A Hybrid Approach for Mining High Utility Rare Itemsets Over Uncertain Database

1Shalini Zanzote Ninoria and 2S. S. Thakur
1Department of Mathematics & Computer Science, R. D. University, Jabalpur, India
2Department of Applied Mathematics, Jabalpur Engineering College, Jabalpur, India
E-mail: 1shalini.ninoria@gmail.com, 2samajh.singh@rediffmail.com

ABSTRACT

In present era, as there are many applications of uncertain data hence more emphasis has paying focus on mining itemsets over uncertain databases. Data mining is a technique which is useful in the extraction of interesting relationships between data in huge databases. Association rule mining is one of the most vital techniques of data mining in which association among the items present in the transactions are discovered. High-utility itemset mining (HUIM) has come up as a most significant research topic in data mining. High utility rare itemsets in a database can be used by retail stores to adapt their marketing strategies in order to increase their profits. Even though the itemsets mined are infrequent, since they generate a high profit for the store, marketing strategies can be used to increase the sales of these items. In this paper, a novel approach named A Hybrid Approach for Mining High Utility Rare Itemsets over Uncertain Database ((Improved High Utility Rare Itemset over Uncertain Database (IHURI) algorithm) is proposed to generate all high utility rare itemsets while keeping the algorithm time-efficient as well as memory-efficient. This proficient approach improved the concept of apriori inverse over uncertain database and it will give blend of Improved Apriori[1],apriori-inverse[2] and UHUI-apriori [3] algorithm approaches in the form of hybrid proposed approach. This paper will also give the new version or extension of the algorithm HURI proposed by Jyothi et al as it will give the improvement in the basic apriori algorithm for item generation as well as implementation of HURI over uncertain database. The implementation of an algorithm for the analysis is done on JDK 6.1 and referred the sample dataset presented by Lan Y.et al,2015[3] for uncertain database.

Keywords: Data Mining,Association Rule Mining,High Utility Itemset Mining,Rare Itemset Mining,Uncertain Database

1. INTRODUCTION

In the current decade the amount of database has increased tremendously. This rapid development has led major interest to develop various tools which can be used to handle the data as well as extract the desired information moreover knowledge. The term data mining or knowledge discovery in database has been implemented for a field of research dealing with the automatic discovery of hidden information or knowledge within the databases. The hidden information within databases, mainly the interesting association relationships among sets of objects that lead to association rules may reveal useful patterns for decision support, financial predict, marketing policies, even medical diagnosis and many other applications[4].Data mining is the method of determining interesting, meaningful, and understandable patterns hidden in large data sets[5]. Now a day’s organization collects numerous data and this data is stored in form of transaction[6]. In the area of business, corporate and customer data are becoming recognized as a tactical asset. The ability to extract valuable knowledge unseen in these data and to act on that knowledge is becoming increasingly important in today's competitive globe. The entire process of application of a computer based methodology, including new techniques, to find out knowledge from data is called data mining[1].Knowledge Discovery in Database (KDD) aims at finding meaningful and useful information in substantial amounts of data[7]. Data mining emerged in 1990s and has a big impact in business, industry, and science. Data mining has been used for many years by many fields such as businesses, scientists and governments, etc. Han and Kamber [8] presented, data mining functionalities include data characterization, data discrimination, association analysis, classification, clustering, outlier analysis, and data evolution analysis. Data mining is the procedure of applying these methods to data with the intention of discovering hidden patterns [9].

