International Journal of Advanced Research in IT and Engineering

ISSN: 2278-6244

A BRIEF DISCUSSION ON ALGORITHMS FOR ASSOCIATION RULE MINING

Rashmi Shikhariya*

Abstract: *Data mining is often defined as finding hidden information in a database. Alternatively, it has been called exploratory data analysis, data driven discovery and deductive learning. Among the area of data mining, the problem of deriving associations from data has received a great deal of attention. Association rules are used to identify relationships among a set of items in database. Frequent pattern mining plays an essential role in association rule mining, which has been a focused theme in data mining research for over the past 15 years. Starting from the pioneering work of Agrawal [1,2], abundant literature has been dedicated to this research. In this paper we are giving the brief discussion on algorithms for association rule mining techniques, the current status of frequent mining and discuss a few promising research directions.*

Keywords: *Data Mining, Frequent Patterns, Association Rules, Data Mining Research.*

*Computer Science & Engineering, Shri Ram Institute of Technology, Jabalpur, India

I. INTRODUCTION

Finding frequent itemsets in databases is crucial in data mining for purpose of extracting association rules. Frequent patterns are itemsets that appear in a data set with frequency no less than a user-specified threshold. Finding frequent patterns plays an essential role in mining associations, correlations, and many other interesting relationships among data. Moreover, it helps in data indexing, classification, clustering, and other data mining tasks as well. Thus, frequent pattern mining has become an important data mining task and a focused theme in data mining research.

Frequent pattern mining was first proposed by Agrawal et al. [1,2] for market basket analysis in the form of association rule mining, which is useful for discovering interesting relationships hidden in large data sets. For example, we may find a strong relationship, which can be represented in the form of association rules or sets of frequent items, exists between the sale of diapers and beer because many customers who buy diapers also buy beer. Retailers can use this relationship to help them identify new opportunities for cross-selling their products to the customers. Besides market basket data, association rule mining is also applicable to other application domains such as bioinformatics, multimedia data mining, Web mining and scientific data analysis.

Since the first proposal of this new data mining task and its associated efficient mining algorithms, there have been hundreds of follow-up research publications. In this article, we perform a brief overview of frequent pattern mining methods, representation and application.

II. BASIC CONCEPTS

This section reviews the basic terminology used in association analysis and presents a formal description of the task.

Itermset and Support Count: Let $I = \{i_1, i_2, ..., i_d\}$ are the set of all items in a market basket data and $T = \{t_1, t_2, ... t_N\}$ are the transactions. Each transaction t_i contains a subset of items chosen from I. In association analysis, a collection of zero or more items is termed an itemset. If an itemset contains k items, it is called a k-itemset.

A transaction t_i is said to contain an itemset X if X is a subset of t_i . Mathematically, the support count $\sigma(X)$, for an itemset X can be stated as:

$$\sigma(X)=|\{t_i|X\subseteq t_i, t_i\in T\}|$$

The symbol $|\bullet|$ denotes the number of elements in a set.

Association Rule: An association rule is an implication expression of the form $X\rightarrow Y$, where X and Y are disjoint itemsets, i.e. $X\cap Y = \phi$. The strength of an association rule can be measured in terms of its support and confidence. Support determines how often a rule is applicable to a given data set, while confidence determines how frequently items in Y appear in transactions that contain X . The formal definitions of these metrics are

$$Support, s(X\rightarrow Y)=\sigma(X\cup Y)/N$$

$$Confidence, c(X\rightarrow Y)=\sigma(X\cup Y)/\sigma(X)$$

Frequent Itemset: A k-itemset *X* is frequent if *X* in a transaction database *T* no lower than $\theta|T|$ times, where θ is a user-specified minimum support threshold (called *min_sup*), and *T* is the total number of transactions in *T.*

Association Rule Mining: The association rule mining problem can be formally stated as follows: Given a set of transactions *T,* find all the rules having support ≥ min_sup and confidence ≥ min_conf, where min_sup and min_conf are the user-specified support and confidence thresholds.

