영역-그룹화 질의 계산 알고리즘
(An Algorithm for Computing Range-Groupby Queries)

이 영 구 † 문 양 세 †† 황 규 영 †††
(Young-Koo Lee) (Yang-Sae Moon) (Kyu-Young Whang)

요 약 온라인 분석처리(On-Line Analytical Processing: OLAP)에서 집계 연산은 중요한 기본 연산이다. 본 논문에서는 OLAP에서의 집계 질의 중 영역-그룹화(range-groupby)라는 새로운 클래스의 질의를 정의하고, 이 질의의 처리 방법을 제시한다. 영역-그룹화 질의는 n-차원 데이타 큐브의 임의의 영역에 속한 셀들에 대하여 주어진 그룹화 속성들의 조합에 따라 집계 값을 구하는 질의이다. 이 질의는 관심의 대상이 되는 임의의 영역 내에서의 경향을 다각적인 측면에서 분석하기 위해서 OLAP에서 자주 사용되는 질의이다. 일반적으로, OLAP에서는 질의를 빠르게 처리하기 위하여 전방-합 배열(prefix-sum array)이라 불리는 집계 결과를 미리 계산하여 유지하는 선계산 기법이 실제적으로 널리 사용되고 있다. 그런데, 영역-그룹화 질의의 경우에는, 그룹화 속성들의 모든 조합에 대하여 집계 결과를 저장해야 하기 때문에, 저장 공간 오버헤드가 너무 크다. 본 논문에서는 가능한 적은 공간 오버헤드를 가지고 영역-그룹화 질의를 빠르게 처리할 수 있는 방법을 제안한다. 제안한 방법은 단지 하나의 전방-합 배열만을 유지하면서도, 가능한 모든 그룹화 속성의 조합에 대하여 영역-그룹화 질의를 효율적으로 처리한다. 이 방법은 가능한 모든 그룹화 속성들의 조합에 대하여, 전방-합 배열을 선계산하여 유지하는 방법과 비교할 때, 액세스되는 셀의 개수는 비슷하면서 공간 오버헤드는 $O(\frac{1}{2^n})$(n은 디멘젼의 개수)로 줄인다.

키워드 : 온라인 분석 처리, 집계 연산, 영역 질의, 전방-합 배열

Abstract Aggregation is an important operation that affects the performance of OLAP systems. In this paper, we define a new class of aggregation queries, called range-groupby queries, and present a method for processing them. A range-groupby query is defined as a query that, for an arbitrarily specified region of an n-dimensional cube, computes aggregations for each combination of values of the grouping attributes. Range-groupby queries are used very frequently in analyzing information in MOLAP since they allow us to summarize various trends in an arbitrarily specified subregion of the domain space. In MOLAP applications, in order to improve the performance of query processing, a method of maintaining precomputed aggregation results, called the prefix-sum array, is widely used. For the case of range-groupby queries, however, maintaining precomputed aggregation results for each combination of the grouping attributes incurs enormous storage overhead. Here, we propose a fast algorithm that can compute range-groupby queries with minimal storage overhead. Our algorithm maintains only one prefix-sum array and still effectively processes range-groupby queries for all possible combinations of the grouping attributes. Compared with the method that maintains a prefix-sum array for each combination of the grouping attributes in an n-dimensional cube, our algorithm reduces the space overhead by $O(\frac{1}{2^n})$, while accessing a similar number of cells.

Key words : on-line analytical processing(OLAP), aggregation, range query, prefix-sum array

· 본 연구는 첨단정보기술연구센터를 통하여 한국과학재단의 지원을 받았음.
† 종신회원 : 한국과학기술원 전자전산학과
yklee@mozart.kaist.ac.kr
†† 비 회 원 : 한국과학기술원 전자전산학과
ysmoon@mozart.kaist.ac.kr
††† 종신회원 : 한국과학기술원 전자전산학과 교수
kywhang@cs.kaist.ac.kr
논문접수 : 2002년 1월 28일
심사완료 : 2002년 5월 8일

1. 서 론

온라인 분석처리(On-Line Analytical Processing: OLAP)는 사용자가 의사 결정에 필요한 지식을 찾아내기 위해 대량의 데이타를 쉽게 분석할 수 있도록 도와주는 데이타베이스 응용이다[1,2]. 의사 결정에 있어서는

개별적인 레코드들보다 레코드들을 요약한 경향이 중요하기 때문에, 상당수의 OLAP 질의들은 레코드들을 요약하는데 사용하는 집계(aggregate) 연산을 포함하고 있다. 그리고, 집계 연산들은 처리 비용이 매우 큰 연산이기 때문에 집계 연산의 처리 성능은 OLAP 시스템의 성능에 큰 영향을 미치는 중요한 요소이다[3,4,5,6].

OLAP에서는 데이타를 다차원 배열로 모델링하는 다차원 데이타 모델을 사용한다[2,7]. 다차원 데이타 모델에서는 데이타를 분석의 대상이 되는 **메져**(*measure*)들과 메져를 결정하는 **디멘젼**(*dimension*)으로 구분한다. 그리고, 각 디멘젼을 다차원 배열의 하나의 축(axis)으로 대응시키고, 메져를 배열의 셀(cell)에 저장된 값으로 대응시킨다. 이러한 다차원 배열을 **데이타 큐브**(*data cube*)라 부른다. 데이타 큐브는 디멘젼들의 값의 변화에 따른 메져의 변화를 분석하는 OLAP 응용에 적합하다고 알려져 있다[2].

OLAP 응용은 사용자가 대화식으로 정보를 분석하는 시스템이므로, 질의 처리 시 빠른 응답 시간을 요구한다. 이러한 요구를 만족시키기 위해, 선계산된 결과를 유지하는 방법이 실제적으로 널리 사용되고 있다. Gray 등[8]에 의해 제안된 CUBE BY 연산은 가능한 모든 조합의 디멘젼들을 그룹화 속성으로 하는 groupby들을 모두 구하는 연산이다. CUBE BY 연산이 제안된 이후, CUBE BY의 결과 (또는 그 일부)를 선계산하여 유지하고 이를 이용하는 연구가 많이 이루어졌다[3,9,10,11]. CUBE BY 연산의 결과는 각 디멘젼의 도메인에 'all'이라는 특수한 값이 추가된 큐브에 저장될 수 있다[8]. 본 논문에서는 이 특수한 값을 갖도록 확장된 데이타 큐브를 **확장 데이타 큐브**(*extended data cube*)[6]라 부른다. n개의 디멘젼 D_1, D_2, ..., D_n으로 구성된 데이타 큐브는 $\prod_{i=1}^{n}|D_i|$($|D_i|$는 디멘젼 D_i의 카디날리티(cardinality)) 개의 셀을 갖는 배열로 표현되고, 이에 대한 확장 데이타 큐브는 $\prod_{i=1}^{n}(|D_i|+1)$개의 셀을 갖는 배열로 표현된다. 그리고, D_{g_1}, D_{g_2}, ..., D_{g_m}을 그룹화 속성으로 하는 집계 질의의 결과들은, 그룹화 속성을 제외한 속성, 즉 **비 그룹화 속성**들의 값이 'all'인 확장 데이타 큐브의 셀에 저장된다.

OLAP에서는 데이타 큐브의 임의의 영역을 대상으로 하는 집계 질의가 자주 사용된다. 본 논문에서는 이러한 질의를 **영역-집계 질의**(*range-aggregation query*)라 정의한다. 이 질의에는 각 디멘젼에 대하여 임의의 연속적인 범위의 값들이 조건으로 주어진다. 이러한 조건을 **범위 조건**(*range condition*)이라 정의한다. 범위 조건은 나이, 수입, 시간 등과 같이 순서를 부여하는 것이 자연스러운 의미를 갖는 숫자형 디멘젼에 대해서 빈번하게 발생한다[4,5,6]. 예를 들어, 보험 회사에 대한 OLAP 응용에서 나이(age), 수입(income), 년도(year)를 디멘젼으로 하고, 판매량(sales)을 메져로 하는 데이타 큐브를 생각해 보자. 이 때, 나이가 30~39이고, 수입이 100,000~200,000 인 사람에게 1990~1999년에 판매한 보험을 년도 별로 구분하여 구하는 질의는 영역-집계 질의이다. 이러한 질의는 자료를 다각적인 측면에서 분석하기 위하여 OLAP 응용에서 매우 유용하게 사용된다.

영역-집계 질의는 그룹화 속성의 유무에 따라 두 가지로 분류할 수 있다. 하나는 그룹화 속성이 없는 질의로서, 임의의 영역을 지정하고, 이 영역에 속한 모든 셀들을 대상으로 하나의 집계 값을 구한다. 다른 하나는, 그룹화 속성이 있는 질의로서, 임의의 영역을 지정하고, 이 영역에 속한 셀들에 대하여 주어진 그룹화 속성들의 값의 조합에 따라 집계 값을 구한다. 본 논문에서는 전자의 질의를 **영역 질의**(*range query*)[6]로, 후자의 질의를 **영역-그룹화 질의**(*range-groupby query*)로 정의한다. 그리고, 영역-그룹화 질의에서 각 그룹을 결정하는 그룹화 속성들의 값의 조합을 **그룹 키 값**(*group key value*)이라 정의한다.

그림 1은 영역 질의와 영역-그룹화 질의들을 SQL로 표현한 것이다. 디멘젼이 D_1, D_2, ..., D_n이고 메져가 m인 데이타 큐브 C에 대해서, 영역 질의와 영역-그룹화 질의는 각각 그림 1(a)와 그림 1(b)로 표현된다. 여기서, $[l_i, h_i](1 \le i \le n)$는 D_i에 대한 질의 범위를 나타내고, $D_{g_j}(1 \le j \le m \le n)$는 그룹화 속성을 나타낸다.

```
SELECT   SUM(C.M)                  SELECT   C.D_g1, C.D_g2, ..., C.D_gm, SUM(C.M)
FROM     C                         FROM     C
WHERE    l1 ≤ C.D1 ≤ h1            WHERE    l1 ≤ C.D1 ≤ h1
  and    l2 ≤ C.D2 ≤ h2              and    l2 ≤ C.D2 ≤ h2
  and    ...                         and    ...
  and    ln ≤ C.Dn ≤ hn              and    ln ≤ C.Dn ≤ hn
                                    GROUP BY C.D_g1, C.D_g2, ..., C.D_gm
```

(a) 영역 질의 (2) 영역-그룹화 질의

그림 1 데이타 큐브의 임의의 영역을 대상으로 하는 영역-집계 질의의 분류.

그러나, 영역-집계 질의는 특수한 경우를 제외하고는 확장 데이타 큐브를 사용해도 질의 처리 속도를 개선할 수 없다. 이것은 영역-집계 질의가 비 그룹화 속성에 대해 범위 조건을 갖기 때문이다. 비 그룹화 속성에 대해 범위 조건이 주어진 질의는 도메인의 일부분인 주어

진 범위에 속하는 셀들만을 집계한 값이 필요하다. 그런데, 확장 데이타 큐브는 그룹화 속성들이 가질 수 있는 각 값에 대하여, 비 그룹화 속성에 대해 전체 도메인에 걸친 모든 셀들을 집계한 값 만을 저장하고 있으므로, 이러한 질의를 처리하는데 아무 소용이 없다. 영역-집계 질의에서 그룹화 속성에만 범위 조건이 주어진 특수한 경우는 확장 데이타 큐브로 쉽게 계산할 수 있다. 그 이유는, 집계 값은 각 그룹 별로 계산되며, 그룹화 속성은 그룹을 나누기 위해서만 사용되는 속성이므로 그룹화 속성에 대한 조건은 집계 값에 영향을 미치지 않기 때문이다. 예제 1은 이러한 이슈를 예를 들어 설명한다.

