Mining uncertain data

Carson Kai-Sang Leung*

As an important data mining and knowledge discovery task, *association rule mining* searches for implicit, previously unknown, and potentially useful pieces of information—in the form of rules revealing associative relationships—that are embedded in the data. In general, the association rule mining process comprises two key steps. The first key step, which *mines frequent patterns* (i.e., frequently occurring sets of items) from data, is more computationally intensive than the second key step of using the mined frequent patterns to *form association rules*. In the early days, many developed algorithms mined frequent patterns from traditional transaction databases of precise data such as shopping market basket data, in which the contents of databases are known. However, we are living in an uncertain world, in which uncertain data can be found almost everywhere. Hence, in recent years, researchers have paid more attention to frequent pattern mining from probabilistic databases of uncertain data. In this paper, we review recent algorithmic development on mining uncertain data in these probabilistic databases for frequent patterns. © 2011 John Wiley & Sons, Inc. *WIREs Data Mining Knowl Discov* 2011 1 316–329 DOI: 10.1002/widm.31

INTRODUCTION

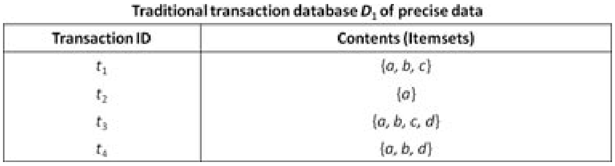
Data mining and knowledge discovery (DMKD)[1] techniques are widely used in various applications in business, government, and science. Examples include banking, bioinformatics, environmental modeling, epidemiology, finance, marketing, medical diagnosis, and meteorological data analysis. Available data in many of these applications are uncertain. Uncertainty can be caused by our limited perception or understanding of reality (e.g., limitations of the observation equipment; limited resources to collect, store, transform, analyze, or understand data). It can also be inherent in nature (e.g., due to prejudice). Moreover, sensors (e.g., acoustic, chemical, electromagnetic, mechanical, optical radiation and thermal sensors) are often used to collect data in applications such as environment surveillance, security, and manufacturing systems. Dynamic errors include inherited measurement inaccuracies, sampling frequency of the sensors, deviation caused by a rapid change of the measured property over time (e.g., drift, noise), wireless transmission errors, or network latencies. There is also uncertainty in survey data (e.g., number ‘1’ vs. uppercase letter ‘I’ vs. lowercase letter ‘L’) and uncertainty due to data granularity (e.g., city, province) in taxonomy. Disguised missing data (which are not explicitly represented as such but instead appear as potentially valid data values) also introduce uncertainty. In privacy-preserving applications, sensitive data may be intentionally blurred via aggregation or perturbation so as to preserve data anonymity. All these scenarios lead to huge amounts of uncertain data in various real-life situations.[2–5] In this paper, we review recent algorithmic development on mining such uncertain data. Note that there are different methodologies (e.g., probability theory, fuzzy set theory, rough set theory) for mining uncertain data. In this paper, we mainly focus on uncertainty in a probabilistic setting.

Over the past few years, various DMKD algorithms have been developed for clustering uncertain data,[6–8] classifying uncertain data,[9,10] detecting outliers from uncertain data,[11] and mining association rules from uncertain data. *Association rule mining*[12,13] is an important DMKD task where one searches implicit, previously unknown, and potentially useful associative relationships embedded in the data. The mining process generally comprises two key steps. The first key step *mines frequent patterns*[14] (i.e., frequently occurring sets of items) from data, and the second key step *forms association rules* of the form ‘$A \rightarrow C$’ using these mined frequent patterns as the antecedent A and consequence C of the rules. Between the two key steps, the first step is more computationally intensive than the second one. This explains why more attention has been focused on the first step, and many algorithms have been developed over the last two decades. In the early days, most of the

*Correspondence to: kleung@cs.umanitoba.ca

University of Manitoba, Winnipeg, Manitoba, Canada

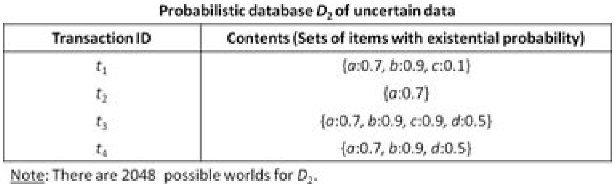
DOI: 10.1002/widm.31



Traditional transaction database D_1 of precise data

Transaction ID	Contents (Itemsets)
t_1	{*a*, *b*, *c*}
t_2	{*a*}
t_3	{*a*, *b*, *c*, *d*}
t_4	{*a*, *b*, *d*}

FIGURE 1 | A traditional transaction database D_1 of precise data.



Probabilistic database D_2 of uncertain data

Transaction ID	Contents (Sets of items with existential probability)
t_1	{*a*:0.7, *b*:0.9, *c*:0.1}
t_2	{*a*:0.7}
t_3	{*a*:0.7, *b*:0.9, *c*:0.9, *d*:0.5}
t_4	{*a*:0.7, *b*:0.9, *d*:0.5}

Note: There are 2048 possible worlds for D_2.

FIGURE 2 | A probabilistic database D_2 of uncertain data, in which items in each transaction are independent.

developed algorithms mined frequent patterns from traditional databases of precise data such as shopping market basket data, in which the contents of databases are known. In this paper, we focus on mining uncertain data in probabilistic databases for frequent patterns.

FREQUENT PATTERN MINING OF UNCERTAIN DATA

Due to the uncertainty in various real-life situations, users may not be certain about the presence or absence of an item x in a transaction t_i. They may suspect, but cannot guarantee, that x is present in t_i. The uncertainty of such suspicion can be expressed in terms of *existential probability* $P(x,t_i)$, which indicates the likelihood of x being present in t_i in a probabilistic database D of uncertain data. The existential probability $P(x,t_i)$ ranges from a positive value close to 0 (indicating that x has an insignificantly low chance to be present in D) to a value of 1 (indicating that x is definitely present). With this notion, each item in any transaction in traditional databases of precise data (e.g., shopping market basket data) can be viewed as an item with a 100% likelihood of being present in such a transaction. Figures 1 and 2 show a traditional transaction database D_1 containing precise data and a probabilistic database D_2 containing uncertain data, respectively.

Using the 'possible world' interpretation[15–17] of uncertain data, there are two possible worlds for an item x in a transaction t_i: (1) a possible world W_1 where x is present in t_i (i.e., $x \in t_i$) and (2) another possible world W_2 where x is absent from t_i (i.e., $x \notin t_i$). Although it is uncertain which of these two worlds to be the true world, the probability of W_1 to be the true world is $P(x,t_i)$ and the probability of W_2 to be the true world is $1 - P(x,t_i)$. To some extent, there are multiple items in each of many transactions in a probabilistic database D of uncertain data. Given a total of q independent items (from a domain of m distinct items, where $m \ll q$) in all transactions of D, there are $O(2^q)$ possible worlds. The *expected support* (*expSup*) of a pattern X in D can then be computed by summing the support of X in possible world W_j (while taking into account the probability of W_j to be the true world) over all possible worlds, i.e.,

$$expSup(X, D) = \sum_j [sup(X, W_j) \times Prob(W_j)],$$

where the probability $Prob(W_j)$ of W_j to be the true world can be computed by the following:

$$Prob(W_j) = \prod_{i=1}^{|D|} \left(\prod_{x \in t_i \text{ in } W_j} P(x, t_i) \times \prod_{y \notin t_i \text{ in } W_j} (1 - P(y, t_i)) \right).$$

The above expression for computing the expected support of X in D can be simplified[18] to become the following:

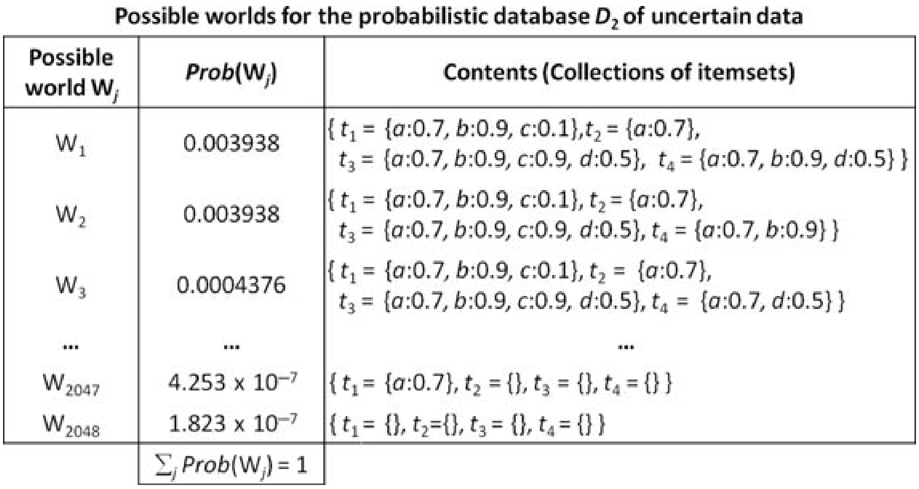
$$expSup(X, D) = \sum_{i=1}^{|D|} \left(\prod_{x \in X} P(x, t_i) \right).$$

In other words, the expected support of X in D can be computed as a sum (over all $|D|$ transactions) of product of existential probabilities of all items within X.

