EFFICIENT MINING OF HIGH UTILITY PATTERNS USING FREQUENT PATTERN GROWTH ALGORITHM

P. Asha[1], T. Jebarajan[2]

ABSTRACT

Data mining aims at extracting only the useful information from very large databases. Association Rule Mining (ARM) is a technique that tries to find the frequent itemsets or closely associated patterns among the existing items from the given database. Traditional methods of frequent itemset mining, assumes that the data is centralized and static which impose excessive communication overhead in case of distributed data, and computational resources are wasted, if the data is dynamic in nature. To overcome this, Utility Pattern Mining Algorithm is proposed, in which itemsets are maintained in a tree based data structure, called as Utility Pattern Tree, which generates the itemset without examining the entire database, and has minimal communication overhead when mining with respect to distributed and dynamic databases. Hence, it enforces the execution to be at a faster rate, means reduced cost and time.

Keywords - Data Mining, Association Rule Mining, High Utility Mining, Rule Generation, Filtering.

I. INTRODUCTION

Data mining is a widely used approach to transform data into useful pattern. Handling and mining huge amount of records seems to be difficult with traditional data mining techniques. The process of mining data into useful patterns took longer duration. After finding out the association rules, filtering out the best rules is really challenging [1].

There exists an imbalance in the current data mining techniques from the performance perspective like Algorithm imbalance, Pattern imbalance, and Decision imbalance [14]. Therefore, to find the best rules out of the huge rule set, several Rule Interestingness measures were considered. There are two basic interestingness Measures, Subjective and Objective measures. Subjective measures of Interestingness states the belief of the user. Objective measures work on the data and the structure of rule in a discovery procedure. Support and Confidence are objective measures [12]. There exist many association rule mining algorithms like Apriori, FP-Tree [2], Dynamic Hashing and Pruning, Dynamic Item Set Counting [15], SEAR and SPEAR [11], PEAR, Eclat, MaxEclat etc. Many parallel rule mining algorithms like parallel Eclat, MaxEclat were put forth [13]. After which various up gradations to

[1]Research Scholar, Computer Science and Engineering Department, Sathyabama University, Chennai. ashapandian225@gmail.com

[2]Head of the Department , Computer Science & Engineering, Rajalakshmi Engineering College, Chennai. drtjebarajan@gmail.com

frequent pattern mining and high utility mining [7] came into existence.

Y.Liu, W-K.Liao, A.Choudhary proposed a two phase algorithm to find high utility itemsets. Only two database scans were needed to find frequent itemsets. The first database scan finds the 1-itemset wu(transaction weighted utilization) and from this 2-itemsets transaction weighted utilization can be found. The second database scan, finds the 3-itemsets based on 2-itemsets. The drawback of this approach is level wise candidate generation and test methodology which consumes time [3], [9].

J Hu et al developed a paper containing algorithm for frequent item set mining that identifies the high utility item combinations. The goal of this algorithm is to find segments of a data, defined through combinations of some items (rules), which satisfy certain conditions as a group and maximize the predefined objective. Whereas in the traditional association rule and frequent item mining techniques, the high utility pattern mining conducts a better rule discovery with respect to individual attributes and also the overall criterion for the mined set, attempting to find groups of such patterns that contribute the most to a predefined objective function [4], [9].

Liu Jian-ping, Wang Ying, Yang Fan-ding et al proposed an algorithm for tree based incremental association rule mining algorithm. It is based on a FUFP (fast update frequent pattern) mining method. The advantage of FUFP is that it reduces the number of candidate set while moving on to updating procedure. In FUFP all links are bidirectional. Due to this, addition/removal of child node would be easy without much reconstruction of tree structure. The algorithm classifies the items into the three categories: frequent, infrequent and pre-large. Pre-large itemsets has two supports threshold value (i.e. upper and lower threshold). The drawbacks of this approach is that the entire procedure is time consuming as it requires multiple passes for its processing [5], [6].

Shih-Sheng Chen et al proposed a method for finding frequent periodic pattern using multiple minimum supports which is based on real time event. The items in the transactions are arranged according to their minimum item support. Another approach PFP (periodic frequent pattern) was used which is similar to FP-Growth and using the conditional pattern base the frequent items are generated. This algorithm is more efficient in terms of memory space, reducing the number of database scans [8], [9].

II.Proposed System

Fig. 1 shows general block diagram of proposed high utility mining method. A user can purchase the items and all purchased items are stored in transaction history which contains day to day transaction. Based on it Utility pattern tree is constructed based on the algorithm described below. Find the promising and unpromising items based on the user specified threshold and consider only the promising items for pattern mining. Based on the frequent patterns obtained, generate association rules and then filter those rules as there may be redundant rules based on the confidence.

