American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-6, Issue-3, pp-128-130
www.ajer.org
Research Paper Open Access

Preference Aware Automatic Search Using Multiple Taxonomies

Geethanjali.A[1], Sharmila.L[2], Prasath.R[3]
[1]*P.G Student, Dept of ME Computer Science and Engineering, Alpha College of Engineering, Chennai, Tamilnadu, India*
[2]*Assistant Professor, Dept of ME Computer Science and Engineering, Alpha College of Engineering,Chennai, Tamilnadu, India.*
[3] *Assistant Professor, Dept of Information Technology, Alpha College of Engineering, Chennai, Tamilnadu, India*

ABSTRACT: *The project aims on frequent itemset mining, which focuses on discovering the different correlations among data. The itemsets that became rare are no longer extracted using frequent generalized itemsets from a certain point. Frequent generalized itemsets includes itemsets that are frequently occurring in source data and itemsets that provide a top level abstraction of the knowledge that is mined. The discovery of relevant data occurrences and their most significant temporal trends are becoming very essential and important research area. In different application contexts the correlation among data has been found out by the application of frequent itemset mining and association rule extraction algorithms. Here provides several query optimization strategies for extended queries and describes an algorithm which includes query execution with performance evaluation while making use of native query engine.*

I. INTRODUCTION

Data mining, which is used for the extraction of hidden information from large databases, is a powerful technology with greater potential to help the company's focus on the important information in their data warehouses. Companies collect and refine massive quantities of data to analyze future trends and behaviors, which allows businesses to make knowledge-driven decisions. Data mining techniques can be implemented efficiently on existing software and hardware platforms. It enhances the value of existing information resources, and thereby integrating with new products and systems. Most data are unstructured and hence it takes some process and methods to extract the needful information from the data and transform it into usable and understandable form. Plenty of tools are available for data mining using machine learning, artificial intelligence and other techniques to extract data.

Data mining and data warehousing

A complete and consistent storage of data obtained from a variety of sources made available to end users in the data mining and warehousing concept. Initially, the data should be extracted from a group of data and later into a database .data is already part of a data warehouse. If the data has already been cleaned for a data warehouse, then most likely it will not need further cleaning in order to be mined.

Pattern recognition

Data mining takes advantage of artificial intelligence (AI) and statistics. Both disciplines have been working on problems of pattern recognition and classification. The main aim of this project is to blend preference evaluation with query execution by several query optimization strategies that are used for extended query plans. Here describes a query execution algorithm that comes up with preference evaluation with query execution, making effective use of the native query engine. Here defines a reference using a condition on the tuples affected, a scoring function that scores these tuples, and a confidence that shows how confident these scores are been. In this data model, tuples carry scores with confidences. Our algebra comprises the standard relational operators extended to handle scores and confidences.

For example, the join operator will join two tuples and compute a new score-confidence pair. This is done by combining the scores and confidences that come with the two tuples. As an addition, the algebra contains a new operator, prefer, that evaluates a preference on a relation. That means, given as inputs a relation and a preference on this relation prefer outputs the relation with new scores and confidences.

During the preference evaluation process, both the conditional as well as the scoring part of the preference are used. The conditional part acts as a 'soft' constraint that determines which tuples are scored. This is being done without disqualifying any of the tuples from the query result. Discovering relevant data recurrences and their most significant temporal trends is becoming an increasingly appealing research topic. The application of frequent itemset mining and association rule extraction algorithms that can be used to discover valuable correlations among data has been investigated in a number of different application contexts.

EXISTING SYSTEM

To integrate preferences into database queries, several approaches have been introduced. These can be mainly divided into two categories. Plug in approaches and native approaches. Plug in approaches operate on top of the database engine while translating preferences into queries. Native approaches focus on specific queries such as top-k or skyline queries by incorporating new operators inside the database engine. Due to this these methods can be used only for one type of query. So these approaches are not suitable for flexible processing of queries with the help of preferences.

PROPOSED SYSTEM

In this paper, first an extended query plan is constructed that contains all operators that comprise a query and optimize it. Then, for processing the optimized query plan, the general strategy is to blend query execution with preference evaluation and taking maximum advantage of the native query engine to process parts of the query that do not involve a prefer operator. If a query is given with preferences, the goal of query optimization is to minimize the cost related with preference evaluation. Based on the algebraic properties of prefer, a set of heuristic rules are being applied, aiming to minimize the number of tuples that are given as input to the prefer operators. Further provides a cost-based query optimization approach. Using the output plan of the first step as a skeleton and a cost model for preference evaluation, the query optimizer calculates the costs of alternative plans that interleave preference evaluation and query processing in different ways. Two plan enumeration methods, i.e., a dynamic programming and a greedy algorithm are proposed.

