Marginal Release under Multi-Party Personalized Differential Privacy

Peng Tang[1,2,†], Rui Chen[3,*], Chongshi Jin[1,2,‡], Gaoyuan Liu[1,2,‡], and Shanqing Guo[1,2,†,*]

[1] Key Laboratory of Cryptologic Technology and Information Security Ministry of Education, Shandong Universtity, Qingdao, China
[2] School of Cyber Science and Technology, Shandong University, Qingdao, China
[†]{tangpeng, guoshanqing}@sdu.edu.cn,
[‡]{jinchongshi, gaoyuan_liu}@mail.sdu.edu.cn
[3] College of Computer Science and Technology, Harbin Engineering University, Harbin, China
ruichen@hrbeu.edu.cn

Abstract. Given a set of local datasets held by multiple parties, we study the problem of learning marginals over the integrated dataset while satisfying differential privacy for each local dataset. Different from existing works in the multi-party setting, our work allows the parties to have different privacy preferences for their data, which is referred to as the multi-party personalized differential privacy (PDP) problem. The existing solutions to PDP problems in the centralized setting mostly adopt sampling-based approaches. However, extending similar ideas to the multi-party setting cannot satisfactorily solve our problem. On the one hand, the data owned by multiple parties are usually not identically distributed. Sampling-based approaches will incur a serious distortion in the results. On the other hand, when the parties hold different attributes of the same set of individuals, sampling at the tuple level cannot meet parties' personalized privacy requirements for different attributes.

To address the above problems, we first present a mixture-of-multinomials-based marginal calculation approach, where the global marginals over the stretched datasets are formalized as a multinomial mixture model. As such, the global marginals over the original datasets can be reconstructed based on the calculated model parameters with high accuracy. We then propose a privacy budget segmentation method, which introduces a privacy division composition strategy from the view of attributes to make full use of each party's privacy budget while meeting personalized privacy requirements for different attributes. Extensive experiments on real datasets demonstrate that our solution offers desirable data utility.

Keywords: Personalized differential privacy · Multiple party · Marginal release.

* Shanqing Guo and Rui Chen are co-corresponding authors.

1 Introduction

In many real-life applications, a mass of data are stored among multiple distributed parties [28]. There are two typical multi-party settings: *horizontally partitioned* and *vertically partitioned*. In the former setting, it is assumed that all the local databases have the same schema and that the parties possess different individuals' information. In the latter one, all of the local datasets are over the same set of individuals, and each party observes a subset of attributes of the individuals. Calculating the marginals over such distributed data can lead to better decision-making. However, since such data may contain highly sensitive personal information, calculating the marginals in the multi-party setting needs to be conducted in a way that no private information is revealed to other participating entities or any other potential adversaries. In the multi-party setting, differential privacy has been widely used in the distributed data analysis [3,8,23,30,32]. All these above studies afford the same level of privacy protection for the individuals of all the local datasets. However, it is common that the parties have different expectations regarding their data's acceptable level of privacy. That is, users in different local datasets can have different privacy needs, where some users are extremely restrictive while others are relatively loose. As a real-world example, an analyst may want to do medical research on a hospital's data. This requires integrated data from different hospital departments. Some departments that treat sensitive diseases may require a higher level of privacy needs than others. As another practical example, medical researchers may want to study a potential correlation between travel patterns and certain types of illnesses. This requires integrated data from different sources, such as an airline reservation system and a hospital database. As medical data is usually more sensitive, the hospital may have a higher privacy need. In the above scenarios, the data analyst employing differential privacy has limited options. Setting high-level global privacy to satisfy all the local datasets will introduce a large amount of noise into the analysis outputs, resulting in poor utility. While, setting a lower privacy level may force the analyst to exclude the local datasets with strict privacy needs from analysis, which may also significantly harm utility.

This leads to the multi-party personalized differential privacy (PDP) problem. To achieve PDP in the centralized setting, Jorgensen et al. [18] propose an advanced method $\mathcal{PE}$. Analogous to the exponential mechanism [21], for each item in a marginal, $\mathcal{PE}$ calculates its noisy count by sampling from an output set. Specifically, the data owner first calculates the item's true count. Based on this count, the owner can compute the score of the true count and the score of the other noisy counts, and then sample one from these counts according to their scores. However, such a method cannot be extended to the multi-party setting. As $\mathcal{PE}$ requires knowing the true global count of each item. While in the multi-party setting, to guarantee differential privacy for each local dataset, it is not allowed for each party to learn the true global count. Other solutions [18,20] mostly adopt sampling-based approaches. Such approaches capture the additional randomness about the input data by employing non-uniform random sampling at the tuple level to yield the precise level of privacy required by each individual.

Unfortunately, extending similar ideas to the multi-party setting cannot satisfactorily solve our problem. In the horizontally partitioned setting, the data owned by multiple parties are usually not identically distributed. Then there will be a serious distortion in the results calculated by sampling-based approaches, i.e., the marginals calculated over the non-uniformly sampled datasets are not equal to the marginals calculated over the original datasets. In the vertically partitioned setting, the parties hold different attributes of the same individuals, and the attributes in different local datasets have different privacy requirements. Then, sampling will be "invalid" to adjust privacy preference because sampling at the tuple level cannot meet the parties' personalized privacy requirements for different attributes.

