MELIORATING MM CUBE TECHNIQUE

Riddhi Dhandha
IT System & Network Security
Gujarat Technological University
Ahmadabad, India
acharya_riddhi@yahoo.com

Nilesh Padariya
Tech Lead
Infebeam Pvt. Ltd
Ahmadabad, India
nileshsp2@gmail.com

Abstract— In this paper I have analyzed and improved MM Cube Technique to better time-efficiency. MM Cube that compute Iceberg Cubes by factorizing the lattice space according to the frequency of values. This approach, different from all the previous dimension-based approaches where the importance of data distribution is not recognized, partitions the cube lattice into one dense subspace and several sparse Subspaces. MM-Cubing is efficient and achieves high performance over all kinds of data distribution compare to previous cube techniques. Shared array is used as Data Structure in existing MM Cube. In this paper I have presented MM cube with tree as Data Structure to improve Time-Efficiency. Tree is used Approiry Pruning Strategy which cut down the node, which doesn't satisfy the Iceberg condition. So, it reduces traversal time, which minimizes cube computation time and such a way it tries to solve fundamental issue of efficient cube computation in Data Warehouse and OLAP System.

Keywords— Iceberg Cube, Tree, Approiry Pruning, Factorization, Major- Minor, Dense - Sparse, Improved Time –Efficiency.

I. INTRODUCTION

Data Cube in the contest of data warehousing and OLAP is core operator. The data cube was proposed to pre - Compute the aggregation for all possible combination of dimension to answer analytical queries efficiently. Data Cube computation and representation are expensive in terms time and space. Thus most of work is dedicated to reduce the computation time or the final cube size, Such as efficient cube computation and cube compression of data cube. Some techniques are also combination of both. Efficient and Compressed computation of data cube are Fundamental issues [1, 2]. Today's Space efficiency is not a big issue compare to time efficiency. Because now a day's many devices are available, these provide a lot of space in terms of RAM (Cache) and Flash Memory. So, in this paper focus point is time-efficiency.

The previous studies on data cubing mainly fall into 5 categories: (1) computation of full or iceberg cubes with simple or complex measures [1, 2, 3], (2) approximate computation of compressed data cubes, such as quasi-cubes, wavelet cubes, etc. (3) closed cube computation with index structure, such as condensed, dwarf, or quotient cubes (4) selective materialization of views and (5) cubes computation in stream data for multi-dimensional regression analysis. Among these categories, the first one, efficient computation of full or iceberg cubes, is a fundamental problem to the other categories.

Current MM cube Technique performs better compare to previous technique like Multiway, BUC, and Star Cube Techniques. In this paper I have presented MM cube Technique with Tree Data Structure to better Time-Efficiency of MM cube technique. MM Cube Technique is very useful for Retailer Application, Financial Application, and Decision Support System. Also beneficial to analysis historical records. We can predict future by analyzing past data. So, as possible as we can reduce computation time is very useful in various applications.

So, in this paper I have presented modified MM Cube Technique with performance study to better time-efficiency.

II. MM CUBE TECHNIQUE

MM-Cubing is a density-based method, where *MM* represents *major* and *minor* values. By distinguishing those values, the lattice space is factorized into one dense subspace and several sparse subspaces, and MM-Cubing applies specialized methods for each subspace. In order to perform highly efficient and highly adaptive Iceberg Cubing, it is necessary to have different treatments of frequent values from infrequent ones, because frequent

values tend to contribute to frequent cells [1] .Let us denote frequent values as major values, and infrequent ones as minor values. Adaptively **factorize** the aggregation space into dense subspaces and sparse subspaces.

Values in each dimension are categorized based on their frequencies:

- **Major Value** – High Frequency - Worth to do simultaneous aggregation
- **Minor Value** – Low Frequency - To do Recursive Calls

A. Factorization of Lattice Space

The factorization of the lattice space differentiates major values from minor ones, and these values are represented by separate points [1]. Thus, the new grid structure has 3*D* nodes. If we denote a major value by, a minor value by *I*, and ALL by *.

3-D lattice space can be written as:({M,I,*},{M,I,*},{M,I,*})

- Total Space: ({M, I,*}, {M, I,*}, {M, I,*})

B. Simeltaneous aggretion

The dense Subspace is computed by simultaneous aggregation by considering only major values and * values [1].