A common strategy adopted by many association rule mining algorithms is to decompose into two major subtasks: **Frequent Itemset Generation**, whose objective is to find all the itemsets that satisfy the min_sup threshold; and **Rule Generation,** whose objective is to extract all the high confidence rules from the frequent itemsets found in the frequent itemset generation step.

The computational requirements for frequent itemset generation are generally more expensive than those of rule generation. Efficient techniques for generating frequent itemsets are reviewed in Section III.

III. EFFICIENT ALGORITHMS FOR MINING FREQUENT PATTERS

(A) *Apriori Algorithm*

Since there are usually a large number of distinct single items in a typical transaction database, and their combinations may form a very huge number of itemsets, it is challenging to develop callable methods for mining frequent itemsets in alarge transaction

database. Agrawal and Srikant [2] observed an interesting downward closure property, called *Apriori*, among frequent k-itemsets: A k-itemset is frequently if all of its sub-itemsets are frequent. This implies that frequent itemsets can be mined by first scanning the database to find the

frequent 1-itemsets, then using the frequent 1-itemsets to generate candidate frequent 2-itemsets, and check against the database to obtain the frequent 2-itemsets. This process iterates until no more frequent k-itemsets can be generated for some k items.

(B) *Direct Hashing and Pruning (DHP)*

It is absorbed that reducing the candidate items from the database is one of the important task for increasing the efficiency. Thus a DHP technique was proposed [7] to reduce the number of candidates in the earlier passes Ck for k>1. In this method, support is counted by mapping the items from the candidate list into the buckets which is divided according to support known as Hash table structure. As the new itemset is encountered, if item exist earlier then increase the bucket count else insert into new bucket. Thus in the end the bucket whose support count is less the minimum support is removed from the candidate set.

(C) *Partitioning Algorithm*

Partitioning algorithm [1] is based to find the frequent elements on the basis partitioning of database in *n* parts. It overcomes the memory problem for large database which do not fit into main memory because small parts of database easily fit into main memory. This algorithm is divided into following steps:

1. In the first, whole database is divided into n number of parts.
2. Each partitioned database is loaded into main memory one by one and local frequent elements are found.
3. All locally frequent elements are combined and global candidate sets are made.
4. Globally frequent elements are built from this candidate set.

It should be noted that if the minimum support for transactions in whole database is min_sup, then the minimum support for partitioned transactions is min-sup number of transaction in that partition.

(D) *Sampling Algorithm*

This algorithm [10] is used to overcome the limitation of I/O overhead by not considering the whole database for checking the frequency. It is just based in the idea to pick a random sample of itemset *R* from the database instead of whole database *D*. The sample is picked in such a way that whole sample is accommodated in the main memory. In this way we try to find the frequent elements for the sample only and there is chance to miss the global frequent elements in that sample. Therefore lower threshold support is used instead of actual minimum support to find the frequent elements local to sample. In the best case only one pass is needed to find all frequent elements if all the elements included in sample and if elements missed in sample then second pass are needed to find the itemsets missed in first pass or in sample [13].

(E) *Dynamic Itemset Counting (DIC)*

This algorithm [4] is also used to reduce the number of database scan. It is based upon the downward disclosure property in which the candidate itemsets are added at different point of time during the scan. In this, dynamic blocks are formed from the database marked by start points and unlike the previous techniques of *Apriori*, it dynamically changes the sets of candidates during the database scan. Unlike the *Apriori*, it cannot start the next level scan at the end of first level scan, it start the scan by starting label attached to each dynamic partition of candidate sets.