Given (1) a probabilistic database D of uncertain data and (2) a user-specified support threshold *minsup*, the research problem of *frequent pattern mining of uncertain data* is to find all frequent patterns from D. Here, a pattern X is *frequent* if and only if its expected support in D is no less than *minsup*, i.e., $expSup(X,D) \geq minsup$. See Figure 3 for all 'possible worlds' of the probabilistic database D_2 shown in Figure 2.

Apriori-Based Mining of Uncertain Data

To mine frequent patterns from uncertain data, Chui et al.[19] uses a levelwise breadth-first bottom-up mining approach with a candidate generate-and-test paradigm. Specifically, they modified the classical Apriori algorithm,[20,21] and called the resulting algorithm *U-Apriori*, to mine uncertain data. Like its



Possible worlds for the probabilistic database D_2 of uncertain data

Possible world W_j	$Prob(W_j)$	Contents (Collections of itemsets)
W_1	0.003938	{ t_1 = {a:0.7, b:0.9, c:0.1}, t_2 = {a:0.7}, t_3 = {a:0.7, b:0.9, c:0.9, d:0.5}, t_4 = {a:0.7, b:0.9, d:0.5} }
W_2	0.003938	{ t_1 = {a:0.7, b:0.9, c:0.1}, t_2 = {a:0.7}, t_3 = {a:0.7, b:0.9, c:0.9, d:0.5}, t_4 = {a:0.7, b:0.9} }
W_3	0.0004376	{ t_1 = {a:0.7, b:0.9, c:0.1}, t_2 = {a:0.7}, t_3 = {a:0.7, b:0.9, c:0.9, d:0.5}, t_4 = {a:0.7, d:0.5} }
...	...	...
W_{2047}	4.253×10^{-7}	{ t_1 = {a:0.7}, t_2 = {}, t_3 = {}, t_4 = {} }
W_{2048}	1.823×10^{-7}	{ t_1 = {}, t_2 = {}, t_3 = {}, t_4 = {} }
	$\sum_j Prob(W_j) = 1$	

FIGURE 3 | Possible worlds for D_2.

counterpart for mining precise data (the Apriori algorithm), U-Apriori also relies on the *Apriori property*[22] (which is also known as the *antimonotonic property* or the *downward closure property*) that all subsets of a frequent pattern must also be frequent. Equivalently, all supersets of any infrequent pattern are also infrequent.

To improve efficiency, Chui et al. incorporated the *LGS-trimming strategy* (which includes local trimming, global pruning, and single-pass patch up) into U-Apriori. The strategy trims away every item with an existential probability below the user-specified trimming threshold (which is local to each item) from the original database D of uncertain data and then mines frequent patterns from the resulting trimmed database D^{Trim}. If a pattern X is frequent in D^{Trim}, then X must be frequent in D. On the other hand, a pattern Y is infrequent in D if $expSup(Y,D^{\mathrm{Trim}}) + e(Y) < minsup$, where $e(Y)$ is the upper bound of the error estimated for $expSup(Y,D^{\mathrm{Trim}})$. Such an infrequent pattern Y can be pruned. Moreover, a pattern Z is potentially frequent in D if $expSup(Z,D^{\mathrm{Trim}}) \leq minsup \leq expSup(Z,D^{\mathrm{Trim}}) + e(Z)$. To patch up (i.e., to recover the missing frequent patterns), the expected supports of these potentially frequent patterns are verified by an additional single-pass scan of D. Although the LGS strategy improves the efficiency of U-Apriori, the algorithm still suffers from the following problems: (1) there is an overhead in creating D^{Trim}, (2) only a subset of all the frequent patterns can be mined from D^{Trim} and there is overhead to patch up (i.e., to recover the missing frequent patterns), (3) the efficiency of the algorithm is sensitive to the percentage of items having low existential probabilities, and (4) it is not easy to find an appropriate value for the user-specified trimming threshold.

To further improve the efficiency of U-Apriori, Chui and Kao[23] proposed a *decremental pruning* technique. Inherited from the Apriori algorithm, U-Apriori relies on the candidate generate-and-test paradigm for mining. The decremental pruning technique helps reduce the number of candidate patterns because it progressively estimates the upper bounds of expected support of candidate patterns after each database transaction is processed. If the estimated upper bound of a candidate pattern X falls below *minsup*, then X is immediately pruned.

Tree-Based Mining of Uncertain Data

Tree-based mining algorithms avoid the candidate generate-and-test mining paradigm used in the Apriori-based mining algorithms. Instead, tree-based algorithms use a depth-first divide-and-conquer approach to mine frequent patterns from a tree structure that captures the contents of the databases. To mine frequent patterns from uncertain data, Leung et al.[24] proposed a tree-based algorithm called *UF-growth*.

Similar to the FP-growth algorithm[25,26] (for mining traditional transaction databases of precise data), UF-growth leads to the construction of a tree structure to capture the contents of the databases. However, it does not use the FP-tree (as in the FP-growth algorithm) because each node in the FP-tree only maintains (1) an item and (2) its occurrence count in the tree path. See Figure 4, which shows the FP-tree capturing the contents of the traditional database D_1 of precise data shown in Figure 1. For traditional transaction databases of precise data, the actual support of a pattern X depends solely on the occurrence counts of items within X. However, for probabilistic databases of uncertain data, the

Items	Actual support
a	4
b	3
c	2
d	2

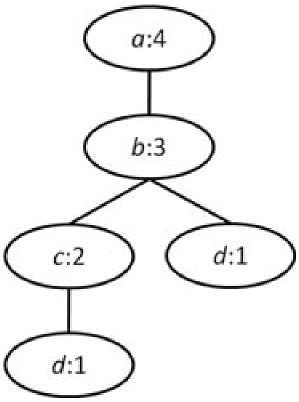


FIGURE 4 | An FP-tree for capturing the contents of D_1.

expected support of X is the sum of the product of the occurrence count and existential probability of every item within X. Hence, Leung et al. extended the FP-tree to capture the contents of probabilistic databases of uncertain data. The resulting tree structure is called *UF-tree*. Each node in the UF-tree consists of three components: (1) an item, (2) its existential probability, and (3) its occurrence count in the path. Figure 5 shows a UF-tree capturing the contents of the probabilistic database D_2 of uncertain data shown in Figure 2. Such a UF-tree is constructed in a similar fashion as the FP-tree, except that a new transaction is merged with a child node only if the same item and the same existential probability exist in both the transaction and the child node. As such, it may lead to a lower compression ratio than the original FP-tree. Fortunately, the number of nodes in a UF-tree is bounded above by the sum of the number of items in all transactions in the probabilistic database of uncertain data. Moreover, Leung et al.[27] also proposed two improvement techniques to reduce the memory consumption. First, they discretized the existential probability of each node (e.g., rounded the existential probability to k decimal places such as $k = 2$), which reduces the potentially infinite number of possible existential probability values to a maximum of 10^k possible values. Second, during the process of mining uncertain data, Leung et al. limited the construction of UF-trees to only the first two levels (i.e., only constructed the global UF-tree for the original D and a UF-tree for each frequent singleton pattern) and enumerated frequent patterns for higher levels (by traversing the tree paths and decrementing the occurrence counts). Figures 5 and 6 show the global UF-tree for D_2 (in Figure 2) and the UF-tree for the frequent singleton pattern $\{d\}$, respectively.

Items	Expected support
a	2.8
b	2.7
c	1.0
d	1.0

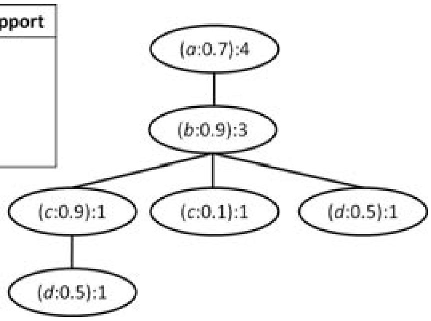


FIGURE 5 | The global UF-tree for capturing the contents of D_2 (for mining all frequent patterns).

