PERFORMANCE EVALUATION OF ASSOCIATION RULE MINING ALGORITHMS

Bhawna Nigam[1], Preeti Dalal[2]
[1]Department of Information Technology, [2]Department of Computer Engineering,
Institute of Engineering & Technology, DAVV Indore

Abstract

Data has become a very important asset in today's world. Several companies, organizations, people and various other sources generate large volumes of data every second that is now a days being processed by smart systems. Thus in order to gain some significant insight from such huge amount of data, several data mining algorithms are employed. Market basket analysis is one important field where association rules [1] play a major role. Whether we talk about the products on a shelf in an actual mall or we consider the online shopping portals, market-basket analysis significantly helps improving the business by providing the sellers and such e-commerce websites with the common buying patterns, which when applied result in increased sales. So, in order to generate these association rules – one needs to perform frequent item set mining. There are several algorithms proposed for performing frequent pattern mining. And in this paper we have compared four well known algorithms: Apriori, FP-Growth, Eclat and dEclat – on the basisof performance factors such as memory usage, processing time, dataset and support.

Keywords: Data mining, Frequent item set mining, Association rules, Support and Confidence, Apriori Algorithm, FP-Growth Algorithm, Eclat Algorithm, dEclat Algorithm, Transactional Databases

I. INTRODUCTION

Association rules are basically if and then statements that help in understanding the relationships between unrelated data that is stored either in a non-relational or relational database or may belong to some other information repository. Association rules help in discovering the relationships between the objects that are used together, quite frequently. The task of association rule mining can be broken into two steps: 1. Frequent itemset generation[2] and 2. Confidence Rules Generation. In this paper we are majorly focusing upon the techniques used in frequent itemset mining. Basically, frequent itemset mining focuses at sequences of actions or events. Here, the database takes the form of sets of transactions where each transaction has a number of items. And then there is a minimum threshold support value and user-specified minimum confidence, define as:

Support(S):-Support(S) of an association rule is defined as the percentage or the fraction of records that contain X∪Y to the total number of records in the database. Suppose the support of an item is 0.2%, it means only 0.2 percent of the transaction contains purchase of this item.

Confidence(C):- Confidence(C) of an association rule can be defined as the percentage or the fraction of the number of transactions that contain X∪Y to the total number of records that contain X. Confidence [3] basically measures the strength of the association rules, for an instance, if the confidence of the association rule X⇒Y is 80%, then it means that 80% of the transactions containing X also contain Y together.

$$Rule: X \Rightarrow Y \quad \begin{cases} Support = \dfrac{frq(X,Y)}{N} \\ Confidence = \dfrac{frq(X,Y)}{frq(X)} \end{cases}$$

Let us consider the following example that consists of a transaction database and the frequent itemsets and association rules generated. This a transaction database of a bookstore sales. There are five different items (names of authors the bookstore has), i.e., I = {A, C, D, T, W}, and the database consists of six customers who bought books by these authors. The Figure 2 shows all the frequent itemsets that are contained in at least three customer transactions, i.e., the min sup = 50 percent. Further it shows the set of all association rules with minimum confidence = 100 percent. The itemsets ACTW and CDW are the maximal frequent itemsets. Since all other frequent itemsets are subsets of one of these two maximal itemsets, we can reduce the problem of frequent itemset search to the task of enumerating only the maximal frequent itemsets. On the other hand, for generating all the confident rules, we need the support of all frequent itemsets. This can be easily accomplished once the maximal elements have been identified by making an additional database pass and gathering the support of all uncounted subsets.

ITEMS

Jane Austen	A
Agatha Christie	C
Sir Arthur Conan Doyle	D
Mark Twain	T
P. G. Wodehouse	W

DATABASE

Transaction	Items
1	A C T W
2	C D W
3	A C T W
4	A C D W
5	A C D T W
6	C D T

Figure 1. Bookstore Database

There are several algorithms to calculate the frequently occurring items, however, we will be discussing the following four:

- Apriori Algorithm
- FP-Growth Algorithm
- Eclat Algorithm
- dEclat Algorithm

FREQUENT ITEMSETS (min_sup = 50%)

