IJCSET |September 2012 | Vol 2, Issue 9, 1415-1419 www.ijcset.net | ISSN:2231-0711

Analysis of Utility Based Frequent Itemset Mining Algorithms

HarishBabu. Kalidasu
Priyadarshini Institute of Technology & Science, Tenali. AndhraPradesh

B.PrasannaKumar
Mandava Institute of Engineering & Technology, Jaggayyapet, AP

Haripriya.P
Priyadarshini Institute of Technology & Management, Guntur, AP

Abstract:Knowledge Discovery Plays major role in the business applications. Association Analysis is one among them which helps in Market Basket Analysis. For Generating Frequent Itemsets and Association rules from large datasets, this helps to improving the business. But, Apriori Algorithm, Frequent Pattern Tree generates only frequent patterns and there is a need to generate high utility itemsets to improve the performance of market basket analysis. Generating High Utility Itemsets means, finding the Itemset with high Intrestedness from the data source. Even though there are number of similar approaches are proposed in recent years. It is the problem to find frequent itemsets when the data base having large amount of transactions. A huge amount of candidate itemsets effects on the performance. In this paper, we are comparing high utility based itemset mining algorithms FP Growth algorithm, UP Growth algorithm for finding frequent itemsets from high utility based itemset mining.

CATEGORIES-Data Mining & Applications of Database
General Terms-Itemset Mining, interestedness, high utility, candidate elimination

1. INTRODUCTION

Mining of Frequent Itemsets plays vital role in mining applications. In Market Basket Analysis, all the frequent itemsets are retrieved from transactional database which are often coming together in transactions. The previous studies are used with candidate and without candidate generation. But it is not producing the customer requirement like profit, sales in particular item. In the view of above, itemset mining with high interestedness or utility plays vital role in mining applications. However, the unit profits and purchased quantities of items are not considered in the framework of frequent itemset mining. Hence, it cannot satisfy the requirement of the user who is interested in discovering the itemsets with high sales profits. In view of this, utility mining [2, 3, 4, 6, 7, 8, 9, 10] emerges as an important topic in data mining for discovering the itemsets with high utility like profits. The basic meaning of utility is the interestedness/importance/profitability of items to the users. The utility of items in a transaction database consists of two aspects: (1) the importance of distinct items, which is called external utility, and (2) the importance of the items in the transaction, which is called internal utility. The utility of an itemset is defined as the external utility multiplied by the internal utility. An itemset is called a *high utility itemset* if its utility is no less than a userspecified threshold; otherwise, the itemset is called a *low utility itemset*. Mining high utility itemsets from databases is an important task which is essential to a wide range of applications such as website click streaming analysis,cross-marketing in retail stores, business promotion in chain hypermarkets and even biomedical applications.

The deficiency of this approach is that it does not consider the statistical aspect of itemsets. Utility-based measures should incorporate user-defined utility as well as raw statistical aspects of data. Consequently, it is meaningful to define a specialized form of high utility itemsets, utility-frequent itemsets, which are a subset of high utility itemsets as well as frequent itemsets. Example 1 indicates differences between frequent, high utility and utility-frequent itemsets.

Example 1.

As an example let us analyze sales in a large retail store. We can find that itemset {bread, milk} is frequent, itemset {caviar, champagne} is of high utility and itemset {beer} is utility frequent. A smart manager should pay special attention to itemset {beer} as it is frequent and of high utility. On the other side, itemset {bread, milk} is frequent but not of high utility and itemset {caviar, champagne} gives high utility but is not frequent.

To address this issue, we propose in this paper a novel algorithm with a compact data structure for efficiently discovering high utility itemsets from transactional databases. The major contributions of this work are summarized as follows:

1. A novel algorithm, called *UP-Growth* (*Utility Pattern Growth*), is proposed for discovering high utility itemsets. Correspondingly, a compact tree structure, called *UP-Tree* (*Utility Pattern Tree*), is proposed to maintain the important information of the transaction database related to the utility patterns. High utility itemsets are then generated from the UP-Tree efficiently with only two scans of the database.
2. Four strategies are proposed for efficient construction of UP-Tree and the processing in UP-Growth. By these strategies, the estimated utilities of candidates can be ell reduced by discarding the utilities of the items which are impossible to be high utility or not involved in the search space. The proposed strategies can not only efficiently decrease the estimated utilities of the potential high utility itemsets but also effectively reduce the number of candidates.

