IJDCST @Sep, Issue- V-2, I-6, SW-25
ISSN-2320-7884 (Online)
ISSN-2321-0257 (Print)

To Evaluate Performances of HUI-Miner Algorithm

[1] V.V.S.N.S.A.D.Bhavani, [2]Mr B.Venkateswarlu,
[1]Final M Tech Student, [2] Associate professor,
[1,2]Dept of Computer Science and Engineering ,
[1,2,3] Raghu Engineering college, Dakamarri, Vishakapatnam, A.P.

Abstract: Utility-based data mining is a new research area interested in all types of utility factors in data mining processes and targeted at incorporating utility considerations in both predictive and descriptive data mining tasks. High utility item set mining is a research area of utility-based descriptive data mining, aimed at finding item sets that contribute most to the total utility. A specialized form of high utility item set mining is utility-frequent item set mining, which – in addition to subjectively defined utility – also takes into account item set frequencies. This paper presents novel efficient algorithms UP-Growth and High Utility Item Set which finds all utility-frequent item sets within the given utility and support constraints threshold. And it is based on efficient methods for frequent item set mining. Experimental evaluation on datasets shows that, in contrast with High Utility Item Set, and also the performances are evaluated through the factors of time and space complexities.

Key Words: Candidate pruning, frequent item set, high utility item set, utility mining, data mining

I. INTRODUCTION

The rapid development of database techniques facilitates the storage and usage of massive data from business corporations, governments, and scientific organizations. How to obtain valuable information from various databases has received considerable attention, which results in the sharp rise of related research topics. Among the topics, the high utility item set mining problem is one of the most important, and it derives from the famous frequent item set mining problem. Mining frequent item sets is to identify the sets of items that appear frequently in transactions in a database. The frequency of an item set is measured with the support of the item set, i.e., the number of transactions containing the item set. If the support of an item set exceeds a user-specified minimum support threshold, the item set is considered as frequent. Most frequent item set mining algorithms employ the downward closure property of item sets [4]. That is, all supersets of an infrequent item set are infrequent, and all subsets of a frequent item set are frequent. The property provides the algorithms with a powerful pruning strategy. In the process of mining frequent item sets, once an infrequent item set is identified, the algorithms no longer check all supersets of the item set. For example, for a database with n items, after the algorithms identify an infrequent item set containing k items, there is no need to check all of its supersets, i.e., $2^{(n-k)-1}$ item sets.

Mining of frequent itemsets only takes the presence and absence of items into account. Other information about items is not considered, such as the independent utility of an item and the context utility of an item in a transaction. Typically, in a supermarket database, each item has a distinct price/profit, and each item in a transaction is associated with a distinct count which means the quantity of the item one bought. Consider the

database in Fig. 1. There are seven items in the utility table and seven transactions in the transaction table in the database. To calculate support, an algorithm only makes use of the information of the first two columns in the transaction table, the information of both the utility table and the other columns in the transaction table are discarded. However, an itemset with high support may have low utility, or vice versa. For example, the support and utility of itemset { bc } appearing in T1, T2, and T6 are 3 and 18 respectively, and those of itemset { de }

Item	a	b	c	d	e	f	g
Utility	1	2	1	5	4	3	1

(a) Utility table

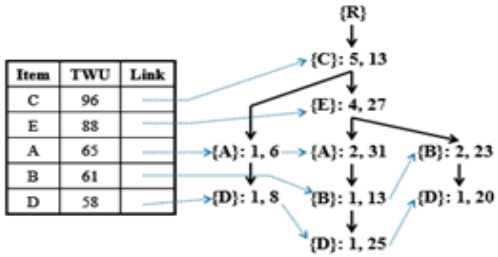
Tid	Transaction	Count
T1	{ b, c, d, g }	{ 1, 2, 1, 1 }
T2	{ a, b, c, d, e }	{ 4, 1, 3, 1, 1 }
T3	{ a, c, d }	{ 4, 2, 1 }
T4	{ c, e, f }	{ 2, 1, 1 }
T5	{ a, b, d, e }	{ 5, 2, 1, 2 }
T6	{ a, b, c, f }	{ 3, 4, 1, 2 }
T7	{ d, g }	{ 1, 5 }