Support	Itemsets
100% (6)	C
83% (5)	W, CW
67% (4)	A, D, T, AC, AW CD, CT, ACW
50% (3)	AT, DW, TW, ACT, ATW *CDW, CTW, ACTW*

Maximal Frequent Itemsets: *CDW, ACTW*

ASSOCIATION RULES (min_conf = 100%)

A → C (4/4)	AC → W (4/4)	TW → C (3/3)
A → W (4/4)	AT → C (3/3)	AT → CW (3/3)
A → CW (4/4)	AT → W (3/3)	TW → AC (3/3)
D → C (4/4)	AW → C (4/4)	ACT → W (3/3)
T → C (4/4)	DW → C (3/3)	ATW → C (3/3)
W → C (5/5)	TW → A (3/3)	CTW → A (3/3)

Figure 2. Frequent Itemsets and Confidence Rules

Each algorithm has a different efficiency and memory requirement as per the intermediate data structure each of the algorithms employs like an FP-tree in FP-Growth algorithm, diffsets in dEclat algorithm and so on. And hence we later compare their performance on distinct factors

II. RELATED WORK

As mentioned above, there are several algorithms in order to generate frequent itemsets and each one of them has their own efficient candidate generation procedure. However, the overall performance depends on the number of database scans and the criticality of the data structure, each form in the intermediate steps. While Apriori seems to be the simplest in terms of working and is efficient for smaller datasets, it shows a poor efficiency when applied on a larger dataset, as it requires multiple database scans [4][5]. When we compare it with FP- Growth Algorithm, FP-Growth has a different candidate generation procedure as it creates an FP-tree. Now, there are different ways of creating the FP-

Tree as well, however when we consider the most standard procedure, we find that tree generation is worthy as the results are obtained comparatively fast. However, memory requirement in order to construct the tree can be a constraint, as in a large dataset, the tree created will also be large as well as complicated. It will take a long processing time for the first run and the later runs could be quick as it can make use of cache. Moving on to the next algorithms – Eclat [12] and dEclat [13] Algorithms, both of them have proven to be far better in performance when compared to the previous two. Moreover, these algorithms have been developed recently and dEclat algorithm is an advanced version of Eclat algorithm which has proven to be really powerful and scalable. Both of these algorithms employ a different layout of the transaction database i.e. vertical layout. And then this database is processed by creating tidsets [14] and diffsets [15] in Eclat and dEclat algorithms respectively. In our research we have applied all four algorithms on three datasets of different sizes. These datasets are bakery items transaction databases of 1000, 2500 and 5000 records that are processed at different threshold support counts.

Apriori Algorithm: Apriori algorithm is used for mining frequent itemsets from the transaction database and in association rule mining. This algorithm makes use of a level-wise search, where k-itemsets (An itemset which contains k items is known as k-itemset) are used to explore (k+1)-itemsets. In this algorithm, a process of candidate generation takes place in which, frequent subsets are extended one item at a time. Then these groups of candidates are tested against the data. In order to count the candidate itemsets efficiently, Apriori employs breadth-first search method and a hash tree structure. It identifies the frequent individual items in the database and extends them to larger and larger item sets as long as those item sets appear sufficiently often in the database. Apriori algorithm ascertains frequent item sets that can be used to determine association rules which further highlight general trends in the database.

Apriori algorithm takes advantage of the fact that any subset of a frequent itemset is also a frequent itemset. Thus, the algorithm can significantly reduce the number of candidates being considered by just exploring the item sets whose support count is greater than the minimum support count. All infrequent item sets can be pruned [4] if they have infrequent subsets. So we build a candidate list of k-item sets and then extract a frequent list of k-item sets using the support count.

After that, we use the frequent list of k-item set in determining the candidate and frequent list of k+1 item sets so we use pruning to do that. We repeat the above procedure until we get to an empty candidate or frequent k-itemsets, in that case we return the list of k-1 item sets.

