Technical Report: Mining Probabilistic Generalized Frequent Itemsets in Uncertain Databases

Erich A. Peterson
Myeloma Institute for Research and Therapy
University of Arkansas for Medical Sciences
contact@erichpeterson.com

Peiyi Tang
Department of Computer Science
University of Arkansas at Little Rock
pxtang@ualr.edu

April 4, 2013

Abstract

Researchers have recently defined and presented the theoretical concepts—rooted in possible world semantics—and algorithm necessary for mining so-called probabilistic frequent itemsets in uncertain databases. Further, there exist algorithms for mining so-called generalized itemset in certain databases, where a taxonomy exists relating concrete items to abstract (generalized) items not in the database. Currently, no research has been done in formulating a theory and algorithm for mining generalized itemsets from uncertain databases. Using probability theory and possible world semantics, we formulate a method for calculating the probability a generalized item will occur within an uncertain transaction. Thus, the new concept of a probabilistic frequent generalized itemset is presented; as well as, an algorithm to mine these new concepts.

1 Introduction

Data mining *generalized frequent itemsets* (GFIs) and generalized association rules was first proposed by Srikant et al. [19]. The methods employed in what will sometimes be referred to as generalized itemset mining in this paper, take much inspiration from the algorithms and concepts of traditional frequent itemset and association rules mining (sometimes referred to as traditional itemset mining in this paper) [2, 3, 16][1]. Generalized itemset mining differs from traditional itemset mining, through the addition of a taxonomy G to an itemset database T. This taxonomy forms an is-a relationship among items, and is represented as an directed acyclical graph. An example taxonomy used throughout this paper is shown in Figure 1. In G one sees that apple and banana is-a fruit, and fruit is-a produce. Further, within G there exists a ancestor/descendent relationship among all certain items within G. For example, an item a is said to be a descendent of a', if and only if, there exists a path from a' to a. Conversely, a' is called an ancestor of a. An arbitrary ancestor of an item a is denoted as $\hat{a}$. With the addition of a taxonomy, one can develop new problems and discover knowledge previously not easily discernible. One such problem, is to find "rules that span different levels of the taxonomy."[19]. This problem is especially important as items at lower levels of the taxonomy may not have the minimum support necessary to qualify as frequent; however, items higher up in the taxonomy may be because "department stores or supermarkets typically have hundreds of thousands of items, [and] the support for rules involving only leaf items (typically UPC or SKU codes) tends to be extremely small." [19]

Another data mining problem, which has received a fair amount of more recent research, is that of mining for so-called *probabilistic frequent itemsets* (PFIs). First proposed by Bernecker et al. [5], this work concerns itself with mining of frequent itemsets in *uncertain* databases; that is, databases in which each item in the database T has an associated existential probability—denoting the probability of its occurrence within a transaction. In this work, an arbitrary itemset is said to be probabilistically frequent if its *frequentness probability*, defined as the probability the item will have a support greater than or equal to a user-defined minimum support threshold *minsup*, is greater than or equal to a user-defined confidence threshold τ. Thus, to

[1]Although generalized itemset mining was discovered after traditional itemset mining, as the name implies, generalized itemset mining constitutes a general case of the special case of itemset mining.

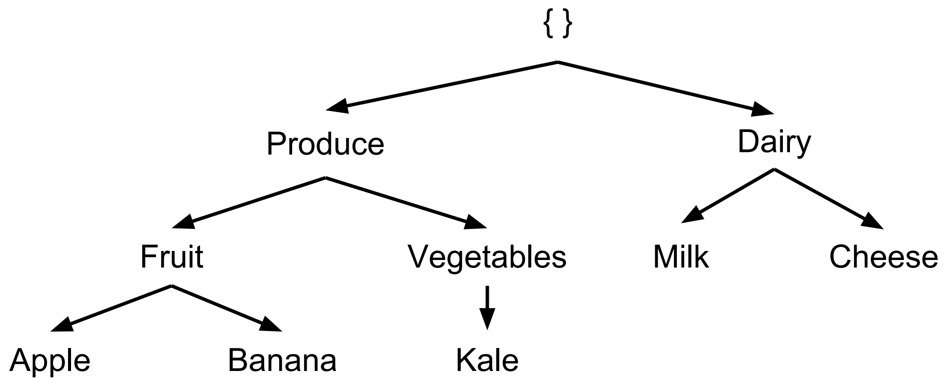


Figure 1: Example Taxonomy G

find the frequentness probability of a particular itemset, a discrete probability distribution function is calculated, which describes the probability of the itemset's support being a certain value, and is based on possible world semantics.

Given an uncertain database T and a taxonomy G, no method currently exists to mine for generalized frequent itemsets in uncertain databases. The major obstacles involved to formulate such a method is two-fold: 1) A method needs to be formulated to calculate the probability of a generalized item occurring within a particular transaction according to possible world semantics; 2) An algorithm is needed which will enumerate candidate itemsets (including generalized items) in an efficient manner. In this paper, we formulate a method calculating the probability of a particular generalized item occurring within a particular transaction. This method can be used to extend the original uncertain database, to then mine for so-called *probabilistic generalized frequent itemsets* (PGFIs). We further present an algorithm for mining these concepts from uncertain databases, and provide an experimental evaluation of the algorithm.

The rest of the paper is laid out as follows: section 2 provides preliminary concepts necessary for understanding the rest of the paper (including concepts such as generalized frequent itemset mining and the uncertain data model used); section 3 disseminates the new PGFI concepts and algorithm `PGFIM` for probabilistic generalized frequent itemset mining; section 4 provides the reader with an experimental evaluation of the newly formulated algorithm; and finally, section 6 provides a conclusion.

2 Preliminaries

2.1 Generalized Frequent Itemsets

The taxonomy G is represented as a weakly connected directed acyclical graph; formally, as the pair $G = (A, E)$, where A and E both sets, and $E \subseteq A \times A$. The elements of A are vertices, which represent distinct items; and the elements of E represent directed edges between elements of $A = \{a_1, a_2, \ldots, a_m\}$. If a directed edge between a vertex a_i and a_j, a_j is called a child of a_i, and conversely, a_i is called the parent of a_j. Further, G has the constraint that it is weakly connected, that is, ignoring the direction of the edges within G, a path must exist from all vertices to all other vertices. Further, it must also be acyclical, i.e., there is no path from a node to itself. A taxonomy G contains vertices representing both items found in the databases (leaf items), and those which are generalized or abstracted (non-leaf) items. A vertex $a \in A$ represents a non-leaf item, if and only if, a has at least one child. Also, a vertex a represents a leaf item, if and only if, a has no children. The set of vertices representing items are partitioned into the two disjoint sets: $A_L = \{a \mid a \text{ is a leaf item in } G\}$ and $A_{NL} = \{a \mid a \text{ is a non-leaf item in } G\}$. Also, let the taxonomy G define the relationships (e.g., ancestor and descendent) among each of the items $a \in A$. If a path in G exists from some item a' via directed edges to another item a, we say a' is an ancestor of a; and conversely, that a is a descendent of a'. Let the set $D(a) = \{x \mid x \in A_L \wedge\ x \text{ is a descendent of } a\}$ denote the set of all descendent leaf items in G of an arbitrary non-leaf item $a \in A_{NL}$.

A generalized itemset I is defined to be any subset of A, i.e., $I \subseteq A$. Normally, only leaf items are found in the actual database, i.e., elements of A_L (alluded to above), and in this paper this is assumed to be the case. That is, given a database $T = \{t_1, t_2, \ldots, t_n\}$, each t_j is a subset of only A_L. For itemsets which contain only leaf items, the set inclusion and subset (i.e., $\in$ and $\subseteq$, respectively) notation meanings are intuitive. However, given the presence of non-leaf (generalized) item, the previously mentioned notions must be redefined.

Definition 1 (Generalized Item Set Inclusion) *An item $a \in A$ is an element of a set S with respect to taxonomy G, denoted as $a \in_G S$, if and* *only if, $a \in S$, or there exists an $a' \in D(a)$ such that $a' \in S$.*

Definition 2 (Generalized Itemset Subset-Superset) *An itemset $I \subseteq A$ is a subset of a set S with respect to taxonomy G, denoted as $I \subseteq_G S$, if and only if, for each item $a \in I$, $a \in_G S$.*

Further, we say that the *support* of an arbitrary *generalized itemset* $I \subseteq A$ over the database T, denoted $Sup_T(I)$, is the the number of transactions in T that contains I with respect to taxonomy G, and is defined formally as:

$$Sup_T(I) = |\{t \mid I \subseteq_G t \wedge t \in T)\}| \tag{1}$$

For example, if given the example itemset database in Figure 2 and the taxonomy in Figure 1, the support of the itemset $\{fruit, cheese\}$ is 2, because $\{fruit, cheese\} \subseteq_G t_2$, $\{fruit, cheese\} \subseteq_G t_3$, but $\{fruit, cheese\} \nsubseteq_G t_1$.

TID	Itemset
t_1	*apple*, *banana*, *kale*
t_2	*apple*, *banana*, *cheese*
t_3	*apple*, *milk*, *cheese*

Figure 2: Example Database

If the support of a generalized itemset $I \subseteq A$ is greater than or equal to some user-defined threshold *minsup*, then I is considered *frequent.* Thus, the definition of a *generalized frequent itemset* (GFI) is given below.

