UAPRIORI: AN ALGORITHM FOR FINDING SEQUENTIAL PATTERNS IN PROBABILISTIC DATA

METANAT HOOSHSADAT, SAMANEH BAYAT, PARISA NAEIMI, MAHDIEH S. MIRIAN, OSMAR R. ZAÏANE

Computing Science Department, University of Alberta, Canada
Email: {hooshsad, samaneh, naiemi,mirianho,zaiane}@cs.ualberta.ca

Uncertainty in various domains implies the necessity for data mining techniques and algorithms that can handle uncertain datasets. Many studies on uncertain datasets have focused on modeling, query ranking, discovering frequent patterns, classification models, clustering, etc. However despite the existing need, not many studies have considered uncertainty in sequential data. This paper introduces UAprioriAll, a method to mine frequent sequences in the presence of uncertainty in transactions. UAprioriAll scales linearly in time relative to the size of the dataset.

1. Introduction

1.1. *Producing Hard Copy Using MS-Word*

Statistical studies on uncertain data have recently attracted significant attention due to the fact that data produced and collected in modern applications are often uncertain or noisy. Uncertainty happens because of the limitations in the equipment, privacy reasons, information conversion or extraction, etc.

In a probabilistic transactional database, an item in a transaction can have a probability attached to it indicating its existential uncertainty of appearing in the transaction. This is a common form of uncertainty, called existential uncertainty. As an example of this assume a health-related database in which data is extracted from hand-written or text-based medical records using machine learning approaches [1]. Each attribute in this database describes a fact about the patient, and may be inaccurate for several reasons including inaccuracy in the information extraction method.

Sequential Pattern Mining or *SPM* is a well known and important problem in data mining and has been addressed by many studies [2-5]. However mining frequent sequences from uncertain datasets is still an open problem. In this

paper, we propose a solution for the aforementioned problem by introducing a new algorithm called UAprioriAll.

UAprioriAll is designed to enable mining frequent sequences in datasets with existential uncertainty. The proposed algorithm uses *expected support* as the measure of frequentness of transactions and sequences. Expected support is a metric that measures the expected frequency of an itemset/sequence in uncertain datasets. It is memory efficient and very fast to compute [6].

In this paper, we briefly overview the studies in the field of uncertain datasets in Section 2, then introduce and describe our proposed algorithm in Section 3. We discuss the experiments designed to evaluate the proposed algorithm in Section 4.

2. Related Works

Frequent sequential pattern mining or *SPM* [2-5] deals with datasets in which each transactions is considered to be associated with an id. Each id may be associated with a sequence of transactions. The definition of the problem is as follows.

Given a set of sequences where each sequence consists of a list of elements and each element consists of a set of items, and given a user-specified minimum support threshold, sequential pattern mining is to find all of the frequent subsequences, i.e., the subsequences whose occurrence frequency in the set of sequences is no less than the minimum support.

Uncertain datasets have attracted much attention recently [6]. Some studies have addressed the SPM problem for specific and limited types of uncertain datasets [2,3]. The main difference between related works and our study is the nature of the problem and the data modeling. In our model, each item has a probability of existence in each transaction, which implies that the spaces of the models in the previous work are subspaces of our model.

The data model that we propose for a realistic capture of uncertainty in sequential datasets is existential probabilistic datasets. In such datasets, each item exists within a transaction with a probability. Equation 1 shows the general form of our datasets. Each dataset D with size $|D|$ is a set of $|D|$ sequences S_i, where each sequence of size S_i contains S_i transactions $t_{i,j}$.

$$D = \{S_i : i = 1..|D|\} \tag{1}$$
$$S_i = < t_{\{i,j\}} : j = 1..|S_i| >$$
$$t_{\{i,j,k\}} = (it_{i,j,k}, pr_{\{i,j,k\}}) : k = 1..|t_{i,j}|$$

Our novel algorithm, UAprioriAll, his algorithm has three phases: a) U-Litemset; b) U-Transformation; c) U-Sequence.

2.1. *UlItemset: Mining Single Sequences*

In this phase, the sequences of size 1 (containing only one transaction) are evaluated. Each single sequence (itemsets) is marked as a candidate, if its expected support is above the minimum threshold. In the probabilistic datasets *D* the expected support is computed by Equation 2.

$$E\big(s(x)\big) = \Sigma_{\{S \in D\}} P(x \in S) \qquad (2)$$
$$P(x \in S) = 1 - \Pi_{\{T \in S\}}\big(1 - p(x \in T)\big)$$
$$P(x \in T) = \prod_{\{i \in x\}} P(i \in T)$$

We mine the *probabilistic frequent patterns* using a UApriori based technique [7]. These itemsets are put in a set called L_1. Next, each of the patterns in set L_1 is mapped to a unique integer number and L_1 is transformed using this map. Set L_1 is the output of this phase.

