Mining High Utility Itemsets over Uncertain Databases

Yuqing Lan[†] Yang Wang[†] Yanni Wang[‡] Shengwei Yi[§] Dan Yu[*]
[†]School of Computer Science and Engineering, Beihang University, Beijing, China
[‡]School of Software, Beihang University, Beijing, China
[§]China Information Technology Security Evaluation Center, Beijing, China
[*]China Standard Software Co. Ltd, Beijing, China
[†]{lanyuqing, yangwang}@buaa.edu.cn, [‡]wangyn@buaa.edu.cn, [§]yishengwei@foxmail.com, [*]yudan@cs2c.com.cn

***Abstract*—Recently, with the growing popularity of Internet of Things (IoT) and pervasive computing, a large amount of uncertain data, i.e. RFID data, sensor data, real-time monitoring data, etc., has been collected. As one of the most fundamental issues of uncertain data mining, the problem of mining uncertain frequent itemsets has attracted much attention in the database and data mining communities. Although some efficient approaches of mining uncertain frequent itemsets have been proposed, most of them only consider each item in one transaction as a random variable and ignore the utility of each item in the real scenarios. In this paper, we focus on the problem of mining high utility itemsets (MHUI) over uncertain databases, in which each item has a utility. In order to solve the MHUI problem over uncertain databases, we propose an efficient mining algorithm, named UHUI-apriori. Extensive experiments on both real and synthetic datasets verify the effectiveness and efficiency of our proposed solutions.**

I. Introduction

Recently, with the growing popularity of Internet of Things (IoT) and pervasive computing, a large amount of uncertain data has been collected from different kinds of devices, such as RFID, sensors, and real-time monitoring systems[1], [2]. As one of the most fundamental issues of uncertain data mining, the problem of mining uncertain frequent itemsets has attracted much attention in the database and data mining communities[3], [4], [5], [6], [7]. Although some efficient approaches of mining uncertain frequent itemsets have been proposed[8], most of them only consider each item in one transaction as a random variable and ignore the unit utilities of items in the real scenarios.

In kinds of recommender system (e.g. music, video) analysis, mining frequent itemsets from an uncertain database refers to the discovery of itemsets which may frequently appear together in the records (transactions). However, in these works, the unit values (utilities) of items are not considered in their frameworks of uncertain frequent itemsets mining. Hence, it cannot satisfy the requirement of a user who is interested in discovering itemsets with high enjoyment values. For example, a table shows some songs and their utility values. The utility value of each song is scored by all the persons listened to it according to their preference degree to the song. Another is the listening records of some users showing which songs they listened to and the probability they like them. Namely, in a song transaction of a user, the probability he likes these songs are 0.8, 0.8, 0.7, and 0.7.When songs are recommended, not only the frequency of songs in records of different user should be considered, but the popularity of songs should also be taken into account. In view of this, utility mining is a necessary topic in data mining for discovering itemsets with high utility, such as values (utilities), in uncertain databases.

Mining high utility itemsets from the databases refers to finding the itemsets with high utilities[9], [10], [11]. The basic meaning of utility is the interestingness / importance / profitability of an item to the users.

To sum up, we make the following contributions::

- To the best of our knowledge, this is the first work to formulate the problem of mining high utility itemsets over uncertain databases (MHUI).
- Due to the challenges from utility constraints, the downward closure could not be directly used to mine HUIs, we propose the UHUI-Apriori algorithm to mine HUIs.
- We verify the effectiveness and efficiency of the proposed methods through extensive experiments on real and synthetic datasets.

The rest of the paper is organized as follows. Problem statement are introduced in Section 2. In Section 3, we present an efficient mining algorithm, named UHUI-Apriori. In Section 4, experimental evaluation on both real and synthetic datasets are reported. Finally, we conclude this paper in Section 5.

II. Problem Statement

In this section, we give several basic definitions about mining high utility itemsets from uncertain data.

Let $I = \{i_1, i_2, ..., i_n\}$ be a set of distinct items. Each item i_p has a unit value $u(i_p)$. We name a non-empty subset, X, of I as an itemset. For brevity, we use $X = x_1x_2...x_n$ to denote itemset $X = \{x_1, x_2, ...x_n\}$. X is a l - item set if it has l items. Given an uncertain transaction database D, each transaction is denoted as $< tid, Y >$, where tid is the transaction identifier, and $Y = \{y_1(p_1), y_2(p_2), ..., y_m(p_m)\}$. Y contains m units. Each unit has an item y_i and probability p_i, denoting the possibility of item y_i appearing in the tid tuple. A threshold, the minimum utility threshold, is defined as μ.

