제25회 한국정보처리학회 춘계학술발표대회 논문집 제13권 제1호 (2006. 5)

공간 데이터 웨어하우스에서 개념 계층을 지원하는 공간 데이터 큐브

옥근형*, 이동욱*, 유병섭*, 배해영*
*인하대학교 컴퓨터 · 정보공학과
e-mail : ghsy0718@dblab.inha.ac.kr

A Spatial Data Cubes with Concept Hierarchy on Spatial Data Warehouse

Geun-Hyoung Ok*, Dong-Wook Lee*, Byeong-Seob You*, Hae-Young Bae*
*Dept. of Computer and Information Engineering, Inha University

요 약

데이터 웨어하우스에서는 OLAP(On-Line Analytical Processing) 연산을 제공하기 위해 다차원 데이터를 큐브의 형태로 관리한다. 특히, 공간 차원과 같이 데이터 큐브의 차원에 개념 계층이 존재하는 경우 사용자는 특정 계층에 대한 집계 결과를 요구한다. 기존의 데이터 큐브의 구조들은 차원의 개념 계층을 지원하지 못하거나 지원하더라도 시간이나 공간적 비용에 대해 비효율적이다.

본 논문에서는 공간 데이터 웨어하우스에서 공간 개념 계층을 이용하여 효율적인 계층별 영역 집계연산을 지원하는 공간 데이터 큐브를 제안한다. 이는 개념 계층을 DAG(Directed Acyclic Graph) 형태로 표현하여 구성된 여러 개의 차원들을 공간차원의 지역성을 기준으로 연결한 구조이다. 이러한 구조를 갖는 큐브를 이용하면, 데이터 검색 시 상위 계층부터 아래 방향으로 탐색하기 때문에 각 차원에 대한 효율적인 검색이 가능하다. 특히, 공간 개념 계층에 대한 DAG 를 이용하면, 공간적 지역성에 따른 영역 검색을 지원할 수 있다. 성능평가에서 개념 계층이 적용된 질의에 대한 실험을 통해 제안 기법이 기존 기법들에 비해 저장 공간 효율성 및 질의 응답 성능이 우수함을 증명한다.

1. 서론

공간 데이터 웨어하우스는 이질적인 데이터 소스로부터 추출된 공간/비공간 데이터를 하나의 저장소에 통합하여 데이터 분석가에게 OLAP 연산을 제공하는 정보 시스템이다. 공간 데이터 웨어하우스에 통합 및 저장되어있는 다차원 데이터는 효율적인 집계 연산을 위해 큐브의 형태로 관리된다. 이때, 개념 계층을 갖는 차원이 존재할 경우 사용자는 특정 차원에 대한 롤업 및 드릴다운 연산을 통해 원하는 계층의 집계값을 요청한다[1, 2]. 특히, 공간 데이터 웨어하우스에서는 공간 차원을 이용한 데이터 분석 질의가 빈번하기 때문에 공간의 개념 계층을 효율적으로 지원하는 공간 데이터 큐브의 필요성이 대두되고 있다.

최근까지 제안된 데이터 큐브들을 살펴 보면, 개념 계층 스키마를 큐브 외부에 저장하여 OLAP 연산시 데이터 큐브 이외에 외부의 데이터를 접근해야 하는 단점이 있었다. 또한, 계층 스키마를 큐브 내부에 저장하는 데이터 큐브가 존재하지만 이는 공간 차원을 지원하지 못한다.

본 논문은 공간 데이터 웨어하우스에서 개념 계층을 지원하는 공간 데이터 큐브를 제안한다. 이는 공간/비공간 차원의 개념 계층 구조를 DAG 형태로 구성한 후, 각 차원의 DAG 를 공간 차원의 지역성을 기준으로 연결하여 개념 계층을 내부에 포함하는 공간 데이터 큐브이다. 이를 통해 계층별 영역집계연산을 효율적으로 제공할 수 있으며 데이터 큐브를 원시적인 다차원 배열에 저장하는 것에 비해 저장 공간 비용을 크게 줄일 수 있다.

본 논문의 구성은 다음과 같다. 2 장에서는 관련 연구에 대해 살펴보고, 3 장에서 개념 계층을 지원하는 공간 데이터 큐브의 구조를 정의한 후, 4 장에서 이를

*본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT 연구센터 육성·지원사업의 연구결과로 수행되었음.

