IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 31, 2019, accepted November 26, 2019, date of publication December 5, 2019,
date of current version December 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2957762

Differentially Private Top-*k* Frequent Columns Publication for High-Dimensional Data

NING WANG[1], (Member, IEEE), ZHIGANG WANG[1], YU GU[2], (Member, IEEE), JIA XU[3], ZHIQIANG WEI[1], AND GE YU[2], (Member, IEEE)

[1]College of Information Science and Engineering, Ocean University of China, Qingdao 266000, China
[2]College of Computer Science and Engineering, Northeastern University, Shenyang 110819, China
[3]College of Computer Science and Engineering, Guangxi University, Nanning 530000, China

Corresponding author: Zhiqiang Wei (weizhiqiang@ouc.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61902365 and Grant 61902366, in part by the China Postdoctoral Science Foundation under Grant 2019M652473 and Grant 2019M652474, and in part by the Fundamental Research Funds for the Central Universities under Grant N171606005.

ABSTRACT Differential privacy (DP) is a promising scheme for releasing the results of statistical queries on sensitive data. This paper focuses on top-*k* frequent columns publication on sensitive data, with high result utility under differential privacy. Existing works directly select frequent columns from all columns (called one-phase scheme), which is far from ideal due to the large privacy consumption or misjudgments for columns with frequencies close to the frequency of the *k*th frequent column (called near-*k*-fluctuation-column). This paper presents a new solution Two Phase Selection (TPS) to carefully choose frequent columns in two phases. The main idea is to classify columns into two distinct categories based on whether it is one near-*k*-fluctuation-column or not. Frequent columns are chosen from the two categories using different techniques, which is totally different from existing solutions without classifying. Furthermore, by analyzing the distribution of near-*k*-fluctuation-columns, we introduce a block-centric column-choosing method privacy-free-mechanism (PFM). By partitioning columns into blocks, PFM makes the privacy consumption proportional to the number of blocks, instead of frequent columns. Extensive experiments on real datasets show that our proposals outperform the state-of-the-art techniques for top-*k* column publication.

INDEX TERMS Differential privacy, privacy protection, top-*k*, high dimensional dataset, privacy consumption, block.

I. INTRODUCTION

With the increasing volumes of personal data collected by mobile devices and web services, privacy protection has become a hot research topic. In particular, for the high-dimensional privacy datasets, a lot of works [1]–[3] have been proposed to publish the frequency of each column in the dataset, i.e., column frequencies. However, in some scenario, the users or administrators are more interested in the top-*k* frequent columns. For example, the tourists are more interested in the top-*k* popular scenic spots (columns) in a city. And the marketers are concerned about top-*k* hot search terms (columns) in the network. This paper focuses on the privacy issues that arise in publishing top-*k* frequent columns, i.e., finding top-*k* heavy hitters [4]–[6]. Top-*k* frequent columns help managers get to know the data distribution and then make smart decisions to improve the quality of service. Specifically, let D be a database, in which every record is a binary vector $\{0, 1\}^d$. Note that d may reach tens of hundreds (i.e., high-dimensional dataset) in many practical applications. The frequency of column j indicates the number of ''1''s in this column, i.e., the number of records whose value in the jth column equals to ''1''. And the top-*k* frequent columns show k columns with the largest frequencies. Note that frequent items mining, as a vital fundamental step in the classic and widely studied top-*k* frequent itemsets mining problem [7]–[10], is consistent with the frequent columns mining, if every item is regarded as one column in the high dimensional dataset.

Take the purchased information generated from web e-commerce platforms (e.g., Amazon [11], Ebay [12]) as an example. In the database D_1, every column indicates one item. User i is associated with the ith record in D_1 and element $D(i, j)$ in the jth column is set to 1 if user i purchased item j.

The associate editor coordinating the review of this manuscript and approving it for publication was Cristina Rottondi.

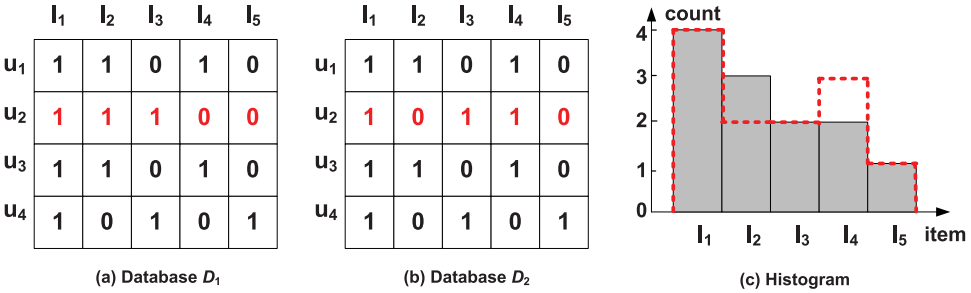


FIGURE 1. An Example of top-*k* frequent columns publication.

As shown in Figure 1(a), every row lists the items purchased by one user. For example, u_1 has purchased items $\{I_1, I_2, I_4\}$. And the grey histogram in Figure 1(c) shows the purchased frequencies of items. For instance, since u_2 and u_4 purchased item I_3, the frequency of I_3 is 2. Obviously, the top-2 frequent items in the database D_1 are items I_1 and I_2.

However, the high-dimensional records involve many personal information and hence directly publishing their top-k statistic values breaks users' privacy. Specifically, in Figure 1, if the purchased item list of u_2 changes from $\{I_1, I_2, I_3\}$ to $\{I_1, I_3, I_4\}$, the top-2 frequent items are $\{I_1, I_4\}$ instead of $\{I_1, I_2\}$. Such deviation may inspire adversaries to infer u_2's purchased information, if adversaries hold background knowledge on all purchased list except u_2's.

Recently differential privacy (DP) [13]–[15] as a well-known privacy protection technique, has attracted a lot of attentions, since it can provide rigorous theoretical privacy guarantee against adversaries with arbitrary background knowledge. This paper thereby aims to publish top-k frequent columns in ε-differential privacy, where ε is a parameter to control the degree of privacy protection. Sensitivity, as another key parameter in DP, indicates the maximum impact a single record can have on all column frequencies. Typically, to guarantee DP, noises proportional to the sensitivity have to be injected into published results. Thus, the great challenge in our problem is that the high dimensional feature in real datasets usually leads to a large sensitivity, and finally makes published results useless.

There are two lines of works for publishing the top-k frequent columns by clarifying the existing methods, which are adopted as the fundamental step (frequent items mining) in the top-k frequent itemsets mining. The first line is designed based on the publication of column frequencies [1]–[3]. Specifically, it publishes the column frequencies, sorts columns in descending order of their noisy frequencies, and then outputs top-k ones. However, the noises in frequencies make the columns with frequencies close to the one of the kth frequent column (called near-k-fluctuation-column) be misjudged, leading to inaccurate results. The other line [16], [17] invokes exponential mechanism to sample one frequent column with probability inversely proportional to k. This is performed k times to get published results. However, it does not work well for a large k in high-dimensional applications.

This paper solves the utility problem by proposing a new Two Phase Selection (TPS) method. Before describing TPS, we first introduce an observation that the ability of columns to tolerate noises varies with their frequency ranks. For example, Figure 2 shows the frequencies of 9 columns $\{A, B, C, D, E, F, G, H, I\}$ respectively. Circles/triangles stand for the true/noisy frequencies. Obviously, the true top-5 frequent columns are $\{A, B, C, D, E\}$. But based on the noisy frequencies, they are $\{A, B, E, F, G\}$. This is because that noises in frequencies make columns $\{C, D, F, G\}$ with frequency ranks around k change from frequent (unfrequent) to unfrequent (frequent). These columns, whose noisy frequencies are located in $[\widehat{\tau} - \delta, \widehat{\tau} + \delta]$, form **unsafe zone** as they are easily affected by noises. Here, $\widehat{\tau}$ and δ are parameters used to limit the size of the unsafe zone and will be explained in Section IV in detail. By contrast, **safe zone** consists of other columns. Since the ranks of columns in safe zone are far away from k, they can tolerate noises. TPS leverages the above observation to choose top-k results. The key idea of TPS is firstly choosing frequent columns from safe zone (the first phase) and then getting the remaining results from unsafe zone (the second phase), so that we can employ different techniques based on the different noise-tolerance abilities of different zones.

Intuitively, TPS is a hybrid solution which is built by combing the two lines of works mentioned above, in order to handle the two-phase selection tasks respectively. Specifically, in the first phase, we use the first line method to publish only some frequent columns in safe zone, instead in the whole column domain, as a part of final results. Since frequencies of columns in safe zone are much higher than that of the kth frequent column, the impact of noises is nearly negligible and hence they are really frequent with a high probability. In the second phase, the exponential mechanism is used to choose the remaining frequent columns from unsafe zone. Usually the number of remaining frequent columns k_Δ is much less than k. Then the exponential mechanism is invoked only k_Δ times ($k_\Delta < k$). Thus, the reduced execution times contribute to improving the probability of choosing true frequent columns.