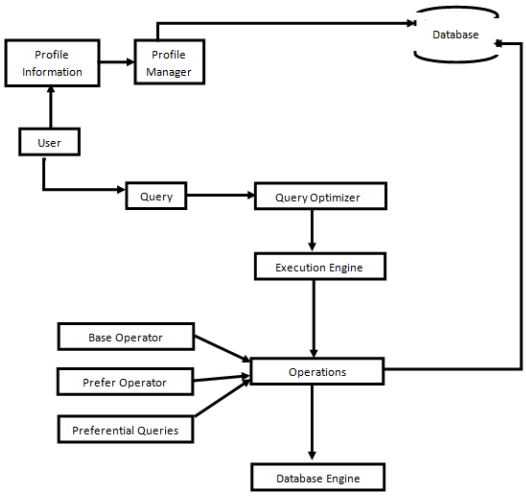

Fig: Architecture Diagram

An extended relational model has been used here, which include prefer operators, preferential queries, base operators etc. The prefer operator augment the standard relational algebra with a special prefer operator. Prefer evaluates a preference on a relation using an aggregate function. The base operators like Select, Project, Intersection, Union, Difference, Inner join etc. are being used inside this. In order to improve the database tuples, with preference scores and confidences, p-relations are defined.

Then the preferential query combines all these and returns a set of tuples. These tuples satisfy the scores and confidence values and Boolean query conditions.In the query processing, the query parser adds a prefer operator for each preference. Finally, the query parser checks for each preference, whether it involves an attribute (either in the conditional or the scoring part) that does not appear in the query and modifies project operators, such that these attributes will be projected as well.The query parser initially constructs a baseline extended query plan keeping the order of the operators as defined in the user query. Next, it creates additional joins with relations that do not appear in the SQL part of the query but have associated preferences.

Extended relational operators and the prefer operator do not change how tuples are filtered or joined; for instance, prefer operator does not filter any tuples. Therefore the extended relational operators do not affect the non-preference related cost.Thus, one can expect that the join order that is suggested by the native query optimizer for a query if no prefer operators were present, will still yield good performance for the non-preference part of the same query with the prefer operators. Based on this observation, will keep the suggested join order and will consider the non-preference related cost as fixed. Then, the goal of query optimizer will be to minimize the cost related with preference evaluation.

The most critical parameter that shapes the processing cost of query evaluation is the disk I/Os, which is proportional to the number of tuples flowing through the operators in the query plan. Assuming a fixed position for the other operators, the goal of query optimizer is essentially to place the prefer operators inside the plan, such that the number of tuples flowing through the score tables is minimized. The execution engine is responsible for processing a preferential query and supports various algorithms. Baseline: A postorder traversal is performed to execute each operator directly following the execution plan. Group Bottom-up algorithm is generally used for this. An extension of the Group Bottom-Up algorithm is used here in order to make the queries more optimized.

II. CONCLUSION

A cost based query optimization is being implemented and effective query processing methods are developed. The proposed framework can efficiently handle different types of preferential queries, closer to database and non obstructive to database engine.

REFERENCES

[1]. P.-N. Tan, V. Kumar, and J. Srivastava, "Selecting the Right Interestingness Measure for Association Patterns," Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining July 2002.

[2]. AnastasiosArvanitis and Georgia Koutrika,"PrefDB: Supporting Preferences as First- Class Citizens in Relational Databases" /IEEE Transactions on knowledge and data engineering 2014

[3]. R. Agrawal and R. Srikant, "Mining Generalized Association Rules," Proc. 21th Int'l Conf. Very Large Data Bases (VLDB '95), pp. 407-419, 1995.

[4]. S.C. Gates, W. Teiken, and K.-S.F. Cheng, "Taxonomies by the Numbers: Building High-Performance Taxonomies," Proc. 14th ACM Int'l Conf. Information pp. 568-577, 2005.

[5]. E. Baralis, L. Cagliero, T. Cerquitelli, V. D'Elia, and P. Garza,"Support Driven Opportunistic Aggregation for Generalized Itemset Extraction," Proc. IEEE Fifth Int'l Conf. Intelligent Systems (IS '10), 2010.

[6]. K. Stefanidis, E. Pitoura, and P. Vassiliadis. Adding context to preferences. In ICDE, 2007.

[7]. P. Georgiadis, I. Kapantaidakis, V. Christophides, E. M. Nguer, and N. Spyratos. Efficient rewriting algorithms for preference queries.In ICDE, 2008.

[8]. Nizar R. Mabroukeh and C. I. Ezeife ,"Taxonomy of Sequential Pattern Mining Algorithms."ACM Computing Surveys, Vol. 43, No. 1, Article 3, Publication date: November 2010.

[9]. NingZhong, Yuefeng Li, and Sheng-Tang Wu," Effective Pattern Discovery for Text Mining," Ieee Transactions On Knowledge And Data Engineering, Vol. 24, No. 1, January 2012.

[10]. J. Han and Y. Fu, "Mining Multiple-Level Association Rules in Large Databases," IEEE Trans. Knowledge and Data Eng., vol. 11,no. 7, Sept. 1999.

[11]. B. Shen, M. Yao, Z. Wu, and Y. Gao, "Mining Dynamic Association Rules with Comments," Knowledge and Information Systems, vol. 23, Apr. 2010.

[12]. P. Giannikopoulos, I. Varlamis, and M. Eirinaki, "Mining Frequent Generalized Patterns for Web Personalization in the Presence of Taxonomies," Int'l J. Data Warehousing and Mining, vol. 6, no. 1, 2010.

[13]. Agrawal, T. Imielinski, and A. Swami, "Mining Association rules between Sets of Items in large Databases," ACM SIGMOD Record, vol. 22, pp. 207-216, 1993.