International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-5, January 2020

An Innovative Procedure for Efficient Mining of Closed Top-K High Utility Itemsets

J. Wisely Joe, S. P. Syed Ibrahim

Abstract: ***High utilization itemset (HUI) mining is the fastest growing ground in association finding between the items. It is a process of finding the itemsets with higher utility values which participates in profitable decision making. The generated HUIs reflect the frequency, importance, profit or the utility of the items present in the database. Proper minimum threshold setting is very difficult for the end users without the knowledge of the data present in the database. Minimum user threshold extracts more number of candidate sets . Higher user threshold gives less number of candidate sets and very few high utility itemsets . In both the cases, the process is inefficient. Some algorithms produce more number of candidate itemsets as HUIs. The set of HUIs may degrade the performance of the candidate set mining by increasing the storage and time when the database has very large number of transactions. The number of candidate itemsets involved in the generation of HUIs may also slow down the entire process. The proposed novel strategy for tapping top-k closed high utility itemsets out of the set of candidate sets addresses these issues . The user defined integer k is the needed count of HUIs to be extracted out of the quality itemsets. The algorithm does not require the user to set the minimum utilization threshold .The closure property is merged with the pruning process and improves the productivity. The results transparently show that the k-closed high utility itemsets generated using this algorithm are productive, profitable and very concise when compared with existing approaches.***

Keywords : Association , utility mining, utility pattern tree, high utilization itemset, closed property

I. INTRODUCTION

Association mining algorithms proposed in [1],[2],[7] generally analyze set of transactions present in the database to find the relation between the items present in the transactions using item's frequency. Frequent pattern growth algorithms given in [5], [19] also do association mining but without candidate itemsets generation. FP growth algorithm introduces a compressed and efficient non linear data structure for keeping and representing the transactions. There are many algorithms used to obtain weighted frequent itemsets by considering weight or profit and the support count of items [19]. Weight is an integer assigned to every item in the database in accordance with successful decision. The weights are mainly assigned to items based on their profitability. In all the algorithms seen so far , we use a threshold value which is used to filter successful rules or patterns or weighted rules. Many of the association rules or patterns generated using the above algorithms may not be productive. Some rare patterns which are not listed by the algorithms may be more productive and profitable. To resolve these issues utility itemset (HUIM) mining [3][6][9][16] algorithms have been given. HUIM algorithms consider purchase quantity of items in every transaction during it's process. In pruning phase, HUIM separates the itemsets with utility more than the specified minimum utility threshold. Threshold is set based on frequency, profit, weight, importance, quantity and cost of itemsets by the user.

HUIM algorithms proposed in the papers [3],[6],[9],[10],[12],[13],[14] resolve different issues in HUIM. The major challenge in HUI mining algorithm is the algorithm's performance efficiency when it handles on huge dataset. Huge number of non-productive candidate sets are generated by HUIM algorithms and the generated itemsets degrade the performance of the algorithm by increasing running time and memory requirement. High Utility Itemset Miner algorithm[18], and FHM[20] were proposed to mine HUIs in single phase without candidate generation. A list used in HUI- Miner algorithm stores the entire node and utility information. The above papers clearly talk about recent utility mining algorithms, performance and memory constraints of the same. Most of the HUIM algorithms generate huge number of HUIs and most of the generated HUIs are non-productive, and they are redundant. So it is very much necessary to find a method to extract only valuable itemsets as high utility itemsets in lesser time when compared with existing algorithms. Other challenge we face in HUI mining is setting of minimum threshold without analyzing the transactions. In real-time, choosing or setting proper minimum utility threshold is difficult for users [11][17]. The output size will vary either less or more only based on that threshold value. Selection of threshold value has high impact on the performance of mining process. If too many HUIs are selected, user may find it difficult to consolidate the findings. Very large number of HUIs may also reduce the performance of algorithm or consumes more memory. On the other side, if threshold set is very high, only few numbers of HUIs or no HUIs are found.

Manuscript published on January 30, 2020.
* Correspondence Author
J. Wisely Joe*, Associate Professor, Department of Computer Science And Engineering,, St. Peter's College of Engineering And Technology, Chennai, India. Email: wiselyjoe.j2013@vit.ac.in
Dr. S. P. Syed Ibrahim, Professor, Department of Computer Science And Engineering,, St. Peter's College of Engineering And Technology, Chennai, India. Email: syedibrahim.sp@vit.ac.in

To fix a correct value for threshold, users may try different threshold values by casual guessing, trial and error method, re-execute the algorithm many times until they are satisfied with number of productive HUIs obtained This process of setting threshold is a time-consuming process.

