Data Cube Materialization with MR Cube and CM Sketch Approach

Amar Sawant[1], Madhav Ingle[2]

[1]PG Student of Computer Engineering, Jayawantrao Sawant College of Engineering, Hadapsar, Pune, India

[2]PG Coordinator, Department of Computer Engineering, Jayawantrao Sawant College of Engineering, Hadapsar, Pune, India

Abstract: ***Data cube computations plays an important role in data warehouse systems. Applications with multidimensional data analysis are looking for unusual patterns. Here aggregation of data is done across many dimensions. Aggregation is done by making use of SQL aggregate functions and Group by operators. As there is need for multidimensional generalization of these operators, data cube is used which is a way for structuring data in multidimensions so that analysis can be done on some measures of interest. One of the key tasks in data warehouse is data cube computations. Several techniques for data cube computations are available but there are some limitations so MapReduce based approach can be used to overcome the limitations. MR-Cube, which is Mapreduced based approach creates lattices using derived data set which are further partitioned using value partitioning techniques followed by batch areas creation, makes an effective distribution of data and computation workload. Data cube computations in parallel using partially algebraic measures is done using MapReduced based algorithm. Extreme data skew is detected for a few cube groups that are unusually large. CM-Sketch is a Count Min Sketch approach, which is a compressed counting data structures used as a solution for extreme data skews.***

Keywords: cube analysis, holistic measures, map reduce, data skew, CM sketch

1. Introduction

One of the powerful tools for analyzing multidimensional data is data cube analysis. With Cube analysis users are provided with a convenient way to discover insights from the data by computing aggregate measures. The techniques are designed for a single machine or clusters with small number of nodes. For the rate at which the data are being accumulated (e.g., terabytes per day) at many companies, it is increasingly difficult to process data with a single (or a few) machine(s). Efficient computation of all the aggregates is the basic cube problem. Computing the aggregates concurrently gives an opportunity to share partitioning and aggregation costs between various group-bys.

Data cube computations plays an important key role in data warehouse systems. The aggregations are mentioned as GROUP- BYS. Online Analytical Processing (OLAP) or multidimensional data analysis applications typically collect data across many dimensions looking for unusual patterns. A data cube is a way of organizing data in N-dimensions in OLAP systems, so as to perform analysis over some measure of interest. Measure is a term which refer numerical facts that can be algebraic (SUM, COUNT etc.) or non algebraic (DISTINCT, TOP-K etc).

2. Literature Survey

Computing interesting measure for data cubes and subsequent mining of interesting cube groups over massive data sets is found critical for many important analysis done in the real world. The algebraic measures such as SUM are open to parallel computation while dealing with holistic (i.e non algebraic) measures is nontrivial. The paper [2] focuses on cube materialization for holistic measures using the MapReduce paradigm. In addition to describing real world challenges associated with holistic cube computation using MapReduce for web scale data sets this paper introduces partially algebraic measures which is an important subset of holistic measures that are MapReduce friendly and two techniques, value partitioning and batch area identification that effectively influence the Mapreduce framework to distribute the data and computation workload. Also a three phase cube computation algorithm MR-Cube is proposed, that employs these techniques to successfully cube billion-tuple sized data sets and optionally surfaces interesting cube groups. A reducer fails due to wrong estimation of group size,in that case all the data for that group is written back to the disk and MapReduce jobs are then run with more aggressive value partitioning, until the cube is completed [4]. Issue of extreme data skew is detected, which occurs if a few cube groups are unusually large. This causes value partitioning to be applied to entire cube and reduces the efficiency of algorithm.

Efficient cube computation is important in data cube technology. Many techniques are used but they have few limitations. Proposed approach effectively distributes data and computation workload. Using key subset of holistic measures cube is computed in parallel and interesting cube groups are identified over Mapreduce [1]. Extreme data skew problem is detected before cube materialization for which compressed counting data structure such as CM-Sketch is suggested as a solution.

