JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 1, 2022, p. 149-158
https://publishoa.com
ISSN: 1309-3452

Optimal Data Cube Materialization in Hyper Lattice Structure in Data Warehouse Environment

Ajay Kumar Phogat[a] and Suman Mann[b]

[a]Research Scholar, Guru Gobind Singh Indraprastha University, Maharaj Surajmal Institute, 110058, New Delhi, India

[b]Information Technology Department, Maharaja Surajmal Institute of Technology, 110058, New Delhi, India

Received 2022 March 15; **Revised** 2022 April 20; **Accepted** 2022 May 10.

ABSTRACT

In the business intelligence field, a quick and accurate response is required for efficient decision-making, for this purpose multidimensional logical model is built as per the current business requirements. All the dimension tables are designed based on a fact table that is treated as the base cuboid in lattice structure, which is not frequently changed, but business requirements may be frequently changed over time and new dimensions are need to be added to the existing structure to fulfill the updated requirements of the business. Adding a new dimension to the lattice structure will increase the structure almost double that requires huge space. So hyper lattice is introduced to overcome the problem of huge structure in which we can add a new dimension above the base cuboid in the existing structure. To achieve the optimal space and time computation , this paper aim to drill down and move up in hyper lattice using an efficient algorithm. There may be multiple paths in hyper lattice to reach the final view and have a different volume of storage and time computation. An efficient algorithm is proposed in this paper that can answer a huge query in the hyper lattice structure with a quick access time.

KEYWORDS

Data Cube; Multidimensional Model; Hyper Lattice; Query Processing

1. Introduction

Data is in the form of an electronic repository in the data warehouse. Data models are the basic building blocks for the data warehouse. In Lattice Structure data cubes or cuboids are the basic forms to view data. A data cube may be built by the subset of database attributes. Some attributes are known as the attribute of interest used for measurement and others can be selected as functional attributes of dimensional attributes. [3][4][5] In hyper lattice structure dimensions and facts are defined in the data cube, where dimensions are the entities concerning the data that is stored in the data cube. Every dimension in hyper lattice may have one or more tables associated with it, such as time or location. On the other hand, the Fact table contains keys that belong to each dimension. N- Dimensional cuboids are used to represent the fact table in the data warehouse. In the hyper lattice of cuboids, there may exist more than one base cuboid having some similar dimensions [5]. OLAP(Online analytical processing) operations deal with aggregate data. A consumer can perform OLAP operations by initial requirements that are specified at the time of developing multidimensional model structure by the developers. These initial requirements may be used for extended data analysis. OLAP constitute some of the basic operations on data model such as roll-up, which includes moving upward in the level of hierarchy, second is drill-down operation, which decreases level of aggregation by moving down in hierarchy in the lattice structure, the third operation is slice and dice, in which slicing includes selecting one dimension in the cube that results in a subcube whereas, dice operation includes a selection of two or more dimension in a cube, forth operation is the pivot that performs a rotation on data axis to view different viewpoint. A high level of abstraction may be achieved through roll-up operation, as well as aggregate cuboids can be computed by dimension reduction in the lattice of cuboids in the data warehouse, and for getting detailed data drill-down operation may be performed. Mapping of different levels of concepts may be done in OLAP to produce different aggregate views, such as low-level to the high-level aggregate concept can be performed in OLAP through concept hierarchy, which means one dimension can be shown in various aggregate levels. The level of abstraction depends on the requirement of the business. Suppose if we have location as one of the dimensions then it can

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 1, 2022, p. 149-158
https://publishoa.com
ISSN: 1309-3452

be expressed as city, state, country, etc. Therefore concept hierarchy may be partial or total order [10][11][21]. An example of total order can be as :

