맵리듀스에서 Grouping Sets 질의의 효율적인 계산 기법

박소정, 박은주, 이기용
숙명여자대학교 컴퓨터과학부
e-mail: {sojeongpark, eunjupark, kiyonglee}@sookmyung.ac.kr

Efficient Computation of Grouping Sets Queries Using MapReduce

So-Jeong Park, Eun-Ju Park, Ki Yong Lee
Division of Computer Science, Sookmyung Women's University

요 약

맵리듀스(MapReduce)는 대용량의 데이터를 여러 컴퓨터에서 분산, 병렬 처리하는 프레임워크이다. Grouping sets 질의는 사용자가 지정한 여러 개의 group-by들을 모두 구하는 질의로서, 롤업(rollup)과 큐브(cube)가 너무 많은 결과를 반환하는 단점을 보완하여 원하는 group-by들에 대한 결과만 얻을 수 있도록 한다. 본 논문은 맵리듀스 환경에서 grouping sets 질의를 효율적으로 계산하는 방법을 제안한다. 제안 방법은 grouping sets 질의를 2개의 맵리듀스 잡(job)을 통해 단계적으로 계산한다. 첫 번째 맵리듀스 잡은 grouping sets 질의에 포함된 group-by들이 모두 계산될 수 있는 '부모' group-by를 먼저 계산한다. 두 번째 맵리듀스 잡은 부모 group-by를 입력으로 하여 grouping sets 질의에 포함된 group-by들을 각각 계산한다. 부모 group-by의 크기가 입력 데이터의 크기에 비해 매우 작은 경우, 제안 방법은 입력 데이터로부터 각 group-by를 독립적으로 구하는 단순 방법보다 좋은 성능을 보인다. 실험을 통해 제안 방법이 각 group-by를 독립적으로 구하는 단순 방법보다 좋은 성능을 가짐을 보인다.

1. 서론

디지털 경제의 확산으로 많은 정보와 데이터가 생산되면서 빅데이터(Big Data)[1]에 대한 관심이 매우 커지고 있다. 빅데이터란 용량이 매우 크거나, 증가 속도가 매우 빠르거나, 형태가 매우 다양해서 현존 기술로는 효율적으로 처리하기 어려운 데이터를 말한다[2]. 이러한 빅데이터를 쉽고 효율적으로 처리하기 위해 맵리듀스(MapReduce)라는 기법이 제안되었다[3]. 맵리듀스는 여러 컴퓨터에 분산되어 저장된 데이터를 손쉽게 병렬처리 할 수 있도록 해주는 프로그래밍 모델이다.

본 논문에서는 맵리듀스 환경에서 grouping sets 질의를 효율적으로 계산하는 방법을 제안한다. Grouping sets 질의는 group-by의 확장된 형태로, 사용자가 지정한 여러 개의 group-by를 모두 구하는 질의이다[4]. 따라서 grouping sets 질의의 결과는 grouping sets 질의에 포함된 각 group-by들을 모두 합한 결과와 동일하다. 맵리듀스 환경에서 grouping sets 질의를 처리하기 위한 연구는 이미 진행된 바가 있다[5]. 하지만 [5]는 grouping sets 질의에 포함된 각 group-by를 각각 독립적으로 구하는 단순한 방법을 사용하고 있다. [4]는 grouping sets 질의를 효율적으로 계산하는 방법을 제안하였으나, 맵리듀스 환경이 아닌 단일 노드 환경을 가정한다.

반면에 제안 방법은 grouping sets 질의를 맵리듀스 환경에서 효율적으로 처리하는 알고리즘을 사용한다. 제안 방법은 grouping sets 질의를 2개의 맵리듀스 잡(job)을 통해 단계적으로 계산한다. 첫 번째 맵리듀스 잡은 grouping sets 질의에 포함된 group-by들이 모두 계산될 수 있는 '부모' group-by를 먼저 계산한다. ('부모' group-by는 4장에서 자세히 설명한다.) 두 번째 맵리듀스 잡은 부모 group-by를 입력으로 하여 grouping sets 질의에 포함된 group-by들을 각각 계산한다. 제안 방법은 2개의 맵리듀스 잡을 수행하지만, 부모 group-by의 크기가 원래 입력 데이터의 크기에 비해 매우 작은 경우, 각 group-by들을 원래 입력 데이터로부터 각각 독립적으로 구하는 단순 방법보다 훨씬 빠르게 모든 group-by들을 계산할 수 있다. 실제 수행 시간을 측정한 실험을 통해, 제안 방법이 기존 방법에 비해 grouping sets 질의의 계산 속도를 크게 향상시킴을 보인다.

