Differentially Private Frequent Sequence Mining
via Sampling-based Candidate Pruning

Motivation

Frequent Sequence Mining (FSM) is one of the fundamental data mining tasks with broad applications (e.g., Web usage analysis and Location-based recommendation system).

<table>
<thead>
<tr>
<th>ID</th>
<th>Record</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>a-b-c-d</td>
<td>freq 1-seq</td>
</tr>
<tr>
<td>200</td>
<td>a-b-c-d</td>
<td>freq 2-seq</td>
</tr>
<tr>
<td>300</td>
<td>a-b-c-d</td>
<td>freq 3-seq</td>
</tr>
<tr>
<td>400</td>
<td>a-b-c-d</td>
<td>freq 4-seq</td>
</tr>
<tr>
<td>500</td>
<td>a-b-c-d</td>
<td>freq 5-seq</td>
</tr>
</tbody>
</table>

Goal: design a differentially private FSM algorithm which can find frequent sequences with high accuracy.

Challenges: the large number of generated candidate sequences leads to a large amount of perturbation noise required by differential privacy.

Our contributions:
1. Introduce a sampling-based candidate pruning technique to improve the utility of the mining results
2. Develop a novel differentially private FSM algorithm PFS2
3. Evaluate our private algorithm with real-world datasets

Overview

PFS2 algorithm discovers frequent sequences in order of increasing length

Basic Idea: Partitioning the original database by random sampling and Using the kth sample database for pruning candidate k-sequences (k = 1, 2, …)

The main step of discovering frequent k-sequences
1. Generate candidate k-sequences Ck based on the downward closure property
2. Use the kth sample database to prune the candidate k-sequences (the core of our PFS2 algorithm)

The step of discovering frequent k-sequences

1. Compute the noisy support of remaining candidate k-sequences Ck in the original database and output the frequent k-sequences

Differential Privacy

Privacy Risk:
- Sensitive Data
- Statistical Queries
- Query results

Differential Privacy: the computation output remains roughly the same with or without the presence of any single tuple.
- Differential Privacy can be achieved by adding Laplace noise:

\[ F(D) = F(D) + \text{Lap}(GS_n / \alpha) \]

Sequence Shrinking

Example: Given candidate 3-sequences \{abc, bcd, blh, blbd\} and length constraint \( l_{max} = 5 \), we shrink sequence \( S = \text{abc} \text{bcd} \text{bd} \text{bd} \text{cd} \) as follows:

Step 1 Irrelevant Item Deletion: The items not contained in candidate sequences do not contribute to the supports of the candidate sequences e.g., \( S = \text{abc} \text{bcd} \text{bd} \text{bd} \text{cd} \) contains same candidate sequences as \( S_1 = \text{abc} \text{bcd} \text{bd} \text{bd} \text{cd} \) (obtained by removing items e, f from S)

Step 2 Consecutive Pattern Compression: The k items in a candidate k-sequence come from at most k consecutive patterns e.g., \( S_2 = \text{bd} \text{bd} \text{bd} \text{bd} \text{cd} \) contains same candidate sequences as \( S_2 = \text{a} \text{bd} \text{bd} \text{bd} \text{cd} \) (obtained by using three patterns bd to replace bd bd bd bd)

Step 3 Sequence Reconstruction

Goal: construct a new sequence \( S' \) \( |S'| = l_{max} \) such that the number of common candidate k-sequences contained both in S and S’ is maximized

Main idea: iteratively append items which can introduce the maximum number of candidate k-sequences into S’ until S’ reaches the length constraint \( l_{max} \)

Data structure: we use a prefix tree (called CS-tree) to group candidate k-sequences and guide the re-construction process

A simple CS-Tree of sequences \{a, b, c, d, bl, bd, blbd\}

Threshold Relaxation

Goal: decrease the probability of misestimating frequent sequences as infrequent in the sample database

Step 1: Quantify the relationship between the actual support in the original database and the noisy support in the sample database

Step 2: Derive the relationship between the decrement of the threshold in sample databases and the probability of misestimating frequent sequences as infrequent

Result and Conclusion

Experiments
- Three real-world datasets: MSNBC, BIBLE, House_Power
- Comparison with sequential data publication algorithms Prefix and n-gram

Mining Utility vs. Threshold

- MSNBC
- BIBLE
- House_Power

Mining Utility vs. Differential Privacy

- MSNBC
- BIBLE
- House_Power

Conclusions
- We propose a sampling-based candidate pruning technique as an effective means of improving utility/privacy tradeoff in frequent pattern mining
- We introduce a novel differentially private FSM algorithm (called PFS2) which can achieve a high degree of privacy and high data utility

*Acknowledgment: This work was supported by NSFC grant 61611003, NSF grant 12177653, Fundamental Research Funds for the Central Universities grant #2014RC1103 and NSF grant #1117763.

# Emory University, Atlanta, GA