스트림 데이터의 다차원 분석에서 평균응답시간을 줄이는 스트림 큐브*

도기석[0], 박석
서강대학교
{neokid, spark}@dblab.sogang.ac.kr

A stream cube to reduce the average response time in the multi-dimensional analysis of stream data

Kiseok Do[0] Seog Park
Sogang University

요 약

유비쿼터스 환경이 도래함에 따라 데이터 흐름이 신속하고 연속적으로 변화하고 있다. 이러한 스트림 형태의 데이터는 데이터의 치명적 변화, 자주 발생하지 않는 패턴 등의 관점에서 데이터 분석을 필요로 하고 있다. 본 논문에서는 다단계의 추상화 데이터 분석이 용이한 다차원 분석에 기반하여 고정적인 공간 활용만이 가능했던 기존 방식을 살펴본 후 이를 유동적으로 보완하여 공간 비용을 최소화 하면서 평균응답시간을 줄여주는 방법에 대해 논의한다. 또한 제안 방법의 시공간 비용을 수식으로 증명하고 기존 방법과의 비교 실험을 통하여 성능을 평가해 본다.

1. 서 론

데이터 흐름이 신속하고 연속적으로 변화함에 따라 센서 네트워크 등의 영역에서 연속적인 스트림 데이터에 요구가 많아지고 있다. 스트림 데이터는 시간에 따라 연속적이고 복잡하며 한시적인 접근만이 가능하다. 또한 제한된 메모리를 사용하며 동적으로 변화하고, 시간에 따른 순서를 가지기 때문에 랜덤한 접근이 불가능하다[1][2]. 스트림 데이터의 분석은 데이터의 치명적인 변화나 자주 발생하지 않는 패턴, 관심 있는 데이터의 모니터링 등의 문제 해결을 위해 필요하며 관련 연구들로 의사 결정 트리[3], 클러스터링[4] 등의 논문들이 발표되었다.

스트림 데이터 분석은 상위 단계의 추상적인 다차원 분석이 요구된다. 그러나 다차원 분석은 상위 단계에서 분석되어 가공되어야 하므로 온라인 분석이 불가능하기 때문에 다차원 데이터를 상위 단계로 추상화하여 미리 저장할 필요가 있다. 이에 본 논문은 스트림 데이터에서 다차원, 다단계 분석을 위해 사용되는 스트림 큐브 기법을 기반으로 제한된 공간에서 질의 응답시간이 더욱 향상된 방법을 제안한다.

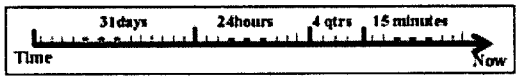
2. 기존 데이터베이스와 스트림 데이터의 다차원 분석

2.1 기존 데이터베이스에서 이용되는 다차원 분석

OLAP을 사용한 다차원 분석은 관계형 데이터베이스를 기반으로 데이터 웨어하우스를 구축한다. 데이터 웨어하우스는 규모가 매우 크기 때문에 필요한 부분만을 추출하고 OLAP 서버를 구성하여 다차원적인 데이터 분석을 수행한다. 이 방법은 데이터를 여러 차원 속성으로 나누어서 데이터의 조합을 미리 계산하므로 다양한 각도의 분석이 용이하며, 요약 정보를 통해 대화식 분석이 가능하고 요약 정보를 통해 즉시 필요한 데이터와 대화할 수 있는 장점이 있다.

2.2 스트림 데이터 환경에서 다차원 분석

요약 데이터를 추가 저장되는 데이터 큐브는 공간 비용이 크기 때문에 스트림 환경에서는 적합하지 못하다. 기존 연구[5][6]에서는 스트림 데이터의 특성을 고려하여 데이터 큐브를 효율적으로 재구성한 스트림 큐브를 발표하였다.

