International Journal of Latest Trends in Engineering and Technology
Vol.(7)Issue(3), pp. 011-021
DOI: http://dx.doi.org/10.21172/1.73.502
e-ISSN:2278-621X

Frame work for association rule mining with updated FP-growth and modified COFI approaches

PDVN KUMAR[1]

***Abstract*: Association rules are interesting correlations among attributes in a database. The discovery of association rules is an extremely computationally expensive task and it is therefore imperative to have fast scalable algorithms for mining these rules. In this paper, we present efficient techniques for discovering association rules from large databases and for removing redundancy from these rules so as to improve the quality of output. Some of the important algorithms that can be considered for Association Mining are FP-Growth and COFI algorithm. FP-Growth algorithm generates frequent item sets from an FP-tree by exploring the tree in a bottom-up fashion. The COFI algorithm is not only one order of magnitude faster and requires significantly less memory than the FP-Growth; it is also very effective with extremely large datasets**

Index Terms - *FP-tree, COFI-tree, Ordered-Partitioning Bases, Leap Traversal approach*

1. INTRODUCTION

In recent years the size of databases has increased rapidly. This has led to a growing interest in the development of tools capable in the automatic extraction of knowledge from data .The term Data Mining, or Knowledge Discovery in Databases, has been adopted for a field of research dealing with the automatic discovery of implicit information or knowledge within databases. The implicit information within databases, and mainly the interesting association relationships among sets of objects, that lead to association rules, may disclose useful patterns for decision support, financial forecast, marketing policies, even medical diagnosis and many other applications. These facts attracted a lot of attention in recent data mining research. Mining association rules may require iterative scanning of large databases, which is costly in processing. Many researchers have focused their work on efficient mining of association rules in databases[1,2,3].

Among the areas of data mining, the problem of deriving associations from data has received a great deal of attention. Association rule mining searches for interesting relationships among items of a given dataset. Association rule mining is often referred as "Market-Basket" problem. The input is both a relational or transaction database consists of large collection of transactions, which are subsets of these item sets, and the task is to find relationships between the presences of various items with in these baskets. There are numerous applications of data mining, which fit into this framework [2,3,4,5].

A typical example of association rule mining is market basket analysis. This process analyzes customers buying habits by finding associations between the different item sets that customers place in their "shopping baskets". The sdiscovery of such associations can help retailers to develop marketing strategies by gaining insight into which items has frequently purchased together by customers. An item

[1] Dr. C S Rao P G Center, Sri Y N College, Narasapur, AP, INDIA

set is a collection of items. The items that are frequently purchased together are called frequent item sets. They often used to gain insight into the drivers of behavior, such as which products, when purchased, drive the purchase of which other products. After finding list of frequent item sets, which are purchased together by the customer, also outputs rules among frequent item sets specifying the probability of purchasing frequent item sets together[6,7,8].

This paper deals with extracting frequent patterns in the given dataset. Implemented algorithms, FP-Growth, COFI*, are tested on benchmark datasets like Retail, Chess and Mushroom. This project takes database file, support and confidence values as input and generate list of frequent patterns and also association rules among them based on particular algorithm chosen.

A. Characterization and Discrimination

Data can be associated with classes or concepts. It can be useful to describe individual classes and concepts in summarized, concise and yet precise terms. Such descriptions of a class or a concept are called class/concept description. These descriptions can be derived via (1) data characterization, by summarizing the data or the class under study in general terms (often called the target class) or (2) data discrimination, by comparison of the target class with one or a set of comparative classes (often called contrasting classes), or (3) both data characterization or discrimination. Association analysis is the discovery of association rules showing attribute-value conditions that occur frequently in a given set of data. It is widely used for market basket or transaction data analysis[9,10,11].

Classification and Prediction: Classification is the process of finding a set of models (or functions) that describe and distinguish data classes or concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown. The derived model is based on the analysis of a set of training data (i.e. data objects whose class label is known). Classification can be used for predicting the class label of data objects. However, in many applications, users may wish to predict some missing or unavailable data values rather than class labels. This is usually the case when predicting values are numerical data and often specifically referred to as prediction. Although, prediction may refer to as both value data prediction and thus is distinct from classification[12,13,14].