So it is very much required to reduce the difficulty of setting minimum utility threshold for pruning non-profitable itemsets. To extract only profitable high utility itemsets without setting minimum threshold, a solution is to define a procedure to clip top-k closed high utilization itemsets. Instead of setting a threshold value, user will be defining k value. The 'k' gives the number of HUIs to be generated in output and it is always an integer. This process is simpler than setting threshold because proper threshold setting is based on the properties of transactions, purchase behavior of the customer, season and based on the available products. Utility mining does not satisfy download closure property (DCP).So the approaches using DCP for mining frequent itemsets not be applied directly to prune the productive itemsets in this proposed work.

The work is summarized as below: A new algorithm named TKCUM is proposed and explained below for tapping the list of top-k closed potential high utilization itemsets (TKCHUIs) without setting minimum threshold. The structure used in this algorithm to store the transactions present in database is UP tree [18].It inherits some properties from UP growth algorithm schemed by Tseng et al [13].The generated Potential closed HUIs are stored in a max heap structure. Top-k closed potential HUIs (TKCHUIs) are deleted from the heap and given to the user. We can also use vertical representation of data using utility lists and find top-k closed potential HUIs. The minimum utility an itemset must possess is initially set to zero and raised using two different strategies a) raising threshold by MIU and b) raising the threshold level wise(LW). The threshold is here called as border threshold . Different kinds of test cases are verified on chess, retail and mushroom datasets and the running times are compared with existing algorithms UP-Growth[13], TKU[4], TKU-BASE[4], UP-OPTIMAL[4] and HUI Miner[18] for different threshold values. The proposed algorithm TKCUM, performance of algorithm and the conclusion are given in the following sections.

II. BACKGROUND

This work recommends a new scheme of clipping top-k closed potential high utilization itemsets from a transactional database is proposed. As the minimum threshold value is automatically raised based on k value, the time taken for setting threshold value is very much reduced. In stead of processing all the candidate sets, we apply closed property and lessening the count of formed candidate sets. Thus lessening the running time of the proposed algorithm and produce only the potential high utility itemsets. All required definitions are listed in detail.

Take $I=\{ I_1 ,I_2 ,\ldots I_n\}$ the set of products , a transactional database of size t denoted as $DB = \{ T_1, T_2, ..., T_t \}$ and t is number of transactions. For every transaction Tc an unique identifier called transaction identifier(TID) is assigned. A positive integer P(i) is connected with each item ($i \epsilon I$) , which is profit or weight of item i . The number is sometimes called as external utility of i . Every purchased item i in transaction T_c is accompanied with an absolute value util(i, T_c) which represents the number of units purchased of product i by the customer transaction T_c. For example, Table -I shows a transactional database with five customer transactions (T_1, T_2, ..., T_5). Table-II gives the external utilities of the items purchased by the customers. For the item's present in transaction T_5 the external utilities are 2,1,3 and 1 for items B,C,E and G. In our approach , number of closed high utility itemsets to be mined is set based on the user's preference not by of minimum utility threshold .

Table- I: Transactional Database (TDB)

TID	Transactions
T1	(A,1) (C,1) (D,1)
T2	(A,2) (C,6) (E,2) (G,5)
T3	(A,1) (B,2) (C,1) (D,6) (E,1) (F,5)
T4	(B,4) (C,3) (D,3) (E,1)
T5	(B,2) (C,2) (E,1) (G,2)

Table- II: Profit Table

A	B	C	D	E	F	G
5	2	1	2	3	1	1

Definition 1: Item utility in any transaction T_c , Util(i, T_c) is calculated as P(i) × Util(i, T_c) if $i \epsilon T_c$.

Definition 2: Itemset utility in any transaction T_c is represented as Utility(X, T_c). Utility(X, T_c) = $\sum_{i \epsilon x}$Util(i, T_c) if $X \subseteq T_c$

Definition 3: Itemset utility in a database , Utility(X) is given as

Utility(X)= $\sum_{Tc \epsilon s(x)}$ Utility(X, T_c), where S(X) is the set of purchases in which itemset X is present.

