데이터 스트림 시스템에서 과거 공간질의 처리를 위한 고속 로딩 기법

신재완*, 백성하*, 이동욱*, 신숭선*, 김경배**, 배해영*
*인하대학교 컴퓨터 정보 공학과
**서원대학교 컴퓨터교육과
{jwshin, shbaek, dwlee, hermit}@dblab.inha.ac.kr, gbkim@seowon.ac.kr, hybae@inha.ac.kr

High-Performance Loading Method for Historical Spatial Query Processing in Data Stream System

Jae-Wan Shin*, Sung-Ha Baek*, Dong-Wook Lee*, Soong-Sun Shin*,
Kyung-Bae Kim**, Hae-Young Bae*
*Dept. of Computer Science and Information Engineering, In-ha University
**Dept. of Computer Education, Korea Seowon University

요 약

무한히 발생되는 실시간 데이터와 디스크에 저장된 히스토리컬 데이터를 동시에 처리하는 하이브리드 질의에 관한 연구가 활발히 이루어 지고 있다. 하이브리드 질의는 디스크에 저장된 대용량의 공간 데이터 처리를 위해 빠른 디스크 입/출력을 요구한다. 이러한 데이터를 처리하기 위해 인덱스, 데이터 축소 기법 등이 연구되었다. 데이터의 빠른 검색을 위한 인덱스 기법은 디스크에 분산 저장된 데이터에 대한 탐색 비용과 입/출력 비용을 줄이지 못한다. 또한, 샘플링을 통해 디스크 입/출력 시간 비용을 줄이는 데이터 축소 기법은 데이터의 정확성을 떨어뜨려 정확성을 요구하는 하이브리드 질의에서는 이용하기 어렵다. 이 논문에서는 디스크 입/출력 시간과 디스크 탐색 시간 비용을 줄이고, 정확성을 보장하는 과거 공간질의 처리를 위한 고속 로딩 기법을 제안한다. 제안기법은 공간을 그리드 형태로 나누고, 인접한 공간 데이터를 함께 관리함으로써 디스크 입/출력 비용을 줄일 수 있다. 또한, 공간적으로 인접한 데이터를 물리적으로 인접한 곳에 저장하여 디스크 탐색 시간 비용을 줄일 수 있다. 이렇게 저장된 데이터는 손실 없이 모두 저장되며, 정확성 또한 보장할 수 있다.

1. 서론

최근 데이터 스트림 관리 시스템(DSMS)에서의 질의는 크게 세가지로 구분된다. 첫 번째, 디스크에 저장된 과거 데이터를 처리하는 히스토리컬 질의(Historical queries). 두 번째, 실시간으로 입력되는 데이터를 처리하는 라이브 질의(live queries). 마지막으로 디스크 데이터와 실시간 데이터를 결합하여 처리하는 하이브리드 질의(Hybrid queries)로 구분된다[1,2,3]. 특히 U-GIS 환경에서는 공간과 결합된 하이브리드 질의 처리를 요구한다[10]. "*인천시에서 온도가 100 도가 넘는 지역(화제가 발생한 지역)의 이름을 출력하시오*"와 같은 질의는 실시간으로 입력되는 온도 데이터와 디스크에 저장된 인천 지역의 공간 데이터를 질의 처리에 이용하는 하이브리드 질의이다. 위의 질의의 경우, 화제가 감지 지역을 찾기 위해서는 화제 지역의 공간 정보를 빠르게 가져와야 한다. 즉, 공간 데이터를 가져오기 위한 디스크 입/출력 작업이 발생한다.

