An Efficient IFP - Tree Based High Utility Pattern Mining of Itemsets with Indexing

Suvitha K, Dhivya S, Ragini A, Preethi P, Yogapriya J

Abstract ***Conventional methods of Association rule mining and Frequent Itemset Mining (FIM) cannot satisfy the anxieties emerging from certain real applications. In real world, for making some decisions the user wants to know the total profit grossed by an itemset or item. To evaluate this it needs to take into account the quantity of the purchased item. The profit of an item considers the gain of single item and the number of item purchased. To address these, utility mining has been introduced. In this the utility of an itemset is calculated as the number of item purchased and the product of the gain of the item. Utility mining concentrates on both the reputation of an item in the knowledge base (i.e.) profit or exterior utility and the reputation of an item in the transaction (i.e.) quantity or interior utility of an item.****In this study, a novel Improved frequent-pattern tree (IFP-tree) structure, which is an extended prefix-tree structure for storing crucial information about frequent patterns, and develop an efficient IFP-tree-based mining method based on the generation of conditional utility pattern base which leads to the conditional utility IFP-tree for mining the complete set of frequent patterns. As the itemset in the sanitized database and original database are segmented in different areas, the itemset kept in the different areas are indexed through candidate keys for increasing the access speed and fast retrieval of data. This process will increase the accuracy of the database and it preserves the sensitive data items for a longer time.The assessment report shows that the generation of less candidate patterns makes algorithms to run faster.*

Keywords: Utility Mining, High-utility itemsets, Rare itemsets, Frequent Itemset mining, Improved Frequent Pattern

I. INTRODUCTION

Data mining can be characterized as a movement that concentrates some learning contained in expansive transaction databases. Information mining, the extraction of concealed prescient data from substantial databases, is an intense new innovation with awesome potential to help organizations concentrate on the most vital data in their information distribution centers.

Learning Discovery in Databases (KDD) is the non-unimportant procedure of recognizing legitimate, already obscure and conceivably helpful examples in information. These examples are utilized to make forecasts or characterizations about new information, clarify existing information, outline the substance of a huge database to bolster basic leadership and give graphical information perception to help people in finding further examples.

The limitations of frequent itemset mining motivated researchers to conceive a utility based mining approach, which allows a user to conveniently express his or her perspectives concerning the usefulness of itemsets as utility values and then find itemsets with high utility values higher than a threshold. In utility based mining the term utility refers to the quantitative representation of user preference i.e. according to an itemsets utility value is the measurement of the importance of that itemset in the user's perspective. The traditional ARM approaches consider the utility of the items by its presence in the transaction set. The frequency of itemset is not sufficient to reflect the actual utility of an itemset. For example, the sales manager may not be interested in frequent itemsets that do not generate significant profit. Recently, one of the most challenging data mining tasks is the mining of high iutility iitemsets iefficiently. iIn iview iof ithis, iutility imining iemerges ias ian iimportant itopic iin i data mining ifor idiscovering ithe iitemsets iwith ihigh iutility ilike iprofits. iIdentification iof ithe iitemsets iwith ihigh iutilities iis icalled ias iUtility iMining. iThe iutility ican ibe imeasured iin iterms iof icost, iquantity, iprofit iand iuser i preference

.for ithis iUtility imining imodel iwas iproposed ito idefine ithe iutility iof iitemset. iIn i[1] ithis imodel iby iconsidering iu(X) ias ia iutility iof ian iitemset iX, iwhich iis ithe isum iof ithe iall iutilities iof iitemset iX iin iall ithe itransactions icontainingiX. ithen ian iitemset iX iis icalled ia ihigh iutility iitems iif iits iutility igreater ior iequal ito iuser- idefined iminimum iutility ithreshresold.

Association irules iare iif/then istatements ithat iaid iuncover irelationships ibetween iseemingly idissimilar idata iin ia irelational idatabase ior iother iinformation irepository i[1, i2]. iAn iexample iof ian iassociation irule iwould ibe i"If a customer buys one dozen eggs, he is 80% likely to also purchase milk." An association rule has two parts, an antecedent (if) and a consequent (then). An antecedent is an item found in the data. A consequent is an item that is found in combination with the antecedent. Association rules are shaped by analyzing data for frequent if/then patterns and consuming the criteria support and confidence to classify the most important relationships. Provision is an indication of how frequently the items appear in the database. Confidence specifies the number of times the if/then statements have been found to be true. The problem of association rule mining is defined as:

Let I = {i1, i2,.....,in} be a set of n binary attributes called items. Let D = {t1, t2,....., tm} be a set of transactions called the database. Each transaction in D has a unique transaction ID and contains a subset of the items in I. A rule is defined as an inference of the form X==>Y where X & Y ∑ 1 and X ∩ Y = θ. The sets of items (for short itemsets) X and Y are called antecedent (left-hand-side or LHS) and consequential (right-hand-side or RHS) of the rule respectively. Example: The set of items is I = {milk, bread, butter, beer}. An example rule for the supermarket could be {butter, bread} ==> {milk} meaning that if butter and bread are bought, customers also buy milk[7]. In data mining, association rules are valuable for analyzing and predicting customer behavior. They play an vital part in shopping basket data analysis, product clustering, catalog design and store layout[7]. Programmers usage association rules to form programs capable of machine learning. Machine learning is a type of artificial intelligence (AI) that seeks to build programs with the ability to become more efficient without being explicitly programmed. In overall, association rule mining can be viewed as a two-step process: (i) Find all Improved frequent patterns and (ii) Generate strong association rules from the frequent patterns [2]. In (i), we can use some mining algorithms like Apriori, DHP, ECLAT, IFP Growth etc. that we discussed later. Also we proposed newer algorithm for frequent pattern mining. In (ii), all frequent pattern rules are checked for minimum support and minimum confidence to generate association rules.