이용한 질의의 예와 그 결과 값을 구하는 알고리즘에 대해 서술한다. 5 장에서는 제안 기법의 성능평가 결과를 분석하고, 6 장에서 결론을 맺고 향후 연구 과제에 대해 논한다.

2. 관련 연구

기존의 데이터 큐브에 대한 관련 연구는 다차원 데이터에 대한 인덱싱을 통해 큐브를 저장하는 기법과 데이터 큐브의 접두 및 접미 중복을 제거하여 큐브를 압축하는 기법이 있다.

인덱스를 사용하여 큐브를 저장하기 위해 Cubetrees, QC-trees 가 제안되었다[3, 4]. 이들은 효율적인 큐브 집계연산을 위해 제안되었지만 개념 계층을 지원하지 않는다. 한편, DC-trees 는 개념 계층을 지원하지만 대용량 큐브의 압축에 대해 고려하지 않아 저장공간 비용이 크다는 단점이 있다[5]. 공간 차원의 계층을 이용하는 큐브 인덱싱 기법으로는 OLAP-Favored Search Tree 가 있다[6]. 이는 기본적으로 공간 인덱스인 R-tree 의 구조를 개선시킨 것으로서, 각각의 하위 계층에 존재하는 모든 객체에 대한 집계 값을 각 노드에 저장하여 질의 처리시 질의 영역에 포함되는 중간 노드에 대해서는 그 하위 노드를 검색하지 않고 직접 집계 값을 구할 수 있다는 장점이 있다. 그러나 OLAP-Favored Search Tree 는 과거로부터 현재까지의 총 집계 값만을 지원하며, 시간 차원이나 이외의 차원에 대한 OLAP 연산은 지원하지 못한다.

큐브의 압축을 위해 접두 및 접미 중복을 제거하여 저장공간의 낭비를 줄이는 기법으로 Dwarf 가 제안되었다. 데이터 큐브는 구조상 실제로 그 데이터가 존재하지 않더라도 모든 가능한 조합의 데이터를 위한 저장공간을 차지한다. 하지만 Dwarf 는 다차원 데이터가 갖는 차원상의 접두 및 접미 중복을 제거하여 DAG 형태의 큐브를 구성하는 무손실 압축기법이다. 이를 응용하여 개념 계층을 지원하도록 변경된 구조의 Hierarchical Dwarf 가 제안되었다[7]. 그러나 Hierarchical Dwarf 는 본래 Dwarf 의 기본적인 구조와 장점을 최대한 유지하기 위해 형식적으로 개념 계층을 지원할 뿐이며, 공간에 대한 개념 계층은 고려하지 않는다. 또한, 서로 다른 상위 개념에 속하는 하위 개념들 중 속성 값이 같은 것이 발생할 경우 Hierarchical Dwarf 는 잘못된 연산 결과를 출력하는 문제점이 있다.

본 논문에서는 공간 데이터 큐브의 개념 계층 스키마를 효율적으로 구성하는 구조를 제안하여 큐브의 연산 시간 및 저장 공간 비용에 대한 성능을 향상시키는 기법을 기술한다.

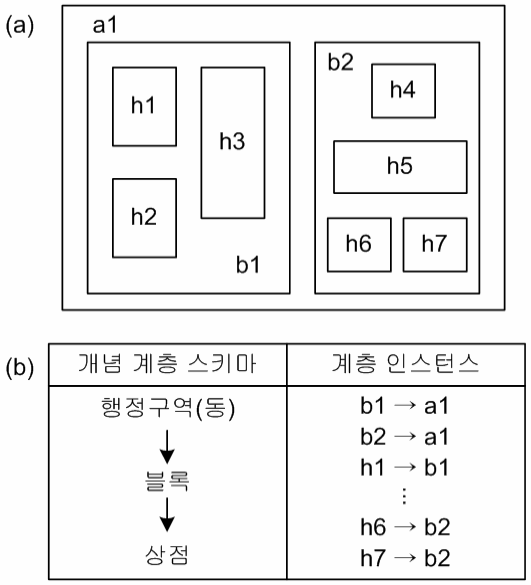
3. 개념 계층을 지원하는 공간 데이터 큐브

이 장에서는 공간 데이터 큐브의 생성과정을 통해 그 구조를 정의한다. 이에 앞서, 개념 계층 스키마가 공간 데이터 웨어하우스에 메타 데이터의 형태로 정의 및 저장되어있다고 가정한다. 큐브로 만들고자 하는 사실 테이블의 예는 <표 1>과 같고, '상점', '날짜', '판매 물건'과 같이 3 개의 차원이 존재한다.