(b) Transaction table

Figure 1: Database

II. EXISTING SYSTEM

In this section, we describe the details of UP-Growth for efficiently generating PHUIs from the global UP-Tree with two strategies, namely DLU (Discarding local unpromising items) and DLN (Decreasing local node utilities). Although strategies DGU and DGN can effectively reduce the number of candidates in phase I, they are applied during the construction of the global UP-Tree and cannot be applied during the construction of the local UP-Tree. The reason is that the individual items and their utilities are not maintained in the conditional pattern base. We cannot know the utility values of the unpromising items in the conditional pattern base. To overcome this problem, a naïve approach is to maintain the utilities of the items in the conditional pattern base. However, this approach may



Item	TWU	Link
C	96	
E	88	
A	65	
B	61	
D	58	

A UP-Tree by applying strategies DGU and DGN.

. Minimum item utility table

Item	A	B	C	D	E
Minimum item utility	5	4	1	2	3

Figure 1.1:UP-tree applying strategies

be impractical since it consumes lots of memory usages. Instead of maintaining exact utility values of the items in the conditional pattern base, we maintain a minimum item utility table , abbreviated as MIUT , to maintain the minimum item utility for all global promising items.

By the rationale of DGU strategy, in a conditional pattern tree, local unpromising items and their utilities can be discarded. Since the minimum item utility of a local unpromising item in a path is always equal to or less than its real utility in the path, we can also discard its minimum item utility from the paths of the conditional pattern tree without losing any PHUI. The purpose of DLU strategy is similar to DGU strategy, while DLU is applied during the second scan of the conditional pattern base. First, we scan conditional pattern base once to identify local promising items and unpromising items. Then, we scan conditional pattern base again to construct a local UP-Tree. When a path is retrieved, each unpromising item is removed from the path and its minimum item utility in this path is eliminated from the path utilities.

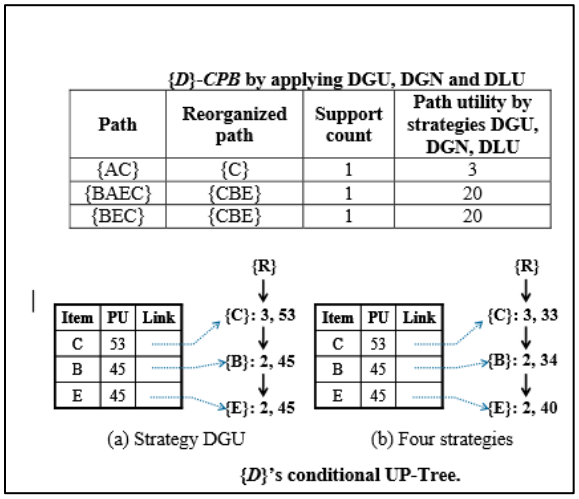


Figure 2.1: Pruning UP-Pattrens

The procedure of the UP-Growth is shown as follows:

Subroutine: UP-Growth (T_x, H_x, X)
Input: A UP-Tree T_x, a header table H_x for T_x and an itemset X.
Output: All PHUIs in T_x.
Procedure UP-Growth (T_x, H_x, X)

(1) **For each** entry a_i in H_x **do**
(2) Generate a PHUI $Y = X \bigcup a_i$;
(3) The estimate utility of Y is set as a_i's utility value in H_x;
(4) Construct Y's conditional pattern base Y-CPB;
(5) Put local promising items in Y-CPB into H_y
(6) Apply strategy DLU to reduce path utilities of the paths;
(7) Apply strategy DLN and insert paths into T_y;
(8) **If** $T_y \neq null$ **then** call **UP-Growth**(T_y, H_y, Y);
(9) **End for**

To prepare the data sets and give as a input to the UP-Growth algorithm so we prepare the data set for transactional table prepare a data sets.

