

POSTER ABSTRACT

SQL Based Frequent Pattern Mining without Candidate Generation

Xuequn Shang
Department of Computer
Science
University of Magdeburg
P.O.BOX 4120, 39106
Magdeburg, Germany
shang@iti.cs.uni-
magdeburg.de

Kai Uwe Sattler
Department of Computer
Science
University of Magdeburg
P.O.BOX 4120, 39106
Magdeburg, Germany
kus@iti.cs.uni-
magdeburg.de

Ingolf Geist
Department of Computer
Science
University of Magdeburg
P.O.BOX 4120, 39106
Magdeburg, Germany
geist@iti.cs.uni-
magdeburg.de

ABSTRACT

Scalable data mining in large databases is one of today's real challenges to database research area. The integration of data mining with database systems is an essential component for any successful large-scale data mining application. A fundamental component in data mining tasks is finding frequent patterns in a given dataset. Most of the previous studies adopt an Apriori-like candidate set generation-and-test approach. However, candidate set generation is still costly, especially when there exist prolific patterns and/or long patterns. In this study we present an evaluation of SQL based frequent pattern mining with a novel frequent pattern growth (FP-growth) method, which is efficient and scalable for mining both long and short patterns without candidate generation. We examine some techniques to improve performance. In addition, we have made performance evaluation on commercial DBMS (IBM DB2 UDB EEE V8).

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Applications—*Data Mining*

General Terms

SQL based mining algorithms

Keywords

Data mining, frequent pattern mining, database mining, mining algorithms in SQL

1. INTRODUCTION

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC '04, March 14-17, 2004, Nicosia, Cyprus
Copyright 2004 ACM 1-58113-812-1/03/04 ...\$5.00.

The integration of data mining with database systems is an emergent trend in database research and development area. This is particularly driven by explosion of the data amount stored in databases such as Data Warehouses during recent years, and database systems provide powerful mechanisms for accessing, filtering, indexing data and SQL parallelization. In addition, SQL-aware data mining systems have the ability to support ad-hoc mining, ie., allowing to mine arbitrary query results from multiple abstract layers of database systems or Data Warehouses. Mining frequent patterns in transaction databases has been studied popularly in data mining research. Most of the previous studies adopt an Apriori-like candidate set generation-and-test approach [3, 7, 8], which is based on an anti-monotone Apriori heuristic: if any length k pattern is not frequent in the database, its length $(k+1)$ super-pattern can never be frequent. The above Apriori heuristic achieves good performance gain by reducing significantly the size of candidate sets. However, in situations with prolific frequent patterns, long patterns, or quite low minimum support thresholds, this kind of algorithm may still suffer from the following two nontrivial costs:

1. It is costly to handle a huge number of candidate frequent itemsets.
2. It is tedious to repeatedly scan the database and check a large set of candidates by pattern matching, which is especially true for mining long patterns.

Recently, an FP-tree based frequent pattern mining method [4], called FP-growth, developed by Han et al achieves high efficiency, compared with Apriori-like approach. The FP-growth method adopts the divide-and-conquer strategy, uses only two full I/O scans of the database, and avoids iterative candidate generation. There are some SQL based approaches proposed to mine frequent patterns [9, 12], but they are on the base of Apriori-like approach. This fact motivated us to examine if we can get sufficient performance by the utilization of SQL based frequent pattern mining using FP-growth-like approach. We propose mining algorithms based on FP-growth to work on DBMS and compare the performance of these approaches using synthetic datasets.

2. FREQUENT PATTERN MINING IN SQL BASED ON FP-GROWTH

In [4], frequent pattern mining consists of two steps:

1. Construct a compact data structure, frequent pattern tree (FP-tree), which can store more information in less space.
2. Develop an FP-tree based pattern growth (FP-growth) method to uncover all frequent patterns recursively.

Although an FP-tree is rather compact, it is unrealistic to construct a main memory-based FP-tree when the database is large. However using RDBMSs provides us the benefits of using their buffer management systems specifically developed for freeing the user/applications from the size considerations of the data. And moreover, there are several potential advantages of building mining algorithms to work on RDBMSs. An interesting alternative is to store a FP-tree in a table. According to the properties of FP-tree, we represent an FP-tree by a table *FP* with three column attributes: item identifier (*item*), the number of transactions that contain this item in a sub-path (*count*), and item prefix subtree (*path*). The field *path* is beneficial not only to construct the table *FP* but also to find all frequent patterns from *FP*. We studied two approaches in this category - FP, CanFP. They are different in the construction of frequent pattern tree table, named FP. FP approach checks each frequent item whether it should be inserted into a table FP or not one by one to construct FP. CanFP approach introduces a temporary table CanFP, thus table FP can generate from CanFP.