Two primary issues in KDD, having many applications in a variety of domains, are frequent itemset mining (FIM) and association rule mining (ARM) [10],[5],[7]. Agrawal and Srikant in 1993. Discovering associations between items is helpful to understand customer behavior. For example, a retail store manager can use this facts to take strategic marketing decisions such as co-promoting products or putting them closer on the shelves[11].The first algorithm has been proposed named Apriori for finding the Frequent Itemsets occurred in the database [12]. Apriori algorithm is useful for searching the association rules among items in market-basket data [13],[14]. Association rules use two main constraints, i.e. minimum support and minimum confidence. The task of frequent itemset mining has various applications, but it can also be viewed with limitations in terms of the assumptions that it makes. These limitations have generated the need of extension to FIM. Some work has done on sampling techniques also which can be one of the solution for the limitation of FIM[15]. One of the remarkable limitations of traditional FIM is that it assumes that all items are equal. But in real-life applications, items are often different from each other[12].

In actual situations some items naturally have more chance of being frequent than others. This show the way to...
generate the rare item problem [12], which means that some items are much less likely to appear in frequent itemsets than others. Only frequency consideration in traditional techniques is not sufficient. Mining rare patterns from Databases has always been overlooked and giving more emphasis on frequent one. The unknown and unusual patterns are proficient in discovering hidden useful information from databases in various domains of applications. The first attempt towards rare association rule mining was made by Liu et al. [12] in their algorithm called MS-Apriori that employs an Apriori like strategy to incorporate some rare items during itemset generation. The authors argued that a single support threshold cannot be used for extracting the rare patterns effectively and ended up proposing a “multiple support framework” for the same. The framework assigns each item their individual support values instead of relying on a single one. The algorithm is efficient in finding rare patterns but it employed an additional parameter $\beta$ that adds to the computational complexity of the algorithm. Kiran et al. [16] in their algorithm IMS-Apriori, improved the initial MS-Apriori algorithm by incorporating another parameter of support difference. Even though it succeeded in generating more number of rare items it increases the burden of assigning two extra parameters $\beta$ and support difference. Lee at al. [17] extended the concept of multiple minimum supports using a model called maximum constraints model. The minimum support considered in this case is the maximum value among the minimum support values assigned to each item. The algorithm is faster due to granular bit string computation but fails to generate the complete set of rare items. Some algorithms extend the Apriori algorithm and use only a single minimum support threshold to find the rare itemsets. The most significant effort in this regard was made in [18]. The algorithm called ARIMA is capable of finding the complete set of rare items but spends a lot of time looking for the rare and frequent itemsets. ARIMA is further extended by Hoque et al. [19] in their algorithm FRIMA that generates both the frequent and rare itemsets. The algorithm maintains the rare, frequent and zero itemsets in three different candidate lists and later on merges the lists containing frequent and rare itemsets into a single list removing the zero itemsets. The algorithm handled the generation of whole set of rare items in less execution time than ARIMA but it increases memory utilizations as it also retain zero itemsets along with frequent and rare itemsets. Adda et al. [20] employed a strategy different from the previous approaches. Their algorithm AIRM, carried out the level-wise search in top-down manner contrasting the traditional bottom up search approach. The algorithm initially generates the largest candidate itemset combining all rare items and then proceeds to generate the smaller candidate itemsets. Similar to FRIMA, it also suffers from the drawback of generating zero support itemsets. Pillai and Vyas [21],[22] identified the need for high-utility rare itemsets and proceed to generate the same in their algorithm HURI. To measure the significance of rare itemsets, HURI consider the utility values of the itemsets along with their frequencies. The itemsets satisfying the predefined minimum utility value are considered to be rare, discarding other itemsets. Despite generating the user interested rare itemsets, the algorithm proves to be tedious due to the pre-assignment of utility values to each individual item. Instead of generating the complete set of rare itemsets, Haglin and Manning [23] developed the MINIT algorithm to generate only a subset of the rare itemsets called minimal infrequent itemsets. The algorithm assigns individual ranks to the items based on their support values and further considers only the higher rank items for itemset generation. The algorithm spends lesser execution time due to the generation of only minimal infrequent itemsets but still misses out some significant rare itemsets. Rarity algorithm proposed by Troiano et al. [24] considers the longest transaction in the database for rare itemset generation and performs a levelwise top-down search like AIRIM. The algorithm maintains a Candidate list for retrieving