예제 1: 그림 2는 확장 데이타 큐브를 사용한 영역-집계 질의의 처리 예를 보인다. 그림 2(a)는 디멘젼이 X와 Y이고, 메져가 m인 데이타 큐브 C를 나타내고, 그림 2(b)는 C에 대한 확장 데이타 큐브 C_{ext}를 나타낸다. 그림 2(b)에서 빗금으로 처리된 부분은 확장 데이타 큐브를 만들기 위해 데이타 큐브에 추가된 셀들을 나타낸다. 그림 2(c)는 확장 데이타 큐브로 처리 속도를 개선할 수 없는 영역-집계 질의의 예이다. 이 질의는 임의의 영역 $X:[x_1:x_2]\times Y:[y_2:y_4]$에 속하는 셀들을 X의 그룹 키 값 별로 집계한 값을 구한다. 그림 2(a)에서 음영으로 처리된 좌측 상자가 질의 영역에 속한 셀들을 나타낸다. 이 질의의 결과 중, $X=x_2$일 때의 집계 값은 $\sum_{v_Y=y_2}^{y_4} C[x_2, v_Y]$이 된다. 그런데, 확장 데이타 큐브에는 이 집계 값이 저장되어 있지 않다. 확장 데이타 큐브를 만들기 위해 추가된 셀 중 그룹 키 값이 x_2인 셀 $C_{ext}[x_2, y_{all}]$은 Y의 전체 도메인에 대하여 셀들을 집계한 값인 $\sum_{v_Y=y_0}^{y_5} C[x_2, v_Y]$를 가지고 있기 때문이다. 따라서, 이 질의는 확장 데이타 큐브를 사용하여 질의 처리 속도를 개선할 수 없다. 반면에, 그림 2(d)는 확장 데이타 큐브로 질의 처리 속도를 개선할 수 있는 질의의 예이다. 그림 2(a)에서 음영으로 처리된 우측 상자가 질의 영역에 속한 셀들을 나타낸다. 이 질의는 데이타 큐브는 액세스할 필요가 없이, 단지 확장 데이타 큐브의 셀 $C_{ext}[x_3, y_{all}]$와 $C_{ext}[x_4, y_{all}]$을 액세스하여 처리할 수 있다. 이것은 이 질의에서는 조건식이 단지 그룹화 속성에만 주어져 있기 때문이다.

영역 질의를 처리하는 방법에 대해서는 Ho 등[6]에 의해 연구되었다. Ho 등은 확장 데이타 큐브와는 다른 요약 정보를 추가로 선계산하여 유지함으로써 영역 질의를 효율적으로 처리하는 방법을 제안하였다. 특히, 영역 질의 중에서 가장 자주 사용되는, 집계 함수가 SUM인 영역 질의를 **영역-합**(*range-sum*) 질의로 정의하고,

	x_0	x_1	x_2	x_3	x_4	x_5
y_0	3	5	1	2	2	4
y_1	7	3	2	6	8	7
y_2	2	4	2	3	3	3
y_3	3	2	1	5	3	5
y_4	4	2	1	3	3	4
y_5	2	3	3	6	1	8

(a) 이차원 데이타 큐브 C

	x_0	x_1	x_2	x_3	x_4	x_5	x_{all}
y_0	3	5	1	2	2	4	17
y_1	7	3	2	6	8	7	33
y_2	2	4	2	3	3	3	17
y_3	3	2	1	5	3	5	19
y_4	4	2	1	3	3	4	17
y_5	2	3	3	6	1	8	23
y_{all}	21	19	10	25	20	31	126

(b) 확장 데이타 큐브 C_{ext}

```
SELECT    C.X, SUM(C.M)
FROM      C
WHERE     x1 ≤ C.X ≤ x2
  and     y2 ≤ C.Y ≤ y4
GROUP BY  C.X
```

(c) 확장 데이터 큐브로 처리 속도를 개선할 수 없는 질의

```
SELECT    C.X, SUM(C.M)
FROM      C
WHERE     x3 ≤ C.X ≤ x4
GROUP BY  C.X
```

(d) 확장 데이터 큐브로 처리 속도를 개선할 수 있는 질의

그림 2 확장 데이타 큐브를 사용한 영역-집계 질의 처리 예

이 질의의 처리를 위하여 요약 정보로서 전방-합(prefix-sum) 배열을 사용하는 방법을 제안하였다. 이 방법은 질의 영역의 크기에 상관 없이 항상 동일한 개수(2^n개, n은 디멘젼의 개수)의 셀을 액세스한다는 장점을 갖는다.

논문의 공헌 영역-그룹화 질의의 처리 방법에 대해서는 연구된 바 없다. 영역-그룹화 질의에서는, 각 그룹 키 값에 따라, 비 그룹화 속성들에 주어진 범위 조건을 만족하는 셀들을 집계한 값을 구해야 한다. 그런데, 영역 질의는 그룹화 속성을 고려하지 않고, 주어진 질의 영역에 속하는 모든 셀들을 집계한 값을 구하는 것이다. 따라서, 영역 질의의 처리에 사용된 방법을 영역-그룹화 질의의 처리에 직접 적용하기는 어렵다.

본 논문에서는 영역-그룹화 질의를 효율적으로 처리할 수 있는 방법에 대해 논의한다. 특히, 영역-그룹화 질의 중에서, 가장 빈번하게 사용되는 집계 함수인 SUM, COUNT, AVERAGE를 사용하는 경우의 계산 방법에 대해 논의한다. 주 아이디어는 Ho 등[6]이 제안한 전방-합 배열을 활용하여 영역-그룹화 질의를 효율적으로 계산하는 것이다. 우선, 전방-합 배열을 사용하

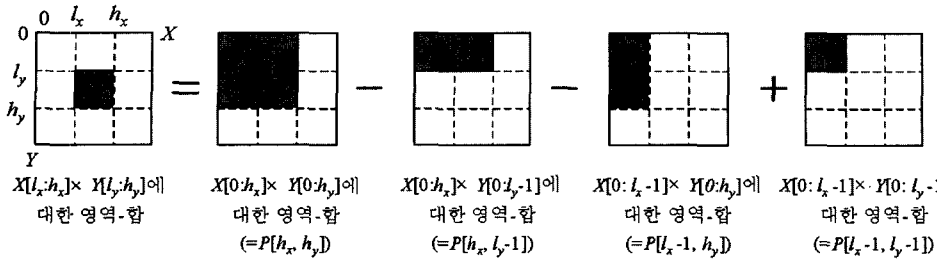


그림 3 전방-합 배열을 이용한 영역-합 질의 처리 방법의 예[6]

여 영역-그룹화 질의를 계산할 때, 그 과정에서 액세스되는 셀들이 몇 개의 클러스터(cluster)를 형성함을 이론적으로 증명한다. 그리고, 이 이론에 근거한 효율적인 계산 알고리즘을 제안한다. 또한, 알고리즘을 정형적으로 분석하여 제안한 알고리즘의 우수성을 입증한다. 제안한 알고리즘은 각 클러스터 단위로 셀들을 액세스함으로써 셀들을 효과적으로 액세스한다. 그 결과로 제안한 방법은 데이타 큐브를 직접 액세스하여 계산하는 방법보다 액세스되는 셀의 개수를 $O(\frac{2^{n-m}}{rs_{ng}})$로 줄인다. 여기서, n은 디멘젼의 개수, m은 그룹화 속성의 개수, rs_{ng}는 질의 영역을 비 그룹화 속성들에 대하여 프로젝션한 영역에 속한 셀의 개수이다. 또한, 가능한 모든 그룹화 속성들의 조합에 대하여, 각 그룹에서 비 그룹화 속성들을 대상으로 한 전방-합들을 선계산하여 유지하는 방법과 비교할 때, 액세스되는 셀의 개수는 비슷하면서 공간 오버헤드는 $O(\frac{1}{2^n})$로 줄인다.

논문의 구성 본 논문의 구성은 다음과 같다. 제2절에서는 관련 연구로서 전방-합 배열 및 이를 이용한 영역-합 질의 처리 방법을 소개한다. 제3절에서는 전방-합 배열을 이용한 영역-그룹화 질의 처리의 기본 개념을 소개한다. 제4절에서는 전방-합 배열을 이용한 영역-그룹화 질의 처리 시 액세스되는 셀들의 패턴을 분석하고, 분석 결과를 바탕으로 한 영역-그룹화 질의 처리 알고리즘을 제안한다. 제5절에서는 알고리즘의 성능에 대하여 논의한다. 마지막으로, 제6절에서는 결론을 내린다.

2. 관련 연구 : 전방-합 배열 및 영역-합 질의 처리 방법

본 절에서는 본 논문에서 사용하는 전방-합 배열 및 이를 이용한 영역-합 질의 처리 방법[6]을 설명한다. 전방-합 배열은 다음과 같이 정의된다. 디멘젼 D_1, D_2, ..., D_n으로 구성된 OLAP 데이타베이스가 있을 때, 이에 대한 데이타 큐브를 배열 A라 하고, A의 전방-합 배열을 P라 하자. 그리고, D_i의 도메인은 {0, 1, ..., $|D_i|-1$}이라 하자.[1] 이때, $P[p_1, p_2, ..., p_n]$는 두 점 $A[0, 0, ..., 0]$와 $A[p_1, p_2, ..., p_n]$로 정의되는 영역에 속한 모든 셀들의 값을 집계한 값을 갖는다. 이를 정형적으로 나타내면 식 (1)과 같다.

$$P[p_1, p_2, ..., p_n] = \sum_{v_1=0}^{p_1} \sum_{v_2=0}^{p_2} ... \sum_{v_n=0}^{p_n} A[v_1, v_2, ..., v_n] \quad (1)$$

그림 3은 X와 Y 축으로 구성된 이차원 데이타 큐브가 있을 때, 영역 $X[l_x:h_x]\times Y[l_y:h_y]$에 대한 영역-합 질의 처리 방법을 나타낸다. 그림에서와 같이 $X[l_x:h_x]\times Y[l_y:h_y]$에 대한 영역-합은 $X[0:h_x]\times Y[0:h_y]$에 대한 영역-합에서 $X[0:h_x]\times Y[0:l_y-1]$에 대한 영역-합과 $X[0:l_x-1]\times Y[0:h_y]$에 대한 영역-합을 뺀 후, $X[0:l_x-1](Y[0:l_y-1]$에 대한 영역-합을 더한 것과 같다. 이때, 데이타 큐브에 대한 전방-합 배열을 P라 하면, 이들 네 가지 영역-합은 각각 전방-합 배열의 $P[h_x, h_y]$, $P[h_x, l_y-1]$, $P[l_x-1, h_y]$, $P[l_x-1, l_y]$에 해당한다. 따라서, $X[l_x:h_x](Y[l_y:h_y]$에 대한 영역-합은 $P[h_x, h_y]$에서 $P[h_x, l_y-1]$과 $P[l_x-1, h_y]$를 뺀 후, $P[l_x-1, l_y-1]$을 더함으로써 쉽게 구할 수 있다.

이와 같이 전방-합 배열을 이용한 영역-합 계산 방법의 아이디어는, 임의의 a, b에 대하여 $a\oplus b\ominus b=a$가 되는 역 이항 연산자(inverse binary operator) '$\ominus$'가 존재하는 이항 연산자 '$\oplus$'에 모두 적용될 수 있다[6]. 집계 함수 COUNT도 이러한 특성을 만족시키므로 이 계산 방법을 사용할 수 있다. 또한, AVERAGE도 이 계산 방법을 사용하여 처리할 수 있는데, 그 이유는 AVERAGE는 SUM과 COUNT를 사용하여 구할 수 있기 때문이다.