Example: An itemset is called a high utility itemset if its utility is no less than a user- specified minimum utility threshold or low- utility itemset represented by min-util.

Table 1: Sample Database

TID	Transaction	TU
T1	(A,1)(C,10)(D,1)	17
T2	(A,2)(C,6)(E,2)(G,5)	27
T3	(A,2)(B,2)(D,6)(E,2)(F,1)	37
T4	(B,4)(C,13)(D,3)(E,1)	30
T5	(B,2)(C,4) (E,1)(G,2)	13
T6	(A,1)(B,1) (C,1)(D,1)(H,2)	12

Table 2: Profit Table

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

From table 1 and 2,

Utility ({A, T1}) =5×1=5

Utility ({AD,T1})=u({A,T1})+u({D,T1})=5+2=7

Utility ({AD}) =u ({AD, T1}) +u ({AD, T3}) =7+17=24

Utility ({BD}) =u ({BD, T3}) +u ({BD, T4}) =16+18=34

This paper is organized as follows. In Section II, we describe the related works of this system. Section III describes the problem identified. Section IV presents the design of our solution for Tree Based High Utility Pattern Approach. The evaluation results of both approaches are discussed in Section V. Finally, we discuss the conclusion and possible enhancement of future works related to the presented solution as section VI.

II. RELATED WORKS

In [2] Vincent S Tseng, Bai-En Shie, Cheng-Wu, Philip Sproposed a Single-pass incremental and intuitive digging for finding weighted frequent examples. The current weighted frequent example (WFP) digging can't be connected for incremental and intelligent WFP digging furthermore for stream information mining since they depend on a static database and its require various database examines. To defeat this, they proposed two novel tree structures IWFPTWA (Incremental WFP tree in light of weight rising request) and IWFPTFD (Incremental WFP tree in view of plunging request) and two new calculations IWFPWA and IWFPFD for incremental and intuitive mining utilizing a solitary database filter. IWFPFD promises that any non-applicantthing can't show up before competitor things in any branch of IWFPTFD and in this manner accelerates the prefix tree. The downside of this approach is that extensive memory space, tedious and it is exceptionally hard to bolster the calculation for bigger databases.

In [3], presented a novel utility IFP- tree, an extensive tree structure for storing essential information about frequent patterns for mining the high utility itemsets. We have utilized the standard IFP growth algorithm for mining the complete set of frequent patterns by pattern growth. The efficiency of the high utility pattern mining is recognized by two important thoughts. One is the construction of the utility IFP-tree and the other one is the mining of utility itemsets from the utility IFP- tree. Our proposed utility IFP-tree-based pattern mining utilized the pattern growth method to avoid the costly generation of a large number of candidate sets in which it dramatically reduces the search space. The experimentation was carried out on our proposed approach using real life datasets and the results showed that the proposed approach is effective on the tested databases.

Frequency - weighted utility *FWU* of an item *ip*, denoted by *FWU* (*ip*), is computed using the Transaction Frequency (*TF*), Transaction Weightage (TW) and the External Utility (*Eu*).

$$FWU\ (ip) = \frac{TF\ (ip)*TW\ (ip)*\ Eu\ (ip)}{UF}$$

An operation in object-oriented databases gives rise to the processing of a path. Several database operations may result into the same path. Choenni et al., [4] address the problem of optimal index configuration for a single path. As it is shown an optimal index configuration for a path can be achieved by splitting the path into subpaths and by indexing each subpath with the optimal index organization. The authors present an algorithm which is able to select an optimal index configuration for a given path. The authors consider a limited number of existing indexing techniques (simple index, inherited index, nested inherited index, multi-index, and multi-inherited index) but the principles of the algorithm remain the same adding more indexing techniques.

Kriegel et al., [5] presented a technique to achieve efficient query processing on data-partitioning index structures within general purpose database systems. The navigational index traversal cost is reduced by using "extended index range scans". If a directory node is "largely" covered by the actual query, the recursive tree

traversal for this node can beneficially be replaced by a scan on the leaf level of the index instead of navigating through the directory any longer. On the other hand, for highly selective queries, the index is used as usual. The author demonstrated the benefits of this idea for spatial collision queries on the relational R-tree. Energy saving is one of the most important issues in wireless mobile computing. Among others, one viable approach to achieving energy saving is to use an indexed data organization to broadcast data over wireless channels to mobile units. Using indexed broadcasting, mobile units can be guided to the data of interest efficiently and only need to be actively listening to the broadcasting channel when the relevant information is present.

