ISSN:0975-9646

Fast Algorithm for Finding the Value-Added Utility Frequent Itemsets Using Apriori Algorithm

G.Arumugam[#1], V.K.Vijayakumar[*2]

[#] *Senior Professor and Head, Department of Computer Science*
Madurai Kamaraj University
Madurai, Tamil Nadu, India

[*]*Associate Professor and Head, Department of Computer Science*
Sourashtra College
Madurai, Tamil Nadu, India

***Abstract*— In association rule mining, frequency of an itemset alone does not assure its interestingness because it does not contain information on subjectively defined utility such as profit in rupees or some other variety of utility. This leads to a fruitful deviation in the frequent itemset mining called utility based data mining. Mining high utility itemsets upgrades the standard frequent itemset mining framework. Fast Utility Frequent Mining (FUFM) is a popular algorithm to find utility frequent itemsets (UFIS) based on the extended support measure. But, when applying FUFM on transactional databases with highly fluctuated transaction utility values, it generates less number of UFIS. This is due to the effect of transactions with low utility values. In this paper, an algorithm Fast Value-Added Utility Frequent Mining (FVAUFM) is proposed to find UFIS. FVAUFM works on the efficient and reduced database got by removing the transactions below a threshold value from the given transactional database. The proposed algorithm FVAUFM uses a novel measure called *average-utility* measure to find UFIS. Experiments show that FVAUFM generates more number of UFIS than FUFM and also saves a significant amount of memory space and running time.**

***Keywords*— Utility frequent itemsets, Association rule mining, Apriori algorithm**

I. INTRODUCTION

In market basket analysis, the problem of finding frequent itemsets does not takes into account the profit or quantity of items. For example, consider the following two transactions:

T1: {10 packets of Razor blade, 5 quantities of shaving cream}

T2: {1 packet of Razor blade, 1 quantity of shaving cream}

While finding frequent itemsets using the Apriori algorithm [3,4], it does not consider the profit or quantity of items in the transactions. In view of the support-confidence frame work the above two transactions are considered to be the same.

Suppose that the profit per packet of Razor blade is one rupee and that of a shaving cream is two rupees. In this case, the total profit for transaction T1 is twenty rupees, and for transaction T2 is three rupees. Thus the transaction T1 gives more profit than the transaction T2. Therefore, in order to make efficient marketing, both the profit (called as external utility) and the quantity (called as internal utility) of items must be taken into account. As a consequence, a new dimension of frequent itemset mining, called utility based data mining emerges.

Utility based data mining [1, 2, 6, 7, 8, 9, 10, 11] is a recent development in the field of association rule mining. User-defined utility is based on information not available in the transaction dataset. It reflects user preference and can be represented by an *external utility* table or utility function. Utility table defines utilities of all items in a given database. In addition, it requires transaction dependent *internal utilities* (e.g. quantities of items in transactions). Utility of an itemset takes into account both internal and external utility of all items in a itemset. It is defined as a sum of products of internal and external utilities of present items.

Fast Utility Frequent Mining(FUFM) is a popular algorithm to find utility frequent itemsets (UFIS) based on the extended support measure. But, when applying FUFM on transactional databases with highly fluctuated transaction utility values, it generates less number of UFIS. This is due to the effect of transactions with low utility values. In this paper, an algorithm Fast Value-Added Utility Frequent Mining (FVAUFM) is proposed to find UFIS. FVAUFM works on the efficient and reduced database got by removing the transactions below a threshold value from the given transactional database. The proposed algorithm FVAUFM uses a novel measure called *average-utility* measure to find UFIS. Experiments show that FVAUFM generates more number of UFIS than FUFM and also saves a significant amount of memory space and running time

II. BACKGROUND STUDY.

The goal of high utility itemset mining is to find all itemsets that give utility greater or equal to the user specified threshold. High-utility itemset mining model was defined by Yao, Hamilton and Butz [9]. The objective of high utility itemset mining is to find all itemsets that give utility greater than or equal to a user specified threshold. The following is the set of definitions given in [9] which is illustrated on small example. .

Definition 1 (External Utility): The external utility of an item i_k is a numerical value y_k defined by the user. It is transaction independent and reflects importance (normally profit) of the item. External utilities are stored in an utility table as shown in Table I. For example, the external utility of item 5 in Table I is 3.

Definition 2 (Internal Utility): The internal utility of an item i_k is a numerical value x_k which is transaction dependent. In general it is defined as the quantity of an item in transaction. Internal utilities are stored in an utility table as shown in Table II. For example, the internal utility of item 5 in transaction T_1, is 4.