One of the most popular forms of the sketch data structure Count-Min Sketch is introduced [3]. The Count-Min Sketch provides a different kind of solution to count tracking. It allocates a fixed amount of space to store count information, which does not vary overtime even as more and more counts are updated. The data structure is parameterized by the constants w and d which determine the time and space needs and the probability of error of the queries. The algorithm needs a two dimensional array with w columns and d rows. A key feature of the operations which manipulate the sketch is that they are largely oblivious to the current state of the data structure. The data structure is highly suitable for parallelization and distributed computation. First, each row of the sketch is updated independently of the others, so the sketch can be partitioned row-wise among threads on a single machine. But one can build sketches of different subsets of the data and these sketches can be combined to give the sketch of the union of the data. Sketch combination is straightforward where for a given sketch arrays of size w x d, they are combined by summing them up, entry-wise.

3. Problem Definition

Data cube analysis is a powerful tool for analyzing multidimensional data. There are two main limitations in the existing techniques that have so far prevented cube analysis being extended to an even broader usage such as analyzing web query logs. Firstly, they are designed for a single machine or clusters with small number of nodes. For the rate at which data are being accumulated (e.g., terabytes per day) at many companies, it is increasingly difficult to process data with a single (or a few) machine(s). Secondly, many established techniques take advantage of the measure being algebraic and make use this property to avoid processing groups with a large number of tuples. A measure is algebraic if the measure of a supergroup can be easily computed from its subgroups. (e.g., SUM(a + b) = SUM(a) + SUM(b)). This allows parallelized aggregation of data subsets whose results are then post processed to derive the final result. Third, to provide a solution to extreme data skews for unusually large cube groups.

4. Proposed System

The goal is to divide the computation into pieces such that no reducer has to deal with extremely large data groups and the overall intermediate data size is controlled. For groups with a large number of tuples, the memory requirement for maintaining such intermediate data can become irresistible. So the input data is distributed across the mapper machines, where each machine then processes a subset of the data in parallel and produces one or more key, value pairs for each data record. The objective is to describe real world challenges associated with holistic cube computation using MapReduce.

The proposed system has the following phases:

Phase I:Cube Lattice Formation
This will consists of first collecting the data and preparing the derived dataset. For identifying dimension attributes and cube lattice it is necessary to prepare dimension attributes. The term dimension attributes refers to the set of attributes that the user wants to analyze. Based on those attributes, a cube lattice can be formed representing all possible grouping(s) of the attributes. Each node in the lattice represents one possible grouping/ aggregation of dataset attributes. Each cube group in turn contains a set of tuples satisfying the same aggregation value. We use the term cube region to denote a node in the lattice and the term cube group to denote an actual group belonging to the cube region. Number of tuples in each cube group is identified using naïve algorithm. Two techniques such as value partitioning and batch area identification are used that influence the MapReduce framework to distribute data and computation workload. MR-Cube algorithm is proposed that employs these techniques.

Phase II: Cube Materialization and Mining
This phase will deal with use of algorithm for cube computation using holistic measures. Use technique to detect and avoid extreme data skew while doing cube computation. For data skew, CM-Sketch algorithm which is a count min sketch approach for use of compressed counting data structure is proposed. Aggregate cube groups to get final measures and identify interesting cube groups.

Analysis of the result will be done by evaluation of performance of proposed approach for computing cube and mining interesting patterns thus enhancing the approach to improve the result.

