International Journal of Applied Engineering Research, ISSN 0973-4562 Vol. 10 No.29 (2015)

Finding High Utility Itemsets using Fuzzy Tail Node Tree (FTNT) from Large Transactional Databases

D.Sathyavani[1], Dr. D.Sharmila[2]

1Faculty of Computer Sciences and Engineering, United Institute of Technology, Coimbatore, Tamil Nadu, India
sathyavani.it@gmail.com
2 Head and Professor, Bannari Amman Institute of Technology ,Sathyamangalam, Tamil Nadu, India
sharmiramesh@rediffmail.com

Abstract

Utility mining is finding the most important profitable items in a large transaction database. Also it allows user to identify the importance of items using different values. In existing system had two steps for finding high utility itemsets: first generates large number of candidate itemsets; then identifies high utility itemsets from generated candidates by doing additional scan in original transaction database. Since large number of candidate itemsets generated and requires more memory space. Finally it decreases the mining efficiency also it doesn't work with minimum utility threshold value. In order to overcome this problem, we proposed a new method fuzzy based tail node tree structure (Fuzzy Tail-Node Tree). Fuzzy high utility item-mine (FTNT-Mine) is applied to find the utility itemsets. The fuzzified quantity information can able to get from transaction database in order to reflect the fuzzy degree of purchased quantity. The performance of FTNT was evaluated in comparison with the state-of-the-art algorithms on different types of datasets. The experimental result shows that proposed system is more effective than the existing methods in terms of execution time and memory space under minimum utility threshold value. It achieves two orders of magnitude faster than the state-of-the-art algorithms on dense dataset, and more than one order of magnitude on sparse datasets.

Keywords: Fuzzy Tail-Node Tree; Fuzzy Mining; fuzzy set; Fuzzification; Utility Mining

1. Introduction

Data mining is the process of extracting important itemsets from the transaction database that is potentially useful for decision making process to increase awareness and sales report analysis. One of the important tasks in data mining is high utility mining which refers to the discovery of more profitable things. Mines the high utility itemset from the transaction database when the utility of an item is greater than or equal to user specified minimum utility threshold, then the item is profitable.

For example, we assume the frequency of item A is 10, item B is 5, and itemset AB is 3. The profit of item A is 3, B is 2. The utility value of A is 10 * 3 = 30, B is 5* 2= 10, and AB is 3 * 3 + 5 * 2 = 19. If the minimum utility threshold is 20, then A is a high utility itemset. So this is called profit item. A number of existing algorithms have been proposed for mining high utility itemset. Most of the existing algorithm based on two phase[1] TWU mining [3], UMMI algorithm[7]. Two phase algorithm mines the high utility itemset in two phases using the transaction weighted down closure property in phase I to find the HTWU (High Transaction Weighted Utility) item. In Phase II mines the high utility itemset from first phase output of HTWU item. TWU mining use a tree structure to capture the utility itemsets information. The exiting UMMI algorithm is introduced to overcome the shortcoming of two phases, TWU mining which consists of large number of HTWU itemsets thereby it take more execution time and memory. Another High utility itemsets can be mined using two phase's algorithm: maximal phase and utility phase. Maximal phase mines maximal transaction weighted utility (MTWU) item using maximal itemset property and utility phase discover the high utility itemset from MTWU item used UP tree. Most of the exiting algorithm has some weakness to mining high utility itemset: a) Itemsets with profit slightly less than the user defined threshold

value is discarded. For example, if the minimum utility threshold is 30, itemsets with profit 29 will be pruning the lower threshold utility itemsets even though the itemset may be important. b) Quantity information about the high utility itemset is not reflecting that a sales manager is interested. c) Profit quantity information is also not provided in the existing algorithm. These shortcomings influence the ability to select the more profitable and cost saving products. The weakness of high utility itemset mining can be overcome by applying fuzzy weight transaction itemset based high utility itemsets mining. Fuzzy set theory yields better results when applied in data mining. Fuzzy logic based mine the high utility uncertainty itemsets of particular item in which the membership value lies in the interval [0, 1] so it will work minimum user threshold value.