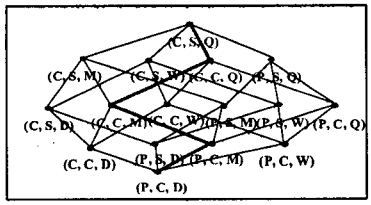


[그림 1] 한 달간의 tilted time frame

이 논문은 첫째로 시간 차원의 요약을 위해 tilted time frame을 사용한다. 스트림 데이터 분석가들은 최근의 변화에 더욱 관심이 많기 때문에 최근일수록 정밀하게 저장되길 원한다. [그림 1]의 tilted time frame 구조는 15분은 1쿼터, 4쿼터는 한 시간의 정밀도로 계산된다. 만약 매 분마다 데이터가 들어오면 정밀도가 버켓 그룹 Minutes에 분마다 데이터를 저장한다. 15분이 저장되면 모두 집계한 값을 quarters 버켓에 저장하고 minutes의 첫 버켓부터 다시 덮어쓴다. 이 방식은 31×24×4×15=44640의 '분' 버켓을 통해 한 달을 구성하는 기존 방식을 31+24+4+15= 74로 약 603배 줄인다. 그러나 '지난 408분 전에 값은?'과 같은 세세한 질의는 수행할 수 없다. 그러나 이 방법은 시간과 공간의 효율성과 다단계 분류의 취약점을 절충할 수 있는 적합한 구조이다.

셋째로 전체 큐보이드를 o-layer와 m-layer의 두 임계 계층으로 나누어 그 사이의 큐보이드의 셀을 저장한다. o-layer는 분석가가 추상적인 분석을 위해 분석을 필요로 하는 최상위 계층이며, m-layer는 비용의 효율성이 떨어지고 분석가에게 흥미가 떨어지는 계층을 제외한 최하위 계층이다. 이 두 계층으로 사이의 큐보이드만을 저장하여 공간 비용을 줄인다.

넷째로 각 단계마다 이용되는 질의가 가장 많이 포함된 큐보이드를 인기경로로 설정하고 인기경로에 포함된 큐보이드만을 메모리에 저장한다. [그림 2]처럼 격자 구조의 단계마다 한 큐보이드를 갖는 인기경로는 단일 스캔이 가능한 H-tree 기법[7]을 사용하여 다중 스캔 문제를 해결한다.인기경로에 포함되지 않는 셀은 가장 가까운 인기경로 내의 큐보이드에 저장된 셀을 사용한다. 특정 질의집합에 따라 각 질의의 우선순위가 크게 변경되지 않는 이상 인기경로는 고정적이다. 기존 논문은 인기경로는 통계적 분석이나 경험에 의해서 계산되어 입력으로 주어진다고 가정한다.



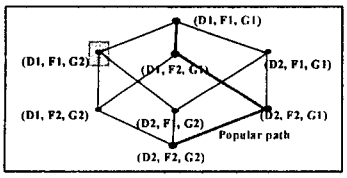
[그림 2] 큐보이드의 격자 구조와 인기경로

3. 스트림 데이터의 평균응답시간을 줄이는 스트림 큐브

스트림 큐브는 단계별로 질의집합에서 사용하는 질의가 가장 많이 포함된 큐보이드를 사용하여 인기경로를 구성한다. 따라서 사용 빈도가 높고 인기경로와 멀리 떨어진 비인기경로 질의는 집계 연산비용이 많이 소요되어 전체 질의집합의 응답시간을 크게 증가시키는 문제가 발생한다.

3.1 스트림 큐브의 문제점

*본 연구는 한국과학재단 목적기초연구(R01-2003-000-10395-0) 지원으로 수행되었음



[그림 3] 전력발전소 시스템의 격자 구조

스트림 큐브는 기존 방식에 비해 공간이 축소되지만 시간이 증가된다. [그림 3]의 (D1, F2, G2), (D2, F1, G2), (D2, F1, G1), (D1, F1, G2)의 집계 연산비용은 각각 3배, 4배, 4배, 12배가 늘어난다. 특히 (D1, F1, G2)는 (D3, F2, G2)와 (D1, F2, G2) 두 단계를 거쳐서 집계되어 비용이 크게 높아진다. 이러한 집계 연산비용은 질의응답시간의 증가를 초래한다.

3.2 평균응답시간을 줄이는 스트림 큐브의 구성

위와 같은 문제점으로 인해 응답시간을 크게 증가시키는 특정 질의를 재구성하여 질의의 평균응답시간을 최소화 할 필요성이 있다. 이를 해결하기 위해 전력발전소 시스템의 예를 통해 우선 가정 및 환경에 대해 살펴본다.