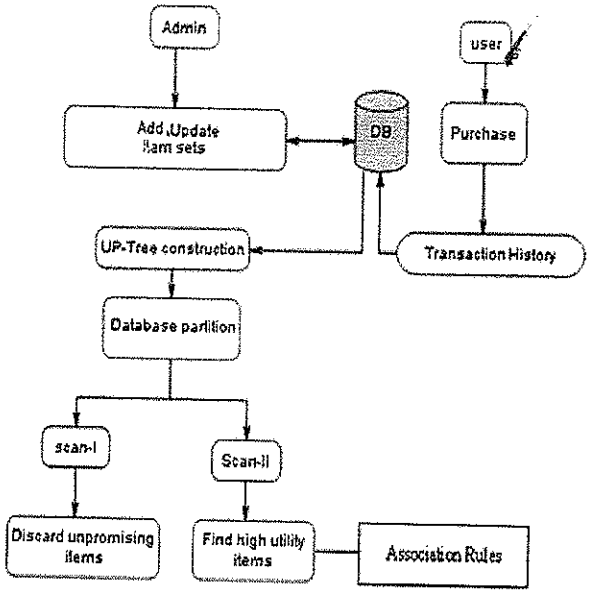

Figure 1 : Architecture Diagram

A. Definitions and Parameters

Consider a Transaction database $T = \{T_{100}, T_{200}, T_{300} \ldots T_n\}$ and every individual transaction consists of various Items and the total items present in the entire database be represented as $I = \{I_1, I_2, I_3, \ldots I_m\}$.

	I_1	I_2	I_3	I_4
T_{100}	5	4	7	9
T_{200}	2	3	14	1
T_{300}	10	2	5	8
T_{400}	3	10	12	16

(a)

ITEMS	PROFIT
I_1	3
I_2	4
I_3	7
I_4	2

(b)

Figure 2 : Sample of (a) Transaction Database and (b) Utility Table.

1) Internal Utility (IU) is represented as the quantity of item I_k in the transaction T_j.

2) External Utility (EU) is denoted by the profit of the particular item I_k

3) Utility is defined as Util (Ik, Tj) = IU * EU.

4) Utility of any itemset is given by

$$Util(I_k, T_j) = \sum_{I \in I_s}^{n} Util(I_k, T_j)$$

5) The transaction utility(tu) of a transaction Tj is the total profit of that transaction.

$$Tu(T_j) = \sum_{I_k \in T_j} Util(I_k, T_j)$$

6) The transaction –weighted utilization of an itemset X, is the sum of the transaction utilities of all transactions containing X.

$$twu(X) = \sum_{X \subseteq T_j \in D} Tu(T_j)$$

7) The transaction frequency (tf) of an item represents the number of transactions in which the item appears.

From Fig. 2, the internal utility of Item I_3 in the transaction T_{300} is 5, external utility is 7 and Util is 5*7=30.The $tu(T_{300}) = Util(I_1, T_{300}) + Util(I_2, T_{300}) = (10*3)+(2*4)=38$.

B. High utility mining Algorithm :

There are three phases of mining. The phase I explains the tree construction. Phase 2 states the creation of conditional pattern base and condition FP-tree, from which the frequent patterns are generated. Phase 3 explains the mining of high utility patterns.

Phase 1 :

The FP-Tree is created as follows:

- The entire transaction database is scanned only once. Then try to retrieve all the one- frequent itemsets, that is the individual itemsets and arrange them in descending order based on the support count, every item posessess. Let it be called as Itemlist L and the Frequent Itemsets are called as F.
- Now create a Tree with its root as NULL. Then for all transcations, sort the frequent itemlist as [P|C],

where P is the Parent element and C is the child element. Continue the procedure until all transcations are appended into the tree structure. Then Call the procedure insert_into_Tree([P|C],T), which can be performed as follows. If T has a child N such that N. itemname = P.itemname, then increment N's count by 1; else create a new node N, and let its count be 1, and its parent link be linked to T, and its node links to nodes with the same item name via the node-link structure. If P is nonempty, call insert_into_Tree(P,N) recursively.

Phase 2 :

The FP Tree is mined by calling FP-Growth (FP_Tree, null), which is implemented as follows [10].

