© 2015 IJSRSET | Volume 1 | Issue 1 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

A Comparative Analysis of NFA and Tree-based approach for Infrequent Itemset Mining

Kalaiyarasi. P[*1], Prof. Manikandan. M[2]
[*1,2] Dept of CSE, Adhiyamaan College of Engineering, Hosur, India

ABSTRACT

Frequent itemset mining is an exploratory data mining technique widely used for discovering valuable correlations among data. Frequent itemset mining is a core component of data mining and variations of association analysis, like association rule mining. Extraction of frequent itemsets is a core step in many association analysis techniques. The frequent occurrence of item is expressed in terms of the support count. An item is said to be frequent whose number of occurrences is greater than threshold value. Recently, Infrequent Item sets are also considered to be important in various situations. Infrequent Item set mining is the just the variation of frequent item set mining, where it takes the rarely occurred items in the database. The use of infrequent item set mining is to help for the decision making systems. There are several existing algorithms like Apriori, F-Miner, FP-growth, Residual tree algorithm, Fast algorithm to mine the frequent item sets which takes more computational time. The proposed system is to mine the infrequent item sets by mathematical modelling technique (NFA) where, results in less computing time.

Keywords: Association rule mining (ARM), Itemset mining, Frequent itemsets, Frequent patterns, Infrequent Items, Non-deterministic finite automata (NFA)

I. INTRODUCTION

Data Mining is used to analyze the data from various perspectives and giving it in meaningful information. The meaningful data can be used for enhance the growth, profit, cut costs or both. In data mining, frequent item set is a core component of data mining and variation of association analysis. Frequent item sets are produced from big data or huge amount of data by applying association rule mining. Extraction of frequent item core step in many association analysis techniques. An item is said to be frequent if it presents a large enough portion of the database. It is measured by the occurrence of the item sets in the database.

Anyways, there is a less attention paid to mine the infrequent item sets where it have also acquired important usages in many applications as fraud deduction where rare items in financial pr tax data may suggest unusual activity, image processing, medical fields etc. Patterns that are rarely found in the database are considered to be infrequent item sets.

RELATED WORK

A. Frequent Itemset Mining Techniques:

1) Apriori Algorithm

Apriori [1] was the first proposed algorithm in association rule mining, to identify the frequent item sets in the large transactional database. Apriori works in two phases. During the first phase it generates all possible Item sets combinations. These combinations will act as possible candidates. The candidates will be used in subsequent phases. In Apriori algorithm, first the minimum support is applied to find all frequent item sets in a database and Second, these frequent item sets and the minimum confidence constraint are used to form rules. The main drawback of Apriori is the generation of large number of candidate sets. The efficiency of apriori can be improved by Mono tonicity property, hash based technique, Portioning methods and so on. By using these techniques the efficiency of apriori can be improved, by reducing the candidate generations. Apriori works well when there is less number of item sets. When the transactional size increases it cannot perform well due to candidates set generation. It creates complex structure when mining the frequent patterns and frequent item sets. Some important items may be missed when mining using apriori technique.

2) Transaction Mapping Algorithm

The easy way to differentiate the data sets is to consider the matrix form. The binary matrix consists of <column, key value> pairs.

The transaction tree is similar to FP-tree [2] but there is no header table or node link. The transaction tree has compact representation of all the transactions in the database. Each node in tree has an id corresponding to an item and a counter that keeps the number of transactions that contain this item in this path. Here we can compress transaction for each itemset to continuous intervals by mapping transaction ids into a different space to a transaction tree. Advantage of this algorithm is the performance can be improved compared to FP-Growth, FP-Growth* algorithms.

3) Residual Tree

Ashish Gupta, akshay mittal and arnab bhattacharya[3][4] also proposed minimally infrequent item set mining using pattern growth and residual trees. It considers mining the multiple minimum support item sets for different length of item sets.

Researchers have proposed the Weighted infrequent item algorithm [6][7] reflect the significance of items. Every infrequent weighted frequent item set mining is satisfying the downward closure belongings. A support for each item is usually decreased as the length of an item set is enlarged, but the weight has unusual characteristic. An item set which has a low weight sometimes can get a higher weight after adding another item with a higher weight. It is different from the normal association rule mining algorithms, where it assigns a unique weight for each item in the transactions.

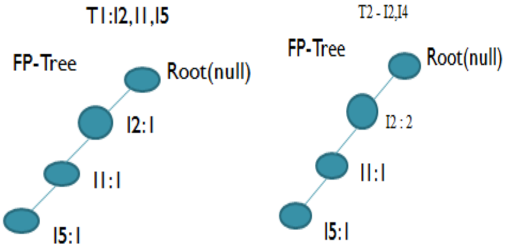
II. METHODS AND MATERIAL

A. FP-Growth Algorithm:

The FP-Growth algorithm follows the divide and conquers strategy. First, it compresses the original database, then generates the tree called frequent pattern tree. Now, divide the compressed database into a set of conditional database. Each and every database must contain the frequent item set.

