Association Rule Mining Using Hash-Based Decision Tree as Directed a-Cyclic Graph

P. V. Vijay kumar
School of Computing Sciences and Engineering
VIT University
Chennai, India.

Prof. A. Vijayalakshmi
School of Computing Sciences and Engineering
VIT University
Chennai, India.

***Abstract* – Association rule mining is one of the important functionality in data mining. It is a pattern discovery technique in the domain of data mining to extract interesting correlations, frequent patterns, associations or casual structures among set of items in the databases. It is used to find the frequent patterns among large dataset. It leads to wastage of time, memory to retrieve the data. Mining data through association rules generates related item sets in a database by using support and confidence factors. One of the method available in association rule mining is using directed acyclic graph (DAG) for reducing conditional FP-trees and generate frequent item sets. Now we propose a new algorithm using hash table and decision tree representation (HDAG) as directed acyclic graph to improve performance and time complexity of a input database. In this paper we compare it to the previous existed algorithms like FP-tree as direct acyclic graph (DAG) structure and MFI algorithm.**

Index Terms - Association rule mining (ARM), DAG, HDAG, directed acyclic graph, FP-tree

1. INTRODUCTION

Data mining gives scope to the various applications due to wide applicability in many areas. Extracting required information from huge data base collections is important in business for decision making. Data mining allows us to do classification, clustering, sequence pattern mining, association rule mining, knowledge discovery and neural networks. It is widely accepted the discovery of association rules dependent on the discovery of frequent set. It is a rule like information in the if-then condition statements. Many algorithms are concerned with efficiently determining the set of frequent itemsets in a given set of the large database [2]. The problem is essentially to compute the frequency of occurrences of each itemset in the database. Since the total number of itemsets is exponential in terms of the number of items, it is not possible to count the frequencies of these sets by reading the database in just one pass. The numbers of counters are too many to be maintained in a single pass. It is suitable for different applications like classification, spatial data analysis, XML mining, stock market analysis. The perception of ARM through support and confidence over the database is to discover the frequent item sets [1][2]. Support and confidence are two measures of rule interestingness. By calculating minimum support and confidence on the existing data frequent item sets are generated. Let A and B be items in the database.

Support = set of related combinations of item/total no of combinations

Confidence=set of related combinations of item/total no of combinations of a particular item present

Suppose A=>B means that if the item A is present in the database relatively high probability existing of the item B in that particular database. A is called as antecedent and B is called as consequent [2][6]. The strength of an item set is calculated through the support and confidence of that dataset.

Support (MN) = support count of MN/total no of transactions in database [3]

Confidence (M/N) = support of (MN) value/support of item M [3]

By satisfying min-support and min-confidence frequent items are mined from the database. The calculated minimum-support and minimum confidence values must greater than or equal to the user specified support and confidence values of a item sets. Researchers developed a lot of algorithms and techniques for implementing association rules. In association rule mining with APRIORI algorithm is efficient way of retrieving frequent dataitems [5]. It mines frequent data without candidate set generation. But the serious problem is generating a candidate item sets while retrieving frequent item sets. To avoid these conflict association rule mining with direct acyclic graph is introduced. In this paper we proposed an efficient algorithm for generating frequent patterns without candidate data set generation. We generate frequent patterns using directed acyclic graph. It is an improvement algorithm for association rule mining using FP-tree as direct acyclic graph. By generating the FP- tree using directed acyclic graph we no need to generate the conditional FP-tree.

This paper is organized as: Section 2 describes the Related work Section 3 describes the Proposed work (Association rule mining using hash-based decision tree as DAG) and Section 4 describes the Implementation work and the performance analysis and Section 5 describes the Conclusion and future work