(F) *FP-Growth Algorithm*

FP-Growth algorithm [7] is an efficient method of mining all frequent itemsets without candidate's generation. The algorithm mines the frequent itemsets by using a divide-and-conquer strategy as follows:

FP-Growth first compresses the database representing frequent itemset into a frequent-pattern tree, or *FP-tree*, which retains the itemset association information as well. The next step is to divide a compressed database into set of conditional databases (a special kind of projected database), each associated with one frequent item. Finally, mine each such database separately. Particularly, the construction of *FP-Tree* and the mining of *FP-Tree* are the main steps in *FP-Growth* algorithm. The detail steps are follows [6]:

FP-Growth Method: Construction of FP-Tree:

1. Create root of the tree as a 'null'.

2. After scanning the database D for finding the 1-itemset then process the each transaction in decreasing order of their frequency.
3. A new branch is created for each transaction with the corresponding support.
4. If same node is encountered in another transaction, just increment the support count by 1 of the common node.
5. Each item points to the occurrence in the tree using the chain of node-link by maintaining the header table.

After above process, mining of the *FP-Tree* will be done by Creating Conditional (sub) pattern bases as follows:

1. Start from node constructs its conditional pattern base.
2. Then, Construct its conditional *FP-Tree* & perform mining on such a tree.
3. Join the suffix patterns with a frequent pattern generated from a conditional *FP-Tree* for achieving *FP-Growth*.
4. The union of all frequent patterns found by above step gives the required frequent itemset.

(G) *H-mine Algorithm*

H-mine [8] algorithm is the improvement over *FP-Tree* algorithm as in *H-mine* projected database is created using in-memory pointers. *H-mine* uses an *H-struct* new data structure for mining purpose, known as hyperlinked structure. It is used upon the dynamic adjustment of pointers which helps to maintain the processed projected tree in main memory. Therefore *H-mine* is proposed for frequent pattern data mining for data sets that can fit into main memory. It has polynomial space complexity therefore more space efficient than *FP-Growth* and also designed for fast mining purpose. For the large databases, first partition the database, then mine each partition in main memory using *H-struct* and then consolide global frequent pattern [8]. If the database is dense then it integrates with *FP-Growth* dynamically by detecting the swapping condition and constructing the *FP-Tree*. This working ensures that it is scalable for both large and medium size databases and for both sparse and dense datasets [14]. The advantage of using in-memory pointers is that their projected database does not need any memory. The memory is required is only for the set of in-memory pointers.

(H) *Eclat Algorithm*

Both the *Apriori* and *FP-Growth* methods mine frequent patterns from a set of transaction horizontal data format (i.e. {T_{ID}: itemset}), where T_{ID} is a transaction-id and itemset is the set of items bought in transaction T_{ID}. Alternatively, mining can also be performed with data presented in vertical data format (i.e. {item: T_{ID}_set}).

Zaki [12] proposed Equivalence CLASS Transformation (*Eclat*) algorithm by exploring the vertical data format. The first scan of the database builds the T_{ID}_set of each single item. Starting with a single item (k = 1), the frequent (k+1)-itemsets grown from a previous k-itemset can be generated according to the *Apriori* property, with a depth-first computation order similar to *FP-Growth* [7]. The computation is done by inter section of the T_{ID}_sets of the frequent k itemsets to compute the T_{ID}_sets of the corresponding (k+1)-itemsets. This process repeats, until no frequent itemsets or no candidate itemsets can be found.

Besides taking advantage of the *Apriori* property in the generation of candidate (k+1)-itemset from frequent k-itemsets, another merit of this method is that there is no need to scan the database to find the support of (k+1)-itemsets (for $k \geq 1$). This is because the T_{ID}_set of each k-itemset carries the complete information required for counting such support.

Another related work which mines the frequent itemsets with the vertical data format is [13]. This work demonstrated that, though impressive results have been achieved for some data mining problems using highly specialized and clever data structures, one could also explore the potential of solving data mining problems using the general purpose database management systems.

IV. COMPACT REPRESENTATION OF FREQUENT ITEMSETS

A major challenge in mining frequent patterns from a large dataset is the fact that such mining often generates a huge number of patterns satisfying the min_sup threshold, especially when min_sup is set low. This is because if a pattern is frequent, each of its sub-patterns is frequent as well. A large pattern will contain an exponential number of smaller, frequent sub-patterns. To overcome this problem, closed frequent pattern mining and maximal frequent pattern mining were proposed [14, 17].