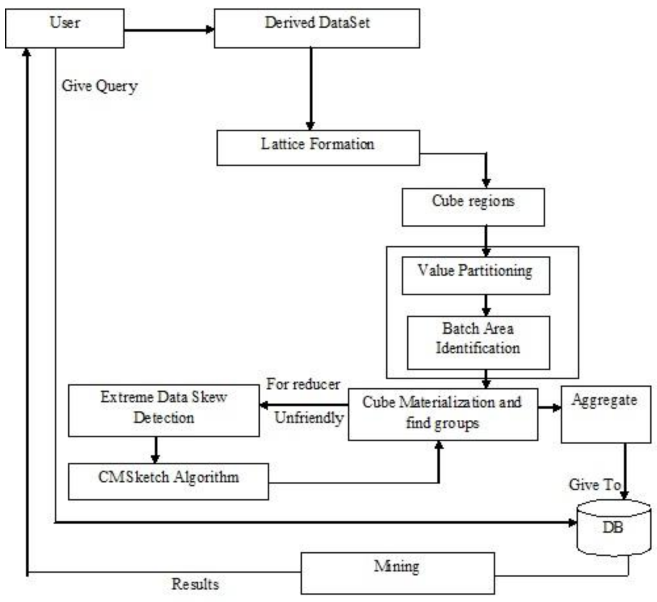

Figure 1: System Architecture

5. Problem Formulation

Consider D as dataset, which is the sample data from which there is need to map the data attributes to form derived datasets. C is the cube lattices which are created by making use of D such that the cube region R is formed.

$$R \leftarrow \{D, C\} \quad (1)$$

Perform value partitioning and batching which annotate map reduce estimate, where e is tuple in data. Let |D| denote the total number of tuples in the data, c denote the reducer limit i.e. the maximum number of tuples a reducer can handle,

$$r = c / |D| \quad (2)$$

Let N denote the sample size. If a cube group G contains less than 0.75rN tuples in the sample, then the probability of G being a reducer unfriendly group.

For the set of reducer friendly region C in cube lattice and ($R_i \prec R_j$) indicating parent child relationship where R_j being the parent between two regions, assign each R ∈ C′ into one of the set of batch areas ($B_1, B_2, \ldots, B_k$) such that following three constraints should be satisfied:

A region with at least one parent that is also reducer friendly must belong to a batch area that contains at least one of its parents indicating the parent-child relationship between two regions in the whole cube lattice.

$$\forall R_1 \in C', \quad R \in B_i \Rightarrow \exists R'', R \prec R'', R'' \in B_i \tag{3}$$

No two regions whose parents are reducer unfriendly can belong to the same batch area.

$$\forall R_1, R_2 \in C', \quad R_1 \prec R_1', R_2 \prec R_2', R_1', R_2' \notin C', \quad R_1 \in B_i \Rightarrow R_2 \in B_j, i \neq j \tag{4}$$

The difference in the number of regions of two batch areas cannot be more than 2, a heuristic used to balance the workload of each batch area.

$$\forall ij, i \neq j, \ | (|B_i| - |B_j|) | \leq 2 \tag{5}$$

For smaller lattices, it is feasible to pick the solution with the lowest total cost

$$min(\sum icost(B_i)) \tag{6}$$

by exhaustive enumeration. The cost function for reflects the amount of intermediate data produced per batch area and is identified as the count of set of attributes required by that batch area.

For each of the tuple in the data set, MR cube map emits key: value pairs for each batch area. The shuffle phase then sorts them by key, yielding four reducer tasks. If a reducer fails due to wrong estimation of group size, all the data for that group is written back to the disk and follow-up MapReduce jobs are then run with more aggressive value partitioning, until the cube is completed.

In core materialization step all value partitioned groups need to be aggregated to compute the final measure. Such that g is group label, p is the partition id, m is the measure value.

$$g \rightarrow m \tag{7}$$

6. Algorithm

Algorithm 1: Naïve Algorithm

```
1. Let C be the cube Lattice
2. Let R be the cube region
3. For each of the tuple e in the R
4. do k=R(e).
5. Emit k→ e
```

Algorithm 2: Cube Region and Value Partitioning

```
1. Let D be the derived dataset, N as sample size
2. Perform all possible groupings of attributes of D to
   represent C.
3. Calculate number of tuples using Naïve algorithm.
4. Calculate r=c / |D|, where r is maximum number of tuples a
   single reducer can handle c as overall data size (|D|)
   percentage.
5. Check whether number of tuples > 0.75 rN, where N is
   sample size.
6. If yes, then the cube region is reducer unfriendly. Place the
   cube region in reducer unfriendly list and perform value
   partitioning on the cube region. Repeat value partitioning
   until the cube region is reducer friendly.
7. If no, then place the cube region in reducer friendly list.
```

