RESEARCH ARTICLE OPEN ACCESS

Finding Frequent Patterns Based On Quantitative Binary Attributes Using FP-Growth Algorithm

R. Sridevi,* Dr. E. Ramaraj**
*Assistant Professor, Department of Information Technology, Dr. Umayal Ramanathan College for women, Karaikudi
**Professor,Department of Computer Science and Engineering, Alagappa University, Karaikudi

ABSTRACT
Discovery of frequent patterns from a large database is considered as an important aspect of data mining. There is always an ever increasing demand to find the frequent patterns. This paper introduces a method to handle the categorical attributes and numerical attributes in an efficient way. In the proposed methodology, ordinary database is converted in to quantitative database and hence it is converted in to binary values depending on the condition of the student database. From the binary patterns of all attributes presented in the student database, the frequent patterns are identified using FP growth,The conversion reveals all the frequent patterns in the student database.
Keywords: Data mining,Quantitative attributes, Frequent patterns, FP-growth algorithm,

I. INTRODUCTION

Data mining has recently attracted considerable attention from database practitioners and researchers because it has been applied to many fields such as market strategy, financial forecasts and decision support [4]. Many algorithms have been proposed to obtain useful and invaluable information from huge databases. One of the most important algorithms is mining association rules, which was first introduced by Agarwal[5]. The association rule mining problem is to find rules that satisfy user-specified minimum support and minimum confidence. It mainly includes two steps: first, find all frequent patterns; second, generate association rules through frequent patterns.

Many algorithms for mining association rules from transactions database have been proposed in [7],[8],[9]. However, most algorithms were based on Apriori algorithm which generated and tested candidate item sets iteratively. It scans the database many times, so the computational cost is high. In order to overcome the disadvantages of Apriori algorithm and efficiently mine association rules without generating candidate item sets, a frequent pattern- tree (FP-Growth) structure is proposed in [8]. According to FP-Growth, the database is compressed into a tree structure which shows a better performance than Apriori. However, FP-Growth consumes more memory and performs badly with long pattern data sets[3].

1.1 Basic concepts

The problem of mining association rules can be explained as follows: There is a item set I= {i1, i2… in} where I is a set of n discrete items, and consider D as a set of transactions, where each transaction, denoted by T, is a subset of items I.

"Table 1" gives an example where a database D contains a set of transaction T, and each transaction consist of one or more items of the set {A,B,C,D,E,F}.
Table 1 : Sample data.

T No.	Item sets			
T1	A	B		
T2	A	C	D	E
T3	B	C	D	F
T4	A	B	C	D
T5	A	B	D	F

An association rule is an inference of the form $X \Rightarrow Y$, where $X, Y \subseteq I$ and $X \cap Y = \varphi$. The set of items X is called antecedent and Y the consequent. Support and confidence are two properties that are generally considered in association rule mining. The same two measures are used in the proposed method to identify the frequent patterns.

Support S for a rule, denoted by $S(X \Rightarrow Y)$, is the ratio of the number of transactions in D that contain all the items in X U Y to the total number of transactions in D.

$$S(X \Rightarrow Y) = \frac{\sigma(X \cup Y)}{|D|} \qquad \text{..... (1)}$$

where the function σ of a set of items X indicates the number of transactions in D, which contains all the items in X.

Confidence C for a rule X ⇒ Y, denoted by C (X ⇒ Y), is the ratio of the support count of (X U Y) to that of the antecedent X.

$$C(X \Rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)} \quad ....(2)$$

The minimum support Smin and minimum confidence Cmin is defined by the user and the task of association rule mining is to mine from a data set D, that have support and confident greater than or equal to the user specified support value [10] .

Association rule mining is a two-step process:
1. Find all frequent item sets: All the item set that occur at least as frequently as the user defined minimum support count.
2. Generate strong association rules: These rules must satisfy minimum support and minimum confidence and derived from the frequent item set.

