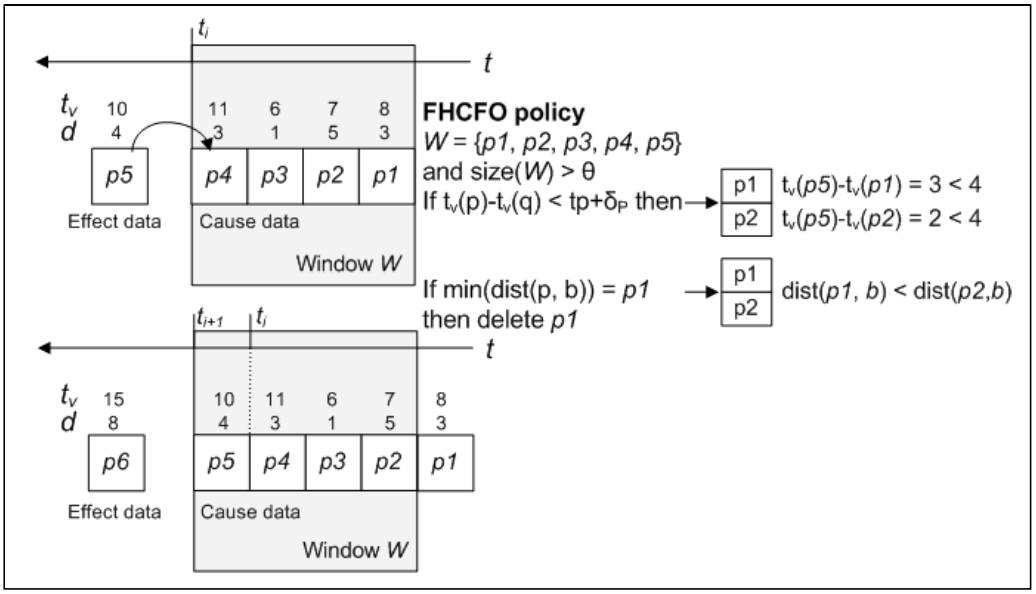
[그림 7]에서 $t_p + \delta_P = 4$라고 가정하고 센서들은 [그림 6]과 같이 분포하고 있다고 가정하자. 새로운 데이터 $p5$가 도착했을 때, 먼저 윈도우 W 내에서 $t_v(p) - t_v(p5) < t_p + \delta_P$를 만족하는 데이터 p가 존재하는지 검색한다. 만약 이 조건을 만족하는 데이터가 존재한다면, 그 데이터들 중에 베이스 스테이션과의 거리가 가장 짧은 데이터를 선택하여 삭제한다. [그림 7]에서 $p1$ 과 베이스 스테이션과의 거리가 가장 짧기 때문에 $p1$이 버퍼에서 삭제된다.

5. 성능 평가

이 장에서는 다양한 실험을 통해 본 논문에서 제안하는 방법들과 기존의 FIFO의 성능을 비교한다.

5.1 실험 환경

다음의 매개 변수들을 이용하여 실험을



[그림 7] Example of FHCFO

수행하기 위한 데이터를 생성하였다.

- 전체 데이터 공간 범위 : $[0, 1]^2$
- DSMS의 위치 : (0.5, 0.5)
- 센서의 수 : 100
- 센서 당 생성되는 데이터의 수 : 50
- 센서 당 생성 주기 : 주어진 기댓값 λ_1에 대하여 지수 분포(exponential distribution)에 따라 랜덤하게 생성
- 한 홉 당 발생하는 지연 : 주어진 기댓값 λ_2에 대하여 지수 분포에 따라 랜덤하게 생성

본 논문에서는 실험 결과의 정확성을 측정하기 위하여 질의 결과의 재현율(recall rate)을 측정하였다. 재현율은 인과관계 조건을 만족하는 모든 인과 질의 결과 쌍의 수에 대해 각 버퍼링 방법을 통해 검색된 인과 질의 결과 수의 비율로 정의된다.

$$Recall(P) = num_{Result}(P)/num_{Causality},$$
$$P = \{FIFO, FHFO, FCFO, FHCFO\}$$

본 논문에서는 인과관계 조인 연산의 수행 시간에 대해서는 논하지 않는데, 그 이유는 슬라이딩 윈도우에 포함되는 데이터의 수가 작기 때문에 조인 연산 시 발생하는 수행 시간은 거의 무시할 수 있기 때문이다. 실험을 위한 매개 변수들과 그 설명은 <표 1>에 나타나 있다.