Itemsets	Expected support
{*b*,*d*}	(0.9 x 2) x **0.5** = 0.9
{*a*,*d*}	(0.7 x 2) x **0.5** = 0.7
{*c*,*d*}	(0.9 x 1) x **0.5** = 0.45

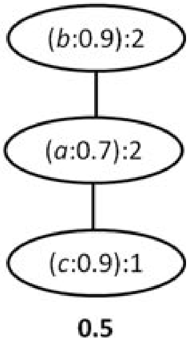
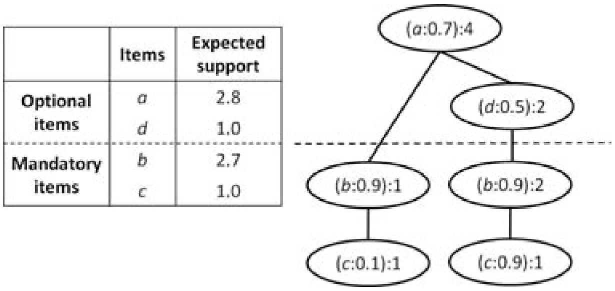


FIGURE 6 | A UF-tree for capturing the contents of $\{d\}$-projected database for D_2 (i.e., contents of only transactions containing the singleton pattern $\{d\}$).

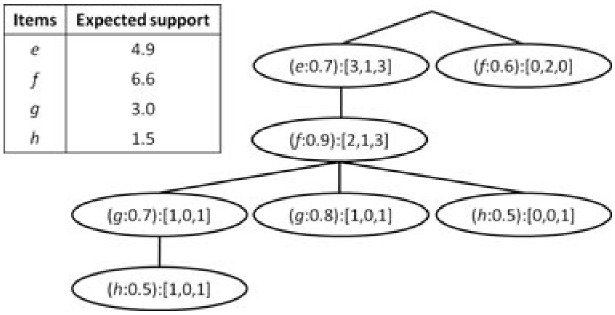
Tree-Based Constrained Mining of Uncertain Data

While the UF-growth algorithm finds all the frequent patterns from probabilistic databases of uncertain data, there are situations in which users are interested in only some of the frequent patterns. In these situations, users express their interest in terms of constraints. This leads to *constrained mining*.[28–30] Leung et al.[31–33] extended the UF-growth algorithm to mine uncertain data for frequent patterns that satisfy user-specified constraints. The two resulting algorithms, called *U-FPS*[31,32] and *U-FIC*,[33] push the constraints in the mining process and exploit properties of different kinds of constraints (instead of a naïve approach of first mining all frequent patterns and then pruning all uninteresting or invalid ones). For instance, U-FPS exploits properties of *succinct constraints*.[34,35] More specifically, by exploiting that 'all patterns satisfying any *succinct and antimonotone* (*SAM*) *constraint* C_{SAM} must comprise only items that individually satisfy C_{SAM}', U-FPS stores only these items in the UF-tree when handling C_{SAM}. Similarly, by exploiting that 'all patterns satisfying any ***succinct but*** *not antimonotone* (*SUC*) *constraint* C_{SUC} consist of at least one item that individually satisfies C_{SUC} and may



	Items	Expected support
Optional items	*a*	2.8
	d	1.0
Mandatory items	*b*	2.7
	c	1.0

FIGURE 7 | The global UF-tree for capturing the contents of D_2 (for mining frequent patterns that satisfy a SUC constraint C_{SUC}).



Items	Expected support
e	4.9
f	6.6
g	3.0
h	1.5

FIGURE 8 | An SUF-tree (with a sliding window of $w = 3$ batches) for capturing the contents of D_3.

contain other items', U-FPS partitions the domain items into two groups (one group contains items individually satisfying C_{SUC} and another group contains those not) and stores items belonging to each group separately in the UF-tree. See Figure 7 on how U-FPS stores the two groups in a UF-tree for D_2 shown in Figure 2. As arranging domain items in decreasing order of their support in the original FP-tree is a just heuristic, U-FIC exploits properties of *convertible constraints*[36–38] and arranges the domain items in the UF-tree according to some monotonic order of attribute values relevant to the constraints. By doing so, U-FIC does not need to perform constraint checking against any extensions of patterns satisfying any *convertible monotone* (*COM*) *constraint* C_{COM} because all these extensions are guaranteed to satisfy C_{COM}. Similarly, U-FIC prunes all the patterns that violate any *convertible antimonotone* (*CAM*) *constraint* C_{CAM} because these patterns and their extensions are guaranteed to violate C_{CAM}. By exploiting the user-specified constraints, computation of both U-FPS and U-FIC is proportional to the selectivity of the constraints.

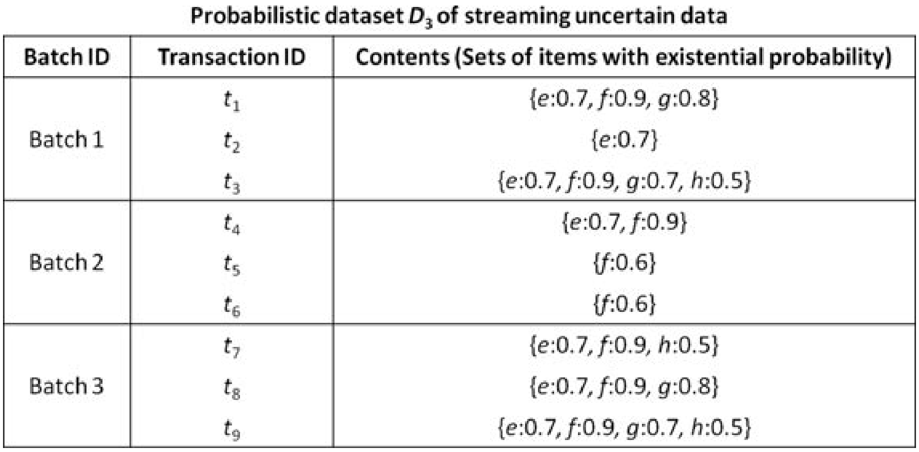
Tree-Based Stream Mining of Uncertain Data

With advances in technology, streams of uncertain data can be generated (e.g., by wireless sensors in applications like environment surveillance). This leads to *stream mining*.[39–45] Leung and Hao[46] extended the UF-growth algorithm and called the resulting algorithm *SUF-growth*, which mines frequent patterns from streams of uncertain data. When using a sliding window model, SUF-growth captures only the contents of streaming data in batches belonging to the current window (of size w batches) in a tree structure called *SUF-tree*. When the window slides, SUF-growth removes from the SUF-tree the data belonging to older batches and adds to the SUF-tree the data belonging to newer batches. Hence, each tree node in the SUF-tree consists of three components: (1) an item, (2) its existential probability, and (3) a list of its w occurrence counts in the path. By doing so, when the window slides, the oldest occurrence counts (representing the oldest streaming data) are replaced by the newest occurrence counts (representing the newest streaming data). Figure 8 shows an SUF-tree capturing the contents of the streaming data in Figure 9. Such an SUF-tree is constructed in a similar fashion as the construction of the UF-tree, except that the occurrence count is inserted as the newest entry in the list of occurrence counts.

Hyperlinked-Structure-Based Mining of Uncertain Data

An alternative to tree-based mining is hyperlinked-structure-based mining, which also employs a pattern-growth mining paradigm to avoid generating a large number of candidates. Instead of constructing many trees and mining frequent patterns from these trees, hyperlinked-array-based mining algorithms capture the contents of the databases in a hyperlinked structure called *H-struct*[47,48] and mine frequent patterns from the H-struct.

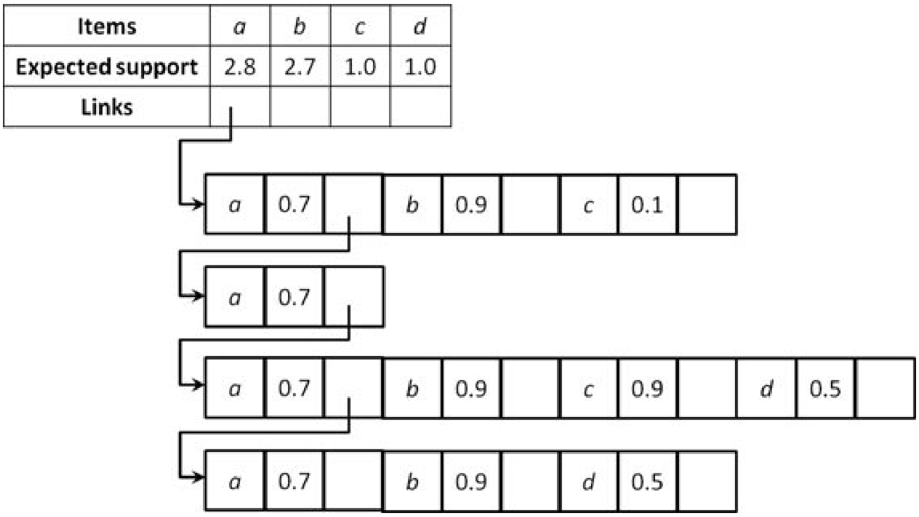
To mine frequent patterns from uncertain data, Aggarwal et al.[49,50] extended H-mine algorithm[47,48] (which mines frequent patterns from traditional transaction databases of precise data) and its corresponding H-struct. The resulting algorithm is called *UH-mine*. Like the original H-struct, each row in the extended H-struct represents a transaction t_i in the database. However, unlike the original H-struct, the extended H-struct maintains the existential probability $P(x,t_i)$ of item x in t_i (in addition to x and its hyperlink). See Figure 10 for how an extended H-struct stores the contents of D_2 shown in Figure 2. The UH-mine algorithm mines frequent patterns by recursively extending every frequent pattern X and adjusting its hyperlinks in the extended H-struct. Although the extended H-struct is not as compact as the UF-tree (used by the UF-growth algorithm), UH-mine keeps only



