Volume 3, Issue 1, January 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Enriching the Efficiency of Association Rule Mining using Enhanced IFP-Growth Algorithm

Banu Priya.M*
Department of Computer Science,
Sri Ramakrishna CAS for Women,
Coimbatore, India

Umarani.V
Department of Computer Science
Sri Ramakrishna CAS for Women,
Coimbatore, India

Abstract*— Data Mining is often considered as a process of automatic discovery of new knowledge from large databases. Association Rule Mining (ARM) is one of the important aspects in data mining, which generates large amount of itemsets from the database. The most important approach is the Frequent Pattern growth (FP-growth) algorithm; it is an idea to compress the information for mining FP-tree. The enrichment of the FP-growth algorithm is the Improved FP-growth (IFP) algorithm, which employs an address-table structure and also uses an FP-tree+ approach (top-down search) to lower the complexity of forming the entire FP-tree. The Enhanced IFP-growth algorithm is proposed to improve the process of the FP-tree construction and also it does not generate conditional FP-tree. By fixing the tree split-level, the search is made easier. The experimental result shows that the proposed algorithm is efficient with less computational time and memory space requirement for frequent itemset generation.

Keywords*— Association rule, FP-tree, Frequent itemset generation.

I. INTRODUCTION

Data mining is used to deal with large amounts of data which are stored in the database, to find out desired information and knowledge [2]. It commonly consists of various algorithms namely clustering, classification, association rule mining and more. Among these algorithms, Association Rule Mining (ARM) is one of the most important techniques in the data mining.

Association rule mining (ARM) technique [3], is the most effective data mining technique to discover hidden or desired pattern among large amount of data. The rules are created by analyzing the data for frequent patterns and it uses the criteria support and confidence to identify the most important relationships. Support is an indication of how frequently the items appear in the database. Confidence indicates the number of times the if/then statements have been found to be true.

Generally an association rule mining algorithm [1] contains the following steps:

1. The set of candidate k-item sets is generated by 1-extensions of the large (k-1) item sets generated in the previous iteration.
2. Supports for the candidate k-item sets are generated by a pass over the database.
3. Item sets that do not have the minimum support are discarded and the remaining item sets are called large k-item sets.

This process is repeated until no large item sets are found.

Association Rule Mining techniques have been widely used in various applications such as marketing, modern business, medical analysis and website navigation analysis [6]. ARM algorithms aim at extracting interesting correlations, frequent patterns, associations or casual structures that satisfy a predetermined minimum support and confidence, among items present in transaction databases or other data repositories.

The remainder of this paper is structured as follows. Chapter 2 presents the existing methodology and Chapter 3 describes the proposed methodology of the research. Chapter 4 discusses the experiments and results achieved. Chapter 5 summarizes the conclusion and future direction of the research.

II. EXISTING METHODOLOGY

This section 2.1 illustrates the original FP-growth algorithm and section 2.2 describes an existing methodology, the improved FP-growth algorithm which efficiently derives frequent itemsets from a database.

A. FP-Growth Algorithm

The FP-Growth Algorithm is an alternative way to find frequent itemsets without using candidate generations and it uses divide-and-conquer strategy [15].The core of this method is that it uses a special data structure named frequent-pattern tree (FP-tree), which retains the itemset association information. In simple words, this algorithm works as follows: first it compresses the input database creating an FP-tree instance to represent frequent items. After the first step, it divides the compressed database into a set of conditional databases, each one associated with one frequent pattern.

Finally, each such database is mined separately. Using this strategy, the FP-Growth reduces the search costs looking for short patterns recursively and then concatenates them in the long frequent patterns, which offers good selectivity. The Table1 represents the transactional database, which consists of transaction ID, items bought and the ordered frequent itemsets.

Table 1. The Transactional Dataset

TID	Items bought	ordered frequent items
1	f, a, c, d, g, i, m, p	**f, c, a, m,p**
2	a, b, c, f, l, m, o	**f, c, a, b, m**
3	b, f, h, j, o	**f, b**
4	b, c, k, s, p	**c, b, p**
5	a, f, c, e, l, p, m, n	**f, c, a, m, p**

The minimum support for the above table is specified as 3. Based on the minimum support, the items are arranged in the decreasing order for the construction of an FP-Tree and the item which does not meet the minimum support value are discarded from the tree construction. The Figure 1 represents the FP-Tree generation for the above table.