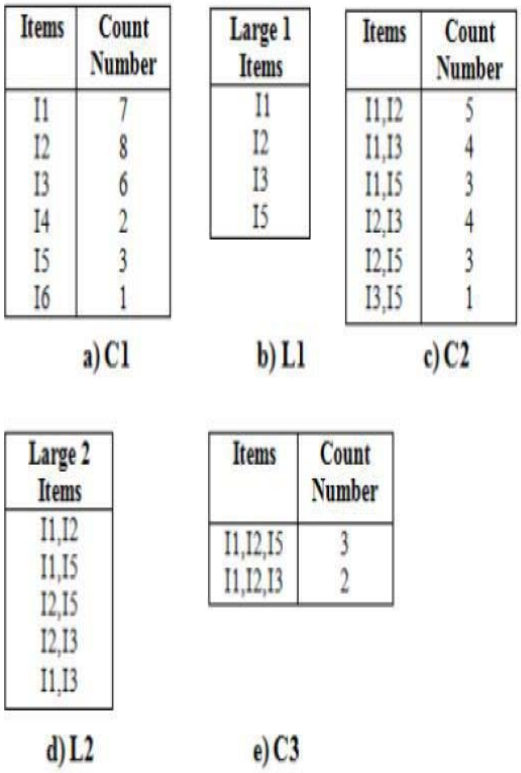

Figure 3. Example of Apriori Algorithm

However simple in implementation, Apriori algorithm has two main drawbacks : First is the complex candidate generation process which is consumes a lot of time, space as well as memory. Another drawback is that it requires multiple scans of the database.

FP-Growth Algorithm: This is another important frequent pattern mining method, which generates frequent item set without candidate generation. Here, the construction of FP-tree takes place. The main idea of the algorithm is to use a divide and conquer strategy:

First step is to compress [6][7] the database so that it provides the frequent sets; then in the next step, this compressed database is divided into a set of conditional databases, where each one of them is associated with a frequent set and then data mining is applied on each database. So now, in order to construct an FP-Tree, the algorithm performs two scans on the database. In the first scan it computes a list of frequent items sorted by frequency in descending order (F-List). In the second scan, the database is compressed into an FP-tree [6]. This algorithm performs mining on FP-tree in a recursive manner. This way the problem of finding frequent item sets is converted into the searching and constructing trees recursively. The frequent itemsets are thus generated with only two passes over the database and no candidate generation process takes place.

The passes work as follows:
Pass 1:
•Scan the data and find support for each item.
•Discard infrequent items.
•Sort frequent items in descending order which is based on their support.

By using this method we can build the FP-tree, so that common prefixes can be shared.
Pass 2:
•Here nodes correspond to items and each has a counter.
•FP-growth reads one transaction at a time and then maps it to a path.
•Fixed order is used, so that paths can overlap when transactions share the items.
In this case, counters are incremented. Some pointers are maintained between nodes which contain the same item, by creating singly linked lists. The more paths that overlap, higher is the compression. FP-tree may fit in memory. Finally, frequent itemsets are extracted from the FP-Tree.

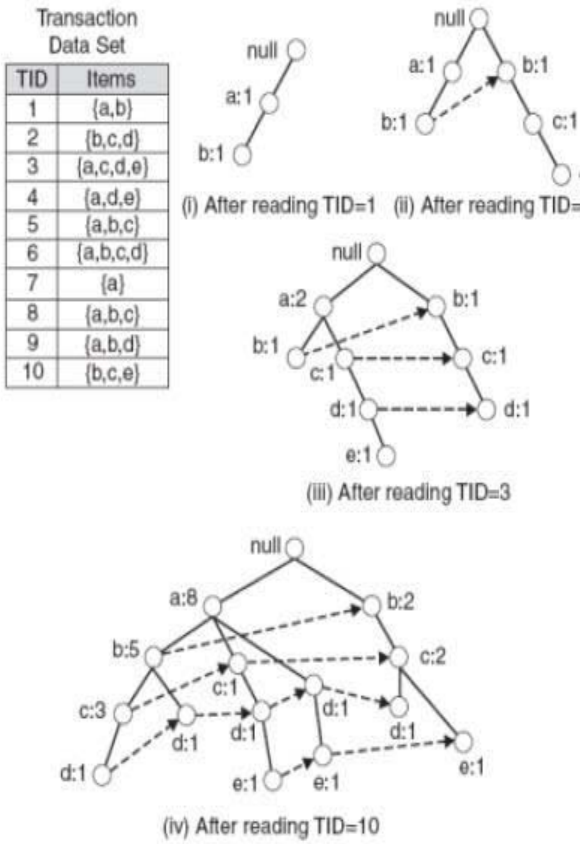

Figure 4. FP-Growth Algorithm

However efficient, it still has a limitation, when it comes to a larger dataset, as an FP-Tree is expensive to build and it may sometimes not fit in the memory.