2. Related work

Data mining is the one of the problem solving technique helps to solve many business oriented problems in real life, among association rule mining is one of the important aspects for knowledge discovery. R. AGARWAL represented interested association rules among the different datasets [2]. Mining frequent patterns is a fundamental part in mining different item sets in database applications, such as sequential patterns and mining association rules etc. According to researcher Sergey Brian ETAL provided a dynamic item set counting (DIC) using APRIORI algorithm to built large item set and makes its subset also large so it will increase memory and time complexity [4]. All algorithms proposed earlier are retrieving frequent item sets continuously using association rule mining with APRIORI algorithms. Every level all subsets of frequent pattern are also retrieved frequently [4]. By these algorithms large frequent patterns with candidate keys are generated. By the earlier systems we need to scan the database continuously, hence efficiency of mining is also reduced. Because of these obstacles, a researcher JIAWEI HAN proposed an algorithm without generating a candidate key, by scanning the database less times, we are going to generate an FP-growth algorithm to increase efficiency compared to previous algorithms of association rule mining using APRIORI algorithm [4]. By avoiding the candidate generation process and less passes over the database, FP-Tree founds to be faster than the Apriori algorithm**.** An FP-Tree is a prefix tree for transactions. Every node in the tree represents one item and each path represents the set of transactions that involve with the particular item. All nodes referring to the same item are linked together in a list, so that all the transactions that containing the same item can be easily found and counted [1][2]. Research experiments shows that ARM using APRIORI algorithm is bit faster than compared to ARM using FP-growth algorithm because of the efficiency. The drawbacks of ARM using FP-growth are mining complete item sets for which if there is a large frequent item sets with size X subset, almost 2X subset of item sets are generated automatically. However to producing a large number of conditional FP-trees in mining the efficiency of association rule mining using FP-growth is having disadvantages [2] [3]. Hence association rule mining using FP-tree as directed acyclic graph is introduced to evaluate the problems [1]. To remove cyclic conditions by using merging the candidate items to reduce the complexity in mining the data. In this paper we propose a hash-based algorithm as directed acyclic representation [1].

Issues in ARM:

1. Finding minimum support :

By calculating minimum support for a frequent pattern the user .specified minimum support threshold value is set by the user [3]. If we place threshold value minimum, infrequent patterns are evolved. If we place threshold value maximum some frequent patterns are not retrieved [3]. Due to this optimization problems occur.

2. *Multiple scans in database* :

For finding frequent item sets is a database we scan the database continuously to retrieve data from datasets. The multiple scanning of data leads to

1) wastage of memory: utilizing lot of space for to store retrieve the data
2) Wastage of time: to scan whole database by continuous scanning

3. *Performance* :

Performance is varied for increasing instructions. We cannot imagine the performance based on scalability [1] [3].

3. PRAPOSED WORK

In this paper we implement an algorithm for association rule mining using hash-based tree as direct acyclic graph. It is a dynamic array data structure. So dynamically resize the array will reduce the memory space. Hash chaining is a way to resolve acyclic collisions. Each slot of array contains a linked list of key value and data pairs connected in symmetric manner in a single linked list. Hence it is efficient than other binary tree structure. Ideally hash tree is perfect suited to binary tree where two nodes as leaf nodes. Initially it contains only the root node and the leaf node contains no candidate itemset. Each candidate item set is inserted in to the tree as a internal node and there is a link in the hash table.

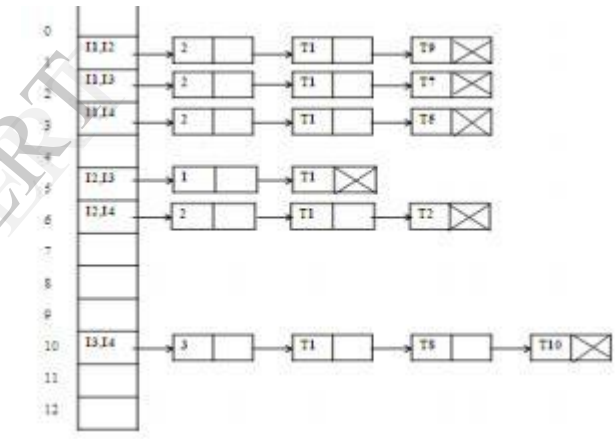

Figure 1 Hash-based technique

Hash table contains number of nodes for hash table processing. Each node consists of two parts. In the first part the required data is stored and another one is mapping function considered as hash function. The hash function stores the way of assigning data in numbers such as way to store data using index. The hash function maps the keys in to corresponding bucket values assigned in the hash table.

