International Journal of Advanced Technology in Engineering and Science
Vol. No.5, Issue No. 03, March 2017
www.ijates.com
ijates
ISSN 2348 - 7550

MINING HIGH UTILITY ITEMSET FROM LARGE DATABASE USING EFFICIENT ALGORITHM

Miss. Snehal D. Ambulkar[1], Dr. Prashant N. Chatur[2]

[1,2]*Department of Computer Science and Engineering,*

Government College of Engineering, Amravati, (India)

ABSTRACT

In the recent year, mining important pattern from a transaction database is a fundamental topic in data mining. Mining important and interesting pattern from a large database is important to many applications and helps to effectively improve a business strategy. While mining high utility itemset frequent itemset mining (FIM) does not consider quantity and profit of item it only considers the frequency of item appear in a transaction. To tackle this problem of frequent itemset mining, Utility itemset mining has been proposed which consider both profit and quantity of item from a transaction database. But utility itemset mining has a one problem, it does not support downward closure property. To solve this problem utility itemset mining uses the transaction weighted utility property (TWU). On the contrary, most of the algorithms generate plenty of high utility itemset which degrade the performance of mining process and require lots of execution time. To solve this problem, in this paper various strategy has been proposed to easily discovers high utility itemset from a large database and improve a mining process efficiently and effectively and requires less database scans.

Keywords: Data mining, Frequent itemset mining, High utility itemset mining, Transaction weighted utility, Utility mining.

I. INTRODUCTION

To increase the business profit it is very important to mine useful and hidden information from a large database. To increase the business profit data mining technique play an important role. Data mining technique uses the various algorithm and strategies to mine useful information which is very important to improve business strategy. Data mining technique uses the frequent itemset mining (FIM) [1] to mine the useful itemset from a large data but frequent itemset mining only consider the frequency of item. Because of that frequent itemset mining losses the useful information and it affect to increase business profit. To solve this problem utility itemset mining (UIM) has been proposed. Utility itemset mining consider both the external utility (importance, profit of item) and internal utility (Quantity) of item. Utility of an item is calculated by considering both the quantity and profit of item and hence utility itemset mining helps to improve a business profit in comparison to frequent itemset mining. Itemset is called high utility itemset (HUI) [2] if its utility value is greater than user specified threshold value. To set a proper threshold value user need to try and gauss multiple threshold value this is a time consuming process. While doing this utility itemset mining algorithm generates large amount of plenty of high utility itemset from a transaction database. It is not easy for user to choose a proper threshold

value. If threshold value is set small plenty of high utility itemset are generated and if it set large it may be the case that no high utility itemset are discover. To solve this problem Top-k high utility itemset mining is proposed [1,2]. Setting k value is easier than setting threshold value, because k value indicates the number of high utility itemset that user wants to find. Initially k value is taken from a user then raise gradually to effectively prone high utility itemset.

High utility itemset mining is use in various applications such as business analysis [3, 4], biomedicine, and streaming analysis [5]. Downward closure property helps to prone high utility itemset effectively from a large database. Frequent itemset mining support downward closure property but high utility itemset mining does not support downward closure property. It is not easy to find high utility itemset in the absence of downward closure property. To simplify this problem concept of transaction weighted utility (TWU) property is use.

This paper suggests a framework for finding high utility itemset effectively from a large database. Various strategies are proposed to raise utility threshold value effectively.

II. RELATED WORKS

In this section, discuss all research which is done on frequent itemset mining, high utility itemset mining, Top-k itemset mining, and Top-k High utility itemset mining and potential Top-k high utility mining which are as follows.

Frequent pattern mining is one of the data mining techniques which support the downward closure property to effectively find high utility pattern from transaction database. Frequent pattern mining only consider frequency of item in transaction. The Apriori algorithm [6] is used in frequent pattern mining, which scans the database several times as the maximum length of frequent pattern. FP-growth [7] data structure is a part of frequent pattern mining which scans the database twice and uses depth-first algorithm. Frequent pattern mining does not consider the quantity and profit that is, it does not consider the external and internal utility of item of item as we discuss above. Because of that it is unable to discover high utility itemset and does not satisfy the user requirement.