I. BASIC CONCEPTS OF ASSOCIATION RULES

Let I= $\{i_1, i_2, i_3, i_n\}$ be a set of items. Let D, the task relevant data, be a set of database transaction T is a set of items such that $T \subseteq I$. Each transaction is associated with an identifier called TID. Let A be a set of items. A transaction T is said to contain A if and only if $A \subseteq T$. An association rule is an implication of the form A=>B, where $A \subset I$, $B \subset I$ and $A \cap B=\Phi$.

An objective measure for association rule of the form A=>B is **Support**, which indicates the percentage of transactions from the given database that the given rule satisfies. Another objective measure is **Confidence**, which assesses the degree of certainty of the detected association [15,16,17,18]. Support and Confidence are defined as follows.

Support (A=>B) = P (A∪B)

Confidence (A=>B) = P (B/A)

Rules that satisfy both minimum support and minimum confidence threshold are called Strong rules. The objective of association rule mining is to generate strong rules. By convention, we write support and confidence values so as to occur between 0% and 100% rather than 0 and 1.0.

A set of items is referred to as an itemset. An itemset that contains k items is k-itemset. The occurrence frequency of the itemset is the number of transactions that contain the itemset. An itemset satisfies minimum support if the occurrence frequency of the itemset is greater than or equal to the product of minimum support threshold and the total number of transactions in D. Such an itemset is

called frequent itemset/pattern. The number of transactions required to satisfy the minimum support is called as minimum support count.

Association analysis discovers interesting relationships hidden in large datasets. The uncovered relationships can be represented in the form of association rules or set of frequent items. For example, the following rule can be extracted from the data set shown in the table.

{Diapers}→ {Beer}

The rule suggests that a strong relationship exists between the sale of diapers and beer because many customers who buy diapers also buy beer. Retailers can use this type of rules to help them identify new opportunities for cross selling their products to the customers.

TABLE I. **MARKET BASKET TRANSACTIONS**

TID	Items
1.	{Bread, Milk}
2.	{Bread, Diapers, Beer, Eggs}
3.	{Milk, Diapers, Beer, Cola}
4.	{Bread, Milk, Diapers, Beer}
5.	{Bread, Milk, Diapers, Cola}

Consider the rule {Milk, Diapers}→ {Beer}. Since the support count for {Milk, Diapers, Beer} is 2 and the total number of transactions is 5, the rule's support is 2/5=0.4. The rule's confidence is obtained by dividing the support count for {Milk, Diapers, Beer} by the support count {Milk, Diapers}. Since there are 3 transactions that contain milk and diapers, the confidence for this rule is 2/3=0.67.

A. Use of Support and Confidence

Support is an important measure because a rule that has the very low support may occur simply by chance. A low support rule is also likely to be uninteresting from a business perspective because it may not be profitable to promote items that customers seldom buy together. For these reasons, support is used to eliminate uninteresting rules.

Confidence, on the other hand, measures the reliability of the inference made by a rule. For a given rule X->Y, the higher the confidence, the more likely it is for Y to be present in transactions that contain X. Confidence also provides an estimate of conditional probability of Y given X.

B. Steps in Association rule mining

Frequent Itemset Generation, whose objective is to find all the itemsets that satisfy the minsup threshold. These itemsets are called frequent itemsets.

Rule Generation, whose objective is to extract all the high-confidence rules from the frequent itemsets .These rules are called strong rules.

The computational requirements for frequent itemset generation are generally more expensive than those of the rule generation.

C. Frequent Itemset Generation

A lattice structure can be used to enumerate the list of all possible itemsets. In general, a data set that contains k items can potentially generate up to 2^k-1 frequent itemsets, excluding the null set. Because k can be very large in many practical applications, the search space of itemsets that need to be explored

exponentially large. A brute-force approach for finding frequent itemsets is to determine the support count for every **candidate itemset** in the lattice structure. To do this, we need to compare each candidate against every transaction. If the candidate is contained in a transaction, its support count will be incremented. There are several ways to reduce the computational complexity of frequent itemset generation[12-18]

- Reduce the number of candidate itemsets (M). The Apriori principle is an effective way to eliminate some of the candidate itemsets without counting their support values.
- Reduce the number of comparisons. Instead of matching each candidate itemset against every transaction, we can reduce the number of comparisons by using more advanced data structures, either to store the candidate itemsets or to compress the data set.