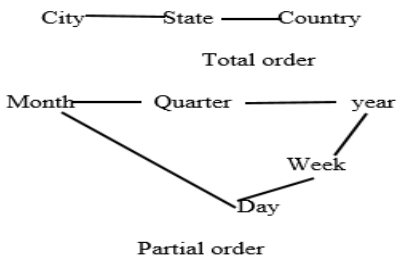

Figure 1
Example of Total and Partial order

A big data warehouse application has several dimensions and each dimension has multiple hierarchy levels, so it is a big challenge for data cubing. Quick response time and accuracy are key factors for any business's success. The query process time is a crucial aspect as timely access to data in large databases, especially in distributed databases, and is the prerequisite for successful business applications. Data is growing rapidly proportionate to the dimension required by a business. In terms of space and time computation process, it is very costly to manage. To process queries efficiently, the data warehouse uses several different views. Pre-computed data cube is useful to solve the problem of space and time computation cost. Queries may take a long time to process as the size of data is very large in data warehouse and complexity of the query, which is not acceptable in a DSS (Decision Support System) environment. Pre-computed data cube may enhance the process of data warehouse design and query process. Different techniques like query optimizers and query evaluation approaches are being used to reduce query execution time [23][24].

Data cube view materialization is one of the effective approaches used in decision support systems to reduce computation time. Therefore, researchers are always in search of better algorithms, which can choose the best views to be materialized. Below is an example of a lattice structure having three dimensions Time (T), Product(P), and Sales(S) in the data warehouse. Cuboid TPM is the base cuboid, moving up in the hierarchy of cuboid, we reach to apex cuboid having 0-dimension [32][27]. There are some drawbacks of the traditional lattice as the addition of new dimensions enhances the structure to almost double as all the new dimensions have to connect the lower and upper bound dimension in the structure. It is very difficult to materialize the whole lattice. So to overcome this problem a concept of hyper-lattice is introduced.

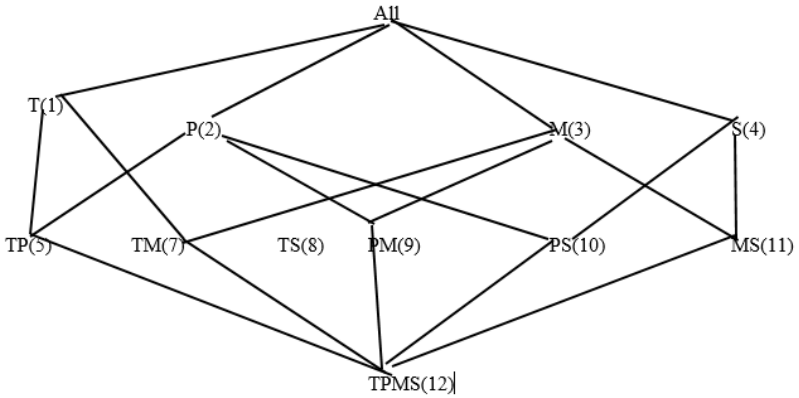

Figure 2.
Lattice structure with 4 dimensions.

Paper is Organization as follow:

Sec 2. Explore the related work done by various authors related to materialization view selection of data cube. Sec 3.

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 1, 2022, p. 149-158
https://publishoa.com
ISSN: 1309-3452

contains the hyper lattice framework and its structure that describe the overlapping of lattices in the structure. In this section important features are explained and the difference between traditional lattice and hyper lattice is discussed. An algorithm is proposed for efficient view selection in hyper-lattice known as MinCostPath for better path selection in hyper-lattice. Sec 4. Describe the implementation of the algorithm with an example of hyper-lattice having two base cuboids with optimal solution of path selection using MinCostPath algorithm for better query response. Sec 5 presents the conclusion and future scope.

2. Literature Survey

In the literature survey section, the existing research work is discussed in terms of view materialization and query path selection in the lattice structure. Query optimization and materialization of view are the main two works on that continuous research have been done in past by many authors. Materialized view problem is also stated as the view selection problem. To determine the best and most effective possible collection of views with the available storage. It is very difficult to make a decision that which view is best to materialize. The problem of view selection is done with some constraints such as time constraints, space constraints, availability constraints, join and group-by constraints, etc. An algorithm related to optimizing the query path selection has been proposed by many authors to reduce query cost.

