American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)

ISSN (Print) 2313-4410, ISSN (Online) 2313-4402

© Global Society of Scientific Research and Researchers

http://asrjetsjournal.org/

An Improved Association Rule Mining Technique Using Transposed Database

Ruchika Yadav[a*], Dr. Kanwal Garg[b]

[a]*Research Scholar,DCSA, Kurukshetra University, Kurukshetra, Haryana, India*

[b]*Assistant Professor, DCSA, Kurukshetra University, Kurukshetra, Haryana, India*

Abstract

Discovering the association rules among the large databases is the most important feature of data mining. Many algorithms had been introduced by various researchers for finding association rules. Among these algorithms, the FP-growth method is the most proficient. It mines the frequent item set without candidate set generation. The setbacks of FP growth are, it requires two scans of overall database and it uses large number of conditional FP tree to generate frequent itemsets. To overcome these limitations a new approach has been proposed by the name TransTrie, it will use the reduced sorted transposed database. After this it will scan the database and generate a TRIE, in the same step it will also compute the occurrences of each item. Then, using Depth first traversal it will identify the maximal itemsets, from which all frequent itemsets are derived using apriori property. It also counts the support of frequent itemsets which are used to find the valuable association rules.

Keywords: Association rule; Transposed Database; Trie; TransTrie; Frequent itemsets.

1. Introduction

Large databases contain numerous hidden information. Data mining is used to extract this information. It emphasize on finding frequent patterns[1].Data mining consists of various techniques like association rules, classification rules, clustering rules, and sequential rules etc [2]. Association rule mining is the most efficient technique to discover hidden or desired pattern among the large amount of data. An association rule [1, 3, 4] implies certain association relationships among a set of objects (such as “occurs together” or “one implies to other”) in a database. According to Agrawal, the formal statement is “Let $I = \{i_1,i_2,\ldots..i_n\}$ be a set of n binary attributes called *items*. Let $D = \{t_1,t_2,\ldots..t_n\}$ be a set of transactions called the *database*. Each transaction in D has a unique transaction ID and contains a subset of the items in I.

----

* Corresponding author.

E-mail address:ruchikaydv@gmail.com

A *rule* is defined as an implication of the form X→ Y where X, Y ⊆ *I* and X∩Y=0. The sets of items (for short *itemsets*) *X* and *Y* are called *antecedent* (left-hand-side or LHS) and *consequent* (right-hand-side or RHS) of the rule". Association rules are based on two measurements which are support and confidence. Support is the probability of an item's occurrence in transaction. If an itemset appears to equal or more than the predefined minimum support then it is frequent. These frequent itemsets are used to generate association rules on the basis of confidence. Confidence is the probability of the rule's consequent that also contain the antecedent in the transaction. All the frequent itemset generation algorithms are based on either with candidate set generation or without candidate set generation approach. This paper is organized in five sections. Section I provides introduction to association rules mining. Section 2 is the related work. The proposed algorithms TransTrie is presented in section 3. Association rule mining from frequent item sets is described in section 4. Section 5 contains the conclusion and after that references are mention.

2. Related Work

To discover association rules, mining of frequent itemsets is the base of the overall process. Various algorithms have been presented by researchers for finding association rules. Apriori[3] is termed as the basic algorithm in data mining field which works on candidate set generation process. To generate candidate sets, it scans the database many times. Treating Apriori as a basic approach many algorithms have been developed- AprioriTID [3], DHP [5], DIC [6], CARMA [7] and Pascal [8]. Apriori based approaches has the problem that they require large number of scans over the database to generate candidate sets and another serious drawback of these techniques is the management of huge amount of data in computer's memory. To avoid these limitations a new approach, FP-Growth [9] was introduced. It adopts the divide and conquers approach to generate frequent itemsets without generation of candidate set. It uses FP-Tree to represent the database, which takes less space and helpful in reducing multiple scans of database. So, FP-Growth is better than Apriori algorithm, in terms of efficiency [10].But it also have some issues, firstly it does scanning of overall database for two times and secondly, it creates a large number of conditional FP tree for the generation of frequent itemsets [11]. Various improved procedures and algorithms have been proposed by different researchers like COFI [12], COFI* [13], MFI [14] and T3A [15]. A new approach is presented in this paper for mining frequent patterns which can reduce the bottleneck of FP-Growth algorithm by using transposition of database and representing it by an advanced data structure Trie [16].

