March - April 2020
ISSN: 0193-4120 Page No. 2038 - 2042

IMINE: Index Support for Item Set Mining

Vemula Rohini [1], V.Sujatha [2]
[1] Assistant professor in Computer Science and Engineeering department at Dr. K V.Subba Reddy College of Engineering for Women, Kurnool
[2] Assistant professor in Computer Science and Engineeering department at Dr. K V.Subba Reddy College of Engineering for Women, Kurnool

Article Info
Volume 83
Page Number: 2038 - 2042
Publication Issue:
March - April 2020

Article History
Article Received*: 24 July 2019*
Revised*: 12 September 2019*
Accepted*: 15 February 2020*
Publication*: 18 March 2020*

Abstract
The IMine index, a general and compact structure which provides tight integration of item set extraction during a relational DBMS. IMine provides an entire representation of the first database, Since no constraint is enforced during the index creation phase. To reduce the I/O cost, data accessed together during an equivalent extraction phase are clustered on an equivalent disk block. The IMine index structure are often efficiently exploited by different item set extraction algorithms. IMine data access method supports the LCM v.2 algorithms and FP-growth, but they will straightforwardly support the enforcement of varied constraint categories. It has been integrated into the Postgre SQL DBMS and exploits its physical level access methods. Experiments, run both sparse and dense data distributions, show the efficiency of the proposed index and its linear scalability also for giant data sets.Item set mining supported by the IMine index shows performance always comparable , and sometimes (especially for low supports) better than, state-of-the-art algorithms accessing data on file.

Keywords: *Data mining, FP-Growth, LCM v.2.*

1. INTRODUCTION:

With the wide use of computers, scanners and data base technique, human accumulated an excellent deal of historical data. These data look simple at the surface of them, but, there's much valuable information behind them. business decision and resource management, the knowledge and rule behind these data are very useful in data prediction. But, if we still use traditional methods of statistical and analyses, these useful information can't be discovered or are often found in infinite time. Hence data processing has been proposed on this occasion. As one of the most research patterns within the field of knowledge mining, association rules are wont to determine the relationships of a group of item, to seek out out valuable information. Frequent item mining, the main task of the association rule mining, is the efficiency of which is the difficult problem. Relevant knowledge of frequent item set mining is introduced and some classic algorithms are analyzed in detail. For the maximum frequent contains all the frequent item sets, in this focuses on how to mining maximum frequent item sets, the maximum frequent mining from generating FP-tree, the prune strategy, superset checking, first searching strategy, reducing dimension are deeply researched.

Many research attempts have been made in maintaining the reliability which is conducted from the past decades, also its efficiency is less. If the database size gets increased its performance may degrade. In these models more number of scans are required to retrieve the data and it increases the complexity.

2. PROPOSED METHODOLOGY

2.1 Proposed Work:

March - April 2020
ISSN: 0193-4120 Page No. 2038 - 2042

In this section we are going to propose the techniques which we are going to use in the system and we consider example data set to illustrate how the techniques can be implemented .and the techniques are as follows:

a) Frequent Item Set Extraction.

b) I Tree Module.

c) I B Tree Module.

The transactional data set D is represented, in the relational model, as a relation R. Each tuple in R is a pair (TransactionID, ItemID). The IMine index provides a compact and complete representation of R. Hence, it allows the efficient extraction of item sets from R, possibly enforcing support or other constraints.

TID	ItemsID
1	g,b,h,e,p,v,d
2	e,m,h,n,d,b
3	p,e,c,i,f,o,h
4	j,h,k,a,w,e
5	n,b,d,e,h
6	s,a,n,r,b,u,i
7	b,g,h,d,e,p
8	a,i,b
9	f,i,e,p,c,h
10	t,h,a,e,b,r
11	a,r,e,b,h
12	z,b,i,a,n,r
13	b,e,d,p,h

Table 2.1: Example Data Set

2.1.1 Frequent Item Set Extraction:

On the IMine index Frequent item set extraction takes place. We are presenting two approaches, FP-based and LCM-based algorithms, which are an adaptation of the FP-Growth algorithm and LCM v.2 algorithm, respectively.