2. Related Work

In many existing algorithm that are often unclear and also inaccurate to find the high utility itemsets from long transaction itemsets. Fuzzy logic is used to represente in real life datasets that deal with uncertainty as it gives a flexible method to derive a high-level concept of given problem. Fuzzy logic and data mining jointly present a means for generating more conceptual utility mining at a higher level. Ferdinando et al. presented a novel method for finding association rules [9] from datasets based on fuzzy transforms techniques. Apriori Gen algorithm [8] was used to extract fuzzy association rules represent in the form of linguistic terms. A pre-processing phase was used to determine optimal fuzzy partitions of quantitative attribute domains.

This work extends appropriately to Frequent Pattern Tree approach in the fuzzy domain. The presented Fuzzy Frequent Pattern Tree (Seferina Mavroudi) retains the efficiency of the crisp Frequent Pattern Tree, while at the same time the careful updating of the fuzzy sets at all the phases of the algorithm tries to preserve most of the original information at the data set. Most fuzzy controllers and fuzzy expert systems must predefine membership functions and fuzzy inference rules to map numeric data into linguistic variable terms and to make fuzzy reasoning work. In this paper[13] proposed a general learning method as a framework for automatically deriving membership functions and fuzzy if-then rules from a set of given training examples to rapidly build a prototype fuzzy expert system.

Utility Mining covers all aspects of "profitable" utilities that affect a business and helps in detecting rare high[5] utility itemsets from the transaction. High Utility Rare Itemset Mining (HURI) is very beneficial in several real-life datasets. In Jyothi et al. Presented a survey of the various approaches and algorithms for mining the high-utility itemsets and rare itemset mining[2] presented a fast and efficient fuzzy ARM algorithm on very large datasets. The algorithm was 8 to 19 times faster than traditional fuzzy ARM on very large standard datasets. In[3], unlike most two-phased ARM algorithms, the authors presented individual itemset processing as opposite to concurrent itemset processing at each k number of level, recording some performance improvements.

In C. Saravanabhavan et al. presented a tree based mining high utility itemsets. Firstly, the authors developed a utility frequent pattern tree structure to store important information of utility itemsets. Next the pattern growth methodology was used to mine the entire utility pattern sets. Two algorithms, UP-Growth (Utility Pattern Growth) and UP-Tree (Utility Pattern Tree) are proposed [10] for mining high utility itemsets. Also a set of effective strategies are discussed by Sadak Murali et al, proposed for efficiently pruning candidate itemsets.

A frequent itemset only reflects the statistical correlation[6] between items, and it does not reflect the semantic significance of the items. In this [4], propose a utility based itemset mining approach to overcome this limitation. The proposed approach permits users to quantify their preferences concerning the usefulness of itemsets using utility values. The usefulness of an itemset is characterized as a utility constraint. That is, an itemset is interesting to the user only if it satisfies a given utility constraint. We show that the pruning strategies used in previous itemset mining approaches cannot be applied to utility constraints. In response, we identify several mathematical properties of utility constraints. Then, two novel pruning strategies are designed. Two algorithms for utility based itemset mining are developed by incorporating these pruning strategies.

Rapid growth of temporal data mining has increased the interest of researchers in this era. Temporal data mining generates temporal

association rules[11]. These rules encapsulate the transaction of entity with respect to time. The result generates a quantitative value. But it is easy manipulate the fussy set rather than quantitative values with the help of member function. The paper presents a review on fuzzy logic and temporal association rule. This paper gives an idea in order to apply fuzzy set on temporal data mining.

we use a maximal itemset property and propose an algorithm called UMMI (high Utility Mining using the Maximal Itemset property)[7] to significantly reduce the number of potential itemsets in the first step. In the second step, UMMI uses an effective lexicographic tree structure to determine all of the high utility itemsets. In general, UMMI outperforms all three of the previously used algorithms, including CTU-PRO, an optimized TWU-mining algorithm, and Two-Phase, in our experiments using synthetic datasets.

One of the most necessary areas of the application of fuzzy set theory is fuzzy rule-based systems[14]. These knowledge extraction tools discover essential associations contained in a data base. Fuzzy techniques improve the explanation and understandability of buyer models. In Casillas et al presented a new approach for buyer behavior modeling which is based on fuzzy association rules (FAS). A behavioral model was presented which centered on consumer attitude towards shopping itemsets. The proposed algorithm efficiently finds the high utility itemsets and pruning the lower utility itemset if used defined threshold value is low.

