A Novelty Approach for Finding Frequent Itemsets in Horizontal and Vertical Layout- HVCFPMINETREE

A.Meenakshi
Research Scholar,
Dept of MCA, Computer Center
Madurai Kamaraj University,
Madurai, India.

Dr.K.Alagarsamy
Associate Professor,
Dept of MCA, Computer Center,
Madurai Kamaraj University,
Madurai, India.

ABSTRACT
In the modern world, we are faced with influx of massive data. Though such trend is most welcome, it poses a challenge to space-time requirement. So the imperative need is to find more efficient algorithms to manage such problem. There are so many existing algorithms to find frequent itemsets in Association Rule Mining. In this paper, we have modified FPTree algorithm as HVCFPMINETREE (Horizontal and vertical Compact Frequent Itemset Pattern Mining Tree). HVCFPMineTree combines all the maximum occurrence of frequent itemsets before converting into the tree structure. We have explained it with algorithm and illustrated with examples in horizontal data format and vertical data format

GENERAL TERMS
Association Rule Mining, Knowledge Discovery, Dataset Organization, Itemsets, Horizontal layout, Vertical Layout, Frequent Patterns.

Keywords
InFreq, FreTD, MOFI, MaxTrans, MOI, SL.

1. INTRODUCTION
1.1. DATA MINING

In the 20th century, we are in the world of handling massive explosion of digital databases. There is always hype when a promising new technology appears. Data Mining is no exception. It blossoms when the pressure is on to gild the lily. Data Mining can help uncover trends in time to make the knowledge actionable. Knowledge discovery includes the data analysis that must be performed to discover the most powerful and relevant select factors for a specific problem. Fayyyad also defined KDD as the process of identifying valid, novel, potentially useful, and ultimately understandable patterns or models in data. Data mining is the process of discovering knowledge out of pile of huge data. The focus of Data mining is to reveal information that is hidden and unexpected. Data mining techniques represent a blend of statistics, pattern recognition and machine learning. As pointed out by Chen et. al (1996), data mining techniques can be classified by different criteria such as databases to work on (relational databases, object-oriented databases, etc), Knowledge to be mined (e.g. association rules or characteristic rules) and Techniques to be utilized (e.g. data-driven or query-driven)

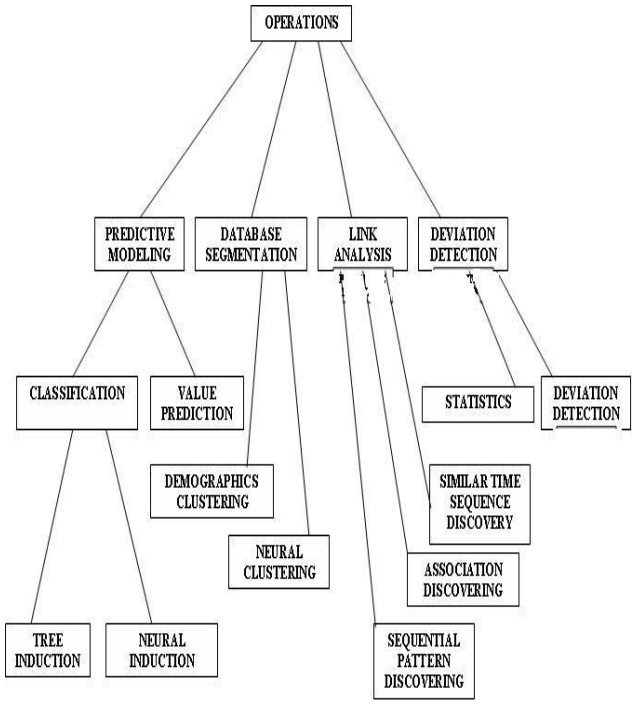

Fig.1 Data Mining Techniques.

