Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 60 (2015) 613 – 622

Procedia Computer Science

19th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems

Approximation to expected support of frequent itemsets in mining probabilistic sets of uncertain data

Alfredo Cuzzocrea[a], Carson K. Leung[b,*], Richard Kyle MacKinnon[b]

[a]*Dept. of Engineering and Architecture (DIA), University of Trieste & ICAR-CNR, Via A. Valerio 6/1, 34127 Trieste (TS), Italy*
[b]*Department of Computer Science, University of Manitoba, Winnipeg, MB, R3T 2N2, Canada*

Abstract

Knowledge discovery and data mining generally discovers implicit, previously unknown, and useful knowledge from data. As one of the popular knowledge discovery and data mining tasks, frequent itemset mining, in particular, discovers knowledge in the form of sets of frequently co-occurring items, events, or objects. On the one hand, in many real-life applications, users mine frequent patterns from traditional databases of precise data, in which users know certainly the presence of items in transactions. On the other hand, in many other real-life applications, users mine frequent itemsets from probabilistic sets of uncertain data, in which users are uncertain about the likelihood of the presence of items in transactions. Each item in these probabilistic sets of uncertain data is often associated with an existential probability expressing the likelihood of its presence in that transaction. To mine frequent itemsets from these probabilistic datasets, many existing algorithms capture lots of information to compute expected support. To reduce the amount of space required, algorithms capture some but not all information in computing or approximating expected support. The tradeoff is that the upper bounds to expected support may not be tight. In this paper, we examine several upper bounds and recommend to the user which ones consume less space while providing good approximation to expected support of frequent itemsets in mining probabilistic sets of uncertain data.

Peer-review under responsibility of KES International

Keywords: Knowledge discovery and data mining; expected support; frequent patterns; uncertain data; upper bounds

1. Introduction and related works

With the automation of measurements and data collection, together with an increasing development and usage of a large number of sensors, high volumes of valuable data have been produced at high velocity from a high variety of data sources in different application areas—such as bio-informatics, chemical informatics, e-commerce, education, engineering, finance, healthcare, science, sports and telecommunications[21]—in the current era of Big data[8,10]. Mostly due to their high volumes, the quality and accuracy of data depend on their veracity (i.e., uncertainty of the data). Moreover, embedded in these data are useful knowledge. Hence, knowledge-based and intelligent informa-

* Corresponding author.
E-mail address: kleung@cs.umanitoba.ca (C.K. Leung)

Peer-review under responsibility of KES International
doi:10.1016/j.procs.2015.08.195

tion & engineering systems—which mine these data for the discovery of implicit, previously unknown, and useful knowledge—are in demand. Common knowledge discovery and data mining tasks include classification[7,19], clustering[18], graph mining[20], and frequent itemset mining.

Frequent itemset mining[2] aims to discover useful knowledge in the form of sets of frequently co-occurring items, events, or objects (i.e., frequent itemsets). It also serves as a building block for various data mining tasks such as stream mining[3] (which mines data that come at a high velocity), social network mining[4] and sports data mining[11]. Many existing algorithms mine frequent itemsets from high volumes of precise data, in which users definitely know whether an item is present in, or absent from, a transaction in databases of precise data. However, there are situations in which users are uncertain about the presence or absence of items (e.g., a physician may suspect, but may not guarantee, that a fevered patient got a flu or West Nile virus) in a probabilistic set of uncertain data. In it, each item x_i in a transaction t_j is associated with an existential probability $P(x_i, t_j)$ expressing the likelihood of the presence x_i in t_j.

To mine frequent itemsets from high varieties of high-value uncertain data, various algorithms have been proposed including UF-growth[13]. The UF-growth algorithm first scans the entire probabilistic set of n uncertain data transactions to accurately compute the expected support $expSup(\{x_i\})$ of each domain item (i.e., a singleton itemset) x_i. Note that x_i is considered *frequent* if $expSup(\{x_i\})$—which can computed by summing existential probability $P(x_i, t_j)$ over every transaction t_j containing x_i—in the entire uncertain dataset meets or exceeds the user-specified minimum support threshold *minsup*[9]. Afterwards, the UF-growth algorithm constructs a UF-tree structure (for capturing frequent domain items in the uncertain data), from which frequent itemsets can then be mined recursively. A 2^+-itemset (i.e., an itemset consisting of $k \geq 2$ items) X is considered *frequent* if its expected support $expSup(X) \geq minsup$. Here, $expSup(X)$ can be computed by summing $expSup(X, t_j)$ over every transaction t_j containing X, where $expSup(X, t_j)$ can be computed as the product of the existential probability $P(x_i, t_j)$ of every independent item x_i within the itemset $X = \{x_1, \ldots, x_k\}$. In order to accurately compute the expected support of each 2^+-itemset, paths in the corresponding UF-tree are shared only if tree nodes on the paths have the same item and the same existential probability. Due to this restrictive path sharing requirement, the UF-tree may be quite large.

Solutions to this large tree-size problem include the exploration of alternative mining approaches such as (i) hyperlinked array structure approaches (e.g., UH-Mine algorithm[1]), (ii) sampling-based approaches[6], and (iii) vertical mining approaches[5,16]. Another solution is to make the tree compact by capturing less but sufficient information about uncertain data. Based on the captured information, the corresponding knowledge discovery and data mining algorithms[12,14,15,17] first compute upper bounds to expected support for finding potentially frequent itemsets (i.e., containing true positives and false positives), and then test if the found itemsets are truly frequent (i.e., true positives). By doing so, the resulting trees are more compact than the UF-tree. This, in turn, shortens the tree traversal time during the knowledge discovery and data mining process and thus helps reduce the runtime. Moreover, the use of these upper bounds is expected to guarantee *not* to generate any false negatives: If an upper bound to expected support of an itemset X is less than *minsup*, then X is guaranteed to be infrequent. Furthermore, these upper bounds are expected to be tight so that not too many false positives are generated-and-tested. In this paper, we (i) present and examine the computation and tightness of some upper bounds to expected support, (ii) reformulate them so that we could compare them and determine which ones provide tighter upper bounds, and (iii) recommend the appropriate ones for frequent itemset mining from probabilistic sets of uncertain data.

