다차원 색인구조에서 효율적인 격리수준 보장 기법

송석일* · 곽윤식*· 유재수**
*충주대학교 전기전자및정보공학부
*충북대학교 전기전자및컴퓨터공학부

Efficient Isolation Level Management Method for Multidimensional Index Structures

Seok Il Song* · Yoon Sik Kwak* · Jae Soo Yoo**
*Chungju National University, Electrical and Information Engineering
*Chungbuk National University, Electrical and Computer Engineering
E-mail : sisong@chungju.ac.kr, yskwak@chungju.ac.kr, yjs@cbnu.ac.kr

요 약

다차원 색인구조가 기존의 데이터베이스 관리시스템에 통합되기 위해서는 모든 격리수준을 보장하는 적절한 동시성 제어 기법이 필요하다. 격리수준을 보장하는 기존의 동시성 제어 기법들이 여럿 제안되었다. 이들은 프레디킷 테이블 기법과 그래뉼러 잠금 기법으로 대표 될 수 있다. 이들은 대체적으로 구현이 어려우며 트리형태의 색인구조에만 적용가능하다. 이 논문에서는 구현이 간단하며 색인구조의 종류에 관계없이 적용이 가능한 기법을 제안한다. 기존의 방법과 비교를 하기 위해 실제로 구현을 하였으며 다양한 환경에서의 실험을 통해 우수성을 입증하였다.

ABSTRACT

In order for multidimensional index structures to be integrated into an existing database management system, proper concurrency control methods that guarantee all isolation levels supported by the database management system. Several concurrency control methods have been proposed. They can be classified into predicate locking based methods and granular locking based methods. Most of them are difficult to implement and can not be applied to non-tree structured index structures. In this paper, we propose a new concurrency control method that guarantee all isolation levels. It is easy to implement and can be applied to any type of index structures. We implement the proposed method and existing methods, and perform various experiments to show the superiority of the proposed algorithm.

키워드
다차원 색인구조, 격리수준, 동시성 제어

Ⅰ. 서 론

지난 20여년간 GIS(Geographic Information System), MLS(Mobile Location Service), CAD (Computer-Aided Design) 와 같은 새로운 데이터베이스 응용들이 주목을 받게 되었다. 이 응용들은 공통적으로 다차원 데이터를 조작한다. 이에따라 다차원의 데이터를 효과적으로 검색하고 접근하는 새로운 색인구조에 대한 요구가 증가하게 되어다. 이 요구를 만족하기 위해서 다양한 다차원 색인구조들이 제안되었다. 데이터 공간을 고정된 영역으로 분할하는 색인구조, 데이터의 분포에 따라 데이터를 분할하는 색인구조, 이 두 방법을 결합한 색인구조 이 외에 트리구조를 따르지 않는 VA-화일이나 해쉬기법을 이용하는 색인구조 등이 있다.

데이터베이스 관리시스템이 위에 언급한 새로운

응용들을 효과적으로 지원하기 위해서는 다차원 색인구조를 시스템에 통합할 필요가 있다. 색인구조를 DBMS에 통합하기 위해서는 동시성 제어와 회복이라는 두 측면을 고려해야 한다. 이중 동시성 제어 기법은 데이터베이스 관리시스템에서 지원하는 격리수준을 모두 지원할 수 있어야 한다. 특히, ISO 와 ANSI SQL 에서는 SERIALIZABLE을 필수 격리수준으로 정하고 있다. 그러나 성능 문제 때문에 대부분의 데이터베이스 관리시스템에서는 격리수준을 4단계로 나누고 SQL 사용자나 응용 프로그래머가 적절한 격리수준을 선택할 수 있도록 하고 있다.

SERIALIZABLE 격리수준이란 유령레코드가 발생하지 않도록 하는 것이며 이를 지원하는 다차원 색인구조용 방법에는 [1, 2, 3] 가 있다. B+-트리에 대해서는 이미 완숙한 단계의 유령레코드 방지기법[4, 5]이 존재한다. 이 기법들은 B+-트리에서는 단말노드에 저장되는 키들이 정렬되어 있다는 특성을 이용한다. 하지만, 다차원 색인구조에서는 B+-트리에서와 같이 키들이 정렬되지 않아서 이 방법들을 적용할 수 없다.