III. PROBLEM DESCRIPTION

The problem of mining utility itemsets is discussed and some basic definitions are described in this subsection. Let I = {i1, i2 ,..., im}be a set of items and D = {t1 , t2 ,..., tn } be a transaction database where the items of each transaction ti is a subset of I . The utility of item ip in transaction tq, denoted as U (ip,tq) is defined as Iu (ip,tq)× Eu (ip) . Let an itemset X be a subset of I. The utility of X in transaction tq, denoted by U(X, tq) is defined as U(X, tq) $\sum ip \in XU$ (ip,tq) . The task of high utility mining is to find all items that have utility above a user-specified min_utility. Since utility is not anti- monotone, the concept of Frequency Weighted Utility (FWU) is used to prune the search space of high utility itemsets.

The internal utility or local transaction utility value Iu(ip,tq) represents the quantity of item ip in transaction tq. The external utility Eu(ip) represents the unit profit value of item ip.

Utility U(ip,tq) is the quantitative measure of utility for item ip in transaction tq defined by U (ip,tq)=Iu(ip,tq)×Eu(ip).

The utility of an itemset X in transaction tq, U(X ,tq) , is defined by U (X ,tq)=$\sum ip \in XU(ip,tq)$; where X ={i1,i2,.....ik}is a k-itemset, X ⊆tq and 1 ≤k≤m.

IV. TREE BASED HIGH UTILITY PATTERN APPROACH.

In data mining, association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. An example rule for the supermarket dataset could be {butter, bread} {milk}.Association rule learning is about selecting the desired rules from the set of all possible rules with constraints on various measures of significance and interest. The best-known constraints are minimum thresholds on support and confidence. Let I = {i1, i2… in} be a set of n binary attributes called items. Let *D* = {t1, t2… tm} be a set of transactions called the database. Each transaction in *D* has a unique transaction ID and contains a subset of the items in *I*. Association rules are usually required to satisfy a user-specified minimum support and a user-specified minimum confidence at the same time.

Association rule generation is usually split up into two separate steps:

- First, minimum support is applied to find all frequent item set in a database.
- Second, these frequent item sets and the minimum confidence constraint are used to form rules.

There are a number of algorithms used to generate Association rules. The Association rule algorithms discussed in this thesis is: **Apriori algorithm**

Apriori is a classic algorithm for learning association rules. Apriori is designed to operate on databases containing transactions (for example, collections of items bought by customers, or details of a website frequentation). Other algorithms are designed for finding association rules in database having no transactions or having no timestamps. As is common in association rule mining, given a set of itemsets (for instance, sets of retail transactions, each listing individual items purchased),

the algorithm attempts to find subsets which are common to at least a minimum number C of the itemsets.

Apriori uses a "bottom up" approach, where frequent subsets are extended as one item at a time (a step known as candidate generation), and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found. The purpose of the Apriori Algorithm is to find associations between different sets of data. It is sometimes referred to as "Market Basket Analysis". Each set of data has a number of items and is called a transaction. The output of Apriori is set of rules that tell us the frequency of occurrence items in the dataset.

Eclat algorithm

Eclat algorithm finds the elements from the bottom like depth first search. Eclat algorithm is very simple algorithm to find the frequent item sets. This algorithm uses vertical database. It cannot use horizontal database. If there is any horizontal database, then we need to convert into vertical database. There is no need to scan the database again and again. Eclat algorithm scans the database only once. Support is counted in this algorithm. Confidence is not calculated in this algorithm.

The IFP-growth (Improved Frequent Pattern – growth) algorithm

The IFP-growth (Improved Frequent Pattern– growth) algorithm differs basically from the level-wise algorithms, that uses a “candidate generate and test” approach. It does not use candidates at all, but it compresses the database into the memory in a form of a so- called IFP-tree using a pruning technique. The patterns are discovered using a recursive pattern growth method by creating and processing conditional IFP-trees. The drawback of the algorithm is its huge memory requirement, which is dependent on the minimum support threshold and on the number and length of the transactions.

Step 1: Frequent Pattern (FP) Mining

Frequent pattern (FP) mining discovers patterns in transaction databases based only on the relative frequency of occurrence of items without considering their utility. For many real world applications, however, the utility of itemsets based on cost, profit or revenue is important. The utility mining problem is to find itemsets that have higher utility than a user specified minimum. Unlike itemset support in frequent pattern mining, itemset utility does not have the anti-monotone property and so efficient high utility mining poses a greater challenge.

The frequent pattern mining problem does not take into account the quantity or an associated weight such as price or profit of an item but it represents only the occurrence of each item in a transaction by a binary value. But, quantity and weight are important factors for solving real world decision problems that intends to maximize the utility of an organization. Hence, all itemsets that have utility value greater than a user specified minimum utility value are identified by high utility itemset mining. Both local transaction utility and external utility contribute to the utility of an item. Identifying high utility item sets which drive a major share of the overall utility is the objective of utility mining. High utility pattern mining approaches have been proposed to overcome this problem. As a result, it becomes a very important research issue in data mining and frequent pattern mining. In this research, we have presented an efficient approach for mining the high utility itemsets from the utility IFP-tree structure. The procedure used for mining high utility items involves the following important steps.