D. Generation of Association Rules from Frequent Itemsets

Once the frequent itemsets are generated from the given database, it is straightforward and simple to generate association rules from them. Each frequent k-itemset, Y, can produce up to 2^k-2 association rules, ignoring rules that have empty antecedents or consequents($\Phi \rightarrow$ Y or Y$\rightarrow \Phi$).An association rule can be extracted by partitioning the itemset Y into two non-empty subsets, X and Y-X, such that X$\rightarrow$Y-X satisfies the confidence threshold.BLet X= {1, 2, 3} be a frequent itemset. There are six data association rules that can be generated from X: {1,2}$\rightarrow${3},{1,3}$\rightarrow${2},{2,3}$\rightarrow${1},{1}$\rightarrow${2,3},{2}$\rightarrow${1,3}, and {3}$\rightarrow${1,2}. As each of their support is identical to the support for X, the rules must satisfy the support threshold.

E. Earlier Association Mining Algorithms

Generally, an association rules mining algorithm contain the following steps [5]:

- The set of candidate k-itemsets is generated by 1-extensions of the large (k-1) itemsets.
- Supports for the candidate k-itemsets are generated by a pass over the database.
- Itemsets that do not have the minimum support are discarded and the remaining itemsets are called large k-itemsets.

This process is repeated until no more large itemsets are found.

Finding frequent items is an inherent part of many data analysis processes. The most important, and at the basis of many other approaches, is Apriori [4]. According to the **Apriori property**, all the supersets of an itemset (infrequent) can be eliminated from the candidate itemsets as soon as the itemset is found to be infrequent. **Apriori algorithm** makes a level-wise exploration of candidate itemset lattice, starting from single itemsets. Before scanning the transaction database for counting the support of a candidate k-itemsets, it is made sure that all its (k-1)-item subsets are frequent. Apriori algorithm requires as many database scans, as there are elements in the largest frequent itemset. In case of very large databases because of limited main memory, each database scan in turn requires swap in/out of database partitions one after the other. Due to this, the basic apriori algorithm is not suitable for handling large transaction databases. The COFI approach for mining frequent itemsets is an efficient algorithm [3] is not only one order of magnitude faster and requires significantly less memory than the popular FP-Growth; it is also very effective with extremely large datasets, better than any reported algorithm. However, COFI has a significant drawback when mining dense transactional databases which is the case with some real datasets. The algorithm performs poorly in these cases because it ends up generating too many local candidates that are doomed to be infrequent. COFI in many cases is shown to be faster than FP-Growth and requires significantly less memory. The idea of COFI is to build projections from the FP-tree each corresponding to sub-transactions of items co-occurring with a given frequent item. These trees are built and efficiently mined one at a time making the footprint in memory significantly small. The COFI algorithm generates candidates using a top-down approach, where its performance shows to

be severely affected while mining databases that has potentially long candidate patterns that turns to be not frequent, as COFI needs to generate candidate sub-patterns for all its candidates patterns. We build upon the COFI approach to find the set of frequent patterns but after avoiding generating useless candidates.

II. SYSTEM STRUCTURE

The problem consists of finding associations between items or itemsets in transactional data. The data could be retail sales in the form of customer transactions, text documents, or images. Association rules have been shown to be useful for other applications such as recommender systems, diagnosis, decision support, telecommunication, and even supervised classification. The problem is formally stated as follows: Let I= $\{i_1, i2, i3\ldots\ldots i_m\}$ be the set of literals, called items and m is considered the dimensionality of the problem. Let D be the set of transactions where each transaction T is a set of items such that $T \subseteq I$. A unique identifier TID is given to each transaction. A transaction T is said to contain X, a set of items in I, if $X \subseteq T$. An association rule is an implication of the form "X =>Y" where $X \subseteq I$, $Y \subseteq I$, and $X \cap Y = \Phi$. An itemset X is said to be frequent if its support s is greater than or equal to a given minimum support threshold α. Discovering Association rules, however is nothing more than an application for frequent itemset mining, like inductive databases, query expansion, document clustering etc. This problem used to mine frequent patterns from the databases like Retail transaction database, Chess database and Mushroom database using association mining algorithms FP-Growth, COFI* and to generate association rules among frequent patterns . Association mining mines transaction database to extract the frequent patterns present in the database. By understanding these patterns, customer's behavior of purchasing items can be analyzed and that information can be used to improve sales by keeping the items, which are purchased together at side by side.