전력발전소 시스템은 구역, 시설, 계열 별로 구분된 측정기를 통해 10초마다 kw 단위의 전력량을 출력한다. 스트림 큐브의 각 셀은 tilted time frame 구조로 구성되며 원시데이터가 변경됨에 따라 수정에 따른 유지비용이 발생하며, tilted time frame의 전체 버켓이 집계되는 집계비용이 발생한다. 또한 모든 셀은 동일 한 tilted time frame으로 구성되기 때문에 하위 셀의 집계 횟수가 같은 상위 셀은 동일한 응답시간을 가진다. 이 환경은 인기 경로가 미리 주어지고 거의 변하지 않으며 복잡한 집계 연산이 수행되어 유지비용보다 집계연산비용이 훨씬 크다고 가정한다.

3.2.1 스트림 큐브의 재구성

질의의 응답시간은 저장된 셀을 이용하여 저장되지 않은 셀에 대응하는 질의를 처리하기 위한 집계 연산비용과 저장된 셀을 수정하는 유지비용으로 크게 나눌 수 있다. 전체 질의집합의 평균응답시간을 줄이기 위해서는 사용 가능한 공간이 S일 때 저장된 셀의 집합 M에 대해 전체 질의집합의 집계 연산비용과 유지비용의 합 COST(M)을 최소화해야 한다. 이 때 Q(c, M)을 저장된 셀의 집합 M을 이용하여 저장되지 않은 셀 c의 질의를 계산할 때 집계 연산비용이라 하고, U(c, M)을 저장된 셀의 집합 M을 이용하여 저장된 셀 c를 유지하는 비용이라 하자. 또한 f_C, g_C를 각각 질의와 수정의 빈도수라 하고 k는 저장되지 않은 셀의 수, m은 저장된 셀의 수라고 하자. 이 같은 가정하에서 COST(M)은 다음과 같이 일반화할 수 있고 저장된 셀의 집합 M의 크기는 S보다 작도록 제한된다.

$$COST(M) = \sum_{i=1}^{k} f_{C_i} \times Q(C_i, M) + \sum_{i=1}^{m} g_{C_i} \times U(C_i, M), \quad \sum_{c \in M} S_c \le S$$

이러한 COST(M)의 비용이 최소화되는 M을 선택하는 문제를 [11]에서는 greedy한 방법으로 해결한다. 그러나 [8]은 모든 질의의 사용 빈도수가 랜덤하고 저장을 위해 선택된 질의가 다른 질의에 영향을 주는 형태이지만 위의 문제는 인기경로의 셀에 대응하는 질의의 빈도수가 훨씬 높기 때문에 인기경로에 저장된 셀을 이용하여 비인기경로의 셀을 구성한다. 따라서 위의 문제는 인기경로의 셀에 대응하는 질의를 우선적으로 고려하여 계산해야 한다.

3.2.1.1 저장된 셀과 저장되지 않은 셀의 응답시간 비용

저장된 셀 집합 M을 이용하여 임의의 셀 c를 구성하면 응답시간 COST(c)는 다음과 같다.

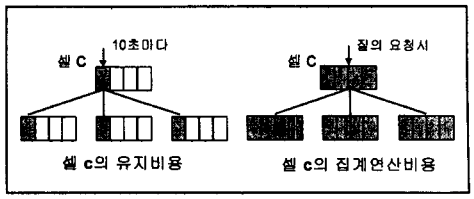
$$COST(c) = f_c \times Q(c, M) + g_c \times U(c, M)$$

저장된 셀 c의 응답시간 비용은tilted time frame의 버켓들이 저장되어 있기 때문에 집계 연산비용은 발생되지 않으며 유지비용만 발생한다. 따라서 셀 c의 집계 연산비용은 $f_c \times Q(c, M) = 0$이므로 COST(c)는 아래와 같다.

$$COST(c) = g_c \times U(c, M)$$

반대로 저장되지 않은 셀 c의 응답시간 비용은tilted time frame의 버켓들이 저장되어 있지 않으므로 유지비용은 사용되지 않으며 집계 연산비용만 발생한다. 따라서 셀 c의 유지비용은 $g_c \times U(c, M) = 0$이므로 COST(c)는 아래와 같다.

$$COST(c) = f_c \times Q(c, M)$$



[그림 4] 셀의 유지비용과 집계 연산비용

앞선 전력발전소 시스템의 예를 이용하여 셀의 유지비용과 집계 연산비용을 표현하면 [그림 4]와 같다. 셀 c의 유지비용을 살펴보면 하위셀 M을 이용하여 c를 구성하는 비용 U(c, M)은 10초 단위의 한 버켓만 집계하는 비용이 발생한다. 질의집합이 처리되는 시간이 T일 때 10초마다 수정되기 때문에 g_C=T/10이 된다. 셀 c의 유지비용은 T시간 동안 U(c, M)의 수정비용을 갖는 셀 c가 수정 빈도수만큼 처리되기 때문에 $g_c \times U(c, M)$으로 표현된다.