Step 2: Construction of Utility IFP-tree

In general, the construction of the IFP-tree and the mining patterns from the IFP- tree are the major important steps in the frequent pattern tree algorithm. Similarly, the proposed approach also contains these two steps, where the utility IFP-tree is constructed using the frequency weighted utility rather than the frequency value. In addition to this, the mining process utilizes pattern growth methodology, where the support is computed based on the frequency weighted utility rather than the frequency. In this section, we describe the construction process of our proposed utility IFP- tree structure based on the frequency weighted utility.

The proposed algorithm is explained with the help of a simple example for easier understanding the entire step including the tree construction and mining processes. Table 3 provides an example of a transaction database and Table 3.2 gives the unit profit for each item belonging to the transaction database.

Table 3: Example of a Transaction Database

Item / TID	A	B	C	D
01	2	1	0	1
02	3	0	2	0
03	0	3	2	0

Table 4: Example of a Utility Table

Item	Profit ($)
A	2
B	1
C	5
D	1

Example: Let us consider the items present in Table 3.1. The FWU of item 'A' is computed as follows: the TF (ip) of the item 'A' is 2; the TW (ip) is 5 and the profit records for that item is 2. Also, the total profit value, called utility factor is found out to be 9 in this case. Now, FWU (A) = 2.22, FWU (B) = 0.77, FWU (C) = 4.44 and FWU (*D*) = 0.11. In this case, we have taken the min_util value as 0.3 and chosen the items which have utility values greater than the min_util value. Based on these computed utility values, the items are re-ordered. The transactions with sorted items are utilized for illustrating the construction of the utility IFP-tree. The ordered transactions and Frequency Weighted Utility (FWU) are shown in Table 3 and Table 4 respectively.

Table 5: The ordered transactions with sorted large items

TID	Frequent Items	
01	A	B
02	C	A
03	C	B

Table 6: Frequency Weighted Utility (FWU)

Item	FWU
A	2.22
B	0.77
C	4.44
D	0.11

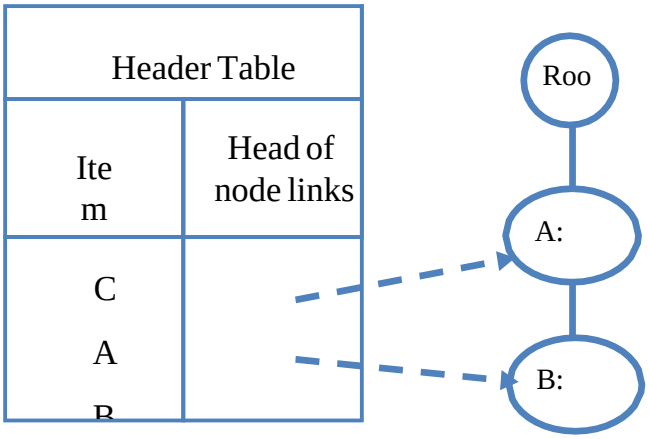

Figure 1: The Utility IFP-tree after the first transaction is processed

Subsequently, the next transaction containing frequent weighted utility items (C, A) is processed. Here, the items do not contain any prefix path in the utility IFP-tree after executing the first transaction, so the new nodes (C: 4.44) are attached to the root node as its child. Also, the other new node (A: 2.22) is created and linked with the child of (C: 4.44). The results after the second transaction are shown in Figure 2.

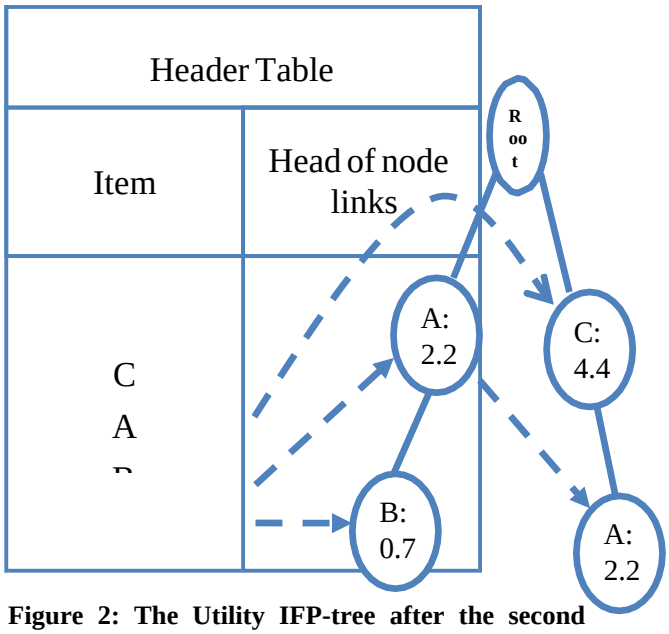

Figure 2: The Utility IFP-tree after the second transaction is processed

For processing the third transaction, the path <"C" "B"> shares the same prefix "C" with the Utility IFP-tree so the count of the node (C: 4.44) is incremented by 4.44 as it shares the common prefix and a newly created node (B: 0.77) is attached to the node (C: 8.88) as its child node. The results after the third transaction are shown in Figure 3