Contributions of this paper are summarized as the following techniques:

(1) We propose a new tree structure named FTNT
(Fuzzy based Tail node Tree) for maintaining important information in global header table related to a transaction dataset.

(2) We also give an algorithm named FT-Mine for over uncertain transaction datasets.

(3) Propose an algorithm that uses FTWU with pattern growth based on a compact utility pattern tree data structure. Our algorithm implements a fuzzy tail node tree scheme to use disk storage when the main memory is inadequate for dealing with large datasets.

(4) Fuzzy tail node tree based mining the high utility itemsets with minimum no of tree creation.

3. Terms and Definitions

Let $D = \{T_1, T_2 \dots T_n\}$ be a transaction dataset which contains n transaction itemsets and m distinct items, i.e. $I = \{i_1, i_2 \dots i_m\}$. Each transaction itemset is represented as $\{i_1:p_1, i_2:p_2 \dots i_v: pv\}$, where $\{i_1, i_2, i_v\}$ is a subset of I, and pu $(1 \le u \le v)$ is the existential probability of item i_u in a transaction itemset. The size of dataset D is the number of transaction itemsets and is denoted as $|D|$. An itemset $X = \{i_1, i_2 \dots i_k\}$, which contains k dissimilar items, is called a k-itemset, and k is the length of the itemset X.

Definition 1: The *support value* (*SV*) of an itemset X in a transaction dataset is defined by the number of transaction itemsets containing X.

Definition 2: The *expected support value (expSV*) of an itemset X in an transaction dataset is denoted as *expSN*(X) and is defined by

$$expSV(X) = \sum_{T_d \supseteq X, T_{d \in D}} P(X, T_d) \quad (1)$$

Definition 3: Given a dataset D, the *minimum expected support threshold* τ is a predefined percentage of $|D|$; correspondingly, the *minimum expected support value* (*minExpSV*) is defined by

$$minExpSV = |D| \times \tau \quad (2)$$

an itemset X is called a utility itemset if its *expected support value* is not less than the value *minExpSV*.

Definition 4: The *minimum support threshold* τ is a predefined percentage of $|D|$; correspondingly, the *minimum support value* (*minSV*) in a dataset D is defined by

$$minSV = |D| \times \tau \quad (3)$$

Definition 5: The fuzzy system mining high utility itemset and itemsets utility is equal to quantity value multiplied by profit. For example, item A occurs in different transaction T1 and T5. If the fuzzy set of (A, T1) = {1/L, 0/M, 1/H} and (A, T5) = {0/L, 0/M, 1/H}. If t heir item utilities are obtain by multiplying fuzzy quantity with profit, item utilities of T1 and T5 will be same even though T5 defer a higher quantity value. The fuzzy utility is defined as,

$$fu(i_p, T_q) - f(i_p, T_q) \times S(i_p) \quad (4)$$

Definition 6: Fuzzy quantity The equation to find the fuzzy quantity value of item i_p in transaction T_q is denoted as $f(i_p, T_q)$ which is defined as,

$$fu(i_p, T_q) - \sum fq(i_p, T_q) \times weight(j) \quad (5)$$

Where fq (i_p, Tq, j) is the fuzzy value of fuzzy region j and weight (j) is a variable parameter defined by the fuzzy region. If a fuzzy region is

low then the weight should be low when compared to the region middle and high. Particular weights are assigned for area low, medium and high.

Definition 7: Fuzzy transaction utility can be defined as the sum of the fuzzy utilities of item occurring in the transaction. The equation denoting fuzzy transaction utility is

$$ftu(T_q)-)- \sum_{i_p \epsilon\, T_q} fu(i_{p,} T_q) \quad (6)$$

Definition 8: Fuzzy transaction weighted utility it is the sum of fuzzy transaction utilities of item occurring in the particular transaction for an item. The equation denoting fuzzy transaction weighted utility can be defined as,

$$ftwu(x)- \sum_{\text{xc}T_q \epsilon\, D} ftu(T_q) \quad (7)$$

4. Proposed Approach

The Proposed novel method namely FTNT (Fuzzy TNT Tree based high utility itemsets) mine the high utility itemsets which can return the quantity information and also profit information. FTNT-Mine algorithm is intended for mining fuzzy high utility itemset by applying fuzzy theory to high utility itemset based on fuzzy membership function is defined to represent the quantities in fuzzy sets. The transaction table is transformed into fuzzy transaction table. Then fuzzy utility will be calculated from the fuzzy transaction table and original utility table. After, Fuzzy transaction weighted utility is formed to discover the high utility itemset. The proposed algorithm TNT-Mine mainly consists of two actions: (1) creating an Array based Tail node Tree; (2) Mining high utility itemsets from the Tail node Tree.