The remainder of this paper is organized as follows. The next section gives background. Section 3 presents and examines several upper bounds to expected support for frequent itemset mining from probabilistic sets of uncertain data. Evaluation results and conclusions are given in Sections 4 and 5, respectively.

2. Background

Definition 1. Let `Item` be a set of m domain items. Each item y_i in a transaction $t_j = \{y_1, y_2, \ldots, y_h\} \subseteq$ `Item` in a probabilistic set of uncertain data is associated with an *existential probability*[9]—denoted as $P(y_i, t_j)$—with value

$$0 < P(y_i, t_j) \leq 1, \tag{1}$$

where $P(y_i, t_j)$ represents the likelihood of the presence of y_i in t_j.

Definition 2. Let Item be a set of m domain items. Then, the expected support[9]—denoted as $expSup(X, t_j)$—of k-*itemset* $X = \{x_1, x_2, \ldots, x_k\} \subseteq$ Item in a transaction $t_j = \{y_1, y_2, \ldots, y_r, \ldots, y_h\}$ (where $x_k = y_r$) can be computed as the product of the existential probability $P(y_i, t_j)$ of every independent item y_i within the itemset X, i.e.,

$$expSup(X, t_j) = \prod_{i=1}^{k} P(x_i, t_j) = \prod_{y_i \in X} P(y_i, t_j), \tag{2}$$

where X=$\{x_1, \ldots, x_k\} \subseteq \{y_1, \ldots, y_{r-1}, y_r, \ldots, y_h\}$=$t_j$.

Definition 3. Given (i) a probabilistic set of n uncertain data transactions and (ii) a user-specified minimum support threshold *minsup*, the research problem of *frequent itemset mining from the probabilistic set of uncertain data*[9] is to discover *frequent* itemsets from the dataset. Here, an itemset X is considered *frequent* if its expected support $expSup(X)$ in the entire uncertain dataset meets or exceeds the user-specified minimum support threshold *minsup*. Note that $expSup(X)$ in the entire probabilistic set of n uncertain data transactions can be computed by summing $expSup(X, t_j)$ over every transaction t_j containing X:

$$expSup(X) = \sum_{j=1}^{n} expSup(X, t_j), \tag{3}$$

where $expSup(X, t_j)$ can be computed as the product of the existential probability $P(x_i, t_j)$ of every independent item x_i within the itemset $X = \{x_1, \ldots, x_k\}$.

3. Upper bounds to expected support

Computing expected support of 2^+-itemsets in tree-based algorithms (e.g., UF-growth) often require large trees (e.g., large UF-trees), which capture existential probability of every item in each transaction of the uncertain dataset. To reduce the tree size, several algorithms capture less information and approximate expected support. For instance, both CUF-growth and CUF*-growth algorithms[14] use a single cap—called *transaction cap*—that serves as an upper bound to expected support of any itemset in the same transaction. As another instance, both DISC-growth and DISC*-growth algorithms[17] use a domain item-specific cap (or *item cap*, for short) to approximate expected support to itemsets. As a third instance, the PUF-growth algorithm[15] and the TPC-growth algorithm[12] use a *prefixed item cap* for the approximation. Among them, which caps lead to more accurate upper bounds to expected support? Which caps require less memory space? In order to compare them easily, we reformulate these caps using a common notion or expression.

3.1. Transaction caps

One way to approximate expected support or to obtain an upper bound to expected support of an itemset X is by using the *transaction cap* (*TC*), which is defined as follows.

Definition 4. Let (i) 2^+-itemset $X = \{x_1, x_2, \ldots, x_k\}$ and (ii) transaction $t_j = \{y_1, y_2, \ldots, y_r, \ldots, y_h\} \subseteq$ Item such that $X \subseteq t_j$ and x_k=y_r. Then, the *transaction cap* (*TC*) of X in t_j, which serves as an upper bound to the expected support $expSup(X, t_1)$ of X in t_j, is defined as the product of the two highest existential probabilities in the entire transaction t_j:

$$TC(X, t_j) = TM_1(t_j) \times TM_2(t_j), \tag{4}$$

where

- $TM_1(t_j) = \max_{i \in [1,h]} P(y_i, t_j)$ is the highest existential probability in t_j; and
- $TM_2(t_j) = \max_{i \in [1,h] \wedge (i \neq g)} P(y_i, t_j)$ is the second highest existential probability in t_j for $y_g = \text{argmax}_{i \in [1,h]} P(y_i, t_j)$, i.e., $TM_1(t_j) = P(y_g, t_j)$.

Example 1. Consider a transaction $t_1 = \{a{:}0.2, b{:}0.4, c{:}0.6, d{:}0.8, e{:}0.9, f{:}0.7, g{:}0.5, h{:}0.1\}$ of uncertain data. Here, each item is associated with an existential probability. For instance, item a is associated with an existential probability

of 0.2 expressing there is a 20% likelihood of item a to be present in transaction t_1. In this transaction, the two highest existential probabilities are $TM_1(t_1)$=0.9 (belongs to e, i.e., y_g=e) and $TM_2(t_1)$=0.8 (belongs to d). Then, the transaction cap $TC(\{d, e\}, t_1)$ is 0.9 ×0.8 = 0.72, which is as tight as its expected support $expSup(\{d, e\}, t_1)$.

However, the transaction cap becomes loose for long patterns (i.e., itemsets of high cardinality). For instance, the transaction cap $TC(\{d, e, f\}, t_1)$ is also 0.9 ×0.8 = 0.72 (cf. $expSup(\{d, e, f\}, t_1)$ = 0.8 ×0.9 ×0.7 = 0.504).

Observation 1. Based on Definition 4 and Example 1, we observed the following:

- The transaction cap $TC(X, t_j)$ of any 2^+-itemset X contained in the same transaction (e.g., $\{d, e\}$ and $\{d, e, f\}$ in t_1) would have the same value.
- As the TC is fixed for each transaction, it can be pre-computed so as to save runtime.
- The TC serves as a good upper bound to 2-itemsets (e.g., $\{d, e\}$ having its TC value identical to its expected support). However, TC may not be too tight for 3^+-itemsets (e.g., $\{d, e, f\}$).

