ISSN:0975-9646

Comparative Study of Association Rule Mining Algorithms with Web Logs

Archita Bonde[#1], Deipali V. Gore[*2]

[#1]*M.E. Scholar, Dept of Computer Engineering, PES's Modern College of Engineering, Pune, Maharashtra, India*
[*2]*Associate Professor, Dept of Computer Engineering, PES's Modern College of Engineering, Pune, Maharashtra, India*

***Abstract*- Association rule mining is one of the most important aspects of data mining. It aims at searching for interesting relationships among items in a large data set or database and discovers association rules among the large no of item sets. The importance of ARM is increasing with the demand of finding frequent patterns from large data sources. Researchers developed a lot of algorithms and techniques for generating association rules. Among the existing techniques Apriori and frequent pattern methods are the most efficient and scalable options. Apriori algorithm mines the frequent item set by generating the candidate data set. But this takes lot of time and space. To overcome this drawback FP growth algorithm is introduced which mines the frequent items without generating candidate data set. But the obstacle is it generates a massive number of conditional FP trees. An alternate approach to this is systolic tree due to its faster execution and high throughput. In this paper the comparative study of these association rule mining algorithms is carried out on a real time market-basket analysis.**

***Keywords*- Association Rule, Data Mining, Apriori, Frequent Pattern Growth, FP-Tree, Systolic Tree, Frequent Itemset.**

I. INTRODUCTION

To extract the useful and required information from huge databases is very important in many processes related to decision making. This task is known as Data Mining or Knowledge Discovery in Databases (KDD). Association Rule Mining [1] is one of the Data Mining tasks. The goal of ARM is to find interesting association or correlation relationships among a large data set of items. Much effort has been put into looking for highly efficient ARM algorithms. It is an important data mining model studied extensively by the database and data mining community. Association Rule Mining assumes all data is categorical. It mines frequently occurring itemsets in a given database. Generally Apriori is used to mine these itemsets. But it is observed that there are certain drawbacks to this technique. So, alternative algorithms are searched to recover these drawbacks. Fp-growth is one of these algorithms which overcome the drawbacks of Apriori. But this technique also has some drawbacks. To overcome these drawbacks systolic tree approach is used.

II. RELATED WORK

The algorithm for Association Rule Mining is an important research field on KDD presented firstly by R. Agrawal and R. Shrikant in 1993 and under the concept of Fast algorithms for mining association rules in 1994 [1]. Mining frequent item sets is the main focus of many data mining applications for eg. Discovery of association rules etc. Apriori algorithm and its improved versions were generally used to extract frequent item sets previously. The main problem with these algorithms is it generates number of candidate item sets. Therefore it needs to scan the original database several times. This cuts off the mining efficiency of these algorithms. To improve Apriori in terms of time and space , J. Han, J. Pei and Y. Yin presented an algorithm called FP-growth [2]. It extracts frequently occurring item sets without generating candidate item sets which only need to scan the database twice. But FP-growth was originally designed from a software developer's perspective and uses recursion to traverse the tree and mine patterns. For high throughput and faster execution, S. Sun and J. Zambreno introduced the concept of data mining using systolic tree [3] in 2008.

III. ASSOCIATION RULE MINING BASIC CONCEPTS

Association Rule Mining is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness. Based on this concept, R. Agrawal introduced association rules for discovering regularities between products in large scale transaction data recorded by point-of-sale system in supermarket.

The association rules are discovered from databases of the relevant task. For a database DB with n records, each record in the database is called as an item. Let $I=\{I_1,I_2,\ldots I_m\}$ be the item set with m items. The itemset containing k items is called as k-itemset. A transaction T is an itemset such that $T \subseteq I$. Each transaction has its own identifier called TID. An association rule is an implication relationship of the form: $A \Rightarrow B$, in which $A \subset I$, $B \subset I$ and $A \cap B = \phi$.

If the ratio that $A \cup B$ contained in DB is s, we say that the support of the association rule $A \Rightarrow B$ in DB is s, which can also be expressed as the probability $P((A \cup B))$; if the ratio that A and B both contained in DB is c, we say that the confidence of the association rule $A \Rightarrow B$ is c, expressed as conditional probability p(B|A). That is:

$$\text{Support } (A \Rightarrow B) = P(A \cup B)$$
$$\text{Confidence } (A \Rightarrow B) = P(B|A)$$

Association rules [4] have two important properties as follows:

Support: $P(A \cup B)$, which is the probability that A or B both appears in the transaction set D.

Confidence: P(B|A), which is the probability that A and B both appears at the same time in the transaction set D.

