Mining Frequent Sequential Patterns with Local Differential Privacy

Huihua Xia, Wenchao Huang, Yan Xiong, and Fuyou Miao
(Corresponding author: Wenchao Huang)
School of Computer Science and Technology, University of Science and Technology of China
West Campus of USTC, Huang Shan Road, Hefei, Anhui Province, China
Email: huangwc@ustc.edu.cn
(Received May 31, 2020; Revised and Accepted Feb. 11, 2021; First Online Aug. 14, 2021)

Abstract
A massive amount of sequential data are being collected and analyzed. Mining frequent sequential patterns among these data can bring benefits for various real-life applications. However, privacy has been a major concern during the analysis of sequential data. We propose LDPSPM for mining frequent sequential patterns while satisfying local differential privacy. Specifically, each user randomly responds with a set of sanitized sequential patterns. Then the collector can estimate frequencies of the sequential patterns from the sanitized data. Moreover, we propose P-LDPSM, which further improves the performance by filtering out nonsignificant items. Theoretical analysis and extensive experiments confirm the effectiveness and efficiency of our mechanism.

Keywords: Local Differential Privacy; Privacy Preserving; Sequential Pattern Mining

1 Introduction
With the development of web, mobile and communication technologies, massive sequential data are being recorded and collected [3][12]. Specifically, sequential data are sequences of items with time correlations, e.g., mobile applications launching sequence, webpage click stream and user moving trajectory. Mining such sequential data can bring benefits for many real-life applications such as travel suggestion, city planning, advertising and etc. For example, the server can make travel suggestions for users after mining a trajectory database. In this paper, we focus on the task of sequential pattern mining, i.e., finding interesting sub-sequences in a sequence database.

Privacy is a major concern in the current era of cloud computing [13][21][22][28][29]. Sequential data contains much privacy information of users. For example, user preference may be inferred from the web browsing history [40]. Personal trajectory data can be used to infer sensitive information such as home address, health status and religious faith [11]. Instead of directly collecting and publishing these sequential data, privacy-preserving mechanisms should be considered during the mining of sequential data.

Some researchers [6][7][10][26][31][11] have studied the privacy-preserving mechanisms when processing sequential data. Most of them consider a trusted centralized server. All the users’ sequential data are collected and stored on the server. The server takes responsibility for the privacy of the users and carries out privacy-preserving mechanisms. Some of the works [6][7][9][20][31] study the problem of privacy-preserving data publishing, i.e., publish the entire sequential database after sanitizing the database. The other works [4][10][41] publish statistical information (e.g., frequent sequential patterns) of the database other than the entire database. However, the assumption of a centralized server is not practical. The data security can not be guaranteed since the server might be corrupted by attackers.

Recently, local differential privacy (LDP) [3][15][17] has become the de facto notion for privacy-preserving while avoiding the assumption of a trusted centralized server. The users take full control of their data and the server never collects the exact value of any personal data. Before sending the data to the server, the users sanitize their data locally with privacy-preserving techniques while ensuring that specific statistical information can be derived from the sanitized data. For example, Google embedded their LDP solutions called RAPPOR [17] into the Chrome browser, which enables Chrome to collect information such as the default homepage of the browser. Samsung [32] developed an LDP system for collecting user data from smart devices, which can deal with not only categorical but also numerical data.

Though solutions of achieving LDP for different tasks have been studied for years, to the best of our knowledge, LDP mechanisms for sequential pattern mining have not been studied yet. Most of the existing works assume the data type as single-valued data [17][37] or set-valued data [34][36][38]. Usually the data of each user is encoded into a binary vector [17], where each bit in the vector de-
notes whether a value is held by the user. Such methods of encoding can not be applied to sequential data since the sequential characteristics between values can not be reflected by a binary vector. Another challenge of designing an LDP protocol for sequential pattern mining comes from the size of the domain. Assume that there are \( d \) items in total, then the number of sequences of length-\( m \) (suppose there are no duplicate items in a sequence) composed of these items is \( d^m \), which is explosive when \( d \) is large.

In this paper, we solve the problem of mining the top-\( k \) sequential patterns among a set of sequences while satisfying LDP. We propose a novel and efficient mechanism called LDPSPM for mining sequential patterns with LDP. We adopt the idea of exponential mechanism from the traditional differential privacy. Instead of encoding the user data, each user randomly responds with a set of sequential patterns, and the size of the set is proved to be optimized. Moreover, to handle the problem of the large domain, we design a pruning step, which narrows down the size of the domain significantly and improves the effectiveness of our mechanism. Theoretical analysis and extensive experimental results on both synthetic and real-world datasets show the effectiveness of our mechanism and the improvement of the pruning step.

The contributions we made in this paper are as follows:

- We propose a novel and efficient LDP mechanism for mining sequential patterns. We analyze the error bound of the results theoretically and optimize the parameters based on the theoretical analysis.
- We design a pruning step for our protocol, which significantly narrows down the item domain and improves the accuracy of the results.
- We evaluate our protocols on both synthetic datasets and real-world datasets. The results show the utility and effectiveness of our protocol.

The rest of this paper is organized as follows. Section 2 reviews the related works on sequential pattern mining and local differential privacy. Section 3 gives the formal definition of LDP and reviews several existing LDP protocols. Section 4 formalizes the problem of LDP sequential pattern mining. Section 5 proposes the solution and the utility analysis of the LDPSPM method for sequential pattern mining. Section 6 improves LDPSPM with a pruning step. Section 7 gives experimental results of our mechanisms. At last, Section 8 concludes the work.

## 2 Related Work

In this section, we review the related works on both sequential pattern mining and local differential privacy.

