Algorithm for Efficient Multilevel Association Rule Mining

Pratima Gautam
Department of computer Applications
MANIT, Bhopal

Dr. K. R. Pardasani
Dept. of Mathematics & Computer Applications
MANIT, Bhopal (M.P.)

Abstract— **over the years, a variety of algorithms for finding frequent item sets in very large transaction databases have been developed. The problems of finding frequent item sets are basic in multi level association rule mining, fast algorithms for solving problems are needed. This paper presents an efficient version of apriori algorithm for mining multi-level association rules in large databases to finding maximum frequent itemset at lower level of abstraction. We propose a new, fast and an efficient algorithm (SC-BF Multilevel) with single scan of database for mining complete frequent item sets. To reduce the execution time and increase throughput in new method. Our proposed algorithm works well comparison with general approach of multilevel association rules.**

*Keywords- **Data mining, association rules, multilevel association rules, transaction database***

I. INTRODUCTION

Data mining has attracted much attention in database communities because of its wide applicability [12]. One major application area of data mining is to discover potentially useful information from transaction databases. The problem of mining association rules from transactional data was introduced in [1]. A transaction in the database consists of a set of items (itemset). An example of such an association rule might be "80% of customers who buy itemset X also buy itemset Y". The support count of an itemset is the number of transactions containing the itemset, and the support of the itemset is the fraction of those transactions. The itemset (X*Y in the example) involved in an association rule must be contained in a predetermined number of transactions. The predetermined number of transactions is called the minimum support count, and the fraction of transactions is called the minimum support threshold. An itemset is called a large itemset if its support is no less than the minimum support threshold. Since the generation of association rules is straightforward after the large itemsets are discovered, finding large itemsets becomes the main work of mining association rules [7]. Many applications at mining associations require that mining be performed at multiple levels of abstraction [14]. For example, besides finding 80 percent of customers that purchase milk may also purchase bread, it is interesting to allow users to drill-down and show that 75 percent of people buy wheat bread if they buy 2 percent milk [6]. The association relationship in the latter statement is expressed at a lower level of abstraction but carries more specific and concrete information than that in the former. Therefore, a data mining system should provide efficient methods for mining multiple-level association rules [4] [8]. To explore multiple-level association rule mining, one needs to provide: 1) data at multiple levels of abstraction, and 2) efficient methods for multiple-level rule mining. The first requirement can be satisfied by providing concept taxonomies from the primitive level concepts to higher levels [15]. In many applications, the taxonomy information is either stored implicitly in the database, such as, ^{a}Wonder wheat bread is a wheat bread which is in turn bread, or provided by experts or users, such as, Freshman is an undergraduate student, or computed by applying some cluster analysis methods [2]. With the recent development of data warehousing and OLAP technology, arranging data at multiple levels of abstraction has been a common practice [3]. Therefore, in this study, we assume such concept taxonomies exist, and our study is focused at the second requirement, the efficient methods for multiple-level rule mining. There are several possible directions to explore efficient mining of multiple-level association rules. This paper presents an efficient version of apriori algorithm for mining multi-level association rules in large databases to finding frequent itemset at different level of abstraction. Proposed algorithms using support count table and bit from table for counting frequent itemset at each level. This algorithm is called SC-BF Multilevel (Support count and bit from multilevel) algorithm. The proposed algorithm has the following advantages. 1) It generates a much smaller set of high quality rules directly from the data set. 2) It groups the items in each level. 3) The number of records / fields in the dataset also be reduced which is useful to maintain large set of databases. This algorithm is also used progressive deepening method. The method first finds frequent data items at the top most level and then progressively deepens the mining process into their frequent descendants at lower concept levels [9]. This method is using concept of reduced support and refine the transaction able at each level. More ever, the proposed algorithm employs the following features to further improve its accuracy and efficiency. It also takes less memory to store the entire data because the data was grouped at branch level and reduce the execution time.

II. APORIORI ALGORITHM

Apriori [4] is the very first efficient algorithm to mine association rules. It works iteratively and makes as many passes over the database as the length of maximal itemset. An itemset is maximal large if it has no superset that is large. Let an itemset having *k* items be denoted as *k*-itemset. The first pass of the algorithm simply counts item occurrences to determine the large 1-itemsets. A subsequent pass, say pass *k*, consists of two phases. First, the set of large {*k*-1} itemsets L_{k_1} found in the (*k*-1) th pass are used to generate the set of candidate *k*-itemsets C_k, using the apriori-gen function [7]. Next, the database is scanned and the support of candidates in C_k is counted using the counting based method in order to determine the large *k*-itemsets L_k. The *apriori-gen* function takes as argument L_{k_1}, the set of all large (*k*-1)-itemsets. It returns a superset of the set of all large *k*-itemsets [8]. The function works as follows:

1. The first step is *join* in which L_{k_1} is joined with L_{k_1}. Select two itemsets *p*, *q* from $L_{k\text{-}1}$ such that first *k*-2 items of *p* and *q* are same, and then form a new candidate *k* itemset *c* as: Common *k*-2 items + 2 differing items
2. Next in the *prune* step, prune that *c*, such that some (*k*-1) subset of *c* is not in $L_{k\text{-}1}$ this is because all subsets of a large itemset must also be large.