For example, let us assume that the minimum Support for the items in Table 1 is 40% and the minimum confidence is 60%. The support and confidence of the rule is checked to determine whether the association rule {A, B} ⇒ {D} is valid rule or not.

II. Frequent Item sets

2.1Mining frequent itemsets

Let I = {x1, x2, x3 … xn} be a set of items. An item set X is also called pattern is a subset of I, denoted by X⊆I. A transaction TX = (TID, X) is a pair, where X is a pattern and TID is its unique identifier. A transaction TX is said to contain TY if and only if Y ⊆X. A transaction database, named TDB, is a set of transactions. The number of transactions in DB that contain X is called the support of X. A pattern X is a frequent pattern, if and only if its support is larger than or equal to s, where s is a threshold called minimum support[1].Given a transaction database, TDB, and a minimum support threshold, s, the problem of finding the complete set of frequent item sets is called the frequent item sets mining problem.

2.2 Existing Algorithms for Mining frequent item sets

Though there are number of algorithms for mining frequent item sets, the most popular algorithms are Apriori and FP-Growth which are discussed below.

2.2.1Apriori algorithm

The *Apriori* algorithm is the most well known association rule algorithm proposed by Agarwal and is used in most commercial products. The Apriori algorithm can be used to mine the frequent itemset in the database. It is based on the fact that algorithm using the prior knowledge of the frequent itemset. Apriori algorithm is actually a layer-by-layer iterative searching algorithm, where k-itemset is used to explore the (k+ 1)-itemset. The use of support for pruning the candidate item sets is guided by the following principles.

Property 1: If an item set is frequent, then all of its subsets must also be frequent.

Property 2: If an item set is infrequent, then all of its supersets must also be infrequent.

The algorithm initially scans the database to count the support of each item. Upon completion of this step, the set of all frequent 1-itemsets, *F*1, will be known. Next, the algorithm will iteratively generate new candidate *k*-item sets using the frequent (*k*-1)-item sets found in the previous iteration. Candidate generation is implemented using a function called *Apriori-gen*. To count the support of the candidates, the algorithm needs to make an additional scan over the database. The subset function is used to determine all the candidate item sets in *Ck* that are contained in each transaction *t*. After counting their supports, the algorithm eliminates all candidate item sets whose support counts are less than *minsup*. The algorithm terminates when there are no new frequent item sets generated.

2.2.2. FP-tree

Han et al. [6] developed an efficient algorithm, FP growth, based on FP-tree. It mines frequent item sets without generating candidates and scans the database only twice. The first scan is to find 1- frequent item set; the second scan is to construct the FP-tree. The FP-tree has sufficient information to mine complete frequent patterns. It consists of a prefix-tree of frequent 1-itemset and a frequent-item header table in which the items are arranged in order of decreasing support value. Each node in the prefix-tree has three fields: *item name*, *count*, and *node-link*. *Item-name* is the name of the item. *Count* is the number of transactions that consist of the frequent 1-items on the path from the root to this node. *Node-link* is the link to the next same item-name node in the FP-tree. From the FP-tree the conditional pattern base and the conditional FP-tree is also generated. The frequent items are obtained only from the conditional FP-tree. A number of combinations are produced as a result which are the necessary frequent patterns[1].
FP-Growth Algorithm:

Algorithm 1: (FP-tree construction)**.**
Input: A transaction database *DB* and a minimum support threshold *ξ*.
Output: FP-tree, the frequent-pattern tree of *DB*.
Method: The FP-tree is constructed as follows.
1. Scan the transaction database *DB* once. Collect *F*, the set of frequent items, and thesupport of each

frequent item. Sort *F* in support-descending order as *FList*, the *list* of frequent items.