Eclat Algorithm: Eclat is another efficient algorithm for mining frequent itemsets, which is a depth first search based algorithm [8]. The most distinctive feature of this algorithm is that it uses a vertical database layout i.e. instead of explicitly listing all transactions; each item is stored together with its cover (also called tidlist) and uses the intersection based approach to compute the support of an item set [10]. Now Eclat is again based on two main steps, namely candidate generation and pruning. In the first step of candidate generation, each k-itemset candidate is generated from two frequent (k-1)-itemsets and then its support is counted. If its support value is lower than the threshold, then it will be discarded, otherwise it is a frequent itemset and used to generate (k+1)-itemsets. Since Eclat uses the vertical layout, counting support is trivial. Candidate generation is indeed

a search in the search tree [11]. This search is a depth-first search and it starts with frequent items in the item base and then 2-itemsets are reached from 1-itemsets, 3- itemsets are reached from 2-itemsets and so on. Thus, in this process, specifically tidsets are created – i.e. transaction id sets. The vertical layout leads in the creation of tidsets which are later pruned for pattern counting.

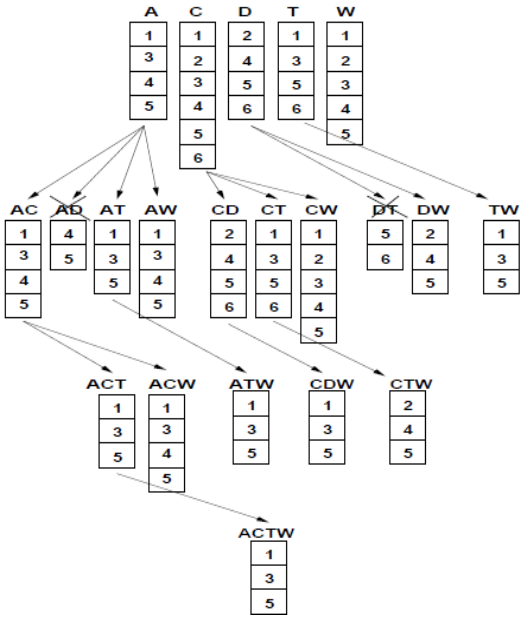

Figure 5. Example of tidsets created in Eclat Algorithm

dEclat Algorithm: dEclat algorithm is an enhanced version of Eclat algorithm that is much more powerful and scalable than the previously discussed algorithms. This algorithm makes use of diffsets rather than tidsets. Unlike tidset which is based on intersection, diffset only keeps track of differences in the tids of a candidate pattern from its generating frequent patterns. To use a diffset[17][18][19] format, the initial transaction database in vertical layout is firstly converted to diffset format in which diffset of items are sets of tids whose transactions do not contain items. They drastically cut down (by orders of magnitude) the size of memory required to store intermediate results. The initial database stored in diffset format, instead of tidsets [20][21] can also reduce the total database size. Since the diffsets are a small fraction of the size of tidsets, intersection operations are performed striking fast!