셀 c의 집계 연산비용을 살펴보면 하위셀 M을 이용하여 c를 구성하는 비용이 Q(c, M) 은 셀의 전체 버켓을 집계하는 비용이 발생한다. 질의집합이 처리되는 시간 T이내에 셀 c가 요청되는 횟수를 f_C라고 하면 셀 c의 집계 연산비용은 $f_c \times Q(c, M)$으로 표현할 수 있다.

COST(M)을 최소화하기 위해 인기경로와 비인기경로의 셀을 구분하여 살펴본다. 인기경로에 저장되는 셀의 집합을 M_P라 하고 COST(M_P)는 인기경로의 모든 셀을 사용할 때 발생하는 응답시간의 합이라 하자. 제안기법은 인기경로의 셀뿐만 아니라 비인기경로의 응답시간 비용이 낮은 셀도 저장한다. 응답시간 비용이 낮은 비인기경로에서 저장되는 셀의 집합을 M_N이라 하고 COST(M_N)은 비인기경로의 모든 셀을 사용할 때 발생하는 응답시간의 합이라 한다. 위의 문제는 인기경로의 모든 셀의 응답시간 비용 COST(M_P)와 비인기경로의 모든 셀의 응답시간 비용COST(M_N)의 합인 COST(M)을 가장 작게 하는 M_P와 M_N의 선택을 통해 해결될 수 있다. ($M = M_P + M_N$)

$$Min[COST(M) = COST(M_P) + COST(M_N)], \quad \sum_{C \in M_P, M_N} S_C \le S$$

3.2.1.2 인기경로에 속한 셀의 이익(Benefit)

우선 인기경로의 셀이 저장됨으로써 응답시간 비용이 얼마나 감소될 수 있는지 살펴보자. 이렇게 감소되는 응답시간의 비용을 인기경로 셀의 이익이라 한다. 인기경로의 셀 c에 대해 c가 저장 되었을 때 비용과 저장되지 않았을 때 비용을 각각 S_COST(c), N_COST(c)라고 하면 아래와 같이 이익이 계산된다.

$$Benefit(c) = N\_COST(c) - S\_COST(c)$$

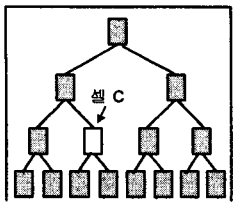
이러한 이익은 셀 c가 저장되지 않을 때 응답시간 비용이 감소되고 그에 반해 저장된 셀이 수정되는 유지비용이 증가되기 때문에 위와 같이 계산된다. 위에서 정의된 저장된 셀의 응답시간 비용의 식을 통해 계산하면 S_COST(c)는 다음과 같다.

$$S\_COST(c) = g_c \times U(c, M)$$

N_COST(c)를 구하는 방법은 조금 복합하다. 인기경로의 셀 c가 삭제된다면 인기경로의 저장된 하위 셀의 집계연산을 사용하여 셀 c가 구성되어야 한다. 또한 삭제하려는 인기경로의 셀 c는 상위 셀을 집계 연산하기 위해서 사용되기 때문에 c를 삭제한다면 상위 셀의 집계 연산비용이 증가한다. 이렇게 c를 이용하여 집계 연산되는 상위 셀의 개수를 N이라 하자. 이 때 N은 자신의 셀도 집계되므로 포함된다. 또한 각 N에 포함된 셀 C_i들의 빈도수를 f_{C_i}라고 한다면 N_COST(c)는 다음과 같으며 인기경로의 셀 c의 이익은 아래와 같다.

$$N\_COST(c) = \sum_{i=1}^{N} f_{C_i} \times Q(C_i, M)$$

$$Benefit(c) = \sum_{i=1}^{N} f_{C_i} \times Q(C_i, M) - g_c \times U(c, M)$$



[그림 5] 중간에서 삭제된 인기경로의 셀

만약 [그림 5]와 같이 인기경로의 셀이 중간에서 삭제된다면 상위 셀을 구성할 수 없다. 따라서 인기경로의 셀을 삭제하기 위해서는 가장 상위에 있는 셀부터 삭제해 주어야 한다. 또한 위의 식을 통해 인기경로의 셀의 이익은 그 셀을 사용하는 모든 상위 셀에 대한 집계 연산비용이 추가되는 것을 알 수 있다. 인기경로의 하위 셀일수록 상위 셀들이 많이 사용하기 때문에 이익이 낮은 인기경로의 상위 셀부터 삭제되어야 한다.