2. Create the root of an FP-tree, *T* , and label it as "null". For each transaction *Trans* in *DB* do the following.

Select the frequent items in *Trans* and sort them according to the order of *FList*. Let the sorted frequent-item list in *Trans* be [*p* | *P*], where *p* is the first element and *P* is the remaining list. Call *insert tree* ([*p* | *P*], *T*).The function *insert tree*([*p* | *P*], *T*) is performed as follows. If *T* has a child *N* such that *N.item-name* = *p.item-name*, then increment *N*'s count by 1; else create a new node *N*, with its count initialized to 1, its parent link linked to *T* , and its node-link linked to the nodes with the same *item-name* via the node-link structure. If *P* is nonempty, call *insert tree* (*P*, *N*) recursively.

Since the most frequent items are near the top or root of the tree, the mining algorithm works well, but there are some limitations [11].

a) The databases must be scanned twice.

b) Updating of the database requires a complete repetition of the scan process and construction of a new tree, because the frequent items may change with database update.

c) Lowering the minimum support level requires complete rescan and construction of a new tree.

d) The mining algorithm is designed to work in memory and performs poorly if a higher memory paging is required.

III. Related Work

A relational database consists of a table in which a row represents a record while a column represents an attribute of the database. For each attribute, it could be Boolean, numerical or character types [2].The algorithms that are discussed above are used to find the frequent set of items from the transactional database.

The representation of database in different format such as multi dimensional, relational, Quantitative database, binary database returns to find out the frequent item sets and the association rule generation in an effective manner. The following figure 1 illustrates the overall architecture of the proposed method. For simplification, the proposed method is implemented in the student database having five attributes.

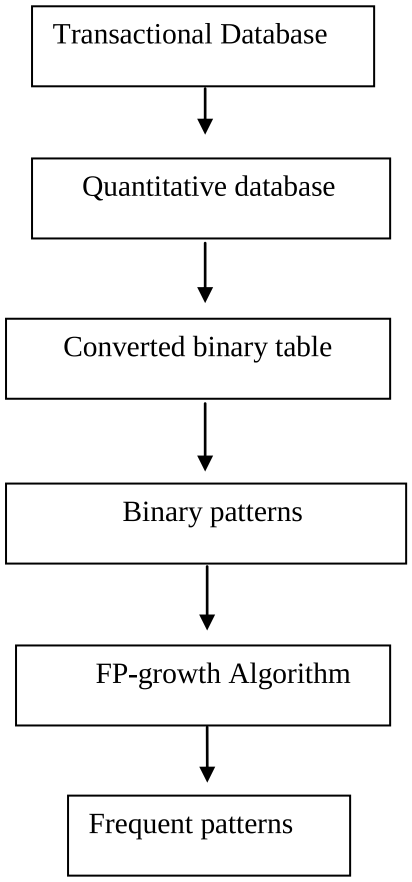

Fig.1 Architecture of the proposed method

In this paper, let us consider a sample of student database with ten rows of items as shown in the Table 2. The database includes the information about the student such as Place, Annual Income, Age, etc. Among the attributes annual income and age are considered as numerical where Place, medium and Sex are categorical attributes. The attributes are chosen according to the problem. Later, these attributes are useful to find the association rules.

In a relational database, storage structure determines the acquisition of association rules with obvious features:

Table 2: Student database

Tid	Place	Annual Income	Age	Medium	Sex
100	Rural	80,000	21	Tamil	Male
101	Urban	90,650	20	English	Female
102	Rural	2,05,000	22	English	Male
103	Rural	85,000	19	Tamil	Female
104	Urban	1,50,000	19	Tamil	Male
105	Rural	90,000	18	English	Male
106	Rural	2,34,000	22	Tamil	Female
107	Urban	2,14,000	21	Tamil	Male
108	Rural	1,36,000	23	English	Female
109	Rural	1,70,000	23	English	Male

1. An attribute of the relational database does not take value of vector. Relational database association rule mining is a problem of multidimensional association rules mining.
2. Attributes in relational database could have different data types and could take on different values. Therefore, the multiple value association rule mining or quantitative association rule mining is the key feature to find the frequent patterns.
3. In a multi-dimensional database, many dimensions can yield a concept hierarchy, and therefore, the relational database association rules tend to cover different attributes to multiple conceptual layers.