TABLE I
EXTERNAL UTILITIES OF ITEMS IN THE SET I = {1,2,3,4,5}

Item	1	2	3	4	5
External Utility (or)Profit (y_k)	5	2	1	4	3

TABLE II
MARKET BASKET DATABASE WITH 5 TRANSACTIONS AND 5 DISTINCT ITEMS

TID	ITEMS BOUGHT	QUANTITY(OR)INTERNAL UTILITY(X_K)
T1	{1,2,5}	{2,1,4}
T2	{1,2,4}	{5,3,6}
T3	{1,3}	{4,3}
T4	{1,2,3,5}	{1,1,1,1}
T5	{1,2,3}	{2,2,2}

Definition 3 (Utility Function): Utility function f is a function of two variables: $f(x,y): (R^+, R^+) \rightarrow R^+$, which is defined as $f(x_k , y_k) = x_k * y_k$.

Definition 4 (Utility of an item in a Transaction): The utility of item i_k in transaction T is defined as $u(i_k ,T) = f(x_k , y_k)$, $i_k \in T$. For example utility of item 2 in transaction T_1 is 1*2 = 2.

Definition 5 (Utility of an itemset in a Transaction): The utility of itemset S in transaction T is defined as $u(S,T)= \sum u(i_k ,T)$, $i_k \in S$, $S \subseteq T$. For example let S={2,4}, then the Utility of itemset S in transaction T_2 is $u(S,T_2)= u(\{2\},T_2)+u(\{4\},T_2)= 3 *2 +6*4 = 30$.

Definition 6 (Utility of an item in an Itemset): The utility of item i_k in itemset S is defined as $u(i_k,S)= \sum u(i_k ,T)$, $T \in D$, $S \subseteq T$. For example, utility of item 5 in itemset S={2,5} is $u(5,\{2,5\})= u(5,T_1)+u(5,T_4)= 4 *3 +1*3 =15$.

Definition 7 (Utility of an itemset in a database): The utility of itemset S in database D is defined as $u(S)= \sum u(S,T)$, $T \in D$, $S \subseteq T$. For example Utility of itemset S={1,5} is $u(\{1,5\})= u(\{1,5\},T_1)+u(\{1,5\},T_4)= 30$.

Definition 8 (Utility of a Transaction): The utility of transaction T is defined as $u(T)=\sum u(i_k ,T)$, $i_k \in T$. For example Utility of transaction T_5 is $u(T_5) = u(\{1\},T_5)+u(\{2\},T_5)+ u(\{3\},T_5) =16$.

Definition 9 (Utility of a Database): The utility of database D is defined as $u(D)=\sum u(T),T\in D$. For example Utility of database D is $u(D) = u(T_1) + ...+ u(T_5) = 24+55+23+11+16 = 129$.

Definition 10 (Utility Share of an itemset in a Database): The utility share of itemset S in database D is defined as $U(S)=u(S) / u(D)$. For example utility share of itemset {1,5} in database from Table II is $U(\{1,5\}) = 30/129 = 0.2325=23.25\%$.

Definition 11 (High Utility Itemset): Itemset S is of high utility iff $U(S) \geq$ *min_util*, where *min_util* is user defined utility threshold in percents of the total utility of the database.

Definition 12 (High Utility Itemset Mining): High utility itemset mining is the problem of finding set H defined as $H = \{S / S \subseteq I, U(S) \geq min\_util \}$ where I is the set of items.

Utility frequent itemsets are a special form of high utility itemsets. For a given utility threshold μ each itemset S is associated with a set of transactions defined as

$\tau (S, \mu) = \{T / S\subseteq T \text{ and } u(S,T) \geq \mu \text{ and } T \in D\}$.

Definition 13 (Extended support measure): Now, the extended support measure is is defined as **Support (S, μ)** $=| \tau (S, \mu) | / |D|$

Definition 14 (Utility Frequent Itemset): Itemset S is utility-frequent if for a given utility threshold μ and support threshold s , the extended support measure **Support (S, μ)** $\geq$ s.

A. FUFM algorithm

The algorithm FUFM (Fast Utility-Frequent Mining) [7] is based on the fact that utility-frequent itemsets are a special form of frequent itemsets. Moreover, the support measure is always greater or equal to the extended support measure. When computing extended support, it requires to count only those transaction containing given itemset S that also gives minimum utility on S. Frequent itemset mining algorithms can be used to mine utility-frequent itemsets. These algorithms are well studied and also very efficient. For this reason, the FUFM algorithm is very simple and fast because the main part is the "external" frequent itemset mining algorithm. It is straight forward to find utility-frequent itemsets among frequent itemsets. Table III shows pseudo code of FUFM algorithm.