With the explosion of data growth, swelling up of digital databases and data warehouses leads to Terabytes of pet bytes and the trend towards further increase. The bitterness behind this massive explosion of growth of data is the scalability of data mining techniques. Therefore, finding scalable algorithms will solve the problem within a reasonable time. Association rule mining was originally applied in Market-basket Analysis which aims at understanding the behavior and shopping preferences of retail customers [1];the concept of Association rules was popularized n the year 1993 by Agrawal.FIM algorithms could be broadly classified as candidate generation algorithms or pattern growth algorithms. Within these categories, further classification can be done based on traversal strategy and data structures used. Frequent pattern mining is the discovery of relationships or correlations between items in a dataset. A set of market basket transactions such as transaction database consists of row as well as column. Each row is a transaction, identified by a transaction

identifier or a TID. A set of transaction might be organized in either on enumerated (dense), or a sparse binary vector format. Data mining tastes two kinds of scalability issues: row (or database size) scalability and column (or dimension) scalability. The row scalability is sometimes referred to as "curse of cardinality" and the column scalability problem is referred to as the "curse of dimensionality".

1.2. Dataset Organization

Dataset organizations can be processed horizontally or vertically. For several decades and especially with the pre-eminence of relational database systems, data is almost always formed into horizontal record structure and then processed vertically. In a horizontal enumerated data organization, each transaction contains only items positively associated with a customer purchase. In a horizontal layout, the database is organized as a set of rows, with each row representing a customer's transaction in terms of the items that are purchased in the transaction.

There is an alternative approach to this data layout such as vertical layout. It consists of each item associated with a column of values representing the transaction in which it is present. It has smaller effective database size, compact storage of the database and better support of dynamic database.

A market-basket database is a two dimensional matrix where the rows represent individual customer purchase transaction and the columns represent the items on sale.

TID	LIST OF ITEMS
1	{MILK,SUGAR, WHEAT}
2	{MILK, ICECREAM,SUGAR,FLOUR}
3	{MILK,SUGAR}
4	{ICECREAM,MILK,WHEAT}
5	{ICECREAM,WHEAT,SUGAR}

Fig.2 Transactions Database

TID	LIST OF ITEMS
1	M, S, W
2	M , I, S
3	M, S
4	I, M, W
5	I, W, S

TID

LIST OF ITEMS			
M	S	I	W
1	1	2	1
2	2	4	4
3	3	5	5
4	5		

Horizontal Layout **Vertical Layout**

Recently attention has been given to the influence of data organization on the performance of the process of frequent pattern discovery. The discovery of interesting relationships hidden in large datasets is the objective of frequent pattern mining

2. ASSOCIATION RULE MINING

There are several algorithms for finding frequent patterns. Association rule mining first mooted by Agrawal has now become one of the main pillars of data mining and knowledge discovery tasks. It is a method of finding relationships of the form x→y itemsets that occur together in a database where X and Y are disjoint itemsets [2]. Support and confidence are two key measures for association rule mining. The association rule indicates that the transactions that contain X, tend to contain Y support.

Let I = {I1, I2, …Im} be a set of items. Let D, the task relevant data, be a set of database transactions where each transaction T is a set of items such that T C J.

Support (A=>B) = P (AUB)

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

2.1. FOOTPRINTS OF PATTERN MINING ALGORITHMS FOR ASSOCIATION RULES

Many algorithms for generation association rules were presented over time. Some well-known algorithms are Apriori, Apriori Tid, AprioriHybrid, AIS and SETM were discussed in the horizontal data format. The Apriori and AprioriTID algorithm, introduced by Agrawal et al. relies on generation and test approach of candidates. The best features of the two proposed algorithms can be combined into a hybrid algorithm called Apriori Hybrid. AIS and SETM have always been outperformed by the Apriori and Apriori TID algorithms

The FP growth algorithm is the most popular method which was developed by Han J.et al. FPGrowth Algorithm adopts a divide and conquer strategy to decompose both the mining tasks and the database. It compresses the database representing frequent items into a frequent pattern free or fp tree but, retains the itemset association information and then divides

such a compressed database into a set of conditional databases. There are two scans needed for mining all frequent itemsets [7]. It uses an extended prefix tree structure to store the database in a compressed form. It uses a pattern fragment growth to avoid the costly process of candidate generation and testing used by Apriori.

Borgelt et al. has presented an algorithm called SAM (Split and Merge Algorithm) for frequent itemset mining. In their algorithm, transactions are sorted lexicographically in descending order. This algorithm computes a conditional database, process recursively and eliminates the split item from the original database.