그래서, 다차원 색인구조를 위해서 제안된 방법들은 [4, 5]의 방법대신에 프레디킷 잠금 기법이나 그래뉼러 잠금 기법을 이용하고 있다. 프레디킷 잠금기반의 기법[3]은 이론적으로는 높은 동시성을 제공할 수 있지만 프레디킷을 관리하는 비용이 높아서 그다지 실용적이지 않다. 이에 착안해서 [1, 2]에서는 그래뉼러 잠금 기법에 기반하는 알고리즘을 제안하게 되었으며 실험결과 [3]에 비해서 상대적으로 효과적임을 보여주고 있다.

[1, 2] 역시 단점이 존재한다. 첫 번째는 공간분할의 특성을 갖는 색인구조에는 효과적이지만 가장 널리 사용되는 R-트리계열의 색인구조에서는 노드간의 겹침 현상 때문에 성능이 상대적으로 떨어진다. 다른 하나는 VA-화일이나 해쉬를 이용하는 기법처럼 트리구조가 아닌 색인기법에는 적용할 수 없다.

이 논문에서는 프레디킷 잠금기법과 그래뉼러 잠금 기법을 적절히 혼용한 새로운 동시성 제어기법을 제안한다. 제안하는 방법의 기본 개념은 다차원 데이터 공간을 균등한 크기를 갖는 고정된 개수의 셀들로 나누고 각각의 셀에 유일한 숫자를 부여하여 이를 잠금 대상으로 사용하는 것이다. 제안하는 방법은 색인구조의 노드에 잠금을 획득하는 방식이 아니다. 그러므로, 다차원 색인구조의 형태에 관계없이 적용가능하며 색인구조의 노드에 잠금을 획득하는 SERIALIZABLE 이하의 격리수준을 지원하는 동시성 제어기법들과도 연동이 쉽다. 제안하는 방법의 우수성을 보이기 위해서 다양한 실험을 수행한다. 실험은 제안하는 방법이 응답시간 측면에서 기존 방법보다 우수함을 보인다.

이 논문은 다음과 같이 구성되어 있다. 2장에서는 기존의 SERIALIZABLE 보장 동시성 제어기법에 대해서 설명한다. 3장에서는 제안하는 알고리즘을 설명하고 4장에서는 제안하는 알고리즘에 대한 성능평가 내용을 기술한다. 마지막으로 5장에서 결론을 맺는다.

II. 관련 연구

이미 언급한 대로 다차원 색인구조는 키들이 정렬되어 있지 않으므로 프레디킷 잠금을 이용하는 방법을 제안하였다. 탐색자들은 탐색 프레디킷을 프레디킷 테이블에 기록한 후 탐색을 수행하고 삽입자(삭제자)는 자신이 삽입(삭제)하려는 객체가 프레디킷 테이블에 기록된 프레디킷과 겹치는지 확인한 후 연산을 수행한다.

프레디킷 잠금 기법을 이용하면 탐색자, 삽입자, 삭제자 모두 하나의 잠금만을 획득하면 되므로 잠금에 대한 부담이 적다. 하지만, 프레디킷 테이블을 전체를 조회한 후 해당 프레디킷들에 잠금을 획득하는 과정이 매우 소모적이다. 그리고, 프레디킷 테이블 자체를 유지하는데 드는 비용이 매우 높다. 이런 부담을 경감하기 위해서 [3]에서는 프레디킷 테이블을 전역적으로 관리하지 않고 색인구조의 각 노드에 분산하는 방법을 제안했다. 하지만 이 방법 역시 색인구조의 각 노드에 별도의 공간을 마련해서 프레디킷을 저장해야 하며 프레디킷의 개수가 지속적으로 변하기 때문에 테이블유지에 어려움이 있다. 또한, 노드가 분할되거나 노드의 MBR이 변경되면 프레디킷 테이블의 내용이 같이 변경되어야 한다.