V. Harinarayan, Rajaraman, A., Ullman; [2] proposed an algorithm based on greedy approach that select the right set of views to materialize, based on various constraints in lattice framework. A. Shukla, PM Deshpande, JF Naughton; [4] proposed a multi-cube algorithm that deals with single cube computation as well as extends it to enhance the performance of multi-cube, for this purpose three algorithms have been purposed, namely SimpleLocal, ComplexGlobal, and SimpleGlobal that choose the aggregates for precomputation from multi cube schemas. It works more efficiently in view selection as compared to the algorithm proposed by Harinarayan. H. Gupta, I. S. Mumick;[5] proposed an AND-OR viewgraph algorithm that works on the polynomial-time greedy framework. In this algorithm AND view has the unique evaluation value whereas, OR view states that every view can be computed by its connected view. It is an approximation greedy algorithm that selects a set of views that to materialized data cube to minimize the response time of query for OR view graphs, to deliver an optimal solution they also design an A* heuristic algorithm.

Zhang, Chuan, Xin Yao, and Jian Yang [7]proposed an algorithm based on the heuristic approach that finds a set of views based on multiple global processing plans of the query with the help of the Multiple View Processing Plan (MVPP). Wie Ye[10] also proposed a greedy-based selection algorithm for view selection under storage cost constraints to solve the problem of view selection in the distributed data warehouse. J.Hen, J.Pei,G.D an J. Yang d K. Wang [6] proposed an effective and practical approach compared to the heuristic for materialized view selection based on genetic algorithms, which is found very effective in reducing the query maintenance & query cost.

I. Mami, R. Coletta, and Z. Bellahsene; [21] proposed a constraint-based programming framework for materialized view selection in the data warehouse. K. Aouiche, P. Jouve, and J. Darmont;[16] proposed a cluster-based framework that selects similar queries, and also proposed an algorithm that constructs a set of candidate view for efficient view selection known as a merging algorithm. Chaudhari, Manoj S., and Chandra shekhar Dhote; [20] also proposed a cluster-based dynamic algorithm for view selection which extracts the representative dimension from a set of queries, then creates a cluster of queries to generate a set of candidate view and then adjust the final materialized view dynamically. A. Gosain[26] proposed a stochastic algorithm called Particle Swarm Optimization that minimizes the query processing time efficiently compare to genetic algorithm in lattice framework. Gosain, Anjana, and Kavita Sachdeva; [30] proposed an algorithm called SRCSAMVS to optimize the process of view selection using lattice structure considering some constraints such as probability, dimension, space, maintenance, dimension, etc., the algorithm performs efficiently even if we increase the number of queries and produce minimum query cost.

S. Soumya, C.Nabendu; [22] proposed an algorithm known as dynamic query path selection in the lattice with a given concept hierarchy. The purpose of the algorithm is to find the optimal path from source to target cuboid at minimum query access time. The algorithm first selects the partially materialized cuboids that are present in cache or primary,

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 1, 2022, p. 149-158
https://publishoa.com
ISSN: 1309-3452

secondly, if the target cuboid is not found in the cache or primary memory then cuboids are constructed from secondary memory using Least Recently Used method.

Sen, Soumya, Agostino Cortesi, and Nabendu Chaki [30] proposed an algebraic structure of hyper-lattice that is more flexible than traditional lattice as it is very convenient to add a new dimension. In hyper lattice, there is an overlapping of two or more lattices that shared both time and space. He also proposed an algorithm to create a hyper-lattice and also proposed selectMinCost algorithm for path selection that reduces query cost time.

3. Hyper- Lattice Framework

3.1. Introduction to Hyper-Lattice

The structure of Hyper lattice includes a lower bound and an upper bound. Hyper Lattice structure is comparatively more flexible than previous traditional lattice in the data warehouses. It is very convenient to add another dimension in a hyper lattice.