the rare itemsets and a Veto list for retaining the frequent itemsets. The rare itemsets generated are finally stored in another list. Despite generating the complete set of rare items, the algorithm undergoes memory overhead. In addition to the usage of a single minimum support threshold or multiple minimum support thresholds, some rare pattern mining techniques employ dynamic thresholds or more than one threshold. RSAA algorithm proposed by Yun et al. [25] employed two thresholds, one for generating the rare itemsets and another for generating the frequent itemsets. The advantage of this algorithm is that it is independent of the parameter $\beta$ employed by MS-Apriori but fails to outperform in terms of execution time. Tao et al. [14] in their algorithm WARM employed weighted support instead of minimum support threshold. Based on the significance of items, a weight is assigned to each item and only those items are considered further that satisfy the predefined weight threshold. However, assigning proper weights to each item adds to complexity of the algorithm. Wang et al. [26] in their algorithm Adaptive Apriori, pushed some support constraints on the itemsets. The lowest minimum support is considered, in case two or more constraints are applied on the itemsets. Maintaining the ordering of items even at run time becomes a tedious affair for the algorithm. DCS Apriori developed by Selvi and Tamilarasi [27] uses two support thresholds: Dynamic and Collective. Using the Dynamic support count, significant rare items are retained and the items that do not satisfy the Collective support are removed. Even though the algorithm is free of the user-defined threshold still it fails to generate the whole set of rare items. Sadhasivam and Tamilarasi [28] proposed Automated Apriori Rare that automatically assigns the support thresholds to items to derive the frequent as well as rare itemsets. The algorithm employs the strategy of MS-Apriori to extract the rare itemsets and Apriori to derive the frequent itemsets. The algorithm has the advantage of operating in parallel but misses out some significant rare itemsets [29]. Practically, the FIM is limited by the intention of the discovered itemsets and quantity is not considered (all items are viewed as having the same importance), high-utility itemsets introduced utility [30], a measure which is a weight/profit associated with each itemset. In [10],[31],[32], Examples explain that all FIMs are not profitable and all high profit needs not be frequent. Due to this fact, many itemsets emerged like [33] rare itemsets, [34] infrequent itemsets, and [35] closed itemsets, [35] lattice-based mining [36]. In real world, for decision making the user wants to know the total profit earned by an item or itemset. For this it needs to take
into account the quantity of the item purchased. The profit of an item considers the profit of single item and the number of item purchased. To address these, utility mining has been launched in which utility considers both the profit and the number of items purchased. The utility of an itemset can be evaluated as the product of profit of an items and number of items purchased. Utility mining considers both the importance of an item in the database (i.e. profit) or external utility and the importance of an item in the transaction (i.e.) quantity or internal utility of an item. An item is said to be high utility item if the sum of the utility of the item in the database is greater than the user specified minimum threshold, otherwise it is low utility item. High utility pattern mining is a major research area in recent years and many researches are carried in different areas of high utility mining (HUI) which includes sequence high utility mining, lossless representation of HUI, incremental HUI etc [37]. Weighted itemset mining is an extension of frequent itemset mining where weights are coupled to each item to indicate their relative importance[38],[39],[40] with the objective to find itemsets that have a minimum weight. The infrequent weighted itemsets is a popular variation to this problem [34]. The major extension of weighted itemset mining is induced in terms of High-utility itemset mining (HUIM) where not only weights are considered but also purchase quantities or utility in transactions [41],[42],[43],[44],[45],[46],[47]. In traditional FIM, either an itemset appears in a transaction or not. In HUIM, the utility or quantity is also indicated in transactions. For example, a transaction could specify that a customer has bought two desktops and one pen drive. In HUIM, weights are used to indicate how much profit is generated by each unit sold of a product. The objective of HUIM is to find all itemsets that have a utility higher than a given threshold in a database (i.e. itemsets generating a high profit). Plenty of work has been done on HUIM so far like Two Phase algorithm[45] where an upper-bound, called the TWU is used to reduce the search space, tighter upper-bounds on the utility is introduced so that the algorithm be able to prune a larger part of the search space and improve the performance of HUIM algorithms [41],[48][49],[46],[47] current fastest HUIM algorithm is EFIM[19], shelf-time periods of items[50], discover the top-k most profitable itemsets [51],[52] etc.The rest of the paper is organized as follows: in Section 2 Comprehensively discusses Preliminaries. In Section 3 A Hybrid Approach for Mining High Utility Rare Itemsets over Uncertain Databases (IHURU-Algorithm) is proposed. In Section 4 Experimental Assessment on the uncertain dataset example given by Lan Y et al, (2015) are reported. Finally, we conclude the paper in Section 5.