1.1 Contributions

To address the above challenges, we first present a mixture-of-multinomials-based marginal calculation approach for the horizontally partitioned setting. In this approach, stretching [2] is used to adjust the parties' different privacy preferences while avoiding the error caused by sampling, and the global marginals can be formalized as a multinomial mixture model. Thus, it is possible first to calculate marginals over the stretched datasets and then accurately reconstruct the global marginals over the original datasets by calculating the model parameters.

For the vertically partitioned setting, we propose a privacy budget segmentation method, which can adjust privacy preferences from the view of the attribute. This method elaborately divides the privacy budget of each party into multiple parts, and let the parties assemble some different teams. Each team calculates an intermediate result by consuming part of the privacy budget. Based on these intermediate results, this method can reconstruct the marginal by employing consistency post-processing. Using such a privacy division composition strategy, this method can fully use each party's privacy budget while satisfying personalized privacy requirements for different attributes.

We conduct an extensive experimental study over several real datasets. The experimental results suggest that our methods are practical to offer desirable data utility.

2 Related work

There exist three kinds of most relevant works, i.e., personalized privacy in the centralized setting, multi-party differential privacy, and local differential privacy.

In the centralized setting, personalized privacy allows the users have quite different expectations regarding the acceptable level of privacy for their data. A line of work, started by Xiao and Tao [36], introduce personalized privacy for k-anonymity, and present a new generalization framework called *personalized anonymity*. For differential privacy, Alaggan et al. [2] develop the privacy notion called *heterogeneous differential privacy*, which considers differential privacy with non-uniform privacy guarantees. Following, Jorgensen et al. [18] propose the privacy definition called *personalized differential privacy* (PDP), where

users specify a personal privacy requirement for their data, and introduce an advanced method $\mathcal{PE}$ for achieving PDP. Recently, Kotsogiannis et al. [20] study the problem of privacy-preserving data sharing, wherein only a subset of the records in a database is sensitive. To pursue higher data utility while satisfying personalized differential privacy, Niu et al. [27] propose a utility-aware personalized Exponential mechanism. These approaches inspire us to initiate a new approach to solving our problem. However, unlike the centralized setting, in the distributed setting, each party is not allowed to reveal the sensitive personal information that contained in their local datasets to other parties. Besides, the data owned by multiple parties are usually not identically distributed, and the attributes of the same individuals may have different privacy requirements. These new challenges are exactly the focus of our work.

In the multi-party setting, differential privacy has been widely used in the distributed data analysis [4,7,13,15,25]. Besides, there are some works [3,8,23,30, 32] for differentially privately data publishing in the multi-party setting. Using the published integrated data, the marginals can also be calculated. Different from our work, all these studies afford the same level of privacy protection for the individuals of all the local datasets. In contrast, our work aims to satisfy each party's different privacy preferences.

In the distributed scenario, another kind of differential privacy exists, i.e., Local Differential Privacy (LDP). There exist some studies of personalized differential privacy in the *local setting* [6, 14, 16, 26, 29, 35, 37]. Both multi-party differential privacy and LDP do not require a trusted data aggregate. However, as discussed in [34], in LDP, each user independently perturbs their own input before the aggregation on an untrusted server. This results in a large error of $O(\sqrt{N})$ in the output, where N denotes the number of users. While in multi-party differential privacy, there is a complementary synergy between secure multiparty computation and differential privacy. Multi-party differential privacy can maintain the same level of accuracy as in centralized differential privacy. The final output has only an error of $O(1)$.

3 Preliminaries

Differential privacy [11] is a recent privacy definition that provides a strong privacy guarantee. Naturally, differential privacy is built upon the concept of *neighboring databases*. Two databases D and $\hat{D}$ are neighbors if they differ on at most one record. Differential privacy can be defined as follows.

Definition 1. *A randomized algorithm φ achieves ε-differential privacy, if for any pair of neighboring databases D and $\hat{D}$, and all $\mathcal{O} \subseteq Range(\varphi)$,*

$$\Pr\left(\varphi(D) \in \mathcal{O}\right) \leq e^{\varepsilon} \times \Pr\left(\varphi(\hat{D}) \in \mathcal{O}\right), \tag{1}$$

where the probability $\Pr(\cdot)$ is taken over coin tosses of φ.

A fundamental concept for achieving differential privacy is *sensitivity* [11]. Let F be a function that maps a database into a fixed-size vector of real numbers. For all neighboring databases D and $\hat{D}$, the sensitivity of F is: $S(F) = \max_{D,\hat{D}} \left\| F(D) - F\left(\hat{D}\right) \right\|_1$, where $\|\cdot\|_1$ denotes the L_1 norm. For a function F whose outputs are real, differential privacy can be achieved by the *Laplace mechanism* [11]. This mechanism works by adding random noise to the true outputs. The noise is drawn from a Laplace distribution with the probability density function $p(x) = \frac{1}{2\lambda} e^{-|x|/\lambda}$, where the scale $\lambda = S(F)/\varepsilon$ is determined by both the function's sensitivity $S(F)$ and the privacy budget ε.