A pattern α is a closed frequent pattern in a dataset D if α is frequent in D and there exists no proper super-pattern β such that β has the same support as α in D. A pattern α is a

maximal frequent pattern (or max-pattern) in set D if α is frequent, and there exists no super-pattern β such that α ⊂ β and β is frequent in D. For the same min_sup threshold, the set of closed frequent patterns contains the complete information regarding to its corresponding frequent patterns; whereas the set of max-patterns, though more compact, usually does not contain the complete support information regarding to its corresponding frequent patterns.

The mining of frequent closed itemsets was proposed by Pasquier et al. [14], where an *Apriori*-based algorithm called *A-Close* for such mining was presented. Other closed pattern mining algorithms include *CHARM* [15], and *FPClose* [16]. The main challenge in closed frequent pattern mining is to check whether a pattern is closed. There are two strategies to approach this issue:

(1). To keep track of the T_{ID} list of a pattern and index the pattern by hashing its T_{ID} values. This method is used by CHARM which maintains a compact T_{ID} list; and

(2). To maintain the discovered patterns in a pattern-tree similar to *FP-Tree*.

This method is exploited by *FPClose*. Mining closed itemsets provides an interesting and important alternative to mine frequent itemsets since it inherits the same analytical power but generates a much smaller set of results. Better scalability and interpretability is achieved with closed itemset mining.

Mining max-patterns was first studied by Bayardo [17], where MaxMiner, an *Apriori*-based, level-wise, breadth-first search method was proposed to find max-itemset by performing superset frequency pruning and subset in frequency pruning for search space reduction. Another efficient method *MAFIA*, proposed by Burdick et al. [18], uses vertical bitmaps to compress the transaction id list, thus improving the counting efficiency. Yang [19] provided theoretical analysis of the complexity of mining max-patterns. The complexity of enumerating maximal itemsets is shown to be NP-hard.

V. APPLICATIONS

Frequent patterns mining has been applied to a variety of application domains, such as indexing and similarity search of complex structured data, multimedia mining, and web mining.

A. *Indexing and similarity search of complex data*

Complex objects such as transaction sequence, event logs, proteins and images are widely used in many fields. Efficient search of these objects becomes a critical problem for many applications. Due to the large volume of data, it is inefficient to perform a sequential scan on the whole database and examine objects one by one. High performance indexing mechanisms thus are in heavy demand in filtering objects that obviously violate the query requirement.

gIndex [20] proposes a discriminative frequent pattern-based approach to index structures and graphs. A fundamental problem arises: if only frequent patterns are indexed, how to find those queries which only have infrequent patterns? *gIndex* solved this problem by replacing the uniform support constraint with a size-increasing support function, which has very low support for small patterns but high support for large patterns. *SeqIndex* is one example using frequent pattern-based approach to index sequences. Taking frequent patterns as features, new strategies to perform structural similarity search were developed such as Grafil [21] and PIS [22].

B. *Multimedia Data mining*

A multimedia database system stores and manages a large collection of multimedia data, such as audio, video, image, graphics, speech, text, document, and hyper text data. Multimedia data mining is finding patterns and knowledge from multimedia data. Frequent pattern analysis in multimedia data plays a similar important role in multimedia data mining. To mine frequent patterns in multimedia data, each image object can be treated as a transaction and frequently occurring patterns among different images can be discovered. Zaïane et al. [23] developed a progressive algorithm for mining multimedia associations. Chengcui Zhang et al. [24] proposed an automatic spatiotemporal mining system of rolling and adherent leukocytes for intravital videos.

C. *Web mining*

Web mining is the application of data mining techniques to discover patterns and knowledge from the Web [25]. There are three different types of web mining: web content mining, web structure mining, and web usage mining. Web content mining is a knowledge discovery task of finding information within web pages, while web structure mining aims to discover knowledge hidden in the structures linking web pages. Web usage mining is

focused on the analysis of users' activities when they browse and navigate through the Web.