For instance, the item utility of D in T_3 is Util(D, T_3) = 6 × 2 = 12. The utility of the itemset {C, D} in T_3 is Utility({C,D}, T_3) = Util(C, T_3) + Util(D, T_3) = 1 × 1 + 6 × 2 = 13. The utility of the itemset {C,D} is Utility({C,D}) = Utility({C,D}, T_1) + Utility({C,D}, T_3) + Utility({C,D}, T_4) = ((1 × 1) + (1 × 2)) + ((1 × 1) + (6 × 2)) + ((3 × 1) + (3 × 2)) = 25.

Table- III: Transaction Utility (TU) of transactions

TID	T1	T2	T3	T4	T5
T U	8	2 7	3 0	2 0	1 1

Table- IV: Transaction Weighted Utility of 1-itemsets

A	B	C	D	E	F	G
65	61	96	58	88	30	38

Definition 4 : Utility of a user purchase T_c, present in database D is TU(T_c) and computed as the sum of item utilities present in T_c. Transaction utility of few purchases in Table 1 are TU(T_1) = 5 + 1 + 2 = 8, TU(T_4) = 8 + 3 + 6 + 3 = 20 , TU(T_2) = 27, TU(T_3) = 30 and TU(T_5) = 11. TU of all transactions in our example are recorded in Table 3.

Definition 5: Number of CPHUIs to be mined is k .It is a positive integer and set based on the system requirements by the user.

Definition 6: Transaction Weighted Utilization (twu)[20] of an itemset is the total of transaction utilities of the transactions holding that itemset. For example in Fig.1, twu (D) =TU(T_1) + TU(T_3) + TU(T_4) = 58 .The twu's of 1-itemsets are listed in Table-IV. If twu(Y) is more than the utilization threshold, the itemset Y is chosen as a high transaction weighted utilization itemset (HTWUI).

Definition 7 : If Y is not a HTWUI, none of the parent of X could be a HTWUI. This property is known as transaction weighted downward closure(TWDC) property.

Definition 8 : An itemset X is a high utilization itemset (HUI) if the total utility is superior to border threshold.

III. RELATED WORKS

A. High utility itemset mining

Plenty of quality surveys have made for choosing frequent itemsets with and without candidate generation .The very common algorithms are Apriori algorithm [1] and tree based approach FP-Growth [5] . But in frequent pattern mining [1],[5] the influence of the item in profit and utility of items in the transaction are not considered. There were few methods proposed for mining high utility itemsets from given database, like UMining [21], Two phase algorithm [15], IIDS [22] and IHUP algorithm [23]. The above mentioned algorithms were proposed for mining high utility itemsets .They have some performance issues. Though UMining proposed by Yao et al., is showing good performance, it cannot bring out all possible high utility itemsets present in the database. Liu et al., proposed the Two-Phase algorithm [7] for tapping HTWUIs in two phases. Level wise candidate sets are generated [8]in the phases and HTWUIs are mined using twu of itemsets. In second part of database ,high utility itemsets are found with their utilities from the HTWUIs generated in phase 1. But too many HUIs are generated and many of them are redundant and not productive. Unusual items or itemsets are neglected with IIDS algorithm to reduce the duplication and non-productivity in the process and this strategy was designed by Li et al.[6] which reduces the candidates generated rather than reducing the HUIs. But the number of scans of database are more in this algorithm. To avoid multiple scanning of database, a tree based procedure was proposed by Ahmed et al. [2], for separating HUIs. Entire information related to the database is maintained in an IHUP-Tree. Though no candidate itemsets are generated, this process gives more number of HTWUIs in first phase which increases the execution time .

B. Top-k high utility itemset mining

Top-k high utility itemset mining differs from other algorithms in minimum threshold setting. Without the clear knowledge of information present in the database, it is very difficult to set the threshold for pruning unpromising items.There are few high utility mining algorithms proposed without any threshold value[24]. Initial border threshold value is null .On execution, border threshold is incremented dynamically .We can use either UP-tree or utility list structure to store the database. For every candidate generated, transaction weighted utility,minimum itemset utility and maximum itemset utility are calculated. The top k itemsets which are having transaction weighted utility(twu) greater than the threshold will be obtained by this algorithm.