2. PROBLEM DEFINITION

A frequent itemset is a set of items that appears at least in a pre-specified number of transactions. Formally, let I = {i1, i2, . . . , im} be a set of items and DB = {T1, T2, ..., Tn} a set of transactions where every transaction is also a set of items (i.e. itemset). Given a minimum support threshold minSup an itemset S is frequent iff: |{T|S ⊆ T, T ⊆ DB, S ⊆ I} / |DB|≥ minSup . Frequent itemset mining is the first and the most time consuming step of mining association rules. During the search for frequent itemsets the anti-

monotone property is used. Let D be the domain of a function f. f has the anti-monotone property when $\forall$ x, y $\in$ D : x $\leq$ y $\Rightarrow$ f(y) $\leq$ f(x). In the case of mining frequent itemsets the anti-monotone property assures that no superset of an infrequent itemset is frequent. Consequently, infrequent candidates can be discarded during the candidate generation phase.

Definition 1 The utility of an item *ip* in the transaction T_d is denoted as $u(ip, Td)$ and defined as $p(i_p) \times q(i_p, T_d)$. For example, in Table 1, $u(\{A\}, T1) = 5 \times 1 = 5$.

Definition 2 The utility of an itemset *X* in *Td* is denoted as $u(X, Td)$ and defined as $\sum_{i_p \in X \wedge X \subseteq T_d} u(i_p, T_d)$ For example, $u(\{AC\}, T1) = u(\{A\}, T1) + u(\{C\}, T1) = 5 + 1 = 6$.

Table 1. An example database

TID	Transaction	TU
T_1	(A,1) (C,1) (D,1)	8
T_2	(A,2) (C,6) (E,2) (G,5)	27
T_3	(A,1) (B,2) (C,1) (D,6) (E,1) (F,5)	30
T_4	(B,4) (C,3) (D,3) (E,1)	20
T_5	(B,2) (C,2) (E,1) (G,2)	11

.

Table 2. Profit table

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

2.1 Related Work

In association with rules mining, Apriori (Agrawal and Srikant, 1995), DHP (Park et al., 1997) and partition-based ones (Lin and Dunham, 1998; Savasere et al., 1995) were proposed to find frequent itemsets. Many important applications have called for the need of incremental mining due to the increasing use of record-based databases to which data are being added continuously. Many algorithms like FUP (Cheung et al., 1996), FUP2 (Cheung et al., 1997) and UWEP (Ayn et al., 1999;Ayn et al., 1999) have been proposed to find frequent itemsets in incremental databases. The tree-based approaches such as FP-Growth [5] were afterward proposed. It's widely recognized that FP-Growth achieves a better performance than Apriori-based approaches since it finds frequent itemsets without generating any candidate itemset and it scans database just twice. A formal definition of utility mining and theoretical model was proposed in Yao et al. (2004), namely MEU, where the utility is defined as the combination of utility information in each transaction and additional resources. Since this model cannot rely on downward closure property of Apriori to restrict the number of itemsets to be examined, a heuristic is used to predict whether an itemset should be added to the candidate set. However, the prediction usually overestimates, especially at the beginning stages, where the number of candidates approaches the number of all the combinations of items. The examination of all the combinations is impractical, either in computation cost or in memory space cost, whenever the number of items is large or the utility threshold is low. Although this algorithm is not efficient or scalable, it is by far the best one to solve this specific problem.

To efficiently generate HTWUIs in phase I and avoid scanning database multiple times, Ahmed et al. [2] proposed a tree-based algorithm, called IHUP, for mining high utility itemsets. They use an IHUP-Tree to maintain the information of high utility itemsets and transactions. Every node in IHUP-Tree consists of an item name, a support count, and a TWU value. The framework of the algorithm consists of three steps: (1) The construction of IHUP-Tree, (2) the generation of HTWUIs and (3) identification of high utility itemsets. The phase I of IHUP In step 1, items in the transaction are rearranged in a fixed order such as lexicographic order, support descending order or TWU descending order. Then, the rearranged transactions are inserted into the IHUP-Tree. Figure 1 shows a global IHUP-Tree for the database in Table 1, in which items are arranged in the descending order of TWU. In step 2, HTWUIs are generated from the IHUP-Tree by applying the FP-Growth algorithm [5]. Thus, HTWUIs in phase I can be found more efficiently without generating candidates for HTWUIs. In step 3, high utility itemsets and their utilities are identified from the set of HTWUIs by scanning the original database once.