Probabilistic dataset D_3 of streaming uncertain data

Batch ID	Transaction ID	Contents (Sets of items with existential probability)
Batch 1	t_1	{e:0.7, f:0.9, g:0.8}
	t_2	{e:0.7}
	t_3	{e:0.7, f:0.9, g:0.7, h:0.5}
Batch 2	t_4	{e:0.7, f:0.9}
	t_5	{f:0.6}
	t_6	{f:0.6}
Batch 3	t_7	{e:0.7, f:0.9, h:0.5}
	t_8	{e:0.7, f:0.9, g:0.8}
	t_9	{e:0.7, f:0.9, g:0.7, h:0.5}

FIGURE 9 | A probabilistic dataset D_3 containing streams of uncertain data.



Items	a	b	c	d
Expected support	2.8	2.7	1.0	1.0
Links				

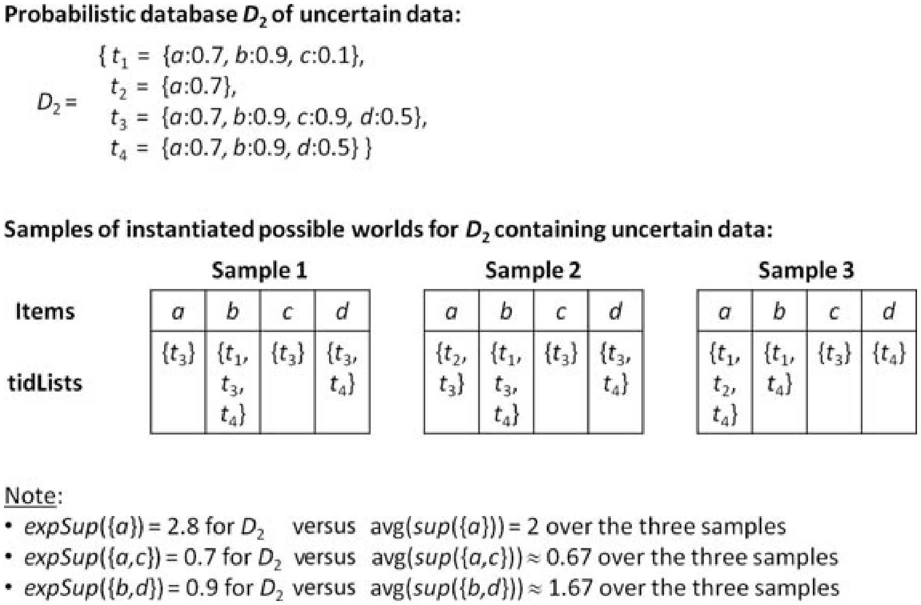
FIGURE 10 | An extended H-struct for capturing the contents of D_2.

one extended H-struct and adjusts the hyperlinks in it (instead of constructing more than one UF-tree as required by UF-growth). Hence, it drastically reduces the memory space requirement. Moreover, the algorithm computes the expected support of X on-the-fly so as to further reduce the space requirement.

Vertical Mining of Uncertain Data

The Apriori-based, tree-based, as well as hyperlinked-structure-based mining algorithms use *horizontal mining*, for which a database can be viewed as a collection of transactions. Each transaction is a set of items. Alternatively, *vertical mining* can be applied, for which each database can be viewed as a collection of items and their associated *lists of transaction IDs* (which are also known as *tidLists*). Each tidList of an item x represents all the transactions containing x. With this vertical representation of databases, the support of a pattern X can be computed by intersecting the tidLists of items within X.

To mine frequent patterns using the vertical representation of probabilistic databases containing uncertain data, Calders et al.[51] instantiated 'possible worlds' of the databases and then applied the Eclat algorithm[52] to each of these samples of instantiated databases. The resulting algorithm is called *U-Eclat*. Given a probabilistic database D of uncertain data, U-Eclat generates an independent random number r for each item x in a transaction t_i. If the existential probability $P(x,t_i)$ of item x in transaction t_i is no less than such a random number r (i.e., $P(x,t_i) \geq r$), then x is instantiated and included in a 'certain' sampled database, which is then mined using the original Eclat algorithm. This sampling and instantiation process is repeated multiple times, and thus generates



Probabilistic database D_2 of uncertain data:

$$D_2 = \begin{array}{l} \{\, t_1 = \{a{:}0.7, b{:}0.9, c{:}0.1\}, \\ \;\; t_2 = \{a{:}0.7\}, \\ \;\; t_3 = \{a{:}0.7, b{:}0.9, c{:}0.9, d{:}0.5\}, \\ \;\; t_4 = \{a{:}0.7, b{:}0.9, d{:}0.5\} \,\} \end{array}$$

Samples of instantiated possible worlds for D_2 containing uncertain data:

	Sample 1				**Sample 2**				**Sample 3**			
Items	a	b	c	d	a	b	c	d	a	b	c	d
tidLists	$\{t_3\}$	$\{t_1, t_3, t_4\}$	$\{t_3\}$	$\{t_3, t_4\}$	$\{t_2, t_3\}$	$\{t_1, t_3, t_4\}$	$\{t_3\}$	$\{t_3, t_4\}$	$\{t_1, t_2, t_4\}$	$\{t_1, t_4\}$	$\{t_3\}$	$\{t_4\}$

Note:

- $expSup(\{a\}) = 2.8$ for D_2 versus avg$(sup(\{a\})) = 2$ over the three samples
- $expSup(\{a,c\}) = 0.7$ for D_2 versus avg$(sup(\{a,c\})) \approx 0.67$ over the three samples
- $expSup(\{b,d\}) = 0.9$ for D_2 versus avg$(sup(\{b,d\})) \approx 1.67$ over the three samples

FIGURE 11 | Some samples of instantiated 'possible worlds' of D_2.

multiple sampled 'certain' databases. The estimated support of any pattern X is the average support of X over the multiple sampled databases. Figure 11 shows three sampled databases for probabilistic database D_2 (shown in Figure 2). As a sampling-based algorithm, U-Eclat gains efficiency but loses accuracy. More instantiations (i.e., more sampled databases) helps improve accuracy, but it comes at the cost of an increase in execution time.

Discussion

So far, we have reviewed various algorithms for mining uncertain data with probabilistic setting. Table 1 shows some key differences among these algorithms. In terms of functionality, the U-Apriori, UF-growth, UH-mine, and U-Eclat algorithms all mine static databases of uncertain data, whereas SUF-growth mines dynamic streams of uncertain data. Unlike these five algorithms that find all frequent patterns, both U-FPS and U-FIC algorithms find only those frequent patterns satisfying the user-specified constraints.

In terms of accuracy, all these seven algorithms except U-Eclat return all the patterns with expected support (over all 'possible worlds') meeting or exceeding the user-specified threshold *minsup*. In contrast, U-Eclat returns patterns with estimated support (over only the sampled 'possible worlds') meeting or exceeding *minsup*. Hence, U-Eclat may introduce false positives (when the support is overestimated) or false negatives (when the support is underestimated). More instantiations (i.e., more samples) helps improve accuracy.