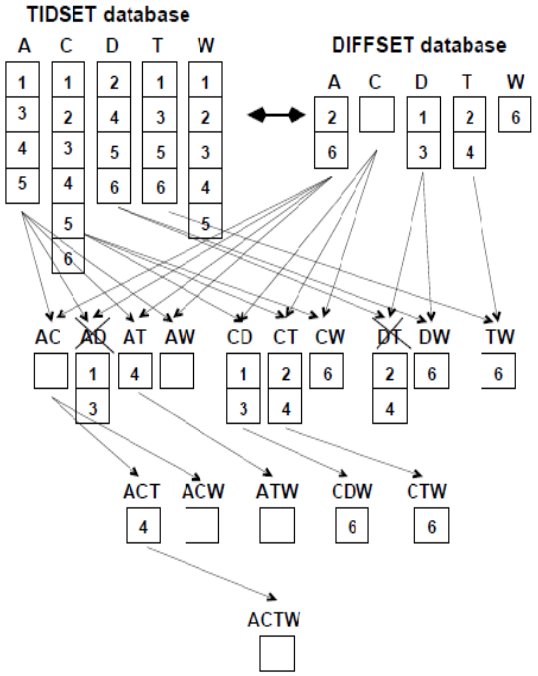

Figure 6. Example of diffsets created in dEclat Algorithm

III. EVALUATION

Association rule mining is very important technique of data mining. Finding frequent item sets is the most important task of association rule mining. There are numerous algorithms in frequent pattern mining – like Apriori, FP-Growth, Eclat and dEclat. While Apriori seems to be simple approach using a breadth-first search, it has the poor performance when compared with the other Algorithms. FP-Growth on the other hand, creates a FP- tree and is fairly good [24]. However, when it comes to Eclat and dEclat, both of them provide amazingly faster results as they employ a depth-first search and with the introduction of diffsets, dEclat gives us unparalleled faster results. Following graphs show a comparative analyses [22][23] of all the four algorithms on various factors that have been explained above:

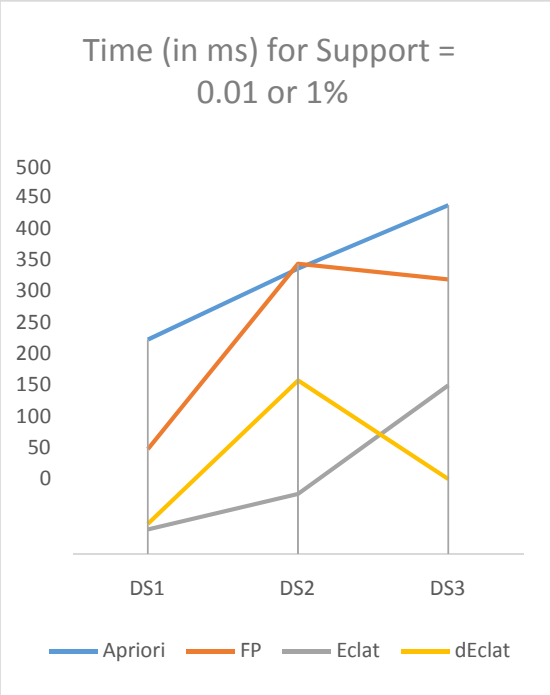

Figure 7. Time taken by Algorithms for various Datasets

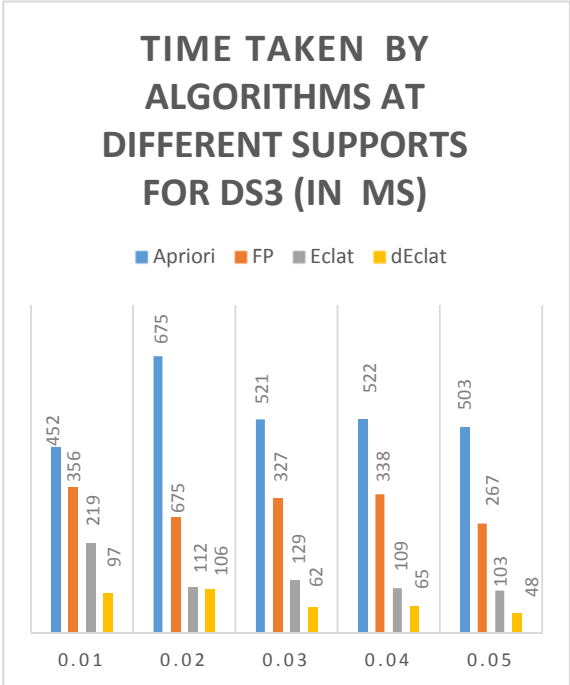

Figure 9. Time taken by Algorithms for Different Support Counts

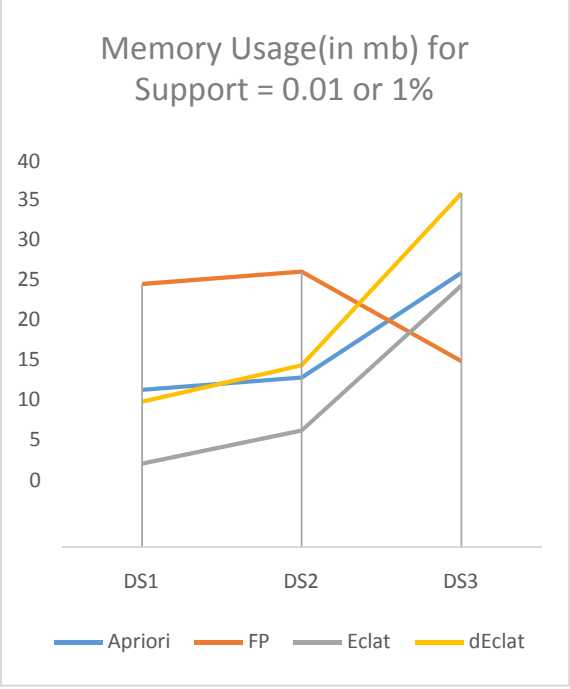

Figure 8. Memory Usage by Algorithms for various data sets

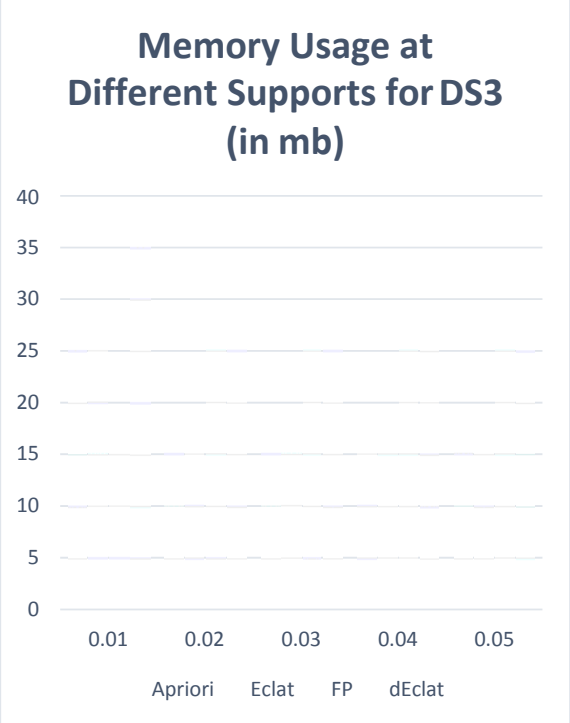

Figure 10. Memory Usage of Algorithms for Different Support Counts

Although we have presented a comparative study of the working of four novel algorithms, association rule mining is still in a stage of exploration and development

It seems to hold a good future scope as we can extend the employability of these algorithms to a larger dataset – Big Data and can apply it on Hadoop [25][26]. Also, the project gives us the scalability to add more algorithms that we can use for mining.

Following are the key points that can help in extending the project scope:

- Employing these algorithms to big data will require careful investigation of the format that is required by the project – i.e. the transaction database format.
- Since, the size of big data is too large, implementing dEclat Algorithm will be the best choice in order to achieve the faster results.
- There are possibilities of large memory storage requirement, as diffsets consume good memory.
- Further, it is observed that after the first run, there is a scope of caching. This will help us save more time and memory in the next future runs of the project.

After collecting the results from these algorithms, these are later used for second part of association rule mining i.e. for calculating confidence and generate the associate rules. These rules when generated are of significant value for major organizations and projects – especially market- basket analysis.

REFERENCES

[1] Qiankun Zhao, Sourav S. Bhowmick, Association Rule Mining: A Survey, Technical Report, CAIS, Nanyang Technological University, Singapore, 2003

[2] KomalKhurana, Mrs. Simple Sharma, A Comparative Analysis of Association Rules Mining Algorithms, International Journal of Scientific and Research Publications, Volume 3, Issue 5 , May 2013 ISSN 2250-3153

[3] IshNathJhaSamarjeet Borah, An Analysis on Association Rule Mining Techniques, International Conference on Computing, Communication and Sensor Network (CCSN) 2012