Associate rules are called strong if they satisfy both the minimum support threshold and the minimum confidence threshold. The minimum support threshold and minimum confidence threshold are selected by the domain expert. If the support of the itemset satisfies the minimum support threshold, then the itemset is a frequent itemset. The frequent k-itemset is denoted as L_k.

Usually, the association rule mining contains two steps [5]:

1) Finding all frequent itemsets in the database.
2) Using frequent itemsets to generate strong association rules satisfying minimum support threshold and minimum confidence threshold.

IV. APRIORI ALGORITHM

Apriori narrows the range of candidate frequent itemset by applying the principle that a superset of items must not be frequent unless any subset of it is frequent. Through repeated iterations Apriori determines the n-length frequent itemset in the n^{th} iteration. Apriori principle [5] holds due to the following property of the support measure:

- Support of an itemset never exceeds the support of its subsets.
- This is known as the anti-monotone property of support.

Apriori algorithm is actually a layer-by-layer iterative searching algorithm, where k-itemset is used to find (k+1)-itemset. Apriori algorithm firstly scans the database to find out the number of each item. Those item satisfying the minimum support threshold are frequent 1-itemset. Then, a loop of two steps is performed.

A. *Steps of Apriori*

1) *Join Step:* In the join step, different combinations of items in the frequent k-itemset join together to generate the candidate itemset C_{k+1} for (k+1)-itemset L_{k+1}.
2) *Prune Step:* In the prune step, L_{k+1} is generated from C_{k+1} by pruning those combination of items which cannot satisfy the minimum support and minimum confidence thresholds.

The property of Apriori states that all nonempty subsets of a frequent itemset must also be frequent items. This property is used to improve the efficiency of the prune step while the size of C_{k+1} is large. The procedure of the loop runs iteratively until no more frequent (k+1)-itemset could be generated.

B. *Bottlenecks:*

Dimensionality of the database increase with the number of items then:

- More search space is needed and I/O cost will increase.
- Number of database scans is increased. Thus candidate generation will increase results as well as computational cost.

V. FP-GROWTH ALGORITHM

FP-Growth (Frequent pattern growth) uses an extended prefix tree (FP-tree) structure to store the database in a compressed form. FP-growth adopts a divide-and-conquer approach to decompose both the mining tasks and the databases. It uses a pattern fragment growth method to avoid the costly process of candidate generation. FP-growth algorithm [6] scans database altogether two times. The first scanning database, we can obtain the frequent 1-itemset, and the second scanning database, we can get the database's non -frequent itemset by using frequent 1-itemset and FP-tree is generated simultaneously. Last we can obtain the association rule by using FP-tree.

A. *Steps of FP-Growth*

Step 1: Build a compact data structure called the FP-tree. Build it using two passes over dataset.

Step 2: Extracts frequent itemsets directly from the FP-tree.

1) *Step1*: FP-TREE CONSTRUCTION

Pass 1:

- Scan data and find support for each item.
- Discard infrequent items.
- Sort frequent items in decreasing order based on their support. Use this order when building the Fp-tree, so common prefixes can be shared.

Pass 2:

A node corresponds to items and has a counter.

- FP-Growth reads 1 transaction at a time and maps it to a path.
- Fixed order is used, so paths can overlap when transactions share items (when they have same prefix). In this case, counters are incremented.
- Pointers are maintained between nodes containing the same item, creating singly linked list. The more paths that overlap, the higher the compression. FP-Tree may fit in memory.
- Frequent itemsets are extracted from the FP-Tree.

2) *STEP 2*: FREQUENT ITEMSET GENERATION

- Conditional Pattern Base:

Once the header table and the FP-tree are constructed, then for each frequent item in the header table, the conditional pattern base, which is the list of nodes that link the frequent items node in the FP-tree to the root node (NULL), is formed. Each pattern base is assigned a support count which is the minimum of the support counts for the items in the pattern base. The set of items in such single pattern bases form the frequent itemset.

- Frequent Itemset:

In the condition pattern base, every item is scanned and the common content is listed, including an individual item and the combination. At the same time the number of corresponding item is counted and the condition FP-tree is generated. Then FP-tree is connected with the corresponding header able-item, and eventually frequent itemset is generated.

B. Advantages Over Apriori Algorithm

- FP-growths reads database only twice where repeated scans of the database in apriori put a lot of pressure on I/O devices and lead to intolerable performance overhead.
- By storing all transactions in a compact tree, frequent itemsets are explored by recursively traversing the tree and performing a brute force enumeration.
- If the tree is predictably large, the entire database can be projected to produce smaller trees.