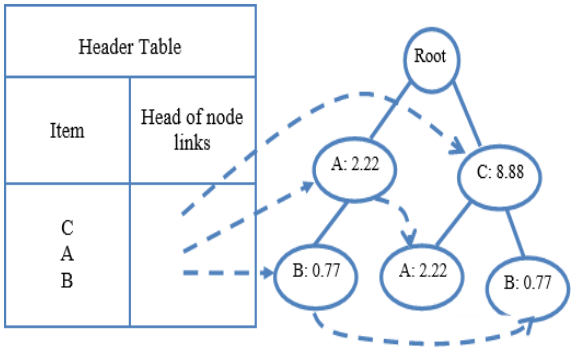

Figure 3: The utility IFP-tree after the third transaction is processed

After the Utility IFP-tree is constructed from a transaction database, a mining process is executed to determine the large items. Utility IFP-tree derives the utility itemsets directly from the utility IFP-tree and do not necessitate generation of candidate itemsets for mining. It recursively processes the utility items one by one and bottom-up with regard to the Header Table. By constructing a conditional utility IFP-tree for each utility item, high utility itemsets are mined recursively from it.

This process is executed until all the items in the utility IFP-tree get processed.

Step 3: Mining of High Utility itemsets from the Utility IFP-tree

The next major step is examining the mining process based on the constructed utility IFP-tree as shown in Figure 3.3. The mining process of utility itemsets from the utility IFP-tree based on the pattern growth methodology is explained as follows.

Generation of Conditional Utility Pattern Base and Conditional Utility IFP Tree,

After the utility IFP-tree is constructed from an ordered transaction database, the mining procedure starts with the generation of the conditional utility pattern base and the conditional utility IFP-tree. We have generated a conditional utility pattern base and the conditional pattern tree as the utility IFP-tree as shown in Figure 3.4.

Here, we start with the mining process from the bottom of the nodes of the utility IFP-tree and their corresponding prefix paths are extracted from it.

Then, their relevant utility pattern base and conditional utility IFP-tree are generated in order to mine 2-length utility patterns.

Example: At first, we process the item "B", which is the bottom item present in the header table so that two prefix paths that exist for item B are extracted. For an item B, the conditional pattern base is (A: 0.77) and (C: 0.77), which are the prefix paths of the item "B".

Then, the conditional utility IFP tree is generated for the item "B". Again, the conditional pattern base is generated for the superset of "A" i.e., "AB" and "AC" but no prefix paths exists for this sequence so it generates a NULL path. Subsequently, the next items A and C are processed. The conditional pattern base for item "A" is (C: 2.22) and the conditional pattern base for the pattern "C" is null.

The conditional pattern-bases and the conditional IFP-trees generated are summarized in Table 7.

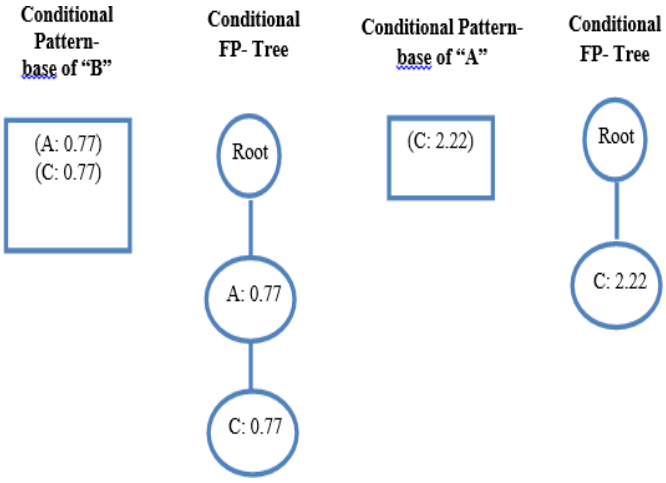

Figure 4: Mining of IFP-tree by creating conditional pattern-bases

Table 7: Mining frequent patterns by creating conditional pattern-bases

Item	Conditional pattern-base	Conditionl IFP-tree
B	{(A:0.77), (C:0.77))	{(A,C)}/B
A	{(C:2.22)}	{(C)}/A
C	Φ	Φ
BA	Φ	Φ
BC	Φ	Φ
AC	Φ	Φ

Mining of utility patterns

After the generation of the conditional utility IFP-tree, high utility patterns are mined from it, based on the minimum support threshold. Here, utility patterns are mined recursively from the conditional utility IFP tree so that all length patterns having frequency weighted utility greater than the minimum threshold are obtained. The patterns are said to be frequent weighted utility patterns if their support is greater than the *min_util*.

Example: The results obtained for the sample database given in Table 3.1 are shown in the Table 3.6. The frequent weighted utility patterns are {(C: 8.88) (A: 4.44), (B: 1.44), (AB: 0.77), (CB: 0.77) and (CA: 2.22)}.

Table 8: Frequent weighted utility patterns for a sample database

Frequent Patterns	
C: 8.88	CA: 2.22
A: 4.44	AB: 0.77
B: 1.44	CB: 0.77

Example 1: Let us consider an example of a simple database with 10 transactions and min_sup is 2 as shown

The sorted transactions are shown in following example Table 3. Then the scanned database is represented by BitTable shown in Table 4. Then Calculate the intersection of transactions that contain certain frequent items one by one.

Let us take frequent item I2 as example,

candidate

= T1 ∩ T7

= 1010111 ∩ 1010111

= 1010111

That is I2 subsume is I6 I1 I3 I5 i.e. (I2, I6 I1 I3 I5)

Continue this process accordingly for all items and finally the subsume index array is (I2, I6 I1 I3 I5), (I4, I1 I3), (I6, I1 I3 I5), (I7, I1 I3), (I1, I3), (I3, Ø), (I5, Ø).