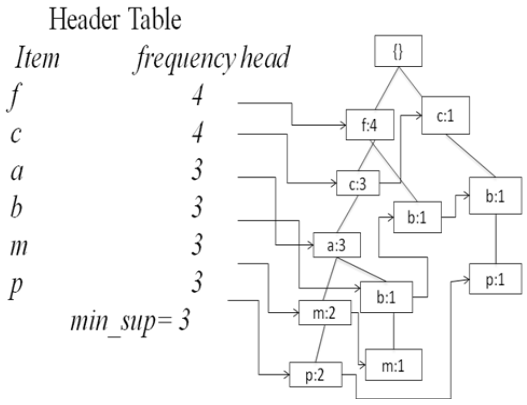

Figure 1. An FP-Tree

B. Improved FP-Growth Algorithm

The Improved FP-growth (IFP) algorithm [17], for frequent itemsets generation incorporates the FP-tree+ mining technique and the address-table into FP-growth. It employs a hybrid method to mine frequent itemsets. The frequent itemsets in a single path is first discovered and placed in the FP-tree. In IFP-Mining, a hybrid technique is used for mining purpose, which is the combination of FP-tree+ and the conditional FP-tree technique. A tree split level is randomly chosen for mining purpose. Above the tree split-level FP-tree+ technique and below the tree split-level conditional FP-tree is used.

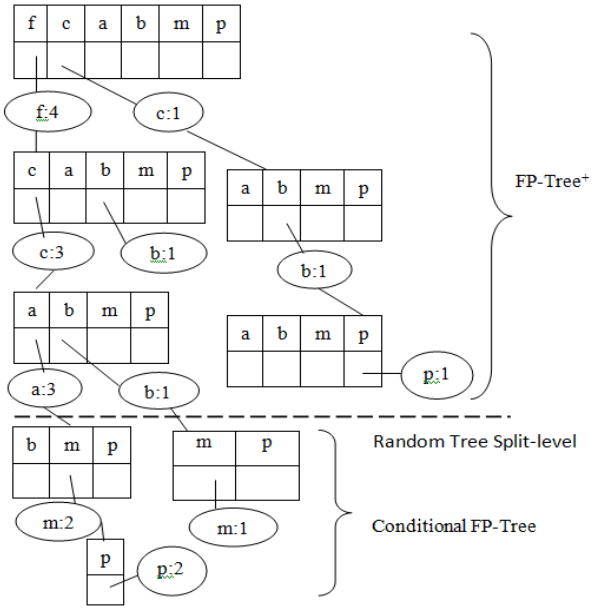

Figure 2. An Improved FP-Tree

The Figure 2 represents the Improved FP-tree which differs from the FP-tree on splitting the tree for the Table 1. Each node in the IFP-tree contains a Node structure and an address table. The Node structure has item name, count, node-link, and a pointer. The FP-tree+ is used for reducing the needs for rebuilding FP-trees in each conditional pattern mining step. The mining procedure of FP-tree+ is similar to that of the conditional FP-trees, but the direction of mining an FP-tree+ is opposite to that of mining a conditional FP-tree. It discovers frequent itemsets by traversing the items from the top to the bottom of a header table whereas the conditional FP-tree mines in bottom-up fashion. According to the feature, each FP-tree+ can be built on the original FP-tree and the memory requirements can be reduced. The major advantages of FP-tree+ and the address-table are that they reduce the need to rebuild conditional FP-trees and facilitate the task of tree construction.

III. PROPOSED METHODOLOGY

The Enhanced IFP-Growth consists of three phases: In first phase, it scans the transactional database only once for generating equivalence classes of frequent items. The equivalence class of an item can be defined as the occurrence of the particular item in all transaction of the database. The equivalence class is constructed for the entire items that are presented in the database. In second phase, it consequently sorts the equivalence classes of frequent items in descending order and filter out non-frequent items. Finally in third phase, the Enhanced IFP tree is constructed and the tree split-level is fixed in order to extract the frequent itemsets.

There are the four rules for constructing the enhanced IFP tree. Each time when a new node is added into the tree, all four rules are taken into consideration. The rules are described as follows:

Rule 1: At first, the root node is checked whether there is child node or not. If there is no child node, then the new node becomes a child to the root node and if there is a child node then, new node is compared with the child node.

