International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE)
Volume 2 Issue 1 –OCTOBER 2013.

Data Mining Analysis using Query Formulation In Aggregation Recommendation

L. Gomathi[#1], K. Ramya[*2]

[#]*Associate Professor, Dept. Of Computer Application, Muthayammal College of Arts & Science*

[2] *geetharamee@gmail.com*

[*] *Dept. Of Computer Application, Muthayammal College of Arts & Science*

***Abstract*- Recommender systems are becoming increasingly important to individual users and businesses for providing personalized Recommendations analyze data efficiently, Data mining systems are widely using datasets with columns in horizontal tabular layout. Preparing a data set is more complex task in a data mining project, requires many QLA queries, joining tables and aggregating columns. Conventional RDBMS usually manage tables with vertical form. Aggregated columns in a horizontal tabular layout returns set of numbers, instead Relational databases are acceptable repository for structured data; integrating , Query Formulation Algorithm with a relational DBMS is an essential research issue for database programmers.**

***Index Terms*- Aggregation, Data Mining, LCM (Linear time Closed item set Miner) , Query Formulation Algorithm.**

I. INTRODUCTION

Aggregation recommendations are new class of function to return aggregated columns in a horizontal layout. Most algorithms require datasets with horizontal layout as input with several records and one variable or dimensions per columns. Managing large data sets without DBMS support can be a difficult task. Trying different subsets of data points and dimensions is more flexible, faster and easier to do inside a relational database with QLA queries than outside with alternative tool. Aggregation recommendations can be performing by using operator, it can easily be implemented inside a query processor, much like a select, project and join. LCM operator on tabular data that exchange rows, enable data transformations useful in data modelling, data analysis, and data presentation There are many existing functions and operators for aggregation in Structured Query Language. The most commonly used aggregation is the sum of a column and other aggregation operators return the average, maximum, minimum or row count over groups of rows.

All operations for aggregation have many limitations to build large data sets for data mining purposes. Database schemas are also highly normalized for On-Line Transaction Processing (OLTP) systems where data sets that are stored in a relational database or data warehouse. But data mining, statistical or machine learning algorithms generally require aggregated data in summarized form. Data mining algorithm requires suitable input in the form of cross tabular (horizontal) form, significant effort is required to compute aggregations for this purpose. Such effort is due to the amount and complexity of QLA code which needs to be written, optimized and tested. Data aggregation is a process in which information is gathered and expressed in a summary form, and which is used for purposes such as statistical analysis. A common aggregation purpose is to get more information about particular groups based on specific variables such as age, name, phone number, address, profession, or income. Most algorithms require input as a data set with a horizontal layout, with several records and one variable or dimension per column. That technique is used with models like clustering, classification, regression and PCA. Dimension used in data mining technique are point dimension. There are several advantages for aggregation recommendation. First one is aggregation recommendations represent a template to generate QLA code from a data mining tool. This QLA code reduces manual work in the data preparation phase in data mining related project. Second is automatically generated code, which is more efficient than end user written QLA code. Thus datasets for the data mining projects can be created in less time. Third advantage is the data sets can be created entirely inside the DBMS.

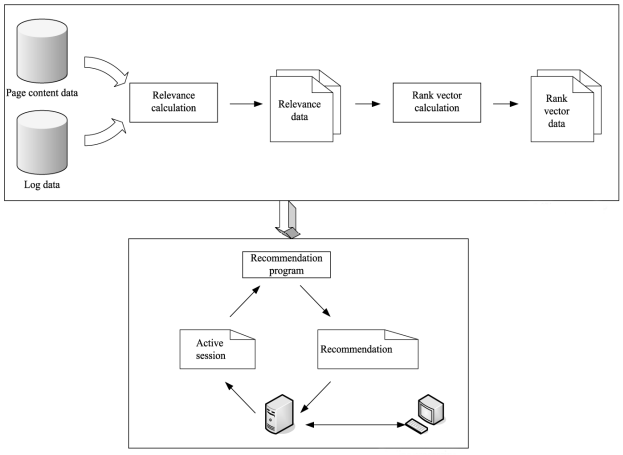

Fig 1: Architecture Diagram

Aggregation Recommendations

We introduce a new class of aggregations that have similar behavior to QLA standard aggregations, but which produce tables with a horizontal layout. In contrast, we call standard QLA aggregations vertical aggregations since they produce tables with a vertical layout. Aggregation recommendations just require a small syntax extension to aggregate functions called in a SELECT statement. Alternatively, aggregation recommendations can be used to generate QLA code from a data mining tool to build data sets for data mining analysis. We start by explaining how to automatically generate QLA code.