Algorithm 3: Batch Area Identification

```
   For each of the reducer friendly list
1. Check if region with parent P is reducer friendly belong to
   batch area if it contains at least one of its parents.
2. Else check if no two regions whose parents are reducer
   unfriendly belong to same batch area.
3. Else check difference in number of regions of two batch
   areas is not more than 2.

For each of the reducer unfriendly list
1. Combine regions based on the partition factor.
2. Add regions with same partition factor in same batch.
```

Algorithm 4: Materialization and Aggregation

```
1. Let Ca be the Cube lattice
2. For each batch areas bi in Ca.batch_areas
3. Calculate Count Min for bi
4. Apply measure function M.
5. Aggregate data based on group label and partition id
6. Merge data based on group label.
```

Algorithm 5: Mining

```
1. For each group g and dimension d in cube region
2. Find secondary key.
3. Calculate $\sum m_j$
4. $g_{best}$=null, $m_{best}$=null
5. For each group g and measure m in each group
6. if($\sum m_j > m_{best}$)
7. then $g_{best}=g_i$
8. $m_{best}=\sum m_j$
9. Emit (p, g, $g_{best}$, $m_{best}$)
```

Algorithm 6: Count Min

```
1. Initialize array C of w x d counters to zero,
   C[1,1]C[d,w] ← 0
2. For j←1 to d do
3. Pick values $a_j$, $b_j$ for hash function based on prime p
4. N=0
5. Update total count N with c
6. Hash i to its counter in each row and update counter.
7. For j← 1 to d do
8. $h_j(i)=(a_j \times i + b_j \bmod p) \bmod w$
9. C[j, $h_j$ (i)]← C[j, $h_j$ (i)] + c
      10. e←∞
```

```
11. Perform hashing and keep track of the smallest value of C[j, $h_j$ (i)] over the d values of j
12. For j← 1 to d do
13.     $h_j(i)=(a_j \times i + b_j \bmod p) \bmod w$
14.     e ← min(e,C[j, $h_j$ (i)])
15. Emit e
```

7. Experimental Setup and Result

The dataset used is example dataset. The following information is retained in the dataset: id, date, uid, country, state, city, topic, category and subcategory. Data included under topic is shopping while under category is computer, camera, clothing and mobile. With respect to category, subcategory data is added.

First data was collected and derived dataset was prepared after raw attributes have been mapped as shown in table 1. The derived dataset file created was read. Dimension attributes were prepared. The term dimension attributes refers to the set of attributes that the user wants to analyze. Based on those attributes such as country, state, city, topic, category and subcategory, a cube lattice was formed representing all possible groupings of the attributes as shown in fig. 2. Each node in the lattice represents one possible grouping/aggregation. Each group in turn contains a set of tuples satisfying the same aggregation value. Accordingly cube regions were created. Navie algorithm was used which gives the count number of tuples of the groupby attributes specified by the cube region.

Table 1: Dataset

Uid	*c'ntry*	*state*	*city*	*topic*	*category*	*subcat*
u1	India	Maha'	Pune	Shop'	Clothing	Shirt
u2	India	Karna'	B'glr	Shop'	Camera	Nikon
u1	India	Maha'	Kop	Shop'	Mobile	Nokia

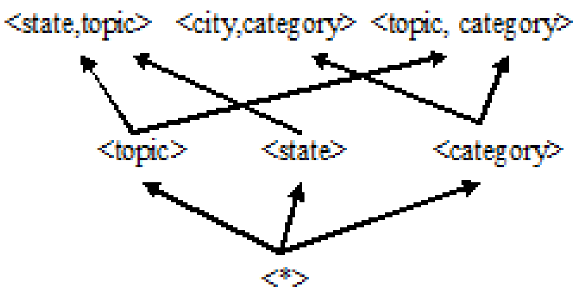

Figure 2: Cube Lattice

The next algorithm for value partitioning and batch area formation will be used for identifying and listing the reducer friendly and reducer unfriendly regions followed by materialization and aggregating the data based on group label and partition id. The result in the figure 3 shows the time required for materialization and mining as the number of tuples increases. For reducer unfriendly group where data skew if detected, CM Sketch algorithm as proposed was used. Finally mining of interesting cube groups will be done based on some measures of interest. The time complexity that impact the performance of algorithm is as shown in table 2.