Simultaneous Aggregation in Dense subspace

- Dense: ({M,*}, {M,*}, {M,*})

C. Recursive Calls

The sparse subspaces are solved by recursive calls [1].

Recursive Calls for Sparse subspaces

- Sparse: (*I, {M, I,*}*, {M, I,*}).

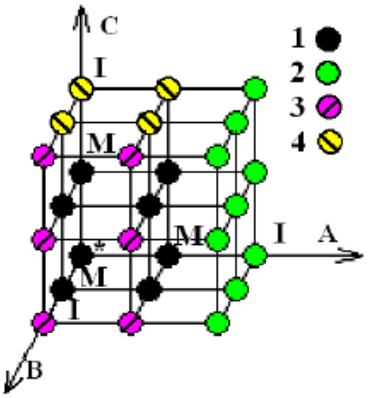

Figure 1: Factorization of lattice cube

III. IMPROVED MM CUBE TECHNIQUE

In section II we have presented the basic MM-Cubing Technique. Its efficiency is based on

(1) Calculating the dense subspace by simultaneous aggregation, which is efficient; [1]

(2) Calculating the sparse subspace by recursive calls, which divide the sparse subspaces into even smaller subspaces [1].

Here we introduce a performance improvement method which may try to improve time-efficiency of existing algorithm.

Proposed Tree Structure:

Existing algorithm is using ***Shared Array*** as a Data Structure; in proposed model I have used ***Tree Structure***. Tree is using Appriory-Pruning strategy which is facilitated Iceberg condition to internal node. If internal node doesn't satisfy *min_sup* value then that node will be cut-down. So, Traversal time is reduced compared to array structure. So, Cube Computation time can be reduced as possible as minimum.

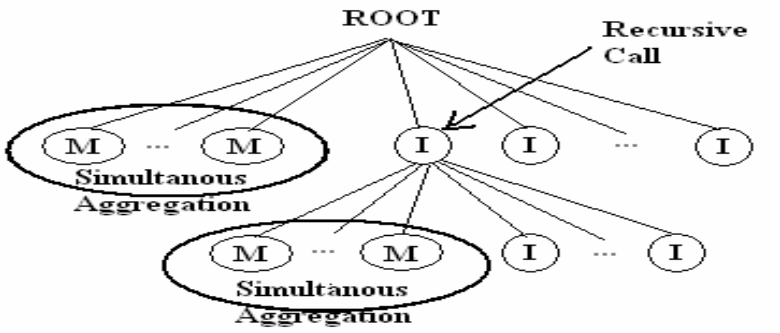

Figure 2: Proposed Tree Structure for MM Cube Computation

Iceberg Cubing: Only output those cubes with a support (frequency) at least *min_sup*.

Values in each dimension are categorized based on their frequencies:

- **Major Value** – High Frequency - Worth to do simultaneous aggregation
- **Minor Value** – Low Frequency - Probably better to check iceberg condition
- **Star Value** – Frequency below *min_sup* - No aggregation needed

Modified MM Cube Algorithm

Compute Iceberg Cube:
Input: (1) A relational table *R*, and (2) an iceberg condition, *min sup* (taking *count* as the measure)
Output: The computed iceberg cube.
Method: The algorithm is described as follows:
BEGIN
1. **for each** dimension *i*
2. **count and sort** the values in dimension *i (Approiry Pruning)*
3. **determine** *major* and *minor* values in each dimension
4. **do** simultaneous aggregation in the *dense subspace*
5. **for each** *sparse subspace*
6. **for each** *minor* value of that subspace{
7. **recursive call** *(*in *Tree data structure)*
8. **stop** further traversal of tree nodes for *minor* values
9. }
10. }
END

A. Count and Sort

Here first we are using count and sort algorithm and related Data Structure, which computes the frequency of the values and provides a data structure that facilitates the major value selection and the computation of the subspaces. It gathers the same values and sort according to count. Then store in tree as a data structure, it prunes the node which doesn't satisfy the *min_sup* value.

Assume we have the following input data. The value row is the values of a specific dimension in all the tuples.