B. FP-Growth work:

1. Scan the entire database.
2. Derive set of frequent item sets and support count.
   Weight value= $\sum$ w(I, T j)
3. Create the header table.
4. Header table has three fields: Item set, support count, pointer.
5. Based on FCFS the items of same support count are arranged.
6. Now list out the items in header table as lexicographic order.
7. Change the each transaction set to the new ordering based in header table.
8. Now create a FP-Tree, which holds the transaction and support value.



9. Iterate until each transaction is scanned.
10. Create a conditional pattern base.
11. Now, compare with the threshold value
12. Discard the items if the weight is greater than the min-threshold.
13. Determine the infrequent items.
14. Compute the time.

III. RESULTS AND DISCUSSION

A. Computing Time

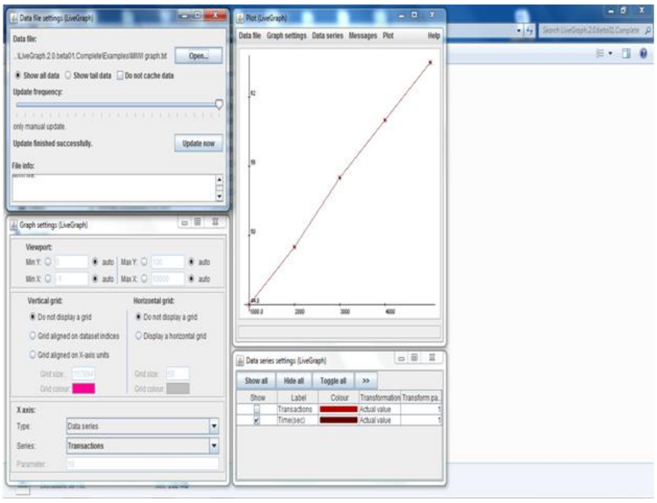
The computing time is calculated with 1k to 5k transactions and took ten distinct items. The graph curve linearly increases when the size of the transactional size increases.

X-axis denotes transactions

It contains 1000 to 5000 transactions.

Y-axis denotes the time (seconds)

It contains 0 to 90 seconds.



Data file:
..\LiveGraph.2.0.beta01.Complete\Examples\MIW graph.txt

● Show all data ○ Show tail data □ Do not cache data

Update frequency:
only manual update.

Update finished successfully.

File info:

Viewport:
Min Y: ○ 0 ● auto | Max Y: ○ 100 ● auto
Min X: ○ 1 ● auto | Max X: ○ 10000 ● auto

Vertical grid:
● Do not display a grid
○ Grid aligned on dataset indices
○ Grid aligned on X-axis units
Grid size: 157894
Grid colour: 🟥

Horizontal grid:
● Do not display a grid
○ Display a horizontal grid
Grid size: 50
Grid colour: ▨

X axis:
Type: Data series
Series: Transactions
Parameter: 10

Show	Label	Colour	Transformation	Transform pa...
□	Transactions	🟥	Actual value	1
☑	Time(sec)	🟥	Actual value	1

Figure 1: Compute time Graph

The graph is plotted by using live graph tool, which is java based open source framework, to plot real time situations. When number of transactions increases the computing time also linearly gets increased.

IV. PROPOSED WORK

The proposed work is mining infrequent weighted itemsets by using mathematical modeling technique (i.e.,) Non-deterministic finite automata.

A. NFA Technique:

In automata theory, a nondeterministic finite automaton (NFA), or nondeterministic finite state machine, is a finite state machine that does not require input symbols for state transitions and is capable of transitioning to zero or two or more states for a given start state and input symbol. This distinguishes it from a deterministic finite automaton (DFA), in which all transitions are uniquely determined and in which an input symbol is required for all state transitions.

The machine starts in the specified initial state and reads in a string of symbols from its alphabet. The automaton uses the state transition function to determine the next state using the current state, and the symbol just read or the empty string. However, the next state of an NFA depends not only on the current input event, but also on an arbitrary number of subsequent input events. If, when the automaton has finished reading, it is in an accepting state, the NFA is said to accept the string, otherwise it is said to reject the string. The set of all strings accepted by an NFA is the language the NFA accepts. This language is a regular language. For every NFA a deterministic finite automaton (DFA) can be found that accepts the same language. Therefore it is possible to convert an existing NFA into a DFA for the purpose of implementing a simpler machine. This can be performed using the power set construction, which may lead to an exponential rise in the number of necessary states.

B. Approach:

The proposed system works with FP-tree structure in combination with the mathematical model. Here it uses the concept of Non-deterministic Finite Automata, to solve the computational problems. The Computing time is expected to be low compared to the existing work, where it involves no complex tree data structures, instead it has number of states as fixed and proceeds with the NFA based technique.

C. Algorithm:

1. Scan the database.
2. The item sets with the weight value is sent as the input.
3. Define the min-threshold value.
4. Compare the total weight with the threshold value.
5. If the item set weight holds greater than the threshold value, enters the final state.
6. If it is smaller than the threshold value, enters in to two states.
7. One is promising state other one is non-promising state.
8. Generate the candidates combinations.
9. Combination is generated by combination generating function.
10. Pass these candidates as input pass to the next state.