5.2 실험 결과

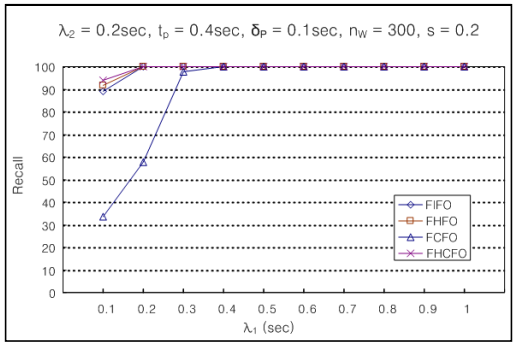
<표 1>의 매개 변수를 이용하여 각각의 버퍼링 방법의 정확성을 측정하기 위하여 다양한 실험을 수행하였다. <표 1>의 매개 변수들의 조합하면 많은 가능한 실험 환경이 만들어지는데 본 논문에서는 대부분의 실험 환경에서 성능 평가를 수행하였다. 본 논문에서는 모든 방법들의 정확도가 100% 에 근접한 실험 결과들은 제외하고 의미있는 결과들에 대해 설명한다.

5.2.1. 발생 시간 λ_1에 대한 실험

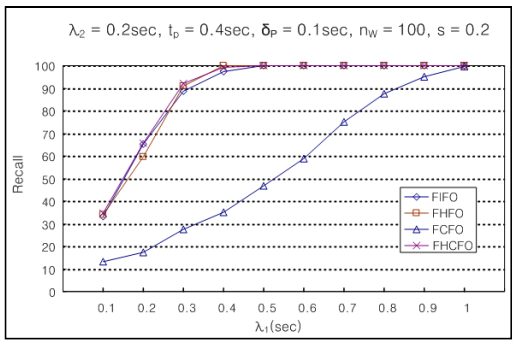
[그림 8]은 발생 주기의 기댓값 λ_1의 변화에 따른 각 버퍼링 방법의 재현율을 보여준다. x, y 축은 각각 발생 주기 λ_1와 재현율을 나타낸다. 발생 주기가 커지면, 스트림 내의 데이터들의 순서가 데이터가 발생했을 때의 순서일 확률이 커진다. [그림 8]에서 FHCFO는 FIFO에 비해 윈도우

<표 1> Parameters and their values for experiments

매개 변수	설명	범위
λ_1	생성 주기의 기댓값(연속된 두 발생 시간 사이의 시간 주기)	0.1, 0.2, ..., 1.0 (sec/frequency)
λ_2	한 홉 당 발생 지연 시간의 기댓값	0.02, 0.04, ..., 0.2(sec)
t_p	원인에서 결과로의 최소 전달 시간	0.2, 0.4, ..., 2.0(sec)
δ_P	t_p의 허용 오차	0.1, 0.2, ..., 1.0(sec)
n_W	슬라이딩 윈도우의 크기	50, 100, ..., 500
s	원인과 결과 사이의 공간적 거리	0.1, 0.15, ..., 0.45