2. PRELIMINARIES

In this section the basic definitions have discussed which are important to understand the problem statement. Let \( I = \{i_1, i_2, ..., i_n\}\)signifies a set of distinct items. A unit value \( u(i)\) is associated with each items \( i_p\). A itemset is a non-empty subset \( X \). \( X = \{x_1, x_2, ..., x_l\}\) denotes the set of itemset. \( X \) is a l-item set if it has l items. In given example D is a uncertain transaction database in which Given an uncertain transaction database D, each transaction is denoted as \( < tid, Y, >\), where tid is the transaction identifier, and \( Y = \{y_1(p_1), y_2(p_2),..., y_m(p_m)\}\). Every unit has an item \( y_i\). Also every unit has a probability \( p_i\), which denotes the possibility of item \( y_i\) appearing in the tid tuple. The minimum utility threshold is also defined as \( \mu \) [3].

Let us take an example of the uncertain dataset in Table 1. The utility table is available in Table 2 shown below [3]. \( \mu = 25\% \) i.e. minimum utility threshold.

<table>
<thead>
<tr>
<th>TID</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>4</td>
<td>12</td>
<td>6</td>
<td>15</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: An Uncertain Database

Table 2: Utility Table

<table>
<thead>
<tr>
<th>TID</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>15</td>
</tr>
</tbody>
</table>

Utility of an Item: The Utility of an item is the quantity of item pu purchased. Utility of an item \( X \) in transaction \( T \) can be defined as:

\[
U(i_j, T_d) = p(i_j, T_d) \times u
\]

(1)

For example, the utility of an items (C) in \( T_3 \) is,

\[
U(C, T_3) = p(C, T_3) \times u(C) = 0.1 \times 12 = 1.2
\]

Definition 2:

Utility of an itemset \( X \) in transaction \( T_d \): The Utility of an itemset \( X \) in transaction \( T_d \) is denoted by \( U(X, T_d) \) [3]. which can be defined as:

\[
U(X, T_d) = \sum_{i_j \in X \subset T_d} U(i_j, T_d)
\]

(2)

For example, the utility of (BC) in \( T_3 \) is calculated as,

\[
U(BC, T_3) = U(B, T_3) + U(C, T_3)
\]

\[
= p(B, T_3) \times U(B) + p(C, T_3) \times U(C)
\]

\[
= (0.2 \times 1) + (0.1 \times 12) = 1.4
\]

Definition 3:
Utility of an itemset $X$ in Uncertain database $D$: The utility of an itemset $X$ in Uncertain database $D$ is denoted as $U(X)$ [3], it can be defined as:

$$U(X) = \sum_{X \subseteq T_d \land T_d \in D} U(X, T_d)$$  \hspace{1cm} (3)

For example,

$$U(C) = U(C, T_1) + U(C, T_2) + U(C, T_3) + U(C, T_4) + U(C, T_5) + U(C, T_{10}) = 3.6 + 1.2 + 2.4 + 2.4 + 3.6 = 16.8$$

$U(BC) = U(BC, T_1) + U(BC, T_2) + U(BC, T_3) + U(BC, T_4) + U(BC, T_5) + U(BC, T_{10}) = (1.4 + 3.8) = 5.2$

Definition 4:
Transaction utility of transaction $T_q$: The transaction utility of transaction $T_q$ is denoted as $tu(T_q)$ [3], which can be defined as:

$$tu(T_q) = \sum_{j=1}^{m} U(i_j, T_d)$$  \hspace{1cm} (4)

in which $m$ is the number of items in $T_d$.