To tighten the upper bound to expected support for 3^+-itemsets (i.e., k-itemsets where $k \geq 3$), the concept of TC can be extended as follows.

Definition 5. Let (i) 2^+-itemset $X = \{x_1, x_2, \ldots, x_k\}$ and (ii) transaction $t_j = \{y_1, y_2, \ldots, y_r, \ldots, y_h\} \subseteq \texttt{Item}$ such that $X \subseteq t_j$ and x_k=y_r. Then, the *extended transaction cap* (*ETC*) of X in t_j, which serves as an upper bound to the expected support $expSup(X, t_1)$ of X in t_j, is defined as the product of the two highest existential probabilities in the entire transaction t_j with the $(k-2)$-th power of the third highest existential probability in the entire transaction t_j:

$$ETC(X, t_j) = \begin{cases} TC(X, t_j) & = TM_1(t_j) \times TM_2(t_j) \quad \text{if } k{=}2 \\ TC(X, t_j) \times \left[TM_3(t_j)\right]^{k-2} & = TM_1(t_j) \times TM_2(t_j) \times \left[TM_3(t_j)\right]^{k-2} \quad \text{if } k{\geq}3 \end{cases} \tag{5}$$

where

- $TM_1(t_j) = \max_{i\in[1,h]} P(y_i, t_j)$ is the highest existential probability in t_j;
- $TM_2(t_j) = \max_{i\in[1,h]\wedge(i\neq g)} P(y_i, t_j)$ is the second highest existential probability in t_j for $y_g = \text{argmax}_{i\in[1,h]} P(y_i, t_j)$, i.e., $TM_1(t_j) = P(y_g, t_j)$; and
- $TM_3(t_j) = \max_{i\in[1,h]\wedge(i\neq g)\wedge(i\neq s)} P(y_i, t_j)$ is the third highest existential probability in t_j for y_s=$\text{argmax}_{i\in[1,h]\wedge(i\neq g)}$ $P(y_i, t_j)$, i.e., $TM_2(t_j) = P(y_s, t_j)$.

Example 2. Reconsider transaction t_1 in Example 1, the extended transaction caps for long patterns (i.e., itemsets of high cardinality) are tightened. For instance, the extended transaction cap $ETC(\{d, e, f\}, t_1)$ is 0.9 ×0.8 ×0.7 = 0.504, which is as tight as its expected support $expSup(\{d, e, f\}, t_1)$.

However, the extended transaction cap may still be loose for some patterns, especially for those do not have all (or some) of the three highest existential probability values. For instance, the extended transaction cap $ETC(\{b, d, e\}, t_1)$ is also 0.9 ×0.8 ×0.7 = 0.504 (cf. $expSup(\{b, d, e\}, t_1)$ = 0.4 ×0.8 ×0.9 = 0.288).

Observation 2. Based on Definition 5 and Example 2, we observed the following:

- The extended transaction cap $ETC(X, t_j)$ of any k-itemset $X \subseteq t_j$ of the same cardinality $k \geq 2$ would have the same value (e.g., $ETC(\{d, e, f\}, t_1) = ETC(\{b, d, e\}, t_1)$).
- As the ETC is fixed for each cardinality in each transaction, it can be pre-computed so as to save runtime.

3.2. Item caps

On the one hand, the transaction cap (TC) and its extension (ETC) can be easily pre-computed. On the other hand, they may not involve any items in X. To tighten the upper bound, the *domain item-specific cap*—or *item cap* (*IC*) for short—involves at least one item in X. Intuitively, item cap could be defined as the product of $P(x_k, t_j)$ and the highest existential probability $TM_1(t_j)$ in t_j. However, for a special case where $P(x_k, t_j)$ happens to be $TM_1(t_j)$, IC would then multiply $TM_1(t_j)$ twice and thus loosen the upper bound to expected support. Hence, instead, we deal with this special case by defining IC as follows, which multiplies $P(x_k, t_j)$ with the second highest existential probability $TM_2(t_j)$ in t_j.

Definition 6. Let (i) 2^+-itemset $X = \{x_1, x_2, \ldots, x_k\}$ and (ii) transaction $t_j = \{y_1, y_2, \ldots, y_r, \ldots, y_h\} \subseteq \texttt{Item}$ such that $X \subseteq t_j$ and x_k=y_r. Then, the *item cap* (*IC*) of X in t_j, which serves as an upper bound to the expected support $expSup(X, t_1)$ of X in t_j, is defined as (i) the product of $P(x_k, t_j)$ and the highest existential probability $TM_1(t_j)$ in t_j for most cases, and (ii) the product of $P(x_k, t_j)$ and the second highest existential probability $TM_2(t_j)$ in t_j for the special case where x_k possesses the highest existential probability:

$$IC(X, t_j) = \begin{cases} P(x_k, t_j) \times TM_1(t_j) \text{ if } x_k \neq y_g \\ P(x_k, t_j) \times TM_2(t_j) \text{ if } x_k = y_g \end{cases} \tag{6}$$

where

- $TM_1(t_j) = \max_{i \in [1,h]} P(y_i, t_j)$ is the highest existential probability in t_j; and
- $TM_2(t_j) = \max_{i \in [1,h] \wedge (i \neq g)} P(y_i, t_j)$ is the second highest existential probability in t_j for $y_g = \text{argmax}_{i \in [1,h]} P(y_i, t_j)$, i.e., $TM_1(t_j) = P(y_g, t_j)$.

Example 3. Reconsider transaction t_1 in Examples 1 and 2, the item caps for many patterns are tightened. For instance, the item cap $IC(\{e, g\}, t_1)$ is $0.5 \times 0.9 = 0.45$, which is as tight as its expected support $expSup(\{e, g\}, t_1)$ (cf. its transaction cap $TC(\{e, g\}, t_1)$ is $0.9 \times 0.8 = 0.72$). The item cap $IC(\{d, e\}, t_1)$ for the special case is $0.9 \times 0.8 = 0.72$, which is as tight as its expected support $expSup(\{d, e\}, t_1)$.