Many research works have been done for discovering high utility itemset from transaction database. High utility itemset mining algorithm overcomes the limitation of frequent itemset mining by considering both the quantity and profit of item as we discus above. Two-Phase [8], IHUP [5], UP-Growth [9], d^2HUP [10], HUI-Miner [11] has been proposed for mining high utility itemset from large database. The two phase algorithms consist of two phases for mining high utility itemset from a database. In first phase, they discover potential high utility itemset and in second phase calculate utility of itemset discover in first phase. In one phase algorithm they discover the high utility pattern in only one phase. d^2HUP, HUI-Miner is one phase algorithm for mining high utility itemset. d^2HUP is a novel algorithm convert the horizontal database into a tree based data structure called CAUL. HUI-Miner algorithm uses the vertical data representation technique and covert the data into utility list [11]. Utility mining does not support downward closure property to solve this problem transaction weighted utilization approach has been proposed. HUG-Miner and GHUI-Miner are also proposed to mine high utility itemset from a transaction database. HUG-Miner and GHUI-Miner algorithm is work on incremental database to reduce the database scans.

Top-k itemset mining discovers the Top-k itemset by a user specified value k by considering the utility of items in a transaction database. Many studies has been proposed for mining Top-k itemset [1], such as Top-K high utility itemset [2], Top-k frequent itemsets [12], Top-k frequent closed itemset [13,14], top-k closed sequential patterns [15], Top-k association rule [16], Top-k sequential rule [17]. The algorithms use various strategies for discovering Top-k patterns. Some algorithm uses FP-tree data structure for finding Top-k itemset from transactional database. In Top-k high utility itemset mining initially K value is taken from a user and the K value is increase or decrease according to requirements.

Many researchers have been done for discovering Top-k high utility pattern from transaction database. It considers both the internal utility and external utility of item. To support downward closure property, it uses the technique of transaction weighted utility (TWU). T-HUDS algorithm [18] has been proposed for mining Top-k high utility patterns from a data streams. Recently, TKU and TKO algorithm [1, 2] has been proposed for mining Top-k high utility itemset. TKU is two phase algorithm and it uses five strategies named PE, NU, MD, MC and SE to raise the border minimum utility threshold effectively to handle large number of high utility itemset. TKO algorithm is one phase algorithm and it uses novel strategies named RUC, RUZ, EPB to greatly improve the performance of algorithm and reduce the execution time.

III. MATHEMATICAL BACKGROUND

In this section, define the related mathematical definitions for discovering interesting and useful information from a transaction database. Table 1 is an example of transaction database and Table 2 contain the unit profit of each item.

TABLE: 1 TRANSACTIONAL DATABASE

Tid	Transaction
T1	(P,1)(R,1)(S,1)
T2	(P,2)(R,6)(X,2)(Z,5)
T3	(P,1) (Q,2)(R,1)(S,6) (X,1)(Y,5)
T4	(Q,4)(R,3)(S,3)(X,1)
T5	(Q,2)(R,2)(X,1)(Z,2)

TABLE: 2 PROFIT TABLE

Item	P	Q	R	S	X	Y	Z
Unit Profit	2	4	7	1	5	6	3

Let I be a set of itemset and I = {i_1, i_2, i_3, i_4…... i_n} and each and every item $i_b \in I$ where $1 \leq b \leq n$. D is a transaction database and D = {t_1, t_2, t_3, t_4…….t_m} such that each transaction $t_v \in D$ where $1 \leq v \leq m$. Each item $i_j \in t_v$ has a positive real value $q(i_j, t_v)$ it is call internal utility of item . Each item $i_j \in I$ is associated with positive real value $p(i_j, I)$ it is external utility of item.

Definition 1: Utility of an item i_b in a transaction t_v is denoted as $u(i_b, t_v)$ and defined as

$$u(i_b, t_v) = Q(i_j, t_v) \times P(i_j, I) \quad (1)$$

For example, u(P, T3) = 2 × 1 = 2 this is calculated from Table 1 and Table 2. Internal utility value of q(P,T3) =1 and external utility of p(P) = 2.