IV. PROPOSED METHODOLOGY

This paper deals with the problem of finding top-k closed high utility itemsets to mine all HUIs present in the dataset without any loss and redundancy. The detailed explanationof the proposed methodology is given in following sub topics with a real time simple example .

A. Improved mining of top-k high utility itemsets

A new improved algorithm named TKCUM(Top-K Closed high Utility itemset Mining) for listing top-k closed high utility itemsets without minimum utility threshold is proposed in this section. The approach explains how to mine top-k CHUIs and a strategy to raise the threshold automatically in a time interval. The algorithm avoids generating redundant HUIs and non profitable HUIs. This improves the performance of the algorithm both in execution time and memory consumption. The performance analysis is clearly given in section V.

B. The methodology

Our proposed algorithm TKCUM takes a dataset, an integer input parameter k and gives back k highest valued utilities. UP-Tree is the data structure used to store and maintain the transactions of the dataset. The algorithm TKCUM is an extension of UPGrowth algorithm proposed in [13] which has a simple utility pattern search procedure for mining high utility itemsets. The algorithm TKCUM's framework has three parts: (1) representing the input transactions in UP-Tree, (2) extraction of closed high utility itemsets (CHUIs) from the built Utility pattern Tree, and (3) finalize top-k CHUIs from the chosen CHUIs stored in a heap structure.

a. Construction of UP-Tree

In UP-Tree, every node N has given elements:
name of the item N ; number of occurrences of the item N ; utility of node ; it's parent node ; and a link which points to a list that contains nodes having same name and in different branches of the UP-Tree. A table is maintained to make the insertions easier in the UP-Tree with a node as header. Each entry in the table has name of the purchased item, an estimated profit and a pointer to the node with same named node which occurred last in the tree. The nodes having same item names are connected with links so that they can be traversed in less time by tracking the pointer of header table and nodes[19].

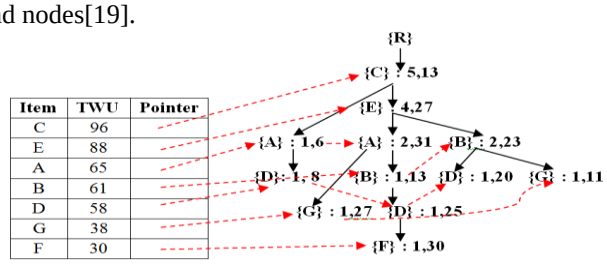

Fig. 1.Utility Pattern –Tree

The Utility Pattern tree is done with two reads of the transactional database. First, the utility of every transaction (TU) and item's transaction weighted utility (twu) are computed. Following that, items are ordered in non increasing order of their TWUs(Table-V).The modified transactions are now called as reorganized transactions. Second , the reorganized transactions are read and structured into the UP-Tree.

Table- V: Reorganized TDB

TID	Transactions
RT1	(C , 1) (A , 1) (D , 1)
RT2	(C , 6) (E , 2) (A , 2) (G , 5)
RT3	(C,1)(E,1)(A,1)(B,2)(D,6)(F,5)
RT4	(C , 3) (E , 1) (B , 4) (D , 3)
RT5	(C , 2) (E , 1) (B , 2) (G , 2)

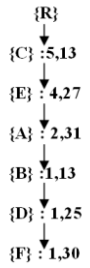

Fig. 2.Conditional Pattern Base -F

Discarding global node utilities (DGN) strategy proposed in paper [19] can not be applied in this approach as the initial minimum utility threshold is 0 . Directly after constructing reorganized transactions insert function is executed and entire dataset turned into a Utility Pattern tree. Now the UP-Tree is complete. Fig.1 shows the tree representation of the database in Table-I with threshold 0.