III. MULTI LEVEL ASSOCIATION RULE MINING

We can mine multilevel association rules efficiently using concept hierarchies, which defines a sequence of mappings from a set of low-level concepts to higher-level, more general concepts [3]. Data can be generalized by replacing low-level concepts within the data by their higher-level concepts or ancestors from a concept hierarchy. In a concept hierarchy, which is represented as a tree with the root as D i.e., Task-relevant data [9]. This uses a hierarchy information encoded transaction table instead of the original transaction table. This is because a data mining query is usually in relevance to only a portion of the transaction database, such as food, instead of all the items. It is beneficial to first collect the relevant set of data and then work repeatedly on the task-relevant set[13][14]. Encoding can be performed during the collection of task relevant data and, thus, there is no extra encoding pass required. Besides, an encoded string, which represents a position in a hierarchy, requires fewer bits than the corresponding object-identifier[10] [11].Therefore, it is often beneficial to use an encoded table, although our method does not rely on the derivation of such an encoded table because the encoding can always be performed on the fly. To simplify our discussion, an abstract example which simulates the real life example of Example 1 is analyzed as follows.

Example1. As stated above, the taxonomy information for each (grouped) item in Example 1 is encoded as a sequence of digits in the transaction table T [1]. For example, the item `2' percent Foremost milk' is encoded as `A11' in which the first digit, `A', represents `Cloths' at level-1, the second, `A', for `Jeans' at level-2, and the third, `A', for the brand `Disel' at level-3. Similar to [2][3], repeated items (i.e., items with the same encoding) at any level will be treated as one item in one transaction

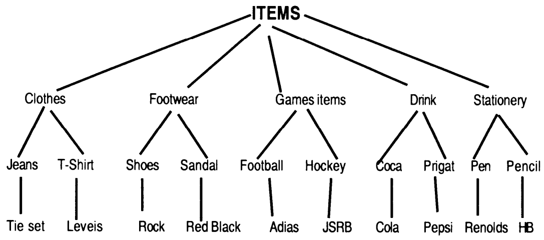

Fig1. The taxonomy for the relevant data items

IV. PROPOSED ALGORITHM

We develop a fast and an efficient algorithm based on the occurrences of items as well as by performing Logical AND operation [5]. We develop a simple and an efficient algorithm (SC-BF Multilevel).

Step- 1

Encode taxonomy using a sequence of numbers and the symbol ''*'', with the *l*th number representing the branch number of a certain item at levels.

Step- 2

Set k = 1, where k is used to store the level number being processed whereas k {1, 2, 3} (as we consider up to 3-levels of hierarchies).

Step- 3

Set the minimum support for each level according top-down deepening method.

Step- 4

Scan the Transactional database to construct the support count (ST) table and Bit from table (BT)

Step- 5

In the table ST, take the set of nodes at first level which satisfies the minimum support count.

get the group of nodes which are totally connected (H) with each other.

Step- 6

For all H [5]

(i) Perform Logical 'AND' operation.

(ii) Find the total value of the resultant value of
Logical 'AND' operation.

(iii) If the whole value of Logical 'AND' operation satisfies the min_support count and then add 'H' to the frequent item sets L_k

Step- 7

Repeat the process step-2 to step-6 for next levels (i.e. k= 2, k= 3)

V. AN ILLUSTRATIVE EXAMPLE

An illustrative example is given to understand well the concept of the proposed algorithm. The process is started from a given transactional database as shown in Table 1

Table 1

Trans_ID	List of items
T1	A11, B11, E11
T2	B11, D11
T3	B11, C11
T4	A11, D11, B22, C22
T5	A22, C22
T6	B11, C11, D11
T7	A22, C22, D11
T8	A11, B11, C11, E11
T9	A11, B11, C11, E11
T10	E22,D22