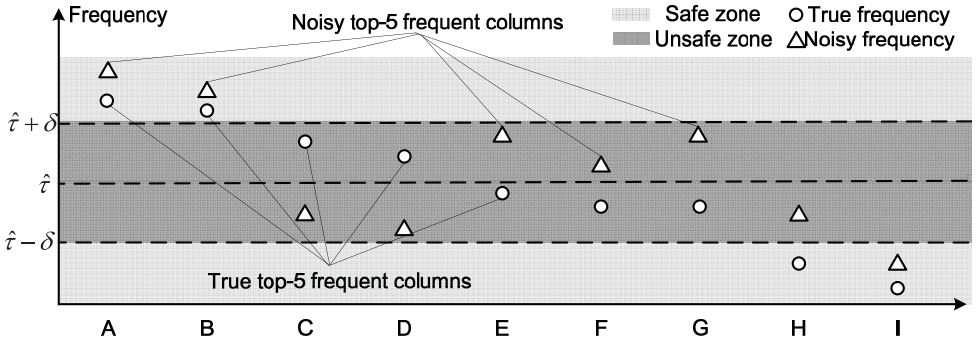


FIGURE 2. An example of the two-phase selection method.

Additionally, we propose a new block-centric column-choosing solution to further reduce the privacy consumption caused by exponential mechanism in the second phase. We find that either frequent columns or unfrequent columns usually appear consecutively in unsafe zone, since the columns in unsafe zone are sorted based on their noisy frequencies. That motivates us to split the columns into blocks so that we can apply a newly designed DP mechanism privacy-free-mechanism (PFM) to get required results in a block-centric way. In particular, PFM detects the boundary of every block using a modified sparse vector technique (SVT) [18]. The original SVT is used to release c columns with frequency above τ from a set of columns, where τ is the frequency of the cth frequent column. To guarantee DP, SVT judges whether one column is frequent or not by comparing the noisy versions of τ and the column frequency, where every found frequent column imports a part of privacy consumption. Our modification is that based on the previous column judgement results output in SVT, the noisy version of τ is updated and the noise scales in the column frequencies are dynamically adjusted to make less noises be injected to column frequencies. With PFM, some frequent columns can be chosen without privacy consumption, since there are possibly multiple frequent columns in one block. In this way, the privacy consumption of PFM is proportional to the number of blocks, which is smaller than that of frequent columns. Consequently, compared with exponential mechanism and original SVT, PFM chooses the frequent columns with lower privacy consumption.

As a short summary, the major contributions of our work are summarized as below:

- We present a novel and principled adaptive two-phase selection framework to publish the top-k frequent columns by taking advantage of the different noise-tolerance abilities of columns in different zones.
- We design a new block-centric column-choosing mechanism denoted by PFM to derive the frequent columns dynamically. And the mechanism makes the privacy consumption proportional to the number of blocks, which is smaller than the number of frequent columns. Besides, we theoretically analyze the privacy guarantee of the mechanism PFM.
- We conduct extensive experiments using real datasets to demonstrate that our method significantly outperforms the baseline approaches.

The rest of this paper is organized as follows. Section II reviews the related works. Section III introduces definitions about DP and the two existing lines for publishing top-k frequent columns under DP. Section IV proposes our two phase selection method. Section V presents privacy-free-mechanism. Section VI reports the experiment results and Section VII concludes this paper.

II. RELATED WORK

Differential privacy [2], [16], [17], [19], as a rigorous privacy protection model, has attracted a lot of attention, since it can provide theoretical privacy guarantee against adversaries with arbitrary background information. Before introducing the definition of differential privacy, we first give the definition of neighboring database in the following.

Definition 1 (Neighboring database [20]): Given two databases D_1 and D_2, if we can get D_2 from D_1 by deleting or adding one record t, i.e., $D_2 = D_1 \cup \{t\}$ or $D_1 = D_2 \cup \{t\}$, the two databases are called as a pair of neighboring databases.

Definition 2 (ε-DP [20]): Given a randomized algorithm $\mathcal{A}$, $\mathcal{A}$ satisfies ε-DP if for any two neighboring databases D_1 and D_2, and any output O of $\mathcal{A}$, there exists

$$P[\mathcal{A}(D_1) = O] \leq e^{\varepsilon} P[\mathcal{A}(D_2) = O].$$

Many representative efforts [1]–[3], [17], [19] have been devoted to publishing statistics for high-dimensional data. Existing works can be classified into three categories. We elaborate them and then distinguish our work from them as follows.

A. PUBLISHING NOISY DATABASE [21], [22]

This kind of works aims at publishing one synthetic database in order to support any possible query, including column frequencies query and top-k frequent columns query. These works [21], [22] depend on the bayesian network and context-free taxonomy tree to publish synthetic databases respectively. However, publishing synthetic databases required by

the general purpose consumes much privacy budget since a lot of information is involved. That is the reason that recent techniques [1]–[3] pay attention to publishing noisy column frequencies only, instead of the whole database. We introduce them in the following.

B. PUBLISHING NOISY COLUMN FREQUENCIES [1]–[3]

Many existing works focus on count publication, which is the key component of publishing top-k frequent columns. To decrease the sensitivity of publishing noisy frequencies of columns, FPA [3] adds noises into Fourier coefficients and then noisy frequencies of columns can be derived. Furthermore, note that the mean for some column frequencies has a lower sensitivity than the original frequency values. GS [2] thereby splits columns into several groups and then publishes the noisy mean of every group to estimate the frequency of each column in the corresponding group. On the other hand, since the sensitivity of publishing frequencies is the dimensionality of the database which introduces large scale of noise, to lower it, DPsense [1] limits the contribution of each record . Because DPsense is closely related with our work and has a superiority to GS, as validated in [1], we will give more details about it in Section 3.3.

C. PUBLISHING FREQUENT COLUMNS PATTERNS [16], [17], [19], [23]

There are a lot of works [16], [17], [19], [23] focusing on the top-k frequent itemsets (column patterns) mining. Since our problem top-k frequent columns mining is just one fundamental step of frequent itemsets mining, we just pay attention to the techniques in existing works [16], [17], [19], [23] designed to optimize this fundamental step. Zeng *et al.* [19] firstly publish the noisy frequencies of all columns and then select frequent columns based on the noisy frequencies. But the noises usually make the columns with frequencies close to the frequency of the kth frequent column fluctuate, which generates inaccurate results. We are also aware that some works [16], [17] employ exponential mechanism to choose k columns as the final top-k frequent results under differential privacy. Specifically, exponential mechanism is invoked k times to sample frequent columns from all columns owned by the database. However, in exponential mechanism, the frequent columns are returned with the probability inversely proportional to k. As k can be large in many cases, the probability of choosing the true frequent column at each time decreases, which leads to inaccurate results. Besides, Wang *et al.* [24] propose sequential exponential mechanism to effectively mine frequent itemsets. But the mechanism cannot work well when doing frequent items mining. Note that Lee *et al.* [23] design the a variant of sparse vector technique to choose an arbitrary number of frequent columns with constant privacy consumption. However, some works [25], [26] point the variant violates ε-differential privacy, without known fix.

III. PRELIMINARY

A. LAPLACE MECHANISM AND EXPONENTIAL MECHANISM

To guarantee ε-DP, **laplace mechanism** and **exponential mechanism** are two most widely used mechanisms. And both of the two mechanisms depend on the sensitivity, which is defined in the following.

Definition 3 (Sensitivity [20]): Given one algorithm $\mathcal{A}$, the sensitivity is defined as

$$\Delta = \max_{D_1, D_2} ||\mathcal{A}(D_1) - \mathcal{A}(D_2)||_1,$$

where D_1 and D_2 are two neighboring databases, $||\mathcal{A}(D_1) - \mathcal{A}(D_2)||_1$ is the L_1 distance between $\mathcal{A}(D_1)$ and $\mathcal{A}(D_2)$.

Specifically, laplace mechanism [27] is adopted for the scenario that the output of $\mathcal{A}$ is numeric. To satisfy ε-differential privacy, it adds noises to the output of $\mathcal{A}$ directly. And the noise is sampled from the Laplace distribution with mean 0 and scale $\frac{\Delta}{\varepsilon}$, where Δ is the sensitivity of the algorithm $\mathcal{A}$.

On the other hand, if the output of $\mathcal{A}$ is categorical, exponential mechanism [14] is utilized. Given the output domain Ω of $\mathcal{A}$, the quality function $q : D \times \Omega \to R$ needs to be given by users. Specifically, for any output $\omega \in \Omega$, the value of quality function $q(D, \omega)$ reflects how accurately $\mathcal{A}$ outputs ω based on D. To guarantee ε-differential privacy, exponential mechanism samples ω from Ω with the probability proportional to $e^{\frac{\varepsilon q(\omega, D)}{2\Delta}}$, where Δ is the sensitivity of quality function.