However, like the TC, the IC becomes loose for long patterns (i.e., itemsets of high cardinality). For instance, the item cap $IC(\{d, e, g\}, t_1)$ is also $0.5 \times 0.9 = 0.45$ (cf. $ETC(\{d, e, g\}, t_1)$ is $0.9 \times 0.8 \times 0.7 = 0.504$ and $expSup(\{d, e, g\}, t_1)$ $= 0.8 \times 0.9 \times 0.5 = 0.36$).

Observation 3. Based on Definition 6 and Example 3, we observed the following:

- The item cap $IC(X, t_j)$ of any 2^+-itemset $X = \{x_1, x_2, \ldots, x_k\} \subseteq t_j$ ending with the same suffix item x_k (e.g., $\{d, g\}, \{e, g\}, \{d, e, g\}$) would have the same value. This comment applies to both (i) $x_k \neq y_g$ and (ii) $x_k = y_g$.
- As the IC is fixed for each suffix item x_k, it can be pre-computed so as to save runtime.
- The IC serves as a good upper bound to 2-itemsets (e.g., $\{d, e\}$ and $\{e, g\}$ having their IC values identical to their corresponding expected support). However, IC may not be too tight for 3^+-itemsets (e.g., $\{d, e, g\}$).

Similar to the extension of TC to become ETC, the concept of IC can be extended as follows to tighten the upper bound to expected support for 3^+-itemsets (i.e., k-itemsets where $k \geq 3$).

Definition 7. Let (i) 2^+-itemset $X = \{x_1, x_2, \ldots, x_k\}$ and (ii) transaction $t_j = \{y_1, y_2, \ldots, y_r, \ldots, y_h\} \subseteq \texttt{Item}$ such that $X \subseteq t_j$ and x_k=y_r. Then, the *extended item cap* (*EIC*) of X in t_j, which serves as an upper bound to the expected support $expSup(X, t_1)$ of X in t_j, is defined as the product of $P(x_k, t_j)$ and the highest existential probability $TM_1(t_j)$ in t_j with the $(k-2)$-th power of the second highest existential probability $TM_2(t_j)$ in the entire transaction t_j. To deal with two special cases where $P(x_k, t_j)$ happens to be $TM_1(t_j)$ or $TM_2(t_j)$, the extended item cap multiplies the second highest existential probability $TM_2(t_j)$ in t_j and/or the $(k-2)$-th power of the third highest existential probability $TM_3(t_j)$ in the entire transaction t_j:

$$EIC(X, t_j) = \begin{cases} IC(X, t_j) = \begin{cases} P(x_k, t_j) \times TM_1(t_j) & \text{if } k{=}2 \wedge x_k \neq y_g \\ P(x_k, t_j) \times TM_2(t_j) & \text{if } k{=}2 \wedge x_k = y_g \end{cases} \\ IC(X, t_j) \times \left[TM_2(t_j)\right]^{k-2} = P(x_k, t_j) \times TM_1(t_j) \times \left[TM_2(t_j)\right]^{k-2} \text{ if } k{\geq}3 \wedge x_k \neq y_g \wedge x_k \neq y_s \\ IC(X, t_j) \times \left[TM_3(t_j)\right]^{k-2} = \begin{cases} P(x_k, t_j) \times TM_1(t_j) \times \left[TM_3(t_j)\right]^{k-2} & \text{if } k{\geq}3 \wedge x_k = y_s \\ P(x_k, t_j) \times TM_2(t_j) \times \left[TM_3(t_j)\right]^{k-2} & \text{if } k{\geq}3 \wedge x_k = y_g \end{cases} \end{cases} \tag{7}$$

where

- $TM_1(t_j) = \max_{i \in [1,h]} P(y_i, t_j)$ is the highest existential probability in t_j;
- $TM_2(t_j) = \max_{i \in [1,h] \wedge (i \neq g)} P(y_i, t_j)$ is the second highest existential probability in t_j for $y_g = \text{argmax}_{i \in [1,h]} P(y_i, t_j)$, i.e., $TM_1(t_j) = P(y_g, t_j)$; and
- $TM_3(t_j) = \max_{i \in [1,h] \wedge (i \neq g) \wedge (i \neq s)} P(y_i, t_j)$ is the third highest existential probability in t_j for y_s=$\text{argmax}_{i \in [1,h] \wedge (i \neq g)}$ $P(y_i, t_j)$, i.e., $TM_2(t_j) = P(y_s, t_j)$.

Example 4. Reconsider transaction t_1 in Examples 1–3, the extended item caps for long patterns (i.e., itemsets of high cardinality) are tightened. For instance, the extended item cap $EIC(\{d, e, g\}, t_1)$ is $0.5 \times 0.9 \times 0.8 = 0.36$, which is as tight as its expected support $expSup(\{d, e, g\}, t_1)$. The extended item cap $EIC(\{c, d, e\}, t_1)$ for a special case where e possesses the highest existential probability of $0.9 \times 0.8 \times 0.7 = 0.504$ (cf. $expSup(\{c, d, e\}, t_1) = 0.432$). The extended item cap $EIC(\{b, c, d\}, t_1)$ for another special case where d possesses the second highest existential probability of $0.8 \times 0.9 \times 0.7 = 0.504$ (cf. $expSup(\{b, c, d\}, t_1) = 0.192$).

Observation 4. Based on Definition 7 and Example 4, we observed the following:

- The extended item cap $EIC(X, t_j)$ of any k-itemset $X = \{x_1, x_2, \ldots, x_k\} \subseteq t_j$ of the same cardinality k ending with the same suffix item x_k (e.g., $\{a, b, d\}, \{b, c, d\}$) would have the same value. This comment applies to (i) $x_k = y_g$, (ii) $x_k = y_s$, and (iii) $x_k \neq y_g x_k \neq y_s$.
- As the EIC is fixed for each cardinality k sharing the same suffix item x_k, it can be pre-computed so as to save runtime.