Table1 [a]
Codes of item name

Code	Description
A**	Cloth
B**	Footwear
C**	Games items
D**	Drink
E**	Stationery
A1*	Jeans
A2*	T-Shrit
B1*	Shooes
B2*	Sandal
C1*	Football
C2*	Hokey
D1*	Coca
D2*	Prigat
EC*	Pen
A11	Cloth Jeans Ti set
A22	Cloth T-shirt Levis
B11	Footwear Shooes Rocky
B22	Footwear Sandal Redback
C11	Games items Football Adias
C22	Games items Hokey JSRB
D11	Drink Coca cola
D22	Drink Prigat Pepsi
E11	Stationery Pen Renolds
E22	Stationery Pencil HB

Level-1
Min_Support = 3.0

Consider the above transactional database. According to the proposed algorithm, first we find the 1-itemset at level-1 in simple way, after that scan the above transactional database to construct the table ST and BT for 2-itemset. Both the tables ST and BT had been constructed using array. Initially in ST, all the location had been initialized to zero. Behind scanning the first transaction (A**, B**, E**), in the ST for the location (A**, B**), (A**, E**), (B**, E**) 1 will be added to their previous value [5]. After scanning the first transaction, the ST & BT as shown below

1-itemset

A = {6}**
B = {7}**
C = {7}**
D = {5}**
E = {4}**

2-itemset

Table 3: ST

	A**	B**	C**	D**	E**
A**		1	0	0	1
B**			0	0	1
C**				0	0
D**					0

Table 4: BT

A**	B**	C**	D**	E**
1	1	0	0	1

After scanning the second transaction (B11, d11) in the location (B11, d11) of ST 1 will be added to its previous value.

Table 5: ST

	A**	B**	C**	D**	E**
A**		1	0	0	1
B**		1	0	1	1
C**				0	0
D**					0

Table 6: BT

A**	B**	C**	D**	E**
1	1	0	0	1
0	1	0	1	0

In this way, after scanning the transactional database at first level (2-itemset), the ST and the BT are as shown below.

Table 7: ST

	A**	B**	C**	D**	E**
A**		4	5	2	3
B**			5	3	3
C**				3	2
D**					1

Table8: BT

A**	B**	C**	D**	E**
1	1	0	0	1
0	1	0	1	0
0	1	1	0	0
1	1	1	1	0
1	0	1	0	0
0	1	1	1	0
1	0	1	1	0
1	1	1	0	1
1	1	1	0	1
0	0	0	1	1

According to step 5, consider the value if it satisfies the min_support count value in ST. Now, the ST is shown below

Table 9: ST

	A**	B**	C**	D**	E**
A**		4	5	2	3
B**			5	3	3
C**				3	0
D**					0

Find the wholly connected nodes now. For the row ‘A**’ under B**, C** and E** the value satisfy the min_support count value. Then check the 3-itemsets are fully connected or not.

3-itemset

Table10: ST

	C**	D**	E**
(A**, B**)	3	1	3

Finally we get A**’, B**, C** and E** items. Because the value assure the min_support count value. Moreover D** item con not be consider for next level. Because the value not satisfy the min_support count value. After the stage logical AND operation [5] for A**, B**, C** and A**, B**, E** using BT we will get the whole value as 3 and 3 respectively. Since the total value of the Logical AND operation satisfy the min_support count value we can add these two itemsets to L_k frequent item sets. For 2-iemsets, we can directly write it from ST with their support count. in the same way find the remaining fully connected item sets at level-1 by checking all rows of ST (table-7) and find the Logical AND operation for them find whether that fully connected itemset are frequent itemset or not. After that we go to next level and find frequent itemset at level-2.

Level-2

Min_Support = 2.0

1-itemset

A1* = {4}
A2* = {2}
B1* = {6}
B2* = {1}
C1* = {4}
C2* = {3}
E1* = {3}
E2* = {1}

B2* and E2* can not consider for further process. Because the value not satisfy the min_support count value.

2-itemset

Table11: ST

	A1*	A2*	B1*	C1*	C2*	E1*
A1*			3	2	1	3
A2*				0	2	0
B1*				4	0	3
C1*						0

Table12: BT

A1*	A2*	B1*	C1*	C2*	E1*
1	0	1	0	0	1
0	0	1	0	0	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	1	0	0
0	1	0	0	1	0
1	0	1	1	0	1
1	0	1	1	0	1
0	0	0	0	0	0

We performing Logical AND operation for A1*, A2*, B1*, B2* and C1*, C2*, E1* using BT. Simply A1*, B1* and C1*, E1* itemset measured. Because total value of the Logical AND operation satisfy the min_support count value. But (A1*, A2*, B1*), (A1*, A2*, C1*), (A1*, A2*, C2*), (A1*, A2*, E1*) itemset are not considered. For the reason that the value not satisfy the min_support count value. We can add A1*, B1* and C1*, E1* itemsets to L_k frequent item sets. After that we find 3-itemset at level-2. Shown below.