이와 같은, 하이브리드 질의 처리의 성능향상을 위한 기존 연구로 인덱스(Index)를 구축하여 디스크 입/출력 비용을 줄이는 방법 있다. 인덱스 기법은 히스토리컬 데이터에 대한 위치 정보를 인덱스로 구성하여 디스크 입/출력 시간을 줄이는 기법이다. 하지만 새로운 데이터 삽입이 발생하는 경우 인덱스를 재구성하는 작업이 발생하며, 인덱스 정보를 저장하기 위한 유지 비용을 요구하게 된다. 공간 데이터의 인덱스를 구축하는 R-tree 의 경우, 인덱스를 이용하여 데이터에 대한 디스크 입/출력 시간을 줄인다. 하지만 공간 데이터가 저장된 디스크에는 물리적으로 분산 저장되어 있다[4,7]. 데이터의 물리적 분산 저장은 데이터 검색과정에서 긴 디스크 탐색 시간을 요구하는 문제점이 발생한다.

데이터 검색과정에서 발생하는 디스크 입/출력 시간과 디스크 탐색 시간을 줄이기 위한 또 다른 방법으로, 샘플링(Sampling)을 통한 데이터 축소(Data

본 연구는 건설교통부 첨단도시기술개발사업 - 지능형국토정보기술혁신 사업과제의 연구비지원(07 국토정보 C05)에 의해 수행되었습니다.

Reduction) 기법이 있다[2,5]. 데이터 축소 기법은 데이터 샘플링 기법을 이용하여 데이터의 크기를 줄이고, 샘플링된 데이터를 디스크의 인접한 영역 저장 하여 디스크 입/출력 시간과 디스크 탐색 시간을 줄인다[9]. 하지만, 데이터 축소 기법은 원본 데이터가 축소되는 과정에서 데이터 손실이 발생하기 때문에 데이터 정확성이 떨어진다. 위의 질의 예제와 같이 정확한 위치정보 데이터를 요구하는 하이브리드 질의에서 데이터 정확성은 큰 문제점으로 다루어진다.

본 논문에서는 하이브리드 질의 처리과정에서 발생하는 디스크 입/출력 시간, 디스크 탐색 시간, 데이터 정확성 문제 해결하기 위한 기법으로 과거 공간질의 처리를 위한 고속 로딩 기법을 제안한다. 제안기법은 전체 공간 영역을 그리드 형태로 분할한다. 분할된 그리드 영역은 자신의 영역에 속하는 공간 데이터를 함께 관리하며, 디스크의 공간 상의 인접한 곳에 저장한다. 이러한 저장구조는 디스크 입/출력 횟수를 줄임으로써 빠른 디스크 입/출력을 보장한다. 또한, 분할된 공간 데이터를 데이터 손실 없이 저장하여 데이터의 정확성을 보장한다.

본 논문의 구성은 다음과 같다. 2 절에서는 효율적인 하이브리드 질의 처리를 위한 기존 기법을 설명한다. 3 절에서는 본 논문에서 제안하는 효율적인 공간 데이터 처리를 위한 그리드 저장구조를 보인다. 4 절에서는 제안한 기법의 성능 측정 결과를 평가하고, 마지막으로 5 절에서는 본 논문의 결론 및 향후 연구를 보인다.

2. 관련연구

2.1 인덱스(Index) 구축 방법

하이브리드 질의 처리 과정에서 가장 중요시 되는 것은 데이터 처리시간 단축이다. 히스토리컬 데이터의 빠른 디스크 입/출력을 위한 공간 데이터 인덱스 기법으로 R-tree, R* Tree, Quad tree 등이 연구되었다. R-tree 인덱스는 공간적으로 인접한 데이터를 같은 노드로 구성하는 공간 데이터 인덱스 기법이다. 이 기법은 공간적으로 인접한 데이터를 검색하는 경우 빠른 디스크 입/출력 시간을 보여준다. 하지만 R-tree 인덱스 구축을 위해서는 인덱스를 유지하기 위한 별도의 공간을 요구하며, 새로운 데이터 삽입이 발생하는 경우 인덱스 재구성 비용이 발생한다. 또한, R-tree 기법은 데이터 저장 방법에 있어서 데이터의 인접성을 유지하지 못한다. 물리적 인접성은 공간적으로 근접한 곳에 위치한 데이터를 물리적으로 근접한 공간에 저장하는 것이다. 물리적 인접성이 유지된 데이터는 데이터를 읽고 검색하는 과정에서 디스크 탐색 시간을 줄일 수 있다. R-tree 기법은 공간적 인접성을 이용하여 디스크 입/출력 시간을 줄일 수 있지만, 물리적 인접성을 보장하지 못하기 때문에 디스크 탐색 시간에 대한 문제점이 발생한다.

