An Efficient Approach to Discover High Utility Itemset from Transactional Database

Shekhar Patel[1], Krunal Panchal[2]
[1] Department of Computer Engineering, LJIET, Ahmedabad, Gujarat, India
[2]Assistant Professor, Department of Computer Engineering, LJIET, Ahmedabad, Gujarat, India
[1]shekharpatel67@gmail.com, [2] krunaljpanhal@gmail.com

ABSTRACT

Data Mining is the process of evaluating data from different outlooks and summarizing it into useful information. It can be defined as the process that extracts information contained in very large database. Traditional Data mining methods have been focused on to finding a correlation between items which are frequently appearing in the database. And relative importance of each item is not consider in frequent pattern mining. High utility mining is an area research where utility based mining can be done. Mining high utility itemset from a transactional database refers to the discovery of itemset with high utility in a terms like weight, unit profit or value. In past algorithm used to mine high utility itemset takes more time to execute and memory. In this paper we proposed an algorithm which uses a tree structure and an array to store the transaction value and then proposed algorithm find the high utility itemset. An extensive experimental study with four real- life datasets shows that our proposed algorithm executes faster than FHM algorithm.

Keywords *- High Utility, Transactional Database, Transaction Weighted Utilization (TWU), FHM*

1. INTRODUCTION

DATA mining is the process of revealing nontrivial, previously unknown and potentially useful patterns form large database Data mining can be used to transform the raw data into meaningful and useful information for business analysis processes referred to as Business intelligence. Business leaders have realized that "getting closer to the Customer" is crucial to the growth of the business. Discovering useful patterns hidden in a database plays an essential in several data mining task such as frequent pattern mining, weighted itemset mining, and high utility mining. Among them frequent pattern mining is a fundamental research topic [1]that has been applied to different kind of database such as transactional database , streaming database and time series database.

Nevertheless, relative importance of each item is not considered in frequent pattern mining. However the important limitation of FIM is that it assumes that each item cannot appear more than once in each transaction and that all items have the same importance (weights, unit profit or value)[2] address this problem, weighted association rule mining was proposed [3]. In this framework, weights of items, such as unit profits of items in transaction databases, are considered. With this concept, even if some items appear infrequently, they might still be found if they have high weights. However, in this framework, the quantities of items are not considered yet. Therefore, it cannot satisfy the requirements of users who are interested in discovering the itemsets with high sales profits, since the profits are composed of unit profits, i.e., weights, and purchased quantities. In other words, statistical correlation may not measure how useful an itemset is in accordance with user's preferences (i.e. profit). The profit of an itemset depends not only on the support (total number of items an itemset occur in a transactional database out of the total number of transactions) of the itemset, but also on the prices of the items in that itemset. The above limitation motivated the researchers [4] to develop a utility based itemset mining approach, which allows a user's to conveniently express his or her perspectives concerning the usefulness of itemset with utility values larger than threshold. Utility based data mining is a new research area [3] interested in all types of types of utility factors in data mining process.

Mining high utility itemset from database refers to finding the itemset utility is interestingness, importance, or profitability, of an items to users. Utility of items in a transactional database consist of two aspects: 1) the importance of distinct item [1], which is called *external* utility, and 2) the importance of item in transactions, which is called *internal* utility..

Utility of an itemset is defined as the product of its external utility and internal utility [5]. An itemset is called a high utility itemset if its utility is no less than a user threshold minimum utility threshold; otherwise it is called a low- utility itemset [3]. Mining high utility itemset from database is an important task has a wide range of applications such as website click stream analysis [1] business promotion in chain hypermarkets, cross marketing in retail stores [1] online e-commerce management, mobile commerce environment planning and even in biomedical applications.

2. BACKGROUND

In this Section, we first give some definition and define the problem of utility mining, and then introduce related work in high utility mining.

2.1 Preliminary

Given a finite set of items I = {i_1, i2, im}, each item i_p has a unit profit pr (i_p). An itemset X is a set of k distinct items {i_1, i2,…..,i_k}. An itemset with k items is called as k-itemset. A transaction database D = {T1, T2, . . ., Tn} contains a set of transactions, and each transaction Td($1 \leq d < n$) has a unique identifier d, called TID. Each item ip in transaction Td is associated with a quantity q (i_p, T_d), that is, the purchased quantity of ip in Td.