C. Bottlenecks

- FP-growth is dependent on the number of items in dataset.
- When the database is too large to build FP-tree in memory, it is difficult to mine favorably.
- The depth of FP-tree is dependent on number of transactions. Also degree of each node is constant i.e. 2.

VI. SYSTOLIC TREE

Systolic tree approach [7] is build the tree based on the principle of maximum node degree estimation. When the actual node degree at some point in the tree exceeds the estimated node degree, some frequent itemsets will not be found. To simplify the complexity, assume there are two interfaces for each node. One is dedicated to the connection with its first child and the other is connected to its nearest sibling.

Each node in systolic tree is referred to as a processing element (PE). Each PE has its own local data structure and corresponding operations upon receiving signals from outside. Depending upon time efficiency any data structure is chosen [8]. There are three types of PE:

- Control PE
- Count PE
- General PE

A. Modes of Operation

1) WRITE mode or Systolic Tree Creation
- To generate systolic tree from the database.

2) SCAN mode or Candidate Itemset Dictation
- To check whether the given itemset is frequent or not.

3) CONTROL mode or candidate Itemset Count Computation
- To operate control signal throughout the tree.

The input and output of the whole systolic tree passes through the control node. The input can be data or the control signals and the output is usually defined by the designer. The letter inside each node represents the item and the number beside the node is the number of times that the item emerges in the transaction database.

Each general PE has one input from its parent two outputs to its child and sibling respectively. Each PE only has a connection with its leftmost child. If it has to send the data to its rightmost child, the data will be passed to its leftmost child and then through its siblings, passing through all the children on the way. The general processing elements which do not contain any item, are empty. If the items in one transaction are transferred into the architecture in an ascending order, any PE must contain a smaller item than that of its children. However, this does not guarantee that the node item in the systolic tree is always greater than its left-side siblings.

B. Advantages Over FP-Growth

- High throughput and faster execution are the major advantages of systolic tree approach.
- Systolic tree have dynamic nodes with local data structure where FP-tree have static nodes.
- The depth of systolic tree is already decided which improves memory efficiency.
- Maximum degree estimation is the principle of Systolic tree approach.

VII. COMPARISION WITH RESPECT TO WEB USAGE MINING

Web mining [7] is often categorized into three major areas: web content mining, web structure mining and web usage mining. Web content mining is the process of extracting knowledge from the content of a number of web documents. Web structure mining is the process of inferring knowledge from the organization and links on the web, while web usage mining is the automatic discovery of user access patterns from web servers.

Web usage mining [9] is to apply the concept of data mining to the web log file data, and automatically discover user access patterns towards a specific web page. Another source for the web usage mining is referrer logs, which contain information about the referring pages for each page reference, and user registration or survey data gathered via CGI scripts. Results of the web usage mining provide decision makers with crucial information about the lifetime value of customers, cross-marketing strategies across products, and effectiveness of promotional campaigns. Among other things, the web usage mining helps organization analyze user access patterns to targeting ads or web pages, categorize user preferences, and restructure a web site to create more effective management of workgroup communication and organizational infrastructure.

Let's take an example of a dataset which contains IP address and the web pages visited from that particular IP. The purpose of mining this dataset is that we get the frequently occurring group of websites i.e. the websites which are visited most of the times. The use of this output is that we can recommend one site to another site user by advertisement, helps making many design decisions depending on one another.

TABLE 1
WEB LOGS

IP	WEBSITES
210.55.01.111	Amazon, Facebook, Lenskart, Jabong
210.55.13.546	Amazon, Facebook, eBay, Snapdeal
210.92.43.731	Twitter, Facebook, Lenskart, Jabong, Olx
210.63.52.497	Twitter, Facebook, Lenskart, Snapdeal
210.86.27.100	Twitter, Amazon, Facebook, Snapdeal, Flipcart
210.21.76.903	Twitter, Amazon, Facebook, Jabong, eBay
210.56.78.342	Twitter, Amazon, Lenskart, Olx

Table 1 shows fragment of a dataset i.e. web log. It contains IP address and the sites visited by that IP address in a particular day. For easier approach, each transaction is

given a unique ID i.e. TID and the sites are represented as alphabets, referred as items. A for Twitter, B for Amazon, C For Facebook, D for Lenskart, E for Jabong, F for eBay, G for Snapdeal, H for Olx and I for Flipcart. Now the dataset will look like this:

TABLE 2
TRANSACTIONAL DATASET

TID	ITEMS
101	B, C, D, E
102	B, C, F, G
103	A, C, D, E, H
104	A, C, D, G
105	A, B, C, G, I
106	A, B, C, E, F
107	A, B, D, H

Now, apply every approach to this dataset.