3. Proposed Algorithm (TransTrie)

3.1. Frequent Pattern Tree

A Frequent Pattern Tree (FP-tree) is a data structure, which is used, to compress the huge dataset by converting it into FP tree. It generates frequent patterns without the candidate item set generation [17]. It is based on the divide and conquers strategy. FP-Tree's construction completes in two steps. In first step, it scan database and count support for each item. On the basis of that support it removes the infrequent items and then it sorts remaining frequent items in descending. In next step, first of all, the root node is marked as "NULL" and then it reads one transaction at a time and places it under the root node. Shared items transactions have the same prefix.

Using singly linked lists, pointers are maintained between nodes containing the same item. Generally a node of FP tree consists of three attributes – name of Item, node link and occurrence of item.

3.2. Key Idea

An algorithm TransTrie is presented which creates a reduced sorted transposed database from the given large database. The items will be stored using Boolean values and in this way the database will take less space for storage. This formation will also be helpful in reduction of scanning time. Moreover it will use an efficient data structure, Trie, which will remove the problem of generation of large number of conditional FP tree.

3.3. TransTrie

The proposed algorithm consists of mainly two components - the transposed database and Trie representation of that database. The transposed database contains only frequent items and it will store them in binary format. So, it takes less space in memory and due to its structure it will take less time for scanning. After this the database will be represented in Trie, It is mainly suitable for generation of candidate sets because transactions that have similar items use the same prefix tree. Using depth first traversal, candidates can be obtained easily.

3.4. Example of Trans Trie

It accomplishes its task in two steps. In step 1, it converts the original database into reduced transposed database. In step 2, it represents the database in Trie for frequent item set generation.

Step 1:

A sample transactional database is shown in Table 1.

Table 1: Sample transactional database

Tid	**Transactions**
T1	AC, LAPTOP, TV
T2	MOUSE, REMOTE, UPS
T3	AC, LAPTOP, UPS
T4	AC, LAPTOP, TV
T5	AC, LAPTOP, MOUSE, TV
T6	KEYBOARD, STABILIZER
T7	MOUSE, UPS
T8	REMOTE, UPS
T9	AC, LAPTOP, MOUSE, TV
T10	AC, TV

Calculate the occurrence of each item by scanning the Table 1 and arrange them in decreasing order. If some items have same count, they are sorted alphabetically. This is shown in Table 2.

Table 2: Transaction items with their frequency

Item	Frequency in Transactions
AC	6
LAPTOP	5
TV	5
MOUSE	4
UPS	4
REMOTE	2
KEYBOARD	1
STABILIZER	1

The given minimum support (M_Supp) is 2. The frequency of items KEYBOARD AND STABILIZER are less than defined minimum support. So, these items are not considered for the transposed database. The items existing in a transaction are represented by 1. In this way a reduced transposed database is created which is shown in Table 4.

Table 3: Reduced transposed database

Item	T1	T2	T3	T4	T5	T7	T8	T9	T10
AC	1	0	1	1	1	0	0	1	1
LAPTOP	1	0	1	1	1	0	0	1	0
TV	1	0	0	1	1	0	0	1	1
MOUSE	0	1	0	0	1	1	0	1	0
UPS	0	1	1	0	0	1	1	0	0
REMOTE	0	1	0	0	0	0	1	0	0

Step 2:

In Trie, there are two kinds of nodes. The root node, which is initialize to NULL and the child nodes, which have three fields: Item name, Frequency of item (occurrence count of item) and link of next item of a transaction.

1) Transaction T1: {AC, LAPTOP, TV}.