Definition 1: Internal Utility of an item ip in a transaction Td is denoted as u (ip, Td) and defined as pr (ip) * q (ip, Td).
For example, in Transaction T1, internal utility of item A is 5.

Definition 2(Utility of an item in Transaction): Utility of an itemset X in T_d is denoted as u(X; T_d).
For example, utility of an item An in Transaction T1, is u (A, T1) = 5*4 = 20.

Definition 3(Utility of an item/itemset in Database). Utility of an itemset X in D is denoted as u(X). For example, utility of itemset *{a,b}* is *u* (*{a, b}*) = *u* (*a*) +*u* (*b*) = *u* (*a*, *T*4) + *u* (*a*, *T*5) + *u* (*a*, *T*8) + *u*(*c*, *T*4) + *u*(*c*, *T*5) + *u*(*c*, *T*8) = 8+12+16 + 24+56+40 =156.

Definition 4(Transaction Utility). Transaction Utility (TU) of a Transaction T is sum of all its items utility in a given Transaction.
For Example in Transaction T1, TU (T1) = 5*4 + 3*10 = 20+30 = 50.

Definition 5(Transaction Weighted Utilization: The Transaction weighted utilization (TWU) of an itemset X is the sum of all the Transaction Utility (TU) of a transaction which contains an itemset X.

Definition 6 (High Utility Itemset): An itemset X is called high utility itemset, if its TWU value is more than defined threshold value.

Example:

Example of a transaction database representing the sales data and the profit associated with the sale of each unit of the items.

TID	Item A	Item B	Item C
T1	5	0	10
T2	0	6	0
T3	4	0	1
T4	2	3	8

T5	3	7	6
T6	3	0	1
T7	0	6	0
T8	4	5	25
T9	3	0	0
T10	0	5	2

Table 1: Transaction Database

Item Name	Unit Profit
Item A	4
Item B	8
Item C	3

Table 2: Unit Profit

Let us consider the itemset AB. Since, there are only 3 transactions T4, T5 and T8 which Contains AB itemset out of 10 transactions. So, support for itemset AB is

Support (AB) = 3 / 10 * 100 = 30 %

In T4 transaction, units gain by item A and B are 2 and 4. Respectively, the profit earned from the sale of itemset AB. In T4 transaction is given by,

Profit (AB, T4) = 2 * profit (A) + 4 * profit (B)

= 2*4 + 3*8

= 32

Since AB appears in transactions T4, T5 and T8, So, total profit of itemset AB is given by

Profit (AB) = profit (AB, T4) + profit (AB, T5) + Profit (AB, T8)

= (2*4+3*8) + (3*4+7*8) + (4*4+5*8)

= (8+24) + (12+56) + (16+40)

= 32+68+56

= 156

Similarly, we can calculate the support values for the different itemsets and also the profit obtained by the sale of those itemsets by all the ten transactions as indicated in table 3.

Itemset	Support (%)	Profit
A	70	96
B	60	256
C	70	159

AB	30	156
BC	40	283
AC	60	237
ABC	30	273

Table 3: Support and Profit

If we consider minimum support 50%, then we can observe that there are only 4 itemsets A, B, C and AC which have the support greater than the threshold value (min_sup). So, they qualify as frequent itemsets. But if we consider it profit wise then we can find out of 4 most profitable itemsets B, BC, AC, ABC only B and AC are frequent itemsets. Itemsets BC and ABC are not frequent but still they fetch the more profit than other itemsets.

As we can see from table 3 single unit of item B fetch more profit than single unit of Itemset A and B.

From this Example, we can illustrate frequent Itemset mining may not always satisfy profit wise requirements of sales manager .In this case, the support (%) attribute of the Itemsets reflects the statistical correlation not the semantic significance of items.

2.2. RELATED WORK

In this section, some related researches are briefly reviewed. They are the FP-growth algorithm and utility mining.

1.1. *The FP-growth algorithm*

Data mining involves applying specific algorithms to extract pat- terns or rules from data sets in a particular representation. One common type of data mining is to derive association rules from transaction data, such that the presence of certain items in a trans- action will imply the presence of some other items. To achieve this purpose, Agrawal et al. proposed several mining algorithms based on the level-wise processing to find association rules [7]. Han et al. then proposed the Frequent-Pattern-tree structure (FP tree) and FP-growth algorithm for efficiently mining frequent itemsets without generation of candidate itemsets [11].

