대용량 데이터를 위한 맵리듀스 기반 데이터 큐브 생성 기법

Data Cube Materialization using MapReduce for Large-scale Data

양해미(Haemi Yang)*, 이기용(Ki Yong Lee)**, 정연돈(Yon Dohn Chung)***
haemiyang@korea.ac.kr, kiyounglee@sookmyung.ac.kr, ydchung@korea.ac.kr

초 록

최근, 대용량 데이터 분석에 대한 관심이 높아지고 있다. 대용량 데이터 분석은 다양한 집계 질의가 많이 사용되는데, 이를 효율적으로 처리하기 위해 데이터 큐브가 사용된다. 본 논문은 맵리듀스에서 대용량 데이터에 대한 새로운 큐브 생성 기법을 제안한다. 기존 큐브 연구는 단일 머신 또는 적은 수의 노드로 구성된 클러스터에 최적화 되어 있어, 확장성에 문제가 있다. 또한 맵리듀스에서 고려해야 하는 네트워크 비용 문제를 고려하지 않았다. 따라서, 기존 큐브 연구는 맵리듀스에서 대용량 데이터에 대해 큐브를 생성하기에 적합하지 않다. 본 논문은 맵리듀스에서 대용량 데이터에 대해 큐브를 생성할 때, 중간 결과 크기를 줄임으로써 네트워크 비용을 줄이는 새로운 큐브 생성 기법을 제안한다. 실험을 통해 제안 기법이 데이터와 차원 수가 증가함에 따라 기존 기법보다 성능이 뛰어남을 보인다.

1. 서론

대용량 데이터 분석은 다양한 집계 질의가 많이 사용되는데, 이를 효율적으로 처리하기 위해 데이터 큐브(Data Cube)[1]가 사용된다.

데이터 큐브는 분석하고자 하는 대상인 차원 속성(Dimension)들의 모든 조합에 대해 측정하고자 하는 대상인 측정 속성(Measure)들의 집계 결과를 저장한다. 이것은 많은 연산 비용이 요구되기 때문에, 효율적으로 큐브를 생성하기 위한 많은 연구들[2,3,4,5,6,7]이 있었다. 그러나, 기존 큐브 연구들은 단일 머신 또는 적은 수의 노드에 최적화 되어 있어, 대용량 데이터를 다루기 어렵다. 따라서, 맵리듀스(MapReduce)[8]와 같은 분산 시스템을 이용해 대용량 데이터에 대한 큐브를 생성하는 새로운 방법이 필요하다.

맵리듀스는 중간 결과를 키 별로 모을 때

본 연구는 중소기업청에서 지원하는 2012년도 산학연공동기술개발사업(No.C0012822)의 연구수행으로 인한 결과물임을 밝힙니다.
* 고려대학교 정보통신대학 컴퓨터학과 석사과정
** 숙명여자대학교 컴퓨터과학과 교수
*** 고려대학교 정보통신대학 컴퓨터학과 교수

노드 간 데이터 이동때문에 네트워크 비용이 발생하는데, 이 네트워크 비용이 매우 크기 때문에 이를 줄이기 위한 방법이 매우 중요하다. 맵리듀스에서 대용량 데이터에 대해 데이터 큐브를 생성하는 가장 직관적인 2-단계 큐브 생성 방법은 먼저, 맵 단계에서 맵퍼(Mapper)들이 데이터를 스캔하고, 이를 대상 속성들의 모든 조합에 대해 그룹바이 질의를 수행한다. 맵(Map)단계에서 처리한 중간 결과는 같은 그룹바이 키를 가진 데이터끼리 모여 리듀서(Reducer)에 전송된다. 마지막으로 리듀스(Reduce)단계에서 리듀서는 전송 받은 데이터를 다시 그룹바이 질의를 수행해 최종 결과를 출력한다. 이 방법은 중간 결과 크기가 매우 커서 네트워크 비용을 증가시킨다. 따라서, 맵리듀스에서 대용량 데이터에 대한 큐브 생성 시, 네트워크 비용을 고려한 새로운 방법이 필요하다.

본 논문의 구성은 2장은 관련 연구에 대해 알아보고, 3장은 대용량 데이터에 대한 큐브 생성 시, 네트워크 비용을 고려한 새로운 기법을 제안한다. 4장은 제안 기법의 성능을 평가하고, 효과적임을 보인다. 마지막으로 5장은 결론을 맺는다.