In Hyper Lattice, overlapping lattice's common elements meet at a common point that is called the least upper bound of each lattice, whereas the GLB should be unique of each lattice [28].

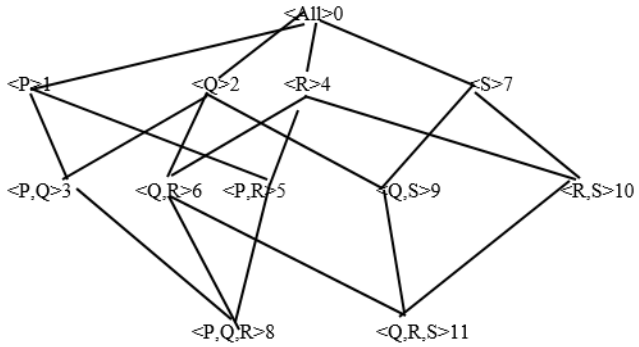

Figure 3.
Hyper-lattice of cuboids having overlapping of two set.

In the above example, there are lattice L1 and L2 overlapping to each other and shared common space.
L1: {<PQR>, <PQ>, <PS>, <QR>, <P>, <Q>, <R>, <ALL>}
L2: {<QRS>, <QR>, <QS>, <RS>, <R>, <S>, <ALL>}

The Hyper-lattice is used to save both time and space, from fig. 2 we may state that common space is shared by different lattices, therefore the cuboids have to be stored only once if they exist at a common location. They don't need to store separately for every single lattice as it is easy to search the data because all the cuboids of lattices are stored in the same storage location. In traditional lattice, it may take more time to search data as each lattice are to be stored in a separate memory location. Hyper lattice also eliminates data redundancy and makes data more consistent.

Below is a hyper-lattice schema that represents the above two overlapped lattices. In the schema, we have two fact tables <PQR> and <QRS> having corresponding associated dimension tables P, Q, R, S.

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 1, 2022, p. 149-158
https://publishoa.com
ISSN: 1309-3452

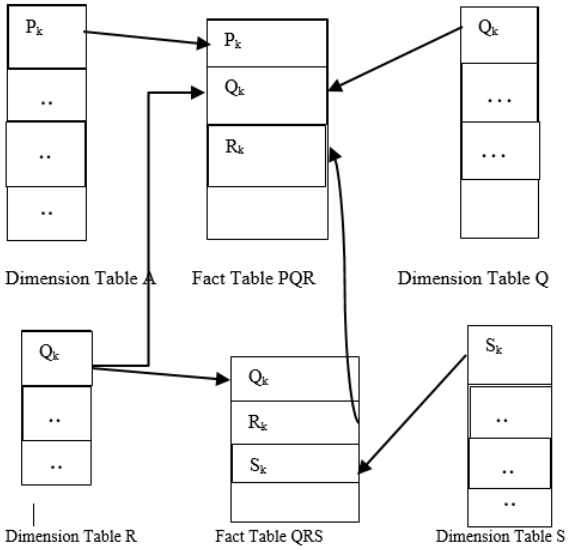

Figure 4.
Hyper Lattice Schema

3.2. Characteristics of Hyper Lattice

1. One or more integrant lattices creates a Hyper-lattice.
2. In Hyper-lattice we can add any number of dimensions.
3. Additional or new dimensions can be added in hyper-lattice between level to (N-1).
4. Apex cuboid will same for the whole Hyper-lattice, but base cuboid may be increased with any number that is dependent on the newly added dimension in the lattice.
5. New dimensions may be added more than once without creating a new lattice structure.

3.3. Traditional Lattice vs Hyper-lattice.

Data warehouse is represented as a Multidimensional data model that is appropriate for the logical arrangement of data in the form of cuboids and also for analytical processing. All the possible combinations of cuboids from the base cuboid form a lattice structure [26].