For some complex queries, composition property [28] is used to design differentially private algorithms.

Definition 4 (Composition Property): Given one algorithm $\mathcal{A}(D)$ consisting of a sequence of sub-algorithms $\{\mathcal{A}_1(D), ..., \mathcal{A}_i(D), ..., \mathcal{A}_n(D)\}$, if $\mathcal{A}_i(D)$ satisfies ε_i-DP, $\mathcal{A}(D)$ satisfies $(\sum_i \varepsilon_i)$-DP.

B. SPARSE VECTOR TECHNIQUE

The sparse vector technique [18] is designed to publish c count queries above one given threshold τ from query set Q in ε-differential privacy. It works as follows. Firstly it calculates a noisy version of threshold $\widehat{\tau}$ with noise scale $\frac{1}{0.25\varepsilon}$.[1] Then, given one count query $q \in Q$, it compares $\widehat{\tau}$ with one noisy version of the query result $\widehat{q}(D)$ which has noise scale $\frac{1}{0.75\varepsilon/c}$. If $\widehat{q}(D)$ is above $\widehat{\tau}$, the index of this query is output. Otherwise, no answer is given. The comparisons are halted when all required c indexes of count queries are derived or all queries in Q have been checked. Dwork et al. [18] have proved this process satisfies ε-DP. The advantage of this technique is that the privacy budget consumed is proportional to c, i.e., the number of queries above $\widehat{\tau}$. In order to find top-k frequent columns, we can derive the frequency of the kth as the threshold τ based on the database, and invoke the sparse vector technique.

C. EXISTING METHODS

This paper aims to publish top-k frequent columns for a high-dimensional dataset. There are two main lines summarized

[1] Note that we assume the sensitivity of one count query is 1.

from the works about frequent itemsets mining [16], [17], [19] to solve this problem, but both of them do not work well in our context. Since our solution builds on top of them, now we introduce their representative algorithms, i.e., freFirst [19] and exponential mechanism [16], [17], respectively.

1) freFirst, GETTING k FREQUENT COLUMNS BASED ON THEIR NOISY FREQUENCIES

freFirst publishes the noisy frequencies of all columns firstly and then selects the top-k frequent columns based on the noisy frequencies. Since the technique for publishing noisy frequencies in [19] does not work well as the state of the art one [1], we use DPsense to publish noisy column frequencies, instead of the method used in [19]. In the following, we introduce DPsense in detail. Consider that the sensitivity for the column frequencies is the maximum sum of the values from all columns in any record in the database, i.e., $\max_i(\sum_{j=1}^{d} D(i,j))$, where $D(i,j)$ denotes the value of the jth column in the ith record. Thus, to improve the accuracy of published noisy frequencies, freFirst limits the maximum sum as θ by normalizing $D(i,j)$ as $D(i,j)\frac{\theta}{\sum_{j=1}^{d} D(i,j)}$. In this paper, θ is called as truncating length. Specifically, DPsense firstly computes a truncating length θ in ε_2-differential privacy by taking both of the error incurred by normalizing and the Laplace noise error into consider. Following that, a truncated database D_θ is derived by:

$$D_\theta(i,j) = \begin{cases} D(i,j), & \text{if } \sum_{j=1}^{d} D(i,j) \le \theta, \\ D(i,j)\dfrac{\theta}{\sum_{j=1}^{d} D(i,j)}, & \text{otherwise.} \end{cases}$$

Following that, based on the truncated records in D_θ, DPsense publishes the noisy frequencies with Lap($\frac{\theta}{\varepsilon_1}$) of all columns in ε_1-differential privacy. Based on the composition property of DP, DPsense satisfies $(\varepsilon_1 + \varepsilon_2)$-differential privacy. Finally, top-k frequent columns are published based on the noisy frequencies. However, the misjudgments for ''near-k-fluctuation-column''s phenomenon makes published results inaccurate.

2) EXPONENTIAL MECHANISM (EM), CHOOSING k FREQUENT COLUMNS DIRECTLY

The EM can be used k times to sample k frequent columns. To guarantee ε-differential privacy, column i is sampled with probability proportional to $e^{\frac{\varepsilon f(D,i)}{k}}$, where $f(D,i)$ denotes the frequency of the column i on database D. Obviously, the probability of the frequent column sampled is inversely proportional to k. A larger k reduces the probability of a frequent column being sampled.

IV. TWO PHASE SELECTION METHOD

This section firstly presents a two phase selection method (TPS) for publishing top-k frequent columns. After that, we analyze its privacy guarantee, and discuss the setting of some parameters used in TPS.

TPS includes two phases. Firstly, it uses a part of privacy budget to publish column frequencies and sorts the columns in descending order of noisy frequencies. Following that, some columns with frequency ranks far from k are chosen as frequent columns with high probability. The remaining results are chosen from the columns with frequency ranks around k in the second phase. As the state-of-the-art techniques, DPsense and EM are used in the two phases respectively. For better understanding, based on the noisy frequencies published by DPsense in the first phase, we define safe zone and unsafe zone in Definition IV. The definitions depend on two variables $\widehat{\tau}$ and δ, which are introduced in the following. By sorting the columns in the descending order of noisy frequencies, we can get the noisy frequency $\widehat{\tau}$ of the kth frequent column. And δ is one parameter which limits the size of unsafe zone and is discussed at the end of this section.

Definition 5 (Unsafe Zone, Safe Zone, Frequent Safe Zone, Unfrequent Safe Zone): Unsafe zone is the set of columns with noisy frequency below $\widehat{\tau}+\delta$ but above $\widehat{\tau}-\delta$, while safe zone contains other columns not in unsafe zone. The latter is further divided into frequent safe zone consisting of columns with noisy frequency above $\widehat{\tau}+\delta$, and unfrequent safe zone in which every column's frequency is below $\widehat{\tau}-\delta$.

Since the frequencies of columns in frequent safe zone are much bigger than $\widehat{\tau}$, these columns are inferred as frequent. On the contrary, columns in unfrequent safe zone are regarded as infrequent. On the other hand, for the columns in unsafe zone, since the frequencies of them are close to $\widehat{\tau}$, we cannot easily judge them whether frequent or not by their sorting ranks. Instead, we use a part of privacy budget to carefully choose the frequent columns from unsafe zone based on exponential mechanism.

Algorithm 1 shows the framework of TPS. Firstly, the privacy budget is split into three parts: ε_1, ε_2 and ε_3 (Line 1). DPsense uses privacy budget ε_1 to compute the truncating length l and then publishes the noisy frequencies of columns $\widehat{C}$ with noise scale $\frac{l}{\varepsilon_2}$ in ε_2-DP (Line 2). Here, $\widehat{C}$ is a vector, in which the jth element stores the jth column and its noisy frequency. And then we sort the columns based on the published noisy frequencies in descending order and store the sorted results in another vector $\widehat{C}_s$, where the jth element stores the jth frequent column e_i and its noisy frequency e_c (Line 3). And $\widehat{\tau}$ denotes the noisy frequency of the kth element in $\widehat{C}_s$ (Line 4). Afterwards we successively visit elements in $\widehat{C}_s$ (Lines 6-12). If the noisy frequency e_c is bigger than $\widehat{\tau}+\delta$, i.e., e_i is in frequent safe zone, e_i is directly added into R_1 as a frequent column (Lines 7-8). If e_c is smaller than $\widehat{\tau}-\delta$, i.e., e_i is in unfrequent safe zone, e_i is skipped. Besides, since the columns in $\widehat{C}_s$ are sorted in descending order, the columns located behind e_i in $\widehat{C}_s$ are all unfrequent and the traversal is terminated (Lines 9-10). If $e_c \in [\widehat{\tau}-\delta, \widehat{\tau}+\delta]$, e_i is in unsafe zone and appended into B (Lines 11-12). Till now, a part of frequent columns in R_1 have been derived. Next, GetExtraColumns focuses on how to choose the remaining $k - |R_1|$ frequent columns from the columns located in unsafe zone B (Line 13). In the GetExtraColumns

Algorithm 1 Two Phase Selection Method

Input : Database D, privacy budget ε, k
Output: top-k frequent columns R

1 Split ε into three parts, ε_1, ε_2 and ε_3;
2 $\widehat{C}, l \leftarrow$ DPsense(D, ε_1, ε_2);
3 Get $\widehat{C}_s$ by sorting columns in $\widehat{C}$ in descending order of noisy frequencies;
4 $\widehat{\tau} \leftarrow$ the noisy frequency of the kth element in $\widehat{C}_s$;
5 Set candidate frequent column list $B = \{\}$, $R_1 = \{\}$;
6 **for** *each element* $[e_i, e_c]$ *in* $\widehat{C}_s$ **do**
7 **if** $e_c \in [\widehat{\tau} + \delta, +\infty]$ **then**
8 $R_1 = R_1 \cup e_i$;
9 **if** $e_c \in [-\infty, \widehat{\tau} - \delta]$ **then**
10 Break;
11 **if** $e_c \in [\widehat{\tau} - \delta, \widehat{\tau} + \delta]$ **then**
12 append e_i to B;
13 $R_2 \leftarrow$ GetExtraColumns(B,D, ε_3, $k - |R_1|$);
14 $R \leftarrow R_1 \cup R_2$;
15 Return R;

algorithm, exponential mechanism is invoked $k-|R_1|$ times to choose $k-|R_1|$ frequent columns. And for any column e_i in B, the sampled probability of e_i is proportional to $e^{\frac{\varepsilon f(e_i,D)}{k-|R_1|}}$, where $f(e_i, D)$ is the frequency of column e_i. Since the algorithm of GetExtraColumns is clear, we omit the description.