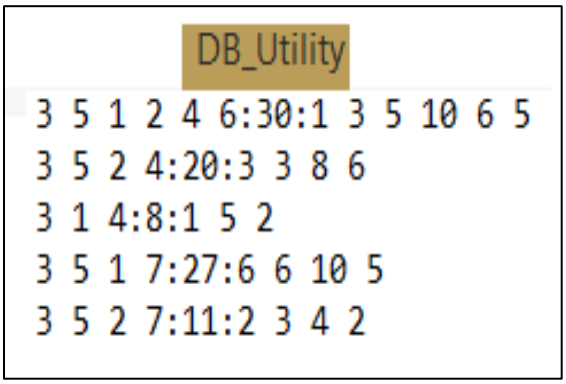


Figure 2.2: Data set for UP Growth Algorithm

And the HUI Miner generate the results

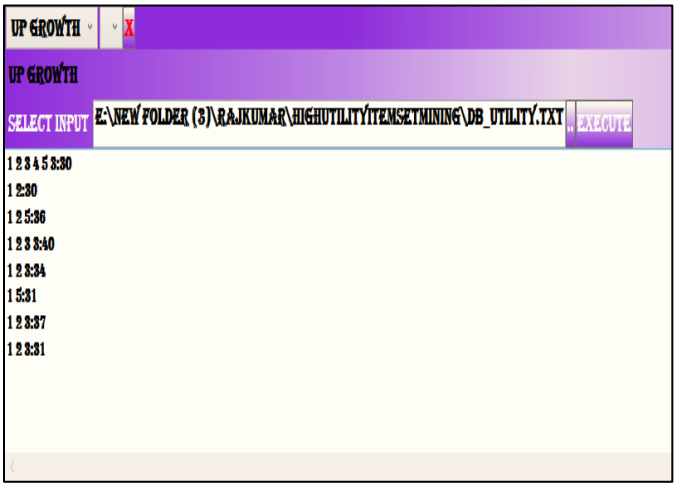


Figure 2.3: UP-Growth Results.

III. PROPOSED APPROACH

The goal of utility mining is to discover all the itemsets whose utility values are beyond a user specified threshold in a transaction database. We start with the definition of a set of terms that leads to the form al definition of utility mining problem. Although DGU and DGN strategies are efficiently reduce the number of candidates in Phase 1(i.e., global UP - Tree). But they cannot be applied during the construct ion of the local UP - Tree (Phase 2). Instead use, DLU strategy (Discarding local unpromising items) to discarding utilities of low utility items from path utilities of the paths and DLN strategy (Discarding local node utilities) to

discarding item utilities of descendant nodes during the local UP-Tree construction. Even though, still the algorithm facing some performance issues in phase-2. To overcome this, maximum transaction weight utilizations (MTWU) are computed from all the items and considering multiple of min_sup as a user specified threshold value as shown in algorithm. By this modification, performance will increase compare with existing UP-Tree construction also improves the performance of UP-growth algorithm. An improved utility pattern growth is abbreviated as High Utility Item miner algorithm.

Input: Transaction database D, user specified threshold.
Output: high utility itemsets.

Begin

1. Scan database of transactions $T_d \in D$
2. Determine transaction utility of T_d in D and TWU of itemset (X)
3. Compute min_sup (MTWU * user specified threshold)
4. If (TWU(X) ≤ min_sup) then Remove Items from transaction database
5. Else insert into header table H and to keep the items in the descending order.
6. Repeat step 4 & 5 until end of the D.
7. Insert T_d into global UP-Tree
8. Apply DGU and DGN strategies on global UP- tree
9. Re-construct the UP-Tree
10. **For** each item a_i in H do
11. Generate a PHUI Y= X U a_i
12. Estimate utility of Y is set as a_i's utility value in H
13. Put local promising items in Y-CPB into H
14. Apply strategy DLU to reduce path utilities of the paths
15. Apply strategy DLN and insert paths into T_d

Figure 3.1:HUI-Miner Algorithm

To prepare the data sets and give as a input to the High Utility Item miner algorithm. so we prepare the data set for transactional table prepare a data sets.