b. Generation of CHUIs from UP-Tree

Our proposed algorithm takes initial border threshold as 0 (denoted as bmu) and raised during the pruning process after capturing k number of itemsets with threshold higher than the bmu. Let us take the number of high utility itemsets to be mined is k = 3 and initial bmu is 0. Utilities of all 1-items present in database are listed in a list . The top-3 HUIs are {A:20,D:20,B:16} . As k is 3, bmu is lifted to the 3rd largest utility value 16, and no itemsets having more utility than 16 will be missed. After the increase in bmu, UP-Growth search procedure[13] is executed to generate TKCHUIs with bmu as min_util. From the generated UP-Tree , conditional pattern bases(CPBs) [13] for the items in the tree are constructed. The order of CPB construction is from leaf to root node[19].Conditional pattern base for the node f is given in Fig.2. Whenever we encounter a candidate X in UP-growth search procedure, we calculate MIU and MAU of that generated candidate itemset. Calculated miu,mau of 1-itemsets are given in Table-VI. For every candidate X generated from CPB,check the twu (X) with the twu of its root node in CPB. If both the values are equal then maintain only the parent node and continue the mining process . After this successful comparison , either parent or child will be selected. If for the selected node , both twu (X) , MAU(X) are greater than bmu, X is a closed high utility itemset(CHUI) and X is sent to output heap. Above steps for the sample dataset are calculated and the results are shown in Table-VII.As given in the above procedure only when the parent and child have different twu value, new node is created.The child node will become the parent otherwise.For the conditional pattern base for node F gives itemset EADF as Closed High Utility Itemset.

Algorithm : *TKCUM* Algorithm

Input : *An UP-Tree T, a H-table H*, number of HUIs to be mined *k* ,minimum utility threshold *bmu* and it is initially null.

Output : *TKCHUIs*, top-k closed high utility itemsets

```
1   for each entry Y in header table H do
2       Construct Y's CPB
3       Start constructing search space tree(SST) for
    the items from leaf element Y to top element in its
    CPB with Y as root
4     for every new item i inserted into SST,
5         Calculate MIU,MAU, twu of i
6         if (twu (i) ≥ bmu) and (MAU(i) ≥bmu)
7             if i is not the root of SST
8                 if twu (i)== twu (Parent(i)) #Closure
9                     Delete Parent(i)
10                    Parent(i) ← i
11                end if
12            end if
13            Output i & min(twu,MAU)
14            if MIU(i) ≥ bmu
15                ADD i to List Li
16            end if
17            if Li is full #Auto increment of
    threshold
18                  Bmu=min (Li)
19                Empty List Li
20            end if
21        end if
22    end for
23  Store the itemsets present in SST
```

Fig. 3.TKCUM Algorithm

Definition 9 : Smallest of all its utilities, Util(i, T_c) in the transactions is called item i's minimum item utility in dataset
Definition 10 : Item i's maximum item utility in dataset mau(i) is largest of all its utilities , Util(i, T_c) in the transactions.
Definition 11 : Minimum Item Utility of an itemset X={a_1, a_2,…, a_m} is formulated as MIU(X)= ∑(miu(a_i) x Support(X))
Definition 12 : The itemset X={ a_1, a_2,…, a_m } 's MAximum item Utility is calculated as MAU(X) =∑(mau(a_i) x Support(X))
Definition 13 : An itemset Y will be a closed high utility itemset(CHUI) if its twu (Y) and MAU(Y) are equal or greater than the bmu and it is present in the state space tree.

For the candidates having twu different from the twu of its root node , do maintain child node with parent and process from child.

Table- VI: miu and mau of 1-itemsets

Item	A	B	C	D	E	F	G
miu	5	4	1	2	3	5	2
mau	10	8	6	12	6	5	5

If MIU(X) is greater than border_min_sup ,MIU(X) is stored in a list of size k. Once the list is full , minimum of the elements present in the list will be set as new minimum utility threshold.To select the minimum of elements in the list easily, we can maintain the list as min heap structure.

c. Identifying top-k CHUIs from CHUIs

In our approach, heaps are used to store the CHUIs and to identify top-k closed high utility itemsets(TKCHUIs) from the chosen CHUIs. To avoid unnecessary comparisons , we store the CHUIs in decreasing order of twu.The entire list is stored in a max heap and first k elements are deleted from the heap .The deleted elements are the mined top-k closed high utility itemsets(TKCHUIs) .

The pseudo code is structured in Fig.3. Search Space Tree used in our procedure is clearly explained in reference papers [2][5].