This system is developed for mining frequent itemset information basing minimum threshold values of support and confidence values from synthetic dataset like retail, chess, mushroom. After mining frequent itemsets from the given dataset, the association rules which satisfy the given threshold confidence value can be obtained

Presenting the efficient techniques for discovering association rules from large databases and removing redundancy from these rules so as to improve the quality of output for some of the important algorithms (1) FP-Growth algorithm, (2) COFI* algorithm. Then Performance comparison of the algorithms by using various transactional datasets is also the main objective of this work. Response time is taken into consideration for comparison of the algorithms.

In Current Systems, the process of mining of Interesting Association Rules and Frequent Itemsets from a given dataset was done by Apriori based algorithms. These algorithms are suffering from frequent data base scans and high response time. In order to avoid the drawbacks of the existing Apriori based algorithms the proposed system is introducing a memory resident data structure called "frequent pattern tree" on which the proposed algorithms built.

The proposed system is used for mining frequent itemsets and association rules information from the given synthetic datasets basing on the threshold values of support and confidence. In this system two algorithms called FPGrowth, COFI* are implemented and tested for their performance on various datasets being response time as criteria. Based on functionality, the proposed system can be viewed into three subsystems. The first subsystem deals with the **construction of FP-Tree** for the database based on minimum support and confidence. The second subsystem deals with, **generation of frequent itemsets** using either FP-Growth algorithm or COFI* algorithms. The third subsystem deals with **generation of association rules using frequent itemsets** The system takes 3 inputs and outputs information of frequent patterns and strong rules formed among them. The detailed description of each input is given below.

A. Algorithms Explanation

Frequent Pattern Growth is a different approach to extract frequent itemsets from large transactional databases [1]. It makes use of a data structure, FP-Tree, to limit the number of database scans to two. Being a recursive algorithm, FP-Growth requires a lot of stack space for extracting long frequent patterns. In order to improve the efficiency of mining process, various attempts are made towards devising a non-recursive approach to mining FP-Tree. COFI* is recent non-recursive algorithm to extract long frequent patterns efficiently. We developed a simplified approach to leap traversal of itemset lattice for finding maximal frequent patterns.

B. Frequent Pattern Growth Algorithm

A different approach to frequent itemset mining known as Frequent-Pattern Growth (FP-Growth) [5], does not require to generate candidate generation. This approach uses a memory resident data structure, FP-Tree, to limit the number of database scans to two. In the first database scan, set of global frequent 1-itemset is identified along with their support count based on given minimum support threshold. In the second database scan, remove infrequent items in each transaction to form the corresponding sub-transaction with frequent items are ordered based on their support count i.e. in descending order of their support count and if two items having the same support count then those items are ordered in alphabetic order. These sub-transactions form the paths of the FP-Tree. Each path in the FP-Tree represents an itemset/pattern along with its frequency of occurrence. Sub-transactions that share the same prefix share the same portion of the path starting from the root. Each node of FP-Tree has two field's item name and its support count. All nodes of FP-Tree are connected by bi-directional parent-child links, to traverse FP-Tree in both bottom-up and top-down fashion. The FP-Tree has a header table, which holds global support of an item and a header link to the first occurrence of the item in the FP-Tree and connects nodes of the same item to facilitate the item traversal during the mining process.

Mining of the FP-Tree is done as follows, starting from the frequent item of lowest support (initial suffix pattern), construct its conditional pattern base (a sub database which consists of the set of prefix paths in the FP-Tree co-occurring with the suffix pattern) FP-Tree and perform mining recursively on such a tree. The pattern growth is achieved by the concatenation of the suffix pattern with the frequent patterns generated from a conditional FP-Tree. The FP-Growth method transforms the problem of finding long-frequent patterns to looking for shorter ones recursively and then concatenating the suffix. It uses the least frequent item as a suffix offering good selectivity. This method substantially reduces the search costs. In this way above procedure is repeated for all global frequent items to find set of all its frequent patterns.

Rather than employing the generation-and-test strategy, FP-Growth focus on the frequent pattern growth, which avoids costly candidate generation, resulting in greater efficiency. FP-Growth method is efficient and scalable for mining both long and short frequent patterns. As it is a recursive algorithm, if the database contains very long frequent patterns, stack overflow problem may occur.