The ROOT is created and then adds AC as child of ROOT and it will contain AC as item name, Frequency of item as 1 and link of next item i.e. LAPTOP. Add LAPTOP as child of AC and it will contain LAPTOP as item name, Frequency of item as 1 and link of next item i.e. TV. Add TV as child of LAPTOP and it will contain TV as item name, Frequency of item as 1 and there is no link for this because TV is a leaf node for this transaction. This formation is shown in Figure 1.

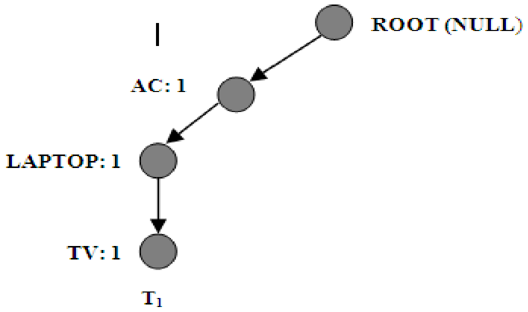

Figure 1: Trie after transaction T1

2) Transaction T2: {MOUSE, UPS, REMOTE}.

Add MOUSE as another child of ROOT and it will contain MOUSE as item name, Frequency of item as 1 and link of next item i.e. UPS. Remaining steps for this transaction are like previous one. This formation is shown in Figure 2.

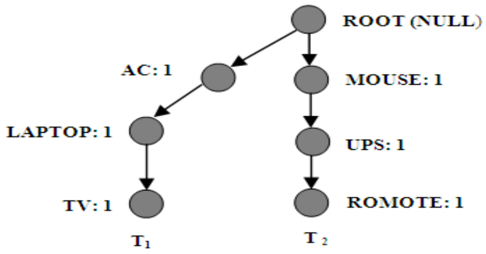

Figure 2: Trie after transaction T2

3) Transaction T3: {AC, LAPTOP, UPS}.

Two items of this transaction i.e. AC and LAPTOP are part of same prefix path which is already exists in Trie.

So, just increase their frequency by 1.UPS is not the part of this prefix path so make it child of LAPTOP with frequency. This formation is shown in Figure 3.

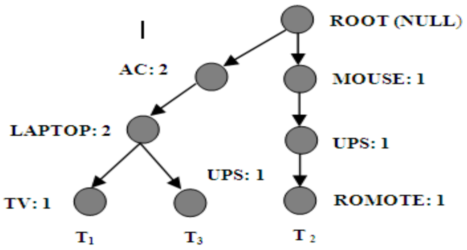

Figure 3: Trie after transaction T3

4) Transaction T4: {AC, LAPTOP, TV}.

Transaction 4 has existing prefix path .So, increase the frequency of its items by 1. This formation is shown in Figure 4.

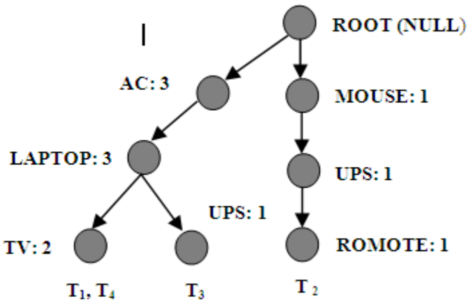

Figure 4: Trie after transaction T3

As there are total 10 transactions in the example so, it will be sufficient to show the precise parts of the proposed algorithm.

5) Transaction T10: {AC, TV}.

Finally, Trie is formed. All the frequent items are shown with their frequency in figure 5.