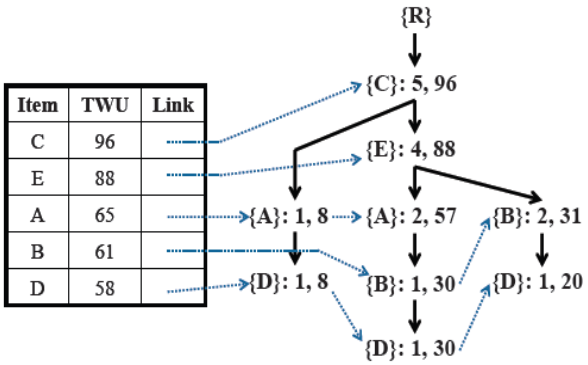


Figure 1: An IHUP-Tree when *min_util* = 40.

3. PROPOSED METHOD

To facilitate the mining performance and avoid scanning original database repeatedly, we use a compact tree structure, called *UP-Tree* to maintain the information of transactions and high utility itemsets.

3.1 The elements in UP-Tree

In UP-Tree, each node *N* includes *N.name*, *N.count*, *N.nu*, *N.parent*, *N.hlink* and a set of child nodes. The details are introduced as follows. *N.name* is the item name of the node. *N.count* is the support count of the node [5]. *N.nu* is called node utility which is an estimate utility value of the node. *.parent* records the parent node of the node. *N.hlink* is a node link which points to a node whose item name is the same as *N.name*. *Header table* is employed to facilitate the traversal of UP-Tree. In the header table, each entry is composed of an item name, an estimate utility value, and a link. The link points to the last occurrence of the node which has the same item as the entry in the UP-Tree. By following the link in the header table and the nodes in UP-Tree, the nodes whose item names are the same can be traversed efficiently.

3.1.2 Discarding global unpromising items during the construction of a global UP-Tree:

The construction of UP-Tree can be performed with two scans of the original database. In the first scan of database, the transaction utility of each transaction is computed. At the same time, TWU of each single item is also accumulated. After scanning database once, items and their TWUs are obtained. By TWDC property, if the TWU of an item is less than minimum utility threshold, its supersets are unpromising to be high utility itemsets. The item is called *unpromising items*. The below Promising item & unpromising item gives a formal definition for unpromising items and promising items.

Promising item and unpromising item: An item *ip* is called a *promising item* if *TWU*(*ip*) ≥ *min_util*. Otherwise, the item is called an *unpromising item*.

Example 1.

Consider the transaction database in Table 1 and the profit table in Table 2. Suppose the minimum utility threshold *min_util* is 40. In the first scan of database, TUs of the transactions and the TWUs of the items are computed. They are shown in the last column of Table 1 and in Table 3, respectively. As shown in Table 3, {*F*} and {*G*} are unpromising items since their TWUs are less than *min_util*. The promising items are reorganized in the header table in the descending order of TWU. Table 4 shows the reorganized transactions and their RTUs for the database in Table 1. As shown in Table 4, unpromising items {*F*} and {*G*} are removed from the transactions *T2*, *T3* and *T5*, respectively. Besides, the utilities of {*F*} and {*G*} are eliminated

from the TUs of *T2*, *T3* and *T5*, respectively. The remaining promising items {*A*}, {*B*}, {*C*}, {*D*} and {*E*} in the transaction are sorted in the descending order of TWU. Then, we insert reorganized transactions into the UP-Tree by the same processes as IHUP-Tree [2]. We use the following example to describe the operation of insertion.

Strategy1. Discarding global unpromising items (DGU). The unpromising items and their utilities are eliminated from the transaction utilities during the construction of a global UP-Tree.

Rationale: The principle of DGU strategy is to discard the information of unpromising items from the database since an unpromising item plays no role in high utility itemsets and only the supersets of promising items are likely to be high utility.

Table 3. Items and their TWUs

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

Table 4. Reorganized transactions and their RTUs

TID	Reorganized transaction	RTU
T_1'	(C,1) (A,1) (D,1)	8
T_2'	(C,6) (E,2) (A,2)	**22**
T_3'	(C,1) (E,1) (A,1) (B,2) (D,6)	**25**
T_4'	(C,3) (E,1) (B,4) (D,3)	20
T_5'	(C,2) (E,1) (B,2)	**9**

3.1.3 Generating PHUIs from the global UP-Tree by FP-Growth:

Each node in the fig. 2., UP-Tree is associated with two numbers: the first one is support count and the second one is node utility. Besides, the nodes which have the same item names are linked in a sequence by their node links. Comparing with the IHUP-Tree in Figure 1, the node utilities of the nodes in UP-Tree are less than the node utilities of the nodes in IHUP-Tree since reorganized transactions are inserted with RTUs instead of TWUs.