Rule 2: At first the list of nodes (equivalence class) in the new node and the list of nodes (equivalence class) in the child node are compared, If it covers only the list of nodes in the child node partially or fully then the new node become the child node of that node. Else the new node becomes the child node for the root node.

Rule 3: If new node is different from the existing root nodes, then it becomes sibling node of the root node or it becomes the child node for the sibling node.

Rule 4: If new node resembles the ancestor nodes partially, then it is split into two nodes.

After the tree construction is completed, the tree split-level is assigned for fastening the search. The formula for tree split-level is

$$\text{TREE SPLIT} - \text{LEVEL} = \frac{\text{NUMBER OF NODES IN THE TREE}}{2(\textit{NUMBER OF DIVISIONS FROM THE ROOT NODE})}$$

In Figure 3, the tree split-level is assigned. The number of nodes in tree is 11 and the number of divisions from the root node is 2. On using the above formula, the tree split-level is 11/2(2) = 2.75 and approximately the value 3 is chosen as a tree split-level. For extracting the frequent itemsets from the database, the two search methods called top-down and bottom-up methods are used to perform the search efficient.

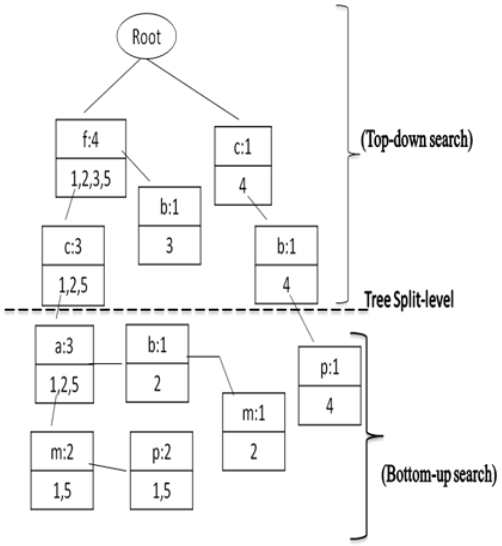

Figure 3. An Enhanced IFP-Tree

The Figure 3 represents the Enhanced IFP-tree with the fixed Tree split-level. Above the tree split-level top-down search and below the tree split-level bottom-up search is performed to make the search easier. Thus, the algorithm will work efficiently for tree construction process in terms of time and memory space requirements.

IV. RESULTS & DISCUSSIONS

This section gives an overview of the conducted experiments and offers the acquired results to evaluate the performance of the basic FP-growth algorithm, the IFP-growth algorithm and the proposed Enhanced IFP-growth algorithm in terms of time and memory consumptions. The datasets which are used for conducting researches are Retail dataset, Generated transactional dataset, Insurance dataset, and Supermarket dataset. Table 2 summarizes the characteristics of the data sets, which are considered for conducting the experimentation.

Table 2. Dataset Characteristic

Datasets	No. Of Transactions	Items	Size(MB)
Retail	5133	470	0.9
Generated Transaction	1000	200	0.4
Insurance	9822	86	1.7
Supermarket	785	231	0.3

The first set of experiment is conducted with the Supermarket dataset, which is a real-time dataset collected from a departmental store, located in Coimbatore. It specifies about the customers behaviour of purchasing an items. The supermarket dataset contains 4 attributes and they are products name, date, quantity, and price. For experiment, only the name of products and its occurrences are taken into account for process. The dataset contains 785 transactional records and 231 items. The experiment was conducted to evaluate the performance of basic FP-growth, Improved FP-growth and the proposed Enhanced IFP-growth algorithms in terms of time (in seconds) and memory (in bytes).