Definition 2: Utility of itemset C in transaction t_s is denoted as u(C, t_v) and defined as follows

$$u(C, t_v) = \sum_{ib \in C} u(i_b, t_v) \quad (2)$$

For example, u(PR, T2) = 2 × 2 + 6 × 7 = 46, u(PR, T1) = 1×2 + 1× 7 = 2 + 7 = 9, u(PR, T3) = 1×2 + 1× 7 = 2 + 7 = 9.

Definition 3: utility of itemset C in database D is denoted as u(C) and defined as follows.

$$u(C) = \sum_{C \subseteq t_v \wedge t_v \in D} u(C, t_v) \quad (3)$$

For example, u(PR) = u(PR, T1) + u(PR, T2) + u(PR, T3) = 9 + 46 + 9 = 64.

Definition 4: The transaction utility of a transaction t_v is denoted as TU (t_v) and defined as follows

$$TU(t_v) = \sum_{i_b \in t_v} u(i_b, t_v) \quad (4)$$

For example, TU(T1) = u(P,T1) + u(R,T1) + u(S,T1) = 2 + 7 + 1 = 10. Transaction utility of transaction is shown in Table

TABLE 3: TRANSACTION UTILITY

Tid	T1	T2	T3	T4	T5
TU	10	71	58	45	33

Definition 5: Total utility of a database D is denoted as TotalD and defined as

$$TotalD = \sum_{t_v \in D} TU(t_v) \quad (5)$$

For example, from Table 1 TotalD = TU(T1) + TU(T2) + TU(T3) + TU(T4) + TU(T5) = 10 + 71 + 58 + 45 + 33 = 217.

Definition 6: Relative utility of itemset C in a database D is denoted as Rtu and defined as follows

$$Rut(C) = u(C) / TotalD \quad (6)$$

For example, utility of itemset u(PR) = 64 and total utility of database is 217 from this relative utility of itemset is 29%.

Definition 7: The transaction weighted utility (TWU) of an itemset C is sum of transaction utility of all transaction that belong to C and denoted as TWU(C) and defined as follows

$$TWU(C) = \sum_{C \subseteq t_v \wedge t_v \in D} TU(t_v) \quad (7)$$

For example, Transaction weighted utility of P is TWU(P) = TU (T1) + TU (T2) + TU (T3) = 10 + 71 + 58 = 139. Transaction weighted utility of itemset is shown in Table 4.

TABLE 4: TWU OF ITEMSETS

Item	P	Q	R	S	X	Y	Z
TWU	139	136	217	113	207	58	104

Definition 8: An itemset is called high utility itemset if Rtu(C) is greater than user define threshold min_threshold i.e. Rtu(C) ≥ min_threshold. Otherwise itemset is called low utility itemset.

For Example, if min_threshold is set to 40 , the set of high utility itemset from table 1 are HUI = {({PQY},40),({PQRY},47),({PRXZ},71),({R},42),({PR},46),({PRX},51)}.

Definition 9: An itemset C from a transaction database is called high transaction weighted utilization itemset if TWU(C) ≥ min_threshold.

Definition 10: An itemset C is called Top-k high utility itemset in D if there are not more than k itemsets whose utility is larger than u(C) in $f_{HUI}(D,0)$.

Definition 11: The Minimum utility of an item denoted as MIU(I) defined as follows

$$MIU(I) = Min\{u(I, t_v) \mid t_v \in D \text{ and } v \in g(I)\}. \quad (8)$$

Table 5 shows the minimum utility of item. The minimum utility of item P is miu({P}) = min{u({P},T1), u({P},T2), u({P},T3)}= min{2,4,2} = 2. The minimum utility of itemset {PR} is MIU({PR}) = [MIU({P}) + MIU({R})] × SC ({PR}) = (2 + 7) × 3 = 27.

TABLE 5: ITEMS AND THEIR MIUS

Item	P	Q	R	S	X	Y	Z
Miu	2	8	7	1	5	30	6

Definition 12: The minimum utility of itemset C = {Y_1, Y_2, Y_3,..., Y_m} is denoted as MIU(C) and defined as follows

$$MIU(C)=\sum_{i=1}^{m} MIU(l_i) \times SC(C) \quad (9)$$

Definition 13: The Maximum utility of an item is denoted as MAU(I) and defined as follows

$$MAU(I) = Max\{u(I, t_v) \mid t_v \in D \text{ and } v \in g(I)\} \quad (10)$$

Table 6 shows the maximum utility of item. The maximum utility of item P is mau({P}) = max{u({P},T1), u({P},T2), u({P},T3)} = max{2, 4, 2 } = 4.