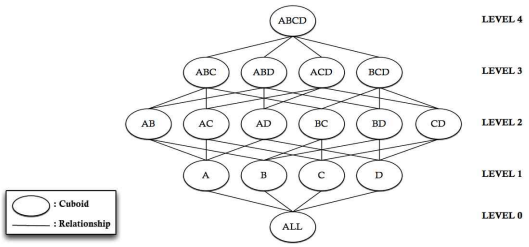
2. 관련연구

2.1 데이터 큐브 알고리즘

데이터 분석은 모든 조합에 대해 집계 결과를 유지하는 것이 중요하기 때문에, 전체 큐보이드를 효율적으로 생성하는 방법[2,3,4]이 제안되었다. 그러나, 데이터가 증가하면서 단일 머신에서 전체 큐보이드를 생성하는 것이 디스크 및 메모리 가용성의 한계로 어려워졌다. 따라서, 큐브의 일부만을 생성하는 방법[5]이 제안되었다. 이 후, 적은 수의 노드로 구성된 클러스터 환경에서 전체 큐보이드를 생성하는 방법[7]과 일부만을 생성하는 방법[6]이 제안되었다. 그러나 이것들은 확장성에 문제가 있어, 대용량 데이터를 다룰 수 있는 다수의 노드로 구성된 분산 환경에 적합하지 않다. 최근 분산 환경에서 대용량 데이터에 대해 큐브를 생성하기 위한 연구[10]가 시도되고 있는데, 이것은 전체 큐보이드 생성을 다루지 못하는 한계가 있다. 본 논문은 데이터 분석에서 중요한 전체 큐보이드를 생성하는 것에 초점을 맞춘다.

3. 제안기법

제안 기법은 큐보이드 간의 종속 관계, 큐보이드 그룹, 큐보이드 그룹 간의 종속 관계를 정의하고, 이를 통해 새로운 큐브 생성 알고리즘을 제안한다.



<그림 1> 4개의 차원 속성으로 구성된 큐브 격자의 예

정의 1: [큐보이드 간의 종속 관계]

주어진 두 큐보이드에 대해 한 큐보이드 C의 모든 차원 속성이 다른 큐보이드 C'의 모든 차원 속성이 포함되면, C'는 C를 *종속한다*.

<그림 1>과 같은 큐브 격자에서 간선으로 연결된 두 큐보이드에 대해, 차원 속성의 수가 더 많은 큐보이드를 *부모 큐보이드*, 더 적은 큐보이드를 *자식 큐보이드*라고 할

때, 두 큐보이드는 종속 관계이므로, 부모 큐보이드로부터 자식 큐보이드를 생성할 수 있다. 이 때, 큐보이드들의 부모 자식 관계에 따라 여러 큐보이드가 *연쇄적으로 생성 될 수 있다.*

정의 2: [큐보이드 그룹]

큐보이드 그룹은 전체 큐보이드 중에서 한 큐보이드로부터 연쇄적으로 생성될 수 있는 큐보이드들의 부분 집합이다. 이 때, 큐보이드 그룹 내에서 차원 속성의 수가 가장 많은 큐보이드를 *헤드 큐보이드*라고 한다.

큐보이드 그룹의 큐보이드들은 종속 관계를 가지는 큐보이드들의 부분 집합이므로, 헤드 큐보이드로부터 그룹 내의 모든 큐보이드들을 *연쇄적으로 생성 할 수 있다.*

정의 3: [큐보이드 그룹 간의 종속 관계]

한 큐보이드 그룹 G의 헤드 큐보이드가 다른 큐보이드 그룹 G'의 헤드 큐보이드를 종속할 때, G는 G'를 종속한다. 이 때 G를 *부모 큐보이드 그룹*, G'을 *자식 큐보이드 그룹*이라고 부른다.