### 2.1 Sequential Pattern Mining

The task of sequential pattern mining is to find all frequent sub-sequences in a sequence database. Much effort has been made to improve the efficiency of sequential pattern mining. AprioriAll [1] is the first algorithm for sequential pattern mining, which is inspired by the Apriori algorithm for frequent itemset mining. GSP [35] is proposed as an improved version of AprioriAll based on the breadth-first search. Several depth-first search algorithms are proposed which exclude irrelevant patterns and show more efficiency than GSP, such as Spade [43], PrefixSpan [33], Spam [2], Lapin [42], and CM-Spam [19].

The task of privacy-preserving sequential pattern mining has been studied for years. Some of the works [6, 7, 20, 31] focus on publishing a sanitized sequence database which can be used for mining sequential patterns. Other approaches aim to mine the sequential patterns directly on the original database while preserving privacy. Most of them are based on the concept of differential privacy [16]. Bonomi et al. [4] propose a two-phase differentially private protocol for mining frequent consecutive-item sequences, which utilizes a prefix tree to find candidate sequences, and then leverages a database transformation technique to refine the support of the candidate sequences. Xiang et al. [10] propose DP-MFSM for finding maximal frequent sequences based on the idea of candidate pruning. Xu et al. [41] estimate the sequences that are potentially frequent based on sample databases, and then reduce the number of candidate sequences. Li et al. [27] solve the problem of time-constrained sequential pattern mining under differential privacy, where the transition time between adjacent items in frequent sequential patterns is constrained. Le et al. [25] mine frequent sequential patterns in electronic medical record systems while considering time interval. They restrain the added noise by adding noise only to a set of candidate closed sequences. However, all these approaches assume a centralized server, which collects the exact data of the users and carries out the privacy-preserving mechanisms. If the centralized server is corrupted, the privacy of the users can no longer be preserved.

### 2.2 Local Differential Privacy

Local differential privacy (LDP) is proposed to avoid the use of the centralized server. LDP protects the privacy of users in a local manner that each user sanitizes his data before sending it to the server.

Most of the existing LDP solutions focus on frequency estimation over categorical data. Duchi et al. [15] propose the LDP mechanism for histogram estimation and theoretically analyze the optimality of utility. Google deploys its LDP algorithm called RAPPOR [17] in practice, which is the first LDP solution in real-world applications. RAPPOR encodes user’s data by a Bloom filter and then applies the randomized response [39] to perturb it. Bassily et al. [3] propose a protocol that uses random matrix projection and reduces the communication cost of each user greater than RAPPOR. Kairouz et al. [23] extend RAPPOR to support categorical attributes with arbitrary number of possible values. Wang et al. [37] propose a
general framework and compare several LDP protocols theoretically, and further optimize the parameters of the protocols to provide better utility of the results.

Other works take frequency estimation as a primitive to solve different tasks. Chen et al. [8] propose a mechanism to learn the spatial distribution of users under LDP. Kim et al. [24] finish the task of collecting indoor positioning data under LDP. Cormode et al. [14] solve the problem of answering range counting queries under LDP. Several works [54][56][38] deal with set-valued data and solve the task of frequent itemset mining. Qin et al. [34] propose a two-phase heavy hitter estimation mechanism to estimate the top frequencies of items from set-valued data. They utilize the item domain first and then improve the estimation accuracy. Wang et al. [36] extend the work of [34], which improves the effectiveness of the results and enables the frequency estimation of itemsets. Wang et al. [38] propose PrivSet for frequency estimation of single items and set cardinality estimation, which privatizes items in set-valued data as a whole, and takes full utilization of the privacy budget.

However, to the best of our knowledge, none of the existing LDP mechanisms consider the data type of sequential data and solve the task of frequent sequential data mining.

3 Background

3.1 Local Differential Privacy

Local differential privacy is a notion of privacy for data collection originated from differential privacy. A collector collects data from users in the local setting. The users perturb their data through a randomized mechanism before sending their data to the collector. In this paper, we consider the situation that each user has a sequence $s$. The formal definition of local differential privacy is as follows:

**Definition 1.** (Local Differential Privacy) A randomized mechanism $K$ satisfies $\epsilon$-local differential privacy ($\epsilon$-LDP), where $\epsilon > 0$, if and only if for any two sequences $s_1$ and $s_2$, we have

$$\forall y \in \text{Range}(K) : \Pr[K(s_1) = y] \leq e^{\epsilon} \cdot \Pr[K(s_2) = y]$$

where $\text{Range}(K)$ denotes the output domain of $K$.

Similar to the differential privacy in the centralized setting, there is a property of sequential composition [30] for $\epsilon$-LDP.

**Theorem 1.** Given a set of randomized mechanism $K_i$, each of which satisfies $\epsilon_i$-LDP, then the whole process of sequentially executing $K_i$ satisfies $(\sum \epsilon_i)$-LDP.

Given the property of sequential composition, each user can partition the privacy budget $\epsilon$ into several portions and adopt several randomized mechanisms, while the whole process satisfies $\epsilon$-LDP.

3.2 Existing Protocols

Next, we overview several existing LDP solutions. All these protocols assume that the users hold categorical data (a single value or a set of values) and aim to estimate the frequencies of the items.

**RAPPOR.** RAPPOR [18] is designed based on the idea of random response [17].

Assume there are $n$ users, and each user $u_i$ possesses exactly one item $v_i$ (an integer). The items come from a domain containing $d$ items, and a collector wants to estimate the frequency of each item. For a user with an item $v_i$, he first encodes $v_i$ into a length-$d$ binary vector $B_i$ such that $B_i[v] = 1$ and the other bits are 0. Then the user applies a random response on $B_i$ and gets a new binary vector $B'_i$, such that:

$$\Pr[B'_i[j] = 1] = \begin{cases} p = \frac{\epsilon/2}{1+\epsilon/2}, & \text{if } B_i[j] = 1 \\ q = \frac{1+\epsilon/2}{1+\epsilon/2}, & \text{if } B_i[j] = 0 \end{cases}$$

Such mechanism satisfies $\epsilon$-LDP for each user. The binary vector $B'_i$ is sent to the collector.

