Volume 3, Issue 11, November 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Comparison of Various Association Rule Mining Algorithm on Frequent Itemsets

[1]Kanu Patel, [3]Vatsal Shah,
Assistant Professor,
BVM Engg. College, Gujarat,
India

[2]Jitendra Patel,
Lecturer in K.D. Poly Technique,
Patan-Gujarat,
India

[4]Jayna Donga,
Assistant Professor,
MBICT College, Gujarat,
India

Abstract: Association rule mining has attracted wide attention in both research and application area recently. Mining multilevel association rules is one of the most important branch of it. This paper introduces an improved apriori algorithm so called FP-growth algorithm that will help resolve two neck-bottle problems of traditional apriori algorithm and has more efficiency than original one. New FP tree method is provide how it affected multilevel association rules mining is discussed. Experimental result shows that the algorithm has mining efficiency in execution time , memory usage and cpu utilizations that most current one like apriori.

Keyword: Rule mining; Association rules; multilevel association rules; FP tree;

I. Introduction

Association rule mining is introduced by Agrawal et al. [1], which finds interesting association or correlation relationship among a large set of data items. With the increasing amounts of data stored in real application system, the discovery of association relationship attracts more and more attention. Mining for association rules can help in business decision making, and the development of customized marketing programs and strategies. Taking market basket analysis as example, the mining problem can be described as: (1) given a database D of transactions, each transaction is a list of items; (2) find all rules that correlate the presence of one set of items with that of another set of items. Agrawal and Srikant proposed the Apriori [2] algorithm for fast association rule mining. Apriori finds all rules of form XY which satisfy some given support and confidence threshold parameters, where X, Y⊂I, XY=⇒∩φ, and I = $\{i_1, i_2, \ldots, i_m\}$ be a set of unique items in the database D. The support denoted by P(XY) is the percentage of transactions in D that contain XY. The confidence denoted by P(Y|X) is the percentage of transactions in D containing X that also contain Y. For example, "sunshine-bread daisy-milk [support=2%, confidence=60%]" represents "60% of the customers who purchased sunshine-bread also bought daisy-milk, and 2% of all transactions under analysis show that sunshine-bread and daisy-milk were purchased together." ∪∪⇒ However, for many applications, it is not always easy to find strong association rules (satisfying the minimum *support* and *confidence*) among data items at low (primitive) levels of abstraction due to the sparsity of data in multidimensional space. The reason has been discussed in [3]. Several algorithms have been proposed for mining of multilevel association rules [3, 4, 5, 6, 7, 8], the unsolved problem are: (1) lack of adequate support for dynamically required hierarchies; (2) algorithm accuracy and efficiency cannot satisfy real application requirements; (3) the association between different concept levels may be missed.

In order to address the above problems, our strategies are as follows: first, we mine from the prime data items using FP-growth [9, 10] to get the association rules of atomic level, called *atomic* rules. Then we mine multilevel rules which can be further classified to two types: the *same-level* rules with antecedent and consequent at the same level of concept hierarchy, and the *cross-level* rules with antecedent and consequent at different levels of concept hierarchy. The multilevel rules are mined based on analyses of the rules mined from atomic level, instead of using traditional methods which mine from database again. Therefore, it has a good potential to reduce the computational complexity and I/O cost while accurately exploit same-level and cross-level association rules. In addition, our proposed method supports multiple hierarchies which can be dynamically constructed according to the input of user knowledge. The encoding scheme in [3] is adopted to group the association rules generated from atomic level.

II. Preliminary

In this section, we explain the concept of multiple level mining. Afterward, we illustrate how one can compute minimum support for each item.

2.1 Mining multiple level association rules

Previous studies on data mining focused on finding association rules at a single concept level. Mining association rules at multiple concept levels may, however, lead to discovery of more general and important knowledge from data. Relevant item taxonomies are usually predefined in real-world applications and can be represented as hierarchy trees. Terminal

nodes on the trees represent actual items appearing in transactions; internal nodes represent classes or concepts formed from lower-level nodes . A simple example is given in Fig.1.