Vertical layout Algorithms

Shenoy et al. described the advantages of vertical organization over the horizontal organization; in their paper they discussed VIPER (Vertical Itemset Partitioning for Efficient Rule Extraction). It stores data in compressed bit vectors called snakes. They proved VIPER has an excellent scalability, dynamic counting, compact storage and better support of dynamic databases [17]. Zaki has presented a novel vertical data representation called Diffset that only keeps track of differences in the tids of a candidate pattern from its generating frequent patterns [14]. He showed the performance of diffsets drastically cutting down the size of memory required to store intermediate results. He has also introduced GenMax algorithm that utilizes a novel backtracking search strategy for efficiently enumerating all maximal itemsets. This method recursively navigates to find the maximal itemsets at high levels without computing the support value of all their subsets. He has presented charm, which mines closet sets [21]; dcharm performs a novel search for closed sets subset properties of diffsets .Mingjun Song et al. has presented an algorithm called Transaction Mapping [TM]. In their algorithm, transaction ids of each itemset are mapped and compressed to continuous transaction intervals in a different space [19].This algorithm has outperformed FPGrowth and DEclat on the basis of runtime and storage cost.

2. PROPOSED SYSTEM

The existing algorithm of FPGrowth has two drawbacks. First, mining of frequent itemsets from the FPTree generates huge conditional FP-tree and takes a lot of time and space. Secondly, if we can change the minsupport count, this algorithm may restart and scan database twice. We have proposed a new approach such as HVCFPMineTree to mine all frequent itemsets; it improves the FP tree algorithm. It requires a very less memory space and it is quite easy to traverse a tree structure. It employs more efficient searching, compact memory structure than the FP tree

3.1 Proposed Method

Definition1 (Transaction Database)

Let the transaction database be T= {t1, t2….tn}. Where Ti is a set of items.

Example1

We have worked with the following transaction database used by Han [10]

TID	ITEMS BOUGHT
1	f a c d g i m p
2	a b c f l m o
3	b f h j o w
4	b c k s p
5	a f c e l p m n

Fig.3**. List of Transaction Items**

It consists of five transactions such as TID {T1, T2, T3, T4, T5} and eleven items are present in the transaction database.

Definition2 (Support and MinSupport threshold)

The support of an item i, denoted by supp (i) is the number of transactions containing i.

Minsupport threshold or minsup is the threshold defined by the user for selection of frequent items.

Definition3 (Frequent Itemsets and Infrequent itemsets)

An Item I is called frequent item if supp(x) ≥minsup; otherwise it is called infrequent Item.

Definition 4 (Prefix Paths)

A child node has a link with root node; the root node is called prefix paths.

Definition 5 (Count)

The maximum number of occurrence of items ε_i is called Count**.** Σ is the total count for element ε_i in database Transaction Database.

.

3.2 PROBLEM STATEMENT

For many years, the research has been going on to find interesting relationships hidden in large databases. The problems such as i) Compact Storage of huge databases ii) Possible effect on processing time iii) Managing efficiently massive frequent itemsets in a very compressed data storage format are present in the data mining algorithms. We should make data searching more effectively by using an efficient algorithm .Our main objective is to make a very compact frequent pattern tree and store it in a compressed and efficient memory structure. We have presented a novelty algorithm such as Horizontal and Vertical Compact Frequent Pattern Mine Tree (HVCFPMINETREE). The existing frequent pattern tree FPTREE) has been modified and presented with algorithms and examples.

3.3 ILLUSTRATION OF PROPOSED APPROACH

In this section, we have presented a novelty approach for finding frequent itemset mining. For example, a customer who is buying milk can also buy bread in the supermarket.

PROCEDURE

1. First create the root of the tree as null.
2. Scan the database and count the number of occurrence of each item.
3. According to the maximum occurrence of items, group them all itemsets.
4. Make a link with the root.
5. In the first level, all the items which have maximum occurrence are grouped and connected with the root node.
6. Combine the remaining itemsets and make a prefix path node with the previous node.
7. Continue steps 5 and 6 until all the nodes have been inserted.