[4] ManishaGirotra, KanikaNagpal Saloniinocha Neha Sharma Comparative Survey on Association Rule Mining Algorithms, International Journal of Computer Applications (0975 – 8887) Volume 84 – No 10, December 2013

[5] Sotiris Kotsiantis, DimitrisKanellopoulos, AssociationRules Mining: A Recent Overview, GESTS International Transactions on Computer Science and Engineering, Vol.32 (1), 2006, pp. 71-82

[6] Gagandeep Kaur, Shruti Aggarwal , Performance Analysis of Association Rule Mining Algorithms, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 8, August 2013, ISSN: 2277 128X

[7] PratikshaShendge, Tina Gupta, Comparitive Study of Apriori& FP Growth Algorithms, Indian journal of research, Volume 2, Issue 3, March 2013.

[8] Ming-Syan Chen, Jiawei Han, P.S.Yu, Data mining: an overview from a database perspective, IEEE Transactions on Knowledge and Data Engineering, Volume:8, Issue: 6 ISSN: 1041-4347, 866 - 883

[9] Parita Parikh, Dinesh Waghela, Comparative Study of Association Rule Mining Algorithms, Parita Parikh et al, UNIASCIT, Vol 2 (1), 2012, 170-172, ISSN 2250-0987.

[10] R. Agrawal, T. Imielinski, and A. Swami, ªMining Association Rules Between Sets of Items in Large Databases, ACM SIGMOD Conf. Management of Data, May 1993.

[11] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. InkeriVerkamo, ªFast Discovery of Association Rules,º Advances in Knowledge Discovery and Data Mining, U. Fayyad and et al., eds., pp. 307±328, Menlo Park, Calif.: AAAI Press, 1996.

[12] R. Agrawal and J. Shafer, ªParallel Mining of Association Rules, IEEE Trans. Knowledge and Data Eng., vol. 8, no. 6, pp. 962±969, Dec. 1996.

[13] R. Agrawal and R. Srikant, ªFast Algorithms for Mining Association Rules,º Proc. 20th Very Large Data Base Conf., Sept. 1994.

[14] R.J. Bayardo, ªEfficiently Mining Long

Patterns from Databases, ACM SIGMOD Conf. Management of Data, June 1998.
[15] S. Brin, R. Motwani, J. Ullman, and S. Tsur, ªDynamic Itemset Counting and Implication Rules for Market Basket Data, ACM SIGMOD Conf. Management of Data, May 1997.
[16] M. J. Zaki and C.-J. Hsiao. CHARM: An efficient algorithm for closed association rule mining. Technical Report 99-10, Computer Science Dept., Rensselaer Polytechnic Institute, October 1999.
[17] D. Cheung, J. Han, V. Ng, A. Fu, and Y. Fu, ªA Fast Distributed Algorithm for Mining Association Rules,º Fourth Int'l Conf. Parallel and Distributed Information Systems, Dec. 1996
[18] B.A. Davey and H.A. Priestley, Introduction to Lattices and Order. Cambridge Univ. Press, 1990.
[19] D. Eppstein, ªArboricity and Bipartite Subgraph Listing Algorithms, Information Processing Letters, vol. 51, pp. 207±211, 1994.
[20] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness,
W.H. Freeman and Company, 1979.
[21] D. Gunopulos, R. Khardon, H. Mannila, and H. Toivonen, ªData Mining, Hypergraph Transversals, and Machine Learning,º Proc. 16th ACM Symp. Principles of Database Systems, May 1997.
[22] D. Gunopulos, H. Mannila, and S. Saluja, ªDiscovering All the Most Specific Sentences by Randomized Algorithms,º Int'l Conf. Database Theory, Jan. 1997.
[23] E.-H. Han, G. Karypis, and V. Kumar, ªScalable Parallel Data Mining for Association Rules, ACM SIGMOD Conf. Management of Data, May 1997.
[24] M. Holsheimer, M. Kersten, H. Mannila, and H. Toivonen, ªA Perspective on Databases and Data Mining, First Int'l Conf. Knowledge Discovery and Data Mining, Aug. 1995.
[25] M. Houtsma and A. Swami, ªSet-Oriented Mining of Association Rules in Relational Databases,º 11th Int'l Conf. Data Eng., 1995.
[26] http://en.wikipedia.org