(a) n_W = 300



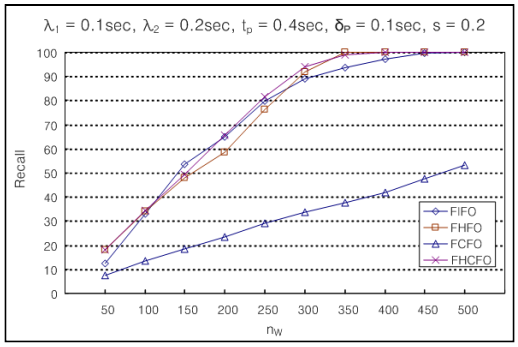
(b) n_W = 100

[그림 8] Experiments on Occurrence Rates λ_1

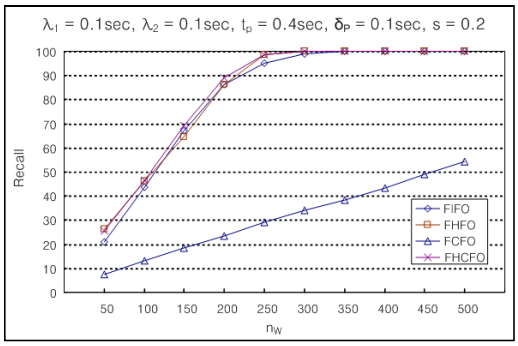
크기가 클 때(n_W = 300) 최대 7% 그리고 윈도우 크기가 작을 때(n_W = 100) 최대 3% 재현율이 높다. FHFO 역시 FIFO에 비해 최대 3% 재현율이 높다. 그 이유는 FHCFO와 FHFO가 데이터의 발생 시간을 이용하여 윈도우 내에서 발생 순서와 다르게 배치된 데이터들을 조절하기 때문이다.

5.2.2 슬라이딩 윈도우의 크기 n_W에 대한 실험

[그림 9]는 슬라이딩 윈도우의 크기 n_W의 변화에 따른 각 버퍼링 방법들의 재현율을 나타낸다. 윈도우 크기가 커지면 각 버퍼링 방법들의 재현율이 증가하는 것은 분명하다. FHCFO는 FIFO에 비해 최대 6% 재현율이 높고 FHFO는 FIFO에 비해 최대 3% 재현율이 높다. 홉 당 지연 시간이 짧을 때(λ_2 = 0.1), FIFO에 대하 FHCFO의 성능 향상 비율이 낮다. 이것은 지연 시간이 짧으면 스트림 내에서 데이터들의 순서가 발생 순서와 같을 확률이 높기 때문이다. 이러한 결과는 [그림 10]에서 더 분명히 보여 진다. 이 실험에서 중요하게 관찰되어지는 한 가지는 슬라이딩 윈도우의 크기가 실험 결과의 정확성에 큰 영향을 끼

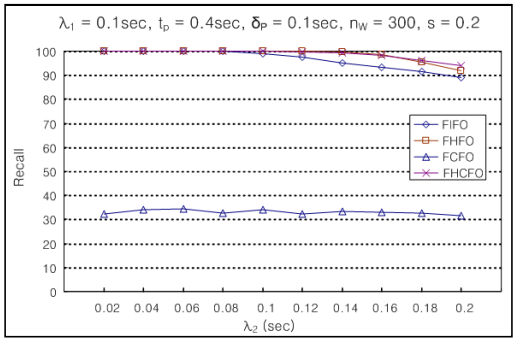


(a) λ_2 = 0.1

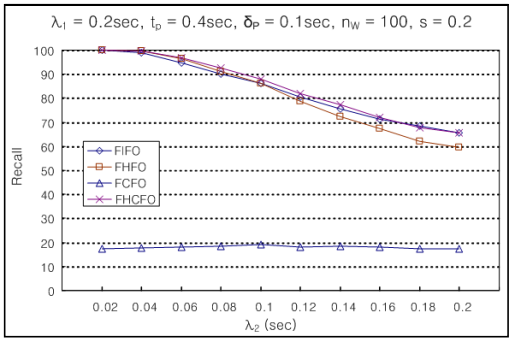


(b) λ_2 = 0.1

[그림 9] Experiments on Window Size n_W



(a) λ_1 = 0.1



(b) λ_1 = 0.2

[그림 10] Experiments on Transfer Delay λ_2

친다는 것이다. 만약 윈도우 크기가 충분히 크지 않으면, 질의 결과의 반 이상을 잃을 수 있다. 그리고 실험 결과 FCFO의 성능이 가장 나쁨을 알 수 있다. 이는 공간적인 고려만을 통해 인과관계 질의 결과의 정확성을 높이는데 한계가 있음을 의미한다.