3.4 MINING OF FREQUENT ITEMSETS BY USING HVCFPMINETREE

In this subsection1, we have presented a new algorithm to mine frequent itemsets without generation of candidate itemsets. The proposed algorithm leads to efficient searching and compact frequent pattern tree.

FREQUENT COUNT OF ITEMS

ITEMS	SUPPORT COUNT
A	3
B	3
C	4
F	4
M	3
P	3

Arrange according to descending order and also check with supportcount >= minsupport; then we can enter the values according to the maximum occurrence of transactions

ITEMS	SUPPORT COUNT
F	4
C	4
A	3
B	3
M	3
P	3

In the transaction database also, we can arrange according to the maximum occurrence of items.

TID	ITEMS BOUGHT
T1	f c a m p
T2	f c a b m
T3	f b
T4	c b p
T5	f c a m p

We can scan the transaction database to combine all the frequent items and insert it as the level 1 of the HVCFPMinetree

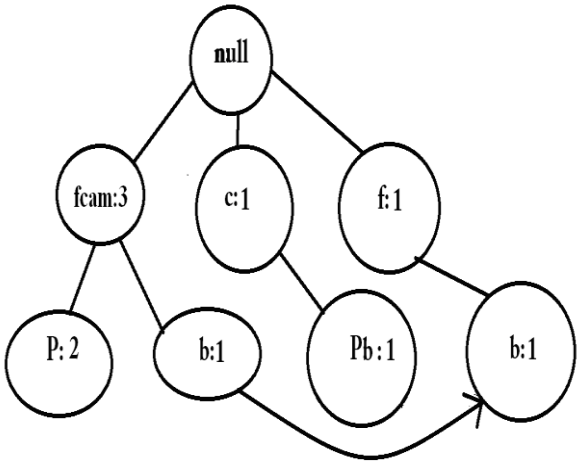

Fig. 4 HVCFPMINETREE

HVCFP(PREMine) Tree algorithm consists of the following steps:

Terms:

InFreq – Infrequent (or) pre frequent itemsets

FreqItem- frequent itemsets

TD- Transaction Database and FreTD – Frequent Items Transaction Database, MOFI – Maximum Occurrence of Frequent Itemsets.

Steps (PREMINE Frequent Itemsets)

1. Present the transaction list in the horizontal transaction format.
2. Combine all the items and present in the Cfp Mine.
3. Calculate the occurrence of each of all items until the end of the TD.
4. Check If Support Count (FreqItem) >= MinSupportCount. Then store it to FreTD Else perform step 7.
5. Perform step3 until items (i) in the TD are equal to items (n).
6. Union all FreqItem and store the output in FreTD.
7. Repeat step 4 for n times.

Construction of HVCFPMineTree (Horizontal Layout)

Algorithm HVCFPMineTree (HVCfpmine frequent Itemsets with HIL)

Input: Horizontal Item list (HIL), Sort list as SL.

Output: The complete set of frequent itemset in compact tree format.

Method: Call PREMINE (LS, HIL, k)

Begin

For each FreqItem ki in the FRETD

Sort according to the maximum occurrence of the items as SL.

While the first item in SL <> last item

Combines all the maximum occurrence of frequent items as MOFI in the HIL in the order of the items in the SL.

MOFI=FS1UFS2UFS3

Repeat the above steps until all the items are combined

End //while

End //for

End//Begin

Procedure HVCFPMINETREE (PreMine)

Begin

Create root node as null

For I to 1 to MOFI

Insert all the MOFI in the CFPMineTree

Make a link with the root node

Repeat for MOFI steps and insert it as next level.

Continue the above steps for the remaining maximum itemsets.

Maintain Itemset Association information as in Transaction database.

Continue until all the Items are inserted.

Store items to HVCFPMINETREE.

End for

3.5. *VERTICAL DATA FORMAT*

The same example we have taken into account was discussed by Han [10].Transaction database and list of items already discussed in the horizontal data format are presented in the vertical data format.