Upon getting all the $n$ users’ responses, the collector can estimate the frequency of an item $v$ as follows:

$$f(v) = \sum_i \mathbb{1}_{[v\in B'_i]}(v) - nq$$

where $\mathbb{1}_{[v\in Y]}(v)$ is an indicator function that

$$\mathbb{1}_X(v) = \begin{cases} 1, & \text{if } v \in Y \\ 0, & \text{if } v \notin Y \end{cases}$$

To transfer a length-$d$ binary vector, the communication cost is $O(d)$ for each user, which is expensive when $d$ is large.

**Random sampling.** The method of RAPPOR above deals with the assumption that each user holds exactly one item.

When each user possesses a set of values (assume that the size of the set is fixed to be $l$) and the collector wants to estimate the frequency of each item, one naive method which changes RAPPOR slightly is as follows. Each user generates a length-$d$ bit vector $B'$, similar to that in RAPPOR, with exactly $l$ ones in $B'$. Then the two probability $p$ and $q$ in Equation (1) changes into $p = \frac{\epsilon/(2l)}{1+\epsilon/(2l)}$ and $q = \frac{1}{1+\epsilon/(2l)}$. The aggregation step is the same as that in RAPPOR.

Qin et al. [34] claim that the naive method introduce high noise in the results, and they propose the method of random sampling. Instead of reporting all the items, each user samples one item from the set randomly and applies RAPPOR on that item. To solve the bias caused by the random sampling, the collector multiplies the frequency by $l$ during the stage of aggregation. The method of random sampling narrows down the error in the results significantly.
4 Problem Definition

This paper focuses on mining frequent sequential patterns among a set of sequences. The system contains \( n \) users and one collector. The collector collects sequential data from the users with local differential privacy and estimates the frequency of sequential patterns.

Formally, the sequential data of user \( u_i \) is a list of items \( s_i = \langle v_{i1}, v_{i2}, ..., v_{il} \rangle \). \( v_{il} \) are the items possessed by the user and we assume that each item only appears once in a sequence. For simplicity, we assume that the number of items in the sequential data of each user is fixed as \( l \). In real applications, if the user has less than \( l \) items, he pads his sequence with dummy items (\( l \) dummy items are needed), which will be ignored by the collector. If the user has more than \( l \) items, he truncates his sequence to the length of \( l \). An example of the padding and truncation of sequences is shown in Figure 1. The loss of information may introduce bias to the results, so \( l \) should be carefully chosen such that most of the users have less than \( l \) items.

We denote the domain of the items (including the dummy items) as \( X \). We assume that the number of real items in \( X \) is \( d \), then the size of \( X \) is \( d + l \).

The collector focuses on sequential patterns of the sequences, which is defined as follows.

**Definition 2.** (Sequential Pattern). A sequence \( sp = \langle v_{i1}, v_{i2}, ..., v_{in} \rangle \) is a sequential pattern of sequence \( s = \langle v_{1}, v_{2}, ..., v_{l} \rangle \), if \( sp \) can be derived from \( s \) by deleting some or no elements without changing the order of the remaining elements.

The length of a sequential pattern is defined as the number of elements in it. We assume that the collector focuses on sequential patterns of fixed length and the length is denoted as \( m \), i.e., there are \( m \) items in the sequential pattern, and we call it length-\( m \) sequential pattern.

Let \( \Phi_i \) be the set of all length-\( m \) sequential patterns in the sequence \( s_i \). Let \( f_{sp} \) be the frequency of a sequential pattern \( sp \). Formally,

\[
f_{sp} = \frac{|\{u_i | sp \in \Phi_i, 1 \leq i \leq n\}|}{n}
\]

The goal of the collector is to estimate the frequencies of all the length-\( m \) sequential patterns and find the top-\( k \) patterns with the highest frequencies, while satisfying LDP. We assume that the results only contain the top few patterns. The accuracy of the results should consider both the ranking of the top-\( k \) patterns and their frequencies.

5 Proposed Method

In this section, we describe the solution for mining top-\( k \) length-\( m \) sequential patterns while satisfying LDP, including the randomization step for each user and the frequency estimation for the collector.

5.1 Data Randomization

Most of the existing \( \epsilon \)-LDP mechanisms encode the data into bit vectors. However, bit vector cannot encode the similarity between \( t \) and \( \Phi_i \). We define \( u(t, \Phi_i) \) as the number of common elements between \( t \) and \( \Phi_i \), i.e., \( u(t, \Phi_i) = |t \cap \Phi_i| \).

Let \( \Omega \) be the set of all the length-\( m \) sequences combined by the items in \( X \). For a user \( u_i \) with a sequence \( s_i \), he randomly responds with a subset \( t \subseteq \Omega \) and the size of \( t \) is fixed as \( w \). Let \( \Phi_i \) be the set of all the length-\( m \) sequential patterns in \( s_i \). The probability that the subset \( t \) is chosen is as follows,

\[
Pr[K(s_i) = t] \propto \exp(\epsilon \cdot \frac{u(t, \Phi_i)}{\Delta u})
\]

where \( u(t, \Phi_i) \) is a utility function which represents the similarity between \( t \) and \( \Phi_i \). We define \( u(t, \Phi_i) \) as the number of common elements between \( t \) and \( \Phi_i \), i.e., \( u(t, \Phi_i) = |t \cap \Phi_i| \). \( \Delta u \) is the sensitivity of the utility function which is defined as

\[
\Delta u = \max_{i,j \in [1, n]} |u(t, \Phi_i) - u(t, \Phi_j)|
\]

Thus we have \( \Delta u = w \) based on our definition of \( u(t, \Phi_i) \). Let \( \lambda \) and \( \varphi \) be the size of \( \Omega \) and \( \Phi_i \), i.e., \( \lambda = |\Omega| \) and \( \varphi = |\Phi| \). We give the detailed process of the data randomization in Algorithm 1.