4 Problem Formulation

4.1 System and Threat Models

Following the common convention [5, 19, 30] in the fields of privacy, we consider a *semi-trusted* curator in our setting. With the assistance of the curator, K parties calculate the marginals over the integrated dataset collaboratively. Both the parties and the curator are semi-trusted (i.e., "honest-but-curious"). That is, the parties and the curator will correctly follow the designed protocols, but act in a "curious" fashion that they may infer private information other than what they are allowed to learn (e.g., sensitive information about the tuples in the local datasets). Our threat model also considers collusion attacks. In particular, there exist two kinds of collusion attacks. One kind is collusion attacks among the parties, and the other is collusion attacks between some parties and the curator.

In our problem, there is a complementary synergy between secure multiparty computation and differential privacy. Together they can prevent attackers from inferring sensitive information about the input local datasets using either intermediate results or outputs. Certainly, this requires an additional assumption of all parties and the curator being computationally bounded in the protocol. Therefore, in our privacy model, the overall scheme actually satisfies *computational differential privacy* [22, 34].

4.2 Problem Definition

In the problem of *multi-party marginal calculation under personalized differential privacy*, there are K parties (i.e., data owners), each of which P_k ($1 \leq k \leq K$) holds a local dataset D_k and specifies a privacy budget ε_k. The attributes contained in D_k can be either numerical or categorical. Over the local datasets, the K parties would like to jointly calculate the marginal of a given attribute set $\mathcal{X}$, while meeting multi-party personalized differential privacy. Multi-party personalized differential privacy is a kind of computational differential privacy, defined below.

Definition 2. *There are K parties. All parties are assumed to be computationally bounded, and each of them specifies a privacy budget ε_k. A randomized*

algorithm φ achieves multi-party personalized differential privacy, if the computing is secure according to secure multiparty computation, and for any two sets of datasets $\{D_1, \ldots, D_K\}$ and $\left\{\hat{D}_1, \ldots, \hat{D}_K\right\}$, where there exists a k in $\{1, 2, \cdots, K\}$, D_k and $\hat{D}_k$ are neighbors ($|D_k \oplus \widehat{D}_k| = 1$), and for any other $k' \neq k$ in $\{1, 2, \ldots, K\}$, $D_{k'} = \hat{D}_{k'}$, and for all $\mathcal{O} \subseteq Range(\varphi)$,

$$\Pr\left(\varphi(\bigcup_{k=1}^{K} D_k) \in \mathcal{O}\right) \leq e^{\varepsilon_k} \times \Pr\left(\varphi(\bigcup_{k=1}^{K} \hat{D}_k) \in \mathcal{O}\right).$$

There are two typical multi-party settings: horizontally partitioned setting and vertically partitioned setting. In the former setting, it is assumed that all the local datasets have the same schema (i.e., attribute set) $\mathcal{A} = \{A_1, \ldots, A_n\}$ and that a single individual's information is exclusively possessed by a single party, and the given attribute set $\mathcal{X} \subseteq \mathcal{A}$. In the latter one, all of the local datasets are over the same set of individuals that are identified by a common identifier attribute. $\mathcal{A}_i$ denotes the set of attributes observed by P_i. It is assumed that, for any two local datasets D_i and D_j, $\mathcal{A}_i \cap \mathcal{A}_j = \emptyset$. The attribute set $\mathcal{X} = \bigcup_{k=1}^{K} \mathcal{X}_k$ and $\mathcal{X}_k \subseteq \mathcal{A}_k$. In the vertically partitioned setting, it is common to assume that different parties share common identifiers of the users and hold mutually exclusive sets of attributes [17, 23, 24]. If the parties have overlapping attributes, they can send their data schemas to the curator to constructs exclusive sets of attributes as a preprocessing step of our solution. Since data schemas are considered public information, such a process does not lead to privacy breaches.

5 Baseline Solutions and Limitations

5.1 Horizontally Partitioned Setting

To solve the problem in the horizontally partitioned setting, there exist three kinds of baseline solutions. Firstly, a straightforward method lets each party add noise of different levels to the local marginals before sharing them with the curator. However, this will lead to the global marginals containing multiple noises, making the results useless (as discussed in Section 7.2). Secondly, in the centralized setting, Jorgensen et al. [18] propose an advanced method $\mathcal{PE}$ to achieve PDP. Analogous to the exponential mechanism [21], $\mathcal{PE}$ calculates its noisy count by sampling from an output set for each item in a marginal. However, such a method cannot be extended to the multi-party setting. As $\mathcal{PE}$ requires knowing the true global count of each item. While in the multi-party setting, to guarantee differential privacy for each local dataset, it is not allowed for each party to learn the true global count. Thirdly, a sampling-based solution can be proposed. Specifically, each party P_k first takes sampling on the tuples in D_k with the probability $p_k = \frac{e^{\varepsilon_k} - 1}{e^{\varepsilon_{\max}} - 1}$ to obtain a sampled dataset $\widetilde{D_k}$, where $\varepsilon_{\max} = \max\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_K\}$. Based on these sampled datasets, the curator and the parties can calculate a noisy marginal of $\mathcal{X}$ using ε_{max} as the privacy parameter. However, the data owned by multiple parties are usually not

identically distributed. Sampling with different probabilities on multiple datasets will seriously distort the results calculated in the sampling-based approaches.