2.2. *U-Transformation: Simplifying the Dataset*

In this phase, we transform the sequential dataset based on set L_1. The transformed dataset has two major differences with the original dataset. First, all the infrequent itemsets of the original dataset, that is the ones that are not contained in L_1, are removed from the transformed dataset. Second, the frequent itemsets are mapped into integer numbers as in L_1.

UC-Sequence: Mining the Sequences

In this phase, we mine the frequent sequences from the transformed dataset (output of U-Transformation) using an UApriori-like algorithm. At each step, the new candidate set (C_k) is filled up based on the frequent sequences of the previous level (L_{k-1}) by evaluating $L_{k-1} \bowtie L_{k-1}$ for all two tuples that have k-2 items in common and then removes those that have infrequent subsets. The procedure starts from L_1 which was calculated in phase 2. We continue till the set L_k is empty.

To form L_k, we choose c from C_k if the expected support of c is greater than the minimum value. To calculate the expected support based on Equation 2, we need to calculate the value of $P(x \in S)$ for x which size is greater than 1.The probability by which the first k items of candidate $c \in C_m$ (m>k) appears at least once within the first j items of the sequence $s \in D$ is denoted by $P_{\{j,k\}}(c,s)$ and

computed by a recursive approach presented in Equation 3. The recursive equation allows us to benefit from the dynamic programming scheme.

$$\begin{aligned} P\{c \subseteq s\} = & \quad P_{\{|c|,|s|\}}(c \subseteq s) \\ P_{\{j,k\}}(c \subseteq s) = & \quad P_{\{j-1,k-1\}}(c \subseteq s) * p(c[k] \subseteq s[j]) + \\ & \quad P_{\{j-1,k\}}(c \subseteq s) * \big(1 - p(c[k] \subseteq s[j])\big) \\ P_{\{j,k\}}(c \subseteq s) = & \quad 0 \; if \; k > j \\ P_{\{j,1\}}(c \subseteq s) = & \quad 1 - \Pi_{\{t_l \subseteq s\}}^{\{l=1..j\},} \big(1 - p(c[1] \subseteq t_l)\big) \end{aligned} \tag{3}$$

The recursive formula is achieved by dividing the problem into two mutually exclusive states. State *a* is when s_j contains c_k and state *b* is otherwise. The probability value is the addition of the probabilities of the two states. State *a* requires two events to happen, both $c[k] \subseteq s[j]$ and *s[1..j-1]* (the *j-1* first elements of *s*) should contain at least one appearance of *c[1..k-1]* (the first *k-1* elements of *c*). State *b* also requires two events, *c[k]*$\notin$*s[j]* and *c* being a subset of *s[1..j-1]*. It is evident that states *a* and *b* are mutually exclusive. The total number of computations required can be assessed by Equation 4.

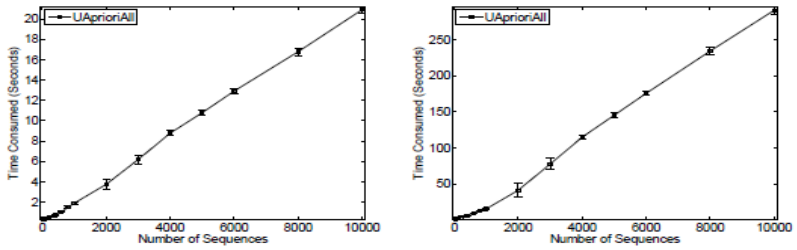
$$\begin{aligned} DC_{\{(c,s)\}} = & \quad \{P_{\{j,k\}(c,s)} : |c| - k > |s| - j \,\} \\ Computation[P(c \subseteq s)] = & \quad |s|.|c| - \frac{\{|c|^2 - |c|\}}{2} - \frac{\{|c|^2 - |c|\}}{2} \\ = & \quad |c|.(|s| - |c| + 1) \end{aligned} \tag{4}$$

Relying on the mathematical meaningfulness of the expected support, the purpose of UAprioriAll is to find all sequences that have higher expected support than the predefined threshold. It is easily verifiable that our algorithm is sound. An induction using the downward closure lemma [7] can prove the algorithm completeness. In addition, the algorithm terminates when the set L_k is empty, that is worst case happens at the K_{max}-th level, where K_{max} is the maximum length of the sequences. Therefore, total correctness can be proven for UAprioriAll.