본 논문의 구성은 다음과 같다. 2장에서는 grouping sets 질의와 맵리듀스에 관한 배경 지식을 설명하고, 3장에서는 관련 연구를 기술한다. 4장에서는 제안 방법을 자세히 설명한다. 5장에서는 실험 결과를 제시하고, 6장에서 결론을 맺는다.

2. 배경지식

2.1 Grouping Sets 질의

Group-by는 group-by 절 다음에 지정된 애트리뷰트들의 값을 기준으로 select 문의 결과를 그룹화하는 질의이다. 이 때 그룹화의 기준이 되는 애트리뷰트들을 그룹화 애트리뷰트라 한다. 본 논문에서는 group-by를 그의 그룹화 애트리뷰트들만으로 간략히 나타낸다. 예를 들어 그룹화 애트리뷰트가 a, b인 group-by 질의는 ab로 나타낸다.

Grouping sets 질의는 사용자가 지정한 여러 개의 group-by를 모두 구하는 질의이다. Grouping sets 질의의 결과는 사용자가 지정한 각 group-by의 결과를 모두 합한 결과와 동일하다. 예를 들어 어떤 릴레이션 $F(a, b, c, m)$이 4개의 튜플 {(1, 1, 1, 2), (1, 1, 3, 5), (1, 2, 3, 4), (2, 3, 4, 5)}을 가진다면, 다음의 SQL로 표현된 grouping sets 질의는 두 개의 group-by ab와 bc를 계산하며, 그 결과는 {(1, 1, null, 7), (1, 2, null, 4), (2, 3, null, 5), (null, 1, 1, 2), (null, 1, 3, 5), (null, 2, 3, 4), (null, 3, 4, 5)}가 된다.

```
SELECT a, b, c, SUM(m)
FROM F
GROUP BY GROUPING SETS ((a, b), (b, c))
```

Grouping sets 질의는 그에 포함된 group-by를 모두 계산해야 하기 때문에 일반적으로 계산 비용이 매우 크다.

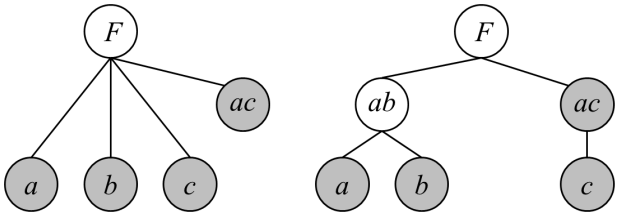
2.2 맵리듀스(MapReduce)

맵리듀스는 여러 컴퓨터에 분산 저장된 대용량의 데이터를 병렬처리하는 프레임워크이다[3]. 맵리듀스의 작업 단위는 잡(job)이라 하며, 각 잡은 한 쌍의 맵(Map)과 리듀스(Reduce) 함수로 표현된다. 사용자는 수행을 원하는 작업을 맵과 리듀스 함수로 표현해야 한다. 맵 함수는 하나의 키-값 쌍(pair) (key, value)를 입력으로 받아 하나 이상의 키-값 쌍 (key1, value1), (key2, value2), ...를 출력으로 내보낸다. 맵 함수가 내보낸 키-값 쌍들은 key 값으로 정렬된 뒤, 같은 key 값을 가지는 (key, value) 끼리 모여 (key, [value1, value2, ...]) 형태로 변환된 후, 리듀스 함수의 입력으로 전달된다. 리듀스 함수는 (key, [value1, value2, ...]) 형태의 입력을 받아 하나 이상의 키-값 쌍 (key1, value1), (key2, value2), ...를 출력으로 내보내고, 이것이 맵리듀스 잡의 최종 수행 결과가 된다. 일반적으로 맵 함수가 내보낸 (key, value)의 개수가 많으면 많을수록, 이들을 리듀스 함수로 전달하는 비용이 증가하여 맵리듀스 잡의 수행 비용이 커지게 된다.