5.2.3 전송 지연 λ_2에 대한 실험

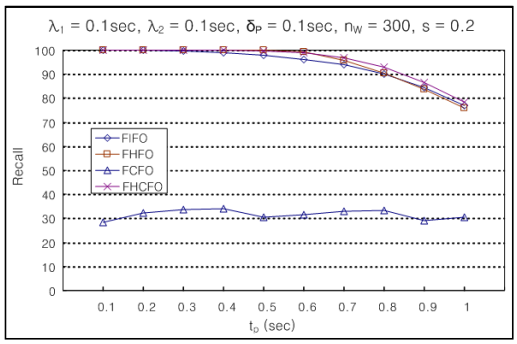
[그림 10]은 홉 당 지연 시간 λ_2과 재현율 간의 관계를 보여준다. 지연 시간이 짧으면, 스트림 내 데이터의 순서가 발생 순서와 같을 확률이 높아지고 그 결과 재현율이 증가한다. [그림 10]에서, FHCFO는 슬라이딩 윈도우의 크기가 큰 경우(n_W = 300) 최대 5% 그리고 윈도우 크기가 작은 경우(n_W = 100) 최대 2% FIFO에 비해 성능이 뛰어남을 보였다. 하지만 FHFO는 [그림 10]-(b)와 [그림 9]-(a)와 같이 지연 시간이 클 때 FIFO보다 뛰어난 성능을 보이지 않는다. 이는 시간적인 관점만을 고려하여 인과관계 질의 결과의 정확성을 높이는데 한계가 있음을 의미한다. 반대로 FHCFO는 윈도우의 크기, 발생 주기 그리고 지연 시간에 상관없이 FIFO 보다 뛰어난 성능을 보인다.

5.2.4 결과로의 최소 전달 시간 t_p에 대한 실험

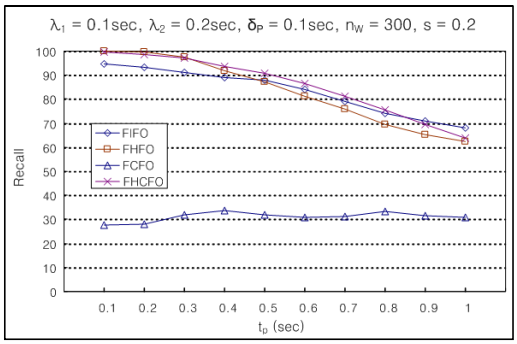
[그림 11]은 원인에서 결과로의 최소 전달 시간 t_p에 따른 재현율의 변화를 보여준다. t_p가 클수록 재현율은 감소한다. 하지만 지연 시간이 클 때 재현율이 급격히 감소하였다. 따라서 질의 결과의 손실을 줄이기 위해는 윈도우의 크기를 충분히 크게 설정해야 한다. t_p가 크지 않을 때, FHCFO는 지연 시간이 짧을 경우(λ_2 = 0.1) 최대 3% 그리고 지연 시간이 길 경우(λ_2 = 0.2) 최대 7% FIFO 보다 성능이 뛰어나다. 그리고 FHFO는 FIFO 보다 최대 2% 뛰어남을 알 수 있다.