프레디킷 잠금 기법의 한계를 인식한 연구자들은 [1, 2]에서 그래뉼러 잠금 기법을 도입한 알고리즘을 제안하였다. 이 방법에서는 색인구조의 단말노드와 중간 노드들을 잠금 대상으로 한다.탐색자들은 탐색 프레디킷과 겹치는 모든 중간 및 단말노드에 잠금을 획득하고 삽입자(삭제자)들은 객체를 포함하는 모든 노드에 잠금을 획득하게 된다. 또한, 노드의 분할 및 노드의 MBR 변경에 적절히 대처하기 위한 여러 가지 방법들을 포함한다.

이 방법은 색인구조의 노드가 잠금 대상이므로 기존의 데이터베이스에서 제공하는 잠금 기법을 그대로 사용할 수 있다. 그리고, 프레디킷을 저장하기 위한 별도의 테이블이 필요 없다. 하지만, 잠금을 획득한 노드의 MBR이 변경되거나 노드 분할이 발생하게 되면 이를 보완하기 위해서 색인구조에서 변경된 MBR과 겹치는 노드들을 다시 검색해서 잠금을 획득해야한다. 또한, 트리구조가 아닌 색인구조에는 적용이 불가능하다.

이 논문에서는 위에서 제시한 문제를 해결하는 새로운 동시성 제어 기법을 제시한다. 제안하는 방법은 색인구조의 구조와 무관하게 적용이 가능하며 구현이 쉽고 높은 성능을 제공한다.

III. 셀 기반의 동시성 제어기법

제안하는 동시성 제어기법은 사용자가 지정한 b

값에 따라 다차원 데이터 공간을 균등한 크기의 2b개 셀들로 분할한다. 각각의 셀은 길이가 b인 고유한 비트 열로 표현이 가능하다. 각 셀은 같은 크기를 갖고, 전체 셀의 합집합은 전체 데이터 공간과 같다.

이 각각의 셀들이 잠금 대상이 된다. 각 셀에 할당되는 길이 b의 고유한 비트열은 정수로 변환이 가능하며 이 정수 값은 통합되는 데이터베이스 관리시스템의 잠금 식별자로 사용된다. 대부분의 데이터베이스 관리시스템이 정수값을 잠금 식별자로 사용하므로 별도의 잠금 장치를 필요로 하지 않는다. 탐색자들은 탐색 영역과 겹치는 셀들에 공유 잠금을 획득하게 된다. 삽입자(삭제자)들은 삽입(삭제) 하려는 객체의 MBR과 겹치는 셀들에 배타 잠금을 획득한다.

그림 1은 제안하는 알고리즘의 예를 보여주고 있다. 이 예는 데이터 공간의 차원이 2이고 (0, 0), (15, 15)의 영역을 차지하고 있으며 b는 8이라고 가정하자. 그러면 데이터 공간은 28 개의 셀로 분할되게 된다. 그림의 어두운 영역은 한 탐색자의 탐색영역이며 그 범위는 (0, 0), (2, 2) 이다. 탐색 영역과 겹치는 셀들은 모두 9개이다. (0, 0)에서 (1, 1)에 해당하는 셀에 길이 b의 비트 열을 할당하면 00000000 이 되며 이것을 정수로 변환하면 0이 된다. 이런 식으로 나머지 8개의 셀을 정수로 변환하면 1, 2, 16, 17, 18, 32, 33, 34 가 된다. 탐색자는 탐색을 수행하기 전에 먼저 이 정수값에 해당하는 셀에 공유잠금을 획득한다.

이어서, 한 삽입자가 (0, 0)의 값을 갖는 객체를 삽입하려 하면 탐색자와 마찬가지로 객체와 겹치는 셀을 골라내고 이 셀들에 배타 잠금을 설정하게 되는데 이 예에서 객체와 겹치는 셀은 (0, 0), (1, 1) 이며 이를 정수로 변환하면 0 이다. 하지만 이미 탐색자가 이 셀에 공유잠금을 획득하고 있으므로 삽입자는 이 탐색자가 완료할 때 까지 대기한후 잠금을 획득하게 될 것이며 그 이후에 연산을 진행할 것이다.