In terms of memory consumption, U-Apriori keeps a list of candidate patterns, whereas the tree-based and hyperlinked-structure-based algorithms construct in-memory structures (e.g., UF-trees, extended H-struct). On the one hand, a UF-tree is more compact (i.e., requires less space) than the extended H-struct. On the other hand, UH-mine keeps only one extended H-struct, whereas tree-based algorithms usually construct more than one tree. Sizes of the trees may also vary. For instance, when U-FPS handles a succinct and antimonotone constraint C_{SAM}, the tree size depends on the selectivity of C_{SAM} because only those items that individually satisfy C_{SAM} are stored in the UF-tree. When SUF-growth handles streams, the tree size depends on the size of sliding window (e.g., a window of w batches) because a list of w occurrence counts is captured in each node of SUF-trees (cf. only one occurrence count is captured in each node of UF-trees). Moreover, when items in probabilistic databases take on a few distinct existential probability values, the trees contain fewer nodes (cf. the number of distinct existential probability values does not affect the size of candidate lists or the extended H-struct). Furthermore, *minsup* and density also affect memory consumption. For instance, for a sparse dataset called kosarak, different winners (requiring the least space) had been shown for different *minsup* values: U-Apriori when *minsup* $< 0.15\%$, UH-mine when $0.15\% \leq$ *minsup* $< 0.5\%$, and

TABLE 1 | Comparison of the Four Uncertain Data Mining Algorithms

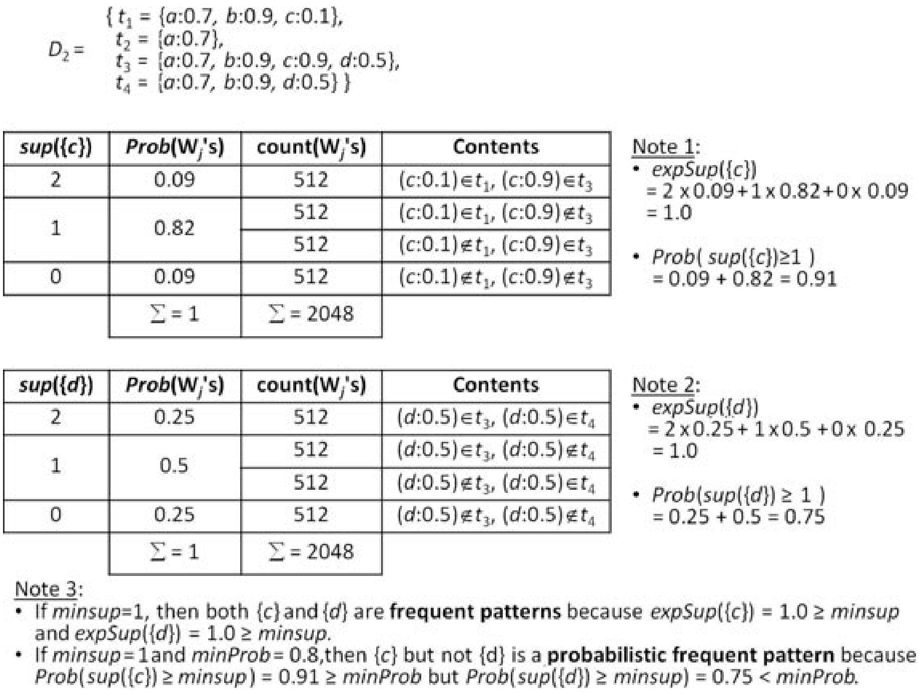
U-Apriori[19,23]	UF-growth[24,27]	UH-mine[49,50]	U-Eclat[51]
Horizontal mining (Apriori-based)	Horizontal mining (Tree-based)	Horizontal mining (Hyperlinked structure based)	Vertical mining
Extension of Apriori[20,21]	Extension of FP-growth[25,26]	Extension of H-mine[47,48]	Application of Eclat[52]
Candidate generate-and-test paradigm	Pattern growth paradigm	Pattern growth paradigm	Equivalence class transformation paradigm
Push uncertain mining inside the mining process	Push uncertain mining inside the mining process	Push uncertain mining inside the mining process	Apply precise mining on sampled databases
Keep lists of candidate patterns	Construct one or more UF-trees	Construct an extended H-struct	Obtain samples of instantiated 'possible worlds'
Scan DB k times (where $k =$ maximum cardinality of frequent patterns)	Scan DB twice	Scan DB twice	Scan DB s times (where $s =$ number of samples)
Return all and only those patterns with expected support (i.e., support over all 'possible worlds') $\geq$ *minsup*; no false positives or false negatives	Return all and only those patterns with expected support (i.e., support over all 'possible worlds') $\geq$ *minsup*; no false positives or false negatives	Return all and only those patterns with expected support (i.e., support over all 'possible worlds') $\geq$ *minsup*; no false positives or false negatives	Return patterns with estimated support (i.e., average support over only sampled 'possible worlds') $\geq$ *minsup*; may introduce false positives and/or false negatives
#distinct existential probability values do not affect the size of candidate pattern lists	Fewer distinct existential probability values lead to smaller UF-trees	#distinct existential probability values do not affect the size of the extended H-struct	#distinct existential probability values do not affect the number of samples
	Has been extended for *constrained mining* (e.g., U-FPS[31,32] and U-FIC[33] algorithms) and *stream mining* (e.g., SUF-growth algorithm[46])		

UF-growth when 0.5% $\leq$ *minsup*; for a dense dataset called connect4, UH-mine was the winner for 0.2% $\leq$ *minsup* < 0.8%.[49,50]

In terms of performance, most algorithms perform well when items in probabilistic databases take on low existential probability values because these databases do not lead to long frequent patterns. When items in probabilistic databases take on high existential probability values, more candidates are generated-and-tested by U-Apriori, more and bigger UF-trees are constructed by UF-growth, more hyperlinks are adjusted by UH-mine, and more estimated supports are computed by U-Eclat. Hence, longer runtimes are required. Similarly, when *minsup* decreases, more frequent patterns are returned and longer runtimes are also required. The density of datasets also affects runtimes. For instance, when databases are dense (e.g., connect4), UF-trees lead to higher compression ratio and thus require less time to traverse than sparse databases (e.g., kosarak). Some experimental results showed the following: (1) databases with a low number of distinct existential probabilities led to smaller UF-trees and shorter runtime for UF-growth (than U-Apriori)[24,27]; (2) U-Apriori took shorter runtime than UH-mine when *minsup* was low (e.g., *minsup* < 0.3% for kosarak, *minsup* < 0.6% for connect4) but vice versa when *minsup* was high[49,50]; (3) depending on the number of samples, U-Eclat could take longer or shorter to run than U-Apriori.[51]

PROBABILISTIC FREQUENT PATTERN MINING OF UNCERTAIN DATA

The aforementioned algorithms—namely, the U-Apriori, UF-growth, U-FPS, U-FIC, SUF-growth, UH-mine, and U-Eclat algorithms—all mine uncertain data for frequent patterns. These are patterns



$D_2 = \{ t_1 = \{a{:}0.7,\ b{:}0.9,\ c{:}0.1\},$
$t_2 = \{a{:}0.7\},$
$t_3 = \{a{:}0.7,\ b{:}0.9,\ c{:}0.9,\ d{:}0.5\},$
$t_4 = \{a{:}0.7,\ b{:}0.9,\ d{:}0.5\} \}$

sup({*c*})	*Prob*(W$_j$'s)	count(W$_j$'s)	Contents
2	0.09	512	$(c{:}0.1) \in t_1, (c{:}0.9) \in t_3$
1	0.82	512	$(c{:}0.1) \in t_1, (c{:}0.9) \notin t_3$
		512	$(c{:}0.1) \notin t_1, (c{:}0.9) \in t_3$
0	0.09	512	$(c{:}0.1) \notin t_1, (c{:}0.9) \notin t_3$
	$\sum = 1$	$\sum = 2048$	

Note 1:
- *expSup*({*c*}) = 2 x 0.09 + 1 x 0.82 + 0 x 0.09 = 1.0
- *Prob*(*sup*({*c*})≥1) = 0.09 + 0.82 = 0.91

sup({*d*})	*Prob*(W$_j$'s)	count(W$_j$'s)	Contents
2	0.25	512	$(d{:}0.5) \in t_3, (d{:}0.5) \in t_4$
1	0.5	512	$(d{:}0.5) \in t_3, (d{:}0.5) \notin t_4$
		512	$(d{:}0.5) \notin t_3, (d{:}0.5) \in t_4$
0	0.25	512	$(d{:}0.5) \notin t_3, (d{:}0.5) \notin t_4$
	$\sum = 1$	$\sum = 2048$	

Note 2:
- *expSup*({*d*}) = 2 x 0.25 + 1 x 0.5 + 0 x 0.25 = 1.0
- *Prob*(*sup*({*d*}) ≥ 1) = 0.25 + 0.5 = 0.75

Note 3:
- If *minsup*=1, then both {*c*} and {*d*} are **frequent patterns** because *expSup*({*c*}) = 1.0 ≥ *minsup* and *expSup*({*d*}) = 1.0 ≥ *minsup*.
- If *minsup*=1and *minProb*=0.8,then {*c*} but not {d} is a **probabilistic frequent pattern** because *Prob*(*sup*({*c*}) ≥ *minsup*) = 0.91 ≥ *minProb* but *Prob*(*sup*({*d*}) ≥ *minsup*) = 0.75 < *minProb*.