Example 1. There exist three parties P_1, P_2, P_3. Each P_k $(1 \leq k \leq K)$ holds a local medical dataset D_k and specifies a privacy budget ε_k. Let the size of each local dataset be $|D_1| = |D_2| = |D_3| = 100$, and $\varepsilon_1 = 0.1, \varepsilon_2 = 0.3, \varepsilon_3 = 0.5$. Each local dataset contains some cancer patients, and the number of patients is 50, 30, 20, respectively. To calculate the probability of the patients over the global dataset $\bigcup_{k=1}^{3} D_k$ while satisfying personalized differential privacy for each local dataset, P_1, P_2, P_3 first take sampling on the tuples in their local dataset with the probability $p_1 = \frac{e^{0.1}-1}{e^{0.5}-1} \approx 0.2, p_2 \approx 0.6, p_3 = 1$ to obtaining sampled datasets $\widetilde{D_1}, \widetilde{D_2}, \widetilde{D_3}$. Based on these sampled datasets, the curator and the parties can calculate a probability of the patients, approximately equal to $\frac{0.2\times50+0.6\times30+1\times20}{0.2\times100+0.6\times100+1\times100} = \frac{48}{180} \approx 0.27$. However, the actual probability of the patients is $\frac{50+30+20}{100+100+100} = \frac{100}{300} \approx 0.33$. Thus, the result calculated on the sampled data set is far from the result calculated on the original data set.

5.2 Vertically Partitioned Setting

In the vertically partitioned setting, each party P_k holds a local dataset D_k with a set of attributes $\mathcal{A}_k$, and keeps a privacy budget ε_k, where $k \in \{1, 2, \ldots, K\}$. For a given attribute set $\mathcal{X}$, where $\mathcal{X} = \bigcup_{k=1}^{K} \mathcal{X}_k$ and $\mathcal{X}_k$ is from D_k, i.e., $\mathcal{X}_k \subseteq \mathcal{A}_k$, the curator and the parties want to calculate its marginal under personalized differential privacy. The intuitive idea is that, following the methods used in the centralized setting, each party P_k first takes sampling on the tuple level in D_k with the probability $p_k = \frac{e^{\varepsilon_k}-1}{e^{\varepsilon_{\max}}-1}$ to get a sampled dataset $\widetilde{D_k}$, where $\varepsilon_{\max} = \max\{\varepsilon_k | 1 \leq k \leq K\}$. Then the curator and the parties calculate the marginal distribution of $\mathcal{X}$ over the integrated sampled dataset $\bowtie_{k=1}^{K} \widetilde{D_k}$ with privacy budget $\varepsilon_{\max}$, where $\bowtie$ denotes the join of two datasets. However, such a sampling method cannot meet the personalized privacy preference of attributes in different local datasets. It would also lead to the sampled global dataset being too sparse, which will reduce the utility of the calculated marginal distribution. The reason lies in that all the local datasets are over the same set of individuals in the vertically partitioned setting. Employing sampling at the tuple level on multiple datasets is equivalent to sampling individuals with the same small probability. Specifically, for any individual with $ID = x$, we have $\Pr\left(x \in \bowtie_{k=1}^{K} \widetilde{D_k}\right) = \prod_{k=1}^{K} p_k$. With the increase of K, $\prod_{k=1}^{K} p_k$ becomes smaller and $\bowtie_{k=1}^{K} \widetilde{D_k}$ becomes sparser.

By careful analysis, we learn that the main cause of the issue is that, in the vertically partitioned setting, personalized privacy requirements are for different attributes, while sampling is working at the tuple level. Therefore, to guarantee personalized differential privacy for each local dataset while enjoying reduced noise in the vertically partitioned setting, we need to propose a privacy adjusting method from the view of the attribute.

6 Our Solution

This section proposes the mixture-of-multinomials-based approach for the horizontally partitioned setting and the privacy budget segmentation method for the vertically partitioned setting. Note that, in the multi-party setting, all the communications between the curator and parties must be secure to guarantee computational differential privacy for each local dataset. We will first focus on the noisy marginals computation methods and then describe their implementation details under encryption.

6.1 Horizontally Partitioned Setting

In this setting, we propose a mixture of multinomials based method. In this method, the global marginals of $\mathcal{X}$ can be seen as a mixture of multinomial distributions and calculated by maximizing a posterior. The details are as follows.