An example of uncertain databases is shown in Table 1. The corresponding utility table is shown in Table 2. In this example, the minimum utility threshold is set at μ (= 25%).

TABLE I. AN UNCERTAIN DATABASE

TID	Transactions
T_1	A (0.2) C (0.3) E (0.2)
T_2	B (0.2) D (0.3)
T_3	A (0.1) B (0.2) C (0.1) E (0.3)
T_4	C (0.2)
T_5	B (0.3) D (0.2) E (0.1)
T_6	A (0.2) C (0.2) D (0.5)
T_7	A (0.1) B (0.1) D (0.4) E (0.1)
T_8	B (0.4) E (0.1)
T_9	A (0.3) C (0.3) D (0.2)
T_{10}	B (0.2) C (0.3) E (0.1)

TABLE II. AN UTILITY TABLE

TID	A	B	C	D	E
Utility	4	1	12	6	15

Definition 1: The utility of an item i_j in T_d is defined as:

$$U(i_j, T_d) = p(i_j, T_d) \times u(i_j) \tag{1}$$

For example, the utility of an item (A) in T_1 is $U(A, T_1) = p(A, T_1) \times u(A) = 0.2 \times 4 = 0.8$.

Definition 2: The utility of an itemset X in transaction T_d is denoted as $U(X, T_d)$, which can be defined as:

$$U(X, T_d) = \sum_{i_j \in X \wedge X \subseteq T_d} U(i_j, T_d) \tag{2}$$

For example, the utility of (AC) in T_1 is calculated as $U(AC, T_1) = U(A, T_1) + U(C, T_1) = p(A, T_1) \times u(A) + p(C, T_1) \times u(C) = (0.2 \times 4) + (0.3 \times 12)$ = 4.4.

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

$$U(X) = \sum_{X \subseteq T_d \wedge T_d \in D} U(X, T_d) \tag{3}$$

For example, $U(A) = U(A, T_1) + U(A, T_3) + U(A, T_6) + U(A, T_7) + U(A, T_8) = (0.8 + 0.4 + 0.8 + 0.4 + 1.2)$ = 3.6, and $U(AC) = U(AC, T_1) + U(AC, T_3) + U(AC, T_6) + U(AC, T_9) = (4.4 + 1.6 + 3.2 + 4.8)$ = 14.

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

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

in which m is the number of items in T_d.

For example, $TU(T_1) = U(A, T_1) + U(C, T_1) + U(E, T_1)$ = 0.8 + 3.6 + 3 = 7.4.

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

$$TU = \sum_{T_d \in D} TU(T_d) \tag{5}$$

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.

TABLE III. DERIVED HUIS OVER AN UNCERTAIN DATABASE

Itemset	Actual Utility
(C)	16.8
(E)	13.5
(AC)	14
(BE)	11.7
(CE)	17.4
(ACD)	12.2
(ACE)	13.5

Definition 6: 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 as:

$$\sum_{X \subseteq T_d \wedge T_d \in D} U(X, T_d) = U(X) \geq \mu \times TU \tag{6}$$

For example, suppose that the minimum utility threshold μ is set at 25%. An item (A) is not considered as a HUI since U(A) = 3.6, which is smaller than $(0.25 \times 44.2) = 11.05$. An itemset(AC) is considered as a HUI in D since U(AC) = 14, which is larger than the minimum utility count = 11.05.

Based on the above definitions, the problem statement of mining HUIs over uncertain databases can be formulated as follows:

Given an uncertain database D with total utility is TU, the minimum utility threshold which is set as μ. The problem of MHUI over uncertain databases is to mine HUIs whose utilities are not less than $(\mu \times TU)$.

From the example given in Tables 1 and Table 2, the set of HUIs is shown in Table 3 when the minimum utility threshold is set at $\mu = 25\%$.