FIGURE 12 | Frequent patterns mined from D_2 based on expected support and probabilistic support.

with expected support meeting or exceeding the user-specified threshold *minsup*. Note that expected support of a pattern X provides users with frequency information of X summarized over all 'possible worlds', but it does not reveal the confidence on the likelihood of X being frequent (i.e., percentage of 'possible worlds' in which X is frequent). However, knowing the confidence can be helpful in some applications. Hence, in recent years, there is also algorithmic development on extending the notion of frequent patterns based on expected support to useful patterns—such as probabilistic heavy hitters, probabilistic frequent patterns, and probabilistic association rules—based on probabilistic support as reviewed below. Figure 12 illustrates the differences between expected support and probabilistic support.

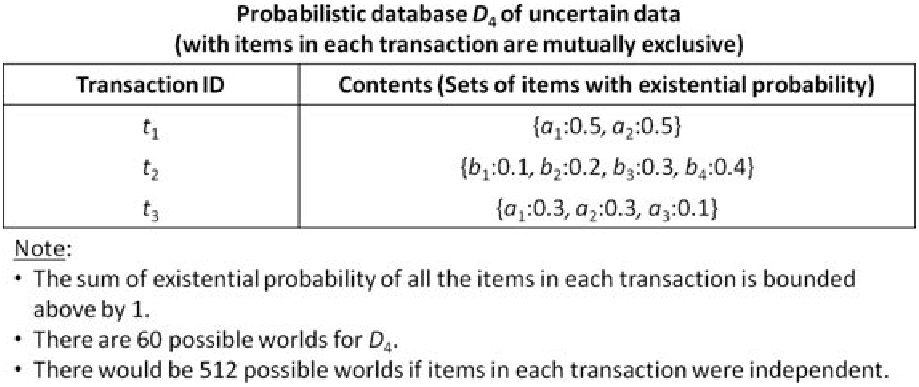
Mining Probabilistic Heavy Hitters

The expected support of an item x (i.e., a singleton pattern x) provides users with an estimate of the frequency of x. However, in some applications, it is also helpful to know the confidence about the likelihood of x being frequent in the uncertain data. Hence, Zhang et al.[53] formalized the notion of *probabilistic heavy hitters* (i.e., *probabilistic frequent items*, which are also known as *probabilistic frequent singleton patterns*) following the 'possible world' semantics[54] for probabilistic databases of uncertain data. Given (1) a probabilistic database D of uncertain data, (2) a user-specified support threshold φ, and (3) a user-specified frequentness probability threshold τ, the research problem of *mining probabilistic heavy hitters* from uncertain data is to find all (φ,τ)-probabilistic heavy hitters (PHHs). An item x is a (φ,τ)-PHH if $P(sup(x,\mathrm{W}_j) > \varphi|\mathrm{W}_j|) > \tau$ (where $sup(x,\mathrm{W}_j)$ is the support of x in a random possible world W_j and $|\mathrm{W}_j|$ is the number of items in W_j), which represents the probability of x being frequent exceeds the user expectation. Equivalently, given (1) D, (2) a user-specified support threshold *minsup*, (3) a user-specified frequentness probability threshold *minProb*, an item x is a PHH if x is highly likely to be frequent, i.e., the probability that x occurs in at least *minsup* transactions of D is no less than *minProb*: $P(sup(x) \geq minsup) > minProb$.

To find these probabilistic heavy hitters from probabilistic databases of uncertain data (where items in each transaction are mutually inclusive) such as D_4 shown in Figure 13, Zhang et al. proposed two algorithms: an exact algorithm and an approximate algorithm. The exact algorithm uses dynamic programming to mine offline uncertain data for PHHs. Such an algorithm runs in polynomial time when there is sufficient memory. When the memory is limited, the approximate algorithm can be applied (which uses sampling techniques) to mine streams of uncertain data for approximate PHHs.

Mining Probabilistic Frequent Patterns

The expected support of a pattern X (that consists of one or more items) provides users with an estimate



Probabilistic database D_4 of uncertain data
(with items in each transaction are mutually exclusive)

Transaction ID	Contents (Sets of items with existential probability)
t_1	$\{a_1{:}0.5, a_2{:}0.5\}$
t_2	$\{b_1{:}0.1, b_2{:}0.2, b_3{:}0.3, b_4{:}0.4\}$
t_3	$\{a_1{:}0.3, a_2{:}0.3, a_3{:}0.1\}$

Note:

- The sum of existential probability of all the items in each transaction is bounded above by 1.
- There are 60 possible worlds for D_4.
- There would be 512 possible worlds if items in each transaction were independent.

FIGURE 13 | A probabilistic database D_4 of uncertain data, in which items in each transaction are mutually exclusive.

of the frequency of X, but it does not take into account the variance or the probability distribution of the support of X. In some applications, knowing the confidence on which pattern is highly likely to be frequent helps interpreting patterns mined from uncertain data. Hence, Bernecker et al.[55] extended the notion of frequent patterns and introduced the research problem of mining probabilistic frequent patterns (p-FPs). Given (1) a probabilistic database D of uncertain data, (2) a user-specified support threshold *minsup*, (3) a user-specified frequentness probability threshold *minProb*, the research problem of *mining probabilistic frequent patterns* from uncertain data is to find (1) all patterns that are highly likely to be frequent and (2) their support. Here, the support $sup(X)$ of any pattern X is defined by a discrete probability distribution function (pdf) or probability mass function (pmf). A pattern X is *highly likely to be frequent* (i.e., X is a *probabilistic frequent pattern*) if and only if its frequentness probability is no less than *minProb*, i.e., $P(sup(X) \geq minsup) \geq minProb$. The *frequentness probability* of X is the probability that X occurs in at least *minsup* transactions of D. Note that frequentness probability is antimonotonic: All subsets of a p-FP are also p-FPs. Equivalently, if X is not a p-FP, then none of its supersets is a p-FP, and thus all of them can be pruned. Moreover, when *minsup* increases, frequentness probabilities of p-FPs decrease.

Bernecker et al.[55] used a dynamic computation technique in computing probability function $f_X(k) = P(sup(X) = k)$, which returns the probability that the support of a pattern X equals to k. Summing the values of such a probability function $f_X(k)$ over all $k \geq minsup$ gives the frequentness probability of X because $\sum_{k \geq minsup}^{|D|} f_X(k) = \sum_{k \geq minsup}^{|D|} P(sup(X) = k) = P(sup(X) \geq minsup)$. Any pattern X having the sum no less than *minProb* becomes a probabilistic frequent pattern.

Sun et al.[56] proposed the top-down inheritance of support probability function (TODIS) algorithm, which runs in conjunction with a divide-and-conquer (DC) approach, to mine probabilistic frequent patterns from uncertain data by extracting patterns that are supersets of p-FPs and deriving p-FPs in a top–down manner (i.e., descending cardinality of p-FPs).

Mining Probabilistic Association Rules

Along the direction of extending the notion of frequent patterns to the notion of probabilistic frequent patterns, Sun et al.[56] introduced the research problem of mining probabilistic association rules (p-ARs). Given (1) a probabilistic database D of uncertain data, (2) a user-specified support threshold *minsup*, (3) a user-specified confidence threshold *minconf*, and (4) a user-specified frequentness probability threshold *minProb*, the research problem of *mining probabilistic association rules* from uncertain data is to find all rules that are highly likely to be interesting. Here, the supports of probabilistic frequent patterns A and C in the antecedent and the consequent of an association rule of the form '$A \rightarrow C$' are defined by discrete pdfs or pmfs. A rule '$A \rightarrow C$', where probabilistic frequent patterns A and C are disjoint (i.e., $A \cap C = \emptyset$), is a *probabilistic association rule* (i.e., '$A \rightarrow C$' is highly likely to be interesting) if and only if its probability is no less than *minProb*, i.e.,

$$\begin{aligned} P(A \rightarrow C) = {} & P(sup(A \rightarrow C) \geq minsup \text{ AND} \\ & confidence(A \rightarrow C) \geq minconf) \\ = {} & P(sup(A \cup C) \geq minsup \text{ AND} \\ & sup(A \cup C)/sup(A) \geq minconf) \\ \geq {} & minProb. \end{aligned}$$

To check whether '$A \rightarrow C$' is a p-AR, Sun et al. computed the probability $P(A \rightarrow C)$ and compared