In the algorithm, given a user’s sequence \( s \) and the domain of items \( X \), \( \Omega \) denotes the domain of all possible length-\( m \) sequences (line 2) and \( \Phi \) denotes the set of all length-\( m \) sequential patterns in \( s \) (line 3). \( B \) is computed as the normalizer (line 6). Specifically, we have

\[
B = \sum_{t \in \Omega} \exp(\epsilon \cdot \frac{|t \cap \Phi|}{w}) = \sum_{j=0}^{w} \binom{w}{j} \left( \frac{\lambda - \varphi}{w-j} \right) \exp(\frac{\epsilon \cdot j}{w})
\]

Note that the value of \( B \) is constant as we assume that the input sequences have the same length. The probability that the user samples a set \( t \) of sequences from \( \Omega \) is

\[
Pr[K(s_i) = t] = \frac{\exp(\epsilon \cdot \frac{u(t, \Phi)}{w})}{B}
\]

The random variable \( r \) (line 7) is used to determine the value of \( int = \lfloor t \rfloor \), i.e., the size of the intersection of \( t \) and \( \Phi \), which is computed in line 11-12. At last, the set \( t \) is sampled with \( int \) sequences coming from \( \Phi \) and \( w-int \) sequences coming from \( \Omega \) (line 14-15). The function \( sample(Y, n) \) samples \( n \) items from \( Y \) without
Algorithm 1 Randomization Step of LDPSPM

**Input:** $s = \langle v_1, v_2, ..., v_l \rangle$: sequential data, $X$: domain of items, $w$: size of the output set.

**Output:** $t$: a set of $w$ length-$m$ sequences.

1. $t = \emptyset$
2. $\Omega = \{< v_1, v_2, ..., v_m > | v_1, v_2, ..., v_m \in X \}$
3. $\Phi = \{ \text{all length-$m$ sequential patterns in } s \}$
4. $\lambda = |\Omega|$
5. $\varphi = |\Phi|$
6. $B = \sum_{j=0}^{w} \left( \left( \frac{\varphi}{\lambda} \right)^j \exp\left( \frac{-t_j}{w} \right) \right)$
7. $r = \text{uniform}(0, 1)$
8. $\text{int} = 0$
9. $\text{prob} = \left( \frac{\lambda - \varphi}{w} \right) / B$
10. while $\text{prob} < r$ do
11. $\text{int} = \text{int} + 1$
12. $\text{prob} = \text{prob} + \left( \frac{\varphi}{\text{int}} \right) \left( \frac{\lambda - \varphi}{w - \text{int}} \right) \exp\left( \frac{\text{int}}{w} \right) / B$
13. end while
14. $t = t \cup \text{sample}(\Phi, \text{int})$
15. $t = t \cup \text{sample}(\Omega, w - \text{int})$
16. return $t$

replacement. The sample function can be done in $O(w^2)$. As we find in the experiments, the optimal $w$ is 1, and thus the randomization step for each user can be finished in $O(1)$ time. Each user responds with a set of $w$ length-$m$ sequences. An item costs $d$ bits during the communication, so the communication cost for each user to transfer $w$ length-$m$ sequences is $O(wmd)$.

Theorem 2. The randomization step of LDPSPM in Algorithm [7] satisfies $\epsilon$-LDP.

**Proof.** For any two possible sequences $s_1$ and $s_2$, and any output $t$, we have
\[
\frac{\Pr[K(s_1) = t]}{\Pr[K(s_2) = t]} = \frac{\exp(\epsilon \cdot u(t, \Phi_1)/w)/B}{\exp(\epsilon \cdot u(t, \Phi_2)/w)/B} = \exp(\epsilon \cdot u(t, \Phi_1) - u(t, \Phi_2)) \leq \exp(\epsilon \cdot \Delta u/w) = \exp(\epsilon)
\]
where $\Phi_s$ denotes the set of all the length-$m$ sequential patterns in $s$. Specifically, we have
\[
P = \sum_{j=1}^{w} \left( \frac{\varphi - 1}{j-1} \right) \left( \frac{\lambda - \varphi}{w-j} \right) \exp\left( \frac{\epsilon \cdot j}{w} \right) / B \tag{6}
\]
\[
Q = \sum_{j=0}^{w-1} \left( \frac{\varphi}{j} \right) \left( \frac{\lambda - \varphi - 1}{w-j-1} \right) \exp\left( \frac{\epsilon \cdot j}{w} \right) / B \tag{7}
\]
The values of $P$ and $Q$ remain the same for any $s$ and $s_p$, as we assume that the length of the users’ sequences are the same and any element only appears once in a sequence.

Let $c_{s_p}$ be the number of times the sequential pattern $s_p$ occurs in the responses $\{t_1, t_2, ..., t_n\}$. Specifically, we have
\[
c_{s_p} = \sum_{i=1}^{n} 1_{t_i}(s_p) \tag{8}
\]
The frequency of $s_p$ existing as a sequential pattern in the original sequences $\{s_1, s_2, ..., s_n\}$ can be estimated as:
\[
\hat{f}_{s_p} = \frac{c_{s_p} - nQ}{n(P - Q)} \tag{9}
\]

Theorem 3. $\hat{f}_{s_p}$ is an unbiased estimation of $f_{s_p}$ for any sequential pattern $s_p$, i.e., $\forall s_p, \mathbb{E}[\hat{f}_{s_p}] = f_{s_p}$, where $f_{s_p}$ is the true frequency of $s_p$ in the original sequences $\{s_1, s_2, ..., s_n\}$.