4.1 Fuzzy Transaction Weighted Utilization Itemset

Fuzzy membership function used to find the fuzzy quantity item which is shown in figure 1. It can transform a quantity value in to fuzzy membership region and membership value thereby improve transaction database as fuzzy transaction database. The input transaction dataset is shown in table 1.

The quantity attribute for fuzzy membership function has three fuzzy regions namely low, middle and high. Thus, fuzzy membership value for the purchased quantity is represented as fuzzy set in terms of (fuzzy value of low/low, fuzzy value of middle/middle, and fuzzy value of high/high). For example, the quantity value '9' is converted in to fuzzy set as {0.0/L, 0.4/M, 0.6/H}, where 'L', 'M' and 'H' are acronym of 'Low', 'Middle' and 'High'.

Table 1. Transaction dataset

	A	B	C	D	E
T01	0	3	6	1	4
T02	3	0	10	0	9
T03	7	0	4	0	0
T04	6	1	0	1	0
T05	0	0	8	0	3
T06	0	2	12	1	0
T07	9	0	0	0	7
T08	2	2	0	0	6

To find the only representing fuzzy quantity from three fuzzy regions, a maximum value is generated from the fuzzy set. Below example, the only representation for quantity value '9' is represented as Max (0.0, 0.4, 0.6) =0.6. The sole representation for the quantity value was done according to equation (2). Weight parameter was fixed for the three region as Low=0.1, Middle=0.5, High=1. According to definition (7), sole representation for quantity '9' is 0. 6. The estimate is preceded as follow. The max value in fuzzy set is 0.6 which is there in region high. The max value is multiplied with weight assign for region high to give up the only value for particular quantity. So this process is adopted for all quantity in transaction dataset to copy fuzzy transaction database which is shown in table 2.

Table 2. Fuzzy Transaction dataset

	A	B	C	D	E
T01	0	0.06	0.5	0.1	0.3
T02	0.06	0	0.8	0	0.6
T03	0.4	0	0.3	0	0
T04	0.5	0.1	0	0.1	0
T05	0	0	0.3	0	0.06
T06	0	0.08	0.9	0.1	0
T07	0.6	0	0	0	0.4
T08	0.08	0.08	0	0	0.5

Table 3: Utility Table

Item	A	B	C	D	E
Utility	5	11	3	20	4

After the transformation of fuzzy transaction dataset from input transaction dataset, the next step is to find the fuzzy utility for each and every transaction according to above equation by multiply fuzzy quantity with profit value in utility table shown in table 3. Fuzzy utility for the item 'B' in transaction T01 is defined as the quantity value of item 'B' is 0.06 which is multiplied by the profit value for 'B' in the utility table which is 11 to yield the fuzzy utility as 0.66. Fuzzy utility values for the corresponding transaction dataset are displayed in table 4.Fuzzy utility calculated and find the fuzzy transaction utility according to definition (8) for each transaction. Fuzzy transaction utility table is displayed in table 5

Table 4. Fuzzy Utility table

	A	B	C	D	E
T01	0	0.06	1.5	2	1.2
T02	0.3	0	2.4	0	2.4
T03	2	0	0.9	0	0
T04	2.5	1.1	0	2	0
T05	0	0	0.9	0	0.24
T06	0	0.88	2.7	2	0
T07	3	0	0	0	1.6
T08	0.4	0.88	0	0	2

Table 5: Fuzzy transaction utility

Transaction ID	Transaction utility
T01	5.36
T02	5.1
T03	2.9
T04	5.6
T05	1.14
T06	5.58
T07	4.6
T08	3.28

Fuzzy transaction weighted utility itemset found from fuzzy utility. Finally FTWU for item 'A' has been evaluated based on user defined threshold value.