1) OLAP 응용에서는 수입과 같이 도메인이 매우 큰 디멘젼의 경우, 여러 개의 연속적인 범위의 값으로 나누고, 각 범위를 하나의 값으로 대응시키는 변환 기법을 사용하여, 랭크 도메인(rank domain)[6]이라 불리는 변환된 도메인을 사용하는 것이 일반적이다.

보조정리 1은 위에서 설명한 전방-합 배열을 이용한 영역-합 처리 방법을 정형화한다.

보조정리 1 [6] 디멘젼 D_1, D_2, ..., D_n으로 구성된 데이타 큐브를 나타내는 배열 A와 이에 대한 전방-합 배열 P가 있다고 하자. 이 때, 각 디멘젼 D_i에 범위 조건 $[l_i, h_i]$가 주어진 영역-합 질의는 다음 식 (2)로 구할 수 있다. 여기서, a는 $P[v_1, v_2, ..., v_n]$에서 v_i 값이 l_i-1인 v_i의 개수이다. $l_i=0$인 경우 v_i는 -1이 될 수 있다. 표현을 간단히 하기 위하여, $v_i(1\le i\le n)$ 값이 하나라도 -1(이것은 존재하지 않는 값이다)인 경우, $P[v_1, v_2, ..., v_n]=0$으로 설정한다.

$$\sum_{v_1=l_1}^{h_1}\sum_{v_2=l_2}^{h_2}\cdots\sum_{v_n=l_n}^{h_n}A[v_1,v_2,...,v_n] = \sum_{v_1\in\{l_1-1,h_1\}}\sum_{v_2\in\{l_2-1,h_2\}}\cdots \sum_{v_n\in\{l_n-1,h_n\}}(-1)^a P[v_1,v_2,...,v_n] \quad (2)$$

□

식 (2)에서 좌변 항은 처리하고자 하는 영역-합 질의를 나타낸다. 그리고, 우변 항은 전방-합 배열 P를 이용하여 이 질의를 처리하기 위한 계산식을 나타낸다. 여기서, a 값은 포함-배제의 원리(inclusion exclusion principle)[13]로부터 나온 결과이다. 식 (2)는 제4절에서 액세스되는 셀들을 정형적으로 분석하기 위하여 사용된다.

따름정리 1 [6] 보조정리 1의 식 (2)를 사용하여 영역-합 질의를 처리할 때, 배열 P에서 액세스되는 셀의 개수는 2^n이다. □

증명: 식 (2)에서 각 $v_i(1\le i\le n)$는 두 개의 값을 가질 수 있으므로, 배열 P에서 액세스되는 셀의 개수는 2^n이다. □

3. 영역-그룹화 질의 계산의 개념

본 절에서는 영역-그룹화 질의를 계산하는 방법의 기본 개념에 대해 논의한다. 디멘젼 D_1, D_2, ..., D_n으로 구성된 데이타 큐브 A에서, 각 D_i에 범위 조건 $[l_i, h_i]$가 주어지고, $D_{g_1}, D_{g_2}, ..., D_{g_m}$을 그룹화 속성으로 영역-그룹화 질의가 있다고 하자. 비 그룹화 속성은 $D_{ng_2}, D_{ng_2}, ..., D_{ng_{n-m}}$으로 표시하기로 한다. 영역-그룹화 질의는 각 그룹 키 값 $(D_{g_1}, D_{g_2}, ..., D_{g_m})=(a_{g_1}, a_{g_2}, ..., a_{g_m})$ $(l_{g_i}\le a_{g_i}\le h_{g_i})$에 대하여, 다음 식(3)에 주어진 집계 값을 구하는 것이다.

$$\sum_{v_{ng_1}=l_{ng_1}}^{h_{ng_1}}\sum_{v_{ng_2}=l_{ng_2}}^{h_{ng_2}}\cdots\sum_{v_{ng_{n-m}}=l_{ng_{n-m}}}^{h_{ng_{n-m}}}A[a_{g_1},...,a_{g_m},v_{ng_1},...,v_{ng_{n-m}}] \quad (3)$$

식 (3)의 집계 값은, 각 그룹 키 값에 대하여, 비 그룹화 속성들에 대한 전방-합 배열을 갖는 자료 구조를 유지함으로써 빠르게 구할 수 있다. 이러한 자료 구조를 정의 1에서 정의하고, 보조정리 2에서는 이를 이용한 영역-그룹화 질의 계산 방법을 기술한다.

정의 1 디멘젼 D_1, D_2, ..., D_n으로 구성된 데이타 큐브를 배열 A라 하자. 식 (4)를 만족하는 배열 $Q_{g_1g_2...g_m}$을 그룹화 속성이 $D_{g_1}, D_{g_2}, ..., D_{g_m}$일 때의 배열 A에 대한 **그룹 전방-합 배열**(*group prefix-sum array*)이라 정의한다.

$$Q_{g_1g_2...g_m}[q_{g_1},...,q_{g_m},q_{ng_1},...,q_{ng_{n-m}}] = \sum_{v_{ng_1}=0}^{q_{ng_1}}\sum_{v_{ng_2}=0}^{q_{ng_2}}\cdots\sum_{v_{ng_{n-m}}=0}^{q_{ng_{n-m}}}A[q_{g_1},...,q_{g_m},v_{ng_1},...,v_{ng_{n-m}}] \quad (4)$$

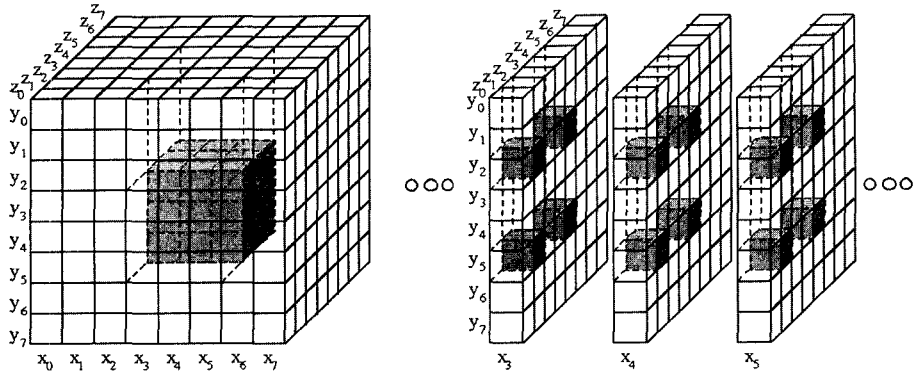
보조정리 2 디멘젼 D_1, D_2, ..., D_n으로 구성된 데이타 큐브 C가 있다고 하자. 그리고, 각 디멘젼 D_i에 범위 조건 $[l_i, h_i]$가 주어지고, $D_{g_1}, D_{g_2}, ..., D_{g_m}$를 그룹화 속성으로 하는 영역-그룹화 질의를 배열 $Q_{g_1g_2...g_m}$을 이용하여 처리한다고 하자. 그러면, 그룹 키 값이 $(a_{g_1}, a_{g_2}, ..., a_{g_m})$일 때의 집계 값을 다음 식 (5)로 구할 수 있다. 여기서, a는 $Q_{g_1g_2...g_m}[a_{g_1},...,a_{g_m},v_{ng_1},...,v_{ng_{n-m}}]$에서 $v_{ng_j}(1\le j\le n-m)$ 값이 $l_{ng_j}-1$인 v_{ng_j}의 개수이다.

$$\sum_{v_{ng_1}=l_{ng_1}}^{h_{ng_1}}\cdots\sum_{v_{ng_{n-m}}=l_{ng_{n-m}}}^{h_{ng_{n-m}}}A[a_{g_1},...,a_{g_m},v_{ng_1},...,v_{ng_{n-m}}] = \sum_{v_{ng_1}\in\{l_{ng_1}-1,h_{ng_1}\}}\cdots\sum_{v_{ng_{n-m}}\in\{l_{ng_{n-m}}-1,h_{ng_{n-m}}\}} (-1)^a Q_{g_1g_2...g_m}[a_{g_1},...,a_{g_m},v_{ng_1},...,v_{ng_{n-m}}] \quad (5)$$

증명: 데이타 큐브 C에서 그룹 키 값 $(a_{g_1}, a_{g_2}, ..., a_{g_m})$을 갖는 셀들은 디멘젼이 $D_{ng_1}, D_{ng_2}, ..., D_{ng_{n-m}}$인 $n-m$ 차원 배열을 구성한다. 이 배열을 B라 하자. 그러면, $Q_{g_1g_2...g_m}$의 정의에 의하여, 배열 $Q_{g_1g_2...g_m}$에서 그룹 키 값 $(a_{g_1}, a_{g_2}, ..., a_{g_m})$을 갖는 셀들은 $n-m$ 차원 배열 B의 전방-합 배열을 형성한다. 한편, 그룹 키 값이 $(a_{g_1}, a_{g_2}, ..., a_{g_m})$일 때의 집계 값은 배열 B를 대상으로 하고, 디멘젼 $D_{ng_j}(1\le j\le n-m)$에 대한 범위가 $[l_{ng_j}, h_{ng_j}]$인 영역-합 질의의 결과와 동일하다. 따라서, 보조정리 1의 식 (2)에 의하여, 그룹 키 값이 $(a_{g_1}, a_{g_2}, ..., a_{g_m})$일 때의 집계 값은 식 (5)로 계산된다. □

앞으로는 혼동이 없는 한 $Q_{g_1g_2...g_m}$을 간단히 Q로 표기한다.

예제 2: 그림 4는 속성 X, Y, Z로 구성된 OLAP 데이타베이스에서, X, Y, Z에 범위가 주어지고, X를 그룹



(a) 배열 A : 데이터 큐브 (b) 배열 Q : 그룹화 속성이 X일 때의 그룹 전방-합 배열

그림 4 영역-그룹화 질의의 처리 예(그룹화 속성=$\{X\}$)

화 속성으로 하는 영역-그룹화 질의를 처리하는 예를 보인다. 그림 4(a)는 데이타 큐브를 나타내는 배열 A이고, 그림 4(b)는 그룹화 속성이 X일 때의 배열 A에 대한 그룹 전방-합 배열 Q이다. 그림 4(b)에서와 같이, 배열 Q는 그룹화 속성 X의 각 값 x_i에 대하여, 속성 Y와 Z에 대한 전방-합 배열을 갖는다. X, Y, Z에 대한 범위 조건은 각각 $[x_3,x_5]$, $[y_3,y_5]$, $[z_2,z_4]$이다. 이때, 영역-그룹화 질의를 처리하기 위해 액세스되는 셀들을 그룹화 속성 X의 값에 따라 나타내면 다음과 같다.