A. Apriori Algorithm

C1 is obtained by counting support frequency of each item in the dataset shown in table 3 and L1, shown in table 4, is obtained by reducing C1 after deleting all the entries which contains support frequency less than minimum support, which is 4 in this case.

TABLE 3
C1

Item	Support Frequency
A	5
B	5
C	6
D	4
E	3
F	2
G	3
H	2
I	1

TABLE 4
L1

Item	Support Frequency
C	6
A	5
B	5
D	4

C2, shown in table 5, can be obtained by Cartesian product of L1. Again L2 is obtained by removing entries below minimum support, shown in table 6.

TABLE 5
C2

Item	Support Frequency
C, A	4
C, B	4
C, D	3
A, B	3
A, D	3
B, D	2

TABLE 6
L2

Item	Support Frequency
A, C	4
B, C	4

L2 gives frequently occurring itemsets. Therefore, {A, C} and {B, C} are most frequently occurring itemset in given dataset.

B. FP-Growth Algorithm

The dataset is constructed in the form of a tree. Root node is null and all the items are leaf nodes. For simplicity, only {A, B, C, D} items are considered. Figure shows an FP-tree that contains all the information of the transactions in the database. The number in each node represents the times the prefix from the root item to it occur in the transaction database. The level of a node is the number of the nodes from it to the root. The higher the level of a node, the larger length of the prefix it represents. Because two transactions can only share a path when one is a prefix of another, the number of the higher level nodes is never greater than that of the lower level nodes.

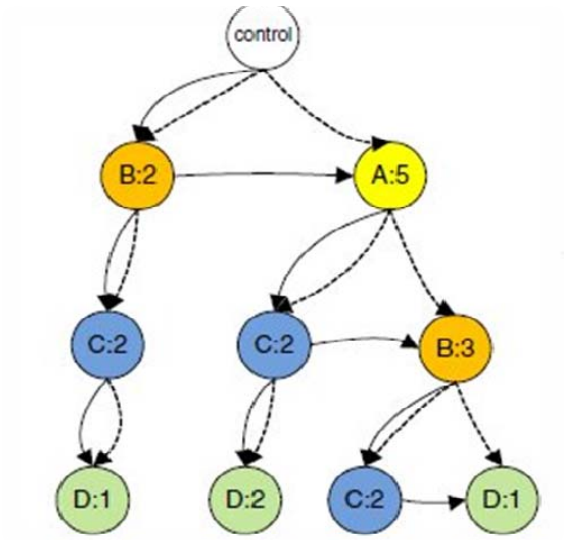

Fig. 1 FP-Tree of the given transactional dataset

The FP-growth algorithm extracts the frequent itemset in a recursive enumeration manner. Starting from the highest level of the FP-tree toward the root, the itemsets ending with item D are checked first. This problem is further divided into several sub problems of finding frequent itemset ending with { C,D} ,{ B,D} and { A,D} . Each sub problem ending with itemsets is further divided into smaller problems by building its conditional FP-tree. Then based on conditional pattern base frequent itemsets are obtained.

TABLE 7
CONDITIONAL PATTERN BASE

Head Item	Conditional Pattern Base
B	A:3
C	A:2, B:2, AB:2
D	AB:1, AC:2, BC:1

TABLE 8
FREQUENT ITEMSET

Head Item	Conditional Pattern Base	Conditional FP-Tree	Frequent Itemset
D	AB:1, AC:2, BC:1	A:3, B:2, C:3	AD:3, CD:3
C	A:2, B:2, AB:2	A:4, B:4	AC:4, BC:4
B	A:3	A:3	AB:3

Thus, {A, C} and {B, C} are the two most frequent itemsets as it has highest support frequency.

C. Systolic Tree Approach

For the given dataset, the interfaces k=2 and depth w=3 is already decided unlike FP-growth. All modes are applied to this dataset.

1) WRITE mode: When the tree is in building phase, PEs are in WRITE mode. An item is loaded into the control PE each cycle which in turn transfer each item into the general PEs. If the item is already contained in PE, the corresponding count value will be increased. Otherwise an empty PE will be allocated to it.

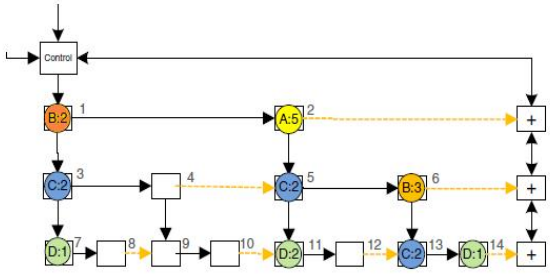

Fig. 2 Systolic tree creation

In order to clearly differentiate PEs a number in light scale is placed in the top-right corner. Suppose a new transaction A, B, D is to be added into the systolic tree, a control signal which indicates the WRITE mode is first broadcasted from the control PE. Then the transaction is sent to the control PE sequentially.