Definition 3 *Given a taxonomy G and an itemset database $T = \{t_1, t_2, \ldots, t_n\}$, where each $t_j \subseteq A_L$, a generalized itemset $I \subseteq A$ is considered a* generalized frequent itemset (GFI), *if and only if, $Sup_T(I) \geq minsup$, where $minsup \in [0, n]$ is a user-defined threshold.*

Thus, the problem statement for mining GFIs, is to discover for all (generalized) itemsets $I \subseteq A$, whose support is greater than or equal *minsup*.

2.2 Uncertain Data Model

The major difference between traditional frequent itemset mining and probabilistic frequent itemset mining, is that in the traditional case, one knows with certitude whether or not an item occurs within a particular transaction or not; whereas, in the probabilistic case, one may only have the probability of an item occurring within a particular transaction. Let $T = \{t_1, t_2, \ldots t_n\}$ be an *uncertain database*, in which each transaction t_j consists of a set of pairs $(a_i, Pr(a_i \in t_j))$, where $a_i \in A$ and $Pr(a_i \in t_j)$, denotes the probability of item a_i occurring in transaction t_j. The probability of a_i occurring within transaction t_j, is between 0 and 1. An *uncertain item* a_i within a transaction t_j, is one which $Pr(a_i \in t_j) \in (0, 1)$. If $Pr(a_i \in t_j) = 1$ then the item a_i is known to occur with certitude and it is a certain item in t_j[2]; otherwise a_i is called an uncertain item. Let each transaction t be partitioned into two disjoint sets—one containing all uncertain items $u_1u_2 \cdots u_{L_t}$, i.e., $Pr(u_i \in t) \in (0, 1)$, and the other which contains all certain items $c_1c_2 \cdots c_{N_t}$, i.e., $Pr(c_i \in t) = 1$. Figure 3 shows an example uncertain database, where t_1 has two uncertain items, *apple* and *banana* with existential probabilities 0.89 and 0.99, respectively, and one certain item *kale*. Transactions t_2 and t_3 both have three uncertain items and no certain items.

TID	Uncertain Itemset
t_1	$(apple, 0.89), (banana, 0.99), (kale, 1.0)$
t_2	$(apple, 0.4), (banana, 0.45), (cheese, 0.12)$
t_3	$(apple, 0.9), (milk, 0.95), (cheese, 0.20)$

Figure 3: Example Uncertain Database

A *possible world* of transaction t is a concrete instance of t which contains all certain items $c_1c_2 \cdots c_{N_t}$ and either contains each uncertain item u_i $(1 \leq i \leq L_t)$ or not. Therefore, a possible world can be derived from an L_t-bit binary string $v = v_1v_2 \cdots v_{L_t} \in \{0, 1\}^{L_t}$ through a bijection from the set of L_t-bit binary strings to the set of possible worlds of t, denoted as $W(t)$, $\phi : \{0, 1\}^{L_t} \to W(t)$, defined as follows:

$$\phi(v) = \phi(v_1v_2 \cdots v_{L_t}) = b_1b_2 \cdots b_{L_t}c_1c_2 \cdots c_{N_t}$$

where

$$b_i = \begin{cases} u_i, & \text{if } v_i = 1 \\ \epsilon, & \text{if } v_i = 0 \end{cases}$$

[2]Items having a probability of zero, of occurring within a particular transaction are not shown in the uncertain database—as they certainly do not occur.

for $i = 1, \ldots, L_t$ and ϵ is the empty symbol. Since there are 2^{L_t} L_t-bit binary strings in $\{0,1\}^{L_t}$, we must have 2^{L_t} possible worlds in $W(t)$. Let a possible world in $W(t)$ be denoted by $w(t) = \phi(v)$. The probability of possible world $w(t) = \phi(v)$, denoted by $Pr(w(t))$, is calculated by:

$$\begin{aligned} Pr(w(t)) &= Pr(\phi(v), t) \\ &= Pr(\phi(v_1 v_2 \cdots v_{L_t}), t) \\ &= \prod_{v_i=1} Pr(u_i \in t) \cdot \prod_{v_i=0} (1 - Pr(u_i \in t)) \end{aligned} \tag{2}$$

if independence among the uncertain items if assumed. From (2), we have

$$\sum_{v \in \{0,1\}^{L_t}} Pr(\phi(v), t) = 1$$

In other words, (2) defines a probability distribution for all possible worlds of $W(t)$.

In a traditional (certain) itemset database, the probability of itemset I occurring in transaction t is 1 if $I \subseteq t$, or 0 otherwise. In contrast, for an uncertain itemset database, the probability of I occurring in t, denoted as $Pr(I \subseteq t)$, is the sum of the probabilities of all the possible worlds $w \in W(t)$ which contain I. Let $W_I(t)$ be the set of possible worlds, which contain I; that is, $W_I(t) = \{w \mid w \in W(t) \wedge I \subseteq w\}$. Further, recall that the function ϕ is a bijection, and therefore, its inverse ϕ^{-1} is a mapping from $W(t)$ to $\{0,1\}^{L_t}$; that is: $\phi^{-1} : W(t) \to \{0,1\}^{L_t}$. Let S be a subset of $W(t)$. We use $\phi^{-1}(S)$ to denote the range of function ϕ^{-1} from S; that is: $\phi^{-1}(S) = \{\phi^{-1}(w) \mid w \in S\}$. Thus, $\phi^{-1}(W_I(t))$ is the set of all L_t-bit binary strings, whose corresponding possible worlds contain I, that is, for all $v \in \phi^{-1}(W_I(t))$, $I \subseteq \phi(v)$. Finally, let $\phi(v)_i$ be the i-th element of $\phi(v)$, $1 \leq i \leq L_t + N_t$, and v_i be the i-th element of v, $i \leq i \leq L_t$. With that, one may calculate $Pr(I \subseteq t)$ as:

$$\begin{aligned}
&Pr(I \subseteq t) \\
&= \sum_{v \in \phi^{-1}(W_I(t))} Pr(\phi(v), t) \\
&= \sum_{v \in \phi^{-1}(W_I(t))} \left(\prod_{v_i=1} Pr(u_i \in t) \cdot \prod_{v_i=0} (1 - Pr(u_i \in t)) \right) \\
&= \prod_{\phi(v)_i \in I} Pr(u_i \in t) \cdot \prod_{\phi(v)_i \notin I} (Pr(u_i \in t) + (1 - Pr(u_i \in t)) \\
&= \prod_{\phi(v)_i \in I} Pr(u_i \in t) \\
&= \prod_{u_i \in I} Pr(u_i \in t)
\end{aligned} \tag{3}$$

Since $Pr(c_i \in t) = 1$ for a certain item in t, one can calculate $Pr(I \subseteq t)$ simply as:

$$Pr(I \subseteq t) = \prod_{a \in I} Pr(a \in t) \tag{4}$$

The set of all possible worlds W induced by *all* transactions in the uncertain database $T\{t_1, \ldots, t_n\}$ is the Cartesian product of $W(t_j)$, $j = 1, \ldots, n$ as follows:

$$W = W(t_1) \times W(t_2) \times \cdots \times W(t_n) \tag{5}$$

TID	Uncertain Itemset
t_1	$(banana, 0.99)$
t_2	$(banana, 0.45), (kale : 0.4)$

(a) Simplified Example

	Possible Worlds
$W(t_1)$	$\langle\ \rangle$, $\langle banana \rangle$
$W(t_2)$	$\langle\ \rangle$, $\langle banana \rangle$, $\langle kale \rangle$, $\langle banana, kale \rangle$

(b) Possible Worlds

Figure 4: Simple Example of Possible Worlds

TID	Possible Worlds
w_1	$\langle\ \rangle, \langle\ \rangle$
w_2	$\langle\ \rangle$, $\langle banana \rangle$
w_3	$\langle\ \rangle$, $\langle kale \rangle$
w_4	$\langle\ \rangle$, $\langle banana, kale \rangle$
w_5	$\langle banana \rangle$, $\langle\ \rangle$
w_6	$\langle banana \rangle$, $\langle banana \rangle$
w_7	$\langle banana \rangle$, $\langle kale \rangle$
w_8	$\langle banana \rangle$, $\langle banana, kale \rangle$

Figure 5: $W(t_1) \times W(t_2)$

A simplified example uncertain database is shown in Figure 4(a), and the possible worlds of each transaction is shown in Figure 4(b). (Each transaction of each possible world in $W(t_j)$ is enclosed by $\langle\ \rangle$.). Further, all possible worlds $w \in W$ induced by the entire database is shown in Figure 5.

If the assumption of independence between the transactions in T is valid, the probability of a possible world $w = (w(t_1), w(t_2), \ldots, w(t_n)) \in W$ can be calculated as follows:

$$Pr(w) = \prod_{i=1}^{n} Pr(w(t_i)) \tag{6}$$

where $Pr(w(t))$ is calculated by (2).

In a traditional (certain) itemset database $T = \{t_1, \ldots, t_n\}$, the support of itemset I in transaction t, denoted $Sup_t(I)$, is 1 if $I \subseteq t$ or 0 otherwise. The support of I over the entire database T, denoted $Sup_T(I)$ is:

$$Sup_T(I) = Sup_{t_1}(I) + Sup_{t_2}(I) + \cdots + Sup_{t_n}(I) \tag{7}$$

Notice (7) is equal to the number of transactions that contain I, i.e., $Sup_T(I) = |\{t \mid I \subseteq t \wedge t \in T\}|$.