- $(X=x_3)$: $\sum_{v_Y=y_3}^{y_5}\sum_{v_Y=y_2}^{y_4} A[x_3, v_Y, v_Z]$
  $= Q[x_3, y_5, z_4] - Q[x_3, y_5, z_1]$
  $- Q[x_3, y_2, z_4] + Q[x_3, y_2, z_1]$
- $(X=x_4)$: $\sum_{v_Y=y_3}^{y_5}\sum_{v_Y=y_2}^{y_4} A[x_4, v_Y, v_Z]$
  $= Q[x_4, y_5, z_4] - Q[x_4, y_5, z_1]$
  $- Q[x_4, y_2, z_4] + Q[x_4, y_2, z_1]$
- $(X=x_5)$: $\sum_{v_Y=y_3}^{y_5}\sum_{v_Y=y_2}^{y_4} A[x_5, v_Y, v_Z]$
  $= Q[x_5, y_5, z_4] - Q[x_5, y_5, z_1]$
  $- Q[x_5, y_2, z_4] + Q[x_5, y_2, z_1]$

즉, 각 그룹 키 값에 대해 영역-합 질의를 수행함으로써 영역-그룹화 질의를 계산한다. 그림 4(b)에서 음영으로 처리된 셀들이 액세스되는 셀들을 나타낸다. □

따름정리 2 보조정리 2의 식 (5)를 사용하여 영역-그룹화 질의를 처리할 때, 배열 Q에서 액세스되는 셀의 개수는 $2^{n-m}\times\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$이다.

증명: 그룹 키 값의 개수는 $\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$이다. 그리고, 식 (5)에 의하여, 하나의 그룹 키 값에 대한 집계 값을 구하는데 2^{n-m} 개의 셀을 액세스해야 한다. 따라서, 액세스되는 셀의 개수는 $2^{n-m}\times\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$이다. □

데이타 큐브로부터 $\prod_{g_i(1\le i\le n)}(h_{g_i}-l_{g_i}+1)$개의 셀을 액세스해야 하는 영역-그룹화 질의를 그룹 전방-합 배열을 사용하면 단지 $2^{n-m}\times\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$개의 셀만을 액세스함으로써 계산할 수 있다. 이것은 $\prod_{ng_j(1\le j\le m)}(h_{ng_j}-l_{ng_j}+1)$이 2^m으로 줄어든 것이다. 그러나, 이 방법은 2^n개나 되는 가능한 그룹화 속성의 조합마다 선계산된 그룹 전방-합 배열을 저장하고 관리해야 하기 때문에 실제로는 사용하기 어려운 방법이다.

4. 영역-그룹화 질의의 계산 알고리즘

본 절에서는 OLAP 데이타베이스를 구성하는 디멘젼들에 대한 전방-합 배열 하나 만을 가지고 영역-그룹화 질의를 효율적으로 처리하는 방법을 제안한다. 이 방법은 제3절에서의 방법과는 달리 가능한 그룹화 속성의 조합마다 그룹 전방-합 배열을 유지할 필요가 없다. 먼저, 제4.1절에서 영역-그룹화 질의와 전방-합 배열의 특징들에 대하여 기술한다. 이러한 특징들은 단지 하나의 전방-합 배열만을 가지고 영역-그룹화 질의를 효율적으로 처리하는 것을 가능하게 한다. 다음으로, 제4.2절에서는 영역-그룹화 질의를 처리할 때 액세스되는 셀들의 패턴을 분석한다. 제4.3절에서는 이러한 패턴들을 이용한 영역-그룹화 질의의 계산 알고리즘을 제시한다.

4.1 영역-그룹화 질의와 전방-합 배열의 특징

영역-그룹화 질의의 특징은 다음과 같다. 첫째, 영역-그룹화 질의는 $\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$개의 부영역(subregion)에 대한 영역-합 질의로 나눌 수 있다. 여기서, 각 부영역은 하나의 그룹에 속한 셀들로 구성된다.

	x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7
y_0	3	5	1	2	2	4	6	3
y_1	7	3	2	6	8	7	1	2
y_2	2	4	2	3	3	3	4	5
y_3	3	2	1				2	8
y_4	4	2	1				7	1
y_5	2	3	3				5	1
y_6	4	5	2	7	1	9	3	3
y_7	2	4	2	2	3	1	9	1

(a) 데이타 큐브(배열 A)에서 집계되는 영역

	x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7
y_0	3	8	9	11	13	17	23	26
y_1	10	18	21	29	39	50	57	62
y_2	12	24					78	88
y_3	15	29	35	51	67	86	99	117
y_4	19	35	42	61	80	103	123	142
y_5	21	40					151	171
y_6	25	49	61	93	114	154	182	205
y_7	27	55	69	103	127	168	205	229

(b) 전방-합 배열(배열 P)에서 액세스되는 셀들

그림 5 이차원 전방-합 배열을 이용한 영역-그룹화 질의에 있어서 집계되는 영역과 전방-합 배열에서 액세스되는 셀들(그룹화 속성=$\{X\}$)

즉, 그룹 키 값이 $(a_{g_1}, a_{g_2}, \ldots, a_{g_m})$인 셀들은 그룹화 속성 D_{g_i}에 대해서는 크기가 1인 범위 조건 $[a_{g_i}, a_{g_i}]$을 갖고, 비 그룹화 속성 D_{ng_j}에 대해서는 범위 조건 $[l_{ng_j}, h_{ng_j}]$을 갖는 부영역을 형성한다. 둘째, 위에서 나누어진 부영역들은 서로 인접해 있다.

전방-합 배열의 특징은 다음과 같다. 첫째, 전방-합 배열을 사용한 영역-합 질의는 영역의 크기와 모양에 상관 없이 따름정리 1에서 보인 바와 같이 동일한 개수 $(=2^n$, n은 디멘젼의 개수)의 셀을 액세스한다. 둘째, 서로 인접한 두 개의 영역에 대한 영역-합 질의를 처리하기 위해 액세스되는 셀들은 서로 중복된다. 이에 대한 자세한 내용은 제4.2절에서 논의한다.

본 논문에서는, 이러한 특징들을 활용하여, 전방-합 배열을 사용하여 영역-그룹화 질의를 효율적으로 처리하는 방법을 제안한다.

4.2 액세스 패턴의 분석

영역-그룹화 질의를 계산하기 위하여 액세스하는 전방-합 배열 내의 셀들은 몇 개의 클러스터를 형성하게 된다. 이것은 그룹 키 값이 $(a_{g_1}, a_{g_2}, \ldots, a_{g_m})$인 그룹에 대한 집계 값을 계산하기 위해 액세스해야 되는 셀들을 분석함으로써 알 수 있다. 이 셀들은 $(a_{g_1}, \ldots, a_{g_m}, l_{ng_1}, \ldots, l_{ng_{n-m}})$와 $(a_{g_1}, \ldots, a_{g_m}, h_{ng_1}, \ldots, h_{ng_{n-m}})$로 정의되는 초사각형(hyper rectangle)에 대한 영역-합 질의를 처리할 때 액세스되는 셀들이다. 먼저, 이해를 돕기 위하여 제4.2.1절에서 그룹화 속성의 개수가 하나인 경우에 대하여 논의한다. 다음으로, 제4.2.2절에서 이를 일반화하여 그룹화 속성의 개수가 여러 개인 경우에 대하여 논의한다.

4.2.1 그룹화 속성의 개수가 하나인 경우

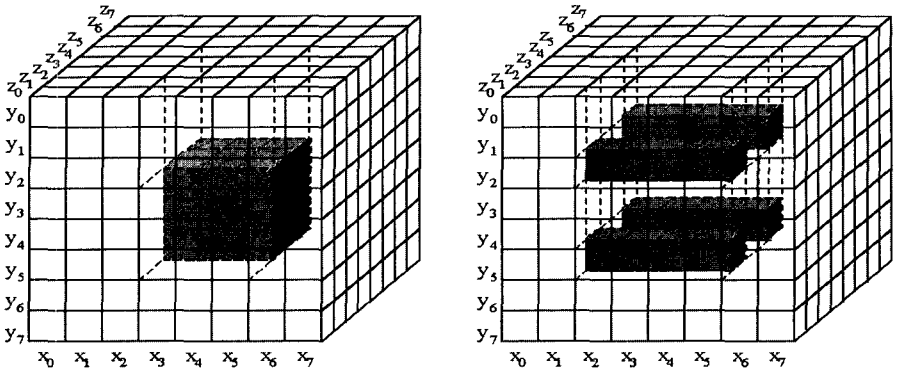
예제 3과 4는 하나의 전방-합 배열을 이용하여 영역-그룹화 질의들을 어떻게 처리할 수 있는지 설명한다. 예제 2는 디멘젼 개수가 두 개인 경우이고, 예제 3은 디멘젼 개수가 세 개인 경우이다.

예제 3: 그림 5는 속성 X와 Y에 대한 전방-합 배열을 이용하여, X와 Y에 범위가 주어지고, X를 그룹화 속성으로 하는 영역-그룹화 질의를 처리하는 예를 보인다. 그림 5(a)는 데이타 큐브(배열 A)에서 집계되는 영역을 나타내고, 그림 5(b)는 A의 전방-합 배열(배열 P)에서 액세스되는 셀들을 나타낸다. X에 대한 범위 조건은 $[x_3, x_5]$이고, Y에 대한 범위 조건은 $[y_3, y_5]$이다. 영역-그룹화 질의를 처리하기 위해 액세스되는 셀들을 그룹화 속성 X의 값에 따라 나타내면 다음과 같다.

- $(X=x_3)$: $\sum_{v_Y=y_2}^{y_5} A[x_3, v_Y]$
  $= P[x_3, y_5] - P[x_3, y_2] - P[x_2, y_5] - P[x_2, y_2]$
- $(X=x_4)$: $\sum_{v_Y=y_2}^{y_5} A[x_4, v_Y]$
  $= P[x_4, y_5] - P[x_4, y_2] - P[x_3, y_5] - P[x_3, y_2]$
- $(X=x_5)$: $\sum_{v_Y=y_2}^{y_5} A[x_5, v_Y]$
  $= P[x_5, y_5] - P[x_5, y_2] - P[x_4, y_5] - P[x_4, y_2]$

이를 분석하면, $X=x_3$와 $X=x_4$일 때 $P[x_3, y_5]$, $P[x_3, y_2]$가 중복 액세스되며, $X=x_4$와 $X=x_5$일 때, $P[x_4, y_5]$, $P[x_4, y_2]$가 중복 액세스된다. 따라서, X의 각 값에 따라 네 개의 셀이 액세스되어 전체적으로는 12개의 셀이 액세스되게 되지만, 중복된 셀을 제외하면 실제로는 여덟 개의 셀만이 액세스된다. 이 셀들은 X 축에 평행한 두 개의 띠(strip)를 형성함을 알 수 있다.

예제 4: 그림 6은 속성 X, Y, Z에 대한 전방-합 배열을 이용하여, X, Y, Z에 범위가 주어지고, X를 그룹화 속성으로 하는 영역-그룹화 질의를 처리하는 예를



(a) 데이타 큐브(배열 A)에서 집계되는 영역 (b) 전방-합 배열(배열 P)에서 액세스되는 셀들

그림 6 삼차원 전방-합 배열을 이용한 영역-그룹화 질의에 있어서 집계되는 영역과 전방-합 배열에서 액세스되는 셀들(그룹화 속성=$\{X\}$)

보인다. 그림 6(a)는 데이타 큐브(배열 A)에서 집계되는 영역을 나타내고, 그림 6(b)는 A의 전방-합 배열(배열 P)에서 액세스되는 셀들을 나타낸다. X, Y, Z에 대한 범위 조건은 각각 $[x_3, x_5]$, $[y_3, y_5]$, $[z_2, z_4]$이다. 영역-그룹화 질의를 처리하기 위해 액세스되는 셀들을 그룹화 속성 X의 값에 따라 나타내면 다음과 같다.