3. 관련연구

맵리듀스로 grouping sets 질의를 계산하는 가장 단순한 방법은 다음과 같다[5]. 입력 데이터를 저장하고 있는 테이블을 F라고 하자. 맵 함수는 F의 각 튜플 t를 입력으로 받아 grouping sets 질의에 포함된 각 group-by에 대해 각각 하나씩의 키-값 쌍 (key, value)를 출력으로 내보낸다. 이 때 key는 대응하는 group-by의 그룹화 애트리뷰트의 값이며, value는 집계화 함수의 대상이 되는 측정(measure) 애트리뷰트의 값이다. 예를 들어 2.1절에서 예로 든 grouping sets 질의의 경우, 맵 함수는 F의 한 튜플 (1, 2, 3, 4)을 입력으로 받아 ab에 대해 ((1, 2, *), 4), bc에 대해 ((*, 2, 3), 4) 총 2개의 키-값 쌍을 출력으로 내보낸다. 여기서 *는 해당 위치에 대응하는 애트리뷰트 값이 사용되지 않음을 나타낸다. 리듀스 함수는 동일한 그룹화 애트리뷰트 값을 가지는 튜플들의 측정 애트리뷰트 값들을 입력으로 받아 그들로부터 지정된 집계 함수(SUM, AVG 등) 값을 계산하고 그 결과를 출력으로 내보낸다. 따라서 grouping sets에 포함된 group-by의 개수가 N이고 F의 튜플 수를 $|F|$라 할 때, 맵 함수는 총 $N \cdot |F|$개의 키-값 쌍을 출력으로 내보내며, 리듀스 함수는 이들을 사용하여 N개의 group-by를 모두 계산하게 된다.

이 방법은 단 1개의 맵리듀스 잡으로 grouping sets 질의에 포함된 모든 group-by를 계산할 수 있다. 하지만 맵 함수가 총 $N \cdot |F|$개의 키-값 쌍을 출력해야 하므로 N과 $|F|$가 증가함에 따라 이들을 리듀스 함수로 전달하는 비용이 매우 커진다는 단점이 있다.



(그림 1) Grouping sets 질의 처리 과정 [4]

한편 맵리듀스 환경이 아닌 전통적인 관계형 데이터베이스 환경에서 grouping sets 질의를 효율적으로 계산하기 위한 방법이 제안되었다[4]. 예를 들어 테이블 F에 대해 4개의 group-by a, b, c, ac를 구하는 grouping sets 질의가 주어졌다고 하자. 이를 구하기 위한 가장 단순한 방법은 (그림 1)의 왼쪽 그림과 같이 F로부터 a, b, c, ac를 각각 구한 후 그 결과를 합하는 것이다. 반면에 (그림 1)의 오른쪽 그림은 F로부터 ab와 ac를 먼저 구하고, ab로부터 a와 b를 구하고, ac로부터 c를 구하는 방식을 나타낸다. 만약 F에 비해 ab와 ac의 크기가 매우 작을 경우, (그림 1)의 오른쪽 그림은 왼쪽 그림에 비해 더 좋은 성능을 보일 수 있다. 제안 방법은 [4]와 유사한 전략을 사용하지만, [4]는 맵리듀스 환경이 아닌 단일 노드 환경을 고려하고 있다.