Table 1 : Transactions Table

TID	Items
T1	I1,I2,I3,I5,I6,I15
T2	I1,I3,I7
T3	I5,I9
T4	I1,I3,I4,I5,I7
T5	I1,I3,I5,I7,I12
T6	I5,I10
T7	I1,I2,I3,I5,I6,I16
T8	I1,I3,I4
T9	I1,I3,I5,I7,I13
T10	I1,I3,I5,I7,I14

Table 2 : 1- Frequent Itemsets

Itemset	Sup Count
I2	2
I4	2
I6	2
I7	5
I1	8
I3	8
I5	8

Table 3 : Sorted Transactions Table

TID	Items	Ordered Items
T1	I1,I2,I3,I5,I6,I15	I2,I6,I1,I3,I5
T2	I1,I3,I7	I7,I1,I3
T3	I5,I9	I5
T4	I1,I3,I4,I5,I7	I4,I7,I1,I3,I5
T5	I1,I3,I5,I7,I12	I7,I1,I3,I5
T6	I5,I10	I5
T7	I1,I2,I3,I5,I6,I16	I2,I6,I1,I3,I5
T8	I1,I3,I4	I4,I1,I3
T9	I1,I3,I5,I7,I13	I7,I1,I3,I5
T10	I1,I3,I5,I7,I14	I7,I1,I3,I5

Table 4 : BitTable Representation of Database

TID	I2	I4	I6	I7	I1	I3	I5
T1	1	0	1	0	1	1	1
T2	0	0	0	1	1	1	0
T3	0	0	0	0	0	0	1
T4	0	1	0	1	1	1	1
T5	0	0	0	1	1	1	1
T6	0	0	0	0	0	0	1
T7	1	0	1	0	1	1	1
T8	0	1	0	0	1	1	0
T9	0	0	0	1	1	1	1
T10	0	0	0	1	1	1	1

Table 5 : Frequent Pattern Generation for Each Item						
ITEMS						
I2	**I4**	**I6**	**I7**	**I1**	**I3**	**I5**
I2 : 2	I4 : 2	I6 : 2	I7 : 5	I1 : 8	I3 : 8	
I2,I6 : 2	I4,I1 : 2	I6,I1 : 2	I7,I1 : 5	I1,I3 : 8	I3,I5 : 6	
I2,I1 : 2	I4,I3 : 2	I6,I3 : 2	I7,I3 : 5	I1,I5 i: i6		
I2,I3 : 2	I4,I1,I3 : 2	I6,I5 : 2	I7,I1,I3 : 5	I1,I3,I5 : 6		
I2,I5 : 2		I6,I1,I3 : 2	I7,I5 i: i4			
I2,I6,I1 : 2		I6,I1,I5 : 2	I7,I1,I5 : 4			
I2,I6,I3 : 2		I6,I3,I5 : 2	I7,I3,I5 : 4			
I2,I6,I5 : 2		I6,I1,I3,I5 : 2	I7,I1,I3,I5 : 4			I5 : 8
I2,I1,I3 : 2						
I2,I1,I5 : 2						
I2,I3,I5 : 2						
I2,I6,I1,I3 : 2						
I2,I6,I1,I5 : 2						
I2,I6,I3,I5 : 2						
I2,I1,I3,I5 : 2						
I2,I6,I1,I3,I5 : 2						
TOTAL = 43						
16	4	8	8	4	2	1

V. EXPERIMENTAL RESULTS AND PERFORMANCE EVALUATION

1. Nursery dataset

The experimental results are obtained for diverse support values on the Nursery dataset. The obtained results are plotted as graphs as shown in Figures 3.15, 3.16, 3.17, 3.18 and 3.19 that shows the performance of the four approaches on Nursery dataset in the effectual mining of high utility item sets. Here, the performance of our proposed approach is evaluated for different support values (Normalized between 0.1 and 0.5) and the corresponding generated length of patterns.

It is evident from the analysis of the plotted graphs, that our proposed approach produces better results than the standard Apriori, Eclat and IFP-growth algorithm.

As the support value varies, the number of generated frequent patterns gets reduced in our proposed approach than the Apriori, Eclat and IFP-growth algorithm by different length of patterns.

In Figure 5, the number of patterns generated by varying the support thresholds of 1 length patterns is constricted compared to the Apriori, Eclat and IFP-growth algorithm.

Likewise, Figures 6, 7 and 8 shows the generated number of patterns of length 2, 3 and 4 respectively for the different supports for the algorithms.

But, as shown in Figure 9, no 5 length patterns are produced by our proposed approach unlike the Apriori, Eclat and IFP- growth algorithm which generated a limited number of patterns.