INPUT: (*min_sup* = 2)

Table1: Sample input table

Tuple ID	1	2	3	4	5	6	7	8	9
Value	1	2	1	1	2	4	3	2	2

OUTPUT:

Table 2: Resulting Output Table

Count	**Value**	**Tuples**
4	2	2, 5, 8, 9
3	1	1, 3, 4

Note: Below *min_sup* values are discarded.

B. FACTORIZATION OF LATTICE SPACE:

Total Space : ({M, I,*}, {M, I,*}, {M, I,*}) All dimensions in Dense Subspaces {M,*}
Dense Subspace : ({M,*}, {M,*},{M,*}) Each Sparse Subspace has one dimension of {I}
3Sparse Subspaces : ({M, *}, {M, *}, {I})
({M, *}, {I},{M, I, *})
({I}, {M, I, *},{M, I,*})

C. Determine Major value: Do Simultaneous Aggregation

$$work\_done = T x \prod_{i=1}^{D} (1 + sum[i] / T)$$

Work done as the total amount of work done by the simultaneous aggregation [1]:
Where *T* is the number of tuples, *sum*[*i*] is the number of tuples that have a major value in dimension *i* and *D* is the number of dimensions. The simultaneous aggregation table size can be calculated as: $\prod_i (1 + MC[i])$, [2] where *MC*[*i*] is the number of major values in the *ith* dimension.
Dense Subspace: ({M,*}, {M,*}, {M,*})

D. Determining Minor Values : Recursive Call

The *sparse subspaces* are computed by recursive calls.Consider the *sparse subspace, (*I,{ M,I,*}*,{M,I,*}).* Decomposing the minor value, we get (I,{M,I,*},{M, I,*})

= (i1, {M, I,}, {M, I,*})*
+ (i2, {M, I,}, {M, I,*})*
+……where the first minor value is denoted by *i*1, the second by *i*2, an

IV. IMPLIMENTATION AND RESULT

I have implemented modified MM Cube Technique, which have Tree as a Data Structure. Tree has applied Approiry Pruning which cut-down the node which doesn't satisfy the min_sup condition. Here, I have represented one sample output of implemented technique. First it counts the value for each node then it applies min_sup suppose it's 2. Then which node has less than 2 values they all cut-down. Remaining values are stored and sorted. Then it find Major and Minor values. Do Simultaneous Aggression for major values, recursive calls for Minor values Dimension wise. Here I have reduced recursive calls by using tree. So, it minimizes computation time compare to Shared Array.

Table 3: Sample Table for Implementation

A	B	C
A1=8	B3=5	C2=3
A2=4	B1=3	C3=3
A3=2	B2=2	C4=3
A4=2	B4=2	C4=2
…..	…..	…..

Major:

A1=8, A2=4, B3=5

Minor:

A3=2, A4=2,

B1=3, B2=2, B4=2,

C2=3, c3=3, c4=3, c8=2

Min_Sup=2

Figure 3 shows flow of proposed Technique. After counting and sorting Major and Minor values are searched. First it searches in dimension A. Find Major and Minor. A1 is considered major values in A. Then it tries to search for same count value 8, if there is, apply Simultaneous Aggression otherwise go for recursive call loop. Here no same count value 8 so, it goes in recursive loop. There is A4 is major as per count value so, its major value again if there is same count value it applies Simultaneous Aggression, otherwise goes in recursive call loop, again find major and minor. Then it displays Dense and Sparse values.

Here A1, A2, B3 are Dense Subspaces. Remaining is Sparse Subspace.

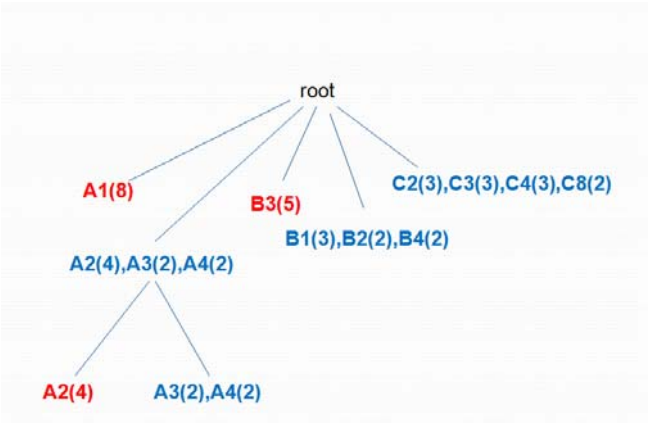

Figure 3: Flow of Proposed Technique

Figure 4 and 5 shows comparison between existing and proposed MM Technique.