11. Calculate the weight value for each of the combinations.
12. Repeat the passes until all the items are scanned.
13. Compute the time.

D. Work flow model of NFA:

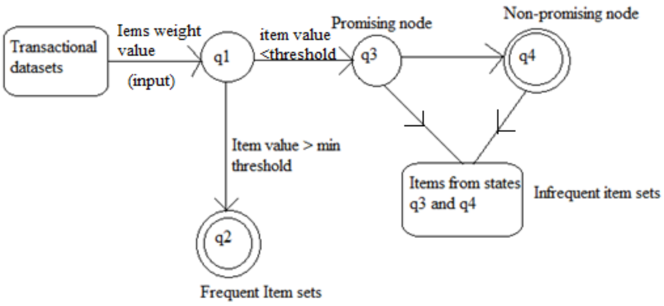


Figure 2: Work flow of NFA

NFA expression:
The NFA consists of five sets namely,
S--->Finite set of states.
∑--->Input alphabet.
§--->transition function.
S_0--->Initial state.
F--->Accepting state.

Figure 2 shows that q1 is the initial state and q3 is the intermediate state and q2, q4 are the final states and hence the grammar or regular will be accepted in the accepting state.

E. Mining Infrequent Itemsets – NFA

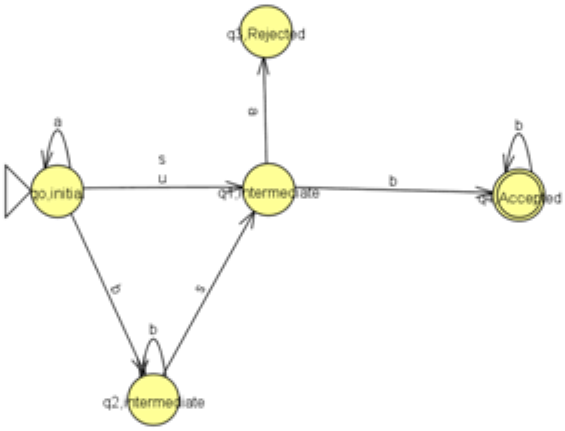


Figure 3 State transition diagram of NFA

Regular Expression:
$(a/b)^m$ (s/u) $(a/b)^n$

Where m<=5 and n<=4
State transition Table:
For the figure 3, the transition table has been created.

Table 1: State transition table

Q	A	b	s	u
q_0	q_0	q_2	q_1	q_1
q_1	q_3	q_4	-	-
q_2	-	q_2	q_1	-
q_3	-	-	-	-
q_4	-	q_4	-	-

Regular Expression for the following grammar:

§$(q_{0,}$ aaabsbbbb)
├ $(q_{0,}$ aabsbbbb)
├ $(q_{0,}$ absbbbb)
├ $(q_{0,}$ bsbbbb)
├ $(q_{2,}$ sbbbb)
├ $(q_{1,}$ bbbb)
├ $(q_{4,}$ bbb)
├ $(q_{4,}$ bb)
├ $(q_{4,}$ b) = b (Accepted)

The input alphabet “b” reaches the final state and hence the grammar is accepting by the machine.

V. CONCLUSION

In this paper, have presented a novel algorithm for mining infrequent weighted itemsets form large real-time databases. The proposed algorithm uses mathematical model- NFA to mine the infrequent data items using some strategies in order to prune frequent item sets and to find infrequent weighted item sets. our proposed algorithm scales well and forms a non-linear curve i.e., even when the number of transactions or number of distinct items increases the computing time varies slightly, unlike previous algorithms which shows a linear variation between computing time and performance parameters.

VI. REFERENCES

[1] Agrawal , R. , Imieliński , T. , & Swami , A.”Mining association rules between sets of items in large

databases". In proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pages 207-216, Washington, DC, 1993.

[2] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proc. 20th Int'l Conf. Very Large Data Bases (VLDB '94), pp. 487-499, 1994.

[3] Luca Cagliero and Paolo Garza "Infrequent Weighted Itemset Mining using Frequent Pattern Growth", IEEE Transactions on Knowledge and Data Engineering, pp. 1-14, 2013

[4] Liu,G. , Lu ,H. , Yu ,J. X., Wang, W., & Xiao, X.. "AFOPT:An Efficient Implementation of Pattern Growth Approach", In Proc. IEEE ICDM'03 Workshop FIMI'03, 2003

[5] J.Han, M.Kamber. Data Ming Concepts and Techniques, Second Edition. Morgan Kaufmann Publisher, Aug. 2000.

[6] G.Grahne, J.Zhu. Efficiently Using Prefix-trees in Mining Frequent Itemsets. In ICDM'03, 2003.

[7] G.Liu, H.Lu, W.Lou, et al. Efficient Mining of Frequent Patterns Using Ascending Frequency Ordered Prefix-Tree. In DASFAA, 2003.

[8] Frans Coenen, Paul Leng & Shakil Ahmed, 2004 'Data Structure for Association Rule Mining: T-Trees and P-Trees', IEEE Transactions on Knowledge and Data Egineering, vol.16, no.6.