**Proof.**
\[
\mathbb{E}[\hat{f}_{s_p}] = \mathbb{E}
\frac{c_{s_p} - nQ}{n(P - Q)} = \mathbb{E}[c_{s_p}] - nQ = n f_{s_p} P + n(1 - f_{s_p}) Q - nQ = \frac{n (f_{s_p} P + Q - f_{s_p} Q - Q)}{n(P - Q)} = f_{s_p}
\]

After getting the estimated frequencies of all the sequential patterns, the collector sorts all the patterns and finds the top-$k$ sequential patterns. The frequency estimation component of LDPSPM for the collector is shown as Algorithm 2.

5.2 Frequency Estimation

During the process of data randomization, a user $u_i$ with sequence $s_i$ sends a randomized set of sequences $t_i$ to the collector. The collector aims to find the top-$k$ length-$m$ sequential patterns of the original sequences based on the responses $\{t_1, t_2, ..., t_n\}$ collected from the users.

Consider a length-$m$ sequence $s_p \in \Omega$, a user’s sequence $s$ and the random response $t$, we define two probability values $P$ and $Q$ as follows,
\[
\Pr[s_p \in t \mid s_p \in \Phi_s] = P,
\Pr[s_p \in t \mid s_p \notin \Phi_s] = Q.
\]

5.3 Utility Analysis

5.3.1 Error Bound

We use mean squared error to measure the error of the estimated frequencies of sequential patterns. For a sequential pattern $s_p$ with frequency $f_{s_p}$, the estimated frequency is denoted as $\hat{f}_{s_p}$. The mean squared error of the estimated frequency is measured as $\mathbb{E}[|f_{s_p} - \hat{f}_{s_p}|^2]$. 
Algorithm 2 Frequency Estimation Step of LDPSPM

\textbf{Input:} \{t_1, t_2, ..., t_n\}: responses from n users.

\textbf{Output:} \(F_{topk}\): the top-k sequential patterns along with their frequencies.

1: \(C = \{c_{sp_1} = 0, c_{sp_2} = 0, ..., c_{sp_k} = 0\}\)
2: \(F = \{f_{sp_1} = 0.0, f_{sp_2} = 0.0, ..., f_{sp_k} = 0.0\}\)
3: \(for \ t \in T\ do\)
4: \(for \ sp \in t \ do\)
5: \(c_{sp} = c_{sp} + 1\)
6: \(end\ for\)
7: \(end\ for\)
8: \(for \ sp \in \Omega\ do\)
9: \(f_{sp} = \frac{c_{sp} - nQ}{n(P - Q)}\)
10: \(end\ for\)
11: Sort \(F\) and get the top-k frequencies \(F_{topk}\)
12: \(return \ F_{topk}\)

Theorem 4. For a sequential pattern \(sp\), the mean squared error of the estimated frequency \(\hat{f}_{sp}\) is:

\[ \text{E}[||f_{sp} - \hat{f}_{sp}||^2] = \frac{f_{sp}P(1 - P) + (1 - f_{sp})Q(1 - Q)}{n(P - Q)^2} \]  \hspace{1cm} (10)

Proof. The variable \(\hat{f}_{sp}\) in Equation (9) is a linear transformation of \(c_{sp}\). According to Equation (8), the variable \(c_{sp}\) is the summation of \(n\) independent Bernoulli random variables. Specifically, \(nf_{sp}\) (resp. \((1 - f_{sp})n\)) of them are drawn from the Bernoulli distribution with parameter \(P\) (resp. \(Q\)). \(f_{sp}\) is an unbiased estimation of \(f_{sp}\) as shown in Theorem 2, thus we have

\[ \text{E}[||f_{sp} - \hat{f}_{sp}||^2] = \text{Var}[\hat{f}_{sp}] = \frac{\sum_{i=1}^{n} \mathbb{1}_{t_i}(sp) - nQ}{n(P - Q)} \]
\[ = \frac{\sum_{i=1}^{n} \text{Var}[\mathbb{1}_{t_i}(sp)]}{n^2(P - Q)^2} \]
\[ = \frac{f_{sp}P(1 - P) + (1 - f_{sp})Q(1 - Q)}{n(P - Q)^2} \]

5.3.2 Choosing \(w\)

We assume that each user responds with a set of \(w\) length-\(m\) sequential patterns in Algorithm 4. The value of \(w\) should be determined to minimize the error of the estimation. We aim to minimize the sum of the mean squared errors of the estimated frequency of all the sequential patterns, \(i.e.,\)

\[ \sum_{sp} \text{E}[||f_{sp} - \hat{f}_{sp}||^2] = \frac{\varphi P(1 - P) + (\lambda - \varphi)Q(1 - Q)}{n(P - Q)^2} \]  \hspace{1cm} (11)

We numerically compute the sum of errors in Equation (11) for every \(w \in [1, \lambda]\) and choose the value of \(w\) which minimizes the sum.

6 Prune the Domain

In this section, we propose an improved mechanism, called P-LDPSPM, which further improves the performance of LDPSPM.

As we assumed in Section 4, the collector is interested in only a few top-\(k\) sequential patterns. When the item domain is large, massive noises will be introduced in responses, due to those items which are not contained in the top-\(k\) sequential patterns, and thus affect the accuracy of the estimated frequencies. As the result shown in Equation (10), the mean squared error of the estimated frequency is proportional to the size of \(\Omega\), and thus has an exponential relationship with the size of the item domain, which is confirmed by numerical computation. If we can narrow down the size of the domain, the accuracy of the results will be improved.

As we observe in real-life sequential datasets, if we treat the sequences as set-valued data (\(i.e.,\) ignore the order of the items in a sequence and treat them as a set), the items that contained in the top sequential patterns show higher frequencies than the others. Based on this observation, we design a step of pruning before the random response to extract several candidate items. Due to the privacy concern, this step should also meet the demand of LDP, which can be done using existing LDP solutions on set-valued data, \(e.g.,\) naive RAPPOR and random sampling we described in Section 3. We choose random sampling based on RAPPOR because of its accuracy, and we show the process in Algorithm 3.