3.3. Prefixed item caps

Recall from Equation (2) that the expected support of X can be computed as the product of $P(x_k, t_j)$ and existential probabilities of the proper prefix of x_k. Hence, it is more logical to obtain an upper bound to expected support of X by involving $P(x_k, t_j)$ and existential probabilities of the *proper prefix* of x_k. This leads to the concept of *prefixed item cap* (*PIC*), defined as follows.

Definition 8. Let (i) 2^+-itemset $X = \{x_1, x_2, \ldots, x_k\}$ and (ii) transaction $t_j = \{y_1, y_2, \ldots, y_r, \ldots, y_h\} \subseteq \texttt{Item}$ such that $X \subseteq t_j$ and x_k=y_r. Then, the *prefixed item cap* (*PIC*) of X in t_j, which serves as an upper bound to the expected support $expSup(X, t_1)$ of X in t_j, is defined as the the product of $P(x_k, t_j)$ and the highest existential probability $PM(x_k, t_j)$ among items in the proper prefix of x_k:

$$PIC(X, t_j) = P(x_k, t_j) \times PM_1(y_r, t_j), \tag{8}$$

where $PM_1(y_r, t_j) = \max_{i \in [1, r-1]} P(y_i, t_j)$ is the *prefixed maximum*, which is defined as the highest existential probability in $\{y_1, \ldots, y_{r-1}\} \subset t_j$.

Example 5. Reconsider transaction t_1 in Examples 1 and 4, the prefixed item caps for many patterns are tightened. For instance, the prefixed item cap $PIC(\{c, d\}, t_1)$ is $0.8 \times 0.6 = 0.48$, which is as tight as its expected support $expSup(\{c, d\}, t_1)$ (cf. $TC(\{c, d\}, t_1)$ is $0.9 \times 0.8 = 0.72$ and $IC(\{c, d\}, t_1)$ is $0.8 \times 0.9 = 0.72$).

However, like the TC and IC, the PIC also becomes loose for long patterns (i.e., itemsets of high cardinality). For instance, the prefixed item cap $PIC(\{a, c, d\}, t_1)$ is also $0.8 \times 0.6 = 0.48$ (cf. $expSup(\{a, c, d\}, t_1) = 0.2 \times 0.6 \times 0.8 = 0.096$).

Observation 5. Based on Definition 8 and Example 5, we observed the following:

- The prefixed item cap $PIC(X, t_j)$ of any 2^+-itemset $X = \{x_1, x_2, \ldots, x_k\} \subseteq t_j$ ending with the same suffix item x_k (e.g., $\{a, d\}, \{c, d\}, \{a, c, d\}$) would have the same value.
- As the PIC is fixed for each suffix item x_k, it can be pre-computed so as to save runtime.
- The PIC serves as a good upper bound to 2-itemsets (e.g., $\{c, d\}$ having its PIC value identical to its expected support). However, PIC may not be too tight for 3^+-itemsets (e.g., $\{a, c, d\}$).

Similar to (i) the extension of TC to become ETC and (ii) the extension of IC to become EIC, the concept of PIC can be extended as follows to further tighten the upper bound to expected support for 3^+-itemsets (i.e., k-itemsets where $k \geq 3$) by multiplying $PIC(X, t_j)$ by the $(k-2)$-th power of the *prefixed second-maximum* (i.e., second highest existential probability $PM_2(y_r, t_j)$ in $\{y_1, \ldots, y_{r-1}\} \subset t_j$). See Definition 9.

Definition 9. Let (i) 2^+-itemset $X = \{x_1, x_2, \ldots, x_k\}$ and (ii) transaction $t_j = \{y_1, y_2, \ldots, y_r, \ldots, y_h\} \subseteq \texttt{Item}$ such that $X \subseteq t_j$ and x_k=y_r. Then, the *extended prefixed item cap* (*EPIC*) of X in t_j, which serves as an upper bound to the expected support $expSup(X, t_1)$ of X in t_j, is defined as the the product of $P(x_k, t_j)$ and the highest existential

probability $PM(x_k, t_j)$ among items in the proper prefix of x_k with the $(k-2)$-th power of the second highest existential probability $PM_2(t_j)$ among items in the proper prefix of x_k:

$$EPIC(X, t_j) = \begin{cases} PIC(X, t_j) & = P(x_k, t_j) \times PM_1(y_r, t_j) \quad \text{if } k{=}2 \\ PIC(X, t_j) \times \left[PM_2(y_r, t_j)\right]^{k-2} & = P(x_k, t_j) \times PM_1(y_r, t_j) \times \left[PM_2(y_r, t_j)\right]^{k-2} \quad \text{if } k{\geq}3 \end{cases} \tag{9}$$

where

- $PM_1(y_r, t_j) = \max_{i \in [1,r-1]} P(y_i, t_j)$ is the *prefixed maximum*, which is defined as the highest existential probability in $\{y_1, \ldots, y_{r-1}\} \subset t_j$; and
- $PM_2(y_r, t_j) = \max_{i \in [1,r-1] \wedge (i \neq g)} P(y_i, t_j)$ is the prefixed second-maximum, which is defined as the second highest existential probability in $\{y_1, \ldots, y_{r-1}\} \subset t_j$ for $y_g{=}\text{argmax}_{i \in [1,h]} P(y_i, t_j)$, i.e., $PM_1(y_r, t_j) = P(y_g, t_j)$.

Example 6. Reconsider transaction t_1 in Examples 1–5, the extended prefixed item caps for many patterns are tightened. For instance, the extended prefixed item cap $EPIC(\{a, c, d\}, t_1)$ is $0.8 \times 0.6 \times 0.4 = 0.192$, which is tighter than its prefixed item cap $PIC(\{a, c, d\}, t_1)$ of $0.8 \times 0.9 = 0.72$).

Observation 6. Based on Definition 9 and Example 6, we observed the following:

- The extended prefixed item cap $EPIC(X, t_j)$ of any k-itemset $X = \{x_1, x_2, \ldots, x_k\} \subseteq t_j$ of the same cardinality k ending with the same suffix item x_k (e.g., $\{a, c, d\}, \{a, b, d\}, \{b, c, d\}$) would have the same value.
- As the EPIC is fixed for each cardinality k sharing the same suffix item x_k, it can be pre-computed so as to save runtime.