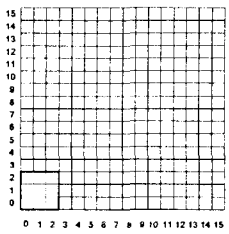

그림 1 탐색 영역의 매핑

탐색자가 획득하는 잠금의 개수는 b 와 탐색영역의 크기에 의존적이다. 한 탐색자가 획득하는 잠금의 개수는 b가 10이고 질의 영역이 전체 데이터 영역에 대한 비율이 0.05 일 때 51개, b가 16일 때는 3237이다. 방식은 간단한 반면 획득해야 하는 잠금의 개수가 너무 많아서 전체적인 성능을 저하하게 된다.

잠금의 개수가 증가하는 문제를 해소하기 위해서 Grid-화일[6]처럼 셀들을 계층적으로 구성한다. 가장 높은 레벨에는 각각의 셀들이 하나의 그룹이 되며 그 다음 레벨에서는 각 차원을 양분하여 셀들을 그룹화하고 다시 그 다음 레벨에서는 각 그룹을 각 차원에 따라서 양분하여 그룹을 만들어 낸다. 이렇게 셀들을 계층적으로 구성한 후에 각 레벨의 각 그룹에 고유한 비트열을 할당한다. 이 비트열은 레벨을 표현하는 최소의 비트와 각 그룹에서 최소의 값을 갖는 셀의 비트열을 합쳐서 구성한다.

탐색영역의 사상은 탐색영역과 겹치는 셀들을 골라낸 다음, 각 레벨별로 선택된 셀들로 구성 가능한 그룹을 찾아낸다. 이 작업은 가장 낮은 레벨에서 높을 레벨로 진행된다. 그림 3은, 탐색자의 탐색 영역 (0, 0), (2, 2)를 잠금 식별자로 사상 하는 예를 보여주고 있다. 탐색영역과 겹치는 셀들을 그림 1에서와 같이 0, 1, 2, 16, 17, 18, 32, 33, 34로 사상하였다. 다음, 선택된 셀들로부터 그룹을 선택해 내는 것이다. 그림에서 판별하기로는 레벨 3에 해당하는 하나의 그룹이 있고 레벨 4에 해당하는 5개의 그룹이 있다. 레벨 3의 그룹은 011(레벨)+00000000(가장 낮은값의 셀 비트열) 의 비트열을 갖게 되며 이것은 768이라는 정수값으로 변환이 된다. 나머지 그룹들도 같은 방법으로 정수값으로 사상된다. 이렇게 하면 잠금의 개수가 그림 1에 비해서 3개가 줄어든 6개이다.

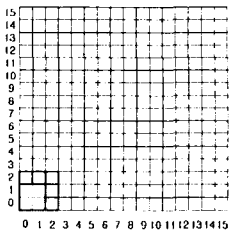

그림 2 계층 방법의 사상

삽입자(삭제자)는 조금 다르게 잠금 대상을 얻어낸다. 이들은 모든 레벨에서 삽입하려는 객체와 겹침이 있는 그룹에 잠금을 획득해야 한다. 즉, (0,0)를 삽입하려면 000+00000000, 001+00000000, 010+00000000, 011+00000000, 100+00000000 에 모두 잠금을 획득한다.

IV. 성능 평가

성능평가는 그래뉼러 잠금 기법을 사용하는 [2]와 제안하는 방법을 구현한 후 두 방법의 성능을 비교하는 방식으로 이루어졌다. 표 1에서는 실험에 사용된 성능평가 파라미터와 값을 보여주고 있다.

표 1 성능평가 파라미터와 값

파라미터	값
데이터베이스 크기	200000 개의 객체
전체 연산중 삽입 비율	20%
질의의 범위	0.8 %
동시수행 프로세스 수	10 ~ 50
비트 수(b)	8(256 cells)

실험에 사용한 데이터는 2~3차원의 균등분포이다. 사용된 데이터의 크기는 200,000 이었다. 실험에 사용된 색인구조는 R-트리였으며 색인구조의 노드 크기는 4KBytes 이었다. 성능평가의 척도로는 먼저 탐색연산이 평균적으로 획득하는 잠금의 개수와 MPL(Multi Program Level)의 변경에 따른 삽입 및 탐색연산의 응답시간이다.