- $(X=x_3)$: $\sum_{v_Y=y_3}^{y_5}\sum_{v_Z=z_2}^{z_4} A[x_3, v_Y, v_Z]$
  $= P[x_3, y_5, z_4] - P[x_3, y_5, z_1] - P[x_3, y_2, z_4]$
  $- P[x_3, y_2, z_1] - P[x_2, y_5, z_4] + P[x_2, y_5, z_1]$
  $+ P[x_2, y_2, z_4] - P[x_2, y_2, z_1]$
- $(X=x_4)$: $\sum_{v_Y=y_3}^{y_5}\sum_{v_Z=z_2}^{z_4} A[x_4, v_Y, v_Z]$
  $= P[x_4, y_5, z_4] - P[x_4, y_5, z_1] - P[x_4, y_2, z_4]$
  $- P[x_4, y_2, z_1] - P[x_3, y_5, z_4] + P[x_3, y_5, z_1]$
  $+ P[x_3, y_2, z_4] - P[x_3, y_2, z_1]$
- $(X=x_5)$: $\sum_{v_Y=y_3}^{y_5}\sum_{v_Z=z_2}^{z_4} A[x_5, v_Y, v_Z]$
  $= P[x_5, y_5, z_4] - P[x_5, y_5, z_1] - P[x_5, y_2, z_4]$
  $- P[x_5, y_2, z_1] - P[x_4, y_5, z_4] + P[x_4, y_5, z_1]$
  $+ P[x_4, y_2, z_4] - P[x_4, y_2, z_1]$

이를 분석하면, $X=x_3$와 $X=x_4$일 때 $P[x_3, y_5, z_4]$, $P[x_3, y_5, z_1]$, $P[x_3, y_2, z_1]$, $P[x_3, y_2, z_1]$이 중복 액세스되며, $X=x_4$와 $X=x_5$일 때 $P[x_4, y_5, z_4]$, $P[x_4, y_5, z_1]$, $P[x_4, y_2, z_4]$, $P[x_4, y_2, z_1]$이 중복 액세스된다. 따라서, X의 각 값에 따라 여덟 개의 셀이 액세스되어 전체적으로 24개의 셀이 액세스되게 되지만, 중복된 셀을 제외하면 실제로는 16개의 셀만이 액세스된다. 이 셀들은 X 축에 평행한 네 개의 띠를 형성함을 알 수 있다. □

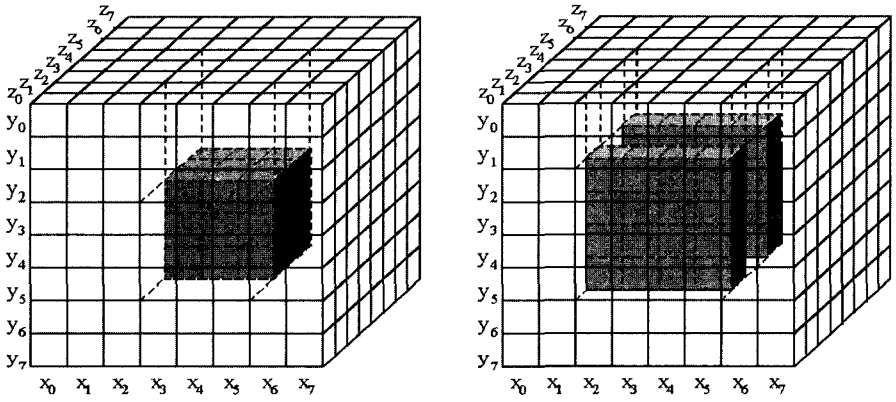
예제 3과 4를 보면 액세스되는 셀들이 띠 모양을 갖는 여러 개의 클러스터를 형성함을 알 수 있다. 보조정리 3은 그룹화 속성의 개수가 하나인 경우에 대하여 액세스되는 셀의 패턴을 일반화한다.

보조정리 3 디멘젼 $D_1, D_2, \ldots, D_n$으로 구성된 데이타 큐브에 대한 전방-합 배열 P가 있다고 하자. 그리고, 각 디멘젼 D_i에 범위 조건 $[l_i, h_i]$가 주어지고, D_g를 그룹화 속성으로 하는 영역-그룹화 질의를 전방-합 배열 P를 이용하여 처리한다고 하자. 그러면, P로부터 액세스되는 셀들의 집합은 $\bigcup_{v_g=l_g-1}^{h_g}\bigcup_{\forall v_i(i\neq g)\in\{l_i,h_i\}}\{P[v_1, v_2, \ldots, v_g, \ldots, v_n]\}$이다.

증명: 영역-그룹화 질의를 처리하기 위해 액세스해야 되는 셀을 그룹화 속성 D_g의 값에 따라 나타내면 다음과 같다. 이 값들은 보조정리 1을 사용하여 구할 수 있다.

- $(v_g=l_g)$:
  $U_{l_g} = \bigcup_{\forall v_i(i\neq g)\in\{l_i,h_i\}}\{P[v_1, v_2, \ldots, l_g-1, \ldots, v_n]\}$
  $\cup\bigcup_{\forall v_i(i\neq g)\in\{l_i,h_i\}}\{P[v_1, v_2, \ldots, l_g, \ldots, v_n]\}$
- $(v_g=l_g+1)$:
  $U_{l_g+1} = \bigcup_{\forall v_i(i\neq g)\in\{l_i,h_i\}}\{P[v_1, v_2, \ldots, l_g, \ldots, v_n]\}$
  $\cup\bigcup_{\forall v_i(i\neq g)\in\{l_i,h_i\}}\{P[v_1, v_2, \ldots, l_g+1, \ldots, v_n]\}$
  …
- $(v_g=h_g)$:
  $U_{h_g} = \bigcup_{\forall v_i(i\neq g)\in\{l_i,h_i\}}\{P[v_1, v_2, \ldots, h_g-1, \ldots, v_n]\}$
  $\cup\bigcup_{\forall v_i(i\neq g)\in\{l_i,h_i\}}\{P[v_1, v_2, \ldots, h_g, \ldots, v_n]\}$

각 그룹 키 값에 대해 액세스되는 셀들을 분석하면, U_{v_g}와 $U_{v_{g+1}}$은 $\bigcup_{\forall v_i(i\neq g)\in\{l_i,h_i\}}\{P[v_1, v_2, \ldots, v_g, \ldots, v_n]\}$가 중복된다. 따라서, 중복되는 셀들을 제거하면 액세스되는 셀들의 전체 집합 $\bigcup_{v_g=l_g}^{h_g} U_{v_g}$는 $\bigcup_{v_g=l_g-1}^{h_g}\bigcup_{\forall v_i(i\neq g)\in\{l_i,h_i\}}\{P[v_1, v_2, \ldots, v_g, \ldots, v_n]\}$이 되어 보조정리가 성립한다. □



(a) 데이타 큐브(배열 A)에서 집계되는 영역 (b) 전방-합 배열(배열 P)에서 액세스되는 셀들

그림 7 삼차원 전방-합 배열을 이용한 영역-그룹화 질의에 있어서 집계되는 영역과 전방-합 배열에서 액세스되는 셀들(그룹화 속성=$\{X,Y\}$)

따름정리 3 보조정리 3에서 영역-그룹화 질의를 처리할 때, P에서 액세스되는 셀의 개수는 $2^{n-1}\times(h_g-l_g+2)$이다.

증명: 각 $v_i(i\neq g)$는 두 개의 값을 가질 수 있고, 각 v_g는 (h_g-l_g+2)개의 값을 가질 수 있으므로 액세스되는 셀의 개수는 $2^{n-1}\times(h_g-l_g+2)$이다. □

보조정리 3에 따르면 액세스되는 셀들은 그룹화 속성 D_g축에 평행한 2^{n-1}개의 띠를 형성한다. 즉, 2^{n-1}개의 클러스터를 형성한다. 그리고, 각 클러스터는 그룹화 속성 D_g에 대해서는 $[l_{g-1}, h_g]$를 범위로 하고, 그룹화 속성이 아닌 속성 $D_i(i\neq g)$에 대해서는 l_i-1 또는 h_i 값을 갖는다.

4.2.2 그룹화 속성의 개수가 여러 개인 경우

본 절에서는 그룹화 속성의 개수가 여러 개인 경우에 영역-그룹화 질의를 처리하기 위해 액세스되는 셀의 패턴을 논의한다. 먼저, 예제 5를 통해 액세스되는 셀의 패턴을 직관적으로 이해할 수 있도록 한다. 그리고, 정리 1에서 액세스되는 셀의 패턴을 일반화하고 이를 증명한다.

예제 5: 그림 7은 속성 X, Y, Z에 대한 전방-합 배열을 이용하여, X, Y, Z에 범위가 주어지고, X, Y를 그룹화 속성으로 하는 영역-그룹화 질의를 처리하는 예를 보인다. 그림 7(a)는 데이타 큐브(배열 A)에서 집계되는 영역을 나타내고, 그림 7(b)는 A의 전방-합 배열(배열 P)에서 액세스되는 셀들을 나타낸다. X, Y, Z에 대한 범위 조건은 각각 $[x_3, x_5]$, $[y_3, y_5]$, $[z_2, z_4]$이다. 영역-그룹화 질의를 처리하기 위해 액세스되는 셀들을 그룹화 속성 (X,Y)의 값에 따라 나타내면 다음과 같다.

- $(X=x_3, Y=y_3)$: $\sum_{v_z=z_2}^{z_4} A[x_3, y_3, v_z]$
  $= P[x_3, y_3, z_4] - P[x_3, y_3, z_1] - P[x_3, y_2, z_4]$
  $+ P[x_3, y_2, z_1] - P[x_2, y_3, z_4] + P[x_2, y_3, z_1]$
  $+ P[x_2, y_2, z_4] - P[x_2, y_2, z_1]$
- $(X=x_3, Y=y_4)$: $\sum_{v_z=z_2}^{z_4} A[x_3, y_4, v_z]$
  $= P[x_3, y_4, z_4] - P[x_3, y_4, z_1] - P[x_3, y_3, z_4]$
  $+ P[x_3, y_3, z_1] - P[x_2, y_4, z_4] + P[x_2, y_4, z_1]$
  $+ P[x_2, y_3, z_4] - P[x_2, y_3, z_1]$
- $(X=x_4, Y=y_3)$: $\sum_{v_z=z_2}^{z_4} A[x_4, y_3, v_z]$
  $= P[x_4, y_3, z_4] - P[x_4, y_3, z_1] - P[x_4, y_2, z_4]$
  $+ P[x_4, y_2, z_1] - P[x_3, y_3, z_4] + P[x_3, y_3, z_1]$
  $+ P[x_3, y_2, z_4] - P[x_3, y_2, z_1]$
- $(X=x_4, Y=y_4)$: $\sum_{v_z=z_2}^{z_4} A[x_4, y_4, v_z]$
  $= P[x_4, y_4, z_4] - P[x_4, y_4, z_1] - P[x_4, y_3, z_4]$
  $+ P[x_4, y_3, z_1] - P[x_3, y_4, z_4] + P[x_3, y_4, z_1]$
  $+ P[x_3, y_3, z_4] - P[x_3, y_3, z_1]$

  ...
- $(X=x_5, Y=y_5)$: $\sum_{v_z=z_2}^{z_4} A[x_5, y_5, v_z]$
  $= P[x_5, y_5, z_4] - P[x_5, y_5, z_1] - P[x_5, y_4, z_4]$
  $+ P[x_5, y_4, z_1] - P[x_4, y_5, z_4] + P[x_4, y_5, z_1]$
  $+ P[x_4, y_4, z_4] - P[x_4, y_4, z_1]$

이를 분석하면, $X=x_3, Y=y_3$과 $X=x_3, Y=y_4$일 때, 즉, Y의 값만이 1 만큼 증가할 때, $P[x_3, y_3, z_4]$, $P[x_3, y_3, z_1]$, $P[x_2, y_3, z_4]$, $P[x_2, y_3, z_1]$이 중복 액세스된다. 그리고, $X=x_3, Y=y_3$과 $X=x_4, Y=y$일 때, 즉, X의 값만이 1만큼 증가할 때, $P[x_3, y_3, z_4]$, $P[x_3, y_3, z_1]$, $P[x_3, y_2, z_4]$, $P[x_3, y_2, z_1]$이 중복 액세스된다. 또한,

$X=x_3, Y=y_3$과 $X=x_4, Y=y_4$일 때, 즉, X와 Y의 값들이 모두 1 만큼 증가할 때, $P[x_3, y_3, z_4], P[x_3, y_3, z_1]$이 중복 액세스된다. 이와 같이, X 또는 Y 값만이 1 만큼 증가할 때는 네 개의 셀이 중복되고, X와 Y 값이 모두 1 만큼 증가할 때는 두 개의 셀이 중복된다. 따라서, (X,Y)의 각 값에 따라 여덟 개의 셀이 액세스되어 전체적으로는 72(=9×8)개의 셀이 액세스되게 되지만, 중복된 셀을 제외하면 실제로는 32 개의 셀만이 액세스된다. 이 셀들은 X,Y 평면에 평행한 두 개의 이차원 사각형을 형성함을 알 수 있다. □

예제 5를 보면, 그룹화 속성이 여러 개인 경우에도 액세스되는 셀들이 여러 개의 클러스터를 형성함을 알 수 있다. 단, 각 클러스터의 모양이 그룹화 속성이 하나인 경우와는 다르게 이차원 사각형 모양(일반적으로 m차원 초사각형, m은 그룹화 속성들의 개수)을 갖는다. 다음 정리 1은 그룹화 속성들의 개수가 여러 개인 경우에 대하여 액세스되는 셀의 패턴을 일반화한다.