4. Evaluation

In this section, we evaluate several aspects on the aforementioned approximations (i.e., upper bounds) to expected support: (i) memory consumption, (ii) accuracy, and (iii) runtime.

4.1. Memory consumption

First, we analytically evaluate the memory consumption of these six different approximations. Among them, we observed the following:

- TC requires the least amount of memory space because they are solely dependent on transaction t_j. In other words, only a single value (TC) is needed for each transaction t_j.
- ETC requires slightly more memory space because, according to Equation (5), two values—both TC and $TM_3(t_j)$—are needed for each transaction t_j in order to compute the ETC value for itemsets of different cardinality k. For both TC and ETC, we do not need to store existential probabilities of any items in transaction t_j.
- In contrast, IC and PIC each requires a total of h values for each transaction t_j. Specifically, for each transaction $t_j = \{y_1, y_2, \ldots, y_r \ldots, y_h\}$ with h items, a single value (IC or PIC) is needed for each item y_i in t_j.
- As an extension to IC, EIC needs to store an additional value—namely, $TM_2(t_j)$ or $TM_3(t_j)$ depending on whether $x_k = y_g$ or y_s—for each item x_k $(= y_r)$in transaction t_j. Similarly, as an extension to PIC, EPIC needs to store an additional value—namely, $PM_2(y_r, t_j)$—for each item y_r in transaction t_j. In other words, both EIC and EPIC require the most amount of memory space because each of them requires a total of $2h$ values for each transaction t_j.

4.2. Accuracy

We measure the accuracy by first comparing the tightness of the upper bounds as approximated expected support. Recall that $expSup(X, t_j) = \prod_{i=1}^{k} P(x_i, t_j) = \prod_{y_i \in X} P(y_i, t_j)$. Then, based on Definitions 4–9, we observed the following for any 2-itemset X:

- $ETC(X, t_j) = TC(X, t_j)$,
- $EIC(X, t_j) = IC(X, t_j)$, and
- $EPIC(X, t_j) = PIC(X, t_j)$.

These observations are confirmed by Fig. 1 that (i) ETC and TC led to the same number of false positives for 2-itemsets (i.e., cardinality = 2), and that (ii) EIC and IC—as well as EPIC and PIC—also led to the same number of false positives for 2-itemsets (i.e., cardinality = 2). Among these three groups of upper bounds, we also observed that (i) PIC involves the item having the maximum existential probability $PM_1(y_r, t_j)$ in the proper prefix of y_r. (ii) IC involves the item having the maximum existential probability $TM_1(t_j)$ in the proper prefix of y_r as well as its suffix. Consequently, as $PM_1(y_r, t_j) \leq TM_1(t_j)$, we get (i) $PIC(X, t_j) \leq IC(X, t_j)$. Moreover, IC also uses $P(x_k, t_j)$, whereas TC uses $TM_2(t_j)$—which may not even involve any items in X—when $x_k \neq y_g$. So, as $P(x_k, t_j) \leq TM_2(t_j)$, we get (ii) $IC(X, t_j) \leq TC(X, t_j)$. Hence, analytically, it is generally that

$$PIC(X, t_j) \leq IC(X, t_j) \leq TC(X, t_j). \tag{10}$$

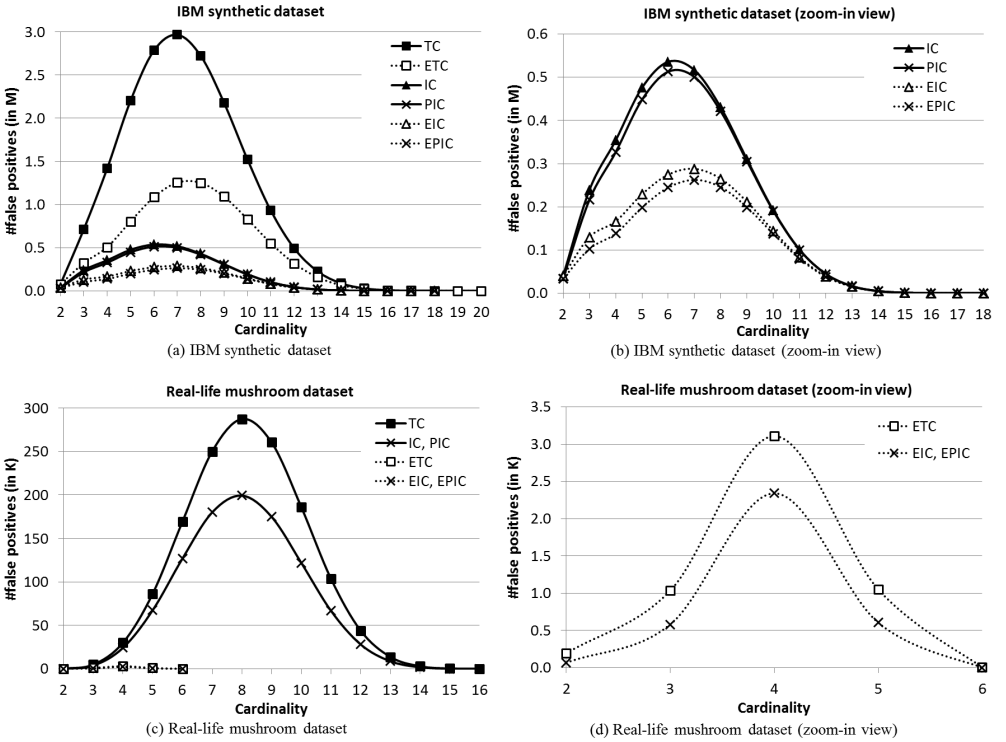
The same inequality is confirmed experimentally, as shown in Fig. 1. In other words, PIC generally provides the tightest upper bounds to expected support when mining frequent 2-itemsets from high volumes of high-value uncertain data. When mining 3^+-itemsets, we observed the following:

- $ETC(X, t_j) \leq TC(X, t_j)$ due to the extra multiplication term $[TM_3(t_j)]^{k-2}$ in ETC such that $0 < [TM_3(t_j)]^{k-2} \leq 1$. Hence, ETC provides tighter upper bounds to expected support than TC when mining frequent 3^+-itemsets from high volumes of high-value uncertain data.
- $EIC(X, t_j) \leq IC(X, t_j)$ and $EPIC(X, t_j) \leq PIC(X, t_j)$ due to the same reason, i.e., the extra multiplication terms—which are in the range (0,1]—in EIC and EPIC.