Association rules discovered for pages that are often visited together can reveal user groups [26] and cluster web pages. Web access patterns via association rule mining in weblogs were proposed by [27]. Sequential pattern mining in weblogs could find browse and navigation orders (i.e. pages that are accessed immediately after another), which might be used to refine cache design and web site design.

VI. RESEARCH DIRECTIONS

Although abundant literature has been published in research into frequent pattern mining, however, there are still several critical research problems that need to be solved before frequent pattern mining can become a cornerstone approach in data mining applications.

First, the set of frequent patterns derived by most of the current pattern mining methods is too huge for effective usage. There are proposals on reduction of such a huge set, including closed patterns, maximal patterns, approximate patterns, representative patterns, clustered patterns, and discriminative frequent patterns. However, it is still not clear what kind of patterns will give us satisfactory pattern sets in both compactness and representative quality for a particular application, and whether we can mine such patterns directly and efficiently. Much research is still needed to substantially reduce the size of derived pattern sets and enhance the quality of retained patterns.

Second, we need mechanisms for deep understanding and interpretation of patterns, and contextual analysis of frequent patterns. The main research work on pattern analysis has been focused on pattern composition and frequency. The semantic of a frequent pattern includes deeper information: what is the meaning of the pattern; what are the synonym patterns; and what are the typical transactions that this pattern resides? In many cases, frequent patterns are mined from certain datasets which also contain structural information. We believe the deep understanding of frequent patterns is essential to improve the interpretability and the usability of frequent patterns. One initial study in this direction is done by Mei et al. [28].

Finally, applications often raise new research issues and bring deep in sight on the strength and weakness of an existing solution. This is also true for frequent pattern mining. Much work is needed to explore new applications of frequent pattern mining. For example,

bioinformatics has raised a lot of challenging problems, and we believe frequent pattern mining may contribute a good deal to it with further research.

VII. CONCLUSION

In this article, we present a brief overview of the current status and future directions of frequent pattern mining. With over the past 15 years of extensive research, there have been hundreds of research publications and tremendous research, development and application activities in this domain. It is impossible for us to give a complete coverage on this topic with limited space and our limited knowledge. Hopefully, this short overview may provide a rough outline of the recent work and give people a general view of the field. In general, we feel that as a young research field in data mining, frequent pattern mining has an achieved tremendous progress and claimed a good set of applications.

REFERENCES

[1] Agrawal R, Imielinsi T, Swami A. Mining association rules between sets of items in large databases. In Proceedings of the 1993 ACMSIGMOD international conference on management of data (SIGMOD'93), pages207–216.1993

[2] Agrawal R, Srikant R. Fast algorithms for mining association rules. In Proceedings of the 1994 international conference on very large databases (VLDB'94), pages487–499.1994

[3] Mannila H, Toivonen H, Verkamo AI(1994) Efficient algorithms for discovering association rules. In Proceeding of the AAAI'94 workshop knowledge discovery in databases (KDD'94), pages181–192 .1994

[4] Park JS, Chen MS, Yu PS. An effective hash-based algorithm for mining association rules, In Proceeding of the 1995 ACM SIGMOD international conference on management of data(SIGMOD'95), pages175–186. 1995

[5] Toivonen H. Sampling large databases for association rules. In Proceeding of the1996 international conference on very large databases (VLDB'96), pages134–145, 1996

[6] Geerts F, Goethals B, Bussche J. A tight upper bound on the number of candidate patterns, In Proceeding of the 2001 international conference on data mining (ICDM'01),pages155–162. 2001

[7] Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. In Proceeding of the 2000 ACM-SIGMOD international conference on management of data(SIGMOD'00),pages1–12.2000

[8] Agarwal R, Aggarwal CC, Prasad VVV. A tree projection algorithm for generation of frequent itemsets. J Parallel Distribut Comput 61. pages350–371.2001

[9] Pei J, Han J, Lakshmanan LVS. Mining frequent itemsets with convertible constraints. In Proceeding of the 2001 international conference on data engineering (ICDE'01),pages433–332. 2001