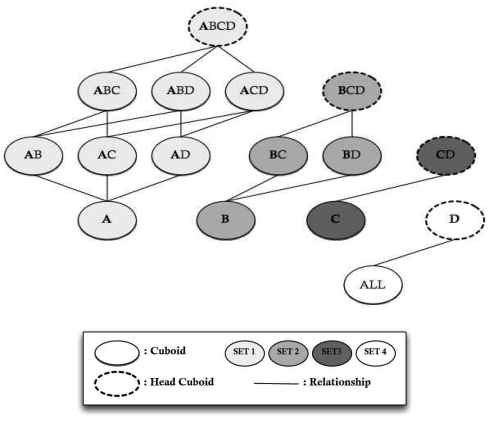
자식 큐보이드 그룹의 헤드 큐보이드는 부모 큐보이드 그룹의 헤드 큐보이드로부터 생성할 수 있다.

3.1 큐보이드 그룹 생성 알고리즘

<그림 1>과 같은 큐브 격자에서 어떤 큐보이드 그룹에도 포함되지 않은 큐보이드들 중 가장 상위레벨에 있는 큐보이드를 헤드 큐보이드로 선택한다. 그리고 나서, 헤드 큐보이드로부터 연쇄적으로 생성될 수 있는 큐보이드들 중, 헤드 큐보이드의 차원 속성 중 첫 속성과 같은 큐보이드들을 포함하는 새로운 큐보이드 그룹을 생성한다. 차원 속성의 수가 n이라면, n-1번 반복하여 그룹들을 생성한다. 만약, 어떤 큐보이드 그룹에도 포함되지 않은 큐보이드가 있다면, 마지막으로 이들을 포함하는 새로운 큐보이드 그룹을 생성하고, 모든 큐보이드가 큐보이드 그룹에 포함 되었을 때, 큐보이드 그룹 생성 알고리즘을 마친다. <그림 2>는 구성된 큐보이드 그룹의 예를 보여준다.

3.2 맵리듀스를 이용한 큐브 생성 알고리즘

먼저, 큐보이드 그룹 생성 알고리즘으로부터 큐보이드 그룹들을 생성한다. 그리고 나서, 맵 단계에서 맵퍼들은 데이터를 스캔하고, 이를 큐보이드 그룹의 헤드 큐보이드의 차원 속성에 대해 그룹바이를 수행한다. 파티션 키는 큐보이드 그룹의 모든 큐보이드들이 공통으로 가진 차원 속성으로 설정한다. 리듀스 단계에서 리듀서들은 헤드 큐보이드가 속한 큐보이드 그룹 내의 모든 큐보이드들을 연쇄적으로 생성한다. 이와 같은 과정을 모든 큐보이드 그룹에 대해 반복한다.



<그림 2> 4개의 차원 속성으로 구성된 큐브 격자에서 큐보이드 그룹의 예

제안 기법은 큐보이드 간의 종속 관계를 활용한 큐보이드 그룹 생성 알고리즘을 통해, 큐보이드 그룹을 생성하고, 큐보이드

그룹 간의 종속 관계를 활용한 맵리듀스를 이용한 큐브 생성 알고리즘을 통해, 큐보이드 그룹에 대해 큐브를 생성함으로써, 중간 결과 크기를 줄일 수 있다. 따라서, 맵리듀스에서 전체 질의 시간에 큰 영향을 미치는 네트워크 비용을 줄일 수 있다.

4. 성능평가

4.1 실험 환경

우리는 제안 기법을 실제 맵리듀스 프레임워크에 구현했다. 실험에 사용한 하둡 클러스터는 i5-2500 3.3Ghz CPU와 16GB의 메모리를 가진 동일한 사양의 컴퓨터 32대로 구성되었다. 실험에 사용한 데이터는 실제 웹 로그 데이터로 크기는 약 75GB, 튜플 수는 400M이다. 전체 속성 중 6개의 차원 속성과 2개의 측정 속성을 사용했다. 제안 기법과 비교할 대상은 2-단계 큐브 생성 방법과 기존 방법 중 전체 큐보이드를 생성하는 하향식 접근 방식의 방법[7]을 맵리듀스 환경에 적용한 것이다.