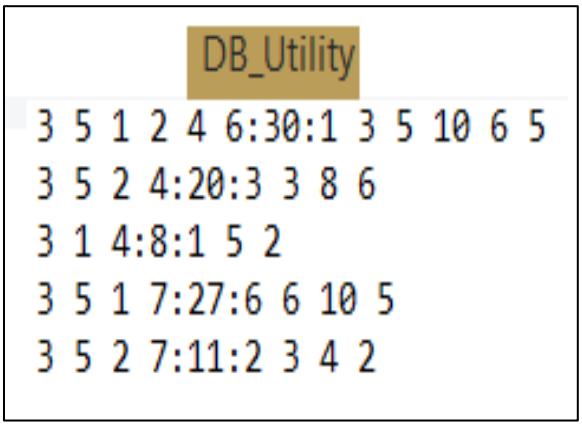


Figure3.2: Data set for HUI Algorithm

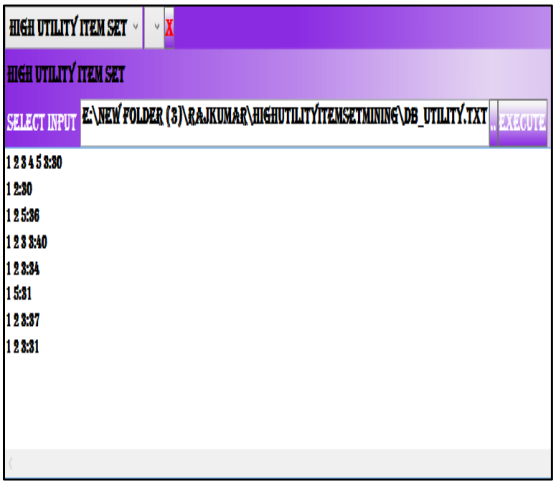


Figure3.3: HUI Miner Results.