정리 1 디멘젼 $D_1, D_2, \ldots, D_n$으로 구성된 데이타 큐브에 대한 전방-합 배열 P가 있다고 하자. 그리고, 각 디멘젼 D_i에 범위 조건 $[l_i, h_i]$가 주어지고, $D_{g_1}, D_{g_2}, \ldots, D_{g_m}$를 그룹화 속성으로 하는 영역-그룹화 질의를 전방-합 배열 P를 이용하여 처리한다고 하자. 그러면, P로부터 액세스되는 셀들의 집합은 다음 식 (6)과 같다.

$$\bigcup_{v_{g_1}=l_{g_1}-1}^{h_{g_1}} \cdots \bigcup_{v_{g_m}=l_{g_m}-1}^{h_{g_m}} \bigcup_{v_{ng_1}\in\{l_{ng_1}, h_{ng_1}\}} \cdots \bigcup_{v_{ng_{n-m}}\in\{l_{ng_{n-m}}, h_{ng_{n-m}}\}} \{P[v_{g_1}, \ldots, v_{g_m}, v_{ng_1}, \ldots, v_{ng_{n-m}}]\} \quad (6)$$

증명: 배열의 색인 순서를 바꾸어도 증명의 정확성에는 영향을 미치지 않으므로, 증명의 편의상 $g_i=i$라고 가정하자. 그러면, 식 (6)을 U'이라 할 때, U'은 다음과 같이 표현된다.

$$U' = \bigcup_{v_1\in[l_1-1, h_1]} \cdots \bigcup_{v_m=[l_m-1, h_m]} \cdots \bigcup_{v_{m+1}\in\{l_{m+1}-1, h_{m+1}\}} \{P[v_1, v_2, \ldots, v_n]\}$$

그룹 키 값이 $(w_1, w_2, \ldots, w_m)$인 그룹의 집계 값을 구하기 위해 액세스되는 셀들의 집합을 $U_{(w_1, w_2, \ldots, w_m)}$이라 하자. 그리고, $\bigcup_{w_1=l_1}^{h_1} \bigcup_{w_2=l_2}^{h_2} \cdots \bigcup_{w_m=l_m}^{h_m} U_{(w_1, w_2, \ldots, w_m)}$이라 하자. 이제, 임을 보인다.

(i) $U\subseteq U'$: 그룹 키 값이 $(w_1, w_2, \ldots, w_m)$인 그룹의 집계 값은 다음과 같다.

$$\sum_{v_1=w_1}^{w_1} \cdots \sum_{v_m=w_m}^{w_m} \sum_{v_{m+1}=l_{m+1}}^{h_{m+1}} \cdots \sum_{v_n=l_n}^{h_n} A[v_1, v_2, \ldots, v_n]$$

그리고, 보조정리 1의 식 (2)에 의하여 이 영역 질의 값을 구하기 위해 액세스되는 셀들의 집합 $U_{(w_1, w_2, \ldots, w_m)}$은 다음 식 (7)과 같다.

$$U_{(w_1, w_2, \ldots, w_m)} = \bigcup_{v_1\in\{w_1-1, w_1\}} \cdots \bigcup_{v_m\in\{w_m-1, w_m\}} \bigcup_{v_{m+1}\in\{l_{m+1}-1, h_{m+1}\}} \cdots \bigcup_{v_n\in\{l_n-1, h_n\}} \{P[v_1, v_2, \ldots, v_n]\} \quad (7)$$

여기서, $l_i\le w_i\le h_i (1\le i\le m)$이므로, $U_{(w_1, w_2, \ldots, w_m)}\subseteq U'$이 되고, 모든 w_i에 대해 성립하므로 이 성립한다.

(ii) $U\supseteq U'$: 모든 $P[v_1, v_2, \ldots, v_n] U\subseteq U'\in U'$에 대하여 이 셀을 포함하는 $U_{(w_1, w_2, \ldots, w_m)}$이 있음을 보인다. $v_i=l_i-1$이면 $w_i=l_i$로 설정하고, $v_i\neq l_i-1$이면 $w_i=v_i$로 설정하자. 그러면, $P[v_1, v_2, \ldots, v_n]$은 식 (7)에서 보인 바와 같이 $U_{(w_1, w_2, \ldots, w_m)}$에 속한다. 따라서, $P[v_1, v_2, \ldots, v_n] \in U_{(w_1, w_2, \ldots, w_m)}\subseteq U$이다.

(i)과 (ii)에 의해 $U=U'$이므로 액세스되는 셀의 집합은 U'(=식 (6))임이 성립한다. □

따름정리 4 정리 1에서 영역-그룹화 질의를 처리하기 위해, P로부터 읽어들이는 셀의 개수는 $\prod_{g_i(1\le i\le m)} (h_{g_i}-l_{g_i}+2)$이다.

증명: 그룹화 속성 D_i에 대해서 v_i는 h_i-l_i+2개의 값을 가질 수 있고, 그 외의 경우는 v_i는 두 개의 값을 갖는다. 따라서, 읽어들이는 셀의 개수는 $\prod_{g_i(1\le i\le m)} (h_{g_i}-l_{g_i}+2)$이다. □

정리 1에 따르면 액세스해야 되는 셀들은 그룹화 속성으로 구성되는 공간에 평행한 2^{n-m}개의 m-차원 초사각형을 형성한다. 즉, 2^{n-m}개의 클러스터를 형성한다. 그리고, 각 클러스터는 그룹화 속성 D_{g_i}에 대해서는 $[l_{g_i}-1, h_{g_i}]$를 범위로 하고, 그룹화 속성이 아닌 속성 D_{ng_i}에 대해서는 $l_{ng_i}-1$ 또는 h_{ng_i} 값을 갖는다.

4.3 알고리즘

그림 8은 정리 1에서 증명한 액세스되는 셀의 패턴을 이용하여 영역-그룹화 질의를 처리하는 알고리즘을 나타낸다. 입력은 n-차원 전방-합 배열 P, 그룹화 속성들의 집합 $G=\{D_{g_1}, D_{g_2}, \ldots, D_{g_m}\}$, 그리고 디멘젼들에 대한 질의 범위 $[l_i, h_i] (1\le i\le n)$이다. 그룹화 속성을 제외한 다른 속성들은 $D_{ng_1}, D_{ng_2}, \ldots, D_{ng_{n-m}}$이라 하자. 단계 1에서는 영역-그룹화 질의 계산에 사용되는 전방-합 배열의 셀들을 클러스터 단위로 주기억장치에 읽어들인다. 각 클러스터는 정리 1의 결과에 따라 그룹화 속성 $D_{g_i} (1\le i\le m)$에 대해서는 $[l_{g_i}-1, h_{g_i}]$이고, 비 그룹화 속성 $D_{ng_i} (1\le i\le n-m)$에 대해서는 크기가 1인 범위 $[v_{ng_i}, v_{ng_i}]$인 영역으로 구성된다. 여기서, v_{ng_i}는 $l_{ng_i}-1$ 또는 h_{ng_i}이다. 단계 2에서는 단계 1에서 검색한 셀들을 보조정리 1의 식 (2)에 대입하여 각 그룹 키 값에 대한

알고리즘 Range_Groupby
입력: (1) n-차원 전방-합 배열 P
(2) 그룹화 속성들의 집합 $G=\{D_{g_1}, D_{g_2}, ..., D_{g_m}\}$
(3) 디멘젼들에 대한 질의 범위 $[l_i,h_i](1\le i\le n)$
출력: 집계 연산 결과
알고리즘:
// 영역-그룹화 질의 계산에 사용되는 전방-합 배열의 셀들을 클러스터 단위로 주기억장치에 읽어들인다.
1 그룹화 속성이 아닌 속성들을 $D_{ng_1}, D_{ng_2}, ..., D_{ng_{n-m}}$이라 하자. 이 때, 각 $(D_{ng_1}, D_{ng_2}, ..., D_{ng_{n-m}})=(v_{ng_1}, v_{ng_2}, ..., v_{ng_{n-m}})(v_{ng_i}\in\{l_{ng_i}-1, h_{ng_i}\})$에 대하여 다음을 수행한다.
1.1 클러스터에 대한 질의 영역을 구성한다. 질의 영역은 그룹화 속성 $D_{g_i}(1\le i\le m)$에 대해서는 $[l_{g_i}-1,h_{g_i}]$이고, 다른 속성 $D_{ng_i}(1\le i\le n-m)$에 대해서는 $[v_{ng_i},v_{ng_i}]$이다.
1.2 구성한 질의 영역에 속하는 배열 P의 셀들을 검색한다
// 각 그룹 키 값에 대한 집계 값을 구한다.
2 각 $(D_{g_1}, D_{g_2}, ..., D_{g_m})=(v_{g_1}, v_{g_2}, ..., v_{g_m})\ (v_{g_i}\in[l_{g_i},h_{g_i}])$에 대하여 다음을 수행한다.
2.1 그룹화 속성 $D_{g_i}(1\le i\le m)$에 대해서는 $[v_{g_i},v_{g_i}]$를 대상으로 하고, 다른 속성 $D_{ng_i}(1\le i\le n-m)$에 대해서는 $[l_{ng_i},h_{ng_i}]$를 범위로 하는 영역-합 질의 값을 구한다. 이는 검색된 배열 P의 셀들을 다음의 식(보조정리 1의 식 (2))에 대입하여 구할 수 있다.

$$\sum_{v_{ng_1}=l_{ng_1}}^{h_{ng_1}}\cdots\sum_{v_{ng_{n-m}}=l_{ng_{n-m}}}^{h_{ng_{n-m}}}A[v_{g_1},...,V_{g_m},V_{ng_1},...V_{ng_{n-m}}]=$$
$$\sum_{v_{ng_1}\in=\{l_{ng_1}-1,h_{ng_1}\}}\cdots\sum_{v_{ng_{n-m}}\in=\{l_{ng_{n-m}}-1,h_{ng_{n-m}}\}}(-1)^aP[v_{g_1},...,v_{g_m},...,v_{ng_{n-m}}]$$

그림 8 전방-합 배열을 이용한 영역-그룹화 질의의 계산 알고리즘

집계 값을 구한다. 그룹 키 값이 $(v_{g_1}, v_{g_2}, ..., v_{g_m})(v_{g_i}\in[l_{g_i}, h_{g_i}])$일 때의 집계 값은 그룹화 속성 $D_{g_i}(1\le i\le m)$에 대해서는 $[v_{g_i}, v_{g_i}]$를 범위로 하고, 다른 속성 $D_{ng_i}(1\le i\le n-m)$에 대해서는 $[l_{ng_i}, h_{ng_i}]$를 범위로 하는 영역-합을 구함으로써 얻을 수 있다.