LIST OF ITEMS					
	B	C	F	M	P
T1	T2	T1	T1	T1	T1
T2	T3	T2	T2	T2	T4
T5	T4	T4 T5	T3 T4	T5	T5

Count the list of Transactions and it can be listed below

TID	
T1	5
T2	5
T3	2
T4	4
T5	4

Check with support count >= minsupport; then we can enter the values according to the maximum occurrence of transactions

LIST OF TRANSACTIONS					
LIST OF ITEMS	F	T1	T2	T3	T4
	C	T1	T2	T4	T5
	A	T1	T2	T5	
	B	T2	T3	T4	
	M	T1	T2	T5	
	P	T1	T4	T5	

3.5.1 STEPS FOR VERTICAL HVCFPMINE TREE

1. First scan the database for finding maximum combination of occurrences of all items.
2. Combine all these combinations and name it as MaxTrans. Insert it as level 1 in the HVCFPMINETREE
3. Link all the remaining Itemsets with the associated related Items as prefix path

CONSTRUCTION OF HVCFPMINETREE (VERTICAL APPROACH)

1. Count the number of occurrences of all items.
2. Check with supportcount >= minsupport. If so, sort the items according to the count.
3. Combine all the maximum occurrences of items as itemset as MOI.
4. Create the root node as null node. Insert MOI in the HVCFPMINETREE as the first level of the tree.
5. Repeat steps3 for all MOI items.
6. Insert the remaining MOI, as the next level of the tree and make a link with prefix path of level1.
7. Continue step 5 and 6 until all the items are inserted in the HVCFPMINETREE.

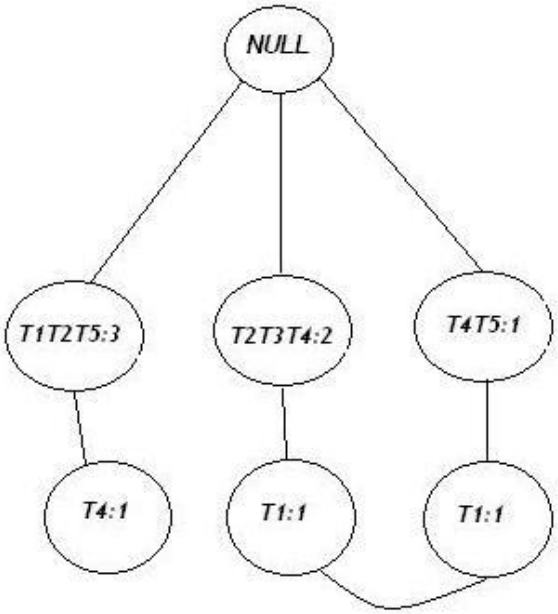

Fig.5 VERTICAL HVCFPMINETREE (TRANSACTION TREE)

Algorithm HVCFPMineTree (HVCFPMINE frequent Itemsets with VIL)

Input: Vertical Item list (VIL), Sort list as SL.

Output: The complete set of frequent itemset in compact tree format.

Method: Call PREMINE(LS,VIL, k)

Begin

For each FreqItem ki in the FRETD

Sort according to the maximum occurrence of the items as SL.

While the first transaction in SL <> last transaction

combine all the maximum occurrence of frequent transaction items as MaxTrans in the VIL in the order of the items in the SL.

MaxTrans=FT1UFT2UFT3

Repeat the above steps until all the transactions are combined

End //while

End //for

End//Begin

Procedure Famine Tree (PreMine)

Begin

Create root node as null

For I to 1 to MaxTrans

Insert all the MaxTrans in the CFPMineTree and make a link with the root node.

Repeat for MOFI steps and insert it as next level.

Continue the above steps for the remaining maximum transactions list.

Maintain Itemset Association information as in Transaction Database.

Continue until all the Items are inserted.

Store items to HVCFPMINETREE

End for

End//Begin

4. CONCLUSION

In our proposed work, we have presented a new algorithm HVCFPMineTree in both horizontal and vertical layouts. It leads to a compressed FPTree structure in an efficient manner. We have explained HVCFPMineTree with algorithms and illustrated with examples. In our future work, we will implement and compare it with the existing pattern mining algorithms.

5. REFERENCES

[1]. R.Agrawal, T. Imielinski and A. Swami, "Mining association rules between sets of Items in large databases", in proceedings of the ACM SIGMOD International conference on Management of data, pp. 207-216, 1993.