2)SCAN mode: This approach is also called as Candidate Itemset Dictation. When we want to check whether a given itemset is frequent or not, it is sent to the systolic tree. The number of itemset will be obtained in the output of the systolic tree after some clock cycles. When the tree is in itemset dictation phase, PEs are in SCAN mode. In this systolic tree structure there is only one path tracing back from any PE to the control PE since each PE has a unique parent. The main principle of dictation is that any path containing the queried candidate itemset will be reported to the control node.

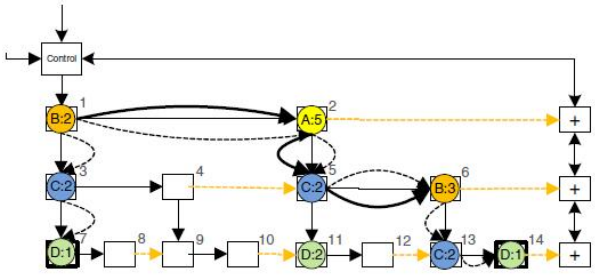

Fig. 3 Frequent itemset

3) COUNT mode: Once all items in a candidate itemset are sent to the systolic tree, a control signal signifying the COUNT mode is broadcasted to the whole systolic tree. The architecture of the systolic tree will change accordingly with response to the COUNT mode signal. We have shown that the first child's input interface is always connected to its parent while others accept input from the first child in WRITE and SCAN mode. In COUNT mode, all PEs receive input from its leftmost neighbor except the first PE in each row. The PEs with flag set is ready to send their own item count and forward the count from their leftmost neighbor in a pipelined fashion when all PEs are in COUNT mode. The counting PEs on the rightmost column always enters the COUNT mode earlier than the general PEs.

The FP-growth algorithm uses a divide and conquers approach to generate frequent itemset by searching the tree in a bottom up fashion. From the hardware perspective, the candidate generation method used in Apriori is more suitable for dictation. There are several candidate generation procedures. The Fk−1×Fk−1 method is well suited for systolic tree. A new candidate itemset with k items is dictated only if it is generated from two frequent itemsets with k−1 items whose first k − 2 items are identical. This approach will decrease the number of candidate itemset dramatically.

VIII. CONCLUSION

In this paper, classic association rule mining algorithms, Apriori, FP-growth are compared. Another approach called systolic tree is also compared to both the algorithms. Each algorithm gives advantages over the bottlenecks of previous algorithm. Systolic tree approach seems best amongst them. Also all the algorithms are applied to web log.

REFERENCES

[1] R. Agrawal and R. Srikant," Fast algorithms for mining association rules," in *Proceedings of the 1994 International Conference on Very Large Data Bases (VLDB)*, 1994.

[2] J. Han, J. Pei, Y. Yin and R. Mao," Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach," in *Data Mining and Knowledge Discovery, vol. 8, no. 1, pp. 53-87*, Jan 2004.

[3] S. Sun and J. Zambreno," Mining Association Rules With Systolic Tree," *Proc. International Conference Field-Programmable Logic and Applications (FPL '08)*, September 2008.

[4] R. Agrawal, Tomasz Imielinski, Arun Swami," Mining association rules between sets of items in larger databases," *Proceedings of ACM SIGMOD International Conference of Management of Data, Washington DC*, May 1993, 207-216.

[5] L. Wang, J. Fan, L. Liu, H. Zhao," Mining Data Association Based on a Revised FP-Growth Algorithm," *Proc. of the 2012 International Conference on Machine Learning and Cybernetics, Xian*, July 2012.

[6] R. Narayanan, D. Honbo, G. Memik, A. Choudhary and J. Zambreno. "An FPGA Implementation of Decision Tree Classification," *Proc. Conf. Design, Automation and Test in Europe (DATE)*, pp. 189-194, April 2007.

[7] P. Sampath, C. Ramesh, T. Kalaiyarasi, S. Sumaiya Banu and G. Arul Selvan," An Efficient Weighted Rule Mining For Web Logs Using Systolic Tree," in *IEEE International Conf. of Advances in Engineering, Science and Management (ICAESM-2012)*, March 2012.

[8] Song Sun and J. Zambreno," Mining Association Rules With Systolic Trees."

[9] Liu Jian and Wang Yan-Qing." Web Log Data Mining Based on Association Rule," in *Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, 2011.