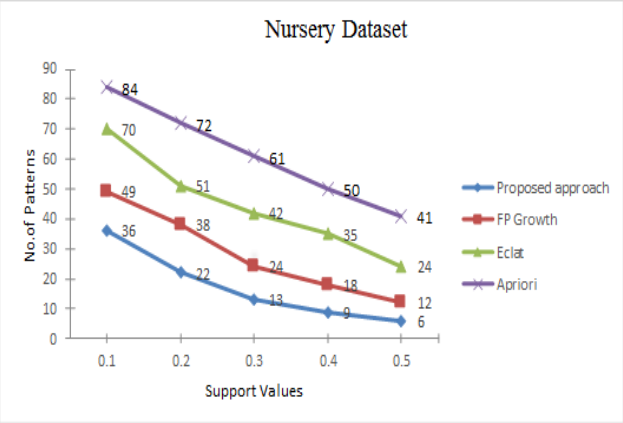

Figure 5: No. of frequent patterns (1- length) generated using various support thresholds

X axis - supporting threshold values range from 0.1 to 0.5

Y axis – Number of frequent patterns (1- length).

The number of patterns generated by varying the support thresholds of 1 length patterns is reduced compared to the Apriori, Eclat and IFP-growth algorithm.

Table 9: No. of frequent patterns (1-length) generated using various support thresholds

Support Value	No of patterns			
	Aprio	Ecl	IFP	Propos
0.1	84	70	49	36
0.2	72	51	38	22
0.3	61	42	24	13
0.4	50	35	18	9
0.5	41	24	12	6

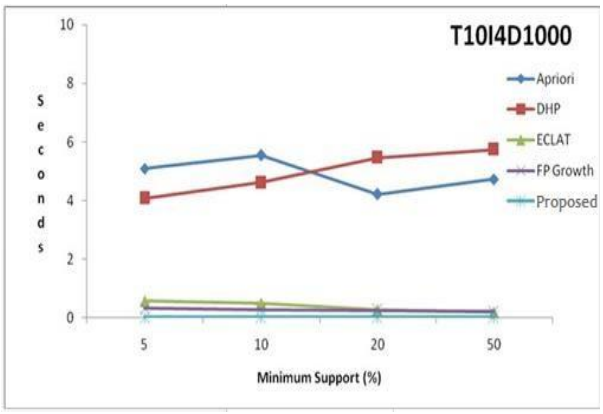

Figure 6: Execution time (in seconds) required by five different algorithms in T10I4D1000 dataset with different minimum support threshold.

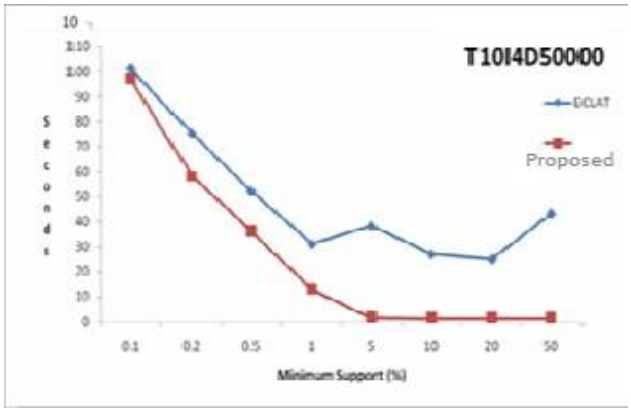

Figure 7: Execution time (in seconds) required by two different algorithms in T10I4D50000 dataset with different minimum support threshold.

VI: CONCLUSION

An approach for weighted utility itemsets mining from the database so as to improve the performance of data mining is proposed. And we have presented a utility IFP-tree by utilizing a tree structure for storing essential information about improved frequent patterns for mining high utility itemsets. We have utilized the standard Apriori, Eclat and IFP- growth algorithms for mining the complete set of frequent patterns by means of pattern growth.

Higher efficiency in mining high utility patterns is realized by implementing two important concepts. One is the construction of the utility IFP-tree and the other one are the mining of utility itemsets from the utility IFP-tree. Our proposed utility IFP-tree-based pattern mining utilizes the pattern growth method to avoid the costly generation of a large number of candidate sets and reduces the search space dramatically. Apart from that, in this paper we have used indexes which generates more frequent patterns directly. Comparing with Apriori, DHP, ECLAT and IFP Growth, the new algorithm reduces time for many frequent itemsets generation and candidate frequent itemsets which support count do not need to be computed. So the efficiency of the new algorithm is better than all of the above discussed algorithms. Performanceevaluation shows that new algorithm in performance is more remarkable for mining frequent patterns.

REFERENCES

1. Pranoti Meshram , Prof. Vikrant Chole, “Efficient Approach for Mining of High Utility Itemsets from Transactional Database”, IJCSMC, Vol. 4, Issue. 6, June 2015, pg.695 – 700.
2. Vincent S Tseng, Bai-En Shie, Cheng-Wu, Philip S, “Efficient algorithms for mining high utility itemsets from transactional databases”, IEEE Transactions on knowledge and data engineering, 2013.
3. C.Saravanabhavan , R.M.S.Parvathi, “Utility Fp-Tree: An Efficient Approach for Mining of Weighted Utility Itemsets”, International Journal of Engineering Research and Development, Volume 8, Issue 7 (September 2013), PP.19-31.