FP-based algorithm:

The FP-growth algorithm stores the info during a prefix-tree structure called FP-tree. First, it computes item support. Then, for every transaction, it stores within the FP-tree its subset including frequent items. Items are considered one by one. For each item, extraction takes place on the frequent-item projected database, which is generated from the first FP-tree and represented during a FP-tree based structure.

LCM-based algorithm:

The LCM v.2 algorithm loads in memory the support-based projection of the first database. First, it reads the transactions to count item support. Then, for every transaction, it loads the subset including frequent items. Data are represented in memory by means of an array-based arrangement , on which the extraction takes place.

2.2 Modules:

We use two modules I Tree module and IB Tree module for efficient extraction of item sets from the database.

2.2.1 I-Tree Module:

The Item set-Tree (I-Tree) may be a prefix-tree which represents relation R by means of a succinct and lossless compact structure. Implementation of the I-Tree is predicated on the FP-tree arrangement , which is extremely effective in providing a compact and lossless representation of relation R. However, since the two index components are designed to be independent, alternative I-Tree data structures can be easily integrated in the IMine index as shown in figure.

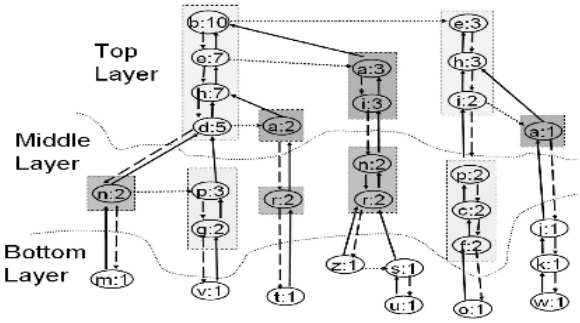


Fig 2.1: IMINE Index for Example data set of I-Tree

March - April 2020
ISSN: 0193-4120 Page No. 2038 - 2042

The I-Tree associated to relation R is really a forest of prefix-trees, where each tree represents a gaggle of transactions all sharing one or more items. Each node within the I-Tree corresponds to an item in R. Each path within the I-Tree is an ordered sequence of nodes and represents one or more transactions in R. Each item in relation R is associated to at least one or more I-Tree nodes and every transaction in R is represented by a singular I-Tree path.

2.2.2 I-BTree Module

The Item-Btree (I-Btree) may be a B+Tree structure which allows reading selected I-Tree portions during the extraction task. For each item, it stores the physical locations of all item occurrences within the I-Tree. It supports efficiently loading from the I-Tree. I-Btree allows selectively accessing the I-Tree disk blocks during the extraction process as within the Fig.2.2.2. It is based on a B+Tree structure. For each item i in relation R, there's one entry within the I-Btree.

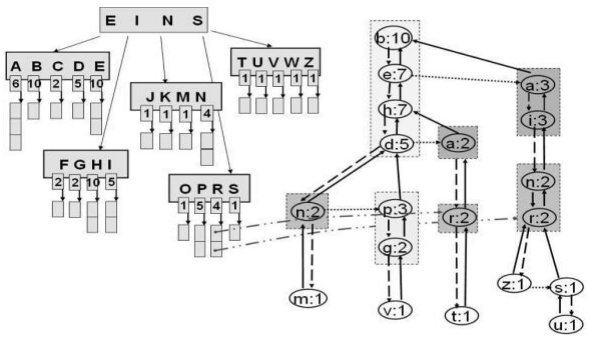
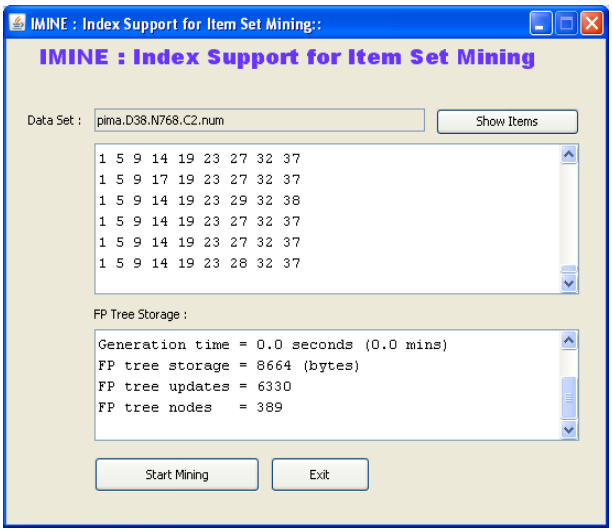