Mining association rules using hash table: Using hash table we can reduce the candidate item sets in finding frequent items. We can prune the infrequent patterns based on minimum support and confidence factors. It plays a major role for finding frequent item sets. While scanning the data sets for finding frequent 1-item sets (L1) all possible outcomes of 2-item set is generated. The generated item sets are mapped in to the hash buckets of hash table structure depending on the hash functions, through this continuous chain process corresponding bucket count are increased. Then the value of each bucket count is checked over minimum support whose count is less

than minimum support count is detected and pruned from the candidate set.

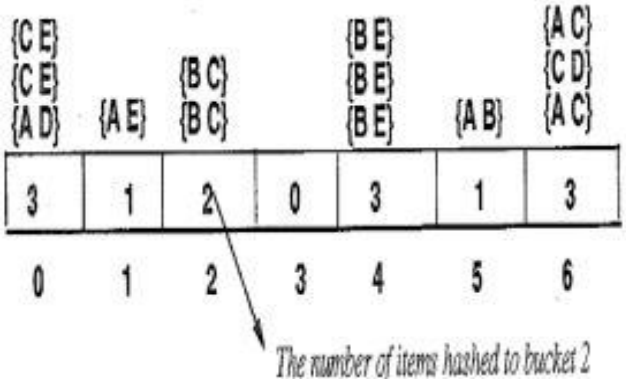

Figure 2 Generating hash-tree inputs

1) First to initialize the item sets in transactional database in vertical format
2) Convert the extracted vertical transactional data base to horizontal transactional data base
3) For mining first level item set of transactional data base h (k) =order mod (n), where n is no of items in a level of that transactional database which is nearest prime number of number of transactions.
4) Place the item sets in an array format, called H- BIT array.
5) These arrays having binary values 0 and 1 value. Where particular item set is present then places 1 other wise place 0 to the corresponding array item set.
6) Represent the transactional item sets in array repeated in 1st level
7) Find the possible combinations of transactions in the candidate-1 itemset and generate candidate-2 item set
8) Find the possible combinations of transactions in the candidate-1 itemset and generate candidate-2 item set
9) Place it in H-bit array and generate and update the values in array and generate candidate-2 item set.

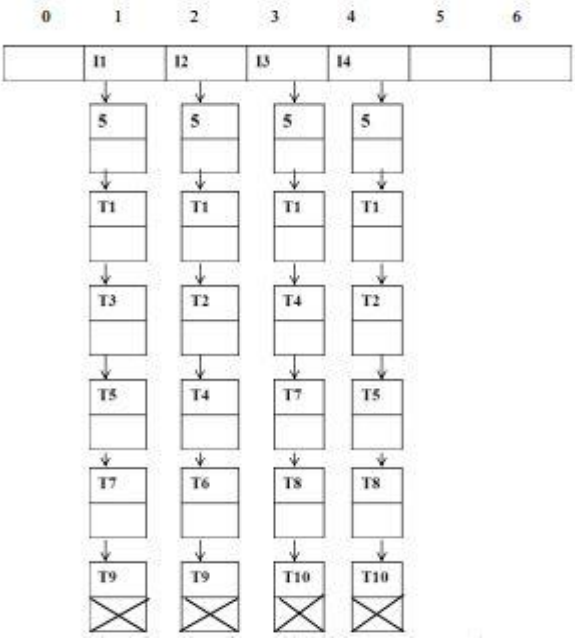

Figure 3 Hash table using single linked list for candidate-1 itemset

10) Find the possible combinations of transactions in the candidate-1 itemset and generate candidate-2 item set
11) Place it in H-bit array and generate and update the values in array and generate candidate-2 item set.
12) Check the minimum support value of each transaction in an item set.
13) Repeat the above steps for mining data and apply the procedure to find frequent item sets

This process terminates when no more item sets exist in the every level of hash table in it. This process is mainly used for reusability of memory in RAM. Where the first level data is deleted then same memory is allocated to the second level of candidate key generation. By using hash chaining using linked list structure frequent item sets are evolved easily compared to other previous algorithms. Hashing is takes place through level by level and equal no of candidate item sets are generated by all levels in a hash table.