II. BACKGROUND WORKS

Recommender System has it's starting from information retrieval [5] and to consumer choice finding in marketing [6]. RS emerged in 1990's in order to overcome overload in information. This system wholly relies on rating concept. For acquiring true ratings, product rating acquisition problem [7] is one of the problem developed. The key input to RS is rating which can be implicit or explicit.

There is an inverse relationship between accuracy and diversity. Diversity should be increased by having minimal loss in accuracy [8]. Recommendation can not only be given to single product, it also allows recommending collection of products [9] [10].

Overcoming trade-off between accuracy and diversity, [11] describes an approach called variance based approach. This problem plays a significant role in RS since having diverse recommendations will give coverage of more products.

"Long-Tail' products are one that has high impact on sales. They are products which are less popular but earn high profit. Hongzhin, Bin says in [12] that by discovering these products, diversity can be increased.

Gediminas [8] gave a graph approach by using Max Flow and Max Bipartite problem. It gives top-N recommendations with maximum diversity. Based on taxonomy [11] i.e., by classifying products based on category, diversity can be attained. Fuguo Zhang tells in [7], diversity can be achieved by using trust between neighbors and gave a trust based algorithm.

III. METHODS

The main goal is to define a template to generate QLA code by combining aggregation and transposition. The proposal has two perspectives such as to evaluate efficient aggregations and perform query optimization. The first one includes the following approaches, LCM ing, transposition and cross-tabulation. LCM ing approach is a built-in method in a commercial DBMS. It can help evaluating an aggregated tabular format for summarized data set. It perform the following steps, The LCM ing method is used to write cross-tabulation queries that rotate rows into columns, aggregating data in the process of the rotation. The output of a LCM operation typically includes more columns and fewer rows than the starting data set. The LCM computes the aggregation functions specified at the beginning of the clause. Aggregation functions must specify a GROUP BY clause to return multiple values; the LCM performs an implicit GROUP BY. New columns corresponding to values in the LCM , each aggregated value is transposed to the appropriate new column in the cross-tabulation. The subclauses of the LCM have the following semantics: expr - specify an expression that evaluates to a constant value of a LCM column. Subquery – to specify a subquery, all values found by the subquery are used for LCM ing. The subquery must return a list of unique values at the execution time of the LCM query.

Comparing Evaluation Methods

On the other hand, the second important issue is automatically generating unique column names. If there are many sub grouping columns R_1; . . .;R_k or columns are of string data types, this may lead to generate very long column names, which may exceed DBMS limits. However, these are not important limitations because if there are many dimensions that is likely to correspond to a sparse matrix (having many zeroes or nulls) on which it will be difficult or impossible to compute a data mining model. On the other hand, the large column name length can be solved as explained below. The problem of d going beyond the maximum number of columns can be solved by vertically partitioning FH so that each partition table does not exceed the maximum number of columns allowed by the DBMS. Evidently, each partition table must have L_1; . . . ; L_j as its primary key. Alternatively, the column name length issue can be solved by generating column identifiers with integers and creating a "dimension" description table that maps identifiers to full descriptions, but the meaning of each dimension is lost. An alternative is the use of abbreviations, which may require manual input.

IV. RESULT ANALYSIS

Query Optimizations

Our first query optimization, applied to three methods. Our goal is to assess the acceleration obtained by precomputing a cube and storing it on F_V . We can see this optimization uniformly accelerates all methods. This optimization provides a different gain, depending on the method: for SPJ the optimization is best for small n, for for large n and for CASE there is rather a less dramatic improvement all across n. It is noteworthy LCM is accelerated

by our optimization, despite the fact it is handled by the query optimizer. Since this optimization produces significant acceleration for the three methods (at least 2 faster) we will use it by default. Notice that pre-computing F_V takes the same time within each method. Therefore, comparisons are fair. We now evaluate an optimization specific to the LCM operator. This LCM optimization is well known, as we learned from QLA Server DBMS users groups. shows the impact of removing (trimming) columns not needed by LCM . That is, removing columns that will not appear in F_H. We can see the impact is significant, accelerating evaluation time from three to five times. All our experiments incorporate this optimization by default.