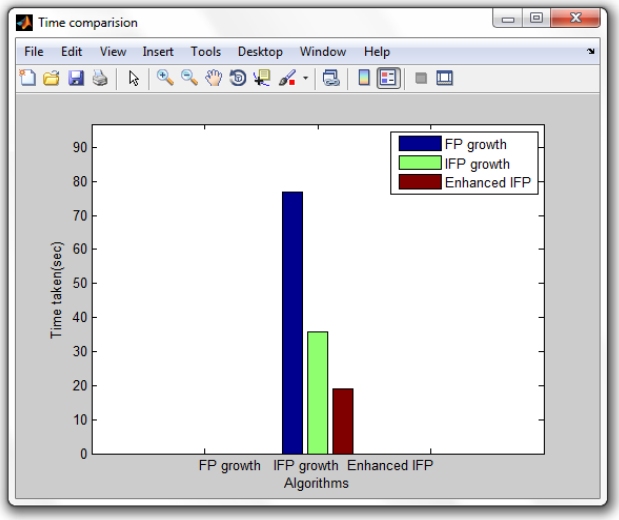

Figure 4. Time Comparison for Supermarket Dataset

The Figure 4 represents the Time Comparison for the Supermarket dataset. On analysing the dataset in various assumptions, the minimum support is specified as 5% and the confidence is specified as 20%. It shows that the more the transaction data in the record is, the more time it takes to execute and at the same time, time and memory difference between basic FP-growth and Improved FP-growth and Enhanced IFP growth algorithm is varied. In particular, the Enhanced IFP-growth outperforms well when compared to FP-growth and Improved FP-growth in terms of execution time.

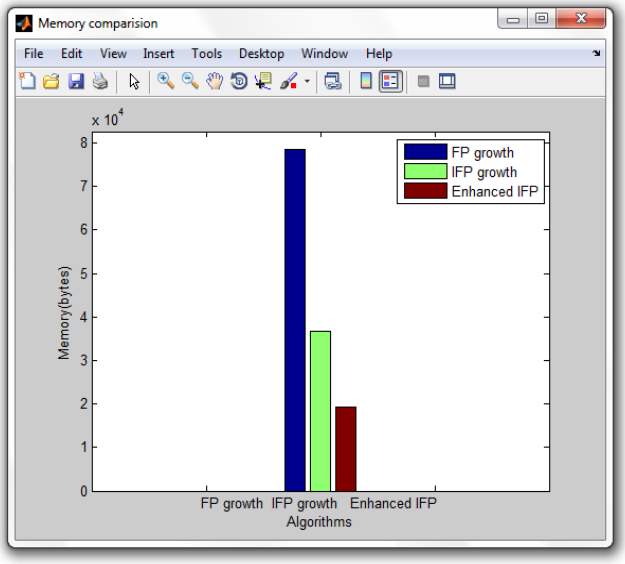

Figure 5. Memory Comparison for Supermarket Dataset

The Figure 5 represents the Memory Comparison for the Supermarket dataset. The proposed algorithm Enhanced IFP-growth performs better when compared to other existing algorithm in terms of memory space requirement.

Table 3. Performance Comparisons for Supermarket Dataset

ALGORITHIMS	**TIME(secs)**	**MEMORY(bytes)**
FP-Growth	76.8024	78646
Improved FP-Growth	35.8856	36747
Enhanced IFP-Growth	18.9326	19387

The performances of the basic FP-growth, Improved FP-growth and Enhanced IFP-growth algorithms in the supermarket dataset are compared and represented in the Table 3. The time and memory space requirement of the proposed algorithm is reduced when compared to the other algorithms. The number of rules generated by the above algorithms for the supermarket dataset is 111.

The Enhanced IFP-growth is very efficient algorithm for mining all frequent itemsets, based on the experimental results in terms of runtime and memory consumption. When the data set is dense, Enhanced IFP-growth has a better speed performance than FP-growth and Improved FP-growth algorithm, and its memory requirement is lower. Even if the minimum support becomes low, the Enhanced IFP-growth algorithm remains efficient when compared to other algorithms. Thus, the Enhanced IFP-growth algorithm is very suitable for high performance applications.

V. CONCLUSIONS AND FUTURE ENHANCEMENT

An efficient algorithm for mining association rules called Enhanced IFP-growth is developed, which is superior to FP-growth and Improved FP-growth construction algorithm. There are three reasons such that the proposed algorithm outperformed well than the other construction algorithm in terms of tree construction. The first one is that our proposed method scans the database only once. The second one is that sorting the items in each transaction record is not employed. The third one is that the header table and links will not be repeatedly searched, while adding a new node in the tree. Finally, the searching performance is increased on fixing the tree-split level. To perform searching the tree traversal techniques, top-down and bottom-up search is used. Thus, the proposed approach is more efficient in terms of time and memory while constructing and searching the tree. In future, the bitmap structure and F-miner can be implemented to improve the performance of association rule mining.