The FP-tree mining algorithm consists of two phases. The first phase focuses on constructing the FP tree from a database, and the second phase focuses on deriving frequent patterns from the FP tree. Three steps are involved in FP-tree construction. The data- base is first scanned to find all items with their counts. The items with their support equal to or larger than a predefined minimum support threshold are then selected as frequent 1-itemsets (items). Next, the frequent items are sorted in the descending order of their frequencies. At last, the database is scanned again to construct the FP tree according to the sorted order of frequent items. The construction process is executed tuple by tuple, from the first transaction to the last one. After all transactions are processed, the FP tree is completely constructed.

After the FP tree is constructed from a database, the FP-growth mining algorithm [10] is then executed to find all frequent itemsets. A conditional FP tree is generated for each frequent item, and the frequent itemsets with the processed item can be recursively derived from the FP-tree structure. Several other algorithms based on the pattern-growth procedure have also been proposed and some related researches are still in progress [12].

1.2. *Utility mining*

In association-rule mining, only binary itemsets are considered in a database. In real-world applications, however, frequent item- sets just reveal the occurrence of itemsets in transactions, but do not reflect any other important factors, such as prices or profits. High profitable products but with low frequencies may thus not be found in traditional association-rule mining. For example, jewels and diamonds are high profitable items but may not be frequent when compared with food or drink products in a database. Yao et al. then proposed the utility model by considering both quantities and profits of items [5]. Chan et al. proposed the topic of utility mining to discover high utility itemsets [13]. Liu et al. then presented a two-phase algorithm for fast discovering high utility item- sets by adopting the downward-closure property [8] and named his approach as the transaction-weighted-utilization (TWU) model. It consisted of two phases. In the first phase, the transaction utility was used as the effective upper bound of each candidate itemset such that the ‘‘transaction-weighted downward closure’’ could be kept in the search space to decrease the number of generated candidate itemsets. In the second phase, an additional database scan was performed to find out the actual utility values of the candidates and to identify the high utility itemsets. Thus, the main idea is to reduce the size of candidates in order to decrease the time of scanning a database. Several other algorithms for utility mining were also proposed [1]

Philippe Fournier et al. [2] presented a novel algorithm for high utility itemset mining named FHM (Fast High-Utility Miner). This algorithm integrates a novel strategy named EUCP (Estimated Utility Co-occurrence Pruning) to reduce the number of join operations when mining high utility itemset using the utility list data structure. It works only on static database. We should try to develop it for dynamic database also.

The proposed utility mining approach thus integrates the two- phase approach and the FP-tree concept to efficiently and effectively find high utility patterns. A new tree structure is designed and a tree-construction algorithm is proposed. They are first de- scribed below.

3. PROPOSED ALGORITHM

The High Utility Tree construction algorithm is first proposed to keep the high utility items found from a database in the tree structure based on the downward-closure property. The proposed algorithm first calculates the transaction utility of each transaction. It then finds the transaction-weighted-utilization values of all the items. If the transaction-weighted-utilization of an item is larger than or equal to the predefined minimum utility threshold, it is thus well-thought-out as a high transaction-weighted 1-itemset. The algorithm then keeps only the high transaction-weighted 1-item-sets in the transactions and sorts them according to their transaction frequencies. The updated transactions are then used to build the HUP tree tuple by tuple, from the first transaction to the last one. Each node in the tree has to store the transaction-weighted-utilization of the item as well as the quantities of its preceding items (including itself) in the path. An array is then attached to a node to keep those values. The Flow diagram of proposed algorithm is show below.