Procedure FP-Growth (Tree, α)

- If the Tree contains a single path P then
- For every item combination(β) of the node in the path P
- Generate pattern β U α with support_count= Min_supp count of nodes in β;
- Else for a_i in the header of Tree{
- Generate pattern β= a_i U α with support_count=a_i, support_count;
- Then prepare conditional pattern base and conditional FP Tree of β;
- If Tree ≠ Ø then,
- Call FP_Growth (Tree, β);}

Phase 3:

- For every transaction T_j
- Sort the items and perform the required data manipulation operations and update the twu and tf in the table.
- Based on the mining request, chek whether the existing α > received α then
- For every item , check If twu of that item >= minutil then create tree (Phase 1)and then perform mining Else delete that item.

III. Experimental Results And Performance Evaluation

The promising items obtained as a result of the algorithm are a1, a3, a4. Next, the rules are generated by the promising items only.

Fig. 3 gives the pictorial representation of the best rules generated by the ARM. Out of 98 rules generated only rules 2, 10, 12, 1, 9, 11 (total=6) were concluded as the best rules.

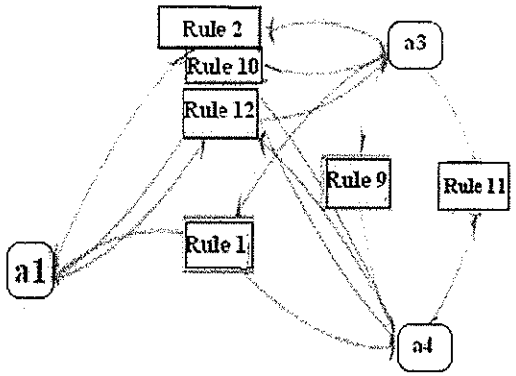

Figure 3 : Generated Rules

Fig. 4 indicates that out of all the items present in the database a4 is the actively participating node and the rules generated by it are rule 1, 2, 12, 10. That is a4 contributes more while generating and filtering rules. It has provided 4 best rules from a total of 6 best rules and filtration of rules.

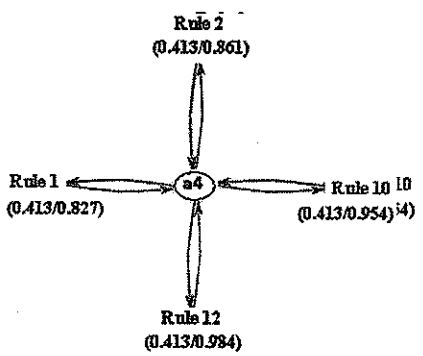

Figure 4 : Most Promising item and its rules

The rules are filtered based on various interestingness measures like lift, support, confidence, LaPlace, Gain, P-S, Conviction etc.

Fig. 5 displays the filtered best rules where the conclusion is the item a3. And these are the best combinations as these rules have a lift >1 and confidence >80% and support >40% and conviction>4 and p-s >0.1.

No.	Premi..	Conclusi..	Sup..	Confi..	LaPla..	Gain	p-s	Lift	Conviction
2	a4	a3, a1	0.413	0.861	0.955	-0.547	0.205	1.987	4.080
6	a1	a3, a4	0.413	0.886	0.964	-0.520	0.196	1.898	4.667
8	a1	a3	0.433	0.929	0.977	-0.500	0.200	1.857	7
11	a4	a3	0.467	0.972	0.991	-0.493	0.227	1.944	18
12	a4, a1	a3	0.413	0.984	0.995	-0.427	0.203	1.968	31.500

Figure 5 : Filtered rules of item a3

Fig. 6 displays the filtered best rules where the conclusion is the item a4.

No.	Premi..	Conclu..	Supp..	Confi..	LaPla..	Gain	p-s	Lift	Conviction
6	a1	a3, a4	0.413	0.886	0.964	-0.520	0.196	1.898	4.667
7	a1	a4	0.420	0.900	0.968	-0.513	0.196	1.875	5.200
9	a3	a4	0.467	0.933	0.978	-0.533	0.227	1.944	7.800
10	a3, a1	a4	0.413	0.954	0.986	-0.453	0.205	1.987	11.267

Figure 6 : Filtered rules of item a4

Fig. 7 displays the filtered best rules where the conclusion is the item a1.