디스크 입/출력 시간과 디스크 탐색 시간을 줄이기 위해서는 공간적으로 인접한 영역에 위치한 공간 데이터를 물리적으로 인접한 영역에 저장하여 디스크 입/출력 시간, 디스크 탐색 시간을 줄일 수 있는 기법이 요구된다.

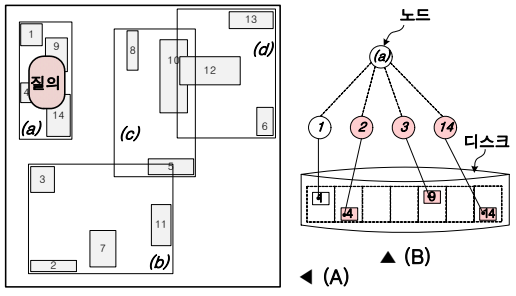
2.2 데이터 축소(Data Reduction) 방법

하이브리드 질의는 실시간 데이터에 대한 빠른 처리를 요구하기 때문에 한번에 처리되는 데이터의 양이 크다. 이렇게 방대한 양의 데이터를 처리하기 위한 또 다른 방법으로 데이터 축소 방법이 있다. 이 방법은 질의 처리에 이용되는 데이터를 정해진 레벨에 따라 0 - 100% 사이의 비율로 축소시킨다. 이렇게 데이터 축소 과정을 거친 데이터는 샘플링이나 윈도우 집계질의(Window aggregation) 처리에 이용된다. 높은 정확성을 요구하지 않는 샘플링 이나 집계 연산 처리에서는 효율적인 방법으로 이용되며[8], 질의 처리시 빠른 디스크 입/출력 시간과 디스크 탐색 시간을 보장해준다. 하지만 데이터 축소 과정 거친 원본 데이터는 데이터의 신뢰성을 떨어뜨리게 된다. 하이브리드 질의에서의 공간 데이터는 높은 신뢰성을 요구하기 때문에 데이터 축소 기법은 하이브리드 질의 처리에 이용되기 힘들다.

3 데이터 스트림 시스템에서 과거 공간질의 처리를 위한 고속 로딩 기법

3.1 데이터 고속 로딩을 위한 저장구조

이번 절에서는 공간 데이터 고속 로딩을 위한 저장구조 (SDCS: Spatial Data Clustered Storage structure)에 대하여 살펴본다. [그림 1] 은 기존 기법에서 질의 처리를 위해 공간 데이터가 탐색 되는 모습을 보여진다. [그림 1] 의 (A)에는 1 ~ 14 까지 14 개의 공간 데이터가 저장되어 있으며, *(a)* ~ *(d)*는 각각 R-tree 에 의해 구성된 노드를 나타낸다. R-tree 는 공간 데이터의 위치에 따라 같은 노드로 구성된다.

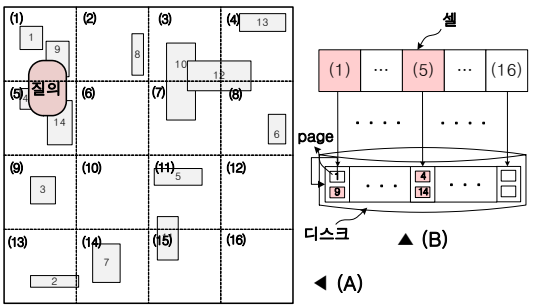


(그림 1) R-tree 에서의 공간 데이터 저장

예를 들어. 노드 *(a)* 는 1, 4, 9, 14 의 공간 데이터를 포함한다. [그림 1] 의 (B)는 노드 *(a)* 에 속하는 공간 데이터가 R-tree 로 구축되어 디스크 에 저장된 모습을 보여준다. 노드 *(a)* ~ *(d)* 는 <MRB, PID, RID> 정보