5.2.5 허용 오차 δ_P에 대한 실험

[그림 12]는 t_p의 허용 오차 δ_P에 따른 재현율의 변화를 나타낸다. 허용 오차가 크다는 것은 원인과 결과의 발생 시간의 차

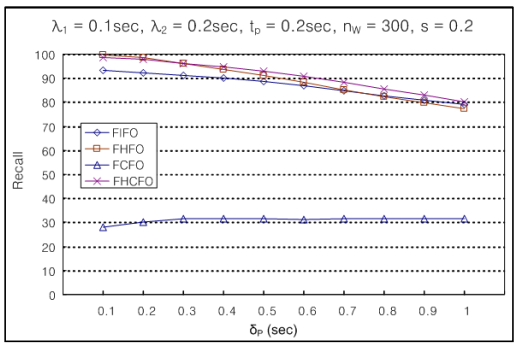


(a) λ_2 = 0.1

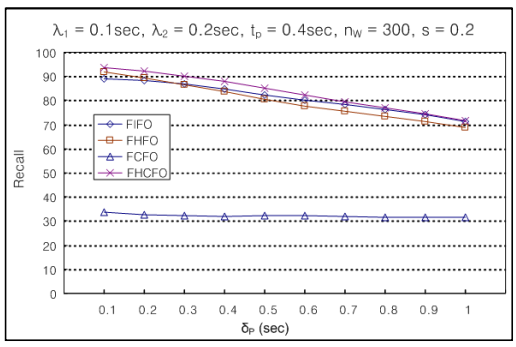


(b) λ_2 = 0.2

[그림 11] Experiments on the effect propagation time t_p



(a) t_p = 0.2



(b) t_p = 0.4

[그림 12] Recall of four policies as δ_P changes

이가 큼을 의미한다. [그림 12]에서 보여지듯이, 허용 오차가 클수록 원인과 결과가 동시에 버퍼에 존재할 확률은 낮아진다. FHCFO와 FHFO는 인과관계 조건을 만족할 확률이 높은 데이터를 보다 오랫동안 버퍼에 남겨두기 때문에 FIFO에 비해 각각 최대 8%, 6% 성능이 뛰어나다.

5.2.6 실험 결과 요약

실험 결과를 요약해보면 <표 2>와 같이 정리할 수 있다.

FHCFO는 모든 실험 결과 다른 방법들에 비해 스트림 데이터의 인과관계 질의 처리 시 높은 재현율을 보인다.

FHFO는 몇 가지 실험을 제외한 대부분의 경우에서 FIFO 보다 높은 재현율을 보인다.

몇 가지 경우에서 모든 방법들의 재현율이 50% 이하로 떨어진다. 올바른 재현율을 얻기 위해서 슬라이딩 윈도우의 크기를 충분히 크게 해줘야 한다.

6. 결 론

센서들로부터 취득된 스트림 데이터는

<표 2> Summary of experimental results

실험 환경	재현율에 따른 성능 우위 순서	비고
λ_1 실험	FHCFO > FHFO > FIFO > FCFO	
λ_2 실험	FHCFO > FHFO > FIFO > FCFO	
t_p 실험	FHCFO > FHFO > FIFO > FCFO	λ_1=0.2 일 때, FIFO > FHFO
δ_P 실험	FHCFO > FHFO > FIFO > FCFO	
n_W 실험	FHCFO > FHFO > FIFO > FCFO	

인과관계 정보를 포함하여 다수의 다양한 정보들을 포함하고 있다. 본 논문에서는 데이터 스트림에서 인과관계 조인질의를 처리하기 위해 필요한 슬라이딩윈도우의 버퍼링 방법을 제안하였다. 센서 스트림에서 인과관계 조인질의를 처리하기 위해서는 원인과 결과의 시간적, 공간적 그리고 시공간적 관계를 관찰하는 것이 중요하다. 하지만 데이터의 도착 시간만을 고려하는 기존의 FIFO 방식은 DSMS의 제한된 윈도우 크기와 지연시간으로 인해 인과관계 질의를 처리하는데 적합하지 못하다. 본 논문에서는 스트림 데이터의 발생 시간을 고려하여 인과관계 조인질의를 만족시키기 위한 시간적, 공간적 그리고 시공간적 관계를 살펴보고 이를 반영하여 인과 질의의 성능을 높이기 위한 세 가지 새로운 윈도우의 버퍼링 정책들을 제안하였다. 그리고 다양한 실험들을 통해 본 논문에서 제안하는 방법들이 기존의 FIFO 방법보다 성능이 뛰어남을 보였다. 본 논문의 기여도를 요약하면 다음과 같다;

- 본 논문에서는 센서 스트림에서 인과관계 질의를 제안하였다. 인과관계 질의는 센서 네트워크를 기반으로 한 다양한 응용분야에 활용될 수 있다.
- 본 논문에서는 인과관계 질의를 처리하기 위한 시간적, 공간적 그리고 시공간적 관계를 관찰하여 각각의 조건들을 정의하였다. 이러한 조건들은 원인과 결과의 명확한 인과성을 반영하기 위해 데이터의 도착 시간이 아닌 발생 시간을 고려하였다.
- 시간적, 공간적 그리고 시공간적 고려를 반영한 세 가지 새로운 윈도우의 버퍼링 정책들을 제안하였다. 실험 결과들은 본 논문에서 제안하는 시공간 윈도우 방법이 기존의 FIFO보다 뛰어남을 보여준다.