[2]. R. Agarwal and R. Srikant, "Fast Algorithms for Mining Association Rules"proc.20th International Conference on very large Databases, pp 487-499,1994.

[3]. R. Agarwal, C. Aggarwal and V.V.V. Prasad: "A Tree projection Algorithm for Generation of Frequent Itemsets". Journal of parallel and Distributed computing (Special issue on high performance data mining) 2000.

[4]. D. Burdick, M. Calimlin and J. Gehrke, "MAFIA: A Maximal frequent Itemset Algorithm for Transactional Databases,"Proc. International Conference on Data Engineering, PP 443-452, April 2001.

[5]. M.S. Chen, J. Han and P.S.YU," Data Mining: An Overview from a Database Perspective", IEEE Transaction on Knowledge Data Engineering 8(6), 866-897(1996).

[6]. J-Cios, W. Pedrycz and R. Swiniarski, Data Mining Methods for knowledge Discovery, Kluwer, Boston, 1998.

[7]. Erwin, A. Gopalan, R.P., and Achuthan, N.R., "CTU-Mine: An Efficient High Utility Itemset Mining Algorithm Using the Pattern Growth Approach", IEEE 7th International conferences on computer and Information Technology, pp 71-76, 2007.

[8]. B. Goethals, "Survey on Frequent Pattern Mining" manuscript, 2003

[9]. J. Han, J. Pei, Y. Yin and R. Mao, "Mining Frequent Pattern without Candidate Generation: A Frequent Pattern Tree", Springer, volume 8, 53-87, 2004.

[10]. Jiawei Han et al, Data Mining: concepts and Techniques, Morgan Kaugmann publishers, 2001.

[11]. Jiawei Han, Jian pei, Yiwen Yin, Runying Mao, "Mining frequent patterns without candidate Generation: A frequent-pattern tree Approach" Data Mining and Knowledge Discovery, volume 8, Issue 1, pp 53-57 , January 2004..

[12]. B. Kalpana, Dr. R. Nadarajan, Optimizing Search Space Pruning in Frequent Itemset Mining with Hybrid Traversal strategies - A comparative performance on different data organizations, IAENG, 2007..

[13]. Laila A. Abd EI. Megid et al, "Vertical Mining of Frequent Patterns using Diffset Groups", International. Conference on Intelligent systems Design and Applications.

[14]. Mohammed J. Zaki and Karam Gouda, "Fast Vertical Mining using Diffsets", In proceedings of the ninth ACM SIGKDD International Conference On Knowledge Discovery and Data Mining, Washington, D.C, 326-335, 2003.

[15]. H. Mannila, Theoretical Frameworks for Data Mining SIGKDD Explorer 1(2), 30-32(2000)

[16].Nittaya Kerdprasop and Kittisak Kerdprasop,Mining frequent patterns with functional programming, and, Internal Journal of computer and Information science and Engineering, 2007.

[17]. Pradeep Shenoy et al, Turbo-Charging Vertical Mining of large Databases , IISC, Technical report, DSL,2000.

[18]. A. Pietragaprina and D. Zandolin, "Mining Frequent Itemsets Using Patricia Tries", proc. ICDM 2003 Workshop Frequent Itemsets Mining Implementations, December 2003.

[19]. M. Song and Rajasekaran , "A Transaction Mapping Algorithm for Frequent Itemsets Mining", IEEE transactions onknowledge and Data Engineering, vol 18, No. 4, 2006.

[20]. Tiwari.A, R.K.Gupta and Agrawal, "A Survey on Frequent Pattern Mining: Current Status and Challenging Issues", Information Technology Journal, pp: 1278 – 1293, 2010.

[21]. M.J Zaki, C.J. Hsiao, "CHARM. An Efficient Algorithm for Closed Association Rule Mining," Technical report 99-10, computer science department Resselaer polytechnic Institute, October 1993.

[22].M.J. Zaki, S.Parthasarthy, M.Ogihara and W.Li. Parallel algorithm for discovery of association rules, Data Mining and Knowledge discovery, 1:343-374, 1997.

[23] ZHENGXIN CHEN, Data Mining and Uncertain Reasoning: Integrated Approach, Wiley-Interscience Publications, 2001.