Here, I have compared existing and proposed MM technique based on parameter like tuple, dimension, cardinality, min_sup value etc. Proposed MM Technique takes less computation time compare to existing MM Technique. Such a way proposed MM Cube Computation Technique is more efficient which is very useful for Data warehouse.

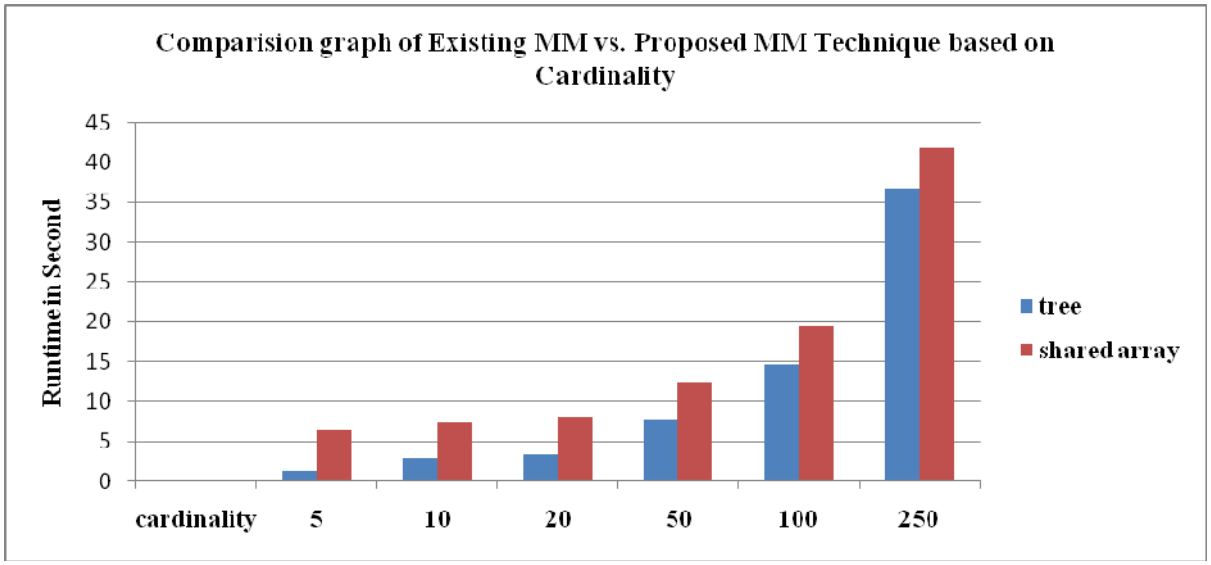

Figure 4: Comparison Graph of Existing vs. Proposed MM Technique based on Cardinality
W.r.t. Dimension=6, Tuple=5000, min_sup=2