사 사

본 연구는 2008년 두뇌한국21사업에 의하여 지원되었습니다.

본 연구는 국토해양부 첨단도시기술개발사업 - 지능형국토정보기술혁신 사업과제의 연구비지원(07국토정보C04)에 의해 수행되었습니다.

참고문헌

A. Arasu, B. Babcock, S. Babu, J. Cieslewicz, M. Datar, K. Ito, R. Motwani, U. Srivastava and J. Widom, 2003, Stream: The Standford Data Stream Management System, in IEEE Data Engineering Bulletin, vol.4(1).

B. Babcock, S. Babu, M. Datar, R. Motwani and J. Widom, 2002, “Models and Issues in Data Stream Systems”, in Proceedings of the 2nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles and Data Systems, pp. 1-16.

D. Freedman, 2004, Graphical Models for Causation and the Identifictation Problem, In Evaluation Review, vol. 28(4):267-293.

J. Wu, Kian-Lee Tan and Y. Zhou, 2007, “Window -Oblivious Join: A Data-driven Memory Management Schema for Stream Join”, in Proceedings of the 19th International Conference on Scientific and Statistical Database Management, pp. 21-30.

L. Golab and M. T. Ozsu, 2003, Issues in Data Stream Management, in ACM SIGMOD Record, vol. 32(2):5-14.

L. King, E. A. Rundenstiener and G. T. Heineman, 2003, “MJoin: A Metadata-aware Stream Join Operator”, in Proceedings of the 2nd International Workshop on Distributed Event-based Systems, pp. 1-8.

LTCCS (The Large Truck Crash Causation Study), 2007, http://ai.fmcsa.dot.gov/ltccs/.

M. A. Hammad, W. G. Aref and A. K. Elmagarmid, 2003, “Stream Window Join: Tracking Moving Objects in Sensor-network Databases”, in Proceedings of the 15th International Conference on Scientific and Statistical Database Management, pp. 75-84.

Pearl. J., 2000, Models, Reasoning and Inference, Cambridge University Press.

P. W. Holland, 1986, Statistics and Causal Inference, in Journal of the American Statistical.

S. Chandrasekaran, O. Copper, A. Deshpande, M. J. Franklin, H. Joseph M, W. Hong, S. Krishnamurthy, S. Madden, V. Raman, F. Reiss and M. Shah, 2003, “Telegraphcq: Continuous Dataflow Processing for an Uncertain Work”, in Proceedings of the Conference on Innovative Data Systems Research, pp. 11-18.

Silverstein C., S. Brin, R. Motwani and J. Ullman, 2000, Scalable Techniques for Mining Causal Structures, in Data Mining and Knowledge Discovery, vol. 4(2-3):163-192.

S. Madden and M. J. Franklin, 2002, “Fjording the Stream: An Architecture for Queries over Streaming Sensor Data”, in Proceedings of the 18th International Conference on Data Engineering, pp. 555-566.

T. Urhan and M. J. Franklin, 2000, XJoin: A Reactively-scheduled Pipelined Join Operator, in IEEE Data Engineering Bulletin, vol. 23(2) :27-33.

XinZhou Qin and Wenke Lee, 2003, “Statistical Causality Analysis of INFOSEC Alert Data”, in Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection, pp. 73-93.

김태일, 김계현, 전방진, 곽태식, 2004, “GIS를 이용한 효율적인 가스사과관리 방법에 관한 연구”, 한국 GIS 학회지 제12권, 제1호, pp. 89-100.

황병연, 1999, “다중-속성 색인기법을 이용한 공간조인 연산의 성능”, 한국 GIS 학회지 제7권, 제2호, pp.271-282.

접수일 (2008년 7월 3일)
심사완료일 (2008년 7월 11일)
수정일 (2008년 7월 24일)
게재확정일 (2008년 7월 25일)