IV. EXPERIMENTAL ANALAYSIS

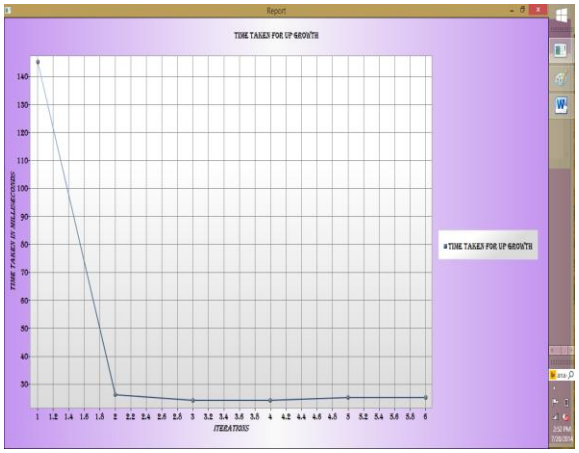


Figure 4.1:Time taken for UP-Growth

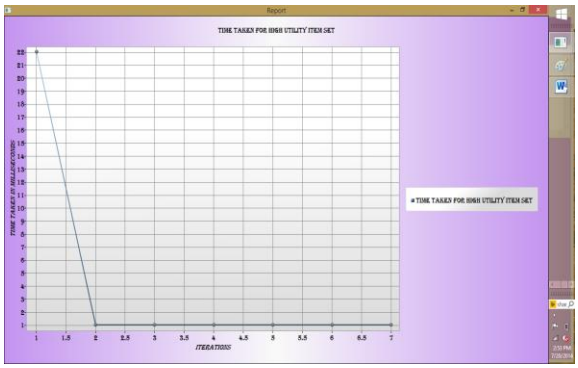


Figure 4.2:Time taken for HUI-Miner

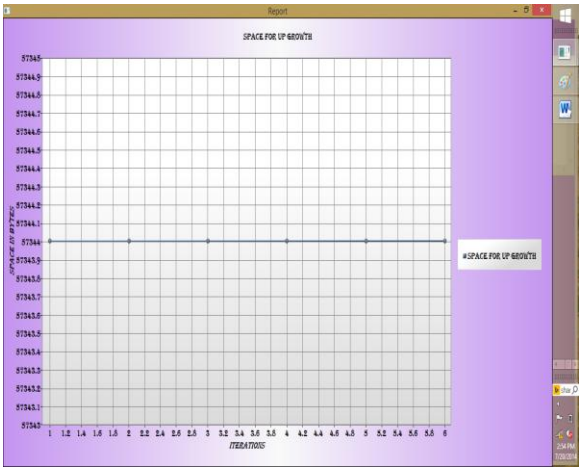


Figure 4.3:Space for UP-Growth

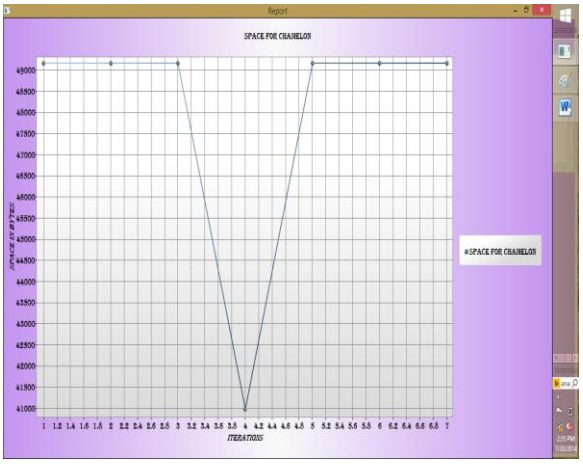


Figure 4.4:Space for HUI-Miner

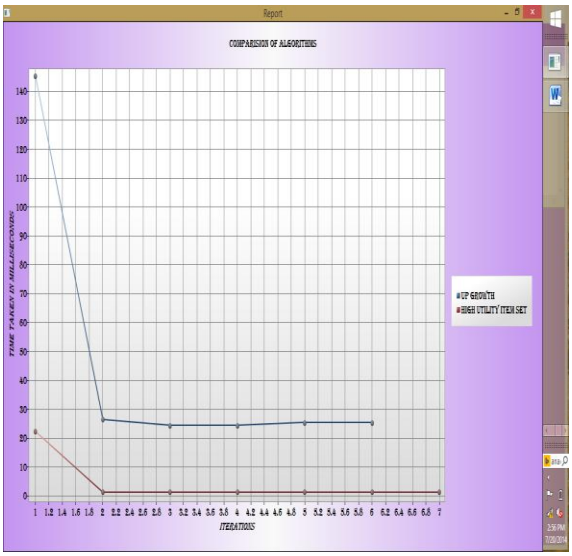


Figure 4.4:Comparison of UP-Growth and HUI-Miner for time complexity

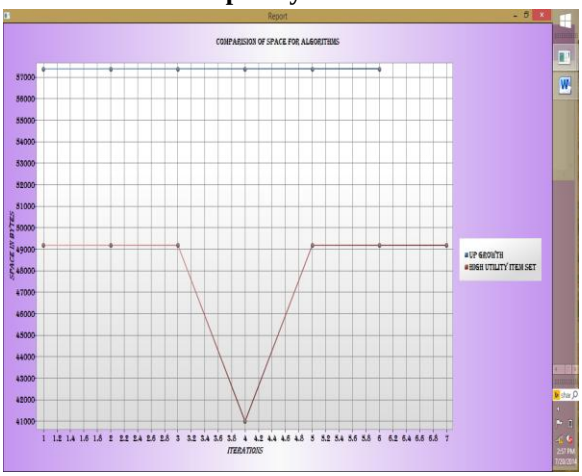


Figure 4.5:Comparison of UP-Growth and HUI-Miner for Space complexity

V. CONCLUSION

Mining high utility item sets becomes more significant. In this paper, the High utility item set miner (HUI) algorithm evaluated with Existing UP-Growth (UPG) algorithm. These algorithms are experimented on synthetic datasets and datasets for different support threshold. From the experimental observation, the conclusion is that, HUI-Miner

algorithm performs well than UPG algorithm for different support values. Also the HUI-Miner algorithm scales well as the size of the transaction database increases.

VI. REFERENCES

[1]. Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases

[2] R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proc.11th Int'l Conf. Data Eng., pp. 3-14, Mar. 1995.

[3] C.F. Ahmed, S.K. Tanbeer, B.-S. Jeong, and Y.-K. Lee, "EfficientTree Structures for High Utility Pattern Mining in Incremental Databases," IEEE Trans. Knowledge and Data Eng., vol. 21, no. 12, pp. 1708-1721, Dec. 2009.

[4] C.H. Cai, A.W.C. Fu, C.H. Cheng, and W.W. Kwong, "Mining
Association Rules with Weighted Items," Proc. Int'l Database Eng.and Applications Symp. (IDEAS '98), pp. 68-77, 1998.

[5] R. Chan, Q. Yang, and Y. Shen, "Mining High Utility Itemsets,"Proc. IEEE Third Int'l Conf. Data Mining, pp. 19-26, Nov. 2003.