4. 제안방법

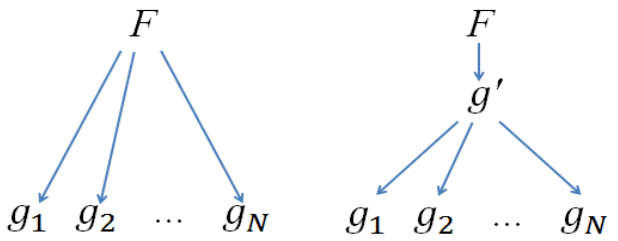
본 장에서는 제안 방법에 대해 설명한다. 제안 방법은

grouping sets 질의를 2개의 맵리듀스 잡(job)을 통해 단계적으로 계산한다. 입력 테이블을 F라 하자. 첫 번째 맵리듀스 잡은 F로부터 grouping sets 질의에 포함된 group-by들을 모두 계산할 수 있는 '부모' group-by를 먼저 계산한다. 부모 group-by는 다음과 같이 정의된다. Grouping sets 질의에 포함된 group-by들의 모든 그룹화 애트리뷰트의 집합을 $\{d_1, d_2, ..., d_m\}$이라 하자. 이 때 grouping sets 질의에 포함된 group-by들의 부모 group-by는 $d_1d_2...d_m$이 된다. 예를 들어 grouping sets 질의에 포함된 group-by가 ab와 bc라고 하면, 이들의 부모 group-by는 abc가 된다. 따라서 첫 번째 맵리듀스 잡에서 맵 함수는 F의 각 튜플 t에 대해 하나의 키-값 쌍 (key, value)을 내보내며, 이 때 key는 부모 group-by의 그룹화 애트리뷰트들의 값이고, value는 측정 애트리뷰트의 값이다. 리듀스 함수는 부모 group-by의 그룹화 애트리뷰트 값들이 동일한 튜플들의 측정 애트리뷰트 값들을 입력으로 받아, 그들로부터 지정된 집계 함수의 값을 계산하고 그 결과를 출력으로 내보낸다.

두 번째 맵리듀스 잡은 첫 번째 맵리듀스 잡에서 계산한 부모 group-by를 입력으로 하여 grouping sets 질의에 포함된 group-by들을 각각 계산한다. 이 때는 F 대신 부모 group-by를 입력으로 사용한다는 것만 다를 뿐 3장에서 설명한 방법과 동일한 방법을 사용하면 된다.

제안 방법은 2개의 맵리듀스 잡을 수행해야 하지만, 부모 group-by의 크기가 원래 입력 데이터의 크기에 비해 매우 작은 경우, grouping sets 질의에 포함된 각 group-by들을 독립적으로 구하는 단순 방법보다 빠르게 모든 group-by들을 계산할 수 있다.

이제 단순 방법과 제안 방법의 수행 비용을 비교 분석한다. 주어진 grouping sets 질의에 포함된 N개의 group-by들을 $g_1, g_2, ..., g_N$이라 하자. 이들의 부모 group-by를 g'로 표시하면, 단순 방법과 제안 방법은 다음과 같이 나타낼 수 있다.



(그림 2) 단순 방법과 제안 방법의 비교

단순 방법의 수행 비용을 C_{naive}라 하고, 제안 방법의 수행 비용을 $C_{proposed}$라고 하면, 각각은 다음과 같이 예측된다.

$$C_{naive} = c_1 \cdot |F| + c_2 \cdot N \cdot |F|$$

$$C_{propsed} = c_1 \cdot (|F| + |g'|) + c_2 \cdot (|F| + N \cdot |g'|)$$

여기서 c_1은 디스크에서 튜플을 읽는 비용을 나타내는 비례상수이고, c_2는 맵 함수가 출력한 키-값 쌍들을 리듀스 함수로 전달하는 비용을 나타내는 비례상수이다. $|g'|$은 부모 group-by g'의 튜플 수를 나타낸다. 단순 방법은 F를 읽는데 $c_1 \cdot |F|$의 비용이 발생하고, 맵 함수가 출력한 키-값 쌍들을 리듀스 함수로 전달하는데 $c_2 \cdot N \cdot |F|$의 비용이 발생한다. 반면에 제안 방법은 부모 group-by를 부가적으로 계산하므로 두 번째 맵리듀스 잡에서 g'를 읽는 비용 $c_1 \cdot |g'|$이 추가적으로 발생한다. 하지만 두 번째 맵리듀스 잡에서 맵 함수가 출력하는 키-값 쌍의 개수는 $N \cdot |g'|$에 불과하므로 이를 리듀스 함수로 전동하는데 드는 비용이 $c_2 \cdot N \cdot |g'|$에 불과하다. 따라서 $|g'| << |F|$인 경우 제안 방법은 단순 방법보다 수행 비용이 더 적을 것으로 예측할 수 있다.