Taking the scenario shown in Figure 2 into consider, Algorithm 1 works as follows: Firstly DPsense publishes the noisy frequencies of all columns in the first phase, where the noisy frequencies are labeled as triangles. After that, by comparing the noisy frequencies with $\widehat{\tau}-\delta$ and $\widehat{\tau}+\delta$, a part of frequent columns $\{A, B\}$ and the columns in the unsafe zone $\{E, G, F, C, H, D\}$ are achieved. And then exponential mechanism is invoked three times to choose the other three frequent columns from $\{E, G, F, C, H, D\}$.

In the following, Theorem 1 gives the privacy analysis of Two-Phase Selection.

Theorem 1: TPS satisfies ε-differential privacy.

Proof: TPS includes three parts. For the first part, DPsense satisfies $(\varepsilon_1 + \varepsilon_2)$-differential privacy to publish the truncating length l and noisy frequencies of columns in Line 2, which has been validated by Day et al. [1]. For the second part, it does not compromise any privacy to choose the columns in frequent safe zone and the ones in unsafe zone in Lines 3-12, since these operations are based on noisy frequencies, independent of the database. For the third part, GetExtraColumns applies exponential mechanism $k - |R_1|$ times to sample $k - |R_1|$ columns without replacement. Since the sensitivity of one column frequency is 1, intuitively the process satisfies ε_3-differential privacy. Based on Definition 4, TPS satisfies $(\varepsilon_1 + \varepsilon_2 + \varepsilon_3)$-differential privacy. ■

Choice of δ: Considering the scale of laplace noise in $\widehat{\tau}$ is $\frac{l}{\varepsilon_2}$, we use $\widehat{\tau} - \frac{l}{\varepsilon_2}$ or $\widehat{\tau} + \frac{l}{\varepsilon_2}$ to estimate the true frequency τ of the column in the kth index of $\widehat{C}_s$. Let c ($\widehat{c}$) be the true (noisy) frequency of one column. In the same way, c is estimated using $\widehat{c} - \frac{l}{\varepsilon_2}$ or $\widehat{c} + \frac{l}{\varepsilon_2}$. If the smaller value of c is larger than the bigger value of τ, i.e., $\widehat{c} - \frac{l}{\varepsilon_2} > \widehat{\tau} + \frac{l}{\varepsilon_2}$ ($\widehat{c} > \widehat{\tau} + \frac{2l}{\varepsilon_2}$), the column is frequent with a high probability. If the bigger value of c is smaller than the smaller value of τ, i.e., $\widehat{c} + \frac{l}{\varepsilon_2} < \widehat{\tau} - \frac{l}{\varepsilon_2}$ ($\widehat{c} < \widehat{\tau} - \frac{2l}{\varepsilon_2}$), this column is unfrequent in high probability. In other cases, we think the columns need to be judged further. So we set the parameter δ to $\frac{2l}{\varepsilon_2}$.

Although TPS can choose the true frequent columns with high probability in the first phase, it has to split ε_3 into $k_\Delta = k - |R_1|$ parts and uses every part to choose one frequent column from the unsafe zone in the second phase. If there are many frequent columns left in unsafe zone, i.e., k_Δ is big, the accuracy of the k_Δ frequent columns chosen is still low. In the next section, we introduce privacy-free-mechanism to optimize the method in the second phase.

V. TWO PHASE SELECTION WITH PRIVACY-FREE-MECHANISM

This section presents our solution for choosing the remaining frequent columns from the unsafe zone. Firstly, we show a phenomenon existing in the non-private scenario, which motivates us to design the privacy-free-mechanism. After that, we elaborate our algorithm and analyze its privacy guarantee.

Recall that the first phase of TPS generates B which stores the sorted columns in descending order of noisy frequencies in unsafe zone. By analyzing the true frequencies of columns in unsafe zone, we find that frequent/unfrequent columns are prone to be continuous. Let ρ denote the threshold, i.e., the frequency of the k_Δth frequent column in B. Now, if the frequency of one column is bigger (smaller) than ρ, the column is regarded as frequent (unfrequent) and labeled as "+1" ("−1"). Based on the distribution of frequent/unfrequent columns, unsafe zone is split into fine-grained blocks. Specifically, we construct two kinds of blocks.

- The first column in one block is frequent and labeled as "1", but the last one is unfrequent and labeled as "−1". Other columns between the first one and the last one are all frequent and labeled as "1". Obviously, there exist at least one frequent column in this block.
- The first column in one block is unfrequent and labeled as "−1", but the last one is frequent and labeled as "1". Other columns between the first one and the last one are all unfrequent and labeled as "−1". This kind of blocks contain one frequent column.

For example, in Figure 3, there exist block {C, D, E} with a label sequence (+1, +1, −1) and block {F, G, H} with a label sequence (−1, −1, +1). Obviously, every block contains one frequent column at least and hence the number of blocks is smaller than that of frequent columns.

The above phenomenon motivates us to design one block-centric mechanism "privacy-free-mechanism" (PFM) to choose frequent columns. PFM aims to pay privacy budget

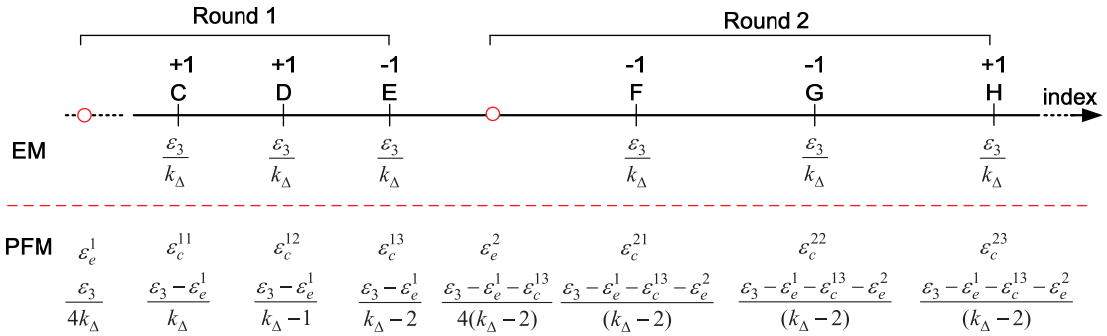


FIGURE 3. An example for privacy-free-mechanism.

only for the label judgement of the last column in one block by comparing ρ with the corresponding frequency, instead of the one of every column in the block. In this way, a lot of privacy budget is saved. Because when traversing the first kind of block containing multiple frequent columns, only the judgement for the last column costs privacy while others are privacy-free. Thus, the privacy consumption is proportional to the number of blocks, instead of frequent columns as exponential mechanism or original SVT. PFM is executed succinctly. In particular, PFM adds laplace noises with scale proportional to the number of blocks into the column frequencies and the threshold ρ, and compares the noisy frequencies with noisy versions of ρ to judge whether columns are frequent ("+1") or unfrequent ("−1"). Note that the noisy column frequencies in one block need to be compared with the same version of noisy $\widehat{\rho}$ but columns in different blocks need to be compared with different versions of noisy $\widehat{\rho}$s. The noise in $\widehat{\rho}$ can prevent the privacy from being posed by the judgement of the first $x-1$ columns in one block, where x denotes the length of this block. However, there are two problems required to be discussed in detail. Firstly, before finishing judgements of all columns in unsafe zone, the number of blocks is not available and hence the scale of noise added into column frequencies and threshold ρ cannot be determined beforehand. Thus, we allocate privacy budget for every column dynamically. Specifically, when making a judgment, the number of remaining blocks is estimated by its upper bound, i.e., the number of remaining frequent columns. And the remaining privacy budget is allocated evenly for these blocks. Secondly, since different blocks use different $\widehat{\rho}$s, we have to update $\widehat{\rho}$ when a new block is found, i.e., the label of the current column is different from that of the 1st column in this block.