However, in an *uncertain* database T, the support of I in transaction t_j, $Sup_{t_j}(I)$, is no longer a concrete 0 or 1. Instead, it is a random variable X_j^I following a Bernoulli distribution with parameter p_j, where $p_j = Pr(X_j^I = 1) = Pr(I \subseteq t_j)$ (calculated by (4)) and $1 - p_j = Pr(X_j^I = 0) = Pr(I \nsubseteq t_j)$ (or the probability of success and failure, respectively). Therefore, in an uncertain database T, the support of I over the entire database T is a random variable $X^I = \sum_{j=1}^{n} X_j^I$ (according to (7)). The random variable X^I follows the Poisson binomial distribution with parameters $p_j = Pr(I \subseteq t_j), j = 1, \ldots, n$, if the assumption of independence between transactions is made.

The probability that $X^I = i$, $(0 \leq i \leq n)$, is:

$$Pr(X^I = i) = \sum_{S \subseteq T, |S| = i} \left(\prod_{t_j \in S} p_j \cdot \prod_{t_j \in T-S} (1 - p_j) \right) \tag{8}$$

Equation (8) is true, because the set $S \subseteq T$ has exactly i transactions, and the probability of an arbitrary possible world $w \in W$ in which all i transactions in S contain I and the rest (n - i) do not, is equal to $\prod_{t_j \in S} p_j \cdot \prod_{t_j \in T-S} (1 - p_j)$. Given *minsup* and a confidence threshold $\tau \in [0, 1]$, an itemset I is said to be *frequent* with confidence τ, if and only if, $Pr(X^I \geq minsup) \geq \tau$. One can calculate $Pr(X^I \geq minsup)$ using the following formula:

$$\begin{aligned} &Pr(X^I \geq minsup) \\ &= \sum_{S \subseteq T, |S| \geq minsup} \left(\prod_{t_j \in S} p_j \cdot \prod_{t \in T-S} (1 - p_j) \right) \end{aligned} \tag{9}$$

Definition 4 *Given an uncertain database T and an itemset I, I is a* probabilistic frequent itemset (PFI) *with confidence τ, if and only if $Pr(X^I \geq minsup) \geq \tau$, where $minsup \in [0, n]$ and $\tau \in [0, 1]$ are user-defined thresholds. [5]*

Finally, the problem of mining probabilistic frequent itemsets, is to discover all itemsets $I \subseteq A$ such that $Pr(X^I \geq minsup) \geq \tau$, where *minsup* and τ are user-defined thresholds.

3 Mining Generalized Probabilistic Frequent Itemsets

In section 2.1, the problem statement and theory behind the mining of arbitrary generalized frequent itemsets $I \subseteq A = A_L \cup A_{NL}$—given a taxonomy G and an itemset database T, in which each transaction t is a subset of A_L—was disseminated. Further, in section 2.2, the problem statement and theory behind the mining of probabilistic frequent itemsets from uncertain databases was presented. In that domain, no taxonomy exists. Instead each transaction t is a set of pairs $(a_i, Pr(a_i \in t)$, where $a_i \in A$ and the $Pr(a_i \in t)$ denotes the probability of item a_i appearing in transaction t. So far, no research has been done to formulate a method for mining generalized itemsets from uncertain databases.

Definition 5 *Given a taxonomy $G = (A, E)$, $A = A_L \cup A_{NL}$, and an uncertain database $T = \{t_1, \ldots, t_n\}$, which only contains items in A_L, an itemset $I \subseteq A$ is considered a* probabilistic generalized frequent itemset *(PGFI) with confidence τ, if and only if, $Pr(X^I \geq minsup) \geq \tau$, where $minsup \in [0, n]$ and $\tau \in [0, 1]$ are user-defined thresholds.*

Thus, the problem statement for probabilistic generalized frequent itemset mining, is to discover all itemsets $I \subseteq A$, such that $Pr(X^I \geq minsup) \geq \tau$, where $minsup$ and τ are user-defined thresholds.

There are two major problems which need solving in order to successfully mine for PGFIs: 1) a way to calculate the existential probability of a generalized itemset occurring within an uncertain transaction; 2) a way to efficient calculate the aforementioned probability, and to enumerate possible generalized itemset candidates. The first problem is solved in section 3.1, and the second in section 3.2.

3.1 Calculating Existential Probabilities for Generalized Itemsets

The first major problem mining for PGFIs, is to formulate a method for calculating the probability that a generalized item, $g \in A_{NL}$, occurs within a particular transaction t, denoted as $Pr(g \in_G t)$. This should be the sum of the probabilities of all possible worlds $\phi(v)$ that contains at least one leaf item that is a descendent of g. Recall that $D(g)$ is the set of leaf nodes in A_L that are descendants of g. In other words,

$$Pr(g \in_G t) = \sum_{\phi(v) \cap D(g) \neq \emptyset} Pr(\phi(v), t) \tag{10}$$

Using (10) one is able to calculate the existential probability of an arbitrary non-leaf (generalized) item g occurring within a transaction t. However, (10) requires the enumeration of all possible worlds, and is therefore infeasible for all but trivial databases. Thus, similar to (3), we re-write the (10) as:

$$\begin{aligned}
Pr(g \in_G t) &= \sum_{\phi(v) \cap D(g) \neq \emptyset} Pr(\phi(v), t) \\
&= 1 - \sum_{\phi(v) \cap D(g) = \emptyset} Pr(\phi(v), t) \\
&= 1 - \sum_{D(g) \subseteq \overline{\phi(v)}} Pr(\phi(v), t) \\
&= 1 - \prod_{x \in D(g)} (1 - Pr(x \in t)) \\
&\cdot \prod_{y \notin D(g)} (Pr(y \in t) + (1 - Pr(y \in t))) \\
&= 1 - \prod_{x \in D(g)} (1 - Pr(x \in t))
\end{aligned} \tag{11}$$

where $\overline{\phi(v)}$ is the complement of $\phi(v)$, i.e., $\overline{\phi(v)} = A_L(t) - \phi(v)$. Here $A_L(t)$ is the set of items in t, $u_1 \cdots u_{L_t} c_1 \cdots c_{N_t}$, and possible world $\phi(v)$ is a subset of A_L. If $x \in D(g)$ is a certain item c_j, we have $Pr(x \in t) = 1$, then $Pr(g \in t) = 1$. This is consistent with the understanding that if a leaf descendent of a generalized item g occurs in t with certitude, then g also occurs in t with certitude.

Example 1 *Calculating the probability of the generalized item $fruit$ occurring in transaction t_1 in Figure 3, given that $D(fruit) = \{apple, banana\}$, can be done as follows: $Pr(fruit \in_G t_1) = 1 - (1 - Pr(apple \in t_1)) \cdot (1 - Pr(banana \in t_1)) = 0.9989$.*

When calculating the probability of a generalized item, one may choose to do so in an ad-hoc manner, or once there is a need to calculate the probability of the item, it can be done for every transaction in T and the corresponding probability added to each transaction—creating an extended database. In this way, future need for the probability can be simply “looked-up”.

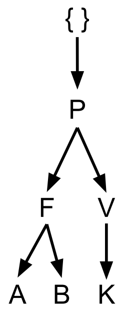


Figure 6: Simplified Taxonomy G

3.2 Efficient Enumeration & Probability Calculation

To discover all PGFIs, two salient questions must be addresses: 1) what type of enumeration scheme will be used to guide the mining process? 2) how will the probability distribution of X^I for an arbitrary $I \subseteq A$ will be calculated?

To simplify discussion of the enumeration technique used, the taxonomy shown in Figure 1 has been recreated in Figure 6, where each item has been abbreviated using the first letter of the corresponding item, and the generalized item *Dairy* (and its descendants) have been pruned.

In [20], Sriphaew et al. present the `SET` algorithm. That algorithm uses an efficient technique to enumerate candidate generalized itemsets in a *certain* itemset database. In this paper, we choose to use its fundamental enumeration scheme to determine which candidate generalized itemsets to enumerate and in which order. The `SET` algorithm's enumeration technique is performed in a top-down fashion, which uses both the subset-superset relationship, defined in Definition 2, over the items in A and the parent-child relationships defined by the taxonomy G. In simple terms, the search space is enumerated in such a fashion, as to ensure a candidate is only generated if all of its subsets are also frequent; and thus, eliminating the gratuitous enumeration of candidates that could not possibly be frequent. This is done using the downward closure property of generalized and non-generalized itemsets. Under Definition 2, the generalized item P in Figure 6 is the smallest subset of all other possible generalized itemsets, with respect to G, because P or a descendent of P must be in any generalized itemset. Conversely, ABK is the largest superset. Thus, when we say the `SET` algorithm is performed in a top-down manner, we mean enumeration starts with the most general (smallest subset) candidates, to the most concrete (largest superset). Thus, the following subset-superset relationship exists, with respect to G, between P, ABK, and any arbitrary generalized itemset I: $P \subseteq_G I \subseteq_G ABK$. All candidate generalized itemsets enumerated by the `SET` algorithm using the simplified taxonomy in Figure 6, are shown in Figure 7. In Figure 8, we see the result of the enumeration tree, given that the generalized item V is found to be infrequent. In particular, one sees that all itemsets containing V or any superset of V i.e., any children of V, are not enumerated. The reader is encouraged to see [20] for full details.