Time Complexity

We now verify the time complexity analysis given in We plot time complexity keeping varying one parameter and the remaining parameters fixed. In these experiments, we generated synthetic data sets similar to the fact table of TPC-H of different sizes with grouping columns of varying selectivities (number of distinct values). We consider two basic probabilistic distribution of values: uniform (unskewed) and zipf (skewed). The uniform distribution is the distribution used by default.

V. CONCLUSION

The proposed approaches implements an abstract but minimal extension to QLA standard aggregate functions to compute efficient summarized data set which just requires specifying sub grouping columns inside the aggregation function call. From a query optimization perspective, The proposed system describes the possibility of extending QLA OLAP aggregations with horizontal layout capabilities. Aggregation recommendations produce tables with fewer rows, but with more columns. The aggregated tables are useful to create data sets with a horizontal layout, as commonly required by data mining algorithms and OLAP cross-tabulation. The output of a query optimization can immediately be applied back to the data gathering, transformation, and analysis processes. Anomalous data can be detected in existing data sets, and new data entry can be validated in real time, based on the existing data. QLA Server Data Mining contains multiple algorithms that can perform churn analysis based on historical data. Each of these algorithms will provide a probability. In future, research issues is proposed on extending QLA code for data mining processing. The related work on query optimization is proposed and compared to aggregation recommendations with alternative proposals to perform transposition or LCM ing. It includes to develop more complete I/O cost models for cost-based query optimization and to study optimization of aggregation recommendations processed in parallel in a shared-nothing DBMS architecture.

VI. REFERENCES

[1] G. Adomavicius and A. Tuzhilin, “Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions,” IEEE Trans. Knowledge and Data Eng., vol. 17, no. 6, pp. 734-749, June 2005.

[2] C. Anderson, The Long Tail. Hyperion, 2006.

[3] M. Balabanovic and Y. Shoham, “Fab: Content-Based, Collaborative Recommendation,” Comm. ACM, vol. 40, no. 3, pp. 66-72, 1997.

[4] R. Bell, Y. Koren, and C. Volinsky, “The BellKor Solution to the Netflix Prize,” www.netflixprize.com/assets/ProgressPrize2007_KorBell.pdf, 2007.

[5] R.M. Bell, Y. Koren, and C. Volinsky, “The Bellkor 2008 Solution to the Netflix Prize,” http://www.research.att.com/~volinsky/netflix/ProgressPrize2008BellKorSolution.pdf, 2008.

[6] J. Bennett and S. Lanning, “The Netflix Prize,” Proc. KDD-Cup and Workshop at the 13th ACM SIGKDD Int’l Conf. Knowledge and Data Mining, 2007.

[7] D. Billsus and M. Pazzani, “Learning Collaborative Information Filters,” Proc. Int’l Conf. Machine Learning, 1998.

[8] K. Bradley and B. Smyth, “Improving Recommendation Diversity,” Proc. 12th Irish Conf. Artificial Intelligence and Cognitive Science, 2001.

[9] J. Canny, “Collaborative Filtering with Privacy via Factor Analysis,” SIGIR ’07: Proc. 25th Ann. Int’l ACM SIGIR Conf. Research and Development in Information Retrieval, pp. 238-245, 2002.

[10] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li, “Context-Aware Query Suggestion by Mining Click-Through and Session Data,” KDD ’08: Proc. 14th ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining, pp. 875-883, 2008.

[11] P.A. Chirita, C.S. Firan, and W. Nejdl, “Personalized Query Expansion for the Web,” SIGIR ’07: Proc. 30th Ann. Int’l ACM SIGIR Conf. Research and Development in Information Retrieval, pp. 7-14, 2007.

[12] N. Craswell and M. Szummer, “Random Walks on the Click Graph,” SIGIR ’07: Proc. 30th Ann. Int’l ACM SIGIR Conf. Research and Development in Information Retrieval, pp. 239-246, 2007.

L.Gomathi received her BCA degree from university of Amman Arts & Science College and MCA degree from Bharathidasan University. She has completed her M.Phil at Periyar University. She is having 7 Yrs of experience in collegiate teaching and She is a Head of the department of computer applications in Muthayammal college of Arts and Science affiliated by Periyar University.

K.Ramya, received her B.com, degree in Muthayammal College of Arts&science from Periyar University, Salem. Then receive her MCA, MCA, degree in Gnanamani College of Technology from Anna University, Chennai. Her Area of interest is Data Mining.