표 2는 탐색연산이 평균적으로 획득한 잠금의 개수이다. 질의 영역이 전체 데이터 공간의 0.2, 0.4, 0.8 일때 탐색연산이 획득했던 잠금의 평균 개수이다. 이때의 b 값은 16 이었다.

표 2 탐색연산의 평균 잠금 개수

질의 범위(%)	0.2	0.4	0.8
잠금의 개수	11.63	13.893	17.445

그림 3 과 4는 동시수행 프로세스의 개수를 변화하면서 측정한 제안하는 기법(PROPOSED)과 그래뉼러 잠금(GL)을 이용하는 기법과의 삽입 및 탐색연산의 응답시간을 보여주고 있다. 동시 수행되는 프로세스의 수가 증가할수록 GL의 응답시간 증가 비율이 PROPOSED에 비해서 커짐을 알 수 있다. 물론 동시 수행 프로세스의 수에 관계없이 전체적인 응답시간이 PROPOSED가 빠르다.

V. 결 론

이 논문에서는 프레디킷 잠금과 그래뉼러 잠금 기법을 적절히 혼용하는 SERIALIZABLE 격리수준을 지원하는 동시성 제어기법을 제안하였다. 제안하는 동시성 제어 기법은 구현이 쉽고 효과적이며 색인구조의 형태에 관계없이 적용이 가능하다. 또한 제안하는 기법의 우수성을 보여주기 위해서 그래뉼러 잠금을 기반으로 하는 기존의 방법과 비교를 수행하였다. 실험결과 제안하는 기법이 기존 방법에 비해서 우수함을 알 수 있었다. 향후 연구에서는 전체 데이터 영역이 동적으로 변하는 상황에서도 적절히 대처할 수 있도록 제안하는 방법을 개선한다.

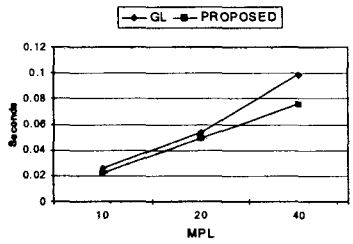

그림 3 탐색연산의 응답시간

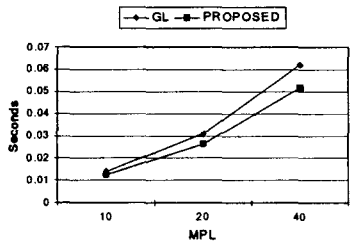

그림 4 삽입연산의 응답시간

참고 문헌

[1] K. Chakrabarti and S. Mehrotra, "Dynamic Granular Locking Approach to Phantom Protection in R-Trees," Proceedings of ICDE, 1998, pp. 446-454.

[2] K. Chakrabarti and S. Mehrotra, "Efficient Concurrency Control in Multidimensional Access Methods," Proceedings of ACM SIGMOD, 1999, pp. 25-36.

[3] M. Kornacker, C. Mohan and J. M. Hellerstein, "Concurrency and Recovery in Generalized Search Trees," Proceedings of ACM SIGMOD, 1997, pp. 62-72.

[4] C. Mohan, "ARIES/KVL: A Key Value Locking Method for Concurrency Control of Multiaction Transactions Operating on B-Tree Indexes,"Proceedings of VLDB, 1990, PP. 392-405.

[5] C. Mohan and F. Levine, "ARIES/IM: An Efficient and High Concurrency Index Management Method Using Write-Ahead Logging," Proceedings of ACM SIGMOD, 1992, pp. 371-380.

[6] J. Nievergelt, H. Hinterberger, and K. C. Sevcik, "The Grid File: An Adaptable, Symmetric Multikey Structure,"ACM Transactions on Database Systems, Vol. 9, No. 1, 1984, pp. 38-71.