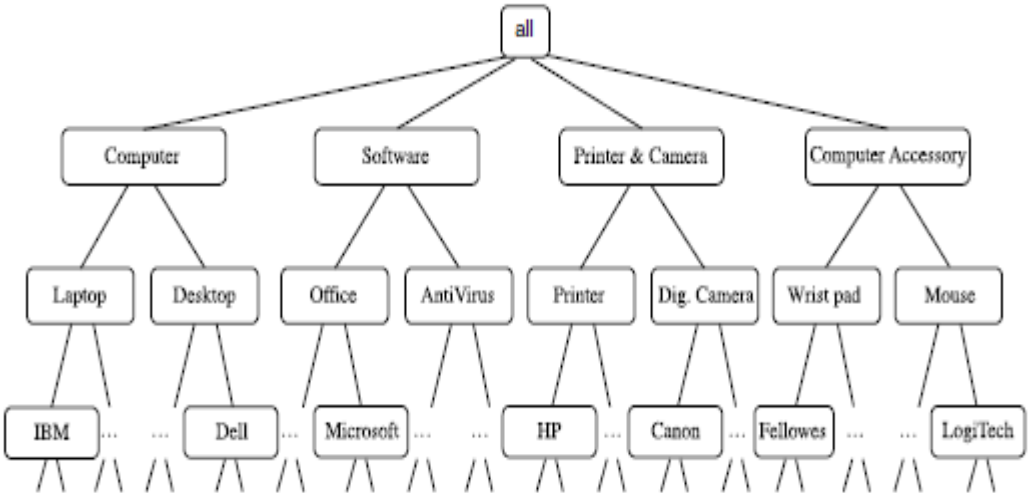

Fig.1 concept hierarchy[6]

The concept hierarchy of Figure 1 has five levels, respectively referred to as levels 0 to 4, starting with level 0 at the root node for all (the most general abstraction level). Here, level 1 includes *computer, software, printer & camera,* and *computer accessory*, level 2 includes *laptop computer, desktop computer*, *office software, antivirus software, . . . ,* and level 3 includes *IBM desktop computer, . . . , Microsoft office software*, and so on. Level 4 is the most specific abstraction level of this hierarchy. It consists of the raw data values. Concept hierarchies for categorical attributes are often implicit within the database schema. A top-down progressively deepening search approach is used and exploration of "level-crossing" association relationships is allowed.

III. Various Rule mining algorithm:

1. Apriori(Candidate set generating) Algorithm

The Apriori Algorithm proposed by Agrawal et. al. in 1994, finds frequent items in a given data set using the ant monotone constraint [5,6]. Apriori is an influential algorithm in market basket analysis for mining frequent item sets for Boolean association rules. The name of Apriori is based on the fact that the algorithm uses aprior knowledge of frequent itemset properties. Apriori employs an iterative approach known as a level-wise search, where k itemsets are used to explore (k+1)-itemsets [9]. First, the set of frequent 1-itemsets is found, denoted by L1. L1 is used to find L2, the set of frequent 2-itemsets, which is used to find L3, and so on, until no more frequent k-itemsets can be found. Property:

All non empty subsets of frequent item sets must be frequent. Apriori algorithm is an influential algorithm for mining frequent itemsets for Boolean association rules. This algorithm contains a number of passes over the database. During pass k, the algorithm finds the set of frequent itemsets Lk of length k that satisfy the minimum support requirement. The algorithm terminates when Lk is empty. A pruning step eliminates any candidate, which has a smaller subset. The pseudo code for Apriori Algorithm is following[6]:

```
Ck: candidate itemset of size k
Lk: frequent itemset of size k
L1 = {frequent items};
For (k=1; Lk != null; k++) do begin
        Ck+1 = candidates generated from Lk;
        For each transaction t in database do Increment the count of all candidates in
Ck+1 that are contained in t
        Lk+1 = candidates in Ck+1 with  min_support
End
Return Lk;
```

In the above example, a frequent itemset with size 3 if finally mined. And its support is higher than the minimum support. With the Apriori algorithm, only frequent itemsets satisfy minimum support threshold can be generated.

2. Frequent pattern growth Algorithm

Apriori needs n+1 scans, where n is the length of the longest pattern， we can use frequent pattern (FP) growth method to reduce the number of scans of the entire database, d to find the frequent itemsets using only two scans of database.