Table- VII: Final result for the sample dataset

Border Minimum Threshold bmu =15								
Itemset	TWU	MAU	MIU	TWU >bmu	MAU> bmu	MIU> bmu	HUI	CHUI
F	30	5	5	Yes	No	--	--	--
DF	30	17	7	Yes	Yes	--	Yes	~~Yes~~
AF	--	--	--	--	--	--	--	--
EF	--	--	--	--	--	--	--	--
ADF	30	27	12	Yes	Yes	--	Yes	~~Yes~~
EDF	--	--	--	--	--	--	--	--
EAF	--	--	--	--	--	--	--	--
EADF	30	33	15	Yes	Yes	Yes	Yes	Yes

Table- VIII: Characteristics of datasets

Dataset	D	I	AL	ML	Type
Accidents	34083	469	33.8	51	Sparse
Chain store	1011129	46086	7.3	12	Sparse
Chess	3196	76	37	37	Dense
Food mart	21556	1559	4.4	7	Sparse
Mushroom	8124	120	23	23	Dense

V. RESULTS AND DISCUSSIONS

a. Results

The real datasets taken for our analysis are Chain store, Accident, Food mart, Chess and Mushroom . A simulating model was developed initially to customize the standard datasets .Some values are randomly assigned to make the entries complete. Parameters of the datasets considered for the process of analysis are total number of transactions D, number of unique items I, average length of the transactions AL, maximum length of the transactions ML and type description. Characteristics of every dataset is listed in Table-VIII. Experiments were done with our proposed algorithm top-k closed high utility itemset mining (TKCUM) and state-of-the art algorithms and the readings were recorded. The results are given in following Fig.4. All these executions were performed on an Intel Core i5 (2.5GHz) PC with 8 GB main memory. PC runs the Windows 10 operating systems. The code was written in python 3.7 and executed with the above real datasets. In which Chess and Mushroom are of dense datasets and others are of sparse datasets. Chain store has more number of transactions (1112949 transactions) and huge in size.

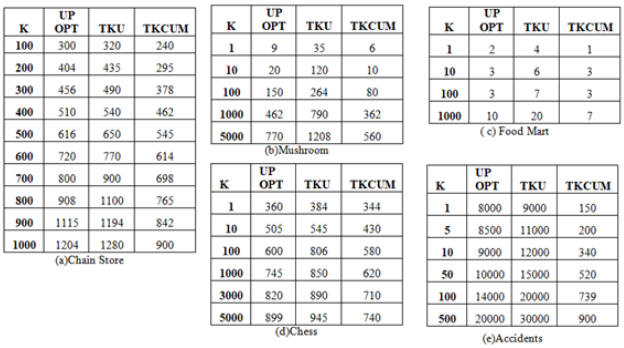

Fig. 4. Execution Time vs k

DB SIZE (1000s)	TKU(s)	UP-OPTIMAL(s)	TKCUM(s)
100	4000	3008	645
200	4200	5230	987
300	6100	6734	1020
400	8112	8970	2078
500	9034	9400	3015

Fig. 5. Execution Time vs Dataset Size

Every transaction in these datasets contains the items purchased and their purchased quantities. The performance of the proposed TKCUM algorithm was compared to the base strategies like Utility Pattern Growth algorithm, UP-Optimal algorithm and top-k high utility itemset mining(TKU) algorithm for different k values. The 'k' is the number of productive itemsets to be generated with highest utility values. The results are plotted in graphs for the different datasets and with different algorithms .The graphs are shown in Fig.6 .

b. Performance

The evaluation was done for the total time taken by each algorithm for a specific dataset for different k values. Recorded values are plotted as graphs. It is observed from the graph that the proposed strategy TKCUM provides good improvement on overall performance compared to the modern UP-optimal ,TKU algorithms. The proposed strategy TKCUM is 2 to 4 times quicker than the contemporary algorithms on the data sets except accident dataset. On accident dataset, proposed algorithm works too good and takes very lesser time than other algorithms.
Fig.5, Fig. 6.(f) indicates increasing the execution time and the size of database.

There is a hike in execution time when the size of dataset increases for all algorithms. But when we analyze the increase in time, TKCUM is 6 times faster than other algorithms on an average. Thus we proved that the proposed TKCUM algorithm can be efficiently used to minimize the running time for pruning the non-productive itemsets and to choose the most productive itemsets called TKCHUIs.