Basic Algorithm for hashing:

Input: Data base as inputs consists of transaction (T)
Output: Frequent item sets
Database contains set of transactions
Transaction < =.TID, {X belongs to set all items in T}
F1, F2 …FN are frequent item sets
IF (F1 = 0) empty set
then add items into the transaction
For (I = 0 to X in a transaction T)
H1.add(x)
End for
End if
Check for hash support value of each transaction find
1-item set

```
If (H1.hashsupport(y)
    then add F1 to y
End if
Remove hash values without having minimum support value
If(y<minimum support)
    H1.purne (minimum support)
    D1= database
For more than 2-item sets apply recursively
```

After finding the frequent item sets based on the algorithm the obtained results are applied to the HDAG structure. Here DAG structure is to reduce collisions among the item sets.

Algorithm for HDAG

Input: VD database induced graph chain, α: minimum support, TID Transactions Id

Output: frequent item sets of VD

1) If VD is the sink node then stop process
2) Return the value
3) End if
4) It two transactions have same transactional ids are same in two different chains
5) Merge the two transactional ids to form a single transactional chain.
6) Apply the procedure for the possible ways to reduce.
7) Repeat the steps for VD[1….N] in a table

Performing the merging operation over the transactions in hash tree, we can minimize the code and maximize the efficiency. Through this, maximum reduction of data is possible for mining infrequent data.

4. Implementation work

In the implementation work, to generate and analyze the performance HDAG over an existed DAG and FP-tree structures. The above algorithm is implemented in the java language. Swing components are used in GUI environment to compile the algorithms. Data are collected as transactional records from the Vmart Supermarket, XYM Company and some service data set for a period of one month. For database storage SQL server is used for managing.

Inputs:

Data set	No of transactions	No of items
Market data set	1256	456
Company XYM	456	543
Service data set	666	43

Rules generated by algorithms:

Data set	No of rules
Market data set	337
Company XYM	321
Service data set	221

Comparisons

No of items	Existed DAG	Proposed HDAG
345 1-item set	223	201
4456 2-item set	344	312
546 3-item set	776	334

Reduce memory: Memory was reduced while compared to the other algorithms. Having less rule, memory reusability is possible.

Reduce time: The rules generated through the HDAG mining technique are less compared to previous algorithms hence it takes less time retrieve.

5. Conclusion

Mining association rules is major task in data mining. Hash based-mining (HADT) is used to improve the performance of finding frequent itemset while comparing with the previous algorithm. Association rule mining using APRIORI iteratively checks candidate key. Through the algorithm presented in this paper we can resolve the problems in better manner .ARM using FP-tree algorithm is a better option for mining but our algorithm can reduce the total steps and less time and less memory usage. Through hashing we can improve performance. For future works we can improve this technique for large datasets.

6. References

[1] V venkateswara rao and E rambabu “Association Rule Mining Using FPTree” IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012) March 30, 31, 2012.

[2] R Agarwal and A swaami “Mining an association rules between set of items in large transactional database”, Proceedings of the 1993 ACM SIGMOD Conference Washington DC, USA.

[3] Ke-chung Lin, I-En Liao Zhi-sheng chen “An improved frequent pattern growth method for mining association rules” Expert Systems with Applications: An International Journal Volume 38 Issue 5, May, 2011

[4] A.B.M Rezbaul Islam, Tae-Sun Chung "An Improved Pattern Tree Based Association Rule Mining Technique" IEEE International Conference on Information Science and Applications (ICISA), 201 I.

[5] Qihua Lan,Defu Zhang, Bo Wo, "A new algorithm for frequent item set mining based on apriori and FP-tree", Global Congress on Intelligent System 2009

[6] Irina Tudor, "Association Rule Mining as Data Mining technique" BULETTNU Luniversi Noll2008,page 49-56.

[7] Jiawei Han, jian pei, and Yiwen Yin,"Mining frequent patterns without candidate generation ",paper id :196, SIGMOD '2000.

[8] Arun K Pujai "Data Mining Techniques UniversityPress (India) Pvt. Ltd 2001.