Procedure FP-growth[6]

1) IF Tree contains a single path P THEN
2) FOR all combination (denoted asβ) to the nodes in the path P DO
*3)*generate patter β_αwith support=minsupport of nodes inβ;
4) ELSE FOR all αi in the header of Tree DO
BEGIN
5) generate patter β=αi_α with support=αi.support;
*6) c*onstruct conditional pattern base and generate Treeβ;
7) IF Tree β≠ϕ THEN CALL FP-growth (Treeβ, β);
8) END;

Example: Consider a small database with six items as shown in Table 1, Figure 1 gives the main execution process of FP algorithm. This is illustrated as follows:

Steps for solved example.

1) Scan database and find items with frequency greater then or equal to a threshold.
2) Order the frequent items in decreasing order,
{B5，C4，A3，D3，E3}
3) Construct a tree which has only the root
4) Scan database again; for each sample:
a) add the items from the sample to the existing tree, using only the frequent items (i.e. items discovered in step 1.)
b) Repeat a. until all samples have been process.

TID	Itemset
1	A，B，C，D，F
2	B，C，E
3	A，B，D
4	A，B，F
5	C，D，F
6	E，F

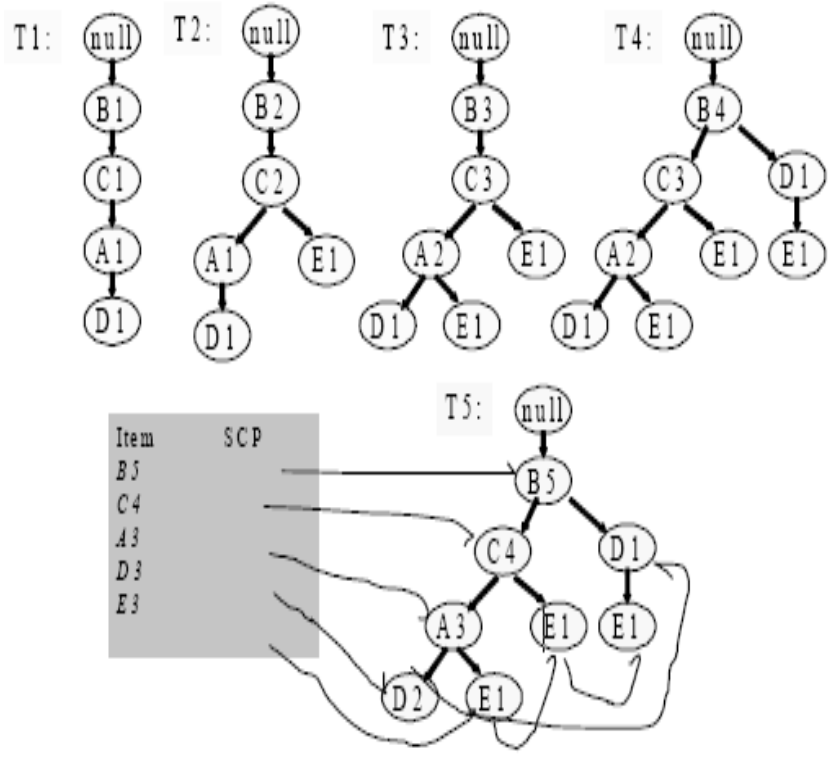

Fig. 2 FP tree of given problem[6]

IV. Experiments & Results

In this section we performed a set of experiments to evaluate the effectiveness of the frequent pattern (FP) growth method, The experimental dataset consists of two kinds of data whose records are set to 10K and 100K . For each experiment, Fig.3 show the total runtime, According to the experimental results, the FP growth method is faster than Apriori Algorithm because FP scans the database at most twice, wherease in Apriori this is not known in advance and may be quite large. From the Fig.4, we can see that the FP based approach is quite effective in reducing memory. Moreover, Let the most original minimum support are 10% 、 20% and 30%, respectively, , the capabilities of FP algorithms are shown in Fig 5.