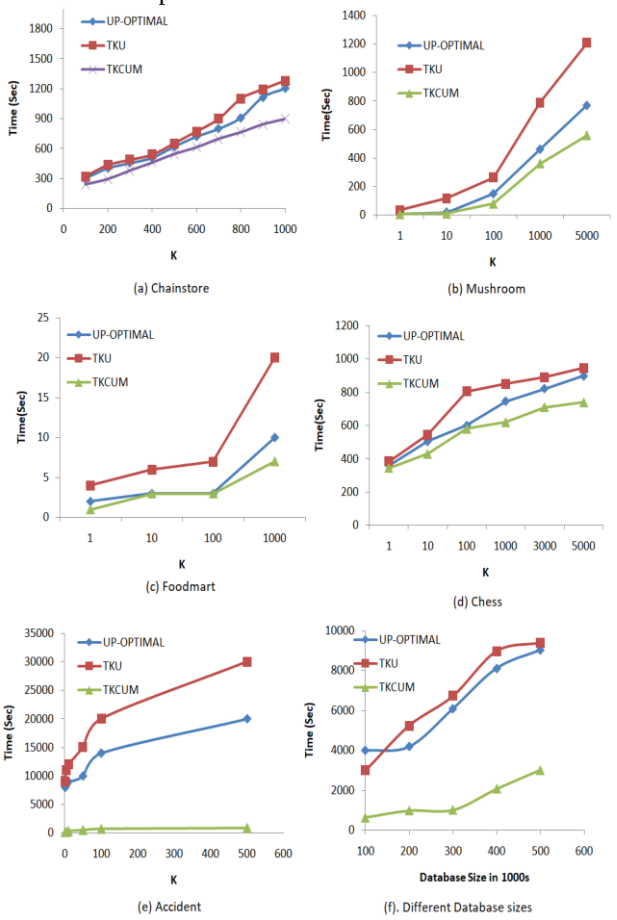

Fig. 6.Performance Analysis

VI. CONCLUSION

In the past, transaction weighted utility was only based on the downward closure property. But in proposed method , we used closed property with automated minimum utility threshold in pruning process to reduce the number of candidate set generated and processed. Also our approach reduces the difficulty in setting minimum threshold, by defining a k value. K indicates the number of itemsets having highest utility values. So no high utility itemsets are left unselected and no redundant itemsets are selected. The proposed new algorithm called TKCUM taps distinct top-k closed high utility itemsets from large databases. The MAU and MIU list structures are used to maintain all the needed information ready for the pruning process. Availability of information leads to a reduction in number of database scans. This drastically reduces the running time of algorithm for various datasets. Every itemset generated by our approach is very productive and profitable. All these are clearly explained and proved with different input sets and results. This TKCUM algorithm outperforms most of the updated versions of the algorithms in this stretch currently in use on every real world datasets.