1. Calculating the local counts. Given the attribute set $\mathcal{X}$, for each $\mathcal{X} = x_i$, where $i \in \{1, 2, \ldots, l\}$ and l denotes the size of the domain of $\mathcal{X}$, each party P_k calculates its local count c_{ik} over dataset D_k and multiplies the count by a scaling factor $s_k = \frac{\varepsilon_k}{\varepsilon_{max}}$, i.e., $\widetilde{c_{ik}} = c_{ik} \cdot s_k$. This can be seen as performing statistics on a stretched dataset $\widetilde{D_k}$. In the stretched dataset, the count of each tuple is multiplied by s_k. Here stretching is used to adjust the parties' different privacy preferences to satisfy personalized differential privacy, while avoiding the error caused by sampling.

2. Construction of the likelihood function. Based on the local counts, the parties and the curator can obtain the number $\widetilde{c_i}$ of the tuples with $\mathcal{X} = x_i$ for each $i \in \{1, 2, \ldots, l\}$ that contained in the stretched datasets, i.e., $\widetilde{c_i} = \sum_{k=1}^{K} \widetilde{c_{ik}}$. As the distribution of $\mathcal{X}$ in each local stretched dataset $\widetilde{D_k}$ (referred to as $\Pr\left(\mathcal{X}|\widetilde{D_k}\right)$) follows a multinomial distribution with parameters $\{\mu_{1k}, \mu_{2k}, \ldots, \mu_{lk}\}$, where $\mu_{ik} = \Pr\left(x_i|\widetilde{D_k}\right)$. And the prior probability of each multinomial element is $\alpha_k = \Pr\left(\widetilde{D_k}\right) = (s_k \cdot |D_k|)/\sum_{j=1}^{K}(s_j \cdot |D_j|)$. Thus the curator can calculate the probability $\mathcal{L} = \prod_{i=1}^{l} \left(\Pr\left(x_i\right)\right)^{\widetilde{c_i}}$, where $\Pr\left(x_i\right) = \sum_{k=1}^{K} \Pr\left(x_i|\widetilde{D_k}\right) \cdot \Pr\left(\widetilde{D_k}\right) = \sum_{k=1}^{K} \mu_{ik} \cdot \alpha_k$. $\mathcal{L}$ can be referred to as the likelihood function. The corresponding logarithmic likelihood function is:

$$\log\left(\mathcal{L}\right) = \log\left(\prod\nolimits_{i=1}^{l} \left(\Pr\left(x_i\right)\right)^{\widetilde{c_i}}\right) = \sum\nolimits_{i=1}^{l} \widetilde{c_i} \cdot \log\left(\sum\nolimits_{k=1}^{K} \mu_{ik} \cdot \alpha_k\right).$$

Note that, $\sum_{i=1}^{l} \mu_{ik} = 1$ and $\sum_{k=1}^{K} \alpha_k = 1$. Thus, it can be seen as a constrained maximization problem.

3. Calculation of model parameters μ_{ik}. Given the local datasets D_k and the scaled factors s_k (where $1 \leq k \leq K$), $\alpha_k = (s_k \cdot |D_k|)/\sum_{j=1}^{K}(s_j \cdot |D_j|)$ can be seen as a constant. And μ_{ik} can be calculated by using the criterion of maximum likelihood estimation. We introduce Lagrange multipliers λ_k $(1 \leq k \leq K)$ to enforce the normalization constraint and then reduce the constrained maximization problem to the unconstrained maximization problem:

$$L = \sum\nolimits_{i=1}^{l} \widetilde{c}_i \cdot \log\left(\sum\nolimits_{k=1}^{K} \mu_{ik} \cdot \alpha_k\right) - \sum\nolimits_{k=1}^{K}\left(\lambda_k\left(\sum\nolimits_{i=1}^{l} \mu_{ik} - 1\right)\right).$$

4. Recalculation of the marginal of $\mathcal{X}$. The local marginal of $\mathcal{X}$ in the stretched dataset $\widetilde{D_k}$ is equal to that in the original dataset D_k, i.e., $\Pr(x_i|D_k) = \Pr\left(x_i|\widetilde{D_k}\right) = \mu_{ik}$. Based on the calculated μ_{ik}, the curator can recalculate the marginal distribution of $\mathcal{X}$ over the original datasets:

$$\widehat{\Pr(x_i)} = \frac{\sum_{k=1}^{K} \mu_{ik} \cdot |D_k|}{\sum_{k=1}^{K} |D_k|} \tag{2}$$

6.2 Vertically Partitioned Setting

In the vertically partitioned setting, we propose a privacy budget segmentation method, which is a privacy adjusting method from the view of the attribute. The method mainly consists of the following 4 steps.