After analyzing the intra-group relationships between the aforementioned upper bounds, let us analyze the inter-group relationships among the four extensions when they mine k-itemsets (for itemset X):

- If x_k=y_g, then $EIC(X, t_j) = ETC(X, t_j)$ because $P(x_k, t_j) = P(y_g, t_j) = TM_1(t_j)$.
- Similarly, if x_k=y_s, then $EIC(X, t_j) = ETC(X, t_j)$ because $P(x_k, t_j) = P(y_s, t_j) = TM_2(t_j)$.

Hence, when x_k is associated with the highest or the second highest existential probability in t_j, both EIC and ETC provide the same upper bounds to expected support when mining frequent 3^+-itemsets. Moreover,



(a) IBM synthetic dataset (b) IBM synthetic dataset (zoom-in view)

(c) Real-life mushroom dataset (d) Real-life mushroom dataset (zoom-in view)

Fig. 1. Accuracy: the number of false positives.

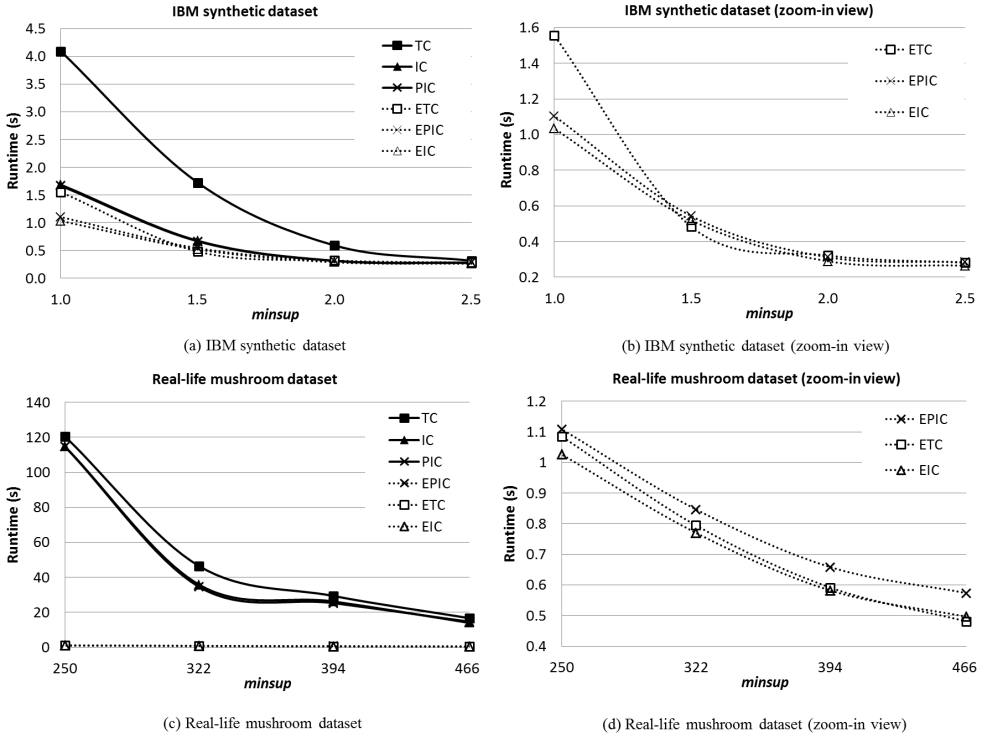
- if $(x_k \neq y_g)$ and $(x_k \neq y_s)$, then $EPIC(X, t_j) \leq EIC(X, t_j)$ because both $[PM_1(y_r, t_j) \leq TM_1(t_j)]$ and $[PM_2(y_r, t_j) \leq TM_2(t_j)]$. Hence, when x_k is not associated with the highest or the second highest existential probability in t_j, EPIC provides tighter upper bounds to expected support than EIC.

The above analysis shows the tightness of these upper bounds to expected support. Note that all these bounds do *not* lead to any false negatives but only false positives. The tighter the bound, the lower is the number of false positives. Our experimental results shown in Fig. 1 support our analytical results. Specifically, TC led to the highest numbers of false positives, whereas EPIC led to the lowest numbers (with EIC led to a close second lowest numbers) of false positives in (i) IBM synthetic dataset and (ii) real-life datasets (e.g., mushroom) from the UC Irvine Machine Learning Depository as well as those from the Frequent Itemset Mining Implementation (FIMI) Dataset Repository. Moreover, it is interesting to note that the tightness of the upper bound to expected support provided by the three extensions (ETC, EIC and EPIC). They did not generate any false positives beyond cardinality 6 for the mushroom dataset, as shown in Fig. 1(c).

4.3. Runtime

Recall that knowledge discovery and data mining algorithms use the aforementioned caps to approximate expected support. The algorithms find itemsets with upper bounds to expected support meeting or exceeding the user-specified threshold *minsup*. This results in a collection of all potentially frequent 2^+-itemsets, which include true positive (i.e., truly frequent itemsets) and false positive (i.e., potentially frequent with respect to upper bounds but truly infrequent with respect to *minsup*). With tighter upper bounds to expected support, fewer false positives are produced. Hence, shorter runtimes result. See Fig. 2, which shows the following:

- Due to its highest number of false positives generated, TC took the longest runtime.
- As all three extensions (ETC, EPIC, and EIC) produced fewer false positives than the counterparts (TC, PIC, and IC), the runtimes for the former were also shorter.
- As usual, when *minsup* increased, the runtime decreased.