를 유지 하며, 이 정보는 노드 검색에 이용된다. [그림 1] 에서 질의 처리과정을 살펴보자. 공간 질의가 발생한 경우, 질의 처리를 위해 크게 두 단계의 과정을 거친다. 첫 번째, 질의의 MBR 조건과 노드의 MBR 을 비교하여 질의를 만족하는 공간 데이터가 속한 노드를 찾는다. 두 번째, 노드 검색 후 질의에 해당하는 공간 데이터를 검색한다. [그림 1] 의 (A) 에서와 같은 공간질의가 발생한 경우, 노드 *(a)* 가 선택되어 공간 데이터 4, 9, 14 를 디스크에서 읽어 온다. 하지만 [그림 2] 의 (B) 와 같이 공간 데이터는 임의적으로 분산되어 저장되어 있기 때문에 데이터를 읽어 오는 과정에서 여러 번의 디스크 입/출력과 긴 디스크 탐색 시간이 발생하게 된다.

[그림 2] 의 (A) 는 디스크 입/출력 시간 및 디스크 탐색 시간을 줄이기 위해 제안된 그리드 형태의 저장구조다. 공간 데이터가 입력되는 경우, [그림 2] 의 (A)와 같이 전체 공간을 임의의 그리드로 분할 한다((1)~(16)). 그리드로 분할된 공간을 셀이라고 부른다. 분할된 셀은 각각 새로운 MBR 을 가지며, 셀의 MBR 의 범위 안에 속하는 공간 데이터를 관리하게 된다. 또한 각각의 셀은 자신만의 페이지 공간을 할당 받게 되며, 할당된 공간에 데이터를 저장한다.

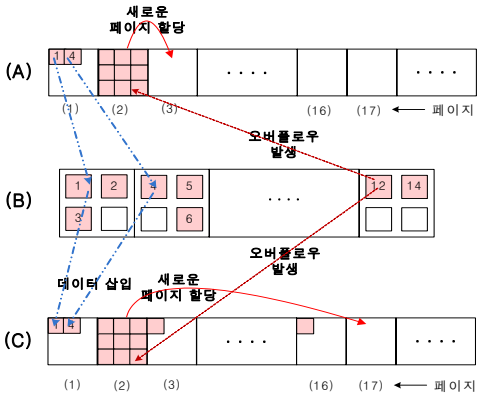


(그림 2) 그리드의 저장구조

이러한 그리드 형태의 저장구조에서 위와 같은 질의가 발생한 경우 다음 단계를 통한 질의 처리과정이 이루어 진다. 첫 번째, 질의의 MBR 조건에 해당하는 셀을 검색한다. 다음으로, 검색된 셀에 해당하는 공간 데이터를 검색한다. [그림 1] 의 (C)에서와 같이 (1), (2) 두 개의 셀이 검색되며, 각 셀은 한번씩의 페이지 탐색으로 해당 질의를 처리할 수 있다. 이와 같은 저장구조는 빠른 데이터 검색과 적은 수의 페이지 접근으로 디스크 입/출력 시간 과 디스크 탐색 시간을 효과적으로 줄일 수 있다.

3.2 SDCS 의 구축과정

본 장에서는 제안기법이 구성되는 과정에 대하여 설명한다. 제안기법을 구성하기 위해서는 디스크에 저장된 기존 데이터의 새로운 구성을 필요로 한다. [그림 3] 은 디스크에 저장된 데이터를 복사하는 과정을 나타낸다. (A), (C) 는 두 가지 형태의 SDCS 의 구조를 나타내며, (B) 는 디스크에 저장된 원본 데이터를 나타낸다. 즉, (A), (B), (C) 는 각각 셀 순차 탐색 방법, 원본 데이터가 저장된 디스크 구조, 디스크 순차 탐색 방법을 나타낸다.