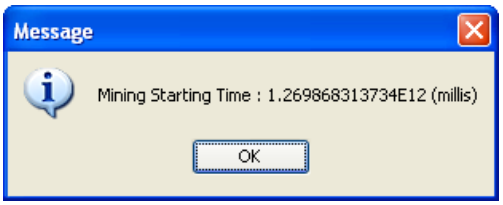
Fig 2.2: IMINE Index for Example data set of I-BTree

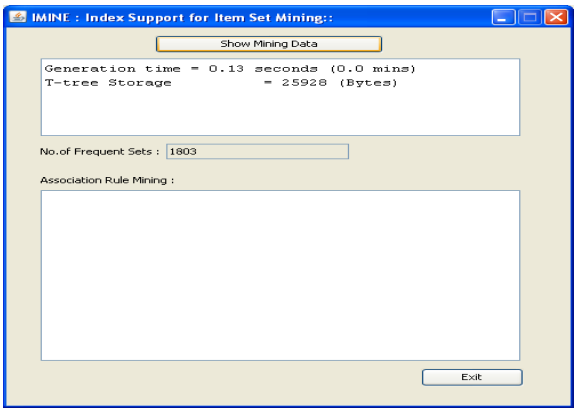
I-Tree associated to R is a forest of prefix-trees, where each tree represents a group of transactions all sharing one or more items. Every node in the I-Tree corresponds to an item in R. Each path within the I-Tree is an ordered sequence of nodes and represents one or more transactions in R. Each item in relation R is associated to at least one or more I-Tree nodes and every transaction in R is represented by a singular I-Tree path.

3. RESULTS AND DISCUSSION:

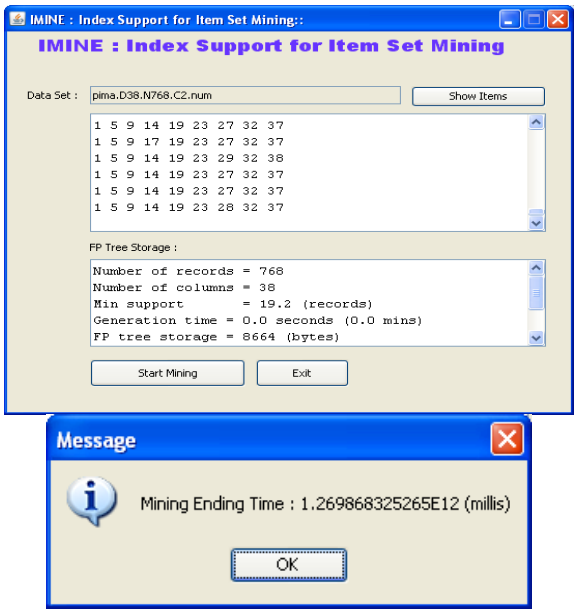
We are uploading a data set for mining, when we start mining the data set it shows number of records, columns, Min Support, Generation time, FP tree storage. On this data set we are applying association rule mining. Finally by using Association rule mining the accuracy was displayed and total mining duration also given .