Figure 5 shows the final representation of the database. Now, it will generate the frequent items sets. For that, first of all it will traverse the Trie using depth first search traversal. After traversal the Maximal itemset with their frequency are given below: -

{AC, LAPTOP, TV, MOUSE: 2}, {AC, LAPTOP, UPS: 1},{AC, TV: 1}, {MOUSE, UPS, REMOTE: 1},{UPS,REMOTE: 1}

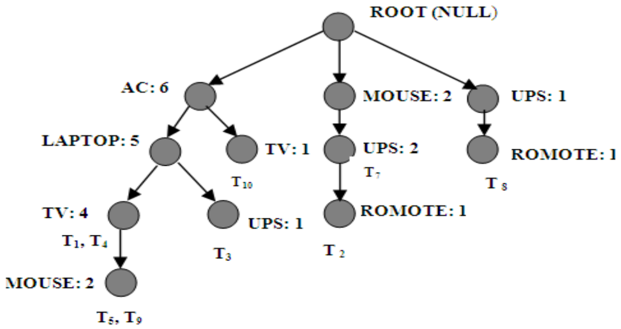

Figure 5: Trie after transaction T10

Now takeout the maximal itemset of any single path one by one and compare its frequency with M_Supp if it qualifies the condition then put all the subsets of maximal itemset in the Frequent Itemset Table i.e. Table 4 otherwise, put all the subsets in the Suspected Itemset Table i.e. Table 5. While putting the subsets in the concerned table, find the frequency of each subset from Trie. To find the frequency of subsets take their values and after confirming the minimum value from these items, assign that to the subset. As it is possible that different paths may contain some same itemset. So, add the frequency of same itemsets of a table. After completion of this step, takeout itemset from suspected table which qualifies the condition of M_Supp and put them in Frequent Itemset Table and add the frequency of same itemsets.

Table 4: FrequentiItemset

Item	Frequent item Set
AC	{AC: 6}
LAPTOP	{LAPTOP: 5}, {LAPTOP, AC: 5}
TV	{TV:5}, {TV,LAPTOP: 5} ,{TV, AC: 5}, {TV, LAPTOP, AC: 4},
MOUSE	{MOUSE: 4},{MOUSE,TV: 2},{MOUSE, LAPTOP:2}, { MOUSE, AC: 2}, { MOUSE, TV, LAPTOP: 2}, { MOUSE, TV, AC: 2}, { MOUSE, LAPTOP, AC: 2},{ MOUSE, TV, LAPTOP, AC: 2}
UPS	{UPS: 4}, {UPS, MOUSE: 2}
REMOTE	{REMOTE: 2}, {REMOTE, UPS: 2}

Table 5: Suspected itemset

Infrequent Itemsets
{REMOTE, MOUSE : 1}
{ REMOTE, MOUSE, UPS : 1}

4. Algorithm

```
Create_Trie(Transposed Database)
{
    for each transaction do
    {
        Insert_In_Trie(Tid)
    }
}
Get_Large_Itemset(Trie, M_Supp)
{
    for each path in Trie
    {
       Si= Create maximal itemset by DFS traversing
       for each subset of Si
         {
            if (subset.supp>= M_Supp)
              Union with large itemset and increment counters
          Else
              Union with suspected itemset and increment counters
       }
       if (subset.supp>= M_Supp)
              Union with large itemset and increment counters
       else
      {
          for each itemset in suspected table
          {
              Itemset  Є FIS table.itemset
              Add counters
           }
       }
   }
}
```

5. Association Rule Mining

Following are the resulting association rules with minimum confidence 50%.

R1: AC and TV ⇒ LAPTOP

Confidence= Supp {TV, LAPTOP, AC}/SUPP{TV,AC} = 4/5 = *80% and R1 is selected.*

R2: MOUSE and AC⇒ TV

Confidence=Supp{MOUSE,TV,AC}/SUPP{MOUSE,AC}= 2/2 = *100% and R2 is selected.*

R3: AC and LAPTOP⇒ MOUSE

Confidence=SUPP{MOUSE,LAPTOP, AC}/SUPP{AC, LAPTOP}= 2/5 = *40% and R3 is rejected.*

6. Conclusion And Future Works References

We presented TransTrie a new algorithm for mining frequent patterns. This new algorithm is based on an advanced data structures Trie and initiates the process by first identifying maximal patterns using a depth first traversal approach. This algorithm finds the set of exact maximal patterns using only two I/O scans of the database then generates all frequent patterns with their respective support. It also introduces a new method of counting the supports of candidates based on the supports of other candidate patterns.