3. SUMMARY AND CONCLUSIONS

We have implemented SQL based frequent pattern mining using FP-growth-like approach. We represent FP-tree using a relational table FP and proposed a method to construct this table. To improve its performance, a table called CanFP is introduced, which is in fact stores all information of frequent itemsets and their prefix path in each transaction. And then, table FP can derived from table CanFP. Compare to the construction of FP, the process of the construction of CanFP avoid testing each frequent item one by one. We next experimented with two approaches that made use of the object-relational extensions like table function. The test results show that improved SQL based frequent pattern mining approach using FP-growth can get sufficient performance.

There remain lots of further investigations. We plan to implement our SQL based frequent pattern mining approach on commercial parallel RDBMS, and to check how efficiently our approach can be parallelized and seeded up using parallel database system.

4. REFERENCES

- [1] R. Agarwal, C. Aggarwal, and V. Prasad. A tree projection algorithm for generation of frequent itemsets. *Journal of Parallel and Distributed Computing*(Special Issue on High Performance Data Mining), 2000.
- [2] R. Agrawal and K. Shim. Developing tightly-coupled data mining application on a relational database system. In *Proc.of the 2nd Int. Conf. on Knowledge Discovery in Database and Data Mining*, Portland,Oregon, 1996.
- [3] R. Agrawal, R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20st VLDB Conference*, Santiago, Chile, pp.487-499, 1994.
- [4] J. Han, J. pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proc. of the ACM SIGMOD Conference on Management of data*, 2000.
- [5] J. Han, Y. Fu, W. Wang, K. Koperski, and O. Zaiane. DMQL: A data mining query language for relational database. In *Proc. Of the 1996 SIGMOD workshop on research issues on data mining and knowledge discovery*, Montreal, Canada, 1996.
- [6] M. Houtsma and A. Swami. Set-oriented data mining in relational databases. *DKE*, 17(3): 245-262, December 1995.
- [7] R. Meo, G. Psaila, and S. Ceri. A new SQL like operator for mining association rules. In *Proc. Of the 22nd Int. Conf. on Very Large Databases*, Bombay, India, 1996.
- [8] J. S. Park, M. Chen, and P. S. Yu. An effective hash based algorithm for mining association rules. In *Proc. of the ACM SIGMOD Conference on Management of data*, pp.175-186, 1995.
- [9] I. Pramudiono, T. Shintani, T. Tamura and M. Kitsuregawa. Parallel SQL based associaton rule mining on large scale PC cluster: performance comparison with directly coded C implementation. In *Proc. Of Third Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, 1999.
- [10] R. Rantzaeu. Processing frequent itemset discovery queries by division and set containment join operators. *DMKD03: 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2003.
- [11] A. Savsere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. In *Proc. of the 21st VLDB Conference*, 1995.
- [12] S. Sarawagi, S. Thomas, and R. Agrawal. Integrating mining with relational database systems: alternatives and implications. In *Proc. of the ACM SIGMOD Conference on Management of data*, Seattle,Washinton,USA, 1998.
- [13] K. Sattel and O. Dunemann. SQL database primitives for decision tree classifiers. In *Proc. Of the 10nd ACM CIKN Int. Conf. on Information and Knowledge Management*, Atlanta,Georgia, 2001.
- [14] S. Thomas and S. Chakravarthy. Performance evaluation and optimization of join queries for association rule mining. In *Proc. DaWaK*, Florence, Italy, 1999.
- [15] H. Wang and C. Zaniolo. Using SQL to build new aggregates and extenders for Object-Relational systems. In *Proc. Of the 26th Int. Conf. on Very Large Databases*, Cairo,Egypt, 2000.
- [16] T. Yoshizawa, I. Pramudiono, and M. Kitsuregawa. SQL based association rule mining using commercial RDBMS (IBM DB2 UDB EEE). In *Proc. DaWaK*, London, UK, 2000.

L. Lamport, *TEX: A Document Preparation System*. Redwood, CA: Addison-Wesley, 1986.