In traditional lattice structure apex cuboid known as 0-dimension may be reached by moving up in hierarchy from a base cuboid having n-dimensions. Alternate paths can be found in the lattice from base cuboid to intermediate level cuboids. A hyper lattice is a hybrid form of the traditional lattice structure having more than one base cuboid. Storage space is shared by two or more lattices in Hyper-lattice structure. Hyper-lattice is flexible than traditional lattice another new dimension may be added but in the case of traditional lattice, we cannot add new dimensions in the structure. Additional lattice structures have to be created for adding new dimensions that increase the problem of storage while in hyper lattice structure we can add any number of dimensions between level N to N-1 level.

Table 1. Difference between Traditional Lattice & Hyper Lattice Structure

S.No	Traditional Lattice Structure	Hyper-Lattice Structure
1.	It has only one base cuboid.	It has more than one base cuboid.

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 1, 2022, p. 149-158
https://publishoa.com
ISSN: 1309-3452

2. Addition of a new dimension will almost double the existing lattice structure
3. More storage space is required as the size of the lattice will be double after the new dimension.
4. Query path selection process requires exploring all the cuboids of the lattice in most cases.
5. Query path selection process requires exploring all the cuboids of the lattice in most cases.

Any number of dimensions can be added to the existing hyper-lattice structure above the base cuboid.
Solve the problem of storage as overlap lattices share a common space
Query path selection process required exploration of related lattice cuboids only.
Query path selection process required exploration of related lattice cuboids only.

Use of Hyper-lattice for efficient query processing

In the lattice structure information retrieval using hyper lattice, we may find more than one path from source to target cuboid. Selecting the best path leads to efficient resource utilization.

The size of the cuboid may be considered as the cost factor. In hyper-lattice the cuboid size can be calculated by multiplying the tuples in a cuboid with the size of the associated cuboid.

Cuboids consisting of dimensions D1, D2, D3. . . Dn. These dimensions in cuboids can be represented as Attributes A1, A2, A3. . . An. There can be one or measure in each associated cuboid. Suppose if a cuboid having Y tuples with size of the measure is X.

Then the cuboid size may be computed as

$$Y^* \left(\sum_{i=1}^{D} Si + X\right) \qquad (i)$$

Suppose there are 50 tuples in a cuboid and 5 dimensions with a measured value of 10, then the size of the cuboid will be 50*5+10=260

Selection of Minimal Cost path in Hyper Lattice of cuboid

The following algorithm MinCostPath is proposed for the selection of data cuboids in hyper-lattice.

Step 1: Create Array A and B to store the source and target index values of the dimensions.
Step 2. If array B is the Subset of array A
Push the index values of source code in an array C
Else
EXIT
Step3: Redo steps 3 to 9 until we reached to target cuboid.
Step 4: Search every possible cuboid in the hierarchy from the current initial cuboid in structure.
Step 5: Create and initialize an array D that stores the valid subset of cuboids index value from the above defined array C
Step 6: push the index value from array D to an array of hashmap

objects having mapping of index value and their distinct value at each level in the hyper lattice that reduce the time complexity to O(n)

```
Hashmap< Int, Int > hm=new Hashmap<Int, Int>();
ArrayList<HashMap< Int, Int >> myArrayMap = new
ArrayList<HashMap< Int, Int >>();
hm.put(index value, distinct value) /* distinct value will be the
size of cuboid associate with the index value*/
myArrayMap.add(hm);
```

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 1, 2022, p. 149-158
https://publishoa.com
ISSN: 1309-3452

Step 7: sort ArrayList index values as per cuboid minimum distinct values.
Step 8: Push the sorted index value in a Queue Q
Step 9 If Q=Target cuboid index value
Break the loop
Else go to Step 3

Suppose there are 50 tuples in a cuboid and 5 dimensions with a measured value of 10, then the size of the cuboid will be 50*5+10=260

4. Implementation of Algorithm

Hyper Lattice in Fig 3. having some distinct values for the hyper-lattice structure having dimensions P, Q, R and S are 10,20,15, and 25 respectively. In the hyper-lattice we have to find the optimal path between the source and target cuboid namely <Q,R,S> and <R> using the MinCostPath algorithm.