5. 성능 분석

본 절에서는 다음 네 가지 방법에 대하여 성능을 분석한다. 성능의 척도로는 액세스되는 셀의 개수와 선계산 결과를 저장하는데 필요한 공간 오버헤드를 사용한다.

No_Precomputation 주어진 질의 영역에 속하는 데이타 큐브의 셀들을 직접 액세스하여 영역-그룹화 질의를 처리하는 방법이다. 이 방법은 선계산된 결과를 사용하지 않아 추가적인 공간 오버헤드가 없는 대신 액세스되는 셀의 개수가 가장 많은 방법으로, 선계산된 결과를 사용하는 다른 방법들의 효용성을 보이는 비교 대상으로 사용된다.

Full_Precomputation 제 3절에서 소개한 방법으로, 가능한 그룹화 속성의 조합마다 그룹 전방-합 배열을 유지한다. 그리고, 각 그룹 키 값에 대한 집계 값을 구하기 위해 식 (5)에 제시된 셀들을 액세스한다.

Duplicated_Access 제 4.1절에서 소개한 바와 같이, 영역-그룹화 질의를 서로 다른 그룹 키 값을 갖는 여러 개의 부영역으로 나눈 후, 각 부영역에 대한 영역-합을 계산하는 방법이다. 이 방법은 Ho 등이 제안한 영역-합 처리 방법[6]으로부터 직관적으로 도출되는 방법으로서, 하나의 전방-합 배열을 사용한다. 각 그룹 키 값에 대한 집계 값을 구하기 위해 식 (2)에 따라 필요한 셀들을 액세스한다. 이 방법에서는 서로 다른 그룹화 속성에 대한 집계 값을 구할 때 셀들이 중복해서 액세스된다.

Range_Groupby 하나의 전방-합 배열을 가지고 제 5.2절에서 제안된 알고리즘 Range_Groupby를 사용하는 방법이다.

표 1은 디멘젼의 개수가 n이고, 그룹화 속성의 개수가 m인 영역-그룹화 연산에 대하여 액세스되는 셀의 개수와 공간 오버헤드를 나타낸다. 공간 오버헤드의 단위는 선계산된 결과에 포함되는 셀의 개수이다. No_Precomputation에서 액세스되는 셀의 개수에 대한 수식은 주어진 질의 범위에 속하는 셀들의 개수인 $\prod_{i(1\le i\le n)}(h_i-l_i+1)$이 된다. 그리고, Full_Precomputation과 Range_Groupby에 대한 수식들은 각각 따름정리 2와 따름정리 4의 결과이다. 다음으로, Duplicated_Access에서 액세스되는 셀의 개수는 $2^n\times\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$이

표 1 계산 방법에 따른 액세스되는 셀의 개수와 공간 오버헤드

	액세스되는 셀의 개수	공간 오버헤드
No_Precomputation	$\prod_{i(1\le i\le n)}(h_i-l_i+1)$	0
Full_Precomputation	$2^{n-m}\times\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$	$2^n\times\prod_{j(1\le j\le n)}\|D_j\|$
Dupliated_Access	$2^n\times\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$	$\prod_{j(1\le j\le n)}\|D_j\|$

된다. 이는 그룹 키 값마다 2^n개의 셀을 액세스해야 하고, 그룹 키 값의 개수는 $\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$이기 때문이다. 표를 보면, Range_Groupby와 Full_Computation의 액세스되는 셀들의 개수의 차이가 작음을 알 수 있다. Full_Precomputation과 Range_Groupby에서 하나의 그룹 키 값에 대한 집계 값을 구하기 위해 액세스되는 셀들의 개수는 각각 2^{n-m}과 2^n이다. 그럼에도 불구하고, Range_Groupby와 Full_Precomputation의 액세스되는 셀의 전체 개수의 차이가 작은 이유는, 제4절에서 논의한 바와 같이 Range_Groupby에서 서로 인접한 그룹 키 값에 대한 집계 값을 구하기 위해 액세스되는 셀들의 중복도가 크기 때문이다. 저장 공간 오버헤드에 대한 수식은 다음과 같이 구한다. No_Precomputation은 선계산된 결과를 용하지 않으므로 추가적인 공간 오버헤드가 없다. 그리고, Duplicated_Access와 Range_Groupby는 전방-합 배열 하나만을 사용하므로, 데이타 큐브와 동일하게 $\prod_{j(1\le j\le n)}|D_j|$개의 셀을 저장할 공간이 필요하다. Full_Precomputation은 가능한 그룹화 속성들의 조합($=2^n$)에 대해 그룹 전방-합 배열을 유지해야 하므로 $2^n\times\prod_{j(1\le j\le n)}|D_j|$개의 셀을 저장할 공간이 필요하다.

먼저, 액세스되는 셀의 개수를 비교한다. 그림 9는 그룹화 속성의 개수(m)가 고정되어 있을 때, 디멘젼에 주어진 범위 조건의 질의 범위 크기($h-l+1$)에 따라 액세스되는 셀의 개수를 분석한 결과이다. 문제를 단순화하기 위하여, 질의 범위의 크기는 모든 속성에 대해 동일한 크기를 가진다고 가정한다. 그림 9에서 사용된 디멘젼의 개수(n)는 다섯 개이고, 그룹화 속성의 개수(m)는 세 개이다. 그림에서 가로 축은 정규화된 질의 범위 크기, 즉 질의 범위 크기($h-l+1$)를 디멘젼의 카디날리티($|D|$)로 정규화한 값을 나타낸다. 그리고, 세로 축은 정규화된 액세스되는 셀의 개수, 즉 액세스되는 셀의 개수를 질의 영역을 그룹화 도메인 공간으로 프로젝션한 영역에 속한 셀의 개수($\prod_{g_i(1\le i\le m)}(h_{g_i}-l_{g_i}+1)$)로 정규화한 값을 나타낸다. 이렇게 정규화한 이유는, 각 그룹 키 값에 대하여 적어도 하나의 셀을 액세스해야 하는데, 그룹 키 값의 개수는 질의 영역을 그룹화 도메인 공간으로 프로젝션한 영역에 속한 셀의 개수와 동일하기 때문이다. 그림 9를 보면, Full_Precomputation과 Range_Groupby의 성능이 가장 우수함을 알 수 있다. 표 1을 보면 두 방법에서 액세스되는 셀의 개수는 서로 다른 값을 갖는데, 두 방법에서 액세스되는 셀의 개수의 비는 $\prod_{g_i(1\le i\le m)}\frac{h_g-l_g+1}{h_g-l_g+2}$로서 1에 가까운 값을 갖기 때문에, 그림 9의 그래프에서는 근사적으로 1로 가정하여 표시하였다. 이 비는 질의 범위가 커질수록 점점 더 1에 가까워진다. 그림 9를 보면, Duplicated_Access는 Full_Precomputation에 비하여 2^m 배나 많은 셀을 액세스하게 된다. 또한, No_Precomputation과 Full_Precomputation의 액세스되는 셀의 개수의 비는 $\prod_{ng_i(1\le i\le n-m)}\frac{h_{ng}-l_{ng}+1}{2}$이 되고, 일반적으로 질의 범위의 크기($h_{ng_i}-l_{ng_i}+1$)는 2보다 상당히 크기 때문에 No_Precomputation은 Full_Precomputation보다 매우 많은 셀을 액세스하게 된다. 그림 9에서, No_Precomputation은 디멘젼의 카디날리티 $|D|$에 따라 다른 방법들과의 성능 차이가 크게 나타나기 때문에 으로 나눈 값을 그래프로 표시한 것이다. 일반적으로, $|D|$는 10보다 상당히 크기 때문에 No_Precomputation과 다른 방법들과의 성능 차이는 그림에서 표시된 것보다 훨씬 더 크게 된다.

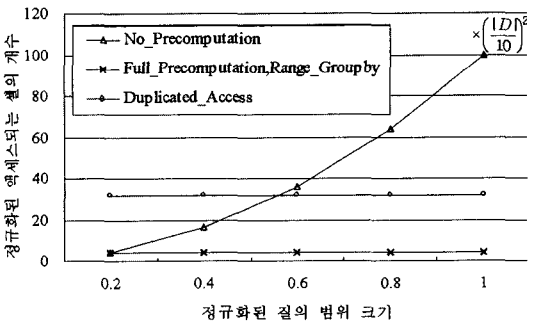


그림 9 디멘젼의 개수 n=5이고 그룹화 속성의 개수 m=3일 때 질의 범위의 크기($h-l+1$)에 따라 액세스되는 셀의 개수의 변화

그림 10은 정규화된 질의 범위의 크기 $(h-l+1)$가 고정되어 있을 때, 그룹화 속성의 개수에 따라 액세스되는 셀의 개수를 분석한 결과이다. 그림 10에서 사용된 디멘젼의 개수 $n=5$이고, 정규화된 질의 범위의 크기는 0.4이다. 그림에서 가로 축은 그룹화 속성의 개수 m을 나타내고, 세로 축은 정규화된 액세스되는 셀의 개수를 나타낸다. 그림을 보면, 그룹화 속성의 개수가 변화더라도 Full_Precomputation과 Range_Groupby의 성능이 가장 우수함을 알 수 있다. Duplicated_Access는 셀들을 중복해서 액세스하고, 그룹화 속성의 개수가 많아질수록 셀 당 중복해서 액세스하는 횟수가 더 많아지기 때문에, 그룹화 속성의 개수가 많아질수록 Range_Groupby와의 격차는 더욱 커지게 된다. 그리고, 그림 10을 보면, m이 커짐에 따라 Duplicated_Access가 No_Precomputation보다도 더 많은 셀들을 액세스함을 알 수 있다.

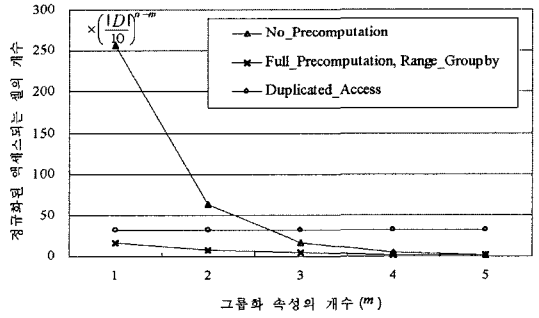


그림 10 디멘젼의 개수 $n=5$이고 질의 범위 크기 $(h-l+1)=60$일 때의 그룹화 속성의 개수에 따라 액세스되는 셀의 개수의 변화.