4.2 Tail Node Tree Creation

The proposed algorithm fuzzy tail node based mining the high utility itemsets mainly consists of two process: first step creating a Tree. (2) Step two mining high utility itemsets from the global Tree. The structure of F tail node tree is designed to efficiently store the transaction information on tail nodes. Global tree constructed by two scans of fuzzy transaction weight dataset. In the first scan, a global header table is created to maintain sorted high utility items. A global Tail node tree is the first maintains itemset information of the whole weight transaction dataset. The proposed algorithm construction is described as follows:

Create Tree (D,τ)
INPUT: An uncertain dataset D consisting of n transaction itemsets and a predefined user minimum expected support thresholdτ.
OUTPUT: A global Tail Node-Tree T.
Step 1: Calculate the minimum expected support value *minExpSV*, i.e. $minExpSV = |D| \times \tau$; count the expected support value and support value of each transaction item by one scan of dataset.

a) Fuzzy transaction utility can be defined as the sum of the fuzzy utilities of item occurring in the transaction.
b) Fuzzy transaction weighted utility it is the sum of fuzzy transaction utilities of item occurring in the particular transaction for an item.

Step 2: place those items whose expected support value are not less than *minExpSV* to a global header table, and sort the items in the header table according to the descending order of their support value.
Step 3: Initially set the root node of the Tree T as null.
Step 4: Remove the items that are not in the header table from each transaction itemset, and sort the remaining items of each transaction itemset according to the order of the global header table, and get a sorted itemset A.
Step 5: If the length of itemset A is 0, process the next transaction itemset; otherwise insert the itemset A into the tail node tree T by the following steps:
a) Store the weight probability value of each item in itemset A sequentially to a list; save the list to an array (which is denoted as ***ProArr***); the equivalent series number of the list in the array is denoted as *ID*.

b) If there has not been a tail node for the itemset *X*, create a tail node *N* for this itemset, where fuzzy*_N.Tail_info.len* is the length of itemset *A*, and fuzzy*_N.Tail_info.Arr_ind*= (*ID*); otherwise, append the sequence number *ID* to *fuzzy_N.Tail_info.Arr_ind*.

Step 6: Process the next transaction itemset based on above steps until meet all the transaction from fuzzy transaction weight dataset.

4.3 Mining high utility itemsets from a global tail node tree

After a tail node tree is constructed, the algorithm Fuzzy tail node tree can directly mine high utility itemsets from the tree without additional scan of dataset. The details of the mining techniques are described below. The algorithm FTN-Mine is similar to the algorithm UP growth: it creates and processes sub trees recursively.

Mining high utility (*T*, G*H, minExpSV***)**

INPUT: An FTNT-Tree *T*, a global header table *H*, and a Minimum expected support value *minExpSV*.

OUTPUT: The high utility itemsets (HUIs).

Step 1: find the high utility items from header table one by one from the last item by the following steps.

Step 2: choose the current *base-itemset* (which is initialized as null); each new *base-itemset* is a high utility itemset.

Step 3: Example consider base item is "Z" Let *Z.links* in the header table *H* contain *k* nodes whose item name is *Z*; we denote these *k* nodes as *N*1, *N*2, …, *Nk*; because item *Z* is the last one in the header table, all these *k* nodes are *tail nodes*, i.e., each of these nodes contains a *Fuzzy_Tail_info*.The following sub steps.

a) Create a sub header table *subH* by scanning the *k base item* branches from these *k* nodes to the root.

b) Suppose the sub header table is zero, go to Step 4.

c): Create sub Tree each and every base item *subTree* = **CreateSubTree** (*Z.link*, *subH*).

Mining (*subTree*, *subH*, *minExpSV*).

Step 4: After find the high utility of that base item *Z* and remove that item from header table.

Step 5: For each of these *k* nodes denote as *ni*≤*N*≤*k*), modify its *fuzzy_Tail_info* by the following sub steps:

a) Alter *fuzzy_Ni.Tail_info.len* values: *fuzzy_Ni.Tail_info.len = fuzzy_Ni.Tail_info.len* - 1.

b) Move *fuzzy_Ni.Tail_info* to the parent of node *Ni*.

Step 6: Process the next item of the header table *H*.