Specifically, Algorithm 2 describes the block-centric method GetExtraColumns_PFM in detail. We firstly use a small part of privacy budget ε_e to generate one noisy version $\widehat{\rho}$ of ρ (Line 3). ε_3 is used to store the remaining privacy budget. And "flag" records the label of beginning column in the current block (Line 4). If flag is equal to "+1" ("−1"), that means the block needs to end up with one column with label "−1" ("+1"). If flag is equal to "0", that means a new block is going to be traversed and the flag label need to be re-initialized. R_Δ stores the frequent columns chosen by GetExtraColumns_PFM (Line 4). Next, we judge whether the columns in B are frequent or not in turn (Lines 5-18). Let ε_c denote the privacy budget paid for the frequency of the current column (Line 7). Note that $(k_\Delta - |R_\Delta|)$ indicates the number of the remaining frequent columns not found which is equal to the maximal number of the remaining blocks. So the remaining privacy budget ε_3 is evenly split into $(k_\Delta - |R_\Delta|)$

Algorithm 2 GetExtraColumns_PFM(B,D, ε_3, k_Δ)

Input : the columns in unsafe zone B, privacy budget ε_3,
the number of frequent columns to be chosen k_Δ
Output: extra frequent columns R_Δ

1 $\rho \leftarrow$ the frequency of the k_Δth frequent column in B;
2 $\varepsilon_e = \frac{\varepsilon_3}{4k_\Delta}$;
3 $\widehat{\rho} = \rho + \mathrm{Lap}(\frac{1}{\varepsilon_e})$, $\varepsilon_3 = \varepsilon_3 - \varepsilon_e$;
4 flag = 0, R_Δ={};
5 **while** ($|R_\Delta| < k_\Delta$ & $|B| > 0$ & $\varepsilon_3 > 0$) **do**
6 $e_i \leftarrow$ the first element in B, delete the first element in B;
7 $\varepsilon_c = \frac{\varepsilon_3}{k_\Delta - |R_\Delta|}$;
8 $\widehat{f} \leftarrow f(e_i, D) + \mathrm{Lap}(\frac{1}{\varepsilon_c})$;
9 **if** ($\widehat{f} > \widehat{\rho}$) **then**
10 $R_\Delta \leftarrow R_\Delta \cup e_i$;
11 **if** ($\widehat{f} > \widehat{\rho}$ & $flag = 0$) **then**
12 flag = +1;
13 **if** ($\widehat{f} < \widehat{\rho}$ & $flag = 0$) **then**
14 flag = -1;
15 **if** ($\widehat{f} > \widehat{\rho}$ & $flag = -1$) || ($\widehat{f} < \widehat{\rho}$ & $flag = 1$) **then**
16 $\varepsilon_3 \leftarrow \varepsilon_3 - \varepsilon_c$, $\varepsilon_e \leftarrow \frac{\varepsilon_3}{4(k_\Delta - |R_\Delta|)}$;
17 $\widehat{\rho} \leftarrow \rho + \mathrm{Lap}(\frac{1}{\varepsilon_e})$, $\varepsilon_3 \leftarrow \varepsilon_3 - \varepsilon_e$;
18 flag = 0;
19 Return R_Δ;

parts and every part is used to choose one frequent column. Thus ε_c is set as $\frac{\varepsilon_3}{k_\Delta - |R_\Delta|}$. Following that, we get the noisy frequency $\widehat{f}$ with noise scale $\frac{1}{\varepsilon_c}$ for the frequency of column e_i (Line 8). If $\widehat{f} > \widehat{\rho}$, that means e_i is frequent and added into R_Δ (Lines 9-10). When $\widehat{f} > \widehat{\rho}$ ($\widehat{f} < \widehat{\rho}$) and flag is equal to 0, e_i is the first column of one new block and the starting flag of this block is set to "+1" ("−1") (Lines 11-14). When $\widehat{f}$ is bigger (smaller) than $\widehat{\rho}$ and flag is equal to "+1" ("−1"), e_i reaches the end of the current block. Then privacy budget ε_c has to been consumed and the value of ε_3 is updated (Line 16). Afterwards, we use $\varepsilon_e = \frac{\varepsilon_3}{4(k_\Delta - |R_\Delta|)}$ to get a new noisy version of ρ, which is applied for the next block (Line 17). At the same time, flag is initialized to 0 (Line 18). The process is terminated when k_Δ frequent columns have been found or the columns in B are all traversed.

Figure 3 gives a running example of Algorithm 2. At the beginning, we get a noisy version of ρ using privacy budget $\varepsilon_e^1 = \frac{\varepsilon_3}{4k_\Delta}$ and the remaining privacy budget is thereby $\varepsilon_3 - \varepsilon_e^1$. Since k_Δ frequent columns are required, $\varepsilon_c^{11} = \frac{\varepsilon_3 - \varepsilon_e^1}{k_\Delta}$ is allocated for column C. Then the noisy frequency of C is bigger than $\widehat{\rho}$ and C is labeled as "+1". Furthermore, since C is the first column in the current block, flag is also set to "+1". Besides, we update the privacy budget for the frequency of next column as $\varepsilon_c^{12} = \frac{\varepsilon_3 - \varepsilon_e^1}{k_\Delta - 1}$, considering one frequent column C has been found. In the same way, the noisy frequency of D with noise scale $\frac{1}{\varepsilon_c^{12}}$ is bigger than $\widehat{\rho}$ and labeled as "+1". The privacy budget is updated again as $\varepsilon_c^{13} = \frac{\varepsilon_3 - \varepsilon_e^1}{k_\Delta - 2}$. Following that, the noisy frequency of E with noise scale ε_c^{13} is smaller than $\widehat{\rho}$ and labeled as "−1", which is different from the label of starting column C in this block. Accordingly, this block is terminated and the current privacy budget ε_c^{13} is consumed. Then there is $\varepsilon_3 - \varepsilon_e^1 - \varepsilon_c^{13}$ privacy budget left. Next, GetExtraColumns_PFM will start a new block and use the remaining privacy budget to choose $k_\Delta - 2$ frequent columns.

Note that the privacy-free-mechanism is not sensitive to the order of columns in the original dataset, because a built-in sorting mechanism in the first phase of TPS can sort the columns with the corresponding noisy frequencies.

In the following, we discuss the privacy guarantee of GetExtraColumns_PFM. Note that the privacy budget ε_3 in GetExtraColumns_PFM is consumed in two cases: the privacy budget is paid for the threshold ρ (Lines 3 and 17); the budget is paid for the frequency of the end column in each block (Line 16).

Theorem 2: GetExtraColumns_PFM shown in Algorithm 2 satisfies ε_3-differential privacy.

Please find the proof in Appendix.

VI. EXPERIMENTAL EVALUATION

This section evaluates the performance of the methods which publish the top-k frequent columns. Specifically, TPS+EM denotes our two-phase selection framework with exponential mechanism in the second phase. By contrast, TPS+PFM indicates TPS with the newly designed privacy-free-mechanism. Note that we set the privacy budget parameters involved in TPS (Algorithm 1) as $\varepsilon_1 = 0.1\varepsilon$, $\varepsilon_2 = 0.6\varepsilon$ and $\varepsilon_3 = 0.3\varepsilon$. We compare our proposals with the two most effective methods, freFirst [19] and EM [16], which are used in frequent items mining as the fundamental step of frequent itemsets mining [16], [17], [19], [23]. Note that freFirst represents the algorithms which publish noisy column frequencies firstly and then select top-k columns as results. Since DPsense is the state of the art technique for publishing column frequencies, we use DPsense to replace the method used in [19] for noisy column frequencies publication. Both freFirst and EM are one-phase-selection methods and have been described in Section III.C. All tests are run on real datasets and Table 1 shows some properties of these datasets, including the size of datasets $|D|$, the number of columns $|I|$ and the maximal/average number of "1"s in one record $\max|t|/avg|t|$.

TABLE 1. Description of real datasets.

Dataset	$\|D\|$	$\|I\|$	$\max\|t\|$	$avg\|t\|$
BMS-POS [29]	515,597	1,657	164	6.5
tafeng [30]	32,266	2012	405	17.7
retail [31]	88,162	16,470	76	10.3
BM1 [31]	59,602	497	267	2.5
pumsb star [31]	49,046	2,088	63	50.5
BM2 [31]	77,512	3,340	161	5.0

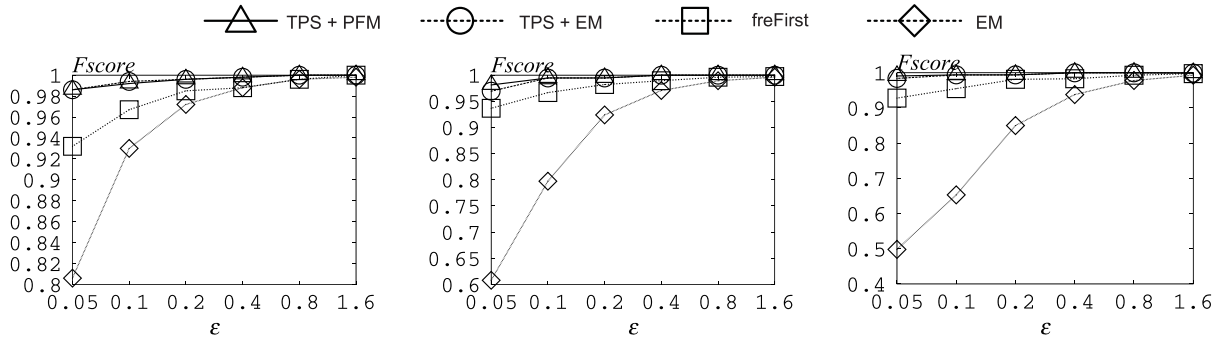
We use *Fscore* [17], [19] to measure the performance of these methods. Let R and $\widehat{R}$ denote the true result and noisy one respectively. Fscore is defined as $Fscore = 2 \times \frac{precision * recall}{precision + recall}$, where $precision = \frac{|R \cup \widehat{R}|}{|\widehat{R}|}$ and $recall = \frac{|R \cup \widehat{R}|}{|R|}$. Note that a higher Fscore means more accurate results are achieved.