<표 1> 각 상점의 판매에 대한 사실 테이블 SALES

상점	날짜	판매 물건	판매 수량
h1	06-01-01	p1	10
h1	06-01-02	p2	20
h2	06-01-01	p1	5
h2	06-01-03	p3	10
h5	06-01-01	p2	30
h7	06-01-02	p2	25
h7	06-01-03	p1	15

공간 차원과 그에 대한 공간 개념 계층 스키마의 예는 (그림 1)과 같다. 이는 'a1'동에 있는 'b1', 'b2'블록의 7 개에 상점에 대한 데이터 이다.

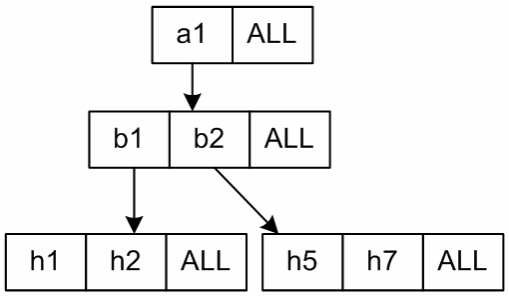


(그림 1) 공간 차원의 예(a)와 개념 계층 스키마 메타 데이터(b)

공간 데이터 큐브를 생성하기 위해 사실 테이블 'SALES'의 각 차원에 대해 개념 계층 DAG 를 순서대로 구성한 후, 이들을 사실 테이블의 내용을 토대로 연결한다. 먼저 사실 테이블에서 상점 차원에 대해 {h1, h2, h5, h7}과 같이 필드 값의 집합을 얻는다. 이 집합에 대해 (그림 1)에 나타나있는 메타 데이터를 참조하여 이들의 상위 개념 집합을 얻는다. 이는 {b1, b2}와 같고, 마찬가지로 이에 대한 상위 개념 집합은 {a1}이 된다. 이렇게 얻은 정보를 토대로 (그림 2)와 같이 개념 계층 DAG 를 구성한다. 각 계층은 하나 이상의 노드로 구성되며, 각 노드에는 속성 값과 다른 노드를 가리키는 포인터로 구성된 여러 개의 셀이 있다. (그림 2)에는 'a1'과 같은 아이디로 표현되었지만 실제로는 이에 해당하는 공간 데이터가 저장된다. 또한 모든 노드에는 노드에 나타난 모든 필드 값에 대한 집계 값을 구할 때 사용할 'ALL' 셀이 추가된다.

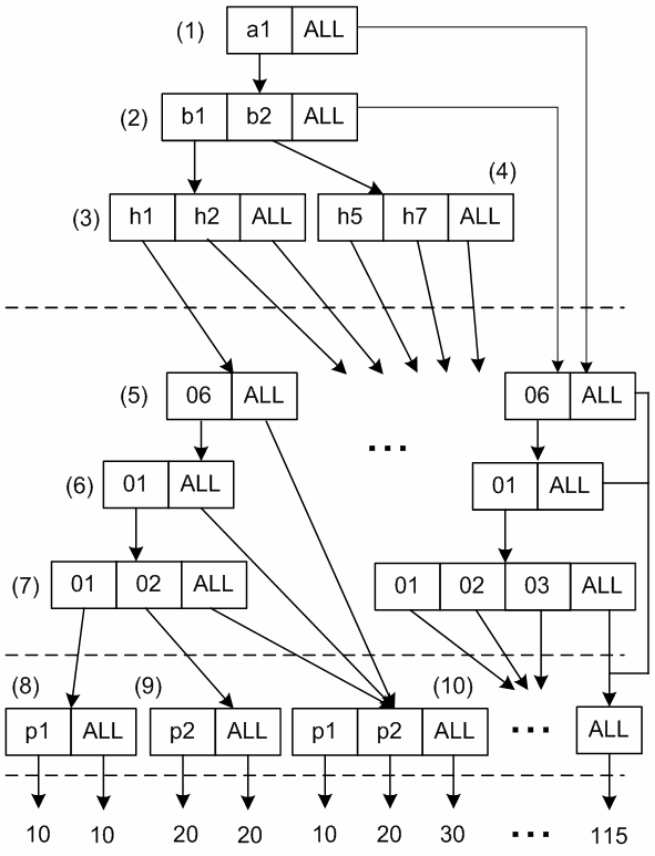
그리고 '날짜' 차원에 대해서 같은 방법으로, 선행하는 '상점' 차원 DAG 의 각 단말노드와 모든 'ALL'