References

[1] R. Agrawal, T. Imielinski and A. "Swami, *Mining association rules between sets of items in large databases"*, In Proc. of the ACM SIGMOD Conference on Management of Data, pp. 207-216, 1993.

[2] J. Han and M. Kamber, "*Data Mining: Concepts and Techniques"*, second ed., Morgan Kaufmann, San Francisco, 2006.

[3] R. Agrawal and R. Srikant, "*Fast algorithms for mining association rules"*, The International conference on Very Large Databases, pp. 487-499, 1994.

[4] Jiawei Han and Micheline Kamber, "*Data Mining Concepts and Techniques"*, Harcourt India Private Limited ISBN: 81-7867-023-2, 2001.

[5] J.S. Park, M.S. Chen and P.S. Yu, An *Effective Hash based Algorithm for Mining Association Rules"*, In Proc., 1995 ACM SIGMOD International Conference on Management of Data, pp. 175-186, 1995.

[6] S. Brin, R. Motwani , J.D. Ullman, and S. Tsur, "*Dynamic itemset counting and implication rules for market basket data"*, In Proceedings of the ACM SIGMOD International Conference on Management of Data, vol. 26(2): 255–264, 1997.

[7] C. Hidber, "*Online association rule mining"*, In Proc. Of the ACM SIGMOD International Conference on Management of Data, vol. 28. pp. 145–156, 1999.

[8] Yves Bastide, Rafik Taouil, Nicolas Pasquier, Gerd Stumme and Lotfi Lakhal, "*Mining Frequent Patterns with Counting Inference"*, In proceeding of ACM SIGKDD, pp. 68-75, 2000.

[9] J. Han, J. Pei, and Y. Yin, " *Mining frequent patterns without candidate generation", i*n Proc. ACM-SIGMOD Int. Conf. Management of Data (SIGMOD '96), pp. 205-216, 2000.

[10] Ding Qin and Sundaraj Gnanasekaran, "*Association rule mining from XML data"*, Proceedings of the conference on data mining.DMIN'06.

[11] Mohammad El-Hajj and Osmar R. Zaoıane. , "*COFItree Mining: A New Approach to Pattern Growth within the context of interactive mining"*, in Proc. 2003 International Conf. on Data Mining and Knowledge Discovery (ACM SIGKDD), August 2003.

[12] Mohammad El-Hajj and Osmar R. Zaoıane, *Inverted matrix: Efficient discovery of frequent items in large datasets in the context of interactive minin"g*, *i*n proce. 2003International Conf. on Data Mining and Knowledge Discovery (ACM SIGKDD), August 2003.

[13] Mohammad El-Hajj and Osmar R. Zaoıane, "*COFI Approach for Mining Frequent Itemsets Revisited Mohammad",* DMKD '04 June 13, 2004, Paris, France, 2004.

[14] A.B.M. Rezbaul Islam and Tae-Sun Chung, *"An Improved Frequent Pattern Tree Based Association Rule Mining Technique",* 978-1-4244-9224-4/11, IEEE, 2011.

[15] Zailani Abdulla, Tutut Herawan and A. Norazia, Mustafa Mat Deris, "*A Scalable Algorithm for Constructing Frequent Pattern Tree",* International Journal of Intelligent Information Technologies, 10(1), pp. 42-56, January-March 2014.

[16] F.Bodon and L. Ronyal, *Trie: An Alternative Data Structure for Data Mining Algorithms,* Proceeding in Mathematical and Computer Modeling Vol. 38, pp.739-751, Elsevier Ltd. 2003.

[17] C. Silverstein, S. Brin, and R. Motwani, "*Beyond Market Baskets: Generalizing Association Rules to Dependence Rules*", Data Mining and Knowledge Discovery, vol. 2(1), pp. 39–68, 1998.