Subprograms: **CreateSubTree** (*all_baseitem_link, subH*)

INPUT: A list *link* which records tree nodes with the same item name, and a header table *subH*.

OUTPUT: A high utility itemsets from *subT*.

Step 1: Initially set the root node of the tree *subT* as zero.

Step 2: Process each node N_i in the list *link* by the following steps.

Step 3: Get the *fuzzy tail-node-itemset* of node N_i from X item sets.

Step 4: Remove those items that are not in the header table *subH* from itemset *X*, and sort the remaining items in itemset *X* according to the order of the header table *subH*.

Step 5: If the length of the sorted itemset k_i is 0, process the next node of the list *link*; otherwise insert the sorted itemset *X* into the FT-Tree *subT* by the following sub steps:

a) Get the original sequential *ID* of each item of the itemset *X* in the corresponding list of *ProArr*: *item_ind* = {*d*1, *d*2, *d*k} where *k* is the length of itemset *X*

b) Make a copy of *fuzzy_N.Tail_info*; denote the copy as *nTail_info*.

b): Alter *nTail_info* as the following:

(1) *nTail_info.len = k*.

(2) *nTail_info. Item_ind = item_ind*.

(3) if *nTail_info.bp* is null, set *nTail_info.bp*[*j*] to be the probability of item *Z*, i.e. *ProArr*[*nTail_info.Arr_ind*[*j*]]; otherwise, set *nTail_info.bp*[*j*] to be the product of *nTail_info.bp*[*j*] and the probability of item *Z* (1 ≤ *j* ≤ *fuzzy_bp.size;* the array *ProArr* is created when the global tree is created.

5. Experimental Results

The performance of the proposed algorithm FTNT-Mine. UP Growth is the state-of-the-art algorithm employing the pattern-growth approach and FTNT is a new proposed algorithm. So compare FTNT-Mine with the algorithms FUF-Growth, UP-Mine and Fuzzy based TNT on both types of datasets: Sparse

transaction datasets and dense transaction datasets. All algorithms were written in java programming language. The configuration of the testing platform is as follows: Windows 7 32bit operating system, 4G Memory, Intel(R) Dual-Core CPU @ 2.60 GHz.

Table 6. Dataset Characteristics

Dataset	*|D|*	T	|I|	Type
T10I4d100k	300,000	33.8	1000	sparse
mushroom	8,124	23	119	dense

Table 6 shows the characteristics of 4 datasets used in our experiments. '|D|' represents the total number of transactions; '|I|' represents the total number of distinct items; 'I' represents the mean length of all transaction itemsets; 'SD' represents the degree of sparsely or density. The synthetic dataset *T10I4d100ks* came from the IBM Data Generator and the datasets and *mushroom* were obtained from FIMI Repository. These four datasets originally do not provide probability values for each item of each.

Table 7. Comparison algorithm of using *T10I4d100k* Vs No tree created.

Algorithms	**No of Tree**					
	0.04	**0.05**	**0.06**	**0.07**	**0.08**	**0.09**
Fuzzy Tail Node Tree	170	62	29	10	6	4
UF-Growth +	369	115	39	17	8	4
UP Growth	410	232	101	52	43	21

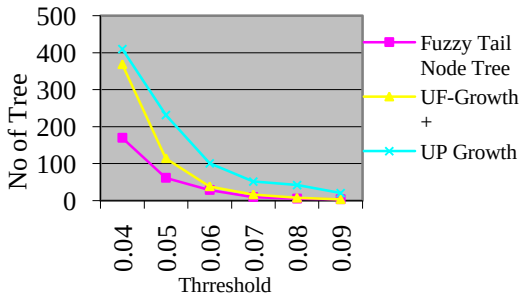

Fig. 1. Comparison of different threshold value with tree creation

Fig 1 and Table 7 Show the total number of tree nodes generated by FTNT-Mine, UF-Growth and UP, respectively, on the synthetic datasets.