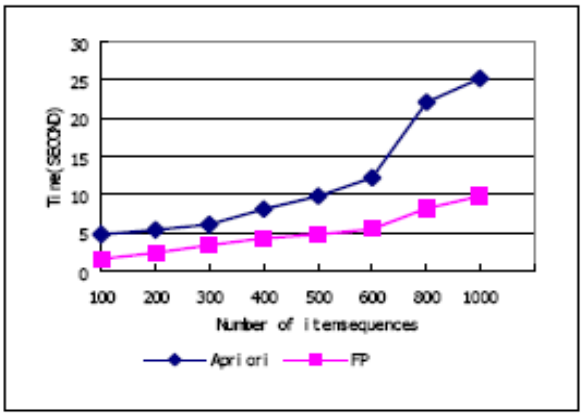

Fig. 3 The total runtime

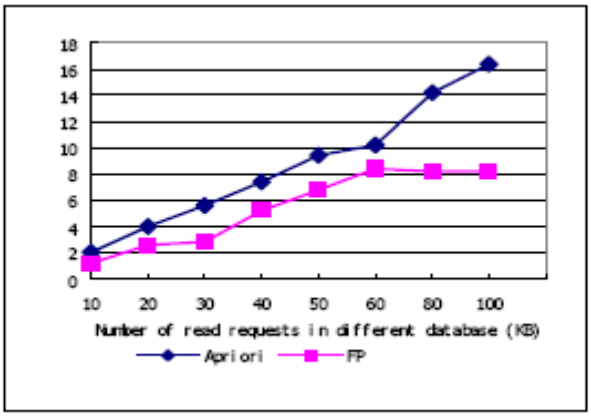

Fig. 4 memory usages

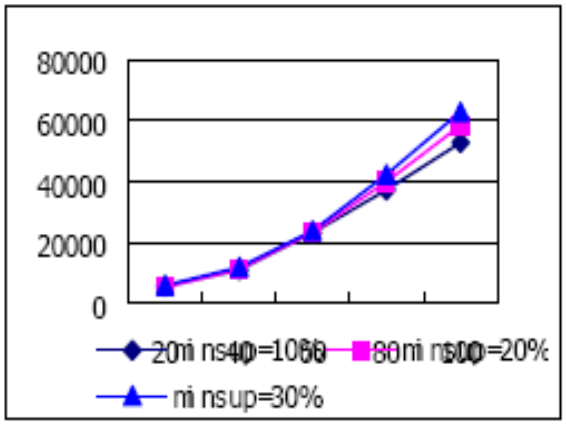

Fig. 5 minimum support

V. Conclusion

In conclusion, this paper analyzes application architecture of data mining algorithm and Association rule mining, creates new mining theoretic models, and designs a new algorithms based on such theories. This technique has been implemented in some parallel compiling system and can get better results for some practical program. Based on experimental result Fp Growth algorithm is better then traditional Apriori algorithm in case of number of rule, cpu time and minimum support where database size should not be large.

References

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

[2] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," Proceeding of the 20th International Conference on VLDB, pp. 478-499, 1994.

[3] J. Han and Y. Fu, "Discovery of multiple-level association rules from large databases", Proceeding of the 21st VLDB Conference, Zurich, Switzerland, pp. 420-431, 1995.

[4] E. Clementini, P.D. Felice and K. Koperski, "Mining multiple-level spatial association rules for objects with a broad boundary", Data and Knowledge Engineering, 34(3), pp. 251-270, 2000.

[5] J. Han, "Mining knowledge at multiple concept levels", Proceeding of In ACM International Conference on Information and Knowledge Management (CIKM'95), Baltimore, Maryland, USA, pp. 19 - 24 , November 1995.

[6] J. Han and M. Kamber, "Data Mining: Concepts and Techniques", Morgan Kaufmann Publishers, August 2000.

[7] K. L. Ong, W. K. Ng and E. P. Lim, "Mining multi-level rules with recurrent items using FP'-Tree", Proceeding of the 3rd Int. Conf. on Information, Communications and Signal Processing, Singapore, Oct. 2001.

[8] S. Thomas and S. Sarawagi, "Mining generalized association rules and sequential patterns using SQL queries", Proceeding of the 4th Int. Conf. on Knowledge Discovery and Data Mining, New York, pp. 344-348, 1998.

[9] J. Han, J. Pei and Y. Yin, "Mining frequent patterns without candidate generation", Proceeding of 2000 ACM-SIGMOD International Conference Management of Data (SIGMOD'00), pp. 1-12, 2000.

[10] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: a frequent pattern tree approach", Data Mining and Knowledge Discovery 8, pp. 53-87, 2004.