셀마다 하나씩 '날짜' 차원 DAG 를 구성한다.



(그림 2) '상점' 차원의 개념 계층 DAG

'날짜' 차원에는 '년→월→일' 과 같은 개념 계층이 있고 '판매 물건' 차원에는 편의를 위해 개념 계층이 없다고 가정한다. 이들을 사실 테이블의 내용을 토대로 연결하고 집계 함수 'SUM(판매 수량)'의 값을 저장하여 완성된 큐브는 (그림 3)과 같다. 이는 (그림 2)의 공간 개념 계층 DAG 의 단말 노드의 모든 셀과 모든 ALL 셀에 대해 해당하는 날짜들의 집합에 대해 개념 계층 DAG 를 구성하여 연결시킨 후, '날짜' 차원에 대해서 같은 방법으로 해당하는 '판매 물건'을 연결하고 마지막으로 '판매 수량'에 대한 집계 값을 계산하여 '판매 물건'과 연결한 것이다.



(그림 3) 사실 테이블 SALES 에 대한 데이터 큐브

각 차원은 점선으로 구분되어 있으며, 위에서부터 '상점', '날짜', '판매 물건' 그리고 집계 값인 '판매 수량'의 순서로 나타난다. 논문의 공간상 제약으로 큐브 구조를 이해하기 위한 최소한의 그래프만 표현하였다.

본 논문에서 제안하는 공간 데이터 큐브는 각 차원의 모든 계층에 대해서 집계 연산을 지원하며, 공간 차원인 '상점'의 DAG 는 공간 인덱스인 R-tree 의 부모 노드의 영역이 모든 자식 노드의 영역을 포함한다는 면에서 비슷한 특성을 가지므로 큐브에 대한 효율적인 공간 영역질의가 가능하다. 또한, '날짜' 차원에서 1 월의 1 일과 2 월의 1 일과 같이 서로 다른 상위 개념에 속하면서 중복되는 필드 값에 대해 올바른 연산이 가능하다.

4. 공간 데이터 큐브를 통한 질의 처리의 예

이 장에서는 제안된 큐브를 이용하여 공간 개념 계층 질의가 처리되는 과정을 예를 들어 설명한다. 공간 질의의 형태는 공간 차원에 대한 점 질의와 영역 질의로 나누어지며, 사용자는 각 차원에 대해 원하는 계층을 설정한 후 질의한다. (그림 3)의 큐브에 대한 점 질의의 예를 들면, "h1 상점에서 2006 년 1 월에 판매한 모든 물건의 수량"을 구하는 질의는 "*QPoint* (상점), 06-01(월), ALL"와 같다.

QPoint 는 공간상에서 사용자가 질의하고자 하는 점을 나타내며 이는 h1 상점의 영역에 포함된다고 가정한다. 이러한 질의가 처리되는 과정을 (그림 3)의 각 노드에 매겨진 번호를 이용하여 설명하면, 첫 번째 차원인 '상점' 차원에 대해 상점 계층까지 R-tree 와 같은 알고리즘으로 탐색하여 *QPoint* 가 위치하는 (3)번 노드의 'h1'을 찾아서 이에 연결되어 있는 (5)번 노드로 이동한다. 시간 차원에 대해서는 2006 년 1 월을 찾아야 하므로 (5)번 노드의 '06'에 연결되어있는 (6)번 노드로 이동하여 다시 '01'에 연결된 (7)노드로 이동한다. 위에서 제시된 질의는 1 월 전체에 대한 집계 결과를 요구하므로 (7)노드의 'ALL'에 연결된 (10)번 노드로 이동한다. 모든 판매 물건에 대한 질의이므로 (10)번 노드의 'ALL'에 연결된 '30'이 위 질의의 결과로 출력된다. 그리고 영역 질의의 경우 점 질의의 *QPoint* 대신 사용자가 지정한 질의 영역을 이용한다. 예를 들어, 사용자가 상점 차원의 개념 계층을 '블록'으로 설정하였고, 지정된 질의 영역이 'b1'과 'b2'에 겹칠 경우, (1)번 루트 노드로부터 블록 계층까지 영역 검색을 하여 'b1'과 'b2'를 찾아내고 이 영역 질의는 'b1'과 'b2'에 대한 두 개의 점 질의로 나뉘어 처리된다. 즉, (3)번 노드의 'ALL'과 (4)번 노드의 'ALL'에 대한 집계 값을 위에서 설명한 것과 같은 방법으로 각각 가져와 두 질의의 결과 값을 합산하여 본래 영역 질의에 대한 집계 결과를 얻을 수 있다.