III. PROPOSED A MINING HIGH UTILITY ITEMSETS ALGORITHM OVER UNCERTAIN DATABASES

To the best of our knowledge, this is the first paper to discuss the MHUI over uncertain databases. In this section, we propose the UHUI-apriori algorithm to mine HUIs over uncertain databases.

A. Pruning strategy by downward closure property

In the well-known Apriori algorithm, the downward closure (DC) property is kept to reduce the number of candidates for association-rule mining (ARM). The DC property is also kept in the designed UHUI-apriori algorithm for mining HUIs. In MUHI over uncertain databases, the DC property of ARM could not be directly extended to mine HUIs. The transaction-weighted downward closure (TWDC) property was proposed to reduce the search space in MHUI.

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

$$TWU(X) = \sum_{X \subseteq T_d \wedge T_d \in D} TU(T_d) \tag{7}$$

Definition 8: An itemset X is considered as a high transaction-weighted utilization itemset (HTWUI) if its $TWU(X) \geq TU \times \mu$.

In Table 1, the TWU of an item (E) is calculated as $TWU(E) = TU(T_1) + TU(T_3) + TU(T_5) + TU(T_7) +$

$TU(T_8) + TU(T_{10})$ = (74 + 63 + 30 + 44 + 19 + 53) = 283. An item (E) is considered as a HTWUI since $TWU(E) = 283 \geq 134$.

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 $TWU(X^{k-1}) \geq TWU(X^k)$.

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,

$$\begin{aligned} TWU(X^k) &= \sum_{X^k \subseteq T_d \wedge T_d \in D} TU(T_d) \\ &\leq \sum_{X^{k-1} \subseteq T_d \wedge T_d \in D} TU(T_d) \\ &= TWU(X^{k-1}). \end{aligned} \quad (8)$$

corollary 1: If an itemset X^k is a HTWUI, every subset X^{k-1} of X^k is a HTWUI.

corollary 2: If an itemset X^k is not a HTWUI, no superset X^{k+1} of X^k is a HTWUI.

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.

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

$$\begin{aligned} U(X) &= \sum_{X \subseteq T_d \wedge T_d \in D} U(X, T_d) \\ &\leq \sum_{X \subseteq T_d \wedge T_d \in D} TU(T_d) \\ &= TWU(X). \end{aligned} \quad (9)$$

B. Detail of UHUI-apriori algorithm

The proposed UHUI-apriori algorithms has two phases: in the first phase, the HTWUIs are found, and in the second phase, the HUIs are drived with an additional database rescan. The TWUDC property inherits the TWDC property of the two-phase model to keep the downward closure property, thus reducing the search space for finding HUIs. Only the remaining $HTWUI^{k-1}$ will be used to generate the next C_k at each level. Based on the above definitions and theorems, detail of the proposed UHUI-apriori algorithm is described in UHUI-apriori algorithm.

Based on the designed TWUDC property, Theorem 2 ensures that the proposed UHUI-apriori algorithm can make sure that no supersets of small transaction-weighted itemsets are in the preliminary candidate set (correctness) and can extract the complete HUIs from the candidate HTWUIs (completeness). Therefore, the results of the proposed UHUI-apriori algorithm are correct and complete.

The proposed UHUI-apriori algorithm first scans the database to find the TWU values and the probabilities of all 1-itemsets in database(Line 1 to 3). The set of $HTWUI^k$ (k is initially set as 1) is then produced (Lines 4 to 8) and will be further used to generate the next candidates C_{K+1} for discovering $HTWUI^{k+1}$ in a level-wise way (Lines 11 to 20). In this process, the original database has to be rescanned to find the $HTWUI^{k+1}$ (Line 14). The first phase of UHUI-apriori algorithm is terminated when no candidate is generated. An additional database rescan is required in the second phase to find the final HUIs from the HTWUIs (Lines 22 to 29).