Table 8. High Utility value using *T10I4d100k*

Algorithms	High Utility Itemsets					
	0.04	**0.05**	**0.06**	**0.07**	**0.08**	**0.09**
Fuzzy Tail Node Tree	73	44	32	23	17	11
UF-Growth +	62	33	28	19	14	8
UP Growth	48	26	22	15	9	2

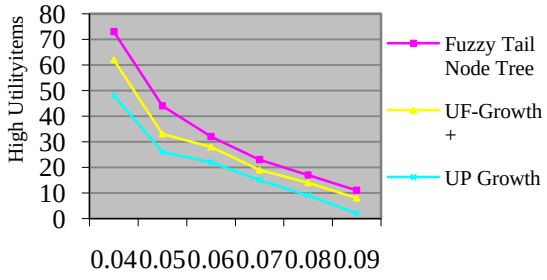

Fig. 2. Comparison of different threshold value with high utility items

Table 9. Total run time using *T10I4d100k*

Algorithms	Time					
	0.04	**0.05**	**0.06**	**0.07**	**0.08**	**0.09**
Fuzzy Tail Node Tree	11922	10209	8362	7952	7118	17897
UF-Growth +	36976	27145	23114	20317	18028	6949
UP Growth	46676	371545	33614	18317	16026	5979

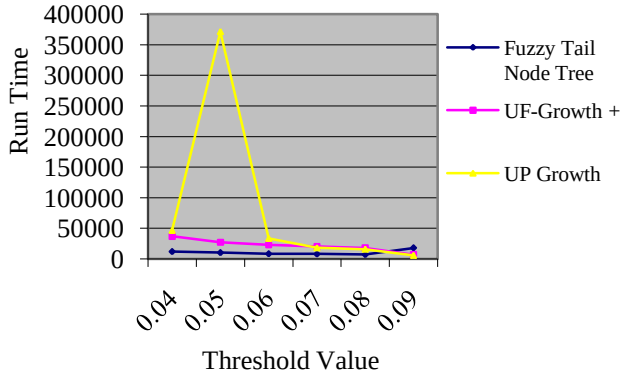

Fig. 3. Run time for Different Threshold Value

Table 10. Comparison of different algorithm using *mushroom*

algorithms	No of Tree					
	0.04	**0.05**	**0.06**	**0.07**	**0.08**	**0.09**
Fuzzy Tail Node Tree	334	254	150	123	103	83
UF-Growth +	443	345	234	189	132	112
UP Growth	546	434	367	264	212	187

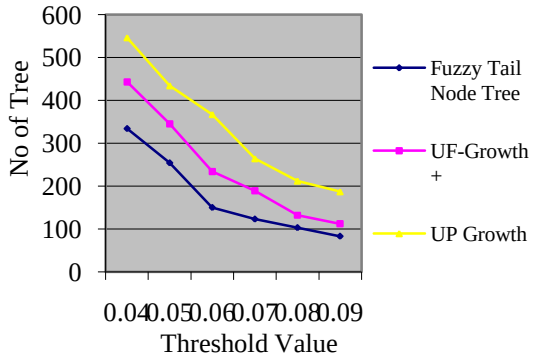

Fig. 4. Comparison of different threshold value with tree creation

Fig 4 and Table 10 Show the total number of tree nodes generated by FTNT-Mine, UF-Growth and UP, respectively, on the synthetic datasets.

Table 11. High Utility value using *mushroom*

Algorithms	High Utility itemsets					
	0.0	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	4	**5**	**6**	**7**	**8**	**9**
Fuzzy Tail Node Tree	78	46	34	27	19	13
UF-Growth +	67	38	29	21	17	10
UP Growth	49	31	28	19	10	8

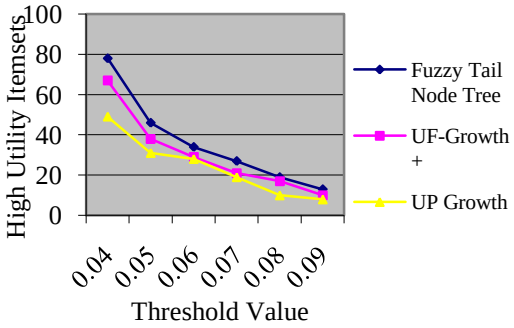

Fig. 5. Comparison of different threshold value with high utility items

Table 12. Total run time using *mushroom*

Algorithms	Run Time					
	0.04	**0.05**	**0.06**	**0.07**	**0.08**	**0.09**
Fuzzy Tail Node Tree	14622	12249	9372	8982	7228	6327
UF-Growth +	32956	28155	24164	21327	19048	7349
UP Growth	56576	381845	34684	19367	17126	6179