3.1. Type conversion based on Binary values

Mostly, the database contains not only the conditional attributes, but also contains a large number of class attributes. Usually before mining, it maps the relational categories and numerical attributes into Boolean attributes, and converts the relational data table into database form as in Table 3. Every record must be converted to a common format to mine the frequent item sets. The numerical attributes are discretized into two ranges, one above the average value and the other below the average value. The categorical attributes are also divided into two categories based on the condition. The range for any attribute is assigned the binary value 0 and 1 depending upon the constraints.

The proposed method is applied in Table 1 to convert the data to the database storage form, and then conduct the pre-processing. For example, the attribute place is categorized as Rural and Urban for which we substitute the value 0 and 1 respectively. Similarly all the categorical attributes are classified into 0 and 1 according to the attribute type. For the numerical attribute, several values exist in the database. Data discretization techniques can be used to reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals.

From Table 2, the attribute age has values from 18 to 23. In such situations, all the values cannot be observed during rule generation. Therefore, the values are divided in to two ranges, one above the average value and the other below the average. Based on the above information, the minimum and maximum values are *MIN* =18, and *MAX* = 23. The midrange can also be used to assess the central tendency of a data set. It is the average of the largest and smallest values in the set. This algebraic measure is easy to compute using the SQL aggregate functions such as max() and min().

Table 3: Converted binary table

Tid	Place	Annual Income	Age	Medium	Sex
100	0	0	0	0	1
101	1	0	0	1	0
102	0	1	0	1	1
103	0	0	1	0	0
104	1	1	1	0	1
105	0	0	1	1	1
106	0	1	0	0	0
107	1	1	0	0	1
108	0	1	0	1	0
109	0	1	0	1	1

IV. Finding the frequent patterns

A pattern *A* is frequent if *A*'s support is no less than a predefined minimum support threshold. In the Table 3, all the attributes are converted in to binary digits. Whenever the frequent items are scanned it returns a sequence of 0s and 1s.It is ambiguous to locate a particular attribute. In order to identify the individual frequent items the proposed method assigns the coordinates as the combination of attribute column number and the corresponding binary value presented in each location of the binary table as shown in Table 4.

Table 4: Binary frequent patterns

id	Place	Annual Income	Age	Medium	Sex
100	(1,0)	(2,0)	(3,0)	(4,0)	(5,1)
101	(1,1)	(2,0)	(3,0)	(4,1)	(5,0)
102	(1,0)	(2,1)	(3,0)	(4,1)	(5,1)
103	(1,0)	(2,0)	(3,1)	(4,0)	(5,0)
104	(1,1)	(2,1)	(3,1)	(4,0)	(5,1)
105	(1,0)	(2,0)	(3,1)	(4,1)	(5,1)
106	(1,0)	(2,1)	(3,0)	(4,0)	(5,0)
107	(1,1)	(2,1)	(3,0)	(4,0)	(5,1)
108	(1,0)	(2,1)	(3,0)	(4,1)	(5,0)
109	(1,0)	(2,1)	(3,0)	(4,1)	(5,1)

Since only the frequent items play a role in the frequent-pattern mining, it is necessary to perform one scan of transaction database *DB* to identify the set of frequent items. The frequent item sets obtained through the conversion are (3,1): 4(3,0):4, (3,0),1,0):4, (2,1):4, (2,0):4,(1,0):5.The frequent 1-itemsets and 2-itemset can be found out from the above binary frequent patterns as in Table 4 using FP growth algorithm.

V. Experimental analysis

In Table 5, the efficiency of the quantitative based binary attributes conversion is compared with the existing method. To verify the conversion, a student database with 100 records and five attributes which have different values and categories is taken for study. The Binary based conversion as well as the existing FP growth is implemented in Java. As shown in Table 5, the results are obtained for different volumes of records and presented. The performance of the proposed method is faster than the existing method.