C. Co-Occurrence Frequent Item (COFI) Algorithm*

Some of the recent developments concentrated on extracting maximal frequent itemsets/patterns rather than extracting all frequent patterns. A frequent itemset is said to be **maximal frequent** if none of its supersets is also frequent in the transaction database. Since all the subsets of a maximal frequent itemset are obviously frequent, a small set of maximal frequent itemsets can be considered as a concise representation of the set of all frequent itemsets. COFI* algorithm [2] is one of the recent algorithms for finding maximal frequent itemsets/patterns efficiently. Following paragraphs explain merits of COFI* algorithm. Existing algorithms generate candidate frequent itemsets/patterns by using one of the two methods, namely breadth-search or depth-search. Breadth-search can be viewed as a bottom-up approach where the algorithms visit patterns of size k+1 after finishing the k sized patterns. The depth-search approach does the opposite where the algorithm starts by visiting patterns of size k before visiting

those of size k-1. Both methods show some efficiency while mining some databases. On the other hand, they show weakness or inefficiency in many other cases. A combination of these approaches that takes into account of partitioning of the search space of a COFI-Tree. This method is known as leap-traversal approach since it selectively jumps within the search space. Leap-traversal approach is used to find all maximal frequent patterns efficiently that are co-occurred with global frequent item X. It looks at the nature of transaction database of X-COFI-Tree and recommends a set of patterns of different lengths to test where the local maximal frequent patterns are subset of this recommended set. From the local maximal frequent patterns we can generate set of all frequent patterns. The approach is as follows.

Step 1. Look at the nature of the transaction database of X-COFI-tree or frequent-path-bases of X-COFI-tree. It recommends a list of frequent-path-bases whose support is less than minimum support threshold. This list consists a set of candidate patterns of different length.

Step 2. Intersect each one of these patterns with all other frequent-path-bases to get set of potential candidates. In this step, we have to consider patterns of size greater than one only and also check whether generated pattern is already exists or not. If exists nothing needs to be done because we already have the global frequency of this pattern.

Step 3. Count the support of each one of the generated patterns. The support of a pattern is calculated as the summation of branch-supports of all its superset of frequent-path-bases.

Step 4. If the support of the generated patterns is equal or greater than minimum support threshold, then add this pattern to corresponding entry in the OPB (based on pattern's length) with calculated support and branch-support equals to zero, i.e. this pattern represents non-frequent-path-base. In this way leap-traversal approach is used to identify hidden maximal frequent patterns.

II. Experimental Results

FP-Growth algorithm is a highly accepted and successful algorithm for extracting the frequent itemsets based on the pattern growth approach. Due to its recursive nature it fails to extract very long frequent patterns from huge transaction database on simple systems with memory limitations. COFI* is one of the recently developed promising algorithm. It has introduced the concept of Leap Traversal for COFI-Tree mining. FP-Growth, COFI* algorithms were implemented in Java and their performance is tested on Pentium4 1.7GHZ processor, with 256 MBRAM, windows XP. Experimental results are shown in following diagrams for different benchmark datasets namely Retail [13], Chess [14] and Mushroom [15]. During the evaluation of FP-Growth & COFI* algorithms on retail data set support in the range of 1-6 and 90% confidence level has been taken into consideration which is shown in Fig 1. During the evaluation of FP-Growth & COFI* algorithms on chess data set support in the range of 90-98 and 90% confidence level has been taken into consideration which is shown in Fig 2. During the evaluation of FP-Growth & COFI* algorithms on mushroom data set support in the range of 80-95 and 90% confidence level has been taken into consideration. which is shown in Fig 3. From theses graphs we clearly says that COFI* has shown better performance which is evident from the obtained experimental results.