다음으로, 저장 공간 오버헤드를 비교한다. 그림 11은 그룹화 속성의 개수 $n=5$일 때, 디멘젼의 카디날리티에 따른 저장 공간 오버헤드를 나타낸다. 그림에서 가로축은 디멘젼의 카디날리티를 나타내고, 세로 축은 저장 공간 오버헤드를 나타낸다. 설명의 편의상, 모든 디멘젼의 카디날리티는 동일하다고 가정한다. 그리고, 이해를 돕기 위하여 저장 공간 오버헤드는 셀의 개수가 아닌 GB 단위로 그림에 표시한다. 여기서는, 하나의 셀이 네 바이트를 차지한다고 가정한다. 그림 11을 보면, Full_Precomputation이 Duplicated_Access와 Range_Groupby에 비해 $32(=2^5)$배나 많은 저장 공간을 사용함으로써 실제적으로 사용이 어려운 방법임을 알 수 있다. 예를 들어, 디멘젼의 카디날리티가 100일 때, Range_Groupby와 Duplicated_Access는 약 37GB의 저장 공간을 필요로 하는 반면에, Full_Precomputation은 1,192GB의 저장 공간을 필요로 한다.

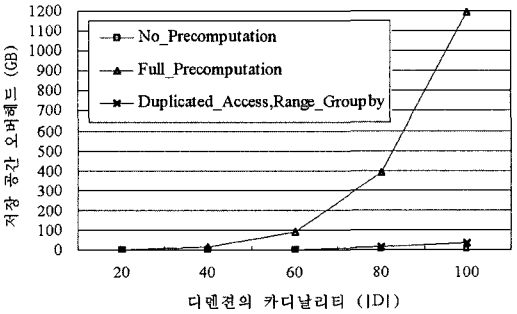


그림 11 디멘젼의 개수 $n=5$일 때의 디멘젼의 카디날리티($|D|$)에 따른 저장 공간 오버헤드의 변화

Full_Precomputation은 액세스되는 셀의 개수 측면에서 가장 우수하나, 저장 공간의 오버헤드가 매우 큰 단점이 있다. Duplicated_Access는 저장 공간 오버헤드는 작지만 액세스되는 셀의 개수 측면에서 Range_Groupby에 비하여 좋지 않다. 그리고, Range_Groupby는 선계산 결과를 사용하지 않는 No_Precomputation에 비하여 액세스되는 셀의 개수를 $\prod_{ng_i(1\le i\le n-m)} \frac{2}{h_{ng}-l_{ng}+2}$로 줄인다. 이것은, $rs_{ng}=\prod_{ng_i(1\le i\le n-m)} h_{ng_i}-l_{ng_i}+1$이라 하면, $O\left(\frac{2^{n-m}}{rs_{ng}}\right)$로 표현된다. 즉, No_Precomputation에 비하여 질의 영역을 비 그룹화 속성들에 대하여 프로젝션한 영역에 속한 셀의 개수로 나눈 만큼의 효과를 나타낸다. 따라서, 저장 공간 오버헤드가 작고 액세스되는 셀의 개수도 Full_Precomputation과 비슷한 Range_Groupby가 전체적 측면에서 볼 때 가장 우수한 계산 방법이다. 이는 Range_Groupby가 본 논문에서 제안한 알고리즘 Range_Groupby를 사용하여, 하나의 전방-합 배열만을 유지하면서, 액세스될 셀들을 클러스터 단위로 한번만 액세스하기 때문이다.

6. 결 론

본 논문에서는 OLAP에서의 집계 질의 중 영역-그룹화 질의를 정의하고, 이 질의를 빠르게 처리할 수 있는 방법을 제안하였다. 영역-그룹화 질의는 n-차원 데이타 큐브의 임의의 영역에 속한 셀들에 대하여 주어진 그룹화 속성들의 값의 조합에 따라 집계 값을 구하는 질의

이다. 이 질의는 관심의 대상이 되는 임의의 영역 내에서의 경향을 다각적인 측면에서 분석하기 위해서 OLAP에서 자주 사용되는 질의이다. 데이타 큐브의 모든 셀들을 대상으로 한 groupby 질의에 대해서는 많이 연구되었지만, 임의의 영역을 대상으로 한 영역-그룹화 질의에 대한 효율적인 처리 방법에 대해서는 연구된 바 없었다.

본 논문에서는 영역-그룹화 질의 중에서 가장 빈번하게 사용되는 집계 함수인 SUM을 사용하는 경우에 대하여 빠른 계산 방법을 제안하였다. 이 방법은 집계 함수가 COUNT와 AVERAGE인 경우에도 적용 가능하다. 제안한 방법은 전방-합 배열을 이용하여 효율적으로 영역-그룹화 질의를 계산한다. 제안한 방법의 특징은 다음과 같다. 첫째, 단지 하나의 전방-합 배열만을 사용하여 처리함으로써 저장 공간 오버헤드를 줄인다. 둘째, 서로 다른 그룹화 속성 값에 대한 집계 값을 구할 때 중복 액세스되는 셀들을, 클러스터 단위로 한번만 액세스함으로써 불필요한 셀의 액세스를 방지한다.

그리고, 이론적인 성능 분석을 통하여 제안한 알고리즘이 공간 오버헤드와 액세스되는 셀의 개수 측면에서, 데이타 큐브를 직접 액세스하거나 가능한 그룹화 속성의 조합마다 그룹 전방-합 배열을 유지하는 방법보다 우수한 계산 방법임을 보였다. 제안한 방법은 데이타 큐브를 직접 액세스하여 계산하는 방법보다 액세스되는 셀의 개수를 로 줄인다. 여기서, n은 디멘젼의 개수, m은 그룹화 속성의 개수, rs_{Ng}는 질의 영역을 비 그룹화 속성들에 대하여 프로젝션한 영역에 속한 셀의 개수이다. 또한, 가능한 그룹화 속성의 조합마다 그룹 전방-합 배열을 유지하는 방법과 비교할 때, 액세스되는 셀의 개수는 비슷하면서 공간 오버헤드는 로 줄인다.

본 논문은 데이타 큐브의 임의의 영역을 대상으로 하는 다른 OLAP 질의 처리 방법에 대한 기초를 제공할 것으로 예상된다. 또한, 본 논문은 영역-합 질의 처리를 위해 제안된 전방-합 배열이 영역-그룹화 질의 처리에도 유용하게 사용될 수 있음을 보였다.

참 고 문 헌

[1] Codd, E.F., Providing OLAP (On-Line Analytical Processing) to User-Analysts: An IT Mandate, Technical Report, E.F. Codd and Associates, 1993.

[2] Chaudhuri, S. and Dayal, U., "An Overview of Data Warehousing and OLAP Technology," *ACM SIGMOD Record*, Vol. 26, No. 1, pp. 65-74, Mar. 1997.

[3] Agarwal, S., Agrawal, R., Deshpande, P.M. et al., "On the Computation of Multidimensional Aggregations," In *Proc. Int'l Conf. on Very Large Data Bases*, pp. 506-521, Mumbai(Bombay), India, Sept. 1996.

[4] Chan, C.-Y. and Ioannidis, Y.E., "Hierarchical Cubes for Range-Sum Queries," In *Proc. Int'l Conf. on Very Large Data Bases*, pp. 675-686, Edinburgh, Scotland, 1999.

[5] Geffner, S., Agrawal, D., Abbadi, A. El, and Smith, T., "Relative Prefix Sums: An Efficient Approach for Querying Dynamic OLAP Data Cubes," In *Proc. Int'l Conf. on Data Engineering*, pp. 328-335, Sydney, Australia, Mar. 1999.

[6] Ho, C.-T., Agrawal, R., Megiddo, N., and Srikant R., "Range Queries in OLAP Data Cubes," In *Proc. Int'l Conf. on Management of Data*, pp. 73-88, ACM SIGMOD, Tucson, Arizona, June 1997.

[7] Agrawal, R., Gupta, A., and Sarawagi, S., "Modeling Multidimensional Databases," In *Proc. Int'l Conf. on Data Engineering*, pp. 232-243, Birmingham, U.K., Apr. 1997.

[8] Gray, J., Bosworth, A., Layman, A., and Pirahesh, H., "Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tabs, and Subtotals," In *Proc. Int'l Conf. on Data Engineering*, pp. 152-159, New Orleans, Louisiana, Feb. 1996.

[9] Chaudhuri, S., Krishnamurthy, S., Potamianos, S., and Shim, K., "Optimizing Queries with Materialized Views," In *Proc. Int'l Conf. on Data Engineering*, pp. 190-200, Taipei, Mar. 1995.

[10] Harinarayan, V., Rajaraman, A., and Ullman, J.D., "Implementing Data Cubes Efficiently," In *Proc. Int'l Conf. on Management of Data*, pp. 205-216, ACM SIGMOD, Montreal, Quebec, Canada, June 1996.

[11] Mumick, I.S., Quass, D., and Mumick, B.S., "Maintenance of Data Cubes and Summary Tables in a Warehouse," In *Proc. Int'l Conf. on Management of Data*, pp. 100-111, ACM SIGMOD, Tucson, Arizona, June 1997.

[12] Zhao, Y., Deshpande, P.M., and Naughton, J.F., "An Array-Based Algorithm for Simultaneous Multidimensional Aggregates," In *Proc. Int'l Conf. on Management of Data*, pp. 159-170, ACM SIGMOD, Tucson, Arizona, June 1997.

[13] Knuth, D.E., *The Art of Computer Programming, Volume 1: Fundamental Algorithms*, 3rd ed., Addison-Wesley, 1997.

이 영 구

1992년 2월 한국과학기술원 과학기술대 전산학과 학사. 1994년 2월 한국과학기술원 전산학과 석사. 2002년 2월 한국과학기술원 전산학과 박사. 2002년 3월 ~ 현재 한국과학기술원 BK21 Post-Doc. 관심분야는 OLAP, 데이타 웨어하우스, 데이타베이스 시스템. Access Method.

문 양 세

1991년 2월 한국과학기술원 과학기술대 전산학과 학사. 1993년 2월 한국과학기술원 전산학과 석사. 2001년 8월 한국과학기술원 전산학과 박사. 1991년 3월 ~ 2001년 6월 현대전자산업(주) 통신사업본부 선임연구원. 2001년 7월 ~ 2002년 2월 (주)인프라밸리 수석연구원. 관심분야는 데이타마이닝, Knowledge Discovery, 저장 시스템, Access Method, 2G/1X 이동통신 시스템, IMT-2000 이동통신 시스템

황 규 영

1973년 서울대학교 전자공학과 졸업(B.S). 1975년 한국과학기술원 전기 및 전자학과 졸업(M.S.). 1982년 Stanford University(M.S.). 1983년 Stanford University(Ph.D.). 1975년 ~ 1978년 국방과학연구소(ADD), 선임연구원. 1983년 ~ 1990년 IBM T.J. Watson Research Center, Research Staff Member. 1992년 ~ 1994년 한국정보과학회 데이타베이스 연구회(SIGDB) 운영위원장. 1995년 한국정보과학회 이사 겸 논문지 편집위원장. 1999년 ~ 2000년 한국정보과학회 부회장. Editior: the VLDB Journal, 1990년 ~ 현재 Editior: Distributed and Parallel Databases: An International Journal, 1991년 ~ 1995년 Editor: International Journal of Geographical Information Systems, 1994년 ~ 현재 Associate Editor: IEEE Data Engineering Bulletin, 1990년 ~ 1993년 IEEE Transactions on Knowledge and Data Engineering, 2002년 ~ 현재. 1998년 ~ 2004년 Trustee, The VLDB Endowment. 1999년 ~ 2005년 Steering Committee Member, DASFAA. 1999년 ~ 현재 한국과학기술원 전자전산학과 전산학전공 교수. 1990년 ~ 현재 첨단정보기술연구센터(과학재단 우수연구센터) 소장. 관심분야는 데이타베이스 시스템, 멀티미디어, GIS.