Algorithm 3 Sampling RAPPOR

\textbf{Input:} \(s = \langle v_1, v_2, ..., v_l\rangle\): sequential data, \(\epsilon_1\): the privacy budget for pruning.

\textbf{Output:} \(B\): the output binary vector.
1: \(B = 0\), \(B' = 0\)
2: Uniformly choose an item from the sequence \(s\)
3: \(B[v] = 1\)
4: Randomize \(B'\) as follows:
\[ \Pr[B'[j] = 1] = \begin{cases} \frac{e^{c_1/2}}{1 + e^{c_1/2}}, & \text{if } B[j] = 1 \\ \frac{1}{1 + e^{c_1/2}}, & \text{if } B[j] = 0 \end{cases} \]
5: \(return \ B'\)

Theorem 5. The sampling RAPPOR algorithm satisfies \(\epsilon_1\)-LDP [34].

One solution for the whole system to achieve LDP is splitting the privacy budget for the pruning step based on the property of sequential composition in Theorem 1. Another solution is dividing the users into two groups, one group for the pruning step, and the other group for the LDPSPM algorithm. Both two groups use the full privacy budget. We choose the second solution, as it is proven [37] that the method of dividing users has a better utility than dividing the privacy budget. We divide the
users into two groups. The first group contains 20% of
the population and finishes the pruning task. The second
group contains 80% of the population and finishes the
LDPSPM task.

After collecting the bit vectors from the first group, the
collector computes the frequencies as in Equation 4, and
multiplies the results by l to get an unbiased estimation
of the frequency of each item. Then the collector picks a
candidate set of \( k_{\text{max}} \) items with the highest frequencies
(excluding the dummy items). Based on the observation on
real-life datasets, we set \( k_{\text{max}} = km \) in our work, which
can cover most of the items in the top-\( k \) sequential patterns.
It’s worth mentioning that when \( k \) exceeds some value, the candidates will cover all items and the pruning
step loses the advantages. Thus we remove the pruning
step if \( km > d \) and all users are engaged in LDPSPM. The
collector then sends the \( k_{\text{max}} \) candidate items to the sec-
dond group. For each user in the second group, he adjusts
his sequence by replacing those items which are not in the
candidate set by dummy items. At last, each user in the
second group adopts the random response in Algorithm 1
for mining frequent sequential patterns.

We summarize the mechanism of our pruning based
LDP sequential pattern mining (P-LDPSPM) as follows:

Step 1: (User) Random sampling. Each user in the
first group randomly samples one item from his se-
quence and adopts random sampling on it. Then he
sends the noisy result to the collector.

Step 2: (Collector) Prune the domain. The collector
picks a set of candidates to narrow down the do-
main based on the results collected from Step 1.

Step 3: (User) Randomization of LDPSPM. Each
user in the second group adjusts his sequence based
on the candidates and adopts random response in
Algorithm 1.

Step 4: (Collector) Frequency estimation. The
collector estimates the frequency of each length-
m sequential pattern and gets the top-\( k \) sequential
patterns.

7 Evaluation

In this section, we conduct experiments to evaluate
LDPSPM and P-LDPSPM on both synthetic and real
datasets. Specifically, we seek to answer the following
questions. First, how key parameters affect the results of
the estimation of top-\( k \) sequential patterns. Second, how
the pruning step improves the results of LDPSPM.

7.1 Datasets

Synthetic Datasets: The synthetic datasets are se-
quencies generated by the IBM Quest data gener-
ator. We vary the parameters and generate several
datasets to see the impact of different parameters.

Real Dataset: The dataset is constructed by sequences
of page views on msnbc.com of different users for the
entire day of September, 28, 1999. Each sequence
recorded the categories of pages that each user requested. We modify the original dataset to make
sure that each category appears at most once in each se-
quence. The modified dataset contains 388,434 sequences.
Each sequence is truncated or padded to a fixed length of
9.

7.2 Parameters

There are several key parameters that may affect the ef-
ectiveness of the algorithms.

Number of users (\( n \)). The \( \epsilon \)-LDP mechanisms pro-
posed in this paper are essentially based on random
response and large noises are contained in the results.
A large population can effectively remove the bias in-
troduced by the noise.

Privacy budget (\( \epsilon \)). The privacy budget determines
the amount of noise to be added, and thus affect the ac-
curacy of the estimation of the collector.

Number of top sequential patterns (\( k \)). We expect
that the results will be better when \( k \) is small because
the top sequential patterns have higher frequencies
and thus can resist the noise to some extent.

In the experiments, we evaluate the impact of different
parameters on the effectiveness of our algorithms. The
optimized value of \( w \) is always 1 when we change the
parameters in our experiments, thus we set \( w = 1 \) by
default in all the next experiments.

7.3 Metrics

The results of the top-\( k \) sequential pattern mining con-
tain two aspects, i.e., the rank of the top-\( k \) sequential
patterns and the corresponding frequencies. Thus the
metrics should cover both the two aspects.

Define \( s_{pi} \) as the \( i \)-th most frequent length-
m sequential pattern in the original dataset. We denote the
ground truth of the top-\( k \) length-
m sequential patterns as \( x_t = \{ s_{p1}, s_{p2}, ..., s_{pk} \} \). There are two metrics we use:

1) Discounted cumulative gain (DCG). The DCG mea-
asures the quality of the estimated rank of the sequen-
tial patterns. Let \( \text{rel}_{sp_i} \) be the relevance of a sequen-
tial pattern \( sp_i \), which is defined as:

\[
\text{rel}_{sp_i} = \begin{cases} 
\log_2(k - \hat{r}_{sp_i}), & \text{if } k - \hat{r}_{sp_i} > 0 \\
0, & \text{if } k - \hat{r}_{sp_i} \leq 0
\end{cases}
\]

where \( \hat{r}_{sp_i} = |\text{rank}_{\text{actual}}(sp_i) - \text{rank}_{\text{estimated}}(sp_i)| \) is
the relative error of the rank of \( sp_i \).
First, we evaluate the impact of the number of users on the effectiveness of our mechanisms. We conduct experiments on two synthetic datasets. The numbers of users in the two datasets are 100,000 and 500,000, respectively. There are 60 real items for both datasets. Each sequence is truncated or padded to a fixed length of 13, thus there are 13 dummy items. We set $m=2$ and $k=20$ in the experiments, and the privacy budget is set with $\epsilon=3$.