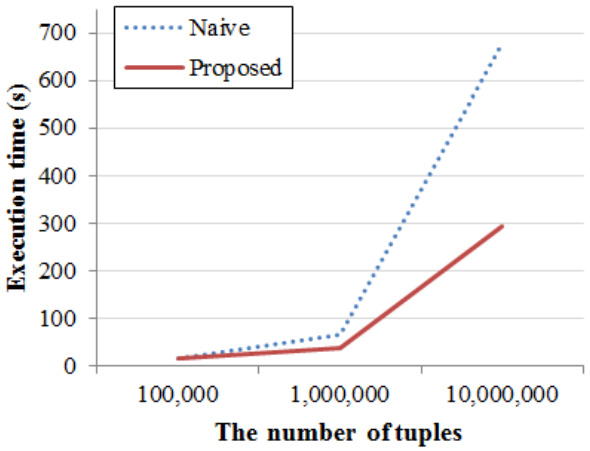
대부분의 경우 F에 비해 g'가 훨씬 작은 크기를 가지므로, 대부분의 데이터에서 제안 방법이 더 나은 성능을 보일 것으로 예상된다. 하지만 F에 비해 g'가 크게 작지 않은 경우 g'의 부가적인 계산으로 인해 큰 이득을 얻지 못할 수도 있다. 따라서 본 논문은 grouping sets 질의 계산 전에 우선 C_{naive}와 $C_{proposed}$를 계산하고, $C_{naive} > C_{proposed}$ 일 때는 제안 방법으로 질의를 처리하고 $C_{naive} < C_{proposed}$ 일 때는 단순 방법으로 질의를 처리하는 것을 제안한다. $|g'|$을 예측할 때는 group-by의 크기를 예측하는 기존의 다양한 방법을 사용할 수 있다[4].

5. 실험 결과

본 장에서는 제안 방법과 단순 방법의 성능을 비교한 실험 결과를 보인다. 각 방법의 성능은 동일한 grouping sets 질의를 처리하는데 걸리는 전체 수행 시간으로 측정하였다. 실험에는 Amazon EC2 서비스[6]를 사용하였고 클러스터는 4대의 가상 PC로 구성되어 있다. 각 PC는 Intel Xeon 프로세서 2.5 GHz CPU와 1 GiB 메모리 사양을 가진다. 실험에서는 맵 함수 수행에 1개의 프로세스를, 리듀스 함수 수행에 3개의 프로세스를 사용하였다.

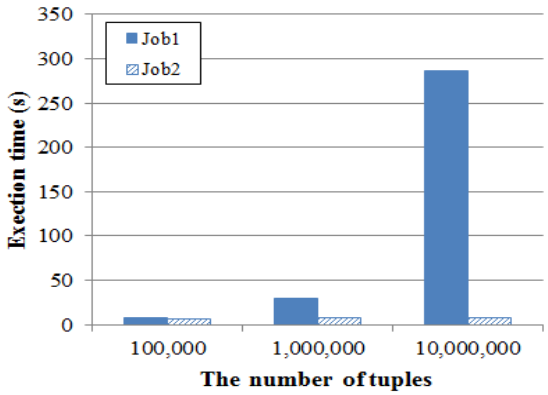
실험은 가상의 데이터를 생성하여 수행하였다. 입력 데이터를 나타내는 테이블 F는 3개의 차원 애트리뷰트 a, b, c와 1개의 측정 애트리뷰트 m으로 이루어져 있다. 각 애트리뷰트의 크기는 4 byte이며 [1, 50] 범위에서 임의로 선택한 자연수 값을 가진다. 실험은 F의 튜플 수를 10^5, 10^6, 10^7개로 증가시켜가며 수행하였다. 실험에 사용한 grouping sets 질의는 F에 대해 두 개의 group-by $\{ab, bc\}$를 구하는 질의이며 집계화 함수는 SUM(m)을 가정하였다.