TABLE 2 | Comparison of Uncertain Data Mining

	Data Source	Additional Input Parameters	Output Results
Frequent pattern mining[19,23,24,27,49–51]	Probabilistic database *D* of uncertain data	*minsup*	Frequent patterns, i.e., patterns with expected support $expSup(X) \geq minsup$
Constrained mining[31–33]	Probabilistic database *D* of uncertain data	*minsup* and constraints (e.g., C_{SAM}, C_{SUC}, C_{COM}, C_{CAM})	Frequent patterns satisfying constraints
Stream mining[46]	Stream of uncertain data	*minsup*	Frequent patterns
Probabilistic heavy hitter mining[53]	Probabilistic database *D* of uncertain data	*minsup* and *minProb*	Probabilistic heavy hitter (PHH), i.e., items with $P(sup(x) \geq minsup) > minProb$
Probabilistic frequent pattern mining[55,56]	Probabilistic database *D* of uncertain data	*minsup* and *minProb*	Probabilistic frequent patterns (p-FPs), i.e., patterns with $P(sup(X) \geq minsup) > minProb$
Probabilistic association rule mining[56]	Probabilistic database *D* of uncertain data	*minsup, minconf,* and *minProb*	Probabilistic association rules, i.e., patterns with $P(sup(A \rightarrow C) \geq minsup$ AND $conf(A \rightarrow C) \geq minconf) > minProb$

it against *minProb*. If the probability is no less than *minProb*, such a p-AR is returned to users. To speed up the mining process, the antimonotonic property of p-AR is exploited. Specifically, given three probabilistic frequent patterns X_1, X_2 and X_3, such that $X_1 \subset X_2 \subset X_3$, if '$(X_3 - X_2) \rightarrow X_2$' is a p-AR, then '$(X_3 - X_1) \rightarrow X_1$' is also a p-AR. Equivalently, if '$(X_3 - X_1) \rightarrow X_1$' is not a p-AR, then for every superset S of X_1 (i.e., $X_1 \subset S \subset X_3$), '$(X_3 - S) \rightarrow S$' is also not a p-AR. Hence, all ARs having superset of X_1 as the consequent can be pruned.

CONCLUSION

Association rule mining is an important DMKD task. It consists of the mining of frequent patterns from data and the formation of association rules using the mined frequent patterns. As the mining of frequent patterns is usually more computationally intensive than the formation of association rule, it has drawn attention of many researchers over the past two decades. The research problem of frequent pattern mining was originally proposed to analyze shopping market basket transaction databases containing precise data, in which the contents of transactions in the databases are known. Such a research problem also plays important role in other DMKD tasks, such as the mining of interesting or unexpected patterns, sequential mining, associative classification, as well as outlier detection, in various real-life applications. Recently, researchers have paid more attention to the mining of frequent patterns from probabilistic databases of uncertain data.

In this paper, we reviewed recent algorithmic development on mining frequent patterns from uncertain data with probabilistic setting. We studied (1) frequent pattern mining of uncertain data and (2) probabilistic pattern mining of uncertain data. See Table 2 for a brief summary. To mine frequent patterns from uncertain data, researchers have proposed Apriori-based, tree-based, hyperlinked-structure-based, and vertical frequent pattern mining algorithms. Among them, the U-Apriori algorithm generates candidate patterns and tests if their expected support meets or exceeds a user-specified threshold. To avoid such a candidate generate-and-test approach, both UF-growth and UH-mine algorithms use a pattern growth mining approach. The UF-growth algorithm constructs an UF-tree and mines frequent patterns from it; UH-mine keeps an extended H-struct and mines frequent patterns from it. Instead of applying horizontal mining, U-Eclat uses vertical mining. It vertically mines frequent patterns from multiple instantiated sampled possible worlds of uncertain data. Moreover, researchers have also extended the UF-growth algorithm for constrained mining and stream mining. The resulting U-FPS and U-FIC algorithms exploit properties of the user-specified succinct constraints and convertible constraints, respectively, to find from uncertain data only those frequent patterns satisfying the constraints. The SUF-growth algorithm uses a sliding window to all frequent patterns from an SUF-tree, which captures the contents of current few batches of streaming uncertain data. Recently, researchers further extended the initial notion of mining frequent patterns from uncertain data based on the expected support of patterns to the notions of mining useful probabilistic patterns such as probabilistic heavy hitters (i.e., probabilistic frequent items), probabilistic frequent patterns, as well as probabilistic association rules, based on the confidence on the likelihood of the patterns being useful. The corresponding algorithms have been proposed to mine uncertain data for items that are highly likely to be frequent, multi-item patterns that are highly likely to be frequent, as well as association rules that are highly likely to be interesting. Future research directions include mining uncertain data for frequent sequences and frequent graphs as well as mining uncertain data in applications areas like bioinformatics.

REFERENCES

1. Frawley WJ, Piatetsky-Shapiro G, Matheus CJ. Knowledge discovery in databases: an overview. In: Frawley WJ, Piatetsky-Shapiro G, eds. *Knowledge Discovery in Databases*. Cambridge, MA: AAAI/MIT Press; 1991, 1–30.
2. Prabhakar S, Cheng R. Data uncertainty management in sensor networks. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 647–651.
3. Suciu D. Probabilistic databases. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 2150–2155.
4. Wasserkrug S. Uncertainty in events. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 3221–3225.
5. Dalvi N. Uncertainty management in scientific database systems. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 3225–3231.
6. Kriegel HP, Pfeifle M. Density-based clustering of uncertain data. In: Grossman R, Bayardo RJ, Bennett KP, eds. *Proceedings of the KDD*. New York, NY: ACM Press; 2005, 672–677.
7. Cormode G, McGregor A. Approximation algorithms for clustering uncertain data. In: Lenzerini M, Lembo D, eds. *Proceedings of the ACM PODS*. New York, NY: ACM Press; 2008, 191–200.
8. Kao B, Lee SD, Lee FKF, Cheung DW, Ho WS. Clustering uncertain data using Voronoi diagrams and R-tree index. IEEE Trans Knowl Data Eng 2010, 22: 1219–1233. doi:10.1109/TKDE.2010.82.
9. Ren J, Lee SD, Chen X, Kao B, Cheng R, Cheung D. Naive Bayes classification of uncertain data. In: Wang W, Kargupta H, Ranka S, Yu PS, Wu X, eds. *Proceedings of the IEEE ICDM*. Los Alamitos, CA: IEEE Computer Society; 2009, 944–949.
10. Qin B, Xia Y, Li F. A Bayesian classifier for uncertain data. In: Shin D, ed. *Proceedings of the ACM SAC*. New York, NY: ACM Press; 2010, 1010–1014.
11. Aggarwal CC, Yu PS. Outlier detection with uncertain data. In: Wang W, ed. *Proceedings of the SDM*. Philadelphia, PA: SIAM; 2008, 483–493.
12. Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large databases. In: Buneman P, Jajodia S, eds. *Proceedings of the ACM SIGMOD*. New York, NY: ACM Press; 1993, 207–216.
13. Pei J. Association rules. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 140–141.
14. Cheng H, Han J. Frequent itemsets and association rules. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 1184–1187.
15. Abiteboul S, Kanellakis P, Grahne G. On the representation and querying of sets of possible worlds. In: Dayal U, Traiger IL, eds. *Proceedings of the ACM SIGMOD*. New York, NY: ACM Press; 1987, 34–48.
16. Green T, Tannen V. Models for incomplete and probabilistic information. In: Grust T, Höpfner H, Illarramendi A, Jablonski S, Mesiti M, Müller S, Patranjan PL, Sattler KU, Spiliopoulou M, Wijsen J, eds. *Proceedings of the EDBT Workshops*. LNCS, Vol. 4254. Berlin, Germany: Springer; 2006, 278–296.
17. Green T, Tannen V. Models for incomplete and probabilistic information. Bull Tech Committee Data Eng 2006, 29:17–24.
18. Dai X, Yiu ML, Mamoulis N, Tao Y, Vaitis M. Probabilistic spatial queries on existentially uncertain data. In: Medeiros CB, Egenhofer M, Bertino E, eds. *Proceeding of the SSTD*. LNCS, Vol. 3633. Berlin, Germany: Springer; 2005, 400–417.
19. Chui CK, Kao B, Hung E. Mining frequent itemsets from uncertain data. In: Zhou ZH, Li H, Yang Q, eds. *Proceeding of the PAKDD*. LNAI, Vol. 4426. Berlin, Germany: Springer; 2007, 47–58.
20. Agrawal R, Srikant R. Fast algorithms for mining association rules in large databases. In: Bocca JB, Jarke

M, Zaniolo C, eds. *Proceedings of the VLDB*.San Francisco, CA: Morgan Kaufmann; 1994, 487–499.