Figure 2 shows several representative results. The results contain the true and the estimated frequencies of the true top-20 sequential patterns. The green bars show their actual frequencies, and the red candlestick bars show the estimated frequencies. Specially, if a sequential pattern with actual rank in top-20 is missed by the algorithm (i.e., not included in the estimated top-20 sequential patterns), we set the candlestick bar to 0, even though the estimated frequency is not 0. In Figure 2(a) when $n=100,000$, LDPSPM fails to capture many top-$k$ sequential patterns, and the estimated frequencies are far away from the actual ones. When we increase the number of users to $n=500,000$, as shown in Figure 2(c), the results are much better. The accuracy of the estimated frequencies gets improved. The number of missed sequential patterns decreases, especially for those patterns with high ranks (e.g., ranks higher than 10).

Meanwhile, Figure 2(b) and Figure 2(d) show the results of the improved algorithm P-LDPSPM with $n=100,000$ and $n=500,000$ respectively. Compared with the results in Figure 2(a) and Figure 2(c) with the pruning step, the number of missed sequential patterns decreases, and the estimated frequencies are closer to the actual ones.

The results in Figure 2 reveal an intrinsic challenge of LDP sequential pattern mining that the accuracy of the

$$DCG_k = \frac{\sum_{i=2}^{k} rel_{sp_i} \cdot \log_2(i)}{\log_2(k)}$$  \hspace{1cm} (12)$$

The DCG of the estimated ranked list of sequential patterns is computed as follows,

$$NDCG_k = \frac{DCG_k}{IDCG_k}$$  \hspace{1cm} (13)$$

The value of NDCG is always between 0 and 1, and we can compare the results of the top-$k$ sequential pattern mining across different $k$.

2) Mean relative error (MRE). We measure the accuracy of the estimated frequencies with the mean relative error between the actual frequency and the estimated frequency. MRE is computed as follows:

$$MRE_k = \frac{1}{k} \sum_{sp \in S_k} \left| \frac{f_{sp} - \hat{f}_{sp}}{f_{sp}} \right|$$  \hspace{1cm} (14)$$

7.4 Results

7.4.1 Impact of $n$

First, we evaluate the impact of the number of users on the effectiveness of our mechanisms. We conduct experiments on two synthetic datasets. The numbers of users in the two datasets are 100,000 and 500,000, respectively.
estimation will be poor if the population of users engaged in the algorithm is insufficient. The LDP algorithms are essentially based on random response that each user responds with a set of sequential patterns. The precision of the results will be affected by the size of the domain of the sequential patterns. The responses are more scattered when the size of the domain is larger, which leads to poor performance of the estimation of the frequencies. This is serious when the number of items and the length of targeted sequential patterns increase. The number of all real and dummy items is $d + l$ and the length of the target sequential patterns is $m$, then the number of all length-$m$ patterns is $A_{m+l}^{d+l}$, which increases exponentially with the increase of $d$, $l$ and $m$.

We conduct experiments on synthetic datasets to explore the correlation between the quality of the results and the length of the target sequential patterns, the number of items and the number of users. We generate datasets with different numbers of items and users. Each sequence is truncated or padded to a fixed length $l = 8$. The number of dummy items is $l$, as well. We fix $\epsilon = 3$ and $k = 20$ and explore the performance of the algorithm when $m$ is set with 2, 3 and 4. The results are shown in Table 1 where each experiment is conducted 10 times to get an average value. Under the setting in Table 1, the P-LDPSPM algorithm degrades into LDPSPM since the pruning step is omitted when $km > d$ as described in Section 6 so the results in Table 1 are conducted with LDPSPM.

In Table 1 we can see that the results are much better when $m$ is smaller. The performance of the algorithm drops dramatically when $m$ increases. Besides, the results are better when the number of items is smaller. Increasing the population of users improves the quality of the results. For example, considering the number of items as 10 and $m = 2$, the value of MRE remains 0.01 when the number of users increases from $1 \times 10^5$ to $5 \times 10^7$. This is caused by the padding and truncation of the original sequences as described in Section 4, which leads to deviations from the original sequences. Furthermore, we find that the quality of the results is closely related with the number of all length-$m$ sequential patterns, i.e., $A_{m+l}^{d+l}$. The number of users needed to be engaged in the algorithm to reach the stable metric is in direct proportion to the number of all length-$m$ patterns. For example, the result under the parameters $d = 10$, $l = 8$, $m = 2$, $n = 1 \times 10^5$ has a similar quality as the result under the parameters $d = 20$, $l = 8$, $m = 3$, $n = 5 \times 10^7$. The limitation of the algorithm comes out that many more users are needed to maintain the quality of the results when the number of items and the length of target patterns increase.