Solution: We have found two paths for the above query given:

First, According to our proposed algorithm, The Cuboids that are generated from <Q,R,S> are <Q,R>,<Q,S><PR> and <RS>.

Array A & B stores the dimension index value of source & target cuboid, then checks whether array B is a subset of array A, among the above four cuboids only two cuboids satisfy the pattern criteria, that are <QR> and <RS>if the condition is true then push the index value to array C.

Calculate the distinct values of each cuboid that satisfied the pattern criteria for path selection.

Cuboid <QR> contains 20*15=300 distinct values, whereas cuboid <RS> contains 15*25=375 distinct values, then in next step create an array of hashmap objects.

Hashmap< Int, Int > hm=new Hashmap<Int, Int>();
ArrayList<HashMap< Int, Int >> myArrayMap = new ArrayList<HashMap< Int, Int >>();
hm.put(index value, distinct value) /* size of distinct value will be the size of cuboid associate with the index value*/
at level 1
hm.put(6,300);
hm.put(9,375);
at level 2
hm.put(2,20);
hm.put(3,15);
select the the cuboid having least distinct value at each level
add the the hashmap object in the array myArrayMap
myArrayMap.add(hm);

Thus our algorithm MinCostPaths selects the cuboid <BC> having index & distinct value as (6,300) in this step. The cuboid generated at next level are <B> with hashmap value (2,20) and <C> (3,15) . In these cuboids <C> will be selected as it is the target cuboid as its distinct value is lower than cuboid B. Thus according to our algorithm the optimal path selected will be <BCD> →<BC>→<C>. In figure 5 the optimal path is denoted by a dark line. By using this algorithm we can achieve a reduction in space utilization. Group-By operation is the same as the roll-up operation in the hyper lattice, so the distinct values in the data set are proportional to the group-by, hence our algorithm also reduces the computation time.

15*25=375 distinct values, then in next step create an array of hashmap objects.

Hashmap< Int, Int > hm=new Hashmap<Int, Int>();
ArrayList<HashMap< Int, Int >> myArrayMap = new ArrayList<HashMap< Int, Int >>();
hm.put(index value, distinct value) /* size of distinct value will be the size of cuboid associate with the index value*/
at level 1 hm.put(6,300);
hm.put(9,375); at level 2 hm.put(2,20);

JOURNAL OF ALGEBRAIC STATISTICS
Volume 13, No. 1, 2022, p. 149-158
https://publishoa.com
ISSN: 1309-3452

hm.put(3,15);
select the the cuboid having least distinct value at each level add the the hashmap object in the array myArrayMap
myArrayMap.add(hm);

Thus our algorithm MinCostPaths selects the cuboid <BC> having index & distinct value as (6,300) in this step. The cuboid generated at next level are <B> with hashmap value (2,20) and <C> (3,15) . In these cuboids <C> will be selected as it is the target cuboid as its distinct value is lower than cuboid B. Thus according to our algorithm the optimal path selected will be <BCD> -> <BC> -> <C>.In fig 3 the optimal path is denoted by a dark line. By using this algorithm we can achieve a reduction in space utilization. Group-By operation is the same as the roll-up operation in the hyper lattice, so the distinct values in the data set are proportional to the group-by, hence our algorithm also reduces the computation time.