No.	Prem..	Concl..	Supp..	Conf..	LaPla..	Gain	p-s	Lift	Conviction
2	a4	a3, a1	0.413	0.861	0.955	-0.547	0.205	1.987	4.080
3	a3	a1	0.433	0.867	0.956	-0.567	0.200	1.857	4
4	a4	a1	0.420	0.875	0.959	-0.540	0.196	1.875	4.267
5	a3, a4	a1	0.413	0.886	0.964	-0.520	0.196	1.898	4.667

Figure 7 : Filtered rules of item a1

IV. Conclusion

The algorithm offers a better performance when compared to the basic methods available. As a future enhancement, parallelization of the task and incremental mining can be appended. As the size of the tree grows, it becomes difficult to mine the frequent pattern from such a huge tree, so if it can be done in parallel, then the total execution time would be reduced. And as the transactions would be continually increasing and may be the same items already present in the tree would reappear in next few transactions, a better algorithm that mines only the incremented tree structure may be required to yield a better performance.

References

[1] R. Agrawal, T. Mielinski, *"Mining association rule between sets of items in large databases"*, in the Proceedings of the ACM International Conference on Management of data, (1993), pp. 207-216.

[2] Grahne, Gösta, and Jianfei Zhu. *"Fast algorithms for frequent itemset mining with FP-Trees,"* In the Transactions on *Knowledge and Data Engineering, IEEE, Vol.*, 17, No.10 (2005): 1347-1362.

[3] Y.Liu, W.K. Liao and Choudhary, *"A two phase algorithm for fast discovery of high utility item set"*, LNCS. PP: 689-695, 2005.

[4] J.Hu, A. Mojsilovic, *"High utility pattern mining: A method for discovery of high utility itemssets"*, in Pattern Recognition. PP: 3317-3324, 2007.

[5] Li, Yu-Chiang, Jieh-Shan Yeh, and Chin-Chen Chang. *"Isolated items discarding strategy for discovering high utility itemsets." Data & Knowledge Engineering* 64.1 (2008): 198-217.

[6] Jian-Ping, Fan-Ding, *"Incremental Mining algorithm Pre-FP in Association Rule Based on FP-tree"*, Networking and Distributed Computing, International Conference, pp: 199-203, 2010.

[7] Ahmed CF,Tanbeer SK, Jeong B-S, Lee Y-K (2011) *"HUC-Prune: An Efficient Candidate Pruning Technique to mine high utility patterns"* Appl Intell PP: 181–198, 2011.

[8] Chen, Shih-Sheng, Tony Cheng-Kui Huang, and Zhe-Min Lin. *"New and efficient knowledge discovery of partial periodic patterns with multiple minimum supports,"* in the Journal of Systems and Software, No.84, Vol. 10, (2011): 1638-1651.

[9] P. Asha, Dr.T. Jebarajan and G. Saranya, *"A Survey on Efficient Incremental Algorithm for Mining High Utility Itemsets in Distributed and Dynamic Database,"* in the International Journal of Emerging Technology and Advanced Engineering, ISSN: 2250-2459, Vol. 4, Issue. 1, pp. 146 - 149,January2014. http://www.ijetae.com/files/Volume4Issue1/IJETAE_0114_25.pdf

[10] Han, Micheline Kamber, *"Text book on Data Mining, II[nd] Edition: Concepts and Techniques,"* year: 2006.

[11] Zaki, M. J., *"Parallel and distributed association mining: A survey,"* in the *Concurrency, Special Issue on Parallel Mechanisms for Data Mining (IEEE),, Vol.* 7, Issue. 4, pp.14—25, 1999.

[12] Avi Silberschatz and Alexander, *"What makes the patterns interesting in Knowledge Discovery Systems," IEEE Trans. on Knowledge and Data Engineering,* Vol. 8,No. 6,Dec 1996..

[13] Rakhi Garg, Mishra, *"Exploiting Parallelism in Association Rule Mining Algorithms,"* in the Intl. Journal of Advancements in Technology, Vol. 2, No. 2, April 2011,pp. 222-232.

[14] Longbing Cao, *"Challenges and Prospects: Domain-Driven Data Mining," in the Trans on Knowledge and Data Engineering, IEEE,* Vol. 22, No. 6, 2010, pp. 755-769.

[15] Brin et al., "*Dynamic Itemset Counting and Implication Rules for Market Basket Data*," Proc. ACM Conf. Management of Data, pp. 255–264, (1997),

Author's Biography

P. Asha, is an Asst. Professor and Research Scholar of Computer Science and Engineering Department, Sathyabama University, Chennai, Tamilnadu, India. Her area of interest are Data

Mining and Grid Computing. She has published many journal papers both National and International on Data mining and Hetergeneous Computing. ashapandian225@gmail.com

Dr. T. Jebarajan, is the Head of the Department, Computer Science & Engineering, Rajalakshmi Engineeing College, Chennai, Tamilnadu, India. His area of interest includes Networking and System Programming. He has publised many journal papers both National and International on Networking drtjebarajan@gmail.com