The programs of all the methods are implemented by C++ and all the experiments are run on a PC with Intel i7 CPU, 8GB RAM and Windows 7 OS. And each experiment is repeated 10 times, with average results reported only.

Figures 4-9 show the Fscores of TPS+EM, TPS+PFM, freFirst and EM by varying εs on six datasets with different values of $k \in \{100, 150, 200\}$. We summarize our observations as follows.

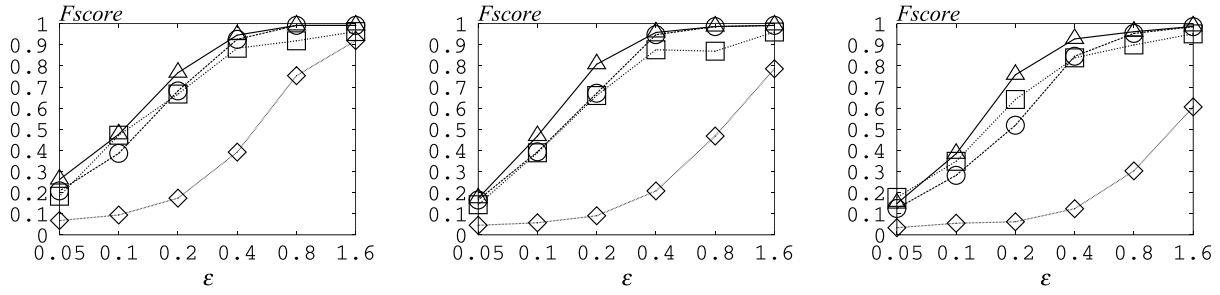
A. TPS vs. EM

Our TPS+EM outperforms EM by a large margin up to 0.7, yielding to nearly 7 times improvement. It is because EM uses exponential mechanism to sample all k frequent columns from the whole column set I. Generally, the size of I is big. For example, $|I|$ is up to 16470 in "retail". The large output domain makes the sampled probability of true frequent column decrease. On the contrary, in the second phase of TPS+EM, benefiting from pruning operations in the first phase, it samples only a part of frequent columns from unsafe zone. Considering the sampled probability of the true frequent column is inversely proportional to the number of frequent columns, TPS+EM performs better than EM.



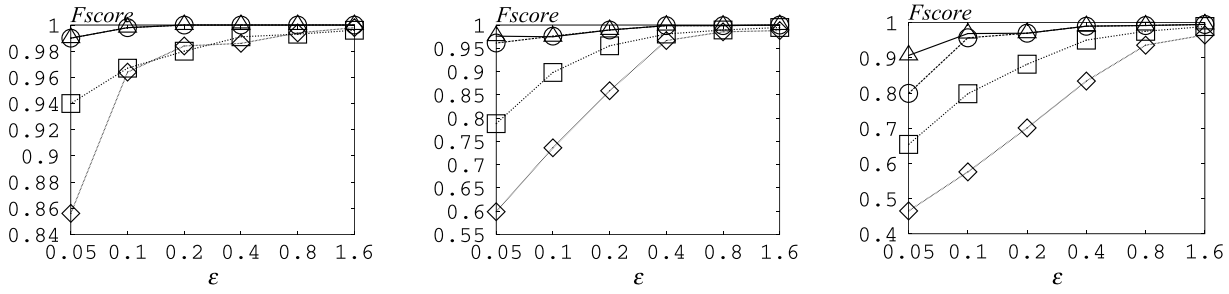
(a) $k = 100$ (b) $k = 150$ (c) $k = 200$

FIGURE 4. Fscore on the BMS-POS dataset.



(a) $k = 100$ (b) $k = 150$ (c) $k = 200$

FIGURE 5. Fscore on the retail dataset.



(a) k=100 (b) k=150 (c) k=200

FIGURE 6. Fscore on the pumsb dataset.

B. TPS VS. freFirst

In most cases, TPS+EM has a superiority to freFirst. For example, when $k = 200$ and $\varepsilon = 0.05$ in Figure 6, Fscore of TPS+EM is roughly 0.2 higher than that of freFirst. freFirst is one phase selection method by choosing top-k frequent columns based on noisy column frequencies. Generally, due to the noises in these frequencies, the ranks of columns near the kth index fluctuate heavily, which makes the chosen columns deviate from the true results. However, TPS+EM uses a part of privacy budget to re-choose the frequent columns from the columns whose ranks are near to k. The re-choosing can improve the accuracy of published results. And it is also observed that in only a few cases including $k = 200$ in tafeng, $k = 200$ in BM1, $k = 200$ in BM2 and $k = 150$ in BM2, freFirst slightly beats TPS+EM. That is because there are many frequent columns in unsafe zone and TPS+EM has to invoke exponential mechanism many times to sample these frequent columns. As a result of that, the privacy budget has to be split into many parts and every part is used to sample one column, which reduces the accuracy of the sampled result. For the above case, TPS+PFM is proposed to improve the accuracy of published results.

C. TPS+PFM VS. TPS+EM

TPS+PFM beats other methods all the time. TPS+PFM beats freFirst by up to 0.3 and 0.2 for TPS+EM, yielding to

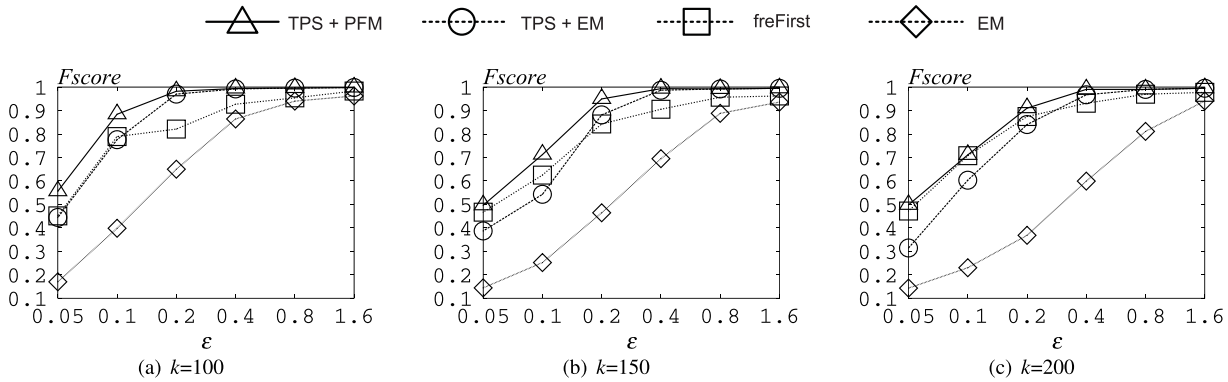


FIGURE 7. Fscore on the tafeng dataset.

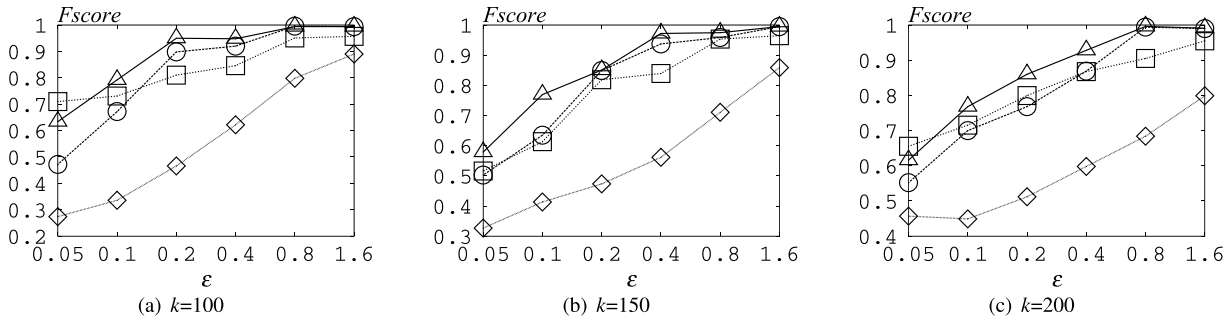


FIGURE 8. Fscore on the BM1 dataset.