3.2.1.3 비인기경로에 속한 셀의 이익

비인기경로의 셀을 저장할 때 얼마나 응답시간 비용이 감소되는지 살펴보자. 이를 비인기경로의 셀의 이익이라 하며 이는 인기경로의 이익과 동일하게 계산된다.

$$Benefit\ (c) = N\_COST\ (c) - S\_COST\ (c)$$

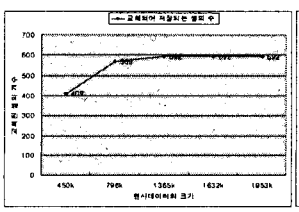
빈도수가 높은 비인기경로의 셀이 소수이기 때문에 비인기경로의 비용이 높은 셀은 각 큐보이드에 드물게 발생한다. 저장된 비인기경로의 전체 하위 셀이 저장되어야 상위 셀을 구성할 수 있기 때문에 상위 셀을 구성할 수 있는 경우가 극히 드물다. 또한 저장된 비인기경로의 셀을 이용하여 상위 셀을 구성하면 인기경로를 이용한 비용과 비교하고 만약비인기경로의 셀의 비용이 적다면 상위 셀을 구성했던 인기경로의 셀들의 이익을 삭제해야 한다. 이러한 문제는 비인기경로의 셀이 추가됨에 따라 고정된 인기경로의 이익이 변경되는 형태이므로 동적 프로그래밍으로 해결되어야 한다. 그러므로 이를 해결하기 위한 복잡도가 매우 크다. 따라서 비인기경로의 저장된 셀을 이용하여 상위 셀들을 구성하지 않는다고 가정한다. 그렇다면 비인기경로의 셀은 저장된 인기경로의 셀을 이용하여 항상 구성된다고 볼 수 있으므로 S_COST(c)와 N_COST(c)는 다음과 같고 비인기경로 셀c의 이익은 아래와 같다.

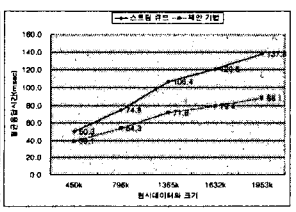
$$S\_COST(c) = g_c \times U(c,M)$$
$$N\_COST(c) = f_c \times Q(c,M)$$
$$Benefit(c) = f_c \times Q(c,M) - g_c \times U(c,M)$$

비인기경로의 셀 c의 빈도수와 집계 연산비용이 높다면 S_COST(c)가 증가하므로 c의 이익이 향상된다. 일반적으로 빈도수도 높으면서 집계 연산비용이 높은 셀은 소수이며 이러한 셀은 다른 비인기경로의 셀보다 훨씬 높은 응답시간 비용을 가진다. 반면에 인기경로의 셀 c는 집계 연산비용과 빈도수가 적고 c를 사용하는 상위 셀의 수가 적다면 이익이 낮아진다.

만약 비인기경로의 셀의 이익이 인기경로의 셀의 이익보다 높은 경우가 발생한다면 질의 응답시간 비용의 크기가 역전되기 때문에 인기경로의 셀을 비인기경로의 셀로 바꾸어 저장하는 것이 전체 응답시간을 더욱 줄일 수 있는 방법이 된다. 이 같은 방법은 사용하여 제한된 공간 S내에서 M_P의 크기를 줄이고 M_N을 증가시켜 전체 질의의 응답시간 COST(M)을 줄일 수 있다($M = M_P + M_N$)