(그림 3) 공간 데이터 삽입 과정

(A), (C) 의 (1) ~ (17) 은 페이지를 나타내며, 각 셀은 데이터 저장을 위해 페이지를 갖게 된다. 만약 공간 데이터 1, 4 가 같은 셀에 속하고, 해당 셀이 (1) 번 페이지를 할당 받게 된 경우, [그림 3] 과 같이 할당 받은 공간에 1, 4 가 저장된다. 같은 방법으로 (B) 의 데이터를 모두 새로운 공간에 저장하게 된다. 이처럼 새로운 구조로 데이터를 저장하는 방법은 탐색 순서에 따라서 디스크 순차 탐색 방법과 셀 순차 탐색 방법이 있다.

디스크 순차 탐색 방법: [그림 3] 의 (C) 는 디스크에 저장된 데이터가 자신이 포함될 셀을 검색한다. 검색된 셀은 연결된 페이지에 데이터를 저장한다. 이 방법은 셀이 자신이 관리한 데이터의 크기를 알 수 없기 때문에 페이지 오버플로우 발생시 물리적으로 연속적인 페이지를 할당 받지 못하게 된다. 결국, 구축시간은 짧지만 셀에 저장된 데이터를 탐색 하는 경우 긴 디스크 탐색 시간이 발생한다.

셀 순차 탐색 방법: 이 방법은 셀을 기준으로 디스크를 순차 탐색하면서 해당 셀에 속하는 공간 데이터를 찾는다. [그림 3]의 (A) 와 같이, (1)번 페이지가 할당된 셀 경우, (1)번 페이지에 속하는 공간 데이터를 디스크에서 검색하여 가져온다. 하나의 셀에 대한 데이터 탐색이 끝난 경우 다음 셀의 탐색이 이루어 진다. 만약, (2)번 페이지를 할당 받아 사용하는 셀의 경우 데이터 검색 중 오버플로우가 발생하게 되면 물리적으로 인접한 페이지를 할당 받게 된다. 이 방법은 셀을 기준으로 디스크에 접근하기 때문에 디스크 순차 탐색 방법에 비하여 구축 시간을 요구한다. 하지만, 구축 완료 후 셀에 저장된 데이터를 탐색하는 경우 짧은 디스크 탐색 시간이 발생한다.

4 성능평가

본 장에서는 제안 기법의 성능평가를 위한 실험 환

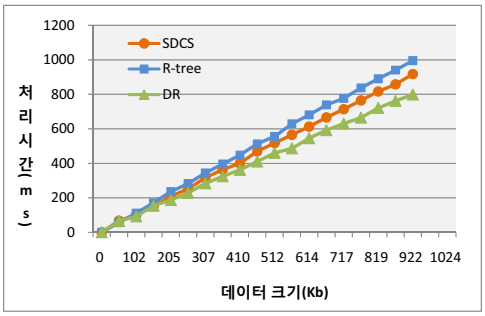
경을 설명한다. 성능 분석은 시간에 따른 디스크 입/출력 횟수 및 데이터 정확성, 공간 조인 처리 속도를 기준으로 기존 기법과 제안 기법을 비교 평가한다.

4.1 실험환경

이 절에서는 실제 실험 환경을 구현하기 위해 제너레이터를 통하여 12,000 개의 공간 데이터를 랜덤으로 생성 하였으며, 8K 크기의 페이지 기반 저장 구조 모델을 사용하여 제안 기법의 저장 구조를 구성하였다. 제안된 그리드 저장 구조는 25X25 개의 그리드로 구성하였다. 데이터 정확성 평가는 25% 의 축소율(75%의 데이터 유지)을 기반으로 평가되었다.