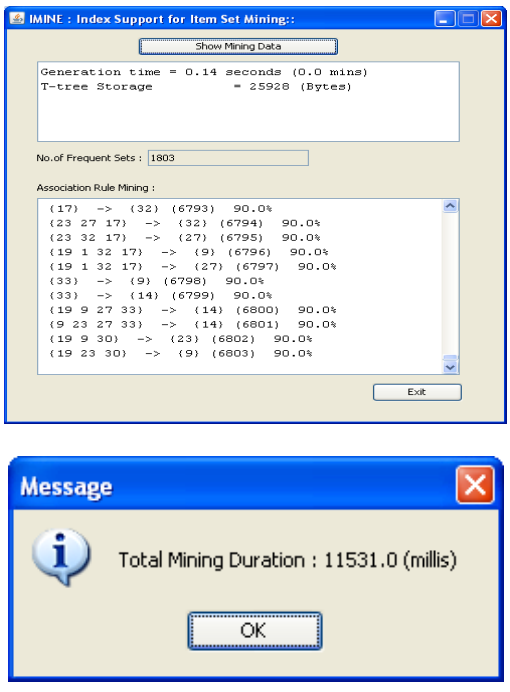




March - April 2020
ISSN: 0193-4120 Page No. 2038 - 2042



On this data set we are applying association rule mining. Finally by using Association rule mining the accuracy was displayed and total mining duration also given.



4. CONCLUSION:

IMine index may be a novel index structure that supports efficient item set mining into a relational DBMS. By exploiting its physical level access methods, it has been implemented into the PostgreSQL open source DBMS. The IMine index provides an entire and compact representation of transactional data. It efficiently supports different algorithmic approaches to item set extraction. The physical index blocks significantly reduces the I/O costs by selective access and efficiently exploits DBMS buffer management strategies. This approach, albeit implemented into a relational DBMS, yields performance better than the state-of-the-art algorithms (i.e., Prefix-Tree and LCM v.2 accessing data on a flat file and is characterized by a linear scalability also for giant data sets).

As further extensions of this work, the following issues may be addressed: Compact structures suitable for different data distributions. We are adopting the prefix-tree structure to represent any transactional database independently of its data distribution. Different techniques may be adopted, possibly ad hoc for the local density of the considered data set portion. Integration with a mining language. The proposed primitives could also be integrated with a question language for specifying mining requests, thus contributing an efficient database implementation of the essential extraction statements. Incremental update of the index. Currently, when the transactional database is updated, the IMine index must be rematerialized. To incrementally update the index when new data become available a different approach would be done. Since no support threshold is enforced during the index creation phase, the incremental update is possible without accessing the first transactional database.

5. REFERENCES:

[1] R. Agrawal and R. Srikant, “Fast Algorithm for Mining Association Rules,” Proc. 20th Int’l Conf. Very Large Data Bases (VLDB ’94), Sept. 1994.

March - April 2020
ISSN: 0193-4120 Page No. 2038 - 2042

[2] R. Agrawal, T. Imilienski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases," Proc. ACM SIGMOD '93, May 1993.

[3] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," Proc. ACM SIGMOD, 2000.

[4] H. Mannila, H. Toivonen, and A.I. Verkamo, "Efficient Algorithms for Discovering Association Rules," Proc. AAAI Workshop Knowledge Discovery in Databases (KDD '94), pp. 181-192, 1994.

[5] Savasere, E. Omiecinski, and S.B. Navathe, "An Efficient Algorithm for Mining Association Rules in Large Databases," Proc. 21st Int'l Conf. Very Large Data Bases (VLDB '95), pp. 432-444, 1995.

[6] H. Toivonen, "Sampling Large Databases for Association Rules," Proc. 22nd Int'l Conf. Very Large Data Bases (VLDB '96), pp. 134-145, 1996.

[7] M. El-Hajj and O.R. Zaiane, "Inverted Matrix: Efficient Discovery of Frequent Items in Large Datasets in the Context of Interactive Mining," Proc. Ninth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD), 2003.