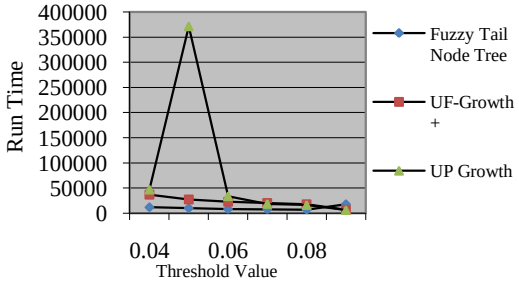

Fig. 6. Run time for Different Threshold Value

6. Conclusion and Future Enhancement

The proposed system reduces the memory space and running time for mining high utility itemset. The fuzzy based tail node tree maintains the utility information about all transactions, in order to mine high utility itemsets without generating high candidate itemsets. The proposed method executed on both sparse and dense dataset. In comparison the result shows that FTTN provides better performance, and their time and memory capacity is stable on both dense and sparse datasets along with lower minimum support value.

References

1. AlokChoudhary, Ying Liu, Wei-keng Liao, (2005), "A Fast High Utility Itemsets Mining Algorithm", UBDM'05 Chicago, USA

2. Ashish Mangalampalli, Vikram Pudi ,"Fuzzy Association Rule Mining Algorithm for Fast and Efficient Performance on Very Large Datasets", IEEE International Conference on Fuzzy Systems (FUZZ - IEEE), Jeju Island, Korea, 2009

3. Bay VO, Huy N guyen, Bac Le, "Mining High Utility Itemset from Vertical Distributed Database", Computing and Communication Technologies, 2009. RIVF '09.

4. H. Yao, H.J. Hamilton, 'Mining itemset utilities from transaction databases', Data & Knowledge Engineering 59 (3), pp..603 – 626, 2006.

5. Jyothi Pillai, O.P. Vyas and Maybin K. Muyeba "A Fuzzy Algorithm for Mining High Utility Rare Itemsets –FHUR" Int. J. on Recent Trends in Engineering and Technology, Vol. 10, No. 1, Jan 2014.

6. Karthikeyan T, Samuel Chellathurai A and Praburaj B, "A study on a novel method of mining fuzzy association using fuzzy correlation analysis", African Journal of Mathematics and Computer Science Research Vol. 5(2), pp. 28-33, 15 January, 2012.

7. Ming-Yen Lin, Tzer-Fu Tu, Sue-chen Hs ueh, "High utility pattern mining using the maximal itemset property and lexicographic tree structures", Science Direct, journal of information sciences 215(2012)1-14.

8. R. Agrawal, R. Srikant, "Fast algorithms for mining association rules", in: Proceedings of 20th International Conference on Very Large Databases, Santiago, Chile, 1994, pp. 487–499.

9. R. Agrawal, T. Imielinskl, A. Swami, "Mining Association rules between sets of items in large databases", Proceedings of the 1993 ACM SIGMOD International Conference in Management of Data, Washington, DC , 1993, pp. 207–216.

10. Sadak Murali, Kolla Morarjee, "A Novel Mining Algorithm for High Utility Itemsets from Transactional Databases", Global Journal of Computer Science and Technology Software & Data Engineering, Volume 13, Issue 11, Version 1.0, Year 2013, pp 1 - 7.

11. Sandeep Kumar Singh, Mr.Ganesh Wayal, Mr.Niresh sharma, "A Review: Data Mining with Fuzzy Association Rule Mining", International Journal of Engineering Research & Technology (IJERT) Vol. 1 Issue 5, July–2012.

12. Tzung-Pei Hong, Kuei-Ying Lin, Shyue-Liang Wang, " Fuzzy data mining for interesting gene realized association rules", Elsevier on fuzzy sets and system 138(2003) 255-269.

13. Luca Cagliero and Paolo Garza, "Infrequent Weighted Itemset Mining using Frequent Pattern Growth",IEEE Transactions on Knowledge and Data Engineering, 2014.

14. Vincent S. Tseng, Bai-En Shie, Cheng-Wei Wu, and Philip S. Yu "Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases", IEEE Transactions on Knowledge and Data Engineering, Vol. 25, NO. 8, August 2013.