1. We first sort the parties $P_1, P_2, \ldots, P_K$ according to their privacy budget. The sorted result can be denoted as $P_{s_1}, P_{s_2}, \ldots, P_{s_K}$, where for any $1 \leq i < j \leq K$, $\varepsilon_{s_i} \leq \varepsilon_{s_j}$. Note that, $\varepsilon_{s_1} = \min\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_K\}$.
2. For each $k \in \{1, 2, \ldots, K\}$, the party P_{s_k} splits ε_{s_k} into $\varepsilon_{s_k} - \varepsilon_{s_{k-1}}$ and $\varepsilon_{s_{k-1}}$, where $\varepsilon_{s_0} = 0$.
3. For each $i \in \{1, 2, \ldots, K\}$, the parties $P_{s_i}, P_{s_2}, \ldots, P_{s_K}$ calculate the noisy marginal distribution of $\bigcup_{k=i}^{K} X_{s_k}$ under $\left(\varepsilon_{s_i} - \varepsilon_{s_{i-1}}\right)$-differential privacy.
4. The curator takes consistency post-processing on the above calculated marginal distributions to obtain a more accurate marginal distribution of $\bigcup_{k=1}^{K} X_k$.

Using the above privacy division composition strategy, such a method can make full use of the privacy budget of each party while satisfying personalized privacy requirements for different attributes according to the composition property of differential privacy. In the above process, the core is Step 4.

For ease of understanding, let us first consider two-party setting. There exist two parties P_1 and P_2, who hold D_1 with attribute set A_1 and D_2 with attribute set A_2, respectively. P_1 and P_2 want to calculate the marginal distribution of (X_1, X_2) while satisfying ε_1-differential privacy for D_1 and ε_2-differential privacy for D_2, where $X_1 \in A_1$ and $X_2 \in A_2$, the domain of X_1 and X_2 are assumed to be both $\{0, 1\}$, and $\varepsilon_1 < \varepsilon_2$. In addition, in order to show the advantages of the proposed method more intuitively, we choose Gaussian noise as

the added noise. This is because Gaussian distribution satisfies additivity. In Gaussian mechanism, the function satisfies (ε, δ)-differential privacy, if it injects a Gaussian noise with the mean $\mu = 0$ and standard deviation $\sigma \geq cs/\varepsilon$ into the output, where $c^2 > 2\ln(1.25/\delta)$, and s denotes the sensitivity of the function and $\delta \in (0, 1)$ denotes the relaxation factor.

At the beginning, we split ε_2 into ε_1 and $\varepsilon_2 - \varepsilon_1$. The parties P_1, P_2 and the curator calculate the marginal distribution $\Pr(X_1, X_2)$ over $D_1 \cup D_2$ and inject Gaussian noises with $\sigma = cs/\varepsilon_1$ into each item of $\Pr(X_1, X_2)$, the noisy results can be denoted as $p'_{00},\ p'_{10},\ p'_{01},\ p'_{11}$, respectively. Given the calculated $p'_{00},\ p'_{10},\ p'_{01},\ p'_{11}$, the curator can calculate $\Pr(X_2 = 0) = p'_{00} + p'_{10} = p'_0$ and $\Pr(X_2 = 1) = p'_{01} + p'_{11} = p'_1$. Besides, the party P_2 calculates the marginal distribution $\Pr(X_2)$ over D_2 and injects Gaussian noise with $\sigma = cs/(\varepsilon_2 - \varepsilon_1)$ into each item of $\Pr(X_2)$. The noisy results are denoted as $p''_0,\ p''_1$, respectively. Thus, we have that, for the attribute X_2, we get two noisy marginals. In reality, there can only exist one marginal distribution of X_2. We recalculate the marginal distribution of X_2 by employing consistency post-processing and learn that:

$$\tilde{p}_0 = \frac{(cs/(\varepsilon_2 - \varepsilon_1))^2 \cdot p'_0 + 2(cs/\varepsilon_1)^2 \cdot p''_0}{2(cs/\varepsilon_1)^2 + (cs/(\varepsilon_2 - \varepsilon_1))^2}, \tilde{p}_1 = \frac{(cs/(\varepsilon_2 - \varepsilon_1))^2 \cdot p'_1 + 2(cs/\varepsilon_1)^2 \cdot p''_1}{2(cs/\varepsilon_1)^2 + (cs/(\varepsilon_2 - \varepsilon_1))^2}$$

As $\tilde{p}_0 + \tilde{p}_1 = 1$, $(\tilde{p}_0, \tilde{p}_1)$ can be an estimated marginal of X_2. Further more, based on the reconstructed of $\Pr(X_2)$, i.e., $\tilde{p}_0$ and $\tilde{p}_1$, we can reconstruct $\Pr(X_1, X_2)$:

$$\tilde{p}_{00} = \tilde{p}_0 \cdot \frac{p'_{00}}{p'_0}, \tilde{p}_{10} = \tilde{p}_0 \cdot \frac{p'_{10}}{p'_0}, \tilde{p}_{01} = \tilde{p}_1 \cdot \frac{p'_{01}}{p'_1}, \tilde{p}_{11} = \tilde{p}_1 \cdot \frac{p'_{11}}{p'_1}.$$