3-itemset

Table13: ST

	B1*	C1*	C2*	E1*
(A1*, A2**)	0	0	0	0

Table14: ST

	C1*	C2*	E1*
(A1*, B1*)	2	0	3

(A1*, B1*, C1*) & (A1*, B1*, E1*) itemset are consider for further process. Now we go third level.

Level-3

Min_Support = 2.0

1-itemset

A11 = {4}
B11 = {6}
C11 = {4}
E11 = {3}

2-itemset

Table15: ST

	A11	B11	C11	E11
A11		3	2	2
B11			4	3
C11				2

(A11), (B11), (C11) and (E11) items consider further process. Because the value satisfy the min_support count value. Again we BT for third level.

Again we find 3-itemset at level-3

Table16: ST

	C11	E11
(A11, B11)	**2**	3

Table17: BT

A11	B11	C11	E11
1	**1**	**0**	**1**
0	**1**	**0**	**0**
0	**1**	**1**	**0**
1	**0**	**0**	**0**
0	**0**	**0**	**0**
0	**1**	**1**	**0**
0	**0**	**0**	**0**
1	**1**	**1**	**1**
1	**1**	**1**	**1**
0	**0**	**0**	**0**

Finally (A11), (B11), (C11), (E11) itemset connected each other and find maximum frequent itemset at lower level.

5. Conclusions:

In this paper, we have presented a simple and new algorithm (SC-BF Multilevel) for taking out maximum frequent item sets at lower level. The proposed mining algorithm can thus generate large itemsets level by level and then derive association rules from transaction dataset. By utilizing this method in data mining techniques for multi level association rule, the time will be saved with the same accuracy of general approach and our algorithm eliminated repeated scanning of transaction database. The results shown in the example implies that the proposed algorithm can derive the multiple-level association rules under different supports in a simple and effective way.

REFERENCE

[1] J. Han, J, Pei ,Y Yin. “Mining Frequent Patterns Without Candidate Generation,” In ACM SIGMOD Conf. Management of Data, May 2000.

[2]Han, Y**.** Fu, “Mining Multiple-Level Association Rules in Large Databases,” IEEE TKDE. vol.1, pp. 798-805, 1999.

[3] H. Ravi Sankar, Dr. M.M. Naidu, “An Innovative Algorithm for mining multilevel association rules,” Proceeding of the 25th IASTED International multi-conference Artificial intelligence and applications February 12-14, 2007, Innsbruck Austria.

[4] Dr. Mahesh Motwani Dr. J.L. Rana Dr R.C Jain, “Use of Domain Knowledge for Fast Mining of Association Rules,” Proceedings of the International Multi Conference of Engineers and Computer Scientists,Vol.1, IMECS, March 18 - 20, 2009, Hong Kong

[5] S.P. Latha, Dr. N. Ramaraj, “Algorithm for Efficient Data Mining”, International Conference on Computational Intelligence and Multimedia Application 2007.

[6] N. Rajkumar, M.R. Karthik, S.N. Sivananda, “Fast algorithm for mining multilevel association rules,” IEEE Trans. Knowledge and Data Eng., Vol. 2, pp. 688-692 , 2003.

[7] R. Agarwal, R. Srikant, “ Fast Algorithms for Mining Association Rules,” Proc. 1995 Int’l Conf. Very Large Data Bases, pp. 487-499, Santiago, Chile, Sept.1994.

[8] R. Agrawal, T. Imielinski, A. Swami, Mining “Association Rules Between Sets of Items in Large Databases,” Proc. 1993 ACM SIGMOD Int’l Conf. Management of Data, pp.207216, Washington, D.C., May 1993.

[9]R.S Thakur, R.C. Jain, K.R.Pardasani, "Fas Algorithm for Mining Multilevel Association Rule Mining," Journal of Computer Science, Vol-1, pp no: 76-81, 2007.

[13] Predrag Stanišić, Savo Tomović, "Apriori Multiple Algorithm for Mining Association Rules," 124X Information Technology and Control, vol.37, pp.311-320, 2008.

[14] Mehmet Kaya, Reda Alhajj, "Mining Multi-Cross-Level Fuzzy Weighted Association Rules," Second IEEE International Conference on Intelligent Systems.vol.1, pp.225- 230, 2004.

[15] Yue XU, Gavin SHAW, Yuefeng LI, "Concise Representations for Association in Multilevel Datasets,"Systems Engineering Society of China & Springer-Verlag, vol.18 (1), pp.53-70, 2009.