### 7.4.2 Impact of $\epsilon$

In this experiment, we evaluate the impact of the privacy budget with the synthetic and the MSN dataset. The synthetic dataset contains 500,000 users and the length of each sequence is 13. There are 60 real items and 13 dummy items. Due to the difference in the number of items of the two datasets, we set $k = 20$ for the synthetic dataset and $k = 5$ for the MSN dataset. As we describe in Section 7.4.1, many more users are needed to maintain the quality of the results when the number of items and the length of patterns increase. Due to the limited number of the users in the real-world MSN dataset, we set $m$ with a relatively small value (i.e., $m = 2$) in the following experiments. Figure 3 shows the results for the two datasets along with the change of $\epsilon$. It is clear that the results get better (with higher NDCG and lower MRE) when $\epsilon$ increases. P-LDPSPM gets higher NDCG and lower MRE than LDPSPM on both two datasets. More specifically, P-LDPSPM gets more improvement than LDPSPM when $\epsilon$ is relatively small. The reason is that when $\epsilon$ increases, fewer noises are contained in the responses, and thus the

<table>
<thead>
<tr>
<th>#real items, #dummies</th>
<th>#users</th>
<th>$m=2$</th>
<th>$m=3$</th>
<th>$m=4$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NDCG</td>
<td>MRE</td>
<td>NDCG</td>
<td>MRE</td>
</tr>
<tr>
<td>$d = 10$, $l = 8$</td>
<td>$n = 1 \times 10^5$</td>
<td>0.98</td>
<td>0.03</td>
<td>0.76</td>
</tr>
<tr>
<td></td>
<td>$n = 5 \times 10^5$</td>
<td>0.99</td>
<td>0.02</td>
<td>0.92</td>
</tr>
<tr>
<td></td>
<td>$n = 1 \times 10^6$</td>
<td>0.99</td>
<td>0.01</td>
<td>0.96</td>
</tr>
<tr>
<td></td>
<td>$n = 5 \times 10^6$</td>
<td>0.99</td>
<td>0.01</td>
<td>0.97</td>
</tr>
<tr>
<td></td>
<td>$n = 1 \times 10^7$</td>
<td>1</td>
<td>0.01</td>
<td>0.98</td>
</tr>
<tr>
<td></td>
<td>$n = 5 \times 10^7$</td>
<td>1</td>
<td>0.01</td>
<td>0.99</td>
</tr>
<tr>
<td>$d = 20$, $l = 8$</td>
<td>$n = 1 \times 10^5$</td>
<td>0.97</td>
<td>0.05</td>
<td>0.24</td>
</tr>
<tr>
<td></td>
<td>$n = 5 \times 10^5$</td>
<td>0.98</td>
<td>0.02</td>
<td>0.57</td>
</tr>
<tr>
<td></td>
<td>$n = 1 \times 10^6$</td>
<td>0.99</td>
<td>0.02</td>
<td>0.74</td>
</tr>
<tr>
<td></td>
<td>$n = 5 \times 10^6$</td>
<td>0.99</td>
<td>0.01</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>$n = 1 \times 10^7$</td>
<td>1</td>
<td>0.01</td>
<td>0.96</td>
</tr>
<tr>
<td></td>
<td>$n = 5 \times 10^7$</td>
<td>1</td>
<td>0.01</td>
<td>0.99</td>
</tr>
</tbody>
</table>
When \( k \) is small, and the results of two algorithms get closer when \( k \) increases. This is because when \( k \) increases, more true top-\( k \) sequential patterns are captured in the estimated top-\( k \) patterns, and the influence of the wrong ranks of the top sequential patterns on the NDCG decreases. The MRE increases when \( k \) increases, i.e., the average accuracy of the frequencies decreases when more sequential patterns are identified. It is obvious that the sequential patterns with higher frequencies appear more in the responses of the users, and thus the estimated frequencies of them are closer to the true frequencies. Also, the advantage of P-LDPSPM compared to LDPSPM is much more significant when \( k \) is small, and the results of two algorithms get closer when \( k \) increases. When \( k \) increases, the ratio of the candidates identified in the pruning step to the whole item domain gets higher, and the utility gain due to the pruned domain is offset by the utility loss caused by the split of user groups. At some values of \( k \) (e.g., when \( k > 30 \) in the synthetic dataset and \( k > 8 \) in the MSN dataset), the candidates cover the whole item domain, as we explained in Section 6, the pruning step is removed and the results become the same for LDPSPM and P-LDPSPM.

8 Conclusions

In this paper, we study the problem of mining frequent sequential patterns under local differential privacy. We propose an efficient and effective mechanism called LDP-SPM. Each user randomly responds with a set of sequential patterns of various lengths and reduce the dependency of nonsignificant items. Both theoretical analysis and extensive experiments show the effectiveness and efficiency of our methods.

The limitation of this work lies in that the algorithms are only suitable for sequential patterns of fixed lengths. Besides, large population of users are needed to maintain the quality of the results when the number of items and the length of the target patterns are big. Future works can focus on mechanisms that support frequency estimation of sequential patterns of various lengths and reduce the dependence on the number of users.
Acknowledgments

The research was supported by the National Key R&D Program of China 2018YFA0803400, 2018YFB2100300, National Natural Science Foundation of China under Grant No.61972369, No.61572453, No.61520106007, No.61572454, and the Fundamental Research Funds for the Central Universities, No.WK2150110009.

References


Biography

Huihua Xia received the B.S. degree in computer science from University of Science and Technology of China in 2014. He is currently working towards the Ph.D. degree at the Department of Computer Science and Technology, University of Science and Technology of China. His current research interests include data privacy and information security.

Wenchao Huang received the B.S. and Ph.D degrees in computer science from University of Science and Technology of China in 2006 and 2011, respectively. He is an associate professor in School of Computer Science and Technology, University of Science and Technology of China. His current research interests include information security, formal methods and mobile computing.

Yan Xiong received the B.S., M.S., and Ph.D degrees from University of Science and Technology of China in 1983, 1986 and 1990 respectively. He is a professor in School of Computer Science and Technology, University of Science and Technology of China. His main research interests include distributed processing, mobile computing, computer network and information security.

Fuyou Miao received his Ph.D of computer science from University of Science and Technology of China in 2003. He is an associate professor in the School of Computer Science and Technology, University of Science and Technology of China. His research interests include applied cryptography, trusted computing and mobile computing.