3. Experiments

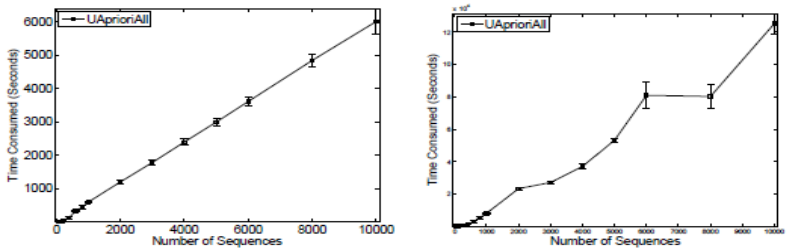
As there are no uncertain sequential datasets publicly available similar to other studies on uncertain data (such as in [8]), we used synthetic datasets in our experiments. Each generated synthetic data is characterized by two parameters: the number of sequences *L* and the total number of items *I*. Our goal is to

investigate the effect of L on the time consumption, so we set I to 20 and L is varied from 50 to 10000.



(a) Time consumption of *UApriori-All* when minimum support = 0.5

(b) Time consumption of *UApriori-All* when minimum support = 0.33



(c) Time consumption of *UApriori-All* when minimum support = 0.25

(d) Time consumption of *UAprioriAll* when minimum support = 0.10

Figure 1-Time consumption of UAprioriAll with different support values

The number of transactions in each sequence is a monotonically distributed pseudo-random number from 1 to 0. We randomly select the items within the transaction, where all items have equal chances. The existence probability attached to each item within the transaction is a randomly generated number between 0 and 1.

To increase the reliability of the results, for each value of L, 5 datasets were generated and the method was applied 5 times to each dataset, and then averaged. In this process, we used the *Scaling Method*. This method scales down a large dataset by randomly eliminating some transactions, to get a smaller dataset with lower number of sequences.

The experiments were carried out on a machine with 2.66 GHz clock speed and 8 GB of RAM. In the implementation of the algorithm we adopted a free online Java implementation of Apriori [9].

The experiments include the time consumption of the algorithm based on the minimum support. This measure is important because decreasing it may cause a dramatic drop in performance and may affect the consistency in the behavior of UAprioriAll. Setting the minimum support in real world applications depends greatly on the domain.

Figure 1 shows the time scalability of UAprioriAll with different values of minimum support. UAprioriAll grows linearly based on the number of sequences.

4. Conclusion

In this paper, we proposed a novel uncertain sequential pattern mining algorithm employing the expected support. UAprioriAll considers attribute level (existential) probability, and mines the frequent sequential patterns. We showed the feasibility of our new algorithm in terms of time consumption. Based on the results of the experiments, UAprioriAll's runtime is linearly scalable based on the number of sequences in the dataset.

5. References

1. A. M. Cohen and W. R. Hersh, A survey of current work in biomedical text mining, *Briefings in Bioinformatics* **6**, 57 (2005).
2. J. Pei, J. Han, B. M. Asl, H. Pinto, Q. Chen, U. Dayal and M. C. Hsu, Prefixspan mining sequential patterns efficiently by prefix projected pattern growth, in *ICDE '01: Proceedings of the 17th International Conference on Data Engineering*, (Heidelberg, Germany, 2001).
3. J. Yang, T. J. Watson, W.Wang, P. S. Yu and J. Han, Mining long sequential patterns in a noisy environment, in *SIGMOD '02: Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, (Madison, Wisconsin, USA, 2002).
4. M. J. Zaki, SPADE: An Efficient Algorithm for Mining Frequent Sequences, *Mach. Learn.* **42**, 31 (2001).
5. R. Agrawal and R. Srikant, Mining sequential patterns, in *Data Engineering, 1995. Proceedings of the Eleventh International Conference on*, (Taipei, Taiwan, 1995).
6. C. C. Aggarwal and P. Yu, A Survey of Uncertain Data Algorithms and Applications, *IEEE Transactions on Knowledge and Data Engineering* **21**, 609(May 2009).
7. C. C. Aggarwal, Y. Li, J. Wang and J. Wang, Frequent pattern mining with uncertain data, in *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, (Paris, France, 2009).
8. Q. Zhang, F. Li and K. Yi, Finding frequent items in probabilistic data, in *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, (Vancouver, Canada, 2008).
9. P. Fournier-Viger, Algorithms/frequent itemset mining algorithms (2010), url: http://www.philippe-fournier-viger.com/spmf/index.php.