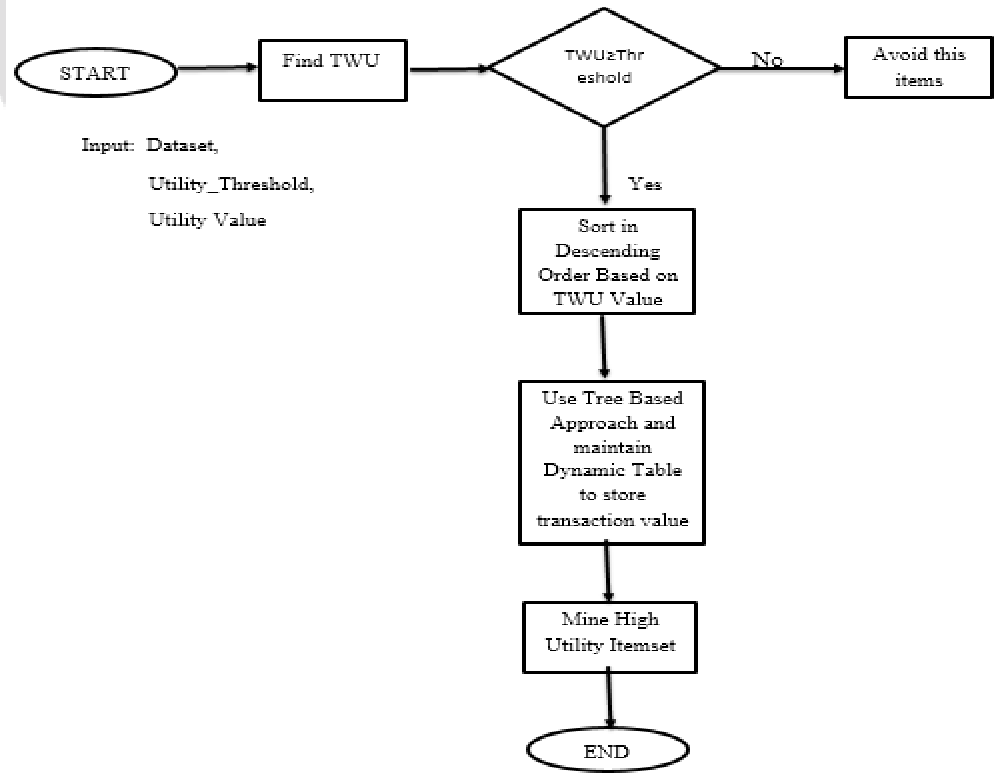

START
Find TWU
TWU≥Threshold
No
Avoid this items
Input: Dataset, Utility_Threshold, Utility Value
Yes
Sort in Descending Order Based on TWU Value
Use Tree Based Approach and maintain Dynamic Table to store transaction value
Mine High Utility Itemset
END

Fig-1. Proposed System

Firstly it takes dataset, utility value and utility threshold as an input and find total weighted utility (TWU) of each item, and checks it with the user threshold. If it is smaller than user threshold, then this items will be ignored. After checking it stores all the items in descending order according to their TWU. Now after that it use tree structure to mine high utility itemset.

ALGORITHM

Step 1: Scan Database and Collect TWU and TU of each items.

Step 2: If TWU (item) < min_utility discard that item, Sort remaining items in descending order

Based on TWU.

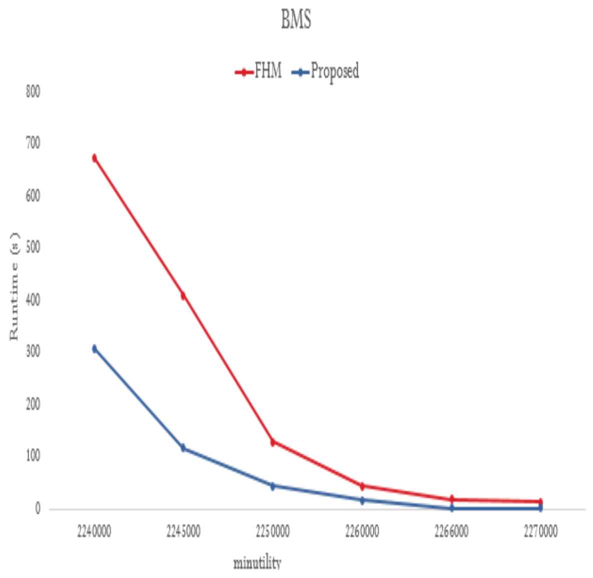
Step 3: Scan Database second time and insert each transaction into tree based on their descending order and maintain each node structure.