[6] J.H. Chang, "Mining Weighted Sequential Patterns in a Sequence Database with a Time-Interval Weight," Knowledge-Based Systems, vol. 24, no. 1, pp. 1-9, 2011.

[7] M.-S. Chen, J.-S. Park, and P.S. Yu, "Efficient Data Mining for Path Traversal Patterns," IEEE Trans. Knowledge and Data Eng., vol. 10,no. 2, pp. 209-221, Mar. 1998.

[8] C. Creighton and S. Hanash, "Mining Gene Expression Databases for Association Rules," Bioinformatics, vol. 19, no. 1, pp. 79-86, 2003.

[9] M.Y. Eltabakh, M. Ouzzani, M.A. Khalil, W.G. Aref, and A.K. Elmagarmid, "Incremental Mining for Frequent Patterns in Evolving Time Series Databases," Technical Report CSD TR#08- 02, Purdue Univ., 2008.

[10] A. Erwin, R.P. Gopalan, and N.R. Achuthan, "Efficient Mining of High Utility Itemsets from Large Data Sets," Proc. 12th Pacific-Asia Conf. Advances in Knowledge Discovery and Data Mining (PAKDD), pp. 554-561, 2008.

[11] E. Georgii, L. Richter, U. Ru¨ ckert, and S. Kramer, "Analyzing Microarray Data Using Quantitative Association Rules," Bioinformatics, vol. 21, pp. 123-129, 2005.

[12] J. Han, G. Dong, and Y. Yin, "Efficient Mining of Partial Periodic Patterns in Time Series Database," Proc. Int'l Conf. on Data Eng., pp. 106-115, 1999.

[13] J. Han and Y. Fu, "Discovery of Multiple-Level Association Rules from Large Databases," Proc. 21th Int'l Conf. Very Large Data Bases, pp. 420-431, Sept. 1995.

[14] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," Proc. ACM-SIGMOD Int'l Conf. Management of Data, pp. 1-12, 2000.

[15] S.C. Lee, J. Paik, J. Ok, I. Song, and U.M. Kim, "Efficient Mining of User Behaviors by Temporal Mobile Access Patterns," Int'lJ. Computer Science Security, vol. 7, no. 2, pp. 285-291, 2007.