Algorithm 1: Sample algorithm

Require: D, an uncertain database; $utable$, utility table; μ minimum utility threshold;
Ensure: the set of high utility itemsets (HUIs).

```
1: for each T_d in D ∧ i_j in T_d do
2:    calculate TWU(i_j).
3: end for
4: for each T_d in D ∧ i_j in T_d do
5:    if TWU(i_j)≥ TU × μ then
6:       HTWUI^1 ← i_j.
7:    end if
8: end for
9: set k ← 2.
10: set X as (k)-itemset.
11: while HTWUI^{k-1} ≠ null do
12:    C_k ← Apriori_gen(HTWUI^{k-1}).
13:    for each k-itemset X in C_k do
14:       scan D to find TWU(X).
15:       if TWU(X)≥ TU × μ then
16:          HTWUI^k ← X.
17:       end if
18:    end for
19:    k ← K+1.
20: end while
21: HTWUIs ← HTWUI^k
22: for each k-itemset X in HTWUIs do
23:    scan D to find u(X).
24:    if u(X)≥ TU × μ then
25:       HUI^k ← X.
26:    end if
27: end for
28: HUIs ← HUI^k.
```

C. An illustrated example of UHUI-apriori algorithm

In order to keep consistency, Tables 1 and 2 are used to illustrate the proposed UHUI-apriori algorithm as the example step-by-step. Assume the minimum utility threshold is also set at 25%. The minimum support count can be calculated as $(44.2 \times 25\%)$ = 11.05. The UHUI-apriori algorithm first scans the database to find the TWU values of all 1-itemsets in the databases. The results are (A:30.3; B:22.2; C:33.6; D:20.9; E:28.3) in (A:30.3) indicates that the TWU(A) = 30.3. In this example, all itemsets satisfies the above conditions and then put into the set of $HTWUI^1$. Based on the designed UHUI-apriori algorithm, C^2 are then generated from $HTWUI^1$. A database scan is required to find the TWU values of C^2 as (AB:10.7; AC:25.9; AD:16.6; AE:18.1; BC:11.6; BD:8.7; BE:20.9; CD:12.2; CE:19; DE:7.4). Among them, only the itemsets (AC, AD, AE, BE, CD, CE) satisfy the condition as $TWU(X) \geq 11.05$; they are then put into the set of $HTWUI^2$. The variable k is then set to 3. This process is repeated until no candidates are generated. The results are shown in Table 4.

After the first phase, the second phase is executed with an additional database scan to find the actual utility value of each remaining candidate. The results of HUIs were shown in Table 3.

TABLE IV. DERIVED HTWUIS OVER AN UNCERTAIN DATABASE

Itemset	TWU
(A)	30.3
(B)	22.2
(C)	33.6
(D)	20.9
(E)	28.3
(AC)	25.9
(AD)	16.6
(AE)	18.1
(BE)	20.9
(CD)	12.2
(CE)	19
(ACD)	12.2
(ACE)	13.7

TABLE V. CHARACTERISTICS OF USED DATASETS

Database	$\|D\|$	$\|I\|$	AvgLen	MaxLen	Type
accidents	340,183	468	33.8	51	dense
T10I4D100K	100,000	870	10.1	29	sparse

IV. EXPERIMENTAL EVALUATION

Experiments for mining HUIs over uncertain datasets were conducted to evaluate the performance of the UHUI-apriori algorithm and the algorithm of direct traversal in terms of runtime, memory consumption. The algorithms in the experiments were implemented in C++ and performed on a PC with an Intel Core i5-3460 processor and 4GB of RAM. The platform is Visual Studio 2010 and 64-bit Microsoft Windows 7 OS. Experiments under varied minimum utility thresholds (MUs) is discussed below.

A. Datasets

Both real-life and synthetic datasets were used in the experiments. A real-life datasets, namely accidents datasets, as well as one synthetic dataset, T10I4D100K, were used in the experiments to evaluate the performance of two algorithms. The utility values are assigned to the items in accidents and T10I4D100K datasets by simulation model. In addition, due to the tuple uncertainty property, each transaction in these datasets is assigned a unique probability value in the range of 0.1 to 0.5. The characteristics of used datasets are shown in Table 5.

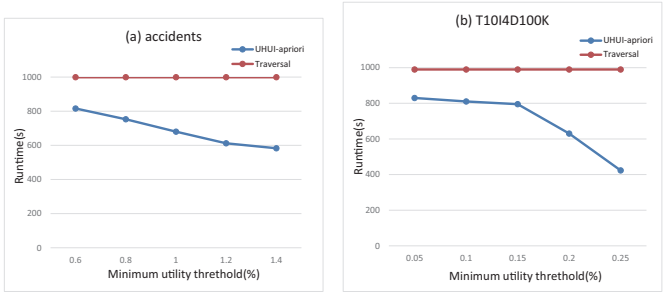

Fig. 1. Runtime of two algorithms under varied MUs

B. Runtime

The runtime of the two algorithms under varied MUs are compared and shown in Fig.1.

From Fig.1, we can observe that the proposed UHUI-apriori algorithm has better performance than the algorithm of direct traversal.

C. Memory Consumption

The memory consumption of the two algorithms under varied MUs are compared and shown in Fig.2.

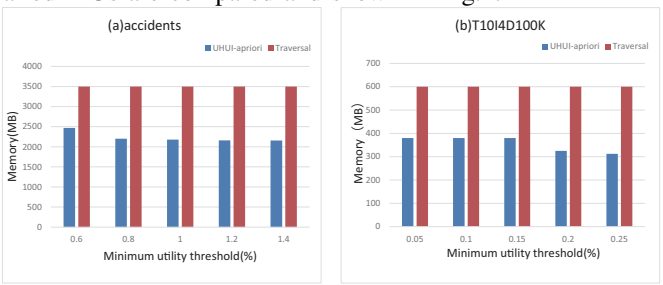

Fig. 2. Memory consumption under varied MUs

From Fig.2, it can be clearly seen that the proposed UHUI-apriori algorithm requires less memory compared to the algorithm of direct traversal under varied MUs.

V. CONCLUSION AND FUTURE WORKS

In this paper, a UHUI-apriori algorithm is proposed for mining high utility itemsets over uncertain databases. Different from the algorithm of direct traversal, UHUI-apriori algorithm is based on the TWUDC property. By the evaluation of experiments, UHUI-apriori algorithm is much better than the algorithm of direct traversal in both running time and memory consumption.

Since this is the first work for mining high utility itemsets over uncertain databases, further research issues including dynamic data mining, stream mining, and top-k patterns mining can also be studied. Besides, designing more efficient and condensed structure based on different uncertain models for mining the desired information is also another critical issue in the nearly future.

ACKNOWLEDGMENT

This work is supported in part by the National Natural Science Foundation of China (Grant No.61232009).

REFERENCES

[1] Y. Tong, X. Zhang, and L. Chen, "Tracking frequent items over distributed probabilistic data," *World Wide Web*, pp. 1–26, 2015.

[2] Y. Tong, L. Chen, and J. She, "Mining frequent itemsets in correlated uncertain databases," *Journal of Computer Science and Technology*, vol. 30, no. 4, pp. 696–712, 2015.

[3] C.-K. Chui, B. Kao, and E. Hung, "Mining frequent itemsets from uncertain data," in *PAKDD 2007*, pp. 47–58.

[4] C. C. Aggarwal, Y. Li, J. Wang, and J. Wang, "Frequent pattern mining with uncertain data," in *SIGKDD 2009*, pp. 29–38.

[5] Y. Tong, L. Chen, and B. Ding, "Discovering threshold-based frequent closed itemsets over probabilistic data," in *ICDE 2012*, pp. 270–281.

[6] T. Bernecker, H.-P. Kriegel, M. Renz, F. Verhein, and A. Zuefle, "Probabilistic frequent itemset mining in uncertain databases," in *SIGKDD 2009*, pp. 119–128.

[7] Y. Tong, L. Chen, and P. S. Yu, "Ufimt: an uncertain frequent itemset mining toolbox," in *SIGKDD 2012*, pp. 1508–1511.

[8] Y. Tong, L. Chen, Y. Cheng, and P. S. Yu, "Mining frequent itemsets over uncertain databases," *PVLDB*, vol. 5, no. 11, pp. 1650–1661, 2012.

[9] R. C. Chan, Q. Yang, and Y.-D. Shen, "Mining high utility itemsets," in *ICDM 2003*, pp. 19–26.

[10] C. F. Ahmed, S. K. Tanbeer, B.-S. Jeong, and Y.-K. Lee, "Efficient tree structures for high utility pattern mining in incremental databases," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 12, pp. 1708–1721, 2009.

[11] V. S. Tseng, C.-W. Wu, B.-E. Shie, and P. S. Yu, "Up-growth: an efficient algorithm for high utility itemset mining," in *SIGKDD 2010*, pp. 253–262.