TABLE 6: ITEMS AND THEIR MAUS

Item	P	Q	R	S	X	Y	Z
Mau	4	16	42	6	10	30	15

Definition 14: The Maximum utility of itemset C = {Y_1, Y_2, Y_3,..., Y_m} is denoted as MAU(C) and defined as follows

$$MAU(C) = \sum_{i=1}^{m} MAU(I_i) \times SC(C) \quad (11)$$

Definition 15: The itemset is called potential Top-k high utility itemset if its TWU value and MAU are more than $min\_threshold_{border}$.

Definition 16: Remaining utility of an itemset C in a transaction t_v is denoted as RU(C, t_v) and defined as follows

$$RU(C, t_v) = \sum_{I_i \in t_v/C} u(I_i, t_v) \quad (12)$$

Definition 17: Remaining utility of an itemset in a database D is denoted as RU(C) and defined as follows

$$RU(C) = \sum_{I_i \in D} RU(C, t_v) \quad (13)$$

For example, remaining utility of itemset {P} in transaction T2 is RU(P, T2) = u(R,T2) + u(X,T2) + u(Z,T2)= 42 + 10 + 15 = 67. The remaining utility of itemset {P} in a database D is RU({P}) = RU({P},T1) + RU({P},T2) + RU({P},T3) = 8 + 67 + 56 = 131. The remaining utility of {QR} is RU({QR}) = RU({QR},T4) + RU({QR},T5) = 8 + 11 = 19.

Definition 18: Utility-list of an itemset C is denoted as UL(C). Utility list is a one type of list containing |g(C)| tuples which is in the form <Tid, iutil, rutil> utility list of itemset C is shown in Fig. 4.

IV. PROPOSED METHOD

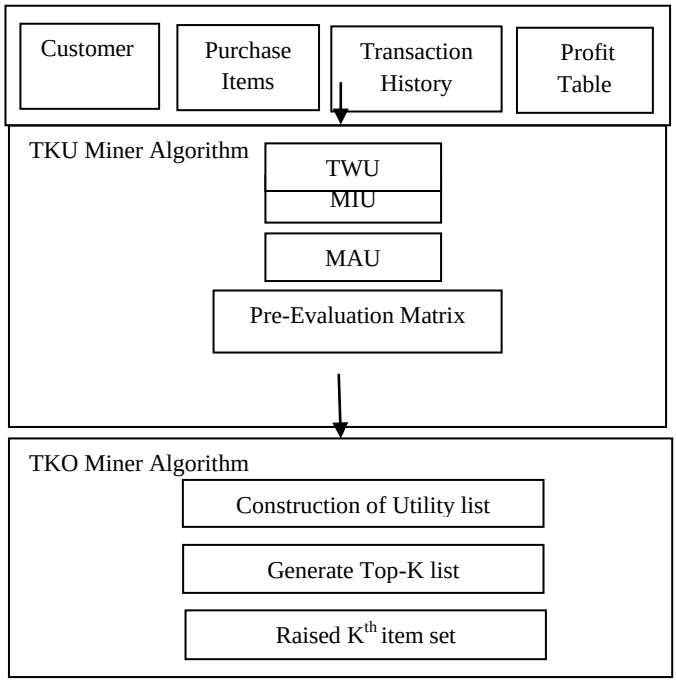

FIG. 1: SYSTEM ARCHITECTURE

The above fig. 1 shows the system architecture for mining high utility item from a transaction database. In the first step the customer purchase history and unit profit of each item is passing to the transaction database. Then in the next step the transaction database is passing to the TKU Miner Algorithm. TKU miner algorithm is a two phase mining algorithm. In TKU Miner Algorithm first calculate the transaction weighted utility (TWU) by using the definition 7. By using TWU the utility mining support downward closure property. After the calculation of TWU, arrange the item in decreasing order according to their transaction weighted utility as shown in Table 7. And then construct the UP-Tree. UP-Tree is constructed by scanning the database twice. The next step is MIU and MAU calculation. This is calculated by using the definition 11, definition 12, definition 13 and definition 14. Construction of UP-Tree is shown in Fig 3.