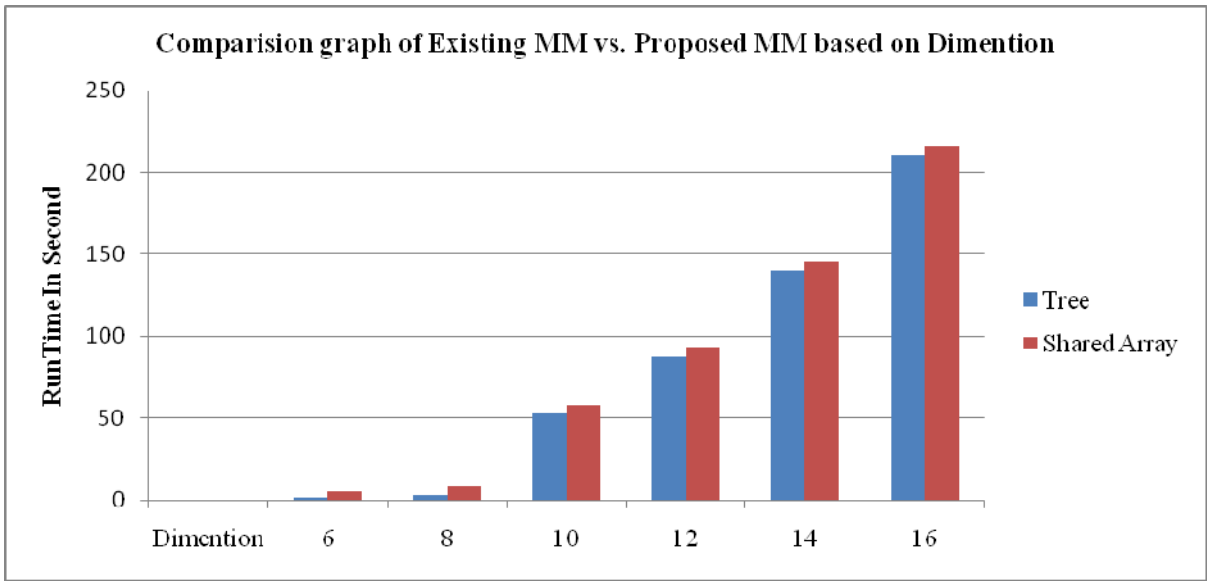

Figure 5: Comparison Graph of Existing vs. Proposed MM Technique based on Dimension
W.r.t Tuple=5000, Cardinality= 100, min_sup=2

- **System Specification:**
- Both the two algorithms Existing and Proposed MM Cube are coded using Microsoft Visual Studio 2008 C#.Net with MS Access 2007 database on a Core i3 M350 @ 2.27GHz, system with 3GB of RAM. The system runs Windows 7 Ultimate with Service Pack 1.

V. CONCLUSION

In this paper I have implemented proposed MM Cube Technique to improve its Time-Efficiency. Existing MM cube has used Shared Array as Data Structure. And I have replaced Data Structure with Tree, which is used Approiry Pruning Strategy. That applies Iceberg condition to internal node. It cut-down that node which doesn't satisfied min_sup values. So, recursive calls reduces. Thus Cube Computation time reduces compared to existing technique. I have compared proposed technique with existing technique based on dimension and cardinality. It's concluded that proposed technique is more faster compare to existing technique. So, proposed MM technique is more efficient and very useful in Data Warehouse and OLAP system.

REFERENCES:

[1] Zheng Shao, Jiawei Han Dong Xing. "A MM-Cubing: Computing Iceberg Cubes by Factoring the Lattice Space ".University of Illusions at-Champaign, Urbana, IL 61801,USA,2004
[2] J. Han, J. Pei, G. Dong, and K. Wang. "Efficient Computation of Iceberg Cubes with Complex Measures." ,SIGMOD'01
[3] K. Beyer and R. Ramakrishnan. "Bottom-up computation of sparse and iceberg cubes" SIGMOD'99, 359–370.

[4] Laks V.S. Lakshmanan, Jian Pei, Yan Zhao "Efficacious Data Cube Exploration by Semantic Summarization and Computation" University of British Columnbia,Canada {laks,yzhao} @cs.ubc.caState University of New York at Buffalo,USA.jianpei@cse.buffalo.edu,2002
[5] Findlater, L., and Hamilton, H.J. "Iceberg Cube Algorithms: An Empirical Evaluationon Synthetic and RealData," Intelligent Data Analysis, 7(2), 2003. Accepted April,2002
[6] Harinarayan,V.Rajaraman,A. Andullman." Implementing data cubes efficiently". In proceedings of ACM Special Interest Group on Management of Data (SIGMOD). 205–216.
[7] Gray, J. Bosworth, A., Layman, A., and Pirahesh, H. 1996."Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-total". In Proceedings of International Conference on Data Engineering (ICDE). 152–159.
[8] Dong Xin Zheng Shao Jiawei Han Hongyan Liu." C-Cubing: Efficient Computation of ClosedCubes by Aggregation-Based Checking "University of Illinoise at Urbana-Champaign,Urbana, IL 61801,USATsinghua University, Beijing., China,2006
[9] J. Shanmugasundaram, U. Fayyad, and P. S. Bradley. "Compressed data cubesfor olap aggregate query approximation on continuous dimensions". In Kdd'99.

AUTHORS PROFILE

RIDDHI DHANDHA has completed her B.E. Computer from Atmiya University, Rajkot. She was working as a Lecturer in Indus University, Ahmadabad for 4 years. Currently she is pursuing M.E. from Gujarat Technological University, Ahmadabad.