Next, one must answer how the probability distribution of X^I for an arbitrary $I \subseteq A$ is calculated. Recall that X^I follows a Poisson binomial distribution. In [5], Bernecker et al. disseminated a method that uses a dynamic programming approach to calculate $Pr(X^I \geq minsup)$ in $O(|T|)$ time. To do so, calculating $Pr(X^I \geq i)$ is recast into the problem of calculating $Pr(X^I \geq i, j)$, which is the probability of X^I being greater than or equal i in the first j transactions of T.

Thus, the recursive equation used to drive the dynamic programming algorithm is (the reader is encouraged to see [5] for full details):

$$\begin{aligned} Pr(X^I \geq i, j) &= Pr(X^I \geq i-1, j-1) \cdot Pr(I \subseteq t_j) \\ &+ Pr(X^I \geq i, j-1) \cdot (1 - Pr(I \subseteq t_j)) \end{aligned}$$

$$\text{where } Pr(X^I \geq 0, j) = 1 \ \forall . 0 \leq j \leq |T|$$
$$Pr(X^I \geq i, j) = 0 \ \forall . i > j$$

4 Experimental Evaluation

In order to perform an experimental evaluation of the `PGFI` algorithm, two types of input must be provided: a taxonomy G and an uncertain transaction databases T. In the next two subsections we tackle how each in turn is generated.

4.1 Taxonomy Generation

Previous research has tended to use small experimental taxonomies, which usually have a depth of 1–5. Further, they do not take into account the

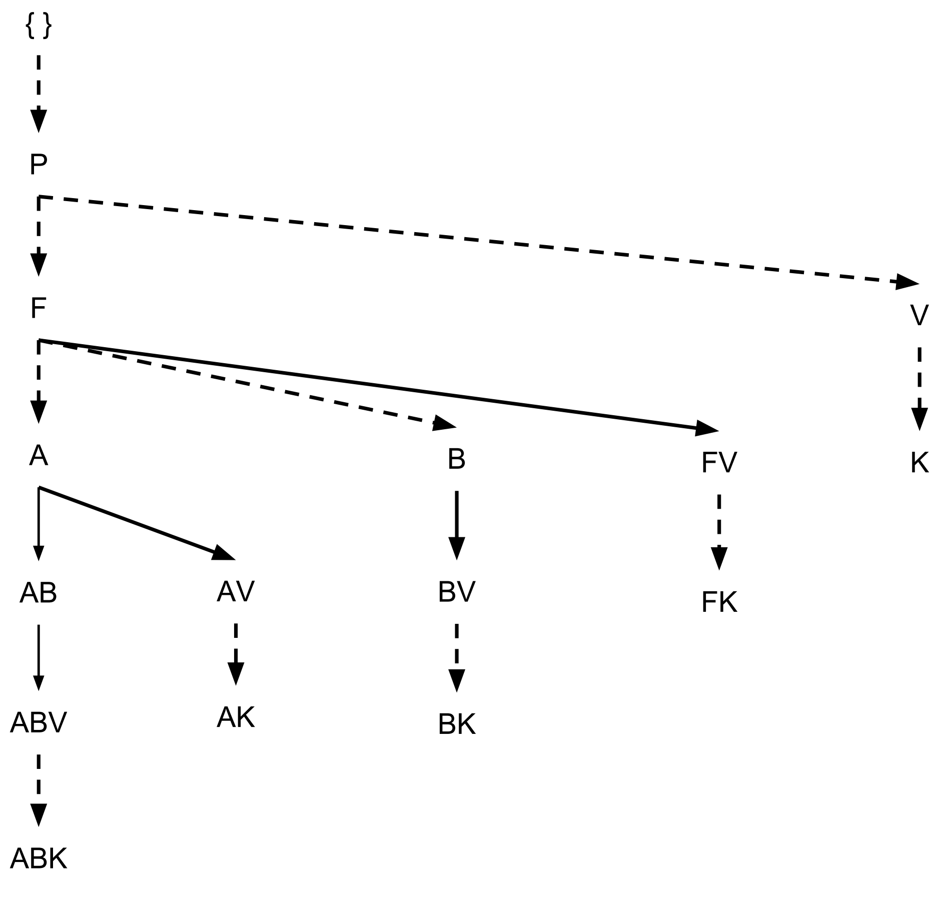


Figure 7: Example SET Enumeration

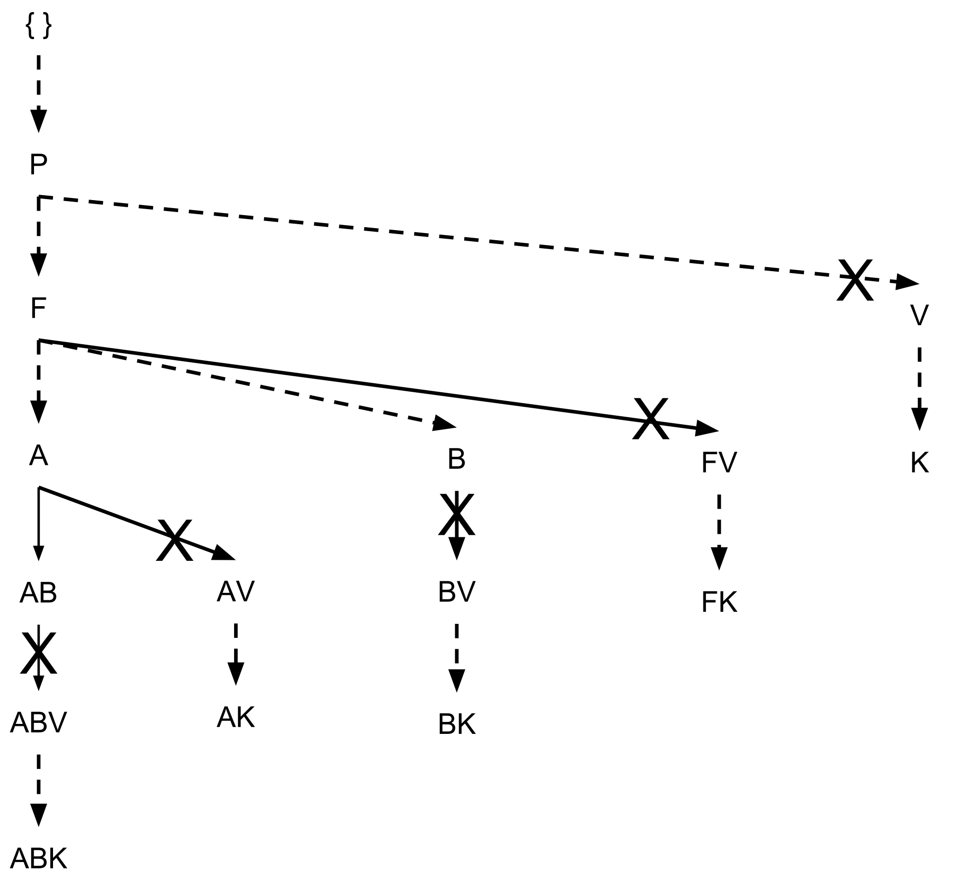


Figure 8: Pruned SET Enumeration

possibility of an item having more than one direct parent, which is feasible given that the taxonomy is a generic connected directed acyclical graph. The scheme which this paper devises allows for a rich diversity of possible experimental taxonomies to be generated using parameters V and E, the number of non-leaf nodes (excluding the empty root) and the number of uniformly distributed random edges connecting those nodes, respectively. Once the the number of non-leaf nodes and random edges have been created, a root node is created to connect all weakly connected components of the graph. Lastly, $|A_L|$ nodes are created (representing the items of the databases) are equally and randomly distributed as children among the nodes of the graph which have no descendants. Figures 9, 10, and 11 show random taxonomies generated with $V = 5$ and various E, where solid arrows indicate random edges created with parameter E, and nodes representing the $|A_L|$ items of database T in the square box. Dashed arrows denote both the connecting of the weakly connected components of the initially created graph, and the connecting of leaf $|A_L|$ items to the graph. Figure 9 shows a random taxonomy generated for with $E = 0$. In Figure 10 two random taxonomies are shown with $E = 3$. Lastly, in Figure 11, two possible random taxonomies are shown with $E = 5$.

By adjusting these parameters, a rich diversity of taxonomies can be generated. The smaller the value of V, the fewer internal (generalized) item nodes there will be, and the lower the depth of the taxonomy—if E is kept constant. If V is constant, a larger E results in a more connected (fewer weakly connected components) and possibly deeper graph. V should be limited by the number of leaf nodes, $|A_L|$, because $|A_L|$ leaf nodes will be evenly distributed and connected to the nodes without children in the generated graph. In this paper, we limit V to be less than or equal to $|A_L|$. Once V is fixed, we use E to control the connectedness and depth of the graph generated. For example, $E = 0$ gives a single-level completely disconnected graph. After adding the root node and connecting the leaf nodes, the taxonomy will be a three-level flat tree as shown in Figure 9. Increasing E causes the graph to become more connected and its depth to increase. Figure 10 shows two graphs generated with $V = 5$ and $E = 3$, with two connected components and three levels. After adding the root node and connecting the leaf nodes, the two taxonomies have five levels, but have different structures. Figure 11 shows two graphs generated with $V = 5$ and $E = 5$.