TABLE 7: ARRANGE TWU IN DESCENDING ORDER

Tid	TWU
R	217
X	207
P	139
Q	136
S	113
Z	104
Y	58

The last step in TKU miner Algorithm is the construction of pre-evaluation matrix. Each entry in pre-evaluation matrix is in the form PEM[x][y], where x, y ∈ I. When the transaction $T_P = \{a_1,a_2,a_3,\ldots\ldots,a_n\}$ ($a_i \in I$, $1< i \leq n$) is taken from the database in first scans. Then the utility of $\{a_1\} \cup \{a_i\}$ ($1< i \leq n$) in T_p is inserted into the corresponding entry of PEM[a_1][a_i] in pre-evaluation matrix. After scanning all the transactions of PEM, if the

k-th highest value in PEM is greater than the $min\_threshold_{border}$ then $min\_threshold_{border}$ is raised to k-th highest value in PEM. The Pre-evaluation matrix is shown in Fig.2.

Item	Q	R	S	X	Y	Z
P	10	64	11	7	32	19
Q		59	19	21	0	14
R			0	0	0	0
S				0	0	0
X					0	0
Y						0
Z						

FIG.2. PRE-EVALUATION MATRIX (PEM)

The next algorithm applies is a TKO Miner Algorithm which is a one phase algorithm. In this algorithm there is a construction of Utility list. Utility list is constructed to effectively find high utility itemset and to reduce a database scans.

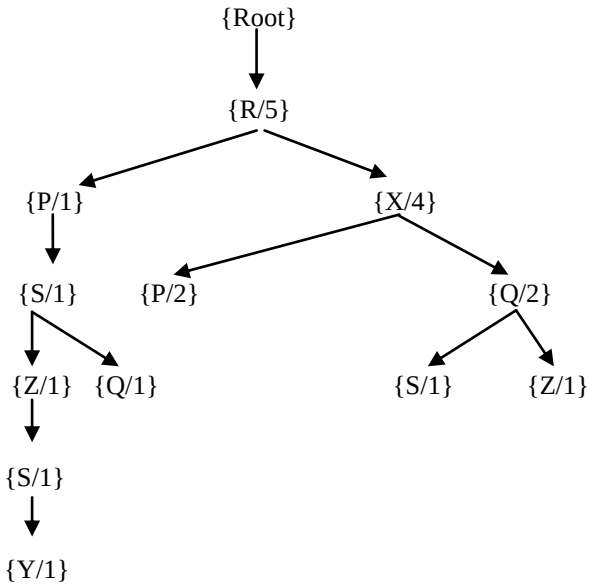

FIG.3. UP TREE

The Fig. 4 shows utility list structure of transactional database. The transaction database is taken from Table 1. The utility list structure contains the tuple T_{id}, iutil, rutil. T_{id} represents the identifier of transaction T_p, iutil represents the utility of item X in transaction T_p and rutil represents remaining utility. The remaining utility is calculated by using the definition 16, definition 17

Y		
3	5	28

Z		
2	5	56
5	2	27

S		
1	1	9
3	6	22
4	3	42
Q		
3	2	14
4	4	26
5	2	19

P		
1	1	7
2	2	52
3	1	12

X		
2	2	42
3	1	7
4	1	21
5	1	14

R		
1	1	O
2	6	0
3	1	0
4	3	0
5	2	0

FIG 4: UTILITY LIST

V. CONCLUSIONS

In this paper, we studied the various data mining algorithm for mining important and relevant information from a transactional database. The algorithm such as TKU and TKO are use to mine top-k high utility itemsets from a transaction database. TKU miner algorithm and TKO miner algorithm is novel algorithm for mining potential Top-k high utility itemset without any need to set threshold value. The TKU is a two phase mining algorithm and TKO is one phase mining algorithm. The TKU algorithm uses various strategies to effectively raise the border minimum threshold. To improve the performance of system TKO algorithm uses the strategies such as RUC, RUZ and EPB. The proposed algorithm uses the UP-Tree data structure to effectively prune the potential high utility itemset by scanning the database twice. In the first scan it calculate the TWU and in second scan it rearrange the TWU in descending order and then insert into the UP-Tree The proposed algorithm reduces database scans, required less memory consumption on incremental database and required less computational time. To improve the performance of system Utility-List structure is use which scans the database twice.