For example,

$$TU(T_3) = U(A, T_3) + U(B, T_3) + U(C, T_3) + U(E, T_3) = 0.4 + 0.2 + 1.2 + 4.5 = 6.3$$

Definition 5:
Total utility in $D$: The total utility in $D$ is the sum of all transaction utilities in $D$ and is denoted as $TU$ [3], which can be defined as:

$$TU = \sum_{T_d \in D} TU(T_d)$$  \hspace{1cm} (5)

For example, the transaction utilities for $T_1$ to $T_{10}$ are respectively calculated as $TU(T_1) = 7.4$, $TU(T_2) = 1.3$, $TU(T_3) = 6.3$, $TU(T_4) = 2.4$, $TU(T_5) = 3$, $TU(T_6) = 6.2$, $TU(T_7) = 4.4$, $TU(T_8) = 1.9$, $TU(T_9) = 6.0$, $TU(T_{10}) = 5.3$.

The total utility in $D$ is the sum of all transaction utilities in $D$, which is calculated as:

$$TU = (7.4 + 1.3 + 6.3 + 2.4 + 3 + 6.2 + 4.4 + 1.9 + 6 + 5.3) = 44.2$$

Definition 6:
High Utility Itemset (HUI): An itemset $X$ is defined as a high utility itemset (HUI) if its utility value $U(X)$ is not less than the minimum utility count [3] as:

$$\sum_{X \subseteq T_d \land T_d \in D} U(X, T_d) = U(X) \geq \mu \times TU$$  \hspace{1cm} (6)

For example, suppose that the minimum utility threshold $\mu$ is set at 25%. An itemset $(C)$ is considered as a HUI in $D$ since $U(C) = 16.8$, which is larger than the minimum utility count $(0.25 \times 44.2) = 11.05$.

Definition 7:
Transaction-weighted utility (TWU): The transaction-weighted utility (TWU) of an itemset $X$ is the sum of all transaction utilities $TU(T_d)$ containing an itemset $X$ [3], which is defined as:

$$TWU(X) = \sum_{X \subseteq T_d \land T_d \in D} TU(T_d)$$  \hspace{1cm} (7)

Definition 8:
High transaction-weighted utilization itemset (HTWUI): An itemset $X$ is considered as a high transaction-weighted utilization itemset (HTWUI) if its

$$TWU(X) \geq TU \times \mu$$

Theorem 1 (Downward Closure Property of HTWUI): Let $X^k$ and $X^{k-1}$ be the HTWUI from uncertain databases, and $X^{k-1} \subseteq X^k$. The TWU ($X^{k-1}$) $\geq$ TWU ($X^k$) [3].

Proof: Let $X^{k-1}$ be a $(k-1)$-itemset and its superset $k$-itemset is denoted as $X^k$. Since $X^{k-1} \subseteq X^k$, thus,

$$TWU(X^k) = \sum_{X \subseteq T_d \land T_d \in D} TU(T_d)$$

$$\leq \sum_{X^{k-1} \subseteq T_d \land T_d \in D} TU(T_d)$$

$$= TWU(X^{k-1})$$  \hspace{1cm} (8)

Corollary 1: If an itemset $X^k$ is a HTWUI, every subset $X^{k-1}$ of $X^k$ is a HTWUI [3].

Corollary 2: If an itemset $X^k$ is not a HTWUI, no superset $X^{k+1}$ of $X^k$ is a HTWUI [3].