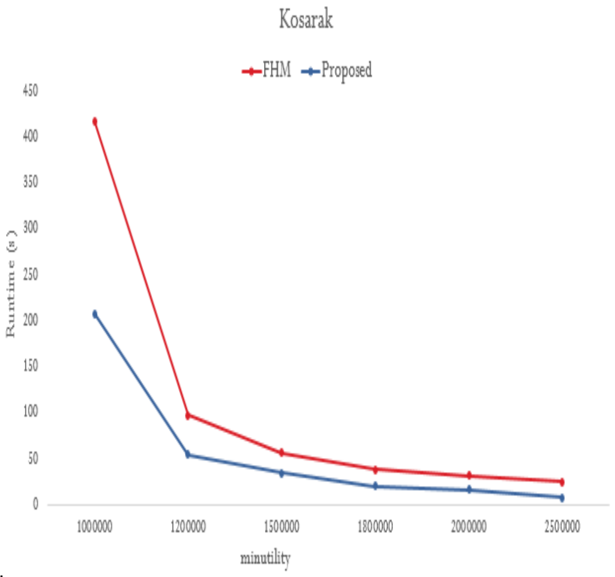
Step 4: Mining Tree to get high utility itemset, to mine match each item with other item in each path with their common transaction and sum their utility count. If utility_count ≥ min_utility then itemset will be consider as output.

4. EXPERIMENTAL STUDY

We performed experiments to assess the performance of the proposed algorithm. Experiments were performed on a computer with a third generation 64 bit core i5 processor running on Windows 7 Os and having 4 GB of free RAM. We compared the performance of proposed algorithm with the FHM algorithm for high utility itemset mining. All memory measurements were done using the Java API. Experiments were carried on four real-life dataset having different characteristics. The BMS dataset contains 59,601 transactions with 497 distinct items and an average transaction length of 4.85 items. The Kosarak dataset contains 9, 90,000 transactions with 41,270 distinct items and having an average length of 8.09 items. The Mushroom dataset contains 8,124 transactions with 119 distinct items having an average length of 23.0 items. The Foodmart dataset contains 4,141 transactions with 1559 distinct items having an average length of 4.4 items.

Execution Time. We first run the FHM and Proposed algorithms on each dataset while decreasing the minutil threshold until algorithms became too long to execute, run out of memory or a clear winner observed. For each dataset we recorded the execution time, the total high utility itemset count and total memory overhead. The comparison of execution times is shown in Fig 2. For BMS, Kosarak, Mushroom and Foodmart database proposed algorithm was respectively up to 5.1 times faster, 2.9 times faster, 2.1 times faster, and 3 times faster than FHM.