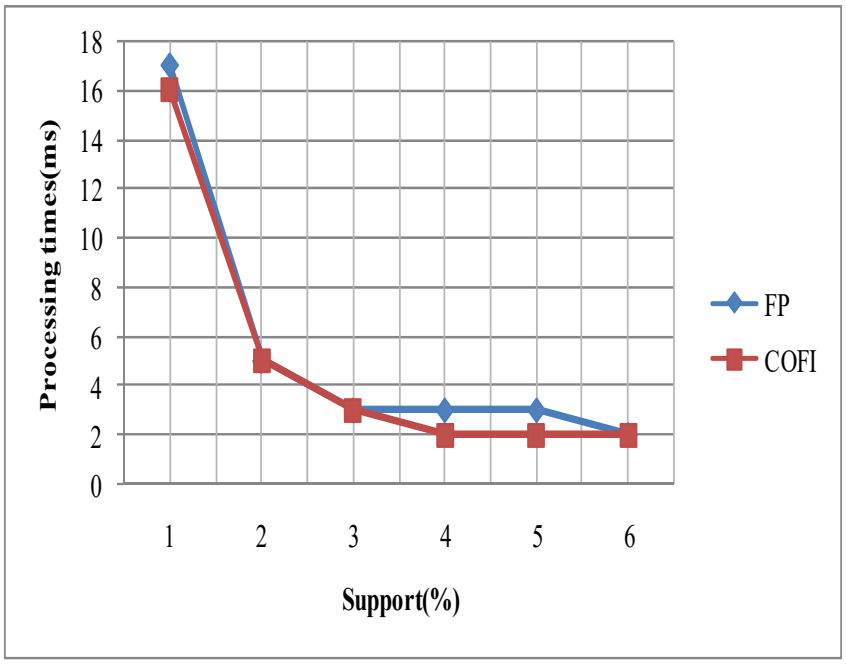

Fig. 1. FP vs COFI for Retail Dat set

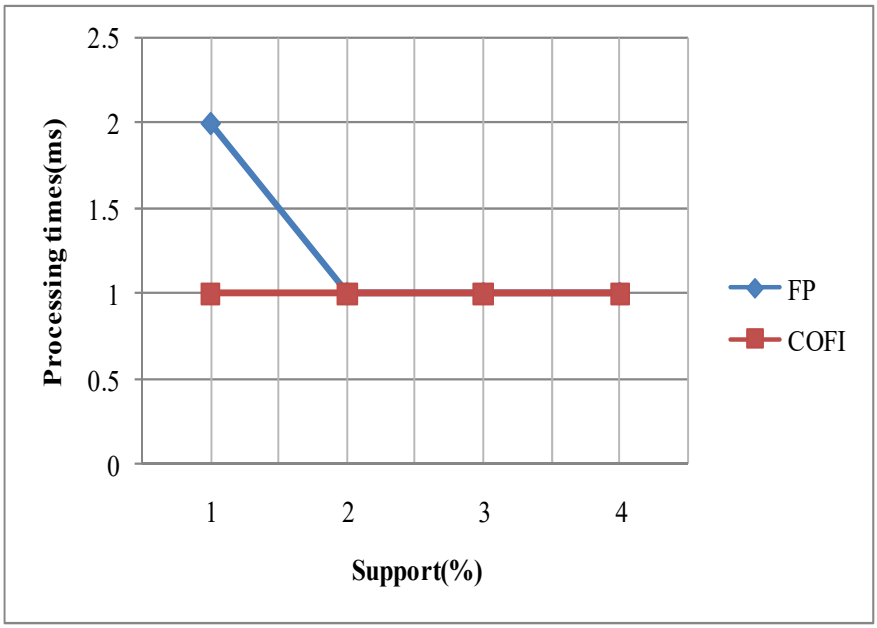

Fig. 2. FP vs COFI for Chess Dataset

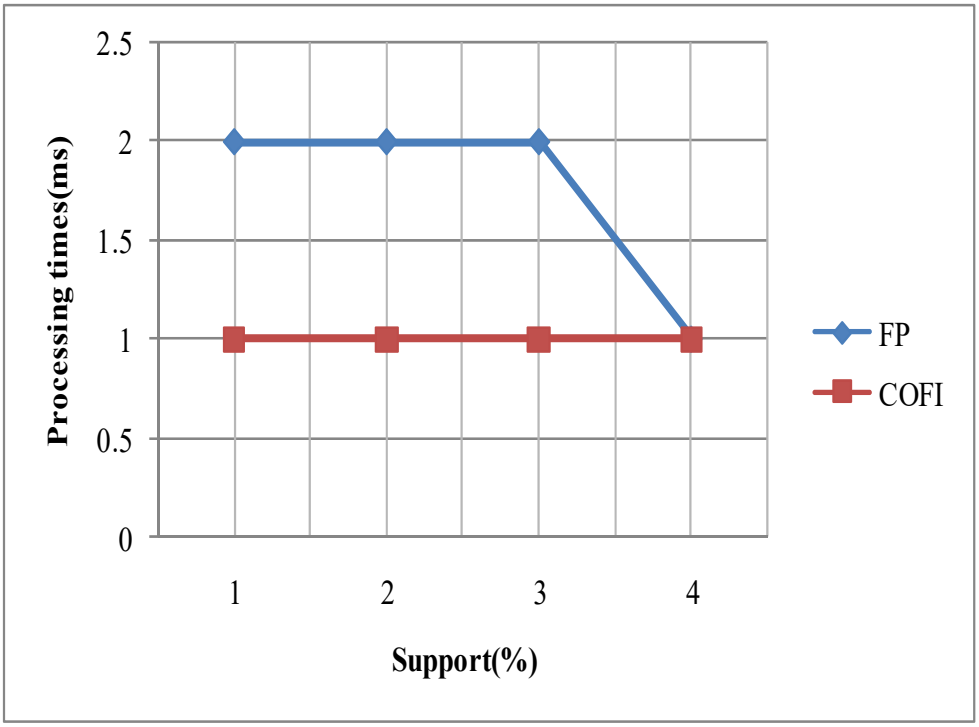

Fig. 3. FP vs COFI for Mushroom Dataset

III. CONCLUSION

Mining for frequent itemsets is a canonical task, fundamental for many data mining applications. Especially for dense data and when fairly long patterns exists, it is difficult to find set of all frequent itemsets. FP-Growth algorithm and COFI* algorithm implemented in this project are efficient algorithms for mining frequent patterns. FP-Growth is not suitable for datasets containing very long frequent itemsets due to its recursive nature where as COFI* is a non-recursive in nature, so it can handle databases containing very long frequent patterns efficiently. COFI* algorithm is based on existing data structures FP-tree and COFI-tree and initiates the process by first identifying maximal patterns using a novel lattice traversal approach. COFI* is non-recursive in nature, so it can handle databases containing very long frequent patterns efficiently. The performance study shows that COFI* outperforms most of the state of the art methods. This algorithm finds the set of exact maximal patterns using only 2 I/O scans of the database, and then generates all frequent patterns with their support. The new traversing approach dramatically minimizes the size of candidate list.

REFERENCES

[1]. Tamura Makio, “Microbial genotype–phenotype mapping by class association rule mining”, pp.1523-1529,2008.

[2]. Mohammad ElHajj, “COFI Approach for Mining Frequent Itemsets Revisited”, pp.70-75, 2004.

[3]. Shi, Xiaodong, “Mining related queries from web search engine query logs using an improved association rule mining model”, Journal of the American Society for Information Science and Technology, pp.1871-1883, 2007.

[4]. Baralis Elena, “Generalized association rule mining with constraints”, Information Sciences, pp.68-84, 2012.

[5]. Cornelia Gyorodi , “A Comparative Study of Association of Association Mining Algorithms”