TABLE III
PSEUDO CODE OF THE FUFM ALGORITHM

```
Algorithm 1: FUFM /* Fast Utility Frequent Mining */
Input:
-Database D
-Constraints min_util and min_sup
Output:
-All utility-frequent itemsets
1  L = 1
2  Find the set of candidates of length L with
   support ≥ min_sup
3 Compute extended support for all candidates and output
   utility-frequent itemsets
4  L += 1
5  Use the frequent itemset mining algorithm to obtain new
set
   of frequent candidates of length L from the old set of
   frequent candidates
6  Stop if the new set is empty otherwise go to step 3.
```

The time and space complexity are both fully determined with the complexity of the frequent itemsets mining method used. In the algorithm FUFM, the utility threshold does not have influence on the candidate generation phase, it performs worse in this special case as it has to inspect all frequent itemsets regardless of their utility.

III. Proposed System and Methodology

The algorithm Fast Utility Frequent Mining (FUFM) [5] generates utility frequent itemsets using the extended support measure. The FUFM algorithm deals with all transactions of the database without considering their utility values, which results in performance degradation. The transactions with low utility value do not add any value to the knowledge and in addition slow down the mining process. This leads a way to define an efficient transaction. Given a transactional database DB, find the maximum utility value of the transaction, MAX{u(T), T∈DB}. A transaction T is said to be an *efficient transaction* if, u(T) / MAX{u(T), T∈DB} ≥ *min_trn_uty*, where *min_trn_uty* (λ) is a user specified minimum transaction utility threshold (MTUT), usually given in percentage and u(T) is the utility value of the transaction T.

In this context, an algorithm Fast Value-Added Utility Frequent Mining (FVAUFM) is proposed to find utility frequent itemsets. FVAUFM works on an effective database say, DB′ formed by taking the efficient transactions from DB. In other words, the effective database DB′ is formed by eliminating all the inefficient transactions from DB. FVAUFM finds utility frequent itemsets using a novel measure called *average-utility measure* defined as follows:

Given an itemset S, the *average utility* of an itemset S is defined as the ratio of the utility of itemset S to the support count of S. (i.e.,) Average-Utility(S) = AU(S) = u(S) / SC(S), where u(S) is the utility of the itemset S and SC(S) is the support count of S. When the utility values of the transactions in the database DB are highly fluctuated, the *average-utility measure* works well on the effective database DB′ than the extended support measure used by FUFM.

Since the algorithm FVAUFM works on the effective database DB′ using average utility measure, it adds more knowledge to the results. Hence the utility frequent itemset generated by FVAUFM is called the *Value-Added Utility Frequent Itemset* (VAUFIS) defined as shown below:

An itemset S in DB′ is said to be a *Value-Added Utility Frequent Itemset* (VAUFIS) if it satisfies the following conditions:

(i) SC(S) / |DB′| ≥ *min_sup* and

(ii) AU(S) / u(DB′) ≥ *min_avg_uty*

Where *min_sup (α)* and *min_avg_uty (γ)* are the user specified minimum support threshold (MST) and minimum average utility threshold (MAUT) respectively.

A detailed performance analysis is done by comparing FVAUFM with the FUFM algorithm on a sample database. The result shows that the data reduction and the average-utility measure together increases the performance of FVAUFM in terms of memory space and running time.

Example 1: Consider the market basket database in Table IV. Here **MAX** {u(T), T∈DB}= 55. Let us assume that *min_trn_uty* = 25 % . Now a transaction T is said to be an efficient transaction if u(T)/**MAX**{u(T), T∈DB} ≥ *min_trn_uty* (or) u(T)≥*min_trn_uty****MAX**{u(T), T∈DB} (i.e.,) u(T) ≥ 25/100 * 55 = 55/4 = 13.75 ≈ 14. Therefore the effective transactions are those transactions T whose utility value u(T) ≥ 14. From Table IV, the effective transactions are {T_1,T_2,T_4,T_5,T_7,T_9} marked with * in Table IV. The effective database DB′ after removing the inefficient transactions from DB are shown in Table IV.