Table 2. Hashmap Table values at different levels

Hyper-lattice Level	Cuboid Index value	Distinct value
Level 1	6	300
9		375
Level 2	2	20
3		15

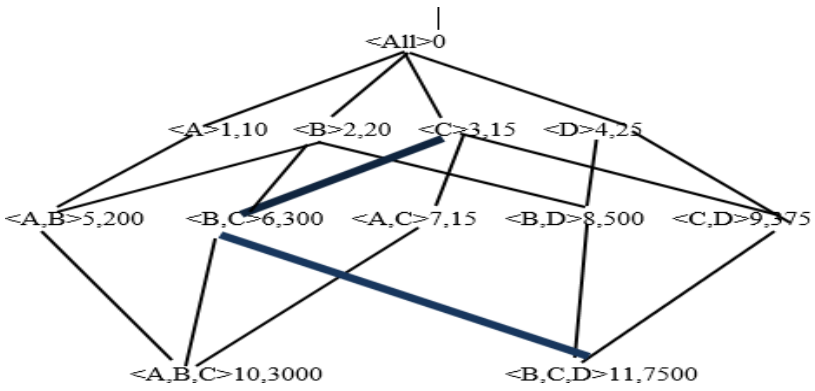

Figure 5.
Optimal path selection in a hyper lattice of cuboid

5. Conclusion & Future Scope

In this paper, we have proposed a MinCostPath algorithm for optimal path selection in Hyper Lattice of Cuboid in the data warehouse which can resolve the issue of space and time computation. The overlapping cuboid can save the memory space by sharing the same memory in the system and time computation is reduced by only traversing the particular base cuboid instead of all base cuboids.

References

[1] Inmon, William H. "Building the data warehouse, 1992." QED Information Sciences, Wellesley, MA (1992).
[2] V. Harinarayan, Rajaraman, A., Ullman,“ Implementing Data Cubes Efficiently”, In ACM SIGMOD International Conference on Management of Data, ACM Press, New York (1996) pp. 205-216.
[3] J.Yang, K. Karlapalem, and Q. Li. “A framework for designing materialized views in data warehousing