REFERENCES

1. Agrawal, R., Imielinski, T., and Swami , " Mining association rules between sets of items in large databases". In Proc. of 1993 ACM SIGMOD International Conf on Management of Data, P. Buneman and S. Jajodia, Eds. Washington, 207-216.
2. Agrawal R. and Srikant R. ,"Fast algorithms for mining association rules" . In Proc. of the 20th International Conf. on , VLDB "94, pp. 487-499
3. A. Erwin, R. P. Gopalan, and N. R. Achuthan. " Efficient mining of high utility itemsets from large datasets". In Proc. of PAKDD 2008, LNAI 5012, pp. 554-561.
4. Tseng VS, Wu CW, F.Viger P, Yu PS (2016)" Efficient algorithms for mining Top-K high utility itemsets". IEEE Trans Knowl Data Eng 28(1):54–67
5. Jiawei Han, Jian Pei, and Yiwen Yin ."Mining Frequent Patterns without Candidate Generation"In Proc. of SIGMOD'2000 Paper ID: 196
6. C.-W. Lin, T.-P. Hong, and W.-H. Lu, "An effective tree structure for mining high utility itemsets," In Proc. of Expert Systems. Appl., vol. 38, no. 6, pp. 7419–7424, 2011.
7. M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I. Verkamo. "Finding interesting rules from large sets of discovered association rules". In Proc. of the 3rd International Conf. on Information and Knowledge Management, pages 401-408,Nov. 1994.
8. M.C. Tseng and W.Y. Lin, "Mining generalized association rules with multiple minimum supports". Proc. of International Conf. on Data Warehousing and Knowledge Discovery (2001) 11-20.
9. B. Le, H. Nguyen, T. A. Cao, and B. Vo, "A novel algorithm for mining high utility itemsets," in Proceedings of 1st Asian Conference Intelligent Information Database Syst., 2009, pp. 13–17.
10. Y. Liu, W. Liao, and A. Choudhary, "A fast high utility itemsets mining algorithm," in Proc. Utility-Based Data Mining Workshop, 2005, pp. 90–99.
11. Shichao, Z. & Jingli, Lu. Chengqi, Z. " A fuzzy logic based method to acquire user threshold of minimum-support for mining association rules", Information Sciences, 1-15, 2003.
12. H.-F. Li, H.-Y. Huang, Y.-C. Chen, Y.-J. Liu, and S.-Y. Lee, "Fast and memory efficient mining of high utility itemsets in data streams," in Proceedings of IEEE International Confeference on Data Mining, 2008, pp. 881–886.
13. Vincent S. Tseng, Cheng-Wei Wu, Bai-En Shie, and Philip S. Yu." UP-Growth: An Efficient Algorithm for High Utility Itemset Mining" , In Proceedings of KDD'10, July 25–28, 2010
14. R. Chan, Q. Yang, and Y. Shen, "Mining high utility itemsets," in Proceedings of IEEE International Confeference on Data Mining ., 2003, pp. 19–26.
15. T. M. Quang, S. Oyanagi, and K. Yamazaki, ExMiner: An Efficient Algorithm for Mining Top-K Frequent Patterns, ADMA 2006, LNAI 4093, pp. 436 – 447, 2006.
16. Souleymane Zida, Philippe Fournier-Viger, Jerry Chun-Wei Lin, Cheng-Wei Wu, Vincent S. Tseng, "EFIM: A Highly Efficient Algorithm for High-Utility Itemset Mining".
17. Bing Liu, Wynne Hsu and Yiming Ma "Mining Association Rules with Multiple Minimum Supports" ,KDD-99.
18. Mengchi Li and Junfeng Qu,"Mining High Utility Itemsets without Candidate Generation" , in the proceedings of CIKM'12, Maui, HI, USA.
19. Feng Tao, Fionn Murtagh, Mohsen Farid," Weighted Association Rule Mining using weighted support and significance framework",KDD'03
20. Philippe Fournier-Viger, Souleymane Zida, Cheng-Wei Wu, Vincent S. Tseng," FHM: Faster High-Utility Itemset Mining using Estimated Utility Co-occurrence Pruning", ISMIS' 14,Springer,LNAI ,pp 83-92
21. H. Yao, H. J. Hamilton, L. Geng, A unified framework for utility-based measures for mining itemsets. In Proc. of ACM SIGKDD 2nd Workshop on Utility-Based Data Mining, pp. 28-37, USA, Aug., 2006.
22. Y.-C. Li, J.-S. Yeh, and C.-C. Chang. Isolated items discarding strategy for discovering high utility itemsets. In Data & Knowledge Engineering, Vol. 64, Issue 1, pp. 198-217, Jan., 2008.
23. C. F. Ahmed, S. K. Tanbeer, B.-S. Jeong, and Y.-K. Lee. Efficient tree structures for high utility pattern mining in incremental databases. In IEEE Transactions on Knowledge and Data Engineering, Vol. 21, Issue 12, pp. 1708-1721, 2009.
24. Cheng Wei Wu, Bai-En Shie, Philip S. Yu, Vincent S. Tseng . Mining Top-K High Utility Itemsets

AUTHORS PROFILE

J. Wisely Joe She is a research scholar in the School of Computing Science & Engineering, VIT University, Chennai. She graduated in Information Technology from Madras University and completed Post Graduate in Computer Science and Engineering from Anna University. She is working as an associate professor in the department of computer science and engineering at ,St.Peter's College of Engineering and Technology, Chennai and her areas of interest includes Data mining, Big Data, Itemset Mining. She has published 5 papers in international journals.

Dr S P Syed Ibrahim He has completed his Bachelors in Engineering under Bharathidasan University, Masters and PhD in Engineering under Anna University. He has been a part of VIT as Professor for the past 6 years. He is a recipient of Best Faculty Award from VIT for the year 2016 and EMC academic alliance award for the year 2013. He has published more than 60 research papers in journals and conferences. He organized four short term courses on analytics for students from universities abroad and also organized DST sponsored national workshop on data science research.