REFERENCES

[1] Serin Lee, Jong Soo Park, Top-K High Utility Itemset Mining Based On Utility List Structures, IEEE International Conference on Data Mining (ICDM), Maebashi, pp. 101 - 108, 2016.

[2] Vincent S. Tseng, Cheng-wei Wu, Philippe Fournier-Viger and Philip S. Yu, Efficient Algorithms For Mining Top-K High Utility Itemsets, In IEEE Transactions on Knowledge and Data Engineering, vol.28 no.1,2015.

[3] Y. Liu, W. Liao, and A. Choudhary, A Fast High Utility Itemsets Mining Algorithm, Utility-Based Data Mining Workshop, pp. 90-99, 2005.

[4] H. Ryang, U Yun and K. Ryu, Discovering High Utility Itemsets with Multiple Minimum Supports, Intelligent Data Analysis, Vol. 18(6), pp. 1027-1047, 2014.

[5] C. Ahmed, S. Tanbeer, B. Jeong and Y. Lee, Efficient Tree Structures for High-utility Pattern Mining in Incremental Databases, IEEE Transactions on Knowledge and Data Engineering, Vol. 21(12), pp. 1708-1721, 2009.

[6] R. Agrawal and R. Srikant, Fast Algorithms For Mining Association Rules, 20th International Conference on Very Large Data Bases, Santiago, Vol. 1215, pp. 487-499, 1994.

[7] J. Han, J. Pei and Y. Yin, Mining Frequent Patterns without Candidate Generation, ACM SIGMOD international Conference on Management of Data, pp. 1-12, 2000.

[8] Y. Liu, W. Liao, and A. Choudhary, A Two-phase algorithm for fast discovery of high utility itemsets, 9th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, Vol. 3518, pp. 689-695, 2005.

[9] V. S. Tseng, C. Wu, B. Shie, and P. S. Yu, UP-Growth: An Efficient Algorithm for High Utility Itemset Mining, ACM SIGKDD international Conference on Knowledge Discovery and Data Mining, pp. 253–262, 2010.

[10] J. Liu, K. Wang and B. Fung, Direct Discovery of High Utility Itemsets without Candidate Generation, IEEE international Conference on Data Mining, pp. 984-989 , 2012.

[11] M. Liu and J. Qu, Mining High Utility Itemsets without Candidate Generation, ACM international Conference on Information and Knowledge Management, pp. 55-64, 2012.

[12] K. Chuang, J. Huang and M. Chen, Mining Top-K Frequent Patterns in the Presence of the Memory Constraint, The VLDB Journal, Vol. 17, pp. 1321-1344, 2008.

[13] J. Han, J. Wang, Y. Lu and P. Tzvetkov, Mining Top-K Frequent Closed Patterns without Minimum Support, International Conference on Data Mining, pp. 211-218, 2002.

[14] J. Wang and J. Han, TFP: An Efficient Algorithm for Mining Top-K Frequent Closed Itemsets, IEEE Transactions on Knowledge and Data Engineering, Vol. 17(5), pp. 652-664, 2005

[15] P. Tzvetkov, X. Yan and J. Han, TSP: Mining Top-K Closed Sequential Patterns, Knowledge and Information System, Vol. 7(4), pp. 438-457, 2005.

[16] P. Fournier-Viger, C. Wu, V. S. Tseng, Mining Top-K Association Rules, International Conference on Canadian conference on Advances in Artificial Intelligence, pp. 61–73, 2012.

[17] P. Fournier-Viger, V. S Tseng, Mining Top-K Sequential Rules, International Conference on Advanced Data Mining and Applications, pp. 180-194, 2011.