Theorem 2 (HUIs $\subseteq$ HTWUIs): The transaction-weighted downward closure (TWDC) property ensures that HUIs $\subseteq$ HTWUIs, which indicates that if an itemset is not a HTWUI, then none of its supersets will be HUIs [3].

Proof: $\forall X \subseteq D$, $X$ is an itemset; thus,

$$U(X) = \sum_{X \subseteq T_d \land T_d \in D} U(X, T_d)$$

$$\leq \sum_{X \subseteq T_d \land T_d \in D} TU(T_d)$$

$$= TWU(X)$$  \hspace{1cm} (9)

Considering the above definitions as base, the statement of problem of mining HUIs over uncertain databases can be stated as, given an uncertain database $D$ with total utility $TU$, $\mu$ the minimum utility threshold, the problem of mining HUIs over uncertain databases is: To mine HUIs whose utilities are not less than $(\mu \times TU)$ [3].

Let us consider transaction database in Table 3 with total utility which have been calculated with the help of Table 4 profit table shown below.
Improved UHUI-Apriori [53] using idea of improved apriori algorithm. The major limitations in apriori algorithm has been focussed and updated the process of item generation method. Extensive experiments have been performed to test the performance of these approaches over our sample example dataset. Mohammad et al in [1] have proposed an approach, here the Apriori algorithm enhanced to reduce the time consuming for candidates itemset generation. Firstly it scans all transactions to generate \( L_1 \) which contains the items, their support count and Transaction ID where the items are found. Then \( L_1 \) is used later as an assistant to produce \( L_2, L_3 \ldots L_k \). At the time when \( C_2 \) get generated, a self-join \( L_1 \ast L_1 \) will be generated to construct 2-itemset \( C(x, y) \), where \( x \) and \( y \) are the items of \( C_2 \). Before scanning all transaction records to count the support count of each candidate, use \( L_1 \) to get the transaction IDs of the minimum support count between \( x \) and \( y \), and thus scan for \( C_2 \) only in these specific transactions. The same thing for \( C_3 \), construct 3-itemset \( C(x, y, z) \), where \( x, y \) and \( z \) are the items of \( C_3 \) and use \( L_1 \) to get the transaction IDs with their minimum support count between \( x, y \) and \( z \), then scan for \( C_3 \) only in these specific transactions and repeat these steps until no new frequent itemsets are identified. The process can be clearer through Figure1 given below:

![Figure 1: Steps for C_k Generation [1]](image)

On the basis of the approach proposed in [22], shalini et al has proposed a novel efficient algorithm HURIU which reduced the extraction time of high utility rare itemsets over uncertain database drastically [53].The explanation of the working process of complete phases have also been explained with the help of experiment on a sample dataset. The proposed approach is a two phase method.

Let us take an example of a sample uncertain dataset shown in Table 1 and the profit value associated with each item in Table 2.Let the minimum utility threshold \( \mu = 25 \% \) and the maximum frequency support is 2. Improved Apriori concept with Apriori-inverse concept is used in HURIU (High Utility Rare Itemset over Certain Dataset) algorithm presented and generates HURIU over Uncertain Dataset. In first phase, rare itemsets are extracted by considering those itemsets which have support value less than the maximum support threshold using apriori-inverse [2] concept. In second phase, rare itemsets are extracted by considering those itemsets which have profit value less than the minimum utility threshold using apriori-inverse [2] concept.
Algorithm: IHURIU algorithm
// D is uncertain database; utable is table of utilities for itemsets, min. threshold given by μ
Input: D, an uncertain database, utable, utility table, μ minimum utility threshold
Output: the set of high utility rare itemsets (HURIs)

// Loop for calculating Transaction Weighted Utility
// Scanning database and probabilities
1. For each $T_d$ in D ∧ $i_j$ in $T_d$ Do
2. calculate $TWU(i_j)$
3. End for loop