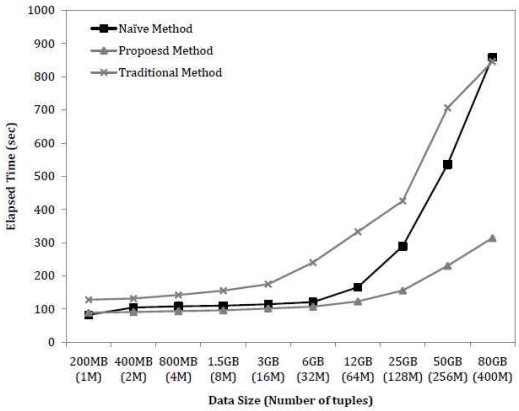
4.2 실험 결과

4.2.1. 데이터 크기 증가에 따른 수행 시간

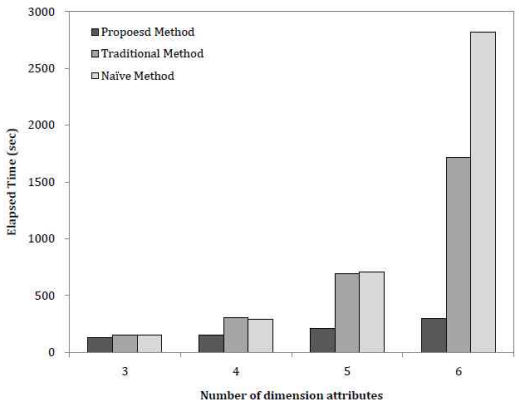
<그림 3>은 데이터 크기 증가에 따른 수행 시간을 나타낸다. 비교한 방법들은 확장성을 고려한 방법이 아니기 때문에, 데이터가 증가할수록 전체 수행 시간에 큰 영향을 받는다. 반면에, 제안 기법은 네트워크 비용에 영향을 미치는 중간 결과 크기를 줄임으로써 전체 수행 시간을 줄였다.

4.2.2. 차원 수 증가에 따른 수행 시간

<그림 4>은 차원 속성 수 증가에 따른 수행 시간을 나타낸다. 비교한 방법들은 차원 속성 수가 증가할수록 중간 결과 크기가 기하급수적으로 증가하기 때문에, 전체 수행 시간에 큰 영향을 받는다. 반면에, 제안 기법은 차원의 수가 증가하더라도 중간 결과 크기가 기하급수적으로 증가하지 않기 때문에 전체 수행 시간을 줄였다.



<그림 3> 데이터 크기 증가에 따른 수행 시간



<그림 4> 차원 수 증가에 따른 수행 시간

5. 결론

본 논문은 맵리듀스에서 대용량 데이터에 대해 데이터 큐브를 생성하기 위한 새로운 큐브 알고리즘을 제안했다. 제안 기법은 큐보이드 간의 종속 관계를 이용해 큐보이드 그룹 생성 알고리즘을 제안하고, 큐보이드 그룹 간의 종속 관계를 이용해 맵리듀스에서 큐브 생성 알고리즘을 제안하여, 중간

결과의 크기를 줄였다. 실험을 통해, 제안 기법은 데이터와 차원 수가 증가하더라도 분산 환경에서 중요한 문제가 되는 네트워크 비용을 줄여, 전체 수행 시간을 줄이는 것을 보였다.

참고문헌

[1] J. Gray, A. Bosworth, A. Layman, and H. Pirahesh. Data Cube: A Relational Operator Generalizing Group-By, Cross-Tab and Sub-Totals. ICDE, 1996.

[2] S. Agarwal, R. Agrawal, P. M. Deshpande, A. Gupta, J. F. Naughton, R. Ramakrishnan, and S. Sarawagi. On the Computation of Multidimensional Aggregates. VLDB, 1996.

[3] Prasad M. Deshpande, Sameet Agarwal, Jeffrey F. Naughton and Raghu Ramakrishnan. Computation of Multidimensional Aggregates. Technical Report-1314, University of Wisconsin-Madison, 1996.

[4] S. Sarawagi, R. Agrawal, and A. Gupta. On Computing the Data Cube. Technical report, IBM Almaden Research Center, 1996.

[5] K. Beyer and R. Ramakrishnan. Bottom-Up Computation of Sparse and Iceberg CUBEs. SIGMOD, 1999.

[6] R. T. Ng, A. S. Wagner, and Y. Yin. Iceberg-cube Computation with PC Clusters. SIGMOD, 2001.

[7] Chen. Y, Dehne. F, Eavis. T, Rau-Chaplin. A. Building large ROLAP data cubes in parallel. IDEAS, 2004.

[8] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. OSDI, 2004.

[9] Apache hadoop. http://hadoop.apache.org.

[10] A. Nandi, C. Yu, P. Bohannon, and R. Ramakrishnan. Distributed Cube Materialization on Holistic Measures. ICDE, 2011.