The results for MR Cube and proposed CM Sketch algorithm is shown graphically in figure 3.

Table 2: Performance Statistics (Time in milliseconds)

Dataset (in Lakhs)	*MR Cube*	*CM Sketch*
1	257	158
2	434	230
3	649	454
4	879	689
5	1227	724

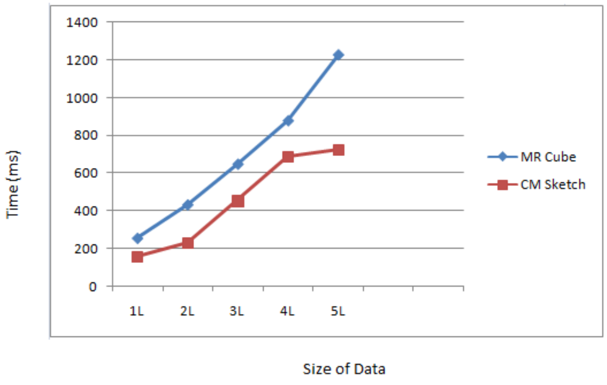

Figure 3: Results for MR Cube and CM sketch

8. Future Work

Currently in the system we have decided the partial algebraic measure which is also holistic measure. System itself does not decide whether a given holistic measure is partial algebraic measure or not. Deciding holistic measure and then detecting its algebraic measure automatically can be done as future work.

9. Conclusion

Here we study cube materialization and subsequent mining of non algebraic measures over extremely large data such as search logs using the Map Reduce framework. A subset of holistic measures that are partially algebraic is identified and technique of value partitioning is proposed to make them easy to compute in parallel. The designed algorithms partition the cube lattice into batch areas to effectively exploit both the parallel processing power of Map Reduce and the pruning power of cube materialization algorithms. Further data skew detected for unusually large cube group causes value partitioning to be applied to the entire cube thereby reducing efficiency of the algorithm. To avoid this proposed CM Sketch algorithm is used.

References

[1] Arnab Nandi, Cong Yu, Philip Bohannon, and Raghu Ramakrishnan,"Data Cube Materialization and Mining over MapReduce," IEEE transaction on Knowledge and Data Engineering, vol. 24, no. 10, Oct 2012.
[2] Dhanashri S. Lad, Sachin S. Patil, "Computing Interesting Measures and Mining for Data Cube, " Elsevier Publication, 2014
[3] Graham Cormode, S. Muthukrishnan, "Approximating Data with the Count-Min Data Structure," Aug 2011.

[4] A. Nandi, C. Yu, P. Bohannon, and R. Ramakrishnan, "Distributed Cube Materialization on Holistic Measures," Proc. IEEE 27th Intl Conf. Data Eng. (ICDE), 2011.

[5] K. Sergey and K. Yury, "Applying Map-Reduce Paradigm for Parallel Closed Cube Computation," Proc. First Intl Conf. Advances in Databases, Knowledge, and Data Applications (DBKDA), 2009.

[6] Zhengkui Wang, Yan Chu,Kian-Lee Tan, Divyakant Agrawal, Amr El Abbaddi, Xiaolong Xu "Scalable Data Cube Analysis Over Big Data," ,Nov 2013.

[7] G. Cormode and S. Muthukrishnan, "The CM Sketch and Its Applications," J. Algorithms, vol. 55, 2005.

[8] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F.Pellow, and H. Pirahesh, Data Cube: A Relational Operator Generalizing Group-By, Cross-Tab and Sub-Totals, Proc.12th Intl Conf. Data Eng. (ICDE), 1996

[9] S. Agarwal, R. Agrawal, P. Deshpande, A. Gupta, J. Naughton, R. Ramakrishnan,and S. Sarawagi, On the Computation of Multidimensional Aggregates, Proc.22nd Intl Conf. Very Large Data Bases (VLDB), 1996

.