

Predator-Miner: Ad hoc Mining of Associations Rules Within a Database Management System

Wee Hyong Tok, Twee Hee Ong, Wai Lup Low, Indriyati Atmosukarto, Stéphane Bressan
School of Computing
National University of Singapore
{tokweehy,ongtweeh,lowwailu,indriyat,steph}@comp.nus.edu.sg

Abstract

In this demonstration, we present a prototype system, Predator-Miner, which extends Predator with an relational-like association rule mining operator to support data mining operations. Predator-Miner allows a user to combine association rule mining queries with SQL queries. This approach towards tight integration differs from existing techniques of using user-defined functions (UDFs), stored procedures, or re-expressing a mining query as several SQL queries in two aspects. First, by encapsulating the task of association rule mining in a relational operator, we allow association rule mining to be considered as part of the query plan, on which query optimization can be performed on the mining query holistically. Second, by integrating it as a relational operator, we can leverage on the mature field of relational database technology.

We extend Predator to support a variant of DMQL, and allow SQL and DMQL to be intermixed in a query. We also demonstrate a cost-based mining query optimization framework.

1. Introduction

The amount of data being stored in databases increases rapidly in many organizations. These nuggets of data are often used by the organizations in data analysis tasks, which aim to discover trends and predictive models within the data. However, most existing data mining algorithms either access the data stored in pre-processed flat files, or use an interface such as Open Database Connectivity (ODBC)/Java Database Connectivity (JDBC) to access a relational database. The loose coupling between the data mining system and the underlying data has a negative impact on the overall performance of the data mining operation, due to the additional interfacing overheads. Many

researchers have focused on developing novel data structure and algorithms to facilitate fast discovery of association rules. These techniques cannot leverage the years of work by the database community in the areas of query optimization, query processing and indexing.

2. Ad Hoc Mining

We define ad hoc data mining to be a flexible and interactive data mining process performed over a sub-set of the data without the need for data pre-processing. The motivation for ad hoc data mining is to allow the end-user to get a quick analysis of the results from the mining process prior to performing mining on the entire dataset, and to reduce the hassle of a pre-processing step.

A main attraction of SQL lies in its usage flexibility and inter-operability. SQL allows a user to express a query, without the need to specify how the query is actually processed. The way in which the query is processed is determined by the query optimizer, which chooses a optimal plan, amongst several other execution plans, that would be used to process the query. We use a combination of SQL and DMQL [3] to give the user the flexibility of SQL as well as support for expressing mining queries. We choose DMQL for its simplicity and similarity to SQL. DMQL extends SQL with the syntax to support mining of characteristics rules, discriminant rules, classification rules and association rules. In addition, to allow the user to 'mix-and-match' relational and mining queries, Predator-Miner support nested queries. We believe this will provide greater flexibility in the expressing queries.

In Figure 1, we show an example of a nested ad hoc mining query, where the inner relational query first selects a subset of the data from the table *trans* which satisfies the selection criteria. It proceeds to mine association rules from this subset with a support threshold of 0.05 and confidence threshold of 0.7.

```

Find association rules
related to item
with key transaction_id
from (
  select * from trans
  where price > 15
  and price < 100 ) trans_new
with support threshold = 0.05
with confidence threshold = 0.7

```

Figure 1. Example of the DMQL variant used in Predator-Miner

Most organizations store their data in a normalized 'vertical' format to reduce data redundancy. However, most association rule mining operations requires its input to be in a 'horizontal' format. Most existing techniques require a pre-processing step to convert data from the vertical format to the horizontal format. Our system uses the vertical format, which is more natural and closer to how normalized transaction data is stored, thus avoiding the hassle of transforming the data to the horizontal format.

3. System Design

Predator [5] is a client-server OR-DBMS, built on top of the Shore [1] Storage manager. In our work, we extend Predator's SQL parser to support a variant of DMQL using Bison/Flex [2, 4]. Predator's query optimizer is also extended to support mining queries. Lastly, a new relational-like association rule mining operator is introduced to Predator's query engine. Figure 2 shows the system architecture for Predator-Miner.

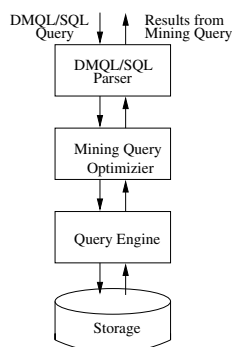


Figure 2. System Architecture for Predator-Miner

4. Structure of the Demo

In this demonstration, we illustrate the novelty of the proposed system in the following ways:

1. Tight integration of association rule mining operation within a relational database - Predator's library of relational operators is extended with a new association rule-mining relational operator. We demonstrate the novelty of this new relational association rule mining operator.
2. Ad Hoc Association Rule Mining - We will demonstrate how ad hoc association rule mining alleviates the user from pre-processing the data from a vertical format to a horizontal format. In addition, we will show how ad hoc mining can allow the user to mine interactively from a subset of the dataset to get a 'feel' of the association rules generated.
3. Integration of Association Rule mining queries and relational queries - We will demonstrate nested queries consisting of a combination of SQL and DMQL. A relational query will be used as an inner query to retrieve a subset of the dataset, which is pipelined to the association rule mining operator.

5. Acknowledgements

We thank Philippe Bonnet for providing us with the opportunity to learn about Predator and Cougar during the summer, which made this work possible. We thank Bing Liu for his comments on the initial work.

References

- [1] M. J. Carey, D. J. DeWitt, M. J. Franklin, N. E. Hall, M. L. McAuliffe, J. F. Naughton, D. T. Schuh, M. H. Solomon, C. K. Tan, O. G. Tsatalos, S. J. White, and M. J. Zwilling. Shoring up persistent applications. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 383-394, 1994.
- [2] R. S. Charles Donnelly. *Bison, the yacc-compatible parser generator*. <http://www.gnu.org/manual/bison>.
- [3] J. Han, Y. Fu, W. Wang, K. Koperski, and O. Zaiane. DMQL: A data mining query language for relational databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 1996.
- [4] V. Paxson. *Flex - A Fast scanner generator*. <http://www.gnu.org/manual/flex>.
- [5] P. Seshadri, M. Livny, and R. Ramakrishnan. The case for enhanced abstract data types. *Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, pages 66-75, August 1997.