In the UP-Tree, each node {*ai*} to the root forms a path ({*ai*} → {*ai*+1} → … → {*an*}). Each path represents a common prefix that shared by multiple reorganized transactions. Besides, {*ai*}.*count* is the number of reorganized transactions that share the path and {*ai*}.*nu* is an estimate utility value for the path. Similar to [2], PHUIs can be generated from the UP-Tree by applying FP-Growth.

3.2 The Proposed Mining Method: UP-Growth:

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 described in previous section 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.

Minimum item utility of an item: The utility of item *ip* in transaction *Td* is called the *minimum item utility* of *ip* if there doesn't exist a transaction *Td*' such that *u*(*ip*, *Td*') < *u*(*ip*, *Td*). The minimum item utility of *ip* is denoted as *miu*(*ip*).

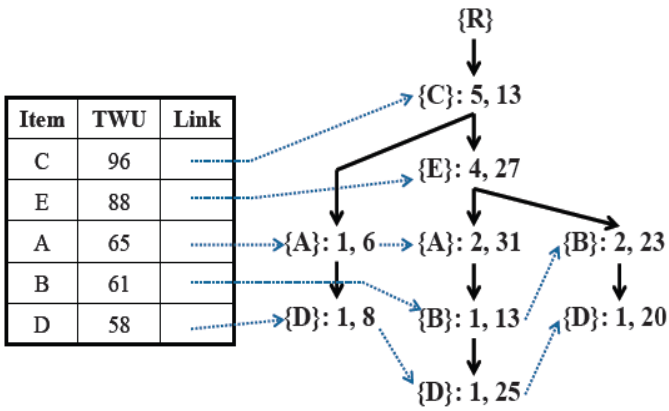


Figure 3. A UP-Tree by applying strategies DGU and DGN.

Table 7. Minimum item utility table

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

Strategy 2. Discarding local unpromising items (DLU). The minimum item utilities of unpromising items are discarded from path utilities of the paths during the construction of a local UP-Tree.

Rationale: 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.

4. EXPERIMENTAL EVALUATION

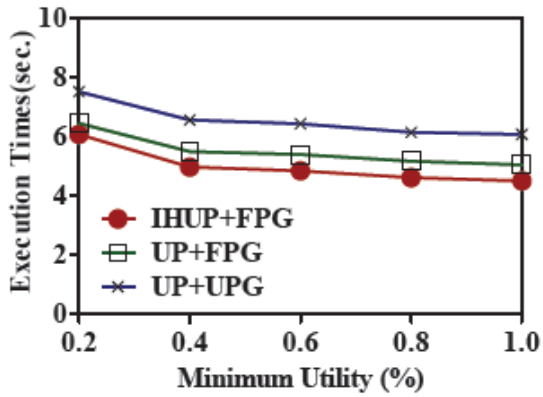
The experiments were performed on a 2.66 GHz Intel Core 2 Quad Processor with 2 gigabyte memory, and running on Windows XP. The algorithms are implemented in Java language. Both synthetic and real datasets are used to evaluate the performance of the algorithms. Synthetic datasets were generated from the data generator in the parameters are described as follows: *D* is the total number of transactions; *T* is the average size of transactions; *N* is the number of distinct items; *I* is the average size of maximal potential frequent itemsets. The utility table and the quantity f each item are generated as the settings in Real world datasets BMS-Web-View-1 and Chess were obtained from FIMI Repository.