REFERENCES

[1] Agarwal. R, & srikant. R (1994), "Fast Algorithm for Mining Association Rules in Large Database", In proceedings of 20th VLDB conference (pp.487-499).

[2] Arun K pujari, "Data Mining Techniques", Universities press (INDIA) private limited.

[3] Chen-Feng Lee & Tsung-Hsien Shen(2005), "An FP-Split Method for Fast Association Rule Mining", IEEE,0-7803-8932-8

[4] Farah Hanna AL-Zawaidah, yosef Hasan Jabra, Marwan AL-abed abu zohona(2011), "An Improved Alogarithm for Mining Association Rules in Large Database", World of computer science and information techonology journal (WCSIT), ISSN:2221-0741 Vol.1, No, 7, 331-316.

[5] Frawley .W, piatetsky-Shapiro .G Mathews .c., (1992), "Knowledge Discovery in Databases: An Overview."AI Magazine, Fall 1992,Pp.213-228.

[6] (FIMI)Frequent Itemset mining dataset Repository, http//fimi.ua.ac.be/data/retail.dat(Accessed on June 2012)

[7] Jiawei Han, Jian pie, Yiwen Yin, Runying Mao (2001)e,"Mining Frequent Patterns Without Candidate Generation:A Frequent-Pattern Tree Approach*", Data mining and Knowledge Discovery, 8, 53-87.

[8] Jia-Ling Koh and Shui-Feng Shieh., (2004), "An Efficient Approach for Maintaining Association Rules Based on Adjusting FP-Tree structures", In proceedings of International conference on Database Systems For advanced Application, Jeju Island, Lectures Notes in Computer Science,Vol.2973,pp.417-424, Springer-Verlag

[9] Ke-Chung Lin, I-En Liao, Zhi-Sheng Chen (2011), "An Improved Frequent Pattern Growth Method for Mining Association Rules", Expert Systems With applications, 5154-5161.

[10] Raymond Chi- Wing Wong, Ada Wai-Chee Fu (2003), "Association Rule Mining and its Application to MPIS".

[11] Rezbaul Islam.A.B.M& Tae-sun Chung (2011), "An Improved Frequent Pattern Tree Based Association Rule Mining Technique", IEEE, 978-1-4244-9224-4.

[12] Racz.B.(2004), "Nonordfp:An FP-Growth Varition Without Rebuilding The FP-Tree", In Proceedings of IEEE ICDM Workshop on Frequent itemmining implementatations Overview", GETS International Science Inc, New York, USA, 179(5), pp.559-583.

[13] K.Sotiris and D.Kanellopoulos (2006), "Association Rule Mining: A Recent Transaction on Computer Science and Engineering", Vol-32(1), pp-7182.

[14] Srikant.R, Vu .Q and Agarwal .R(1997), "Mining Association Rules with Item Constraints", In the proceedings of 3rd Intl Conf on Knowledge Discovery and Data Mining.

[15] Syed Khairuzzaman Tanber, Chowdhury Farhan Ahemeda, Byeong-Soo Jeong, and Young-Koo Lee., (2009,) "Efficient Single-Pass Frequent pattern Mining Using a Prefix-Tree", An International Journal Of Information Science,Elsevier

[16] UCI Machine Learing Repository http://Kdd.ics.uci.edu/database (Accessed on July 2012).

[17] V.Umarani and M.Punithavalli (2012), "A Study on Effective Mining of Association Rules from Huge Database", IJCSR International Journal of Computer science and Research, Vol. 1 Issue 1.

[18] Vaibhav Kant Singh, Vijay shah, Yogendra kumar Jain, Anupam Shukla, A.S Thoke, Vinay Kumar Singh, Chhaya Dule, Vivek Parganitha (2008), "Proposing an Efficient Method for Frequent Pattern Mining", World Academy of Science, Engineering and Technoloy.

[19] Wang, L. Tang, J. Han and J. Liu (2002), "Top-Down FP-Growth for Association Rule Mining", Lecture Notes in Computer Science Springer Berlin, Eidelberg Vol.2336, pp.334-340.