environment", proceedings of 17th IEEE International conference on Distributed Computing Systems, Maryland, U.S.A., May 1997.
[4] A. Shukla, PM Deshpande, JF Naughton, "materialized view selection for multidimensional datasets", Proceeding of 24th international conference on very large databases, NEW York, August 1998,pp 488-499.
[5] H. Gupta and I. S. Mumick, "Selection of views to materialize in a data warehouse," in IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 1, pp. 24-43, Jan. 2005.
[6] Zhang and J. Yang, "Genetic algorithm for materialized view selection in data warehouse environments," Proceedings of the International Conference on data Warehousing and Knowledge Discovery, LNCS, vol.1676,pp. 116-125, 1999.
[7] S. Amit, D Prasad, N.F. Jeffrey, "Materialized view selection for multi-cube data models, In Proc. of 7th Int.conference on Extending database Technology: Advances in Database Technology, Springer 2000, pp 269-284.
[8] Zhang, Chuan, Xin Yao, and Jian Yang. "An evolutionary approach to materialized views selection in a data warehouse environment." IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 31.3 (2001): 282-294.
[9] J.Hen, J.Pei,G.D and K. Wang, 'Effffficient computation of iceberg cubes with complex measures," in proc. 2001 ACM-SIGMOD Int. conference Management of data (SIGMOD'01),May2001,PP1-12.
[10] Y.Chen, G. Dong,J.Han, B.W.Wah, and J.Wang, "Multidimensional regression analysis of time series data streams," in proc.2002 International conference on very large data Bases(VLDB'02),Hong kong, China, Aug.2002,pp.323-334
[11] Antoaneta Ivanova, Boris Rachev, "Multidimensional models – Constructing data cube", International Conference on Computer Systems and Technologies- CompSysTech'2004.
[12] X.Li, J.Han, and H. Gonzalez, "High dimensional OLAP: a Minimal cubing approach,"in Proc. 2004 Int.Conf.Very Large Databases (VLDB'04),Toronto Canada,Aug. 2004,pp. 528-539.
[13] I. Antoaneta, R Boris, "Multidimensional models-constructing data cube" Int. conference on computer systems and technologies-CompSysTech'2004, V-5pp1-7
[14] L.Y.Wen, K.I. Chung, "A genetic algorithm for OLAP data cubes" Knowledge and information systems, January 2004,volume 6,Issue1,pp 83-102
[15] H. Gupta and I. S. Mumick, "Selection of views to materialize in a data warehouse," in IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 1, pp. 24-43, Jan. 2005, doi: 10.1109/TKDE.2005.16
[16]K. Aouiche, P. Jouve, and J. Darmont. Clustering-based materialized view selection in data warehouses.In ADBIS'06, volume 4152 of LNCS, pages 81–95, 2006.
[17] Sen, Soumya, Nabendu Chaki, and Agostino Cortesi. "Optimal space and time complexity analysis on the lattice of cuboids using galois connections for data warehousing" 2009 Fourth International Conference on Computer Sciences and Convergence Information Technology. IEEE, 2009.
[18] Mann, Suman, Anjana Gosain, and Sangeeta Sabharwal, "OO Approach for Developing Conceptual Model for A Data Warehouse." *Journal of Technology and Engineering Science* 1.1 (2009): pp 79-82.
[19] Y.A Kudryavtsev, S.D.Kuznetsov, "A Mathematical model of the OLAP cubes",Programming and computer software,Vol35, No 5,2009, pp 257-265
[20] Chaudhari, Manoj S., and Chandrashekhar Dhote. "Dynamic materialized view selection algorithm: a clustering approach." In International Conference on Data Engineering and Management, pp. 57-66. Springer, Berlin, Heidelberg, 2010.
[21] I. Mami, R. Coletta, and Z. Bellahsene, "Modeling view selection as a constraint satisfaction problem", In DEXA, pp 396-410,2011
[22] S. Soumya, C.Nabendu, "Efficient traversal in data warehouse based on concept hierarchy using Galois Connections, In proc. of second Int. Con on Emerging applications of information technology,2011,pp 335- 339
[23] I. Mami and Z. Bellahsene,"A survey of view selection method" SIGMOD Record, March 2012 (Vol. 41, No. 1), pp 20-30
[24] Gosain, Anjana, and Suman Mann. "Space and Time Analysis on the Lattice of Cuboid for Data

Warehouse." International Journal of Computer Applications, volume 77-No.3 (2013).

[25] Gosain, A., Mann, S. Empirical validation of metrics for object oriented multidimensional model for data warehouse. International Journal of System Assurance Engineering Management 5, 262–27(2014). https://doi.org/10.1007/s13198-013-0155-8.

[26] Gosain, Anjana. "Materialized cube selection using particle swarm optimization algorithm." Procedia Computer Science 79 (2016): 2-7.

[27] Gosain, A., Mann, S. Object Oriented Multidimensional Model for a Data Warehouse with Operators. International Journal of Database Theory and Application Vol. 3, No. 4, December,2010

[28] Sen, Soumya, Agostino Cortesi, and Nabendu Chaki. "Hyper-lattice algebraic model for data warehousing". Springer International Publishing, 2016.

[29] Kumar, TV Vijay, and Biri Arun. "Materialized view selection using HBMO." International Journal of System Assurance Engineering and Management 8, no. 1 (2017): 379-392.

[30] Gosain, Anjana, and Kavita Sachdeva. "Selection of materialized views using stochastic ranking based Backtracking Search Optimization Algorithm." International journal of system assurance engineering and management" 10.4 (2019): 801-810.

[31] Mann, Suman, and Ajay Kumar Phogat. "Dynamic construction of lattice of cuboids in data warehouse." Journal of Statistics and Management Systems 23.6 (2020): 971-982.

[32] Prashant, R., Suman, M., & Eashwaran, R. (2021). "Efficient Data Cube Materialization". In *Advances in Communication and Computational Technology* (pp. 199-210). Springer, Singapore.