The above conclusions can be extended to the multi-party setting , where there exist K parties, where $K \geq 3$. Specifically, after Step 3, the curator can obtain noisy marginals, $\Pr\left(\bigcup_{j=1}^{K} X_{s_j} | \bowtie_{j=1}^{K} D_{s_j}\right)$, $\Pr\left(\bigcup_{j=2}^{K} X_{s_j} | \bowtie_{j=2}^{K} D_{s_j}\right)$, …, $\Pr(X_{s_K} | D_{s_K})$. Based on each of these marginals, the curator can calculate:

$$\omega_{ki} = \begin{cases} \sum\limits_{X_{s_k}} \cdots \sum\limits_{X_{s_{i-1}}} \Pr\left(\bigcup_{j=k}^{K} X_{s_j} | \bowtie_{j=k}^{K} D_{s_j}\right), & k < i \leq K \\ \Pr\left(\bigcup_{j=k}^{K} X_{s_j} | \bowtie_{j=k}^{K} D_{s_j}\right), & i = k \end{cases} \tag{3}$$

$$\beta_{ki} = \begin{cases} \prod_{j=k}^{i-1} \left|\Omega_{X_{s_j}}\right|, & k < i \leq K \\ 1, & i = k \end{cases} \tag{4}$$

At this time, for each $i \in \{1, \ldots, K\}$, the curator gets multiple noisy marginal distributions of attribute set $\bigcup_{j=i}^{K} X_{s_j}$, i.e., $w_{1i}, \ldots, w_{ii}$. Based on these results, the curator can calculate:

$$\Pr\left(\widetilde{\bigcup_{j=i}^{K}} X_{s_j}\right) = \left(\sum\nolimits_{k=1}^{i} \frac{\omega_{ki}}{\beta_{ki}\sigma_k^2}\right) / \left(\sum\nolimits_{k=1}^{i} \frac{1}{\beta_{ki}\sigma_k^2}\right). \tag{5}$$

where $\sigma_k = cs/(\varepsilon_k - \varepsilon_{k-1})$. Furthermore, for each $k \in \{1, 2, \ \ldots, K-1\}$, the curator can iteratively calculate:

$$\Pr\left(\widehat{\bigcup_{i=k}^{K} X_{s_i}}\right) = \Pr\left(\widehat{\bigcup_{i=k+1}^{K} X_{s_i}}\right) \cdot \Pr\left(\widetilde{\bigcup_{i=k}^{K} X_{s_i}}\right) / \sum_{X_{s_k}} \Pr\left(\widetilde{\bigcup_{i=k}^{K} X_{s_i}}\right).$$

$\Pr\left(\widehat{\bigcup_{i=1}^{K} X_{s_i}}\right)$ is the final noisy marginal of the attribute set $\mathcal{X} = \bigcup_{i=1}^{K} X_{s_k}$.

6.3 Implementation Details

We first consider the problem in the horizontally partitioned setting. After stretching, the parties and the curator privately calculate the marginal distribution of a given attribute set $\mathcal{X} \subseteq \mathcal{A}$ over the stretched datasets by using the threshold Homomorphic encryption [9]. In particular, for each $\mathcal{X} = x_i$, where $x_i \in \Omega_X$ and $1 \leq i \leq |\Omega_{\mathcal{X}}|$, the parties first jointly generate a Laplace noise η_i with scale $\lambda = \frac{2}{\varepsilon_{\max}}$ by employing the Distributed Laplace Noise Generation (DLNG) method proposed in [31]. DLNG can allow the parties jointly generate a Laplace noise η_i while preventing any parties and the curator from learning the value of η_i and facilitate subsequent calculation. Specifically, DLNG randomly divides η_i into K parts and shared among the parties, i.e., $\eta_i = \sum_{k=1}^{K} \eta_{ik}$, and $P_1, \ldots, P_K$ hold $\eta_{i1}, \ldots, \eta_{iK}$, respectively. In [31], it has be proven that the randomness of each η_{ik} is greater than η_i and the privacy η_i cannot be violated even when there exist some (even $K-1$) colluding parties. Then, each party P_k locally counts the number of tuples that have $\mathcal{X} = x_i$, which can be referred to as $\widetilde{c_{ik}}$. Next, P_k calculates $\widetilde{c_{ik}} + \eta_{ik}$ and sends it to the curator. After that, the curator calculates $\widetilde{c(x_i)} = \sum_{k=1}^{K} (\widetilde{c_{ik}} + \eta_{ik}) = \sum_{k=1}^{K} \widetilde{c_{ik}} + \eta_i$. Based on the above results, the curator can construct the likelihood function and solve the model parameters, and then calculate the noisy marginal distribution of $\mathcal{X}$ over the original datasets.

Calculating the marginal distribution in the vertically partitioned setting is rather complicated because the attributes are in different local datasets. We need some other security protocols to solve the problem, e.g., the secure scalar product protocol [12]. In particular, after privacy budget sort and segmentation, for each $i \in \{1, 2, \ldots, K\}$, the parties $P_{s_i}, \ldots, P_{s_K}$ jointly calculate the marginal distribution of $\mathcal{X} = \bigcup_{k=i}^{K} \mathcal{X}_{s_k}$ over their local datasets, while satisfying $\left(\varepsilon_{s_i} - \varepsilon_{s_{i-1}}\right)$-differential privacy, where $\mathcal{X}_{s_k} \subseteq \mathcal{A}_{s_k}$.