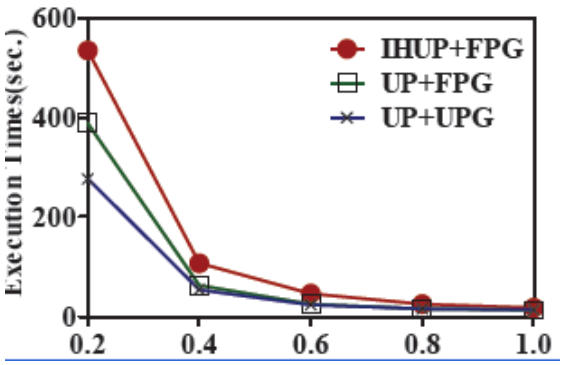
4.1 Evaluation on Synthetic Datasets

As shown in below Table, UP+FPG generates fewer candidates than IHUP+FPG since the node utilities of the nodes in the UP-Tree are less than the IHUP-Tree. This shows the effectiveness of strategies DGU and DGN. By applying strategy DGU, global unpromising items and their utilities are discarded from the transactions and TUs. By applying DGN strategy, the node utilities of the nodes in a global UP-Tree are effectively decreased since the utilities of their descendants are discarded. In this, when the minimum utility threshold is less than 0.6%, the number of candidates generated by UP+UPG becomes smaller than UP+FPG obviously. This indicates that strategies DLU and DLN work well and more itemsets which are impossible to be high utility are reduced than FP-Growth when the minimum utility threshold is low. By applying DLU strategy, local unpromising items are removed from the paths of conditional pattern base and their minimum item utilities are eliminated from the path utilities. By applying DLN, the node utilities of the nodes in a local UP-Tree are decreased since they discard the minimum item utilities of their descendants.

Table: Number of candidates on T10I6D100K

Minimum utility	IHUP+FPG	UP+FPG	UP+UPG
0.2%	20,651	15,057	10,664
0.4%	4,003	2,347	1,990
0.6%	1,684	910	844
0.8%	873	527	521
1.0%	566	411	411





5. CONCLUSIONS

In this paper, we have proposed an efficient algorithm named UP-Growth for mining high utility itemsets from transaction databases. A data structure named UP-Tree is proposed for maintaining the information of high utility itemsets. Hence, the potential high utility itemsets can be efficiently generated from the UP-Tree with only two scans of the database. Besides, we develop four strategies to decrease the estimated utility value and enhance the mining performance in utility mining. In the experiments, both of synthetic and real datasets are used to evaluate the performance of our algorithm. The mining performance is enhanced significantly since both the search space and the number of candidates are effectively reduced by the proposed strategies. The experimental results show that UP-Growth outperforms the state-of-the-art algorithms substantially, especially when the database contains lots of long transactions.

REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th Int'l Conf. on VeryLarge Data Bases*, pp. 487-499, 1994.

[2] 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.

[3] R. Chan, Q. Yang, and Y. Shen. Mining high utility itemsets. In *Proc. of Third IEEE Int'l Conf. on Data Mining*, pp. 19-26, Nov., 2003.

[4] 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.

[5] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proc. of the ACM-SIGMOD Int'l Conf. on Management of Data*, pp. 1-12, 2000.

[6] 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.

[7] Y. Liu, W. Liao, and A. Choudhary. A fast high utility itemsets mining algorithm. In *Proc. of the Utility-Based Data Mining Workshop*, 2005.
[8] B.-E. Shie, V. S. Tseng, and P. S. Yu. Online mining of temporal maximal utility itemsets from data streams. In *Proc. of the 25th Annual ACM Symposium on Applied Computing*, Switzerland, Mar., 2010.
[9] 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.
[10] S.-J. Yen and Y.-S. Lee. Mining high utility quantitative association rules. In *Proc. of 9th Int'l Conf. on Data Warehousing and Knowledge Discovery, Lecture Notes in Computer Science 4654*, pp. 283-292, Sep., 2007.
[11] Frequent itemset mining implementations repository, http://fimi.cs.helsinki.fi/

AUTHORS:

HarishBabu Kalidasu has completed his Bachelor of Technology in Andhra University. He is pursuing Post graduation M.Tech (CSE) at JNTU Kakinada University, at present he is working as Asst.Professor CSE Dept, in Priyadarshini Institute of Technology & Science, Tenali. He has 3 years of experience in teaching field. He is author or coauthor of more than ten papers includes Networking, software engineering, Data Mining.

Mr. B.Prasanna Kumar has completed his Bachelor of Technology of CSE in Rao & Naidu College of Engineering Ongole. He is completed M.Tech (CSE) in Acharya Nagarjuna University, Guntur, Andhra Pradesh. He has 8 Yrs of experience in teaching field, presently working as Associate Professor & Head, Department of Computer Science & Engineering at Mandava Institute of Engineering and Technology, Jaggayyapet, Krishna (Dt), Andhra Pradesh.

Ms. Haripriya.P has completed her Bachelor of Technology in JNTU Kakinada. She is Pursuing post graduation in M.Tech (CSE) at JNTU Kakinada University, at present she is working as Asst.Professor of CSE Dept, in Priyadarshini Institute of Technology & Management.