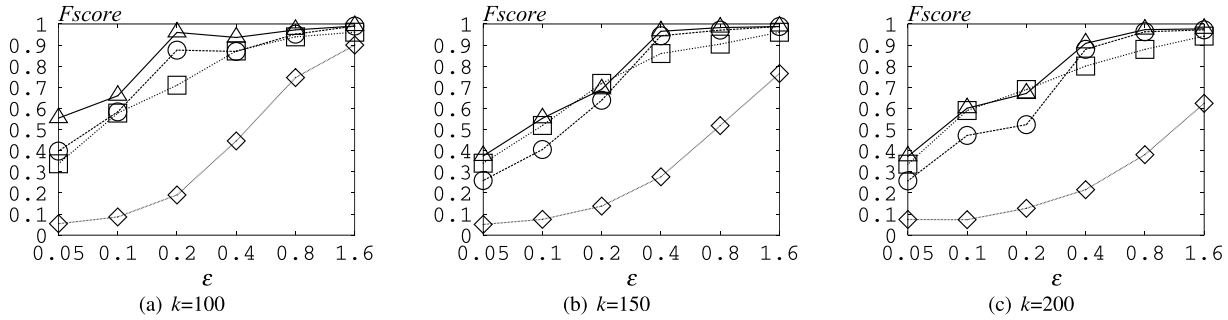


FIGURE 9. Fscore on the BM2 dataset.

nearly 50% improvement and 30% improvement respectively. We have already given the reason for the advantage of two phase selection method before. Then we discuss the advantage of TPS+PFM compared with TPS+EM. In the second phase of TPS+PFM, it applies privacy-free-mechanism to choose frequent columns. The privacy budget is split into some parts. The number of these parts is equal to the number of blocks generated during the choosing process and is smaller than the number of frequent columns to be sampled. TPS+PFM thereby can utilize more privacy budget to choose one frequent column. Then TPS+PFM can achieve a better performance.

VII. CONCLUSION

In this paper, we consider the problem of publishing top-k frequent columns for the high dimensional dataset in ε-differential privacy. We propose a novel two-phase selection framework denoted by TPS to choose frequent columns in the safe zone and unsafe zone respectively, since the abilities of columns to tolerate noises vary with their frequency ranks. Besides, we also design the differentially private block-centric column-choosing mechanism, PFM, to derive the frequent columns in the unsafe zone with low privacy consumption. Thorough experiments with real datasets are conducted to demonstrate the superiority of our techniques against the benchmarks.

THE PROOF OF THEOREM 2

Proof: Let D_1 and D_2 be neighboring databases, where $D_1 = D_2 \cup \{t\}$. And O denotes the label sequence of columns in B output by *GetExtraColumns_PFM*, where the label sequence consists of "+1"s and "−1"s. Besides, O_i denotes the output label sequence based on the ith block b_i in B. B^+ (B^-) is used to denote the set of the blocks which end up with

"+1" ("−1"). Let $p[O|B, D_1]$ ($p[O_i|b_i, D_1]$) be the probability that *GetExtraColumns_PFM* returns label sequence O (O_i) corresponding to all the columns in B (b_i) given database D_1. Then, the privacy cost of *GetExtraColumns_PFM* is:

$$\max_{D_1,D_2}\left|\log\left(\frac{p[O|B,D_1]}{p[O|B,D_2]}\right)\right| = \max_{D_1,D_2}\left|\log\left(\frac{\prod_{b_i\in B^+} p[O_i|b_i,D_1]\cdot\prod_{b_i\in B^-} p[O_i|b_i,D_1]}{\prod_{b_i\in B^+} p[O_i|b_i,D_2]\cdot\prod_{b_i\in B^-} p[O_i|b_i,D_2]}\right)\right| \quad (1)$$

To prove the theorem, it suffices to show that the right hand side (r.h.s.) of Equation 1 is no more than $\exp(\varepsilon_3)$.

Let $\widehat{\rho}_i^1$ ($\widehat{\rho}_i^2$) be the noisy version of ρ for the block b_i on D_1 (D_2). ε_e^i denotes the privacy budget allocated to ρ in block b_i. f_{ij}^1 (f_{ij}^2) and $\widehat{f}_{ij}^1$ ($\widehat{f}_{ij}^2$) are the frequency of the jth column in block b_i and its noisy version based D_1 (D_2). And ε_c^{ij} is the privacy budget allocated to f_{ij}^1 and f_{ij}^2. To show that the r.h.s of Equation 1 is no more than $\exp(\varepsilon_3)$, we firstly validate Formula 2 is satisfied for $b_i \in B^-$.

$$\exp(-\varepsilon_e^i - \varepsilon_c^{i|b_i|}) \le \frac{p[O_i|b_i,D_1]}{p[O_i|b_i,D_2]} \le \exp(\varepsilon_e^i + \varepsilon_c^{i|b_i|}). \quad (2)$$

In what follows, we focus on the case when $|D_1| = |D_2|+1$ and $b_i \in B^-$; the case when $|D_2| = |D_1| + 1$ can be proved in a similar manner.

First, for $b_i \in B^-$, $p[O_i|b_i, D_1]$ is the probability that *GetExtraColumn_PFM* outputs $(|b_i| - 1)$ "+1"s associated with the first $(|b_i| - 1)$ columns in b_i and one "−1" corresponding to the last column in b_i. And it can be written as the following formula:

$$p[O_i|b_i,D_1] = \int_{-\infty}^{+\infty}\Pr[\widehat{\rho}_i^1 = x]\left\{\left[\prod_{j=1}^{|b_i|-1}\int_x^{+\infty}\Pr[\widehat{f}_{ij}^1 = y]dy\right]\times\int_{-\infty}^{x}\Pr[\widehat{f}_{i|b_i|}^1 = y]dy\right\}dx$$

Considering the case when $|D_1| = |D_2| + 1$, where $f_{ij}^1 = f_{ij}^2$ or $f_{ij}^1 = f_{ij}^2 + 1$, we have

$$\frac{\Pr[\widehat{\rho}_i^1 = x]}{\Pr[\widehat{\rho}_i^2 = x-1]} \le e^{\varepsilon_e^i}, \quad \frac{\int_{-\infty}^{x}\Pr[\widehat{f}_{i|b_i|}^1 = y]dy}{\int_{-\infty}^{x-1}\Pr[\widehat{f}_{i|b_i|}^2 = y]dy} \le e^{\varepsilon_c^{i|b_i|}},$$

$$\int_x^{+\infty}\Pr[\widehat{f}_{ij}^2 = y]dy \le \int_x^{+\infty}\Pr[\widehat{f}_{ij}^1 = y]dy \le \int_{x-1}^{+\infty}\Pr[\widehat{f}_{ij}^2 = y]dy.$$

Then, the upperbound of $p[O_i|b_i, D_1]$ is shown in Formula 3:

$$\begin{aligned} &p[O_i|b_i,D_1] \\ &\le \int_{-\infty}^{+\infty}\Pr[\widehat{\rho}_i^1 = x]\left\{\left[\prod_{j=1}^{|b_i|-1}\int_{x-1}^{+\infty}\Pr[\widehat{f}_{ij}^2 = y]dy\right]\times\int_{-\infty}^{x}\Pr[\widehat{f}_{i|b_i|}^1 = y]dy\right\}dx \\ &\le \int_{-\infty}^{+\infty} e^{\varepsilon_e^i}\Pr[\widehat{\rho}_i^2 = x-1] \\ &\quad\times\left\{\left[\prod_{j=1}^{|b_i|-1}\int_{x-1}^{+\infty}\Pr[\widehat{f}_{ij}^2 = y]dy\right] e^{\varepsilon_c^{i|b_i|}}\int_{-\infty}^{x-1}\Pr[\widehat{f}_{i|b_i|}^2 = y]dy\right\}dx \\ &= e^{\varepsilon_e^i+\varepsilon_c^{i|b_i|}}p[O_i|b_i,D_2] \end{aligned} \quad (3)$$