To facilitate the creation of experimental taxonomies, the statistical analysis software R and the package igraph was used to craft a script to randomly

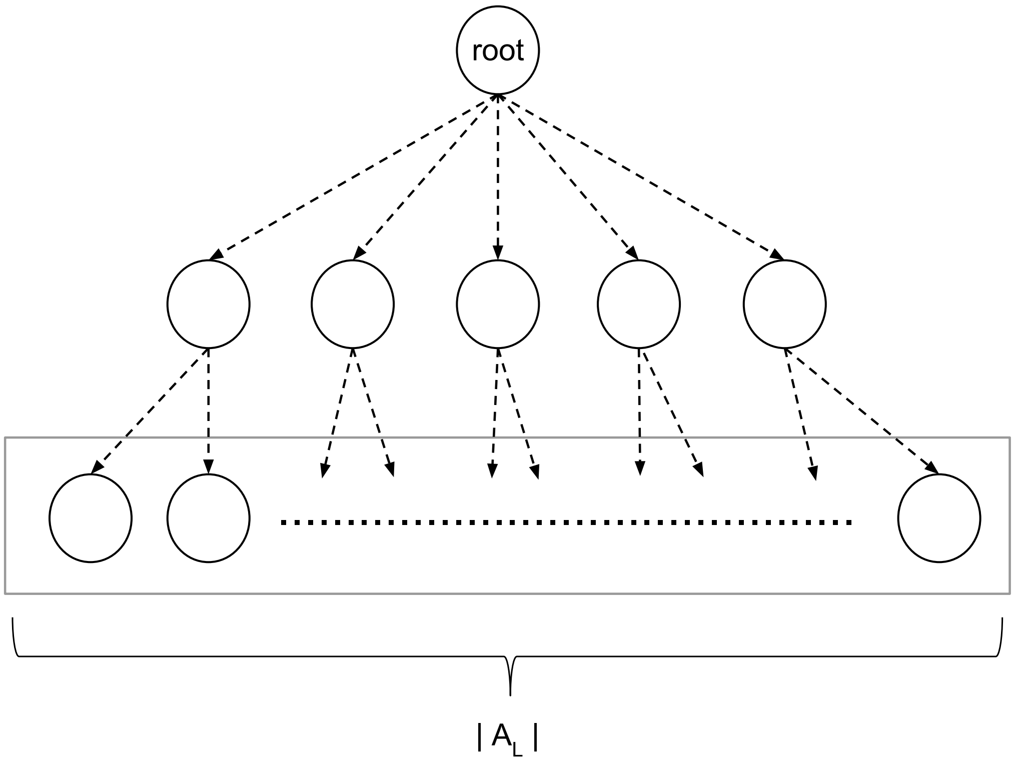
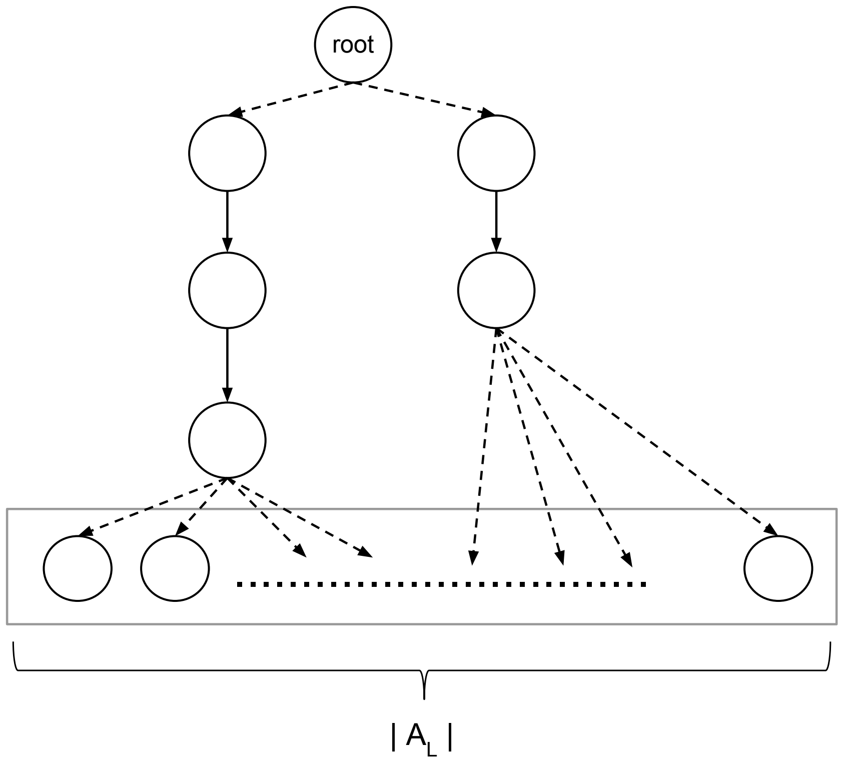
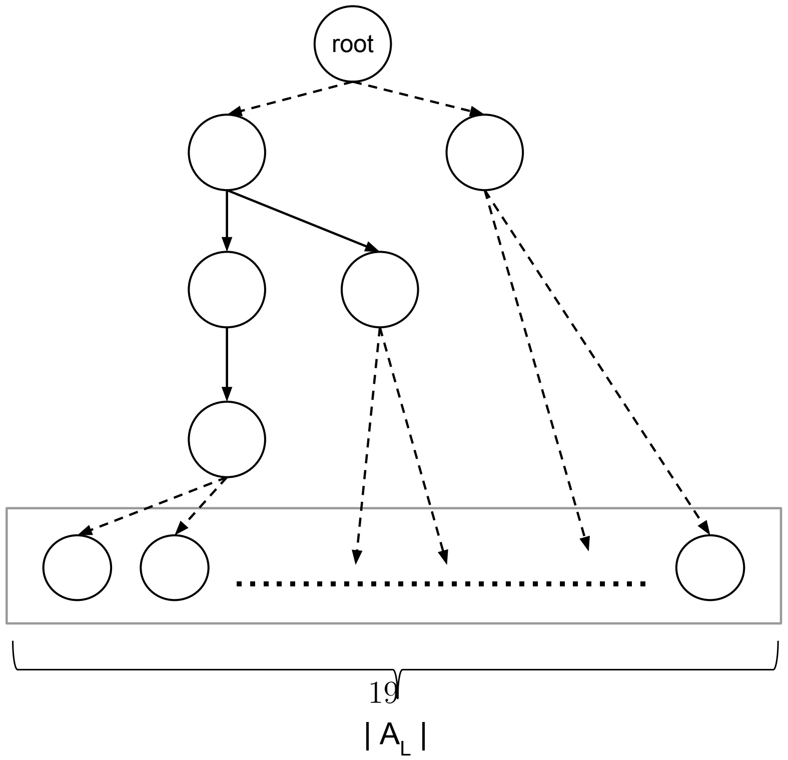