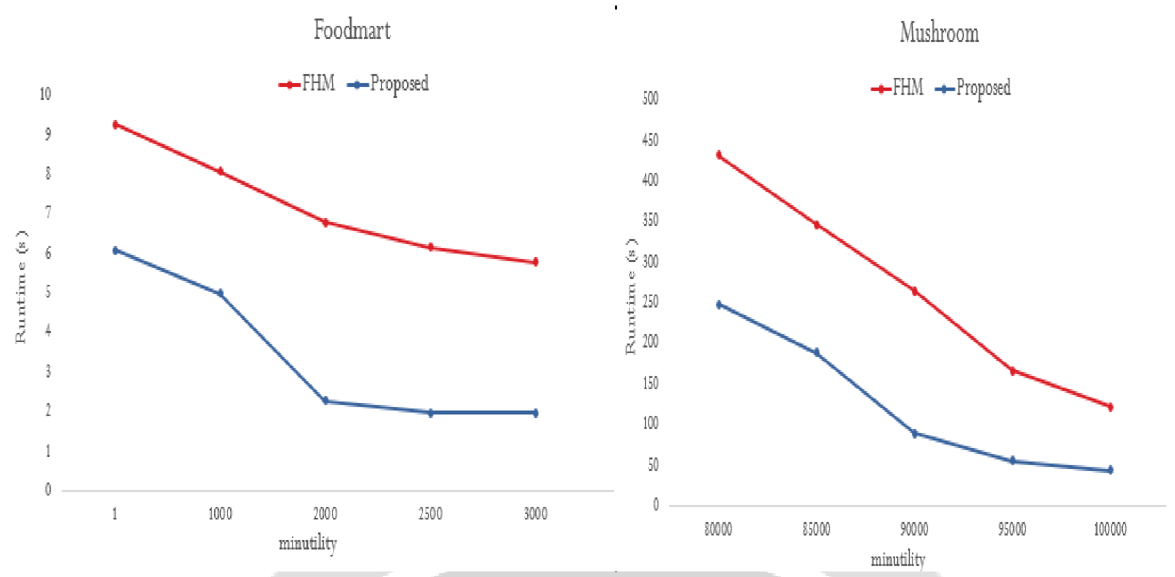


Fig 2: Execution Time Analysis for FHM and Proposed Algorithm

Memory Overhead. We also studied the memory overhead of using the tree structure. We found that the BMS, Kosarak, Foodmart, and Mushroom datasets, the memory usage of our proposed algorithm was respectively 76 MB, 186 MB, 89 MB, and 76 MB. The detailed comparison is shown in Table 4.

Dataset	FHM	Proposed
BMS	610	76
Kosarak	351	186
Foodmart	230	89
Mushroom	219	76

Table 4: Memory Comparison on Different dataset (MB)

5. CONCLUSIONS

The Primary focus is to enable high utility mining that provides itemset with interestingness factor which is very useful in real time application right now. In this paper, a new tree structure has been proposed. The helps keep related mining information such that execution time can be reduced. Tree Structure is little like FP-Growth tree but it stored the transaction id and its utility value in a different array structure same time when tree is generate. We also proposed an algorithm to effectively mine high utility itemset from the proposed tree structure. High utility item sets can be derived effectively and efficiently from tree structure. Experimental result shows that our proposed algorithm performs faster than FHM algorithm and it occupies less memory than FHM.

In this paper we assume that database is static, but in real life data will be inserted into or deleted from a database. In future we will attempt to handle incremental database where transactions are inserted, deleted or modified.

6. REFERENCES

[1] Vincent S. Tseng, Bai-En Shie, Cheng Wu, philip S. Yu Efficient Algorithms For Mining High Utility Itemset from Transactional Databases, IEEE transactions on knowledge and data engineering (Vol 25, no. 8), Aug 2013, PP1772-1786.

[2] Philippe Viger, Cheng Wu, Souleymane Zida, Vincent S. Tseng- FHM: Faster High Utiltiy Mining Itemset mining Using Estimated Utility Co-occurrence purning., Springer International, Switzerland 2014, PP 83-92.

[3] Shankar S, Dr. Pursothaman T, Jayanthi S-Novel Algorithm for mining High Utility Itemsets, International Conference on Computing, Communication and Networking, St. Thomas, 2008, PP 1-6.

[4] H J Hamilton, H Yao ,'Mining Itemset Utilities from Transactional Database', Elsevier 2006, PP-603-626.

[5] Yao H and Hamilton H j, 'Mining itemset utilities from transaction database', Data and Knowledge Engineering, 2006, PP-603-626.

[6] C.F.Ahmed, S. Khairuzzaman Tanbeer,Byeong-Soo Jeong, Young-Koo Lee-'Efficient Tree Structure for High Utility Pattern Mining in Incremental Databases.', IEEE Transactions on knowledge and data Engineering (Vol 21, No. 12), Dec 2009, PP 1708-1781.

[7] Agrawal R, Srikant R, 'Fast algorithms for mining association rules', Proceedings of 20th International conference on Very Large Databases, Santiago, Chile, 1994, PP. 487-499.

[8] Yung Liu, Woe heng Liao, and Alok Choudhary-'A two Phase algorithm for fast discovery of high utility Itemsets', Springer, Heidelberg, 2005, PP 689-695.

[9] Alva Erwin, Raj P. Gopalan, N.R. Achuthan-'Efficient Mining of High Utility Itemset from Large Datasets.' Springer, Berlin 2008, PP 554-561.

[10] Menghchi Liu, Junfeng Qu-'Mining High Utility Itemsets without Candidate Generation', CIKM, Maui. USA, Nov 2012, PP 55-63.

[11] Jia-Ling Koh and Shui-Feng Shieh- 'an efficient approach for maintaining association rules based on adjusting FP-tree structures', the 9th international conference on database system for advanced applications, PP 417-424.

[12] Yin, J. Zheng, Cao L,-'Uspan: An effective Algorithm for Mining High Utility Sequential Patterns', ACM SIG KDD 2012, and PP 660-668.

[13] Raymond Chand, Quing Yang, Shen- 'Mining High Utility Itemsets', Third International Conference on Data Mining, ICDM 2003, PP 19-26