(a) IBM synthetic dataset

(b) IBM synthetic dataset (zoom-in view)

(c) Real-life mushroom dataset

(d) Real-life mushroom dataset (zoom-in view)

Fig. 2. Runtime.

- Recall that $EPIC(X, t_j) \leq EIC(X, t_j)$ if $(x_k \neq y_g)$ and $(x_k \neq y_s)$. For cases where $(x_k = y_g)$ or $(x_k = y_s)$, it is possible—but not guarantee—that $EPIC(X, t_j) \leq EIC(X, t_j)$. However, for some other cases (e.g., for short transactions in the IBM synthetic dataset or short frequent patterns mined from the real-life mushroom dataset), EIC beat EPIC.

After evaluating the six approximations as upper bounds to expected support, we observed that (i) TC requires the least amount of memory space (with a single value per transaction) and ETC requires the second least amount of memory space (with two values per transaction), (ii) EIC and EPIC produced fewest false positives due to the tightness of their bounds, and (iii) EIC took the shortest runtime and the other two extensions (EPIC and ETC) took just slightly longer than EIC. Our recommendation is that (i) if memory is an issue, it seems better to use ETC due to the small memory requirements, production of a few of false positives, and short runtimes. Otherwise, it seems better to use EIC or EPIC because their memory requirements are not too high ($2h$ values for h items in a transaction) but they produced fewer false positives and ran faster than others.

5. Conclusions

In this paper, we presented and examined six upper bounds to expected support of frequent k-itemsets when mining probabilistic sets of uncertain data, including transaction cap (TC), item cap (IC), and prefixed item cap (PIC), as well as their extensions. Among these upper bounds, PIC provides the tightest upper bounds when mining frequent 2-itemsets, and thus produces the fewest false positives (i.e., potentially frequent 2-itemsets) and runs the fastest. When mining frequent 3^+-itemsets, the concepts of TC, IC, and PIC were extended to become ETC, EIC, and EPIC. Our experimental results confirm our analytical findings and recommendations that these extensions provide tighter upper bounds to expected support of frequent 3^+-itemsets when mining probabilistic sets of uncertain data for potentially frequent 3^+-itemsets, which are then verified to obtain truly frequent 3^+-itemsets.

Acknowledgements. This project is partially supported by NSERC (Canada) and the University of Manitoba.

References

1. Aggarwal CC, Li Y, Wang J, Wang J. Frequent pattern mining with uncertain data. In: *Proceedings of the ACM KDD 2009*, p. 29–37.
2. Agrawal R, Srikant R. Fast algorithms for mining association rules. In: *Proceedings of the VLDB 1994*, p. 487–499.
3. Braun P, Cameron JJ, Cuzzocrea A, Jiang F, Leung CK. Effectively and efficiently mining frequent patterns from dense graph streams on disk. *Procedia Computer Science* 2014; **35**:338–347.
4. Braun P, Cuzzocrea A, Leung CK, MacKinnon RK, Tanbeer SK. A tree-based algorithm for mining diverse social entities. *Procedia Computer Science* 2014; **35**:223–232.
5. Budhia BP, Cuzzocrea A, Leung CK. Vertical frequent pattern mining from uncertain data. In: *Proceedings of the KES 2012*, p. 1273–1282.
6. Calders T, Garboni C, Goethals B. Efficient pattern mining of uncertain data with sampling. In: *Proceedings of the PAKDD 2010, Part I*, p. 480–487.
7. Czarnowski I, Jedrzejowicz P. Ensemble classifier for mining data streams. *Procedia Computer Science* 2014; **35**:397–406.
8. Jiang F, Leung CK, MacKinnon RK. BigSAM: mining interesting patterns from probabilistic databases of uncertain big data. In: *Proceedings of the PAKDD Workshops 2014*. Springer; 2014, p. 780–792.
9. Leung CK. Uncertain frequent pattern mining. In: Aggarwal CC, Han J, editors. *Frequent pattern mining*. 2014; p. 417–453.
10. Leung CK, Jiang F. A data science solution for mining interesting patterns from uncertain big data. In: *Proceedings of the IEEE BDCloud 2014*. IEEE Computer Society; 2014, p. 235–242.
11. Leung CK, Joseph KW. Sports data mining: predicting results for the college football games. *Procedia Computer Science* 2014; **35**:710–719.
12. Leung CK, MacKinnon RK, Tanbeer SK. Tightening upper bounds to expected support for uncertain frequent pattern mining. *Procedia Computer Science* 2014; **35**:328–337.
13. Leung CK, Mateo MAF, Brajczuk DA. A tree-based approach for frequent pattern mining from uncertain data. In: *Proceedings of the PAKDD 2008*, p. 653–661.
14. Leung CK, Tanbeer SK. Fast tree-based mining of frequent itemsets from uncertain data. In: *Proceedings of the DASFAA 2012, Part I*, p. 272–287.
15. Leung CK, Tanbeer SK. PUF-tree: a compact tree structure for frequent pattern mining of uncertain data. In: *Proceedings of the PAKDD 2013*, p. 13–25.
16. Leung CK, Tanbeer SK, Budhia BP, Zacharias LC. Mining probabilistic datasets vertically. In: *Proceedings of the IDEAS 2012*, p. 199–204.
17. MacKinnon RK, , Strauss TD, Leung CK. DISC: efficient uncertain frequent pattern mining with tightened upper bounds. In: *Proceedings of the IEEE ICDM 2014 Workshops*, p. 1038–1045.
18. Mulyono NB, Ishida Y. Clustering inventory locations to improve the performance of disaster relief operations. *Procedia Computer Science* 2014; **35**:1388–1397.
19. Perner P. Mining sparse and big data by case-based reasoning. *Procedia Computer Science* 2014; **35**:19–33.
20. Ríos SA, Videla-Cavieres IF. Generating groups of products using graph mining techniques. *Procedia Computer Science* 2014; **35**:730–738.
21. Tanbeer SK, Leung CK, Cameron JJ. Interactive mining of strong friends from social networks and its applications in e-commerce. *Journal of Organizational Computing and Electronic Commerce* 2014; **24**(2–3):157–173.