Figure 9: Example Random Taxonomy $V5E0$

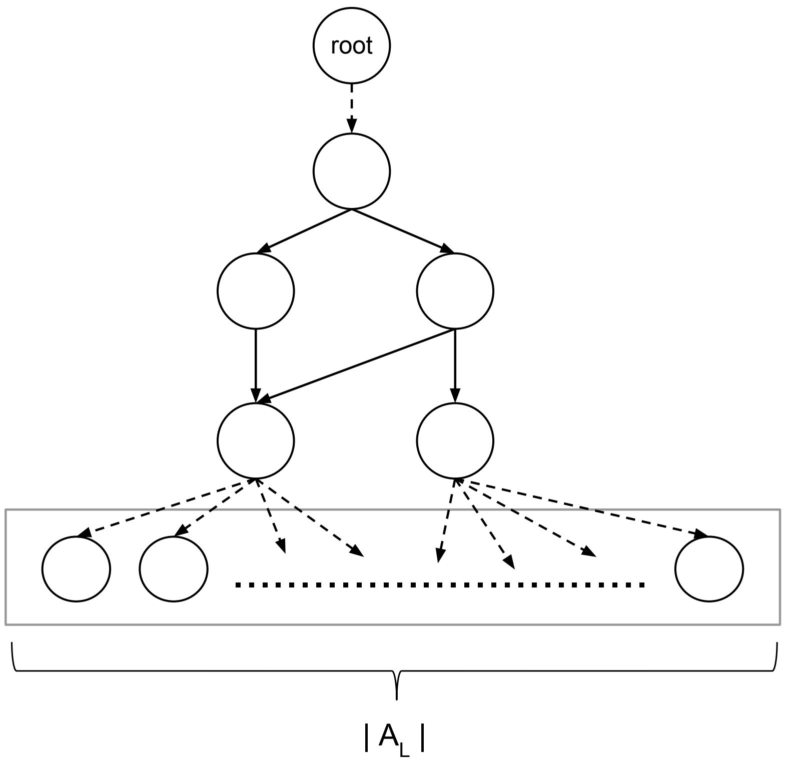


(a) Random $V5E3$

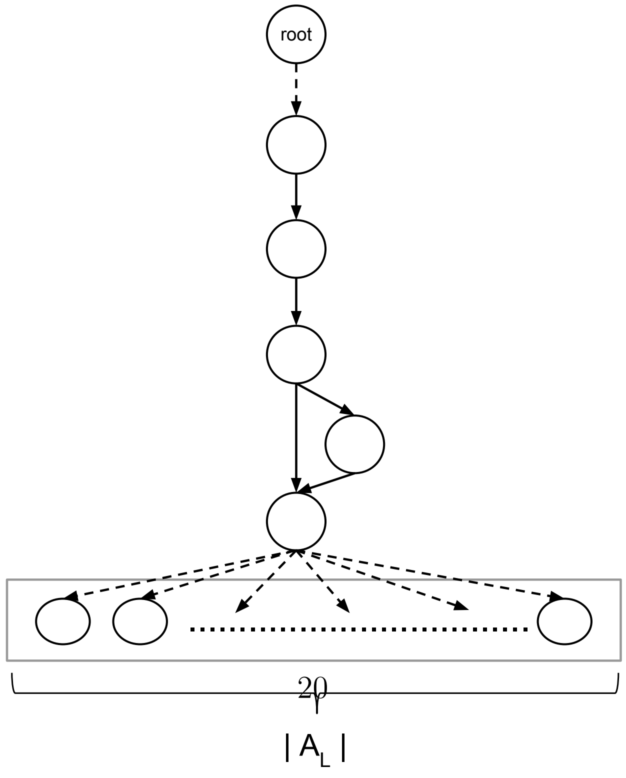


(b) Random $V5E3$

Figure 10: Example Random Taxonomies $V5E3$



(a) Random $V5E5$



(b) Random $V5E5$

Figure 11: Example Random Taxonomies $V5E3$

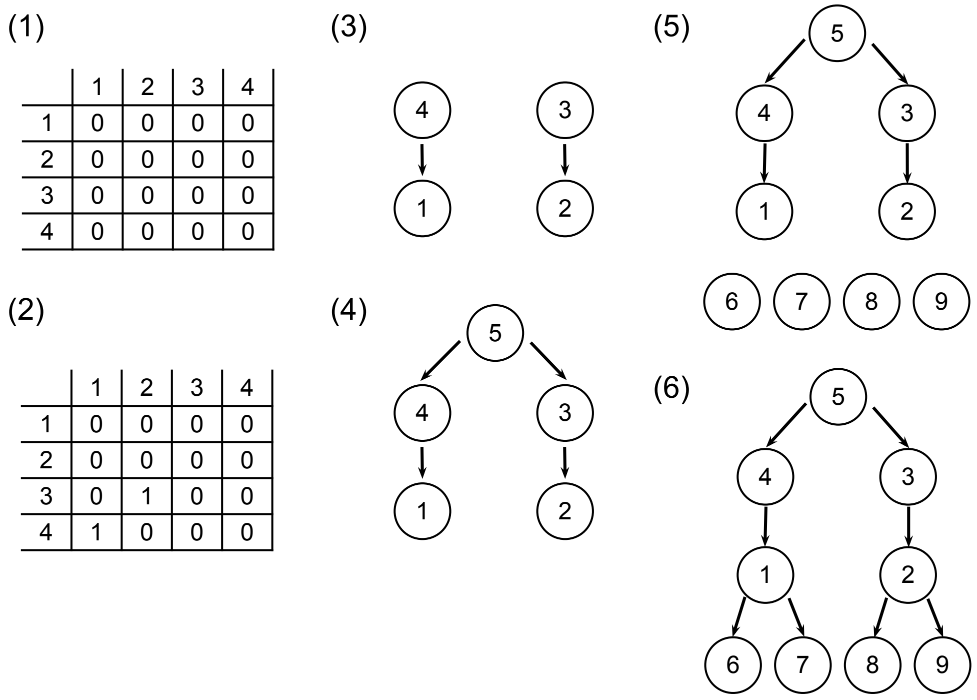
generate various taxonomical structures, and does so in the following steps:

1. A lower triangular binary matrix is created of size V. Being a lower triangular matrix, ensures the graph is directed and acyclical. A 1 in the i-th row and j-th column represents a directed edge form vertex i to vertex j.
2. Exactly E number of 1's are placed in a uniform random manner into the matrix, where E denotes the number of edges in the initial taxonomy.
3. The matrix is used as input to the igraph R package, to create a graph in the igraph format.
4. All weakly connected components are found in the graph. A root vertex is created, and a directed edge from the root to all nodes which have no incoming edges—in each weakly connected component—is created.
5. A vertex is then created for each of the leaf items in the uncertain database; that is $|A_L|$ vertices are created.
6. Each of the $|A_L|$ leaf vertices are evenly added as children to all vertices in the taxonomy which have no children.

Figure 12 shows the six steps above for creating a taxonomy of $V = 4$ and $E = 2$ (denoted as $V4E2$)—and 4 leaf nodes, i.e., $|A_L| = 4$. For a given dataset, four taxonomies are generated, all following the naming convention: V <parmater V> E <parameter E>; specifically, the four taxonomies created will be named $V\lceil\frac{|A_L|}{2}\rceil E0$, $V\lceil\frac{|A_L|}{2}\rceil E\lceil\frac{|A_L|}{4}\rceil$, $V|A_L|E0$, and $V|A_L|E\lceil\frac{|A_L|}{2}\rceil$. For example, if $|A_L| = 4$, then the following four taxonomies will be created: $V2E0$, $V2E1$, $V4E0$, and $V4E2$.

4.2 Dataset Generation

All datasets used in this evaluation were taken from the Frequent Itemset Mining Dataset Repository <http://fimi.ua.ac.be/data/>. However, since the datasets are exact or certain, transforming them into uncertain datasets was required. The procedure used to perform this transformation was done as follows: for each certain item in a transaction, the item is copied to the new uncertain dataset; a random probability $p \in (0, 1]$ is then chosen from



(1)

	1	2	3	4
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

(2)

	1	2	3	4
1	0	0	0	0
2	0	0	0	0
3	0	1	0	0
4	1	0	0	0

Figure 12: Taxonomy Generation Steps

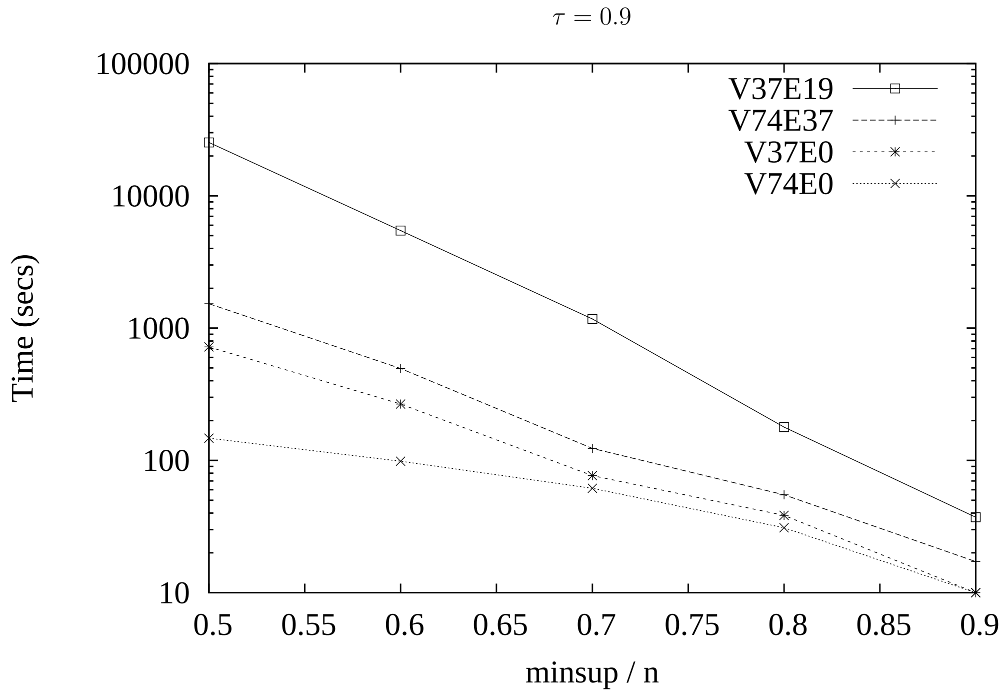
the beta distribution with parameters $\alpha = 5$ and $\beta = 1$ for this item; finally, p is assigned to the item with probability 1/2 and $1 - p$ is assigned with probability 1/2. This method of transforming a certain itemset database into an uncertain one, is different from other methods, in which the probability p is drawn from a uniform distribution [5, 4], or from a normal distribution [23]. We believe drawing probabilities from the beta distribution gives a possibly better representation of a real-world dataset, in which items are close either to existing or not, rather than being uniformly random (uniform distribution), or "ho-hum" average (normal distribution).