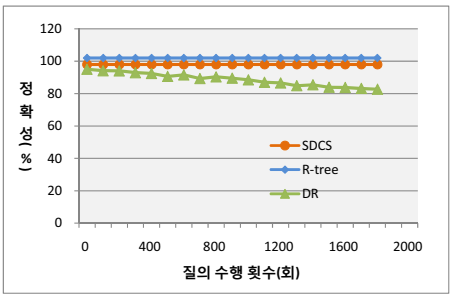
4.2 성능평가

본 성능평가에서는 처리되는 데이터 크기에 따른 데이터 처리 속도와 정확성을 평가하였다. 제안기법은 기존에 제안된 R-tree, 데이터 축소(DR) 기법을 비교 평가하였다.



(그림 4) 데이터 크기에 따른 데이터 처리시간

[그림 4] 는 데이터 크기에 따른 데이터 처리시간을 측정하였다. 성능평가 결과 데이터 크기가 증가함에 DR > SDCS > R-tree 순서로 빠른 처리 결과를 보였다. 제안 기법의 처리시간은 기존의 인덱스 기법인 R-tree 에 비하여 빠른 처리시간을 나타내고 있음을 보인다. 또 다른 성능 분석으로 질의 수행 횟수에 따른 데이터의 정확성을 평가하였다.



(그림 5) 데이터 크기에 따른 정확도

[그림 5]는 질의 수행 횟수에 따른 정확성 평가 결과를 보여준다. 성능 분석 결과 제안 기법과 R-tree 는 100% 정확한 처리 결과를 보인 반면 DR 기법은 데이터 처리 크기가 약 20% 의 데이터 손실이 발생함을 알 수 있다. 데이터 크기에 따른 처리시간과 정확성을 측정한 결과 기존의 인덱스 기법에 비하여 빠른 데이터 처리가 가능하고 정확성 또한 보장함을 보여준다.

5 결론 및 향후 연구

본 논문에서는 하이브리드 질의 처리 과정에서 발생하는 디스크 입/출력 시간과 디스크 탐색 시간을 줄이기 위한 그리드 저장구조 기법을 제안하였다. 제안된 저장 구조는 공간 적으로 인접한 데이터를 그리드 영역으로 구분 관리함으로써 디스크 입/출력 시간을 단축 시켰으며, 공간적으로 인접한 데이터에 대하여 물리적으로 인접하도록 디스크에 저장함으로써 디스크 탐색 시간을 줄일 수 있다. 또한, 데이터 손실이 발생되지 않기 때문에 데이터의 정확성을 요구하는 하이브리드 질의 처리에 적합함을 보였다. 성능 분석 결과 기존 기법에 비하여 질의 처리 속도 측면에서 성능이 향상됨을 보였다.

향후 연구로 제안 기법의 구축 시간을 단축 시키는 기법에 대한 연구가 필요하다.

참고문헌

[1] Madden et al. “Query Processing, Approximation, and Resource Management in a Data Stream Management System”. In CIDR 2003.

[2] Thomas B. “The Impact of Global Clustering on Spatial Database Systems” VLDB. 1994

[3] Sirish C, Michael F. “Remembrance of Streams Past: Overload-Sensitive Management of Archived Streams”. In Proceedings of the 30th VLDB Conference. Canada 2004.

[4] Antomn Guttman. “R-tree: a dynamic index structure for spatial searching”. In Proceeding of ACM-SIGMOD. 1984.

[5] Bronimann dt al. “Efficient DR Methods for On-Line Association Rule Dicovery”.

[6] J M. Hellerstein, et al. “Information under CONTROL: Online Query Processing”. Data Min. Knowl. Discov. 4(4):281-314(2000)

[7] Gisbert D, Schek H J. “Query-Adaptive Data Space Partitioning using Variable-Size Storage Clusters”

[8] B. Babcock, M. Datar and R. Motwani. “Load Shedding for Aggregation Queries over Data Streams”. In Proceeding of the 19th International Conference on Data Engineering. 2004.

[9] Gongde G et al. “Data Reduction Based on Spatial Partitioning”. Springer-Verlag Berlin Heidelberg. 2001.

[10] 이충호, 안경환, 이문수, 김주완, “u-GIS 공간정보 기술 동향,” 전자통신동향분석, ETRI, 2007.