[6]. Nasraoui, “World wide web personalization”, Encyclopedia of Data Mining and Data Warehousing, 2005.

[7]. Jiao, J, “Association rule mining for product and process variety mapping”, International Journal of Computer Integrated Manufacturing , pp.111-124, 2008.

[8]. Kianmehr, “Fuzzy association rule mining framework and its application to effective fuzzy associative classification”, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery , pp.477-495, 2011.

[9]. Kavšek Branko, “APRIORI-SD: Adapting association rule learning to subgroup discovery”, Applied Artificial Intelligence, pp.543-583, 2006.

[10]. Alves Ronnie, “Gene association analysis: a survey of frequent pattern mining from gene expression data." Briefings in bioinformatics 11, no. 2, pp.210-224, 2010.

[11]. Shi Fuqian , “Employing rough sets and association rule mining in KANSEI knowledge extraction”, Information Sciences, pp.118-128, 2012.

[12]. Liu Chenguang, “Uncertain association rule mining algorithm for the cell formation problem in cellular manufacturing systems”, International Journal of Production Research, pp.667-685, 2009.

[13]. Kim Chulhyun, “Identifying core technologies based on technological cross-impacts: An association rule mining (ARM) and analytic network process (ANP) approach”, Expert Systems with Applications, pp.12559-12564, 2011.

[14]. Nahar Jesmin, “Association rule mining to detect factors which contribute to heart disease in males and females”, Expert Systems with Applications, pp.1086-1093, 2013.

[15]. Shaheen, “Context based positive and negative spatio-temporal association rule mining”, Knowledge-based systems, pp.261-273, 2013.

[16]. Kim, “Recommender system based on click stream data using association rule mining” , Expert Systems with Applications, pp.13320-13327, 2011.

[17]. Martinez, “GenMiner: mining non-redundant association rules from integrated gene expression data and annotations”, pp.2643-2644, 2008.

[18]. Jain, “Supplier selection using fuzzy association rules mining approach”, International Journal of Production Research, pp.1323-1353, 2007.