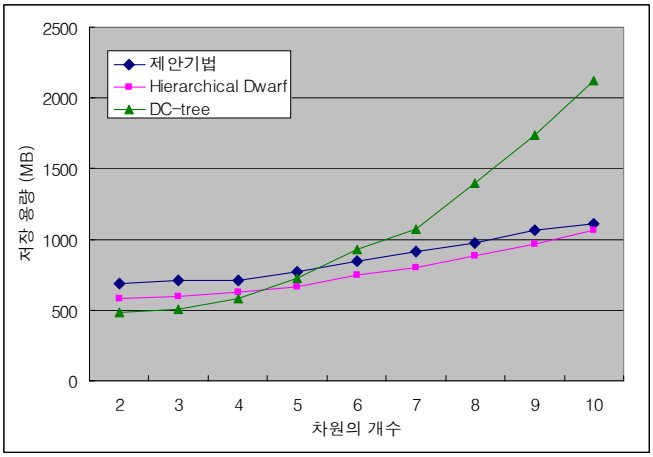
5. 성능평가

본 장에서는 제안 기법을 통해 구성된 큐브의 저장 공간 비용 및 질의 응답 시간을 측정하여 큐브의 확장성 및 효율성을 평가한다. DC-tree, Hierarchical Dwarf, OLAP-Favored Search Tree 를 제안 기법과 비교 분석하여 평가한다[5, 6, 7]. 실험 환경은 <표 2>와 같다.

<표 2> 실험 환경

중앙처리장치	Pentium4 3.0Ghz
주 기억장치	1GB
보조 기억장치	150GB
운영체제	Windows XP Professional
개발 언어	C/C++

첫 번째 실험평가에서는 데이터 큐브의 확장성을 분석하기 위해 개념 계층을 갖는 차원의 개수를 변화시키면서 100,000 개의 레코드를 갖는 사실 테이블에 대한 큐브를 생성하여 저장 비용을 측정하였다. 공간 차원으로 서울특별시 강남구 지번도 데이터를 사용하였으며 주소에 대한 개념 계층을 적용하였다. DC-tree 와 Hierarchical Dwarf 는 공간 데이터를 지원하지 못하므로 비공간 속성인 주소를 링크로 사용해 공간 데이터에 접근하는 방식을 이용하였다. 측정 결과는 (그림 4)와 같다.