TABLE IV
MARKET BASKET DATABASE DB, WITH 9 TRANSACTIONS AND 5 DISTINCT ITEMS

TID	**Items bought with quantity**					**Uty. of tran. u(T)**
	A 5	**B 2**	**C 1**	**D 4**	**E 3**	
T_1	2	1	-	-	4	24^*
T_2	-	3	-	5	-	26^*
T_3	-	1	1	-	-	3
T_4	5	3	-	6	-	55^*
T_5	3	-	5	-	-	20^*
T_6	-	5	2	-	-	12
T_7	4	-	3	-	-	23^*
T_8	1	1	1	-	1	11
T_9	2	2	2	-	-	16^*
Total Transaction Utility - u(DB)						190

TABLE V
EXTERNAL UTILITIES OF ITEMS OF THE DATABASE DB IN TABLE I

Item	A	B	C	D	E
Ext.Utility (or)Profit	5	2	1	4	3

TABLE VI
EFFECTIVE DATABASE DB′

TID	**Items bought with quantity**					**Uty. of tran. u(T)**
	A (5)	**B (2)**	**C (1)**	**D (4)**	**E (3)**	
T_1	2	1	-	-	4	24
T_2	-	3	-	5	-	26
T_4	5	3	-	6	-	55
T_5	3	-	5	-	-	20
T_7	4	-	3	-	-	23
T_9	2	2	2	-	-	16
Total Transaction Utility - u(DB′)						164

A. *FVAUFM algorithm*

The Pseudo Code of the algorithm FVAUFM is shown in Table VII

TABLE VII
PSEUDO CODE OF THE FVAUFM ALGORITHM

```
Algorithm FVAUFM
Input:
-Transactional database DB
-constraints min_util and min_Sup
Output:
-all value-added utility frequent itemsets in the set V
1. L = 1;
2. Find the set of candidates 'C' of length L with support
   count ≥ min_Sup (i.e.,) C = {c / SC( c ) ≥ min_Sup};
3. For each candidate c ∈ C, Compute the average utility
   value using the formula: AVG-UTIL (c) = u(c) / SC(c) ;
4. Form the set V of value-added utility frequent itemsets,
   V={ c ∈ C / AVG_UTIL(c) ≥ min_util};
5. L = L + 1;
6. Use the frequent itemset mining algorithm to obtain new set
   of frequent candidates of Length L from the old set of
   frequent candidates;
7. Stop if the new set is empty otherwise go to step 3.
```

IV. Experiments

Let us compare the proposed algorithm FVAUFM with FUFM using the sample database given in Table IV.

A. *FUFM algorithm*

The FUFM algorithm is given in section 2. The utility frequent itemsets are generated in FUFM using extended support measure defined in section 2.

Let us apply the FUFM algorithm on the database DB in Table IV and generate UFIS.

From Table IV, u(DB) = 190 and |DB| = 9

Let MST = s = *min_sup* = 20% and MUT = μ = *min_util* = 10 %

An itemset S in DB is said to be UFIS if Support (S, μ) ≥ *s*

(i.e,) S is UFIS, if | τ (S, μ) | / |DB| *≥ 20% (or)* | τ (S, μ) | *≥ 20% * 9 = 1.8 ≈ 2*

u(S,T) *≥ 10% of u(DB) = 10*190/100 = 19*

When the FUFM algorithm is applied on the database in Table IV it generates the UFIS as shown in Table VIII. FUFM algorithm uses Apriori algorithm to generate the frequent itemsets and extended support measure to generate UFIS.

Table VIII
Result of FUFM algorithm on the sample database in table IV

S.No.	FIS(S)	SC(S)	|τ (S, μ)|	UFIS ?
1	A	6	2	Yes
2	B	7	0	-
3	C	6	0	-
4	D	2	2	Yes
5	E	2	0	-
6	{A,B}	4	1	-
7	{A,C}	4	2	Yes
8	{A,E}	2	1	-
9	{B,C}	4	0	-
10	{B,D}	2	2	Yes
11	{B,E}	2	0	-
12	{A,B,C}	2	0	-
13	{A,B,E}	2	1	-

From the above table we observe that there are only four utility frequent itemsets namely {A}, {D}, {A,C} and {B,D}. Let us apply the FVAUFM algorithm on the effective database DB′ given in Table III. From Table III, u(DB′) = 164 and |DB′| =6

Let the MST = s′ = 20% (i.e,) SC(S) ≥ 20% *|DB′| = 20 / 100 * 6 = 1.2 ≈ 1

(i.e,) S is a frequent itemset if SC(S) ≥ 1.