단순 방법은 3장에서 설명한 바와 같이 F를 입력으로 하여 하나의 맵리듀스 잡으로 ab, bc를 모두 계산한다. 이에 비해 제안 방법은 첫 번째 맵리듀스 잡에서 F를 입력으로 하여 ab와 bc의 부모 group-by인 abc를 계산하고, 두 번째 맵리듀스 잡에서 abc를 입력으로 하여 ab와 bc를 계산한다.



(그림 3) 튜플 수에 따른 수행 시간 측정 결과

(그림 3)은 $|F|$가 10^5, 10^6, 10^7일 때 단순 방법(Naive)과 제안 방법(Proposed)으로 grouping sets 질의를 처리하는데 걸린 전체 수행시간을 나타내는 그래프이다. $|F| = 10^5$일 때는 두 방법이 비슷한 성능을 보였지만 $|F|$가 증가할수록 제안 방법이 더 좋은 성능을 보이는 것을 알 수 있다. 이것은 F의 a, b, c 애트리뷰트가 고정된 범위 내의 값을 가질 때, F의 튜플 수가 증가할수록 F에 중복된 값을 가진 튜플이 많아져 $|F|$에 비해 $|abc|$의 값이 크게 작아지기 때문이다. 따라서 맵 함수가 총 $2{\cdot}|F|$개의 키-값 쌍을 출력하는 단순 방법보다, 첫 번째 맵리듀스 잡에서는 맵 함수가 $|F|$개의 키-값 쌍을 출력하지만 두 번째 맵리듀스 잡에서는 맵 함수가 $2{\cdot}|abc|$개의 키-값 쌍만을 출력하는 제안 방법의 성능이 크게 좋아지게 된다.



(그림 4) 제안 방법에서 각 잡의 수행시간

(그림 4)는 제안 방법에서 $|F|$가 10^5, 10^6, 10^7일 때 첫 번째 맵리듀스 잡(Job1)와 두 번째 맵리듀스 잡(Job2) 각각의 수행 시간을 나타낸다. 앞서 설명한 바와 같이, 첫 번째 잡은 F를 입력으로 하므로 맵 함수가 출력으로 내보내는 키-값 쌍이 많아서 수행 비용이 크다. 그에 비해 두 번째 잡은 abc를 입력으로 하므로 맵 함수가 출력으로 내보내는 키-값 쌍이 크게 줄어들어 수행 비용이 매우 작음을 볼 수 있다.

4. 결론

본 논문은 맵리듀스 환경에서 grouping sets 질의를 효율적으로 계산하는 방법을 제안하였다. 제안 방법은 기존의 단순 방법에 비해 수행해야 하는 맵리듀스의 잡의 개수는 증가한다. 하지만 첫 번째 맵리듀스 잡에서 계산한 중간 결과를 두 번째 맵리듀스 잡의 입력으로 사용함으로써, 두 번째 맵리듀스 잡의 입력 크기를 크게 줄인다. 그에 따라 두 번째 맵리듀스 잡에서 맵 함수가 출력으로 내보내는 키-값 쌍의 수가 대폭 감소되어 총 수행비용이 크게 감소된다. 가상 데이터를 사용한 시험에서 제안 방법은 기존 방법에 비해 총 수행 시간을 약 1/2로 감소시킴을 확인하였다.

Acknowledgement

본 연구는 미래창조과학부 및 정보통신산업진흥원의 ICT/SW 창의연구과정의 연구결과로 수행되었음 (NIPA-2014-H0502-14-3007)

참고문헌

[1] Big Data, http://en.wikipedia.org/wiki/Big_Data

[2] Mark Beyer, "Gartner Says Solving 'Big Data' Challenge Involves More Than Just Managing Volumes of Data," Gartner, June 27, 2011.

[3] Jefferey Dean, Sanjay Chemawat, "MapReduce: simplified data processing on large clusters," In Proceedings of OSDI '04,pp.137-150, 2004.

[4] Zhimin Chen, Vivek Narasayya. "Efficient Computation of Multiple Group By Queries", 2005

[5] Jie Pan, Frederic Magoules, Yann Le Biannic, "Executing Multiple Group by Query Using MapReduce Approach: Implementation and Optimization", 2010

[6] Amazon Web Services, http://aws.amazon.com/ec2/