4. 성능 평가

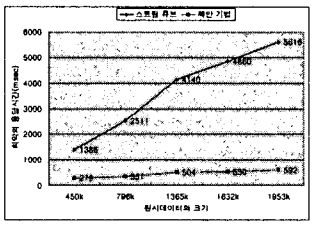




[그림 6] 원시데이터에 따른 교체 셀 개수 [그림 7] 원시데이터에 따른 평균응답시간

[그림 6]은 원시데이터에 따라 인기경로 셀이 비인기경로 셀로 교체되는 수를 나타낸다. 원시 데이터의 크기가 낮다면 널 값을 가진 하위 셀이 많기 때문에 비인기경로의 집계 연산비용이 적어지므로 셀의 이익이 줄어든다. 원시 데이터가 어느 한계에 이르면 교체 셀이 일정하게 유지된다. 인기경로 셀의 이익은 이용되는 상위 셀이 많을수록 이익이 높아지기 때문에 인기경로의 상위 셀부터 교체된다.

[그림 7]은 원시 데이터 크기에 따른 질의집합의 평균응답시간을 나타낸다. 원시 데이터가 증가함에 따라서 널 값을 가진 셀이 줄어들기 때문에 평균응답시간이 높아진다. 제안 기법은 비인기경로의 응답시간이 높은 셀을 응답시간이 낮은 인기경로의 셀과 교체하여 저장하기 때문에 스트림 큐브 기법보다 약 20~30% 줄어드는 것을 확인할 수 있다.



[그림 8] 원시데이터에 따른 최악의 응답시간

[그림 8]은 원시 데이터 크기에 따른 최악의 응답시간을 나타낸다. 스트림 큐브는 인기경로에서 멀리 떨어지고 빈도수가 높은 셀의 집계 연산비용이 크게 증가하기 때문에 응답시간이 폭등한다. 그러나 제안 기법은 응답시간이 높은 특정 질의의 셀을 응답시간이 낮은 인기경로의 셀과 대체하여 저장하므로 최악의 응답시간이 작아지는 것을 알 수 있다

5. 결론

본 논문은 인기경로의 셀이 저장될 때 발생하는 응답시간의 이익과 비인기경로의 셀이 저장될 때 발생하는 응답시간의 이익을 비교하여 비인기경로 셀의 이익이 높다면 대체하여 저장하는 방법을 제시한다. 이를 통해 전체 질의집합의 평균응답시간을 줄이는 스트림 큐브를 구성한다. 제안 기법에서는 tilted time frame의 구체적인 활용방법을 제시한다. 자주 접근되는 셀을 미리 저장하여 일정 주기로 한 버켓만 수정하고, 자주 접근되지 않는 셀은 질의가 요청될 때마다 전체 버켓을 집계하여 평균응답시간을 줄인다. 또한 우선순위를 큐보이드 단위에서 질의 단위로 변경하여 정밀한 셀의 비용을 계산하고, 인기경로와 비인기경로를 비교하여 평균응답시간이 낮아지는 쪽으로 질의를 유동적으로 저장한다. 그리고 기존 방법에서 매우 높은 응답시간을 갖는 특정 셀에 해당되는 질의를 미리 저장하여 최악의 응답시간을 줄인다.

참조 문헌

1. Shivnath Babu, Jennifer Widom. "Continuous queries over data streams". ACM SIGMOD Record, 2001. pp.109-120

2. B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. "Models and issues in data stream systems". In Proceedings of 21st ACM Symposium (PODS 2002). pp. 1-16

3. G. Hulten, L. Spencer, and P. Domingos. "Mining time-changing data streams". Proceedings of the 7th ACM SIGKDD (KDD 2001). pp. 97-106

4. S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. "Clustering data streams". In Proceedings of the Annual Symposium on Foundations of Computer Science. IEEE 2000. pp. 1-8

5. Y. Chen, G. Dong, J. Han, J. Pei, B. W. Wah, and J. Wang. "Online analytical processing stream data: Is it feasible?" 2002 Workshop on Research Issues in Data Mining and Knowledge Discovery. pp.13-24

6. Y. Chen, G. Dong, J. Han, B. W. Wah, and J. Wang. "Multi-dimensional regression analysis of time-series data streams". In VLDB International Conference, 2002. pp. 323-334

7. Jiawei Han, Jian Pei, Guozhu Dong, Ke Wang. "Efficient computation of iceberg cubes with complex measures". Proceedings of the 2001 ACM SIGMOD international conference on Management of data. pp. 1-12

8. Himanshu Gupta. "Selection of views to materialize in a data warehouse". In Proceedings of the International Conference on Database Theory, 1997. pp. 98-112