// Condition Check for High Utility and generate set of HTWUI
4. For each $T_d$ in D ∧ $i_j$ in $T_d$ do check
5. If $TWU(i_j) \geq TU \times \mu$ Then
6. Set $HTWUI^1 \leftarrow i_j$
7. End If
8. End For loop
9. set $k \leftarrow 2$
10. set $X$ as $(k) - itemset$

// start candidate generation and check condition for HURI.
11. while $HTWUI^{k-1} \neq null$ Do
12. $C_k \leftarrow$ Improved – apriori
13. For each $k - itemset X$ in $C_k$ Do
14. Scan D to Find $TWU(X)$
15. If $TWU(X) \geq TU \times \mu$ Then
16. $HTWURI^k \leftarrow X$
17. End If
18. End For
19. Do $k \leftarrow k + 1$
20. End While Loop
21. $HTWURIs \leftarrow HTWUI^k$

// Rescanning of D
22. For each $k - itemset X$ in HTWURIs Do
23. Scan D to Find $u(X)$
24. If $u(X) \geq TU \times \mu$ Then
25. $HURI^k \leftarrow X$
26. End If
27. End For
28. $HURIs \leftarrow HURI^k$

4. EXPERIMENTAL ASSESSMENT

This section illustrates the experimental analysis and results of proposed IHURIU algorithm. The experiment is implemented in Java (jdk 1.6), NetBeans IDE 7.0 & MySQL. This implementation have been done with sample example dataset with varying thresholds. The screen of sample dataset for implementation using MySQL can be seen in Figure 2.

Figure 2: Sample Dataset

For the experimental assessment consider the sample uncertain database is given in Table 1 and Table 2 where each transaction is assigned a unique probability value in the range of 0.1 to 0.5. The experiment have performed for different higher and lower threshold values viz 25%, 35%, 40%, 50%, and 60% respectively. This implementation have done for the performance analysis of the proposed algorithm IHURIU, HURIU[53], UHUI-apriori[3]. The performance has been evaluated on the basis of three parameters i) Run Time
ii) Memory Utilization iii) Total Number of Itemsets

4.1 Run Time

In computer science, the analysis of algorithms is for the determination of the amount of resources such as time which is necessary to execute them. Here, comparison analysis between the proposed algorithms IHURIU, HURIU and UHUI-apriori is done with various threshold values on the sample dataset. Results can be seen in the graph shown in the below Figure 3.

4.2 Memory Utilization

Memory complexity can be given in terms of the size used by the algorithm for the work. The memory utilization is analyzed for both higher and lower threshold. We can observe easily that our proposed algorithm act differently for various threshold values. Results are shown in below Figure 4.

4.3 Total No. of Items Generation

It can be clearly observed in the below graph that the Number of high utility rare items (HURI) are more for the lower threshold and reduced when the threshold is high. The comparison can also be seen between high utility items, high utility rare items using HURIU approach and high utility rare items using proposed approach. The analysis can be seen in the below Figure 5.

5 CONCLUSION

In this paper the working analysis has been done on the approaches to generate variety of high utility items. The high utility items(HUI) generated by UHUI-apriori algorithm, high utility itemsets which are rare in nature(HURI) using HURIU algorithm and high utility rare itemsets(HURI) using proposed IHURIU algorithm over uncertain database. It can be concluded that proposed algorithm IHURIU can be suggested for better utilization of memory as it consumes less memory for the generation of High Utility Rare Itemsets.