Next, we infer the upperbound of $p[O_i|b_i, D_2]$ in Formula 4 :

$$\begin{aligned} &p[O_i|b_i,D_2] \\ &= \int_{-\infty}^{+\infty}\Pr[\widehat{\rho}_i^2 = x]\left\{\left[\prod_{j=1}^{|b_i|-1}\int_x^{+\infty}\Pr[\widehat{f}_{ij}^2 = y]dy\right]\times\int_{-\infty}^{x}\Pr[\widehat{f}_{i|b_i|}^2 = y]dy\right\}dx \\ &\le \int_{-\infty}^{+\infty}\Pr[\widehat{\rho}_i^2 = x]\left\{\left[\prod_{j=1}^{|b_i|-1}\int_x^{+\infty}\Pr[\widehat{f}_{ij}^1 = y]dy\right]\times\int_{-\infty}^{x}\Pr[\widehat{f}_{i|b_i|}^2 = y]dy\right\}dx \\ &\le \int_{-\infty}^{+\infty} e^{\varepsilon_e^i}\Pr[\widehat{\rho}_i^1 = x]\left\{\left[\prod_{j=1}^{|b_i|-1}\int_{x-1}^{+\infty}\Pr[\widehat{f}_{ij}^1 = y]dy\right] e^{\varepsilon_c^{i|b_i|}}\right. \\ &\quad\left.\times\int_{-\infty}^{x}\Pr[\widehat{f}_{i|b_i|}^1 = y]dy\right\}dx \\ &= e^{\varepsilon_e^i+\varepsilon_c^{i|b_i|}}p[O_i|b_i,D_1] \end{aligned} \quad (4)$$

So Formula 2 is proved by combining Formula 3 with Formula 4 when $b_i \in B^-$. In addition, for the case $b_i \in B^+$, we can prove Formula 2 in a similar way. Based on Eqn. 1, we have:

r.h.s. of Eqn. 1

$$\begin{aligned} &= \max_{D_1,D_2}\left|\log\left(\prod_{b_i\in B^+}\frac{p[O_i|b_i,D_1]}{p[O_i|b_i,D_2]}\cdot\prod_{b_i\in B^-}\frac{p[O_i|b_i,D_1]}{p[O_i|b_i,D_2]}\right)\right| \\ &= \log\left(\prod_{b_i\in B^+}\exp(\varepsilon_c^i+\varepsilon_e^{i|b_i|})\cdot\prod_{b_i\in B^-}\exp(\varepsilon_c^i+\varepsilon_e^{i|b_i|})\right) \\ &= \sum_{b_i\in B^+\cup B^-}\varepsilon_e^i + \sum_{b_i\in B^+\cup B^-}\varepsilon_c^{i|b_i|} \\ &\le \exp(\varepsilon_3) \end{aligned}$$

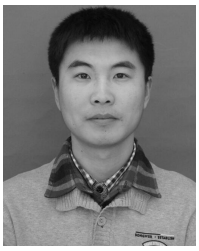
Thus, the theorem is proved. ■

REFERENCES

[1] W.-Y. Day and N. Li, "Differentially private publishing of high-dimensional data using sensitivity control," in *Proc. ASIA CCS*, 2015, pp. 451–462.
[2] G. Kellaris and S. Papadopoulos, "Practical differential privacy via grouping and smoothing," *Proc. VLDB Endowment*, vol. 6, no. 5, pp. 301–312, 2013.
[3] V. Rastogi and S. Nath, "Differentially private aggregation of distributed time-series with transformation and encryption," in *Proc. SIGMOD*, 2010, pp. 735–746.
[4] Z. Qin, Y. Yang, T. Yu, I. Khalil, X. Xiao, and K. Ren, "Heavy hitter estimation over set-valued data with local differential privacy," in *Proc. CCS*, 2016, pp. 192–203.

[5] J. Hsu, S. Khanna, and A. Roth, "Distributed private heavy hitters," in *Proc. ICALP*, 2012, pp. 461–472.
[6] T. H. H. Chan, M. Li, E. Shi, and W. Xu, "Differentially private continual monitoring of heavy hitters from distributed streams," in *Proc. Int. Conf. Privacy Enhancing Technol.*, 2012, pp. 140–159.
[7] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *Proc. VLDB*, 1994, pp. 487–499.
[8] R. J. Bayardo, "Efficiently mining long patterns from databases," *SIGMOD*, vol. 27, pp. 85–93, Jun. 1998.
[9] D. Burdick, M. Calimlim, J. Flannick, J. Gehrke, and T. Yiu, "MAFIA: A maximal frequent itemset algorithm," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1490–1504, Nov. 2005.
[10] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: A frequent-pattern tree approach," *Data Mining Knowl. Discovery*, vol. 8, no. 1, pp. 53–87, 2004.
[11] (2017). *Amazon*. [Online]. Available: https://www.amazon.com/
[12] (2017). *Ebay*. [Online]. Available: http://www.ebay.com/
[13] C. Dwork, G. N. Rothblum, and S. P. Vadhan, "Boosting and differential privacy," in *Proc. FOCS*, 2010, pp. 51–60.
[14] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Proc. FOCS*, 2007, pp. 94–103.
[15] X. Xiao, G. Wang, and J. Gehrke, "Differential privacy via wavelet transforms," in *Proc. ICDE*, 2010, pp. 225–236.
[16] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta, "Discovering frequent patterns in sensitive data," in *Proc. SIGKDD*, 2010, pp. 503–512.
[17] N. Li, W. Qardaji, D. Su, and J. Cao, "Privbasis: Frequent itemset mining with differential privacy," *Proc. VLDB Endowment*, vol. 5, no. 11, pp. 1340–1351, 2012.
[18] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, nos. 3–4, pp. 211–407, 2014.
[19] C. Zeng, J. F. Naughton, and J. Cai, "On differentially private frequent itemset mining," *PVLDB*, vol. 6, no. 1, pp. 25–36, 2012.
[20] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. TCC*, 2006, pp. 265–284.
[21] R. Chen, N. Mohammed, B. C. M. Fung, B. C. Desai, and L. Xiong, "Publishing set-valued data via differential privacy," *Proc. VLDB Endowment*, vol. 4, no. 11, pp. 1087–1098, 2011.
[22] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "Privbayes: Private data release via Bayesian networks," in *Proc. SIGMOD*, 2014, pp. 1423–1434.
[23] J. Lee and C. W. Clifton, "Top-K frequent itemsets via differentially private FP-trees," in *Proc. KDD*, 2014, pp. 931–940.
[24] N. Wang, X. Xiao, Y. Yang, Z. Zhang, Y. Gu, and G. Yu, "Privsuper: A superset-first approach to frequent itemset mining under differential privacy," in *Proc. ICDE*, 2017, pp. 809–820.
[25] J. Zhang, X. Xiao, and X. Xie, "Privtree: A differentially private algorithm for hierarchical decompositions," in *Proc. SIGMOD*, 2016, pp. 155–170.
[26] Y. Chen and A. Machanavajjhala, "On the privacy properties of variants on the sparse vector technique," *CoRR*, vol. abs/1508.07306, 2015.
[27] C. Dwork, "Differential privacy," in *Proc. ICALP*, 2006, pp. 1–12.
[28] F. McSherry, "Privacy integrated queries: An extensible platform for privacy-preserving data analysis," in *Proc. SIGMOD*, 2009, pp. 19–30.
[29] (2019). *KDD Cup 2000*. [Online]. Available: http://www.kdd.org/kdd-cup/view/kdd-cup-2000
[30] (2019). *Ta Feng Grocery Dataset*. [Online]. Available: https://stackoverflow.com/questions/25014904/download-link-for-ta-feng-grocery-dataset
[31] (2019). *An Open-Source Data Mining Library*. [Online]. Available: https://www.philippe-fournier-viger.com/spmf/index.php?link=dataset.php

NING WANG received the B.E., M.E., and Ph.D. degrees in computer science from Northeastern University, China, in 2011, 2013, and 2017, respectively. She is currently a Lecturer with the College of Information Science and Engineering, Ocean University of China. Her research interest lies in data privacy protection and data management. She is a member of the China Computer Federation (CCF).



ZHIGANG WANG received the Ph.D. degree in computer software and theory from Northeastern University, China, in 2018. He is currently a Lecturer with the College of Information Science and Engineering, Ocean University of China. His research interests include cloud computing, distributed graph processing, and machine learning. He is a member of the China Computer Federation (CCF). He received the CCF Outstanding Doctoral Dissertation Award, in 2018.

YU GU received the Ph.D. degree in computer software and theory from Northeastern University, China, in 2010. He is currently a Professor and the Ph.D. Supervisor with Northeastern University. His current research interests include big data analysis, spatial data management, and graph data management. He is a Senior Member of the China Computer Federation (CCF).

JIA XU received the Ph.D. degree in computer science from Northeastern University, China, in 2013. She is currently an Associate Professor with the School of Computer, Electronics and Information, Guangxi University, China. Her research interests include data query processing and data privacy protection.

ZHIQIANG WEI received the Ph.D. degree from Tsinghua University, China, in 2001. He is currently a Professor with the Ocean University of China. He is also the Director of the High Performance Computing Center at the Pilot National Laboratory for Marine Science and Technology, Qingdao. His current research interests are in the fields of intelligent information processing, social media, and big data analytics. He is a member of CCF.

GE YU received the Ph.D. degree in computer science from Kyushu University, Japan, in 1996. He is currently a Professor and the Ph.D. Supervisor with Northeastern University, China. His research interests include distributed and parallel database, OLAP and data warehousing, data integration, and graph data management. He is a member of ACM a Fellow of the China Computer Federation (CCF).