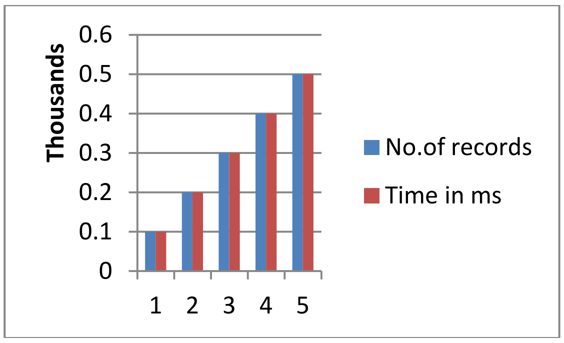

Fig. 2. Quantitative binary method

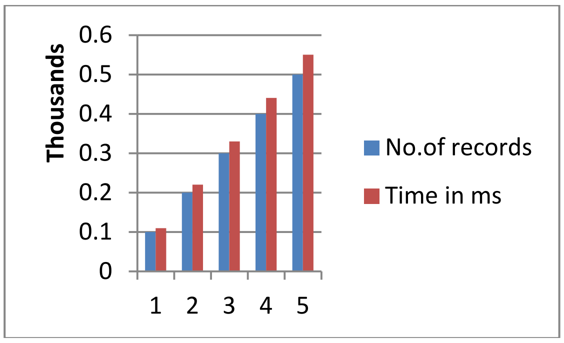

Fig. 3. Existing method

Table 5: Performance Comparison

	Performance of FP-growth without the conversion	Performance of FP-growth with the conversion
Memory usage	98.00567 mb	97.01559 mb
Computational speed	560 ms	500 ms

VI. Conclusion

In this paper, a new method is proposed to convert the categorical attributes and numerical attributes. The main feature of the method is that it considerably reduces the memory usage and execution time. The data representation helps many users to find the frequent items for the transactional database. It stores all the in the form of binary digits and hence require less memory and less computational time. Using the above method, association Rule mining can also be deployed. The same method can be used for different types of datasets such as Market basket analysis and Medical data set.

References

[1] Qihua Lan, Defu Zhang, Bo Wu ,A New Algorithm For Frequent Itemsets Mining Based On Apriori And FP-Tree,Department

of Computer Science, Xiamen University, Xiamen China *2009 IEEE*

[2] Lei Wangt, Xing-Juan Fan2, Xing-Long Lwt, Huan Zha Mining data association based on a revised FP-growth Algorithm *Proceedings of the 2012 International Conference on Machine Learning and Cybernetics, Xian, 15- 17 July,*

[3] Bo Wu, Defu Zhang, Qihua Lan, Jiemin Zheng ,An Efficient Frequent Patterns Mining Algorithm based on Apriori Algorithm and the FP-tree Structure *Department of Computer Science, Xiamen University, Xiamen*

[4] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. *In Proc.1993 ACM-SIGMOD Int. Conf. Management of Data, Washington, D.C., May 1993, pp 207–216*

[5] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. *In VLDBY94, pp. 487-499.*

[6] J.S .Park ,M.S.Chen and P.S.Yu.An effective hash based algorithm for mining association rules. *In SIGMOD1995, pp 175-186.*

[7] A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. *Proceedings of the 21st International Conference on Very large Database,1995.*

[8] J. Han, J. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation(PDF), (*Slides), Proc. 2000 ACM-SIGMOD Int. May 2000*.

[9] Agarwal R,Aggarwal C,Prasad V V V.A tree projection algorithm for generation of frequent item sets. *In Journal of Parallel and Distributed Computing (Special Issue on High Performance Data Mining),2000*

[10] A.B.M.Rezbaul Islam, Tae-Sun Chung An Improved Frequent Pattern Tree Based Association Rule Mining Techniques *Department of Computer Engineering Ajou University Suwon, Republic of Korea*

[11] E. Ramaraj and N. Venkatesan, ― Bit Stream Mask Search Algorithm in Frequent Itemset Mining, *European Journal of Scientific Reasearch,‖ Vol. 27 No.2 (2009),*