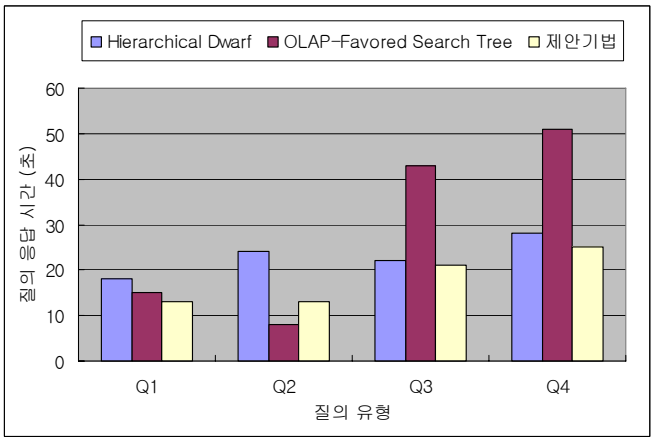


(그림 4) 차원의 개수에 따른 큐브의 저장용량 비교

DC-tree 를 이용한 큐브는 데이터에 대한 압축을 지원하지 않기 때문에 개념 계층의 개수가 늘어남에 때라 저장용량이 급격히 증가하였다. 반면, 제안 기법에 의해 생성된 데이터 큐브의 크기는 레코드의 개수가 적을 때에는 효율적인 무손실 압축을 지원하는 Hierarchical Dwarf 에 비해 다소 크게 나타났지만, 레코드의 개수가 늘어남에 따라 점점 차이가 줄어들었다. 이는 제안 기법이 구조상 Hierarchical Dwarf 에 비해 다소 많은 포인터를 갖게 되기 때문이다.

다음으로 데이터 큐브에 대한 질의 응답 시간을 측정하기 위해 3 가지 큐브에 대해 전체 150,000 개의 레코드에 대한 4 가지 개념 계층 질의를 사용하였다. (그림 5)의 'Q1'과 'Q2'는 각각 공간 차원에 대한 점 질의와 전체의 5%를 선택하는 영역 질의이며, 'Q3'과 'Q4'는 각각 공간을 포함하지 않는 차원에 대한 선택 질의와 범위 질의이다.

4 가지 질의를 각각 1000 번 실행하는데 걸리는 각 큐브의 질의 응답 시간은 (그림 5)에 나타나있다. Hierarchical Dwarf 는 데이터 검색 시 개념 계층에 대해 bottom-up 방식으로 검색하기 때문에 모든 하위 개념들을 접근한다. 하지만 제안 기법은 top-down 방식으로 검색하므로 각 차원에 대해 인덱스를 구축한 효과를 갖기 때문에 검색비용이 적어 모든 질의에 대해 상대적으로 우수한 성능을 보였다. 또한, 'Q2'에 대해서는, 이에 가장 최적화되어 있는 OLAP-Favored Search Tree 가 가장 좋은 성능을 보였다. 그러나 'Q3','Q4'에 대해서는 원본 데이터에 접근하기 때문에 가장 낮은 성능을 보였다.



(그림 5) 질의 유형별 질의 응답 시간

6. 결론 및 향후 연구

본 논문에서는 공간/비공간 차원에 대한 개념 계층을 지원하는 공간 데이터 큐브의 구조를 제안하였다. 이는 기존 큐브들의 문제점들을 개선하고 공간 차원에 대한 개념 계층을 지원하는 데이터 큐브로서, 보다 정확하고 효율적인 공간 OLAP 연산을 제공한다. 성능평가에서 Hierarchical Dwarf 보다는 저장 비용이 다소 크지만, 공간 데이터를 포함하는 여러 가지 유형의 질의에 대한 응답 속도가 대체로 향상 되었다.

향후 연구는 다중 개념 계층구조를 지원하기 위한 연구가 진행되어야 할 것이며, 제안된 큐브의 점진적 업데이트 기법에 대한 연구가 필요하다.

참고문헌

[1] E.F. Codd, S.B. Codd, C.T. Sally, "Provision OLAP to User-Analysts: An IT Mandate", Technical Report, 1993.

[2] S. Chaudhuri, U. Dayal, "An Overview of Data Warehousing and OLAP Technology", ACM SIGMOD, 1997.

[3] N. Roussopoulos, Y. Kotidis, M. Roussopoulos, "Cubetree: Organization of and Bulk Incremental Updates on the Data Cube" ACM SIGMOD, 1997.

[4] L.Lakshmanan, J. Pei, Y. Zhao, "QC-Trees: An Efficient Summary Structure for Semantic OLAP", ACM SIGMOD, 2003.

[5] M. Ester, J. Kohlhammer, H.-P. Kriegel, "DC-tree: A Fully Dynamic Index Structure for Data Warehouses", IEEE ICDE, 2000.

[6] F. Rao, L. Zhang, X.L. Yu, Y. Li, Y. Chen, "Spatial Hierarch and OLAP-Favored Search in Spatial Data Warehouse", ACM DOLAP, 2003.

[7] Y. Sismanis, A. Deligiannakis, Y. Kotidis, N. Roussopoulos, "Hierarchical Dwarfs for the Rollup Cube", ACM DOLAP, 2003.