4.3 Algorithm Performance

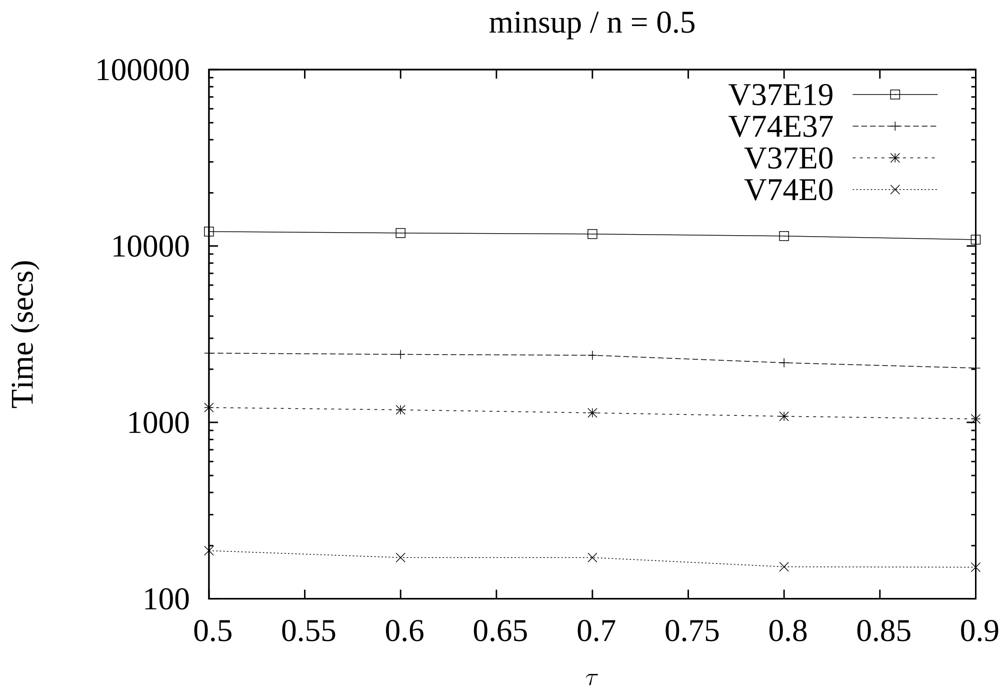
All experiments were carried out on a Intel Core 2 Quad-Core desktop computer, with 4GB of RAM, running Mac OS X v10.6; further, all code was written in C/C++ using Apple's LVM v3.0 compiler. All code and datasets used can be downloaded from <http://www.erichpeterson.com/publications/>. Experiments where carried out on both the Chess and Mushroom datasets. All execution times include the time needed to extend the database to include the probability of a generalized item occurring. When the probability of a generalized item occurring within a particular transaction is needed, the database is extended to include the existential probability for all transactions in the database. For each of the datasets, we evaluate the algorithm's performance in time when varying $minsup/n$ and τ. The results for the Chess datasets can be seen in Figure 13 and for the Mushroom dataset in Figure 14.

For each data point in Figures 13(a) and 14(a), five random taxonomies were generated and the average of the five were taken. In Figure 13(a), one sees time in seconds as a function of $minsup/n$ ($0.5 \leq minsup/n \leq 0.9$) and $\tau = 0.9$—for each of the taxonomies, i.e., V37E19, V74E37, V37E0, and V74E0. The range of parameters tested were chosen to cover a majority (above 50%) of frequency values, and to perform the experiments in a reasonable amount of time. Also, in Figure 13(b), one sees time as a function of τ ($0.5 \leq \tau \leq 0.9$) and $minsup/n = 0.5$—for each of the taxonomies. Notice the effect of τ is virtually nil.

In Figure 14(a) and Figure 14(b) one sees the effect of varying $minsup/n$ and τ, respectively, for the taxonomies V37E19, V74E37, V37E0, and V74E0.

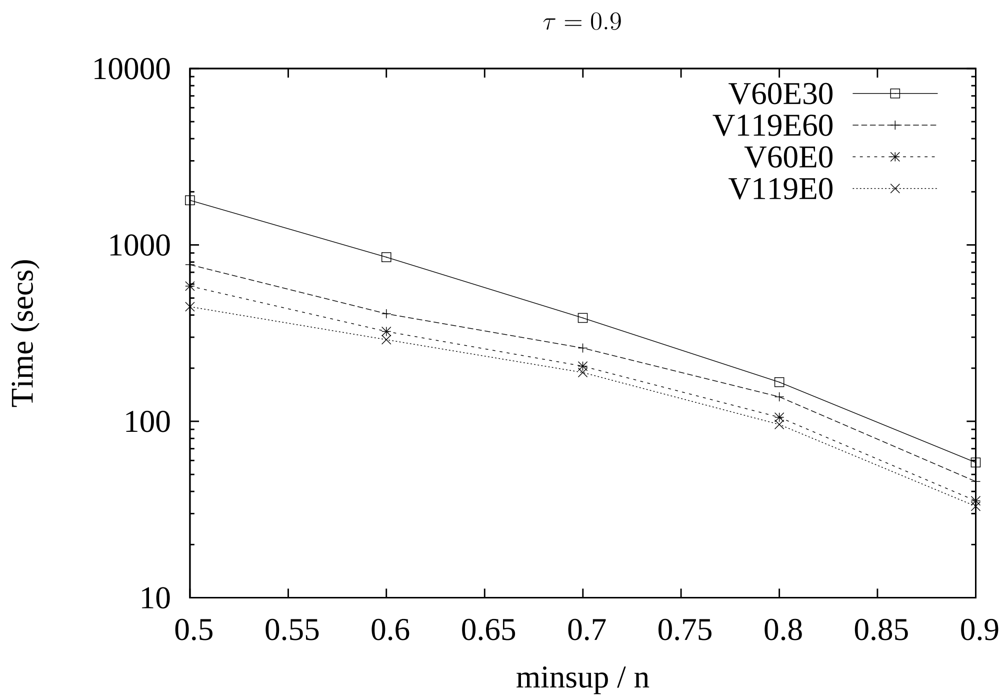


(a) Effects of Varying $minsup/n$

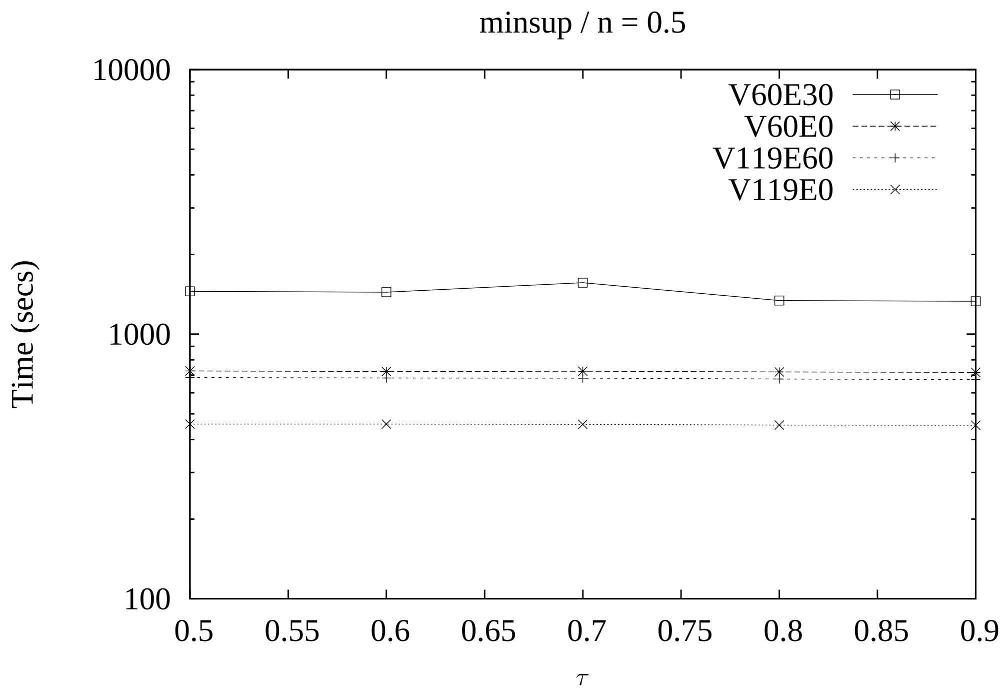


(b) Effects of Varying τ

Figure 13: Chess Dataset



(a) Effects of Varying $minsup/n$



(b) Effects of Varying τ

Figure 14: Mushroom Dataset

5 Related Work

In [19], Srikant et al. develop a naive algorithm called `Basic`, which transforms the each original transaction t into a transaction t', by adding each descendent of each item $a \in t$ to it. This new extended transaction database can then be mined using traditional itemset mining techniques—in particular the `Apriori` algorithm previously developed by them. Afterwards, the authors develop the more sophisticated algorithms `Cumulate`, `Stratify`, `Estimate`, and `EstMerge`.

In [9], Han et al. the authors present a concept similar but decidedly different from generalized itemset mining. That work is similar to generalized itemset mining in that a taxonomy is given, and association rules are generated from this taxonomy. However, "a multi-level itemset is restricted to only contain items from the same level of" [12] the taxonomy. Another noteworthy aspect of this work, is the ability to set different minimum support and confidence thresholds for different levels of the taxonomy.

Hipp et al. in [10], using insights gleamed from a study of the problem, develop the depth-first-search (DFS) algorithm `Prutex` which outperforms the breadth-first-search (BFS) `Apriori`-style algorithms of Srikant et al. [19]. Their experiments support the work in [11], and conclude that the counting of support of candidate $C = I \cup I'$ through the intersection of I's and I''s tid-list (denoted as $I.tids$, which is simply a list of transaction identifiers which support itemset I), and taking the magnitude of the resulting list, tends to be less costly then an `Apriori` BFS style algorithm when certain conditions hold. (As an example of this operation: if $I.tids = \{1, 2, 3\}$ and $I'.tids = \{2, 3, 4\}$, then the support of $C = I \cup I'$ would be calculated by the intersection $C.tids = I.tids \cap I'.tids$ ($C.tids = \{2, 3\}$) and then evaluating $|C.tids|$, which equals 2.) They note these conditions are two-fold: 1) When there is a shrinking of the "average gap between the number of actual occurrences of a candidate $C = I \cup I'$ and $min(\{|I.tids|, |I'.tids|\})$."; 2) When there is "a growing average size of candidates and frequent itemsets." The authors note that in generalized itemset mining, these two conditions tend to be true. However, as noted in [21] "a limitation of this work is the cost of checking whether their ancestor itemsets are frequent or not using a hash tree."