4. Choenni S, Blanken H, Chang T. Index selection in relational databases. InComputing and Information, 1993. Proceedings ICCI'93., Fifth International Conference on 1993 May 27 (pp. 491-496). IEEE.
5. Berchtold S, Keim DA, Kriegel HP. An index structure for high-dimensional data. Readings in multimedia computing and networking. 2001 Aug 10;451.
6. Adinarayanareddy B, Srinivasa Rao O & Krishna Prasad MHM 2012, 'An Improved UP-Growth High Utility Itemset Mining', International Journal of Computer Applications (0975 – 8887) vol.58, no.2.
7. Adnan M & Alhajj R 2009, 'DRFP-tree: disk resident frequent pattern tree', Appl Intell., vol.30, pp. 84-97.
8. Aggarwal CC & Yu PS 2005, 'On Variable Constraints in Privacy-Preserving Data Mining', SIAM Conference.
9. Aggarwal G, Feder T, Kenthapadi K, Khuller S, Motwani R, Panigrahy R, Thomas D & Zhu A 2006, 'Achieving Anonymity via Clustering, ACM PODS Conference.
10. Agrawal R & Srikant R 1994, 'Fast algorithms for mining association rules', in Proceedings ofthe 20th International Conference on Very Large Databases, Santiago, Chile, pp. 487-499.
11. Agrawal R & Srikant R 2000, 'Privacy- preserving data mining', In Proc. of ACM SIGMOD'00, Dallas, Texas, USA, pp. 439- 450.
12. Agrawal R, Imielinski T & Swami AN 1993, 'Mining association rules between sets of items in large databases', in Proceedings of the ACM SIGMOD International Conference on Management of Data, Washington, D.C., pp. 207-216.
13. Alaa Kh. Juma'a, Sufyan TF Al-Janabi & Nazar A Ali 2013, 'Hiding Sensitive Frequent Itemsets over Privacy Preserving Distributed Data Mining', Raf Journal of Comp &Math's, vol. 10, no. 1.
14. Alva Erwin, Raj P & Gopalan, Achuthan NR 2007, 'CTU-Mine: An Efficient High Utility Itemset Mining Algorithm Using the Pattern Growth Approach', in proceedings of the Seventh IEEE International Conference on Computer and Information Conference on Computer and Information Technology, Aizu-Wakamatsu, Fukushima, pp. 71-76.
15. Bac Le, Huy Nguyen, Tung Anh Cao & Bay Vo 2009, 'A Novel Algorithm for Mining High Utility Item sets', First Asian Conference on Intelligent Information and Database Systems, Dong Hoi, pp. 13 - 17.
16. Barber B & Hamilton HJ 2000, 'Algorithms for mining share frequent itemsets containing infrequent subsets', In: D. A. Zighed, H. J. Komorowski, J. M. Zytkow (eds.): 4th European Conf. on Principles of Data Mining and Knowledge Discovery. Lecture Notes in Computer Sciences, Springer-Verlag, Berlin Heidelberg New York, vol. 1910, pp. 316-324
17. Barber B & Hamilton HJ 2001, 'Parametric algorithm for mining share frequent itemsets', Journal of Intelligent Information Systems, vol.16, pp. 277-293.
18. Barber B & Hamilton HJ 2003, 'Extracting share frequent itemsets with infrequent subsets', Data Mining and Knowledge Discovery, vol.7, pp. 153-185.
19. Bay Vo, Huy Nguyen & Bac Le 2009, 'Mining High UtilityItemsets from Vertical Distributed Databases', in proceedings of the International Conference on Computing and Communication Technologies, Da Nang, pp. 1-4.
20. Christo Paul.E Preethi.P, Baskaran.G, An Eyes-Free Model Implementation: Voice Based Optical Character Recognition for Mobile Devices, International Journal of Advanced Research in Computer Science & Technology, Volume 2, Issue 1, March 2014.
21. Bertino E, Fovino IN & Provenza LP 2005, 'A framework for evaluating privacy preserving data mining algorithms', Data Mining and Knowledge Discovery, vol. 11, no. 2, pp.121-154.
22. Chapman P, Clinton J, Kerber R, Khabaza T, Reinartz T, Shearer C & Wirth R 2000, 'CRISP-DM 1.0: Step-by-step data mining guide', NCR Systems Engineering Copenhagen, USA and Denmark, Daimler Chrysler AG (Germany), SPSS Inc. (USA) and OHRA Verzekeringenen Bank Group B.V (The Netherlands).
23. Yogapriya, J., Saravanabhavan, C., Asokan, R., Vennila, I., Preethi, P., Nithya, B. 2018. A Study of Image Retrieval System Based on Feature Extraction, Selection, Classification and Similarity Measurements. Journal of Medical Imaging and Health Informatics, 8(3), 479-484.
24. D. Padmini Bai and P. Preethi , "Security Enhancement of Health Information Exchange Based on Cloud Computing System", International Journal of Scientific Engineering and Research, pp. 79-82, Volume 4 Issue 10, October 2016.
25. S.Sangeetha P.Dhivya Bharathy, P.Preethi, K.Karthick, Hand Gesture Recognition for Physical Impairment Peoples, SSRG International Journal of Computer Science and Engineering (SSRG-IJCSE), pp. 6-10, Volume 4 Issue 10, October 2017.
26. Preethi P, Asokan R, A High Secure Medical Image Storing and Sharing in Cloud Environment Using Hex Code Cryptography Method—Secure Genius, Journal of Medical Imaging and Health Informatics, Volume 9, Number 7 (September 2019) pp.1337 1546.