Let the Minimum Average Utility Threshold = MAUT = μ′ = 1%

(i.e,) u(T) ≥ 1% of u(DB′) = 1 % of 164 = 16.4 ≈ 16

Thus an itemset S is said to be VAUFIS, if SC(S) ≥ 1 and u(T) ≥ 16.

Let us apply the apriori algorithm on DB′ to find the FIS and then apply FVAUFM to find VAUFIS as shown in Table IX.

Table IX
Result of FVAUFM algorithm on the sample database in Table IV

S.No.	FIS(S)	SC(S)	u(S)	Avg.util(S) AU(S) u(S)/SC(S)	VAUFIS ?
1	A	5	80	16	Yes
2	B	4	18	4.5	-
3	C	3	10	3.3	-
4	D	2	44	22	Yes
5	E	1	12	12	-
6	{A,B}	3	57	19	Yes
7	{A,C}	3	55	18.3	Yes
8	{A,D}	1	49	49	Yes
9	{A,E}	1	22	22	Yes
10	{B,C}	1	6	6	-
11	{B,D}	2	56	28	Yes
12	{B,E}	1	14	14	-
13	{A,B,C}	1	16	16	Yes
14	{A,B,D}	1	55	55	Yes
15	{A,B,E}	1	24	24	Yes

Thus from Table IX, it is noted that FVAUFM finds ten Value Added Utility Frequent Itemsets (VAUFIS) . They are listed below:

{A}, {D},{A,B},{A,C}, {A,D},{A,E}, {B,D},{A,B,C}, {A,B,D}, {A,B,E}

Comparing FUFM with FVAUFM, it is observed that FUFM finds only four UFIS using extended support measure. But FVAUFM finds ten number of VAUFIS. It is observed that among the ten VAUFIS, the rarely occurring but profitable itemsets are: {AD, ABD} found by FVAUFM algorithm, whereas FUFM fails to find such itemsets.

V. Conclusion

In this chapter a novel fast algorithm FVAUFM is introduced for mining all value-added utility frequent itemsets. It is considerably faster than the algorithm FUFM and also much simpler to implement. FVAUFM is based on efficient methods for mining frequent itemset. It also performs well on real-sized databases.

FVAUFM algorithm and FUFM algorithm were both implemented and tested on a few synthetic databases. The algorithm FVAUFM is also tested on a real dataset from a retail store and the results are analyzed which could be used in practice. Since the FVAUFM algorithm works on the effective database, the speed of the process of finding VAUFIS is increased and also a significant amount of memory space is saved.

Acknowledgment

The authors thank the sourashtra college authorities for providing encouragement and support during the preparation of this research work.

REFERENCES

[1] [ACM05]ACM SIGKDD *Workshop on utility-based data mining*, 2005. Available at: http://storm.cis.fordham.edu/˜gweiss/ubdm-kdd05.html

[2] [ACM06]ACM SIGKDD *Workshop on utility-based data mining*, 2006. Available at: http://www.ic.uff.br/ bianca/ubdm-kdd06.html

[3] [AS94] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proceedings of the VLDB, Santiago de Chile, Chile, pp. 487-499, September 1994.

[4] [HK01] J. Han and M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, San Francisco, CA, 2001.

[5] V.Podpecan , N. Lavrac , I. Kononenko : A Fast Algorithm for Mining Utility-Frequent Itemsets, 2007.

[6] [Pod07] V.Podpecan : *Utility-based Data Mining*. BSc Thesis (in Slovene), University of Ljubljana, 2007.

[7] [PLK07]V.Podpecan , N. Lavrac , I. Kononenko : *A Fast Algorithm for Mining Utility-Frequent Itemsets*, 2007.

[8] [WZS06] G.Weiss, B.Zadrozny, T.Saar, M.sechansky : *Utility-based data mining 2006* workshop report. SIGKDD Explorations, volume 8, issue 2.

[9] [YHB04] H.Yao, H.J.Hamilton , C.J.Butz: *A Foundational Approach to Mining Itemset Utilities from Databases*. In The Fourth SIAM International Conference od Data Mining SDM, pp. 428–486, 2004.

[10] [YHG06] H.Yao , H.J.Hamilton , L.Geng.: *A Unified framework for Utility based Measures for Mining Itemsets*. Second International Workshop on Utility-Based Data Mining, Philadelphia, Pennsylvania, 2006.

[11] [YLC07] J.S.Yeh, Y.C.Li,, C.C.Chang : *A Two-Phase Algorithm for Utility-Frequent Mining.* To appear in Lecture Notes in Computer Science, International Workshop on High Performance Data Mining and Applications, 2007.