[10] Liu J, Pan Y, Wang K, Han J. Mining frequent itemsets by opportunistic projection. In Proceeding of the 2002 ACM SIGKDD international conference on knowledge discovery in databases (KDD'02), pages239–248. 2002

[11] Grahne G, Zhu J. Efficiently using prefix-trees in mining frequent itemsets .In Proceeding of the ICDM'03 international workshop on frequent itemset mining implementations(FIMI'03), pages123– 132.2003

[12] Zaki MJ. Scalable algorithms for association mining. IEEE Trans Knowl Data Eng 12. Pages 372–390. 2000

[13] Holsheimer M, Kersten M, Mannila H, Toivonen H. Aperspective on databases and datamining. In Proceeding of the 1995 international conference on knowledge discovery and data mining(KDD'95), pages150–155. 1995

[14] Pasquier N, Bastide Y, Taouil R, Lakhal L. Discovering frequent closed itemsets for association rules. In Proceeding of the 7th international conference on database theory(ICDT'99). pages398–416. 1999

[15] Zaki M J,Hsiao C J. CHARM: an efficient algorithm for closed itemset mining. In Proceeding of the 2002 SIAM international conference on data mining(SDM'02), pages457–473. 2002

[16] Grahne G, Zhu J. Efficiently using prefix-trees in mining frequent itemsets .In Proceeding of the ICDM'03 international workshop on frequent itemset mining implementations(FIMI'03), pages123– 132.2003

[17] Bayardo RJ. Efficiently mining long patterns from databases. In Proceeding of the 1998 ACM SIGMOD international conference on management of data(SIGMOD'98), pages85–93. 1998

[18] Burdick D, Calimlim M, GehrkeJ. MAFIA: a maximal frequent itemset algorithm for transactional databases. In Proceeding of the 2001 international conference on data engineering (ICDE'01), pages443–452. 2001

[19] Yang G. The complexity of mining maximal frequent itemsets and maximal frequent patterns. In Proceeding of the 2004 ACM SIGKDD international conference on kowledge discovery in databases(KDD'04), pages344–353. 2004

[20] Yan X, Yu PS, Han J. Graph indexing: a frequent structure-based approach. In Proceeding of the 2004 ACM SIGMOD international conference on management of data(SIGMOD'04), pages335–346. 2004

[21] Yan X, Yu PS, Han J. Substructure similarity search in graph databases. In Proceeding of the 2005 ACM SIGMOD international conference on management of data(SIGMOD'05), pages766–777. 2005

[22] Yan X, Zhu F, Han J, Yu PS. Searching substructures with superimposed distance. In Proceeding of the 2006 international conference on data engineering(ICDE'06),pages88. 2006

[23] Zaïane OR, Han J, Zhu H. Mining recurrent items in multimedia with progressive resolution refinement. In Proceeding of the 2000 international conference on data engineering(ICDE'00), pages461–470. 2000

[24] Chengcui Zhang, Wei-Bang Chen, Lin Yang, Xin Chen, and John K. Johnstone. Automatic in vivo Microscopy Video Mining for Leukocytes. .SIGKDDExplor 9:30–37 2007

[25] Srivastava J, Cooley R, Deshpande M, Tan PN. Web usage mining: discovery and applications of usage patterns from web data.SIGKDDExplor 1:12–23 2000

[26] Eirinaki M, Vazirgiannis M. Web mining for web personalization .ACM Trans Inter Tech 3:1–27 2003

[27] Kuramochi M, Karypis G. Frequent subgraph discovery. In Proceeding of the 2001 international conference on data mining (ICDM'01),pages313–320. 2001

[28] Mei Q, Xin D, Cheng H, Han J, Zhai C. Generating semantic annotations for frequent patterns with context analysis. In Proceeding of the 2006 ACM SIGKDD international conference on knowledge discovery in databases(KDD'06), pages337– 346.2006.

[29] Xun Zhu, Hongtao Deng, Zheng Chen, A Brief Review on Frequent Pattern Mining