In [20], Sriphaew et al. disseminates the `SET` algorithm, which is similar in spirit to, but not to be confused with `SE-tree` enumeration—devised by Rymon [18] for the enumeration of traditional itemsets. This algorithm works through the merging of the conceptual (graphical) subset-superset (lattice) and parent-child relationships, which helps during "set enumeration that can avoid intensive checking [of] meaningless itemsets.". In addition, constraints prevent the counting of infrequent generalized itemsets. Finally, the algorithm uses a vertical database representation, and was shown through experimental evaluations to outperform the `Prutex` algorithm.

As was found to be the case with traditional itemset mining, generating so-called closed itemsets [26, 22, 15] were explored in the context of generalized itemset mining [21, 17] and were found to be as useful. The idea and discovery of all closed itemsets, within the field of formal concept analysis (FCA), actually predates the formulation of frequent closed itemsets in the field of data mining. However, "at that point in time, the frequency was not used for pruning, i.e., implicitly a minimum frequency of 0% was assumed." [13][3] The discovery of only closed frequent itemsets has been shown to be advantageous, because oftentimes the set of closed frequent itemsets is much smaller than the set of all frequent itemsets; and moreover, it is possible to enumerate all frequent itemsets and their supports from the set of frequent closed itemsets. Some of the algorithms in the area of mining generalized frequent closed itemsets include: `cSET` by Sriphaew et al. [21], and `g-Apriori` and `g-ARMOR` by Pudi et al. [17].

The aforementioned works are all concerned with mining generalized frequent itemsets from *certain* itemset databases, and are therefore in a different domain than this paper—where we mine from uncertain databases.

There also exist algorithms which seek to reduce the redundancy of generated association rules—sometimes through the use of definitions of "interest" and/or "essential" rules [12].

Some modern applications are known to produce incomplete or noisy data; one salient example being a sensor network [24, 28]. Further, privacy-preserving data mining applications [8, 25] in particular have a need for frequent itemset mining algorithms that operate within an uncertain data context.

Starting with the work of Agrawal et al. [2], mining frequent itemsets has been extensively studied; that research, however, has focused on so-called certain databases (i.e., where each transaction and the items it contains is known for sure). This contrasts sharply with research into mining for frequent

[3]See this reference for further background and connections between FCA and traditional itemset mining.

itemsets in databases that contain transactions or items that have existential probabilities, which has been studied in [27, 6, 7, 5, 1, 14]. In these so-called *uncertain databases*, one can not be certain about whether an itemset is frequent or not.

Recent research into mining frequent itemsets from uncertain databases include Chui at et al. [7, 6], who uses the expected support of an itemset from an uncertain database to define whether it is frequent or not. Using this technique, itemsets are considered frequent if its expected support exceeds the minimum support threshold *minsup*. However, as indicated by [5], a frequent itemset based the expected support cannot express how close the estimate is that it is frequent. In [5], Bernecker et al. defined the support distribution of an itemset (based on possible world semantics) and used it to define the frequentness probability as the sum of the probabilities of the support equal to or above the minimum support *minsup*. Thus, the frequent itemsets with conference threshold τ are the itemsets whose frequentness probabilities exceeds τ.

The aforementioned algorithms and research, is in the domain of mining frequent itemsets from uncertain databases. However, they do not allow for the mining of generalized itemsets in uncertain databases. That is, there is no taxonomy relating generalized and non-generalized items.

6 Conclusion

In this paper, we have disseminated the new concept of a *probabilistic generalized frequent itemset* (PGFI)—rooted in probabilistic mathematics and possible world semantics. Further, an algorithm to mine for such concepts was presented. Lastly, an experimental evaluation of the new algorithm—named `PGFIM`—was shown.

References

[1] C. Aggarwal, Y. Li, J. Wang, and J. Wang. Frequent pattern mining with uncertain data. *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, June 2009.

[2] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. *SIGMOD '93: Proceedings of the 1993 ACM SIGMOD international conference on Management of data*, June 1993.

[3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. *Proc. 20th Int. Conf. Very Large Data Bases*, Jan. 1994.

[4] T. Bernecker, H.-P. Kriegel, M. Renz, and F. Verhein. Probabilistic Frequent Pattern Growth for Itemset Mining in Uncertain Databases (Technical Report). *arXiv.org*, cs.DB, Aug. 2010.

[5] T. Bernecker, H.-P. Kriegel, M. Renz, F. Verhein, and A. Zuefle. Probabilistic frequent itemset mining in uncertain databases. *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 119–127, June 2009.

[6] C. Chui and B. Kao. A decremental approach for mining frequent itemsets from uncertain data. *PAKDD'08: Proceedings of the 12th Pacific-Asia conference on Advances in knowledge discovery and data mining*, pages 64–75, Jan. 2008.

[7] C. Chui, B. Kao, and E. Hung. Mining frequent itemsets from uncertain data. *Advances in Knowledge Discovery and Data Mining*, pages 47–58, 2007.

[8] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, July 2002.

[9] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. *VLDB '95 Proceedings of the 21th International Conference on Very Large Data Bases*, pages 420–431, Jan. 1995.

[10] J. Hipp, A. Myka, R. Wirth, and U. Güntzer. A New Algorithm for Faster Mining of Generalized Association Rules. *PKDD '98: Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery*, Sept. 1998.

[11] M. Holsheimer, M. Kersten, H. Mannila, and H. Toivonen. A perspective on databases and data mining. *KDD-95 Proceedings, AAAI*, pages 150–155, Apr. 1995.

[12] D. Kunkle, D. Zhang, and G. Cooperman. Mining frequent generalized itemsets and generalized association rules without redundancy. *Journal of Computer Science and Technology*, 23(1):77–102, 2008.

[13] L. Lakhal and G. Stumme. Efficient mining of association rules based on formal concept analysis. *Formal Concept Analysis, LNAI*, 3626:180–195, Jan. 2005.

[14] C. Leung, M. Mateo, and D. Brajczuk. A tree-based approach for frequent pattern mining from uncertain data. *PAKDD'08: Proceedings of the 12th Pacific-Asia conference on Advances in knowledge discovery and data mining*, pages 653–661, 2008.

[15] C. Lucchese, S. Orlando, and R. Perego. Fast and memory efficient mining of frequent closed itemsets. *Knowledge and Data Engineering, IEEE Transactions on*, 18(1):21–36, 2006.

[16] H. Mannila, H. Toivonen, and A. I. Verkamo. Efficient algorithms for discovering association rules. *Knowledge Discovery in Databases (KDD'94). AAAI Press.*, pages 181–192, 1994.

[17] V. Pudi and J. Haritsa. Generalized closed itemsets for association rule mining. *Data Engineering, 2003. Proceedings. 19th International Conference.*, pages 714–716, 2003.

[18] R. Rymon. Search through systematic set enumeration. *University of Pennsylvania Technical Report (CIS)*, Jan. 1992.

[19] R. Srikant and R. Agrawal. Mining Generalized Association Rules. *Research Report*, 1995.

[20] K. Sriphaew and T. Theeramunkong. A New Method for Finding Generalized Frequent Itemsets in Generalized Association Rule Mining. *ISCC '02: Proceedings of the Seventh International Symposium on Computers and Communications*, July 2002.

[21] K. Sriphaew and T. Theeramunkong. Mining generalized closed frequent itemsets of generalized association rules. *Knowledge-Based Intelligent Information and Engineering Systems*, pages 476–484, 2003.

[22] J. Wang and J. Han. Closet+: Searching for the best strategies for mining frequent closed itemsets. In *Proceedings of the ninth ACM SIGKDD . . .*, 2003.

[23] L. Wang, R. Cheng, S. D. Lee, and D. Cheung. Accelerating probabilistic frequent itemset mining: a model-based approach. In *CIKM '10: Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 429–438. ACM Request Permissions, Oct. 2010.

[24] S. Wang, G. Wang, X. Gao, and Z. Tan. Frequent Items Computation over Uncertain Wireless Sensor Network. *Hybrid Intelligent Systems, 2009. HIS '09. Ninth International Conference on*, 2:223–228, 2009.

[25] Y. Xia, Y. Yang, and Y. Chi. Mining association rules with non-uniform privacy concerns. In *Proceedings of the 9th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 27–34, New York, NY, USA, 2004. ACM.

[26] M. Zaki and C. Hsiao. CHARM: An efficient algorithm for closed itemset mining. *2nd SIAM International Conference on Data Mining*, Jan. 2002.

[27] Q. Zhang, F. Li, and K. Yi. Finding frequent items in probabilistic data. *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, June 2008.

[28] D. Zhi-Feng, L. Yuan-Xiang, H. Guo-Liang, T. Ya-La, and S. Xian-Jun. Uncertain Data Management for Wireless Sensor Networks Using Rough Set Theory. *Wireless Communications, Networking and Mobile Computing, 2006. WiCOM 2006.International Conference on*, pages 1–5, 2006.