At the beginning, each party P_{s_k} first locally generates a vector $v_{s_k} = \left\{v_{s_k 1}, \ldots, v_{s_k |D_{s_k}|}\right\}$ with length $|D_{s_k}|$ for each $\mathcal{X}_{s_k} = x_{s_k}$, where $|D_{s_k}|$ denotes the size of the local dataset D_{s_k}. Note that, all of the local datasets have the same size, which can be referred to as $|D|$. Each element $v_{s_k j}$ in v_{s_k} is 1 if $X_{s_k} = x_{s_k}$ in the t^{th} tuple of D_{s_k}, otherwise $v_{s_k j} = 0$. Then, the parties calculate the number of tuples that have $\bigcup_{k=i}^{K} X_{s_k} = (x_{s_i}, \ldots, x_{s_K})$ by computing $\sum_{j=1}^{|D|} \prod_{k=i}^{K} v_{s_k j}$ in a secure way, and divide the result into K parts $r_1, \ldots, r_K$, and share among the parties. Next, by employing DLNG, the parties generate a Laplace noise

η with scale $\lambda = \frac{2}{\varepsilon_{s_i} - \varepsilon_{s_{i-1}}}$, and divide η into K parts $\eta_{i1}, \ldots, \eta_{iK}$ and shared among the parties. After that, each party sends $r_1 + \eta_{i1}$ to the curator, and the curator calculates $\widetilde{c(x)} = \sum_{k=1}^{K} (r_k(x) + \eta_k) = \sum_{k=1}^{K} r_k(x) + \eta$. Finally, based on $\widetilde{c(x)}$'s, the curator can calculate:

$$\widetilde{\Pr(x)} = \widetilde{c(x)} / \sum\nolimits_{x' \in \Omega_{\mathcal{X}}} \widetilde{c(x')}.$$

6.4 Privacy Analysis

Combining secure multiparty computation with differential privacy, we can guarantee that both the mixture-of-multinomials-based method and the privacy budget segmentation method satisfy ε_k-multi-party personalized differential privacy for each local dataset D_k.

7 Experiments

7.1 Experimental Settings

Datasets . In our experiments, we use two real datasets, *NLTCS*[†] and *BR2000*[‡]. NLTCS contains records of 21,574 individuals who participated in the National Long Term Care Survey. BR2000 consists of 38,000 census records collected from Brazil in the year 2000. Each of the two datasets contains both continuous and categorical attributes. For each continuous attribute, we discretize its domain into a fixed number b of equi-width ranges (we use $d = 16$).

To simulate the horizontally partitioned setting, we employ two categories of sampling methods (i.e., uniform sampling and non-uniform sampling) on the input dataset to obtain multiple local datasets that follow identically distributed, and that not. In addition, the size of each local dataset can be flexibly set. To simulate the vertically partitioned setting, we vertically partition the attributes among different parties randomly. We observe similar trends under different random partitionings.

Competitors . We first demonstrate the utility of the mixture-of-multinomials-based approach (denoted *MM*) for the horizontally partitioned setting by comparing it with four main approaches:

- **Independent.** The parties first add different noise levels to the local marginals independently according to their privacy preferences before sharing it with the curator. Then the curator aggregates these noisy local marginals together.
- **Minimum (MH).** The parties and the curator jointly calculate marginals over the original datasets using ε_{min} as the privacy parameter, where $\varepsilon_{min} = \min\{\varepsilon_k | 1 \leq k \leq K\}$ and ε_k denotes the privacy preference specified by P_k.

[†] http://lib.stat.cmu.edu/.
[‡] https://international.ipums.org.

- **Sampling-based method (SAH).** It works by first sampling D_k with probability $\frac{e^{\varepsilon_k}-1}{e^{\varepsilon_{max}}-1}$ and then calculating marginals over the sampled datasets using ε_{max} as the privacy parameter, where $\varepsilon_{max} = \max\{\varepsilon_k | 1 \leq k \leq K\}$.
- **Stretching-based method (STH).** It works by first multiplying the value of each tuple in D_k by a scaling factor $\frac{\varepsilon_k}{\varepsilon_{max}}$ and then calculating over the stretched datasets using ε_{max} as the privacy parameter.
- **Product.** In the Product method, the attributes are assumed to be independent and the k-way marginal is estimated with the product of k 1-way marginals.

Then, we evaluate the utility of the privacy budget segmentation method (denoted *PBS*) for the vertically partitioned setting by comparing it with Minimum and sampling-based methods for the vertically partitioned setting (denoted as *MV* and *SAV*, respectively).

Metrics . To measure the accuracy of a noisy marginal obtained by each method, we calculate the *total variation distance* [33] between the noisy marginal and its noise-free version, i.e., half of the L_1 distance between the two distributions. For each task, we repeat the experiment 100 times and report the average.