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Algorithms</th>
<th>Support (Threshold %)</th>
<th>Run Time Utilization (in seconds)</th>
<th>Memory Utilizations (in bytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>UHUI-apriori</td>
<td>25%</td>
<td>0.322</td>
<td>2.107421875</td>
</tr>
<tr>
<td></td>
<td>Algorithm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>35%</td>
<td>0.402</td>
<td>2.143608093</td>
</tr>
<tr>
<td></td>
<td></td>
<td>40%</td>
<td>0.386</td>
<td>2.187011719</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50%</td>
<td>0.4</td>
<td>2.10710144</td>
</tr>
<tr>
<td></td>
<td>Improved-UHUI-apriori</td>
<td>60%</td>
<td>0.326</td>
<td>2.106452942</td>
</tr>
<tr>
<td>2</td>
<td>Algorithm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>25%</td>
<td>0.36</td>
<td>2.151176453</td>
</tr>
<tr>
<td></td>
<td></td>
<td>35%</td>
<td>0.372</td>
<td>2.106758118</td>
</tr>
<tr>
<td></td>
<td></td>
<td>40%</td>
<td>0.359</td>
<td>2.10773468</td>
</tr>
<tr>
<td></td>
<td></td>
<td>50%</td>
<td>0.396</td>
<td>2.107284546</td>
</tr>
</tbody>
</table>
It consumes more time hence not recommended for better time utilization. It is also concluded that as the threshold value is increases the generation of HURI is decreases and with lower threshold values it increases the number of HURI generated.

Table 6 : Conclusion Table of comparison of Time utilized and Memory utilized

<table>
<thead>
<tr>
<th>Sr.No.</th>
<th>Algorithms</th>
<th>Support (Threshold %)</th>
<th>No. Of Itemsets Generated</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>HURI itemsets using proposed approach</td>
<td>25%  35%  40%  50%  60%</td>
<td>10  15  16  3  2</td>
</tr>
<tr>
<td>2</td>
<td>HUI itemsets using UHUI-apriori approach</td>
<td>25%  35%  40%  50%  60%</td>
<td>10  9  5  3</td>
</tr>
<tr>
<td>3</td>
<td>HURI itemsets using HURIU approach</td>
<td>25%  35%  40%  50%  60%</td>
<td>10  5  4  2  2</td>
</tr>
</tbody>
</table>

Table 7 : Conclusion Table of comparison of different types of itemsets generated using different approaches

Hence as per the above conclusion tables we can conclude that in the proposed hybrid approach using IHURIU algorithms the memory utilization is reduced drastically for both the higher and lower threshold values. It can also be concluded as per Table 6 & Table 7 that if the threshold values are low the number of high utility rare itemsets will be more and as the threshold values are moving higher the generation of high utility rare itemsets will be less. Hence we can recommend the Hybrid approach for the extraction of High Utility Rare Itemset (HURI) over Uncertain Database for better memory utilization.

REFERENCES
association rules”, IEEE Symposium on Compu-
tational Intelligence and Data Mining, IEEE, March 2009.
[26] K. Wang, Y. He and J. Han, “Pushing support constraints into association rules mining”. IEEE Transactions on Knowledge and Data Engineering, 15(3), 2003.


AUTHOR PROFILES

I. Ms. Shalini Zanzote Ninoria pursued Master of Computer Application from Nagpur University, India in year 2005. She has also pursued MPhil(CS) from R.D.University, Jabalpur, India. She is currently pursuing Ph.D.(CS) at Department of Mathematics and Computer Science, R.D.University, India since 2016 under the guidance of Dr. S.S. Thakur. She has published research papers in reputed international journals. Her main research work focuses on Data Mining, Association Rule Mining, High Utility Mining, and Rare Itemset Mining. She has 9 years of teaching experience and 4 years of Research Experience.

II. Dr.S.S.Thakur is currently designated as Principal and Professor & Head at Jabalpur Engineering College, Jabalpur, India with 35 years of Teaching and Academic experience. He has pursued Ph.D. from Dr. Hari Singh Gour University, India in 1982 and is a member of various computer societies with more than 247 research publications with one book of his credit in reputed national and international journals. Dr.S.S.Thakur has also guided many Research Scholars for the successful award of Ph.D. His main research work focuses on field of fuzzy topology, intuitionistic fuzzy topology and Data Mining.