21. Agrawal R, Mannila H, Srikant R, Toivonen H, Verkamo AI. Fast discovery of association rules. In: Fayyad UM, Piatetsky-Shapiro G, Smyth P, Uthurusamy R, eds. *Advances in Knowledge Discovery and Data Mining*. Menlo Park, CA: AAAI/MIT Press; 1996, 307–328.
22. Goethals B. Apriori property and breadth-first search algorithms. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 124–127.
23. Chui CK, Kao B. A decremental approach for mining frequent itemsets from uncertain data. In: Washio T, Suzuki E, Ting KM, eds. *Proceeding of the PAKDD*. LNAI, Vol. 5012. Berlin, Germany: Springer; 2008, 64–75.
24. Leung CKS, Carmichael CL, Hao B. Efficient mining of frequent patterns from uncertain data. In: Tung AKH, Zhu Q, Ramakrishnan N, Zaïane OR, Shi Y, Clifton CW, Wu X, eds. *Proceedings of the IEEE ICDM Workshops*. Los Alamitos, CA: IEEE Computer Society; 2007, 489–494.
25. Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. In: Chen W, Naughton JF, Bernstein PA, eds. *Proceeding of the ACM SIGMOD*. New York, NY: ACM Press; 2000, 1–12.
26. Han J, Pei J, Yin Y, Mao R. Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Min Knowl Discov 2004, 8:53–87. doi:10.1023/B:DAMI.0000005258.31418.83.
27. Leung CKS, Mateo MAF, Brajczuk DA. A tree-based approach for frequent pattern mining from uncertain data. In: Washio T, Suzuki E, Ting KM, Inokuchi A, eds. *Proceedings of the PAKDD*. LNAI, Vol. 5012. Berlin, Germany: Springer; 2008, 653–661.
28. Leung CKS. Frequent itemset mining with constraints. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 1179–1183.
29. Ng RT, Lakshmanan LVS, Han J, Pang A. Exploratory mining and pruning optimizations of constrained associations rules. In: Haas LM, Tiwary A, eds. *Proceeding of the ACM SIGMOD*.New York, NY: ACM Press; 1998, 13–24.
30. Lakshmanan LVS, Leung CKS, Ng RT. Efficient dynamic mining of constrained frequent sets. ACM Trans Database Syst 2003, 28:337–389.
31. Leung CKS, Brajczuk DA. Efficient algorithms for mining constrained frequent patterns from uncertain data. In: Pei J, Getoor L, de Keijzer A, eds. *Proceeding of the U*. New York, NY: ACM Press; 2009, 9–18.
32. Leung CKS, Brajczuk DA. Efficient algorithms for the mining of constrained frequent patterns from uncertain data. SIGKDD Explor 2009, 11:123–130. doi:10.1145/1809400.1809425.
33. Leung CKS, Hao B, Brajczuk DA. Mining uncertain data for frequent itemsets that satisfy aggregate constraints. In: Shin D, ed. *Proceedings of the ACM SAC*. New York, NY: ACM Press; 2010, 1034–1038.
34. Leung CKS. Succinct constraints. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 2876.
35. Leung CKS, Lakshmanan LVS, Ng RT. Exploiting succinct constraints using FP-trees. SIGKDD Explor 2002, 4:40–49. doi:10.1145/568574.568581.
36. Leung CKS. Convertible constraints. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*.New York, NY: Springer; 2009, 494–495.
37. Pei J, Han J, Lakshmanan LVS. Mining frequent itemsets with convertible constraints. In: Buchmann A, Georgakopoulos D, eds. *Proceedings of the IEEE ICDE*. Los Alamitos, CA: IEEE Computer Society; 2001, 433–442.
38. Pei J, Han J, Lakshmanan LVS. Pushing convertible constraints in frequent itemset mining. Data Min Knowl Discov 2004, 8:227–252.
39. Giannella C, Han J, Pei J, Yan X, Yu PS. Mining frequent patterns in data streams at multiple time granularities. In: Kargupta H, Joshi A, Sivakumar K, Yesha Y, eds. *Data Mining: Next Generation Challenges and Future Directions*. Menlo Park, CA: AAAI/MIT Press; 2004, 105–124.
40. Gaber MM, Zaslavsky AB, Krishnaswamy S. Mining data streams: a review. SIGMOD Rec 2005, 34:18–26. doi:10.1145/1083784.1083789.
41. Leung CKS, Khan QI. DSTree: a tree structure for the mining of frequent sets from data streams. In: Clifton CW, Zhong N, Liu J, Wah BW, Wu X, eds. *Proceedings of the IEEE ICDM*. Los Alamitos, CA: IEEE Computer Society; 2006, 928–933.
42. Cormode G, Hadjieleftheriou M. Finding frequent items in data streams. In: Buneman P, Ooi BC, Ross K, Rastogi R, Milo T, Markl V, eds. *Proceedings of the VLDB*. New York, NY: VLDB Endowment; 2008, 1530–1541.
43. Yu PS, Chi Y. Association rule mining on streams. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 136–140.
44. Metwally A. Frequent items on streams. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 1175–1179.
45. Han J, Ding B. Stream mining. In: Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009, 2831–2834.
46. Leung CKS, Hao B. Mining of frequent itemsets from streams of uncertain data. In: Hristidis V, Yu S, eds. *Proceedings of the IEEE ICDE*. Los Alamitos, CA: IEEE Computer Society; 2009, 1663–1670.
47. Pei J, Han J, Lu H, Nishio S, Tang S, Yang D. H-Mine: hyper-structure mining of frequent patterns in large databases. In: Cercone N, Lin TY, Wu X, eds. *Proceedings of the IEEE ICDM*. Los Alamitos, CA: IEEE Computer Society; 2001, 441–448.

48. Pei J, Han J, Lu H, Nishio S, Tang S, Yang D. H-Mine: fast and space-preserving frequent pattern mining in large databases. IIE Trans 2007, 39:593–605. doi:10.1080/07408170600897460.
49. Aggarwal CC, Li Y, Wang J, Wang J. Frequent pattern mining with uncertain data. In: Melli G, ed. *Proceedings of the KDD*. New York, NY: ACM Press; 2009, 29–37.
50. Aggarwal CC, Li Y, Wang J, Wang J. Frequent pattern mining algorithms with uncertain data. In: Aggarwal CC, ed. *Managing and Mining Uncertain Data*. New York, NY: Springer; 2009, 427–459.
51. Calders T, Garboni C, Goethals B. Efficient pattern mining of uncertain data with sampling. In: Zaki MJ, Yu JX, Ravindran B, Pudi V, eds. *Proceedings of the PAKDD*. LNAI, Vol. 6118. Berlin, Germany: Springer; 2010, 480–487.
52. Zaki MJ, Parthasarathy S, Ogihara M, Li W. New algorithms for fast discovery of association rules. In: Heckerman D, Mannila H, Pregibon D, eds. *Proceedings of the KDD*. Menlo Park, CA: AAAI Press; 1997, 283–286.
53. Zhang Q, Li F, Yi K. Finding frequent items in probabilistic data. In: Wang J, ed. *Proceedings of the ACM SIGMOD*. New York, NY: ACM Press; 2008, 819–832.
54. Dalvi N, Suciu D. Efficient query evaluation on probabilistic databases. In: Nascimento MA, Özsu MT, Kossmann D, Miller RJ, Blakeley JA, Schiefer KB, eds. *Proceedings of the VLDB*. San Francisco, CA: Morgan Kaufmann; 2004, 864–875.
55. Bernecker T, Kriegel H-P, Renz M, Verhein F, Zuefle A. Probabilistic frequent itemset mining in uncertain databases. In: Melli G, ed. *Proceedings of the KDD*. New York, NY: ACM Press; 2009, 119–127.
56. Sun L, Cheng R, Cheung DW, Cheng J. Mining uncertain data with probabilistic guarantees. In: Rao B, Krishnapuram B, Tomkins A, Yang Q, eds. *Proceedings of the KDD*. New York, NY: ACM Press; 2010, 273–282.

FURTHER READING

Aggarwal CC. *Managing and Mining Uncertain Data*. New York, NY: Springer; 2009.

Aggarwal CC, Yu PS. A survey of uncertain data algorithms and applications. *IEEE Trans Knowl Data Eng* 2009; 21:609–623. doi:10.1109/TKDE.2008.190.

Cheng R, Chau M, Garofalakis M, Yu JX. Guest editors' introduction: special section on mining large uncertain probabilistic databases. *IEEE Trans Knowl Data Eng* 2010, 22:1201–1202. doi:10.1109/TKDE.2010.118.

Han J, Kamber M, Pei J. *Data Mining: Concepts and Techniques*. 3rd ed. Burlington, MA: Morgan-Kaufmann; 2011.

Liu L, Özsu MT, eds. *Encyclopedia of Database Systems*. New York, NY: Springer; 2009.

Maimon O, Rokach L, eds. *Data Mining and Knowledge Discovery Handbook*. New York, NY: Springer; 2010.

Mitra S, Acharya T. *Data Mining: Multimedia, Soft Computing, and Bioinformatics*. New York: John Wiley & Sons; 2003.

Pei J, Getoor L, de Keijzer A, eds. *Proceedings of the U*. New York, NY: ACM Press; 2009.

Tan PN, Steinbach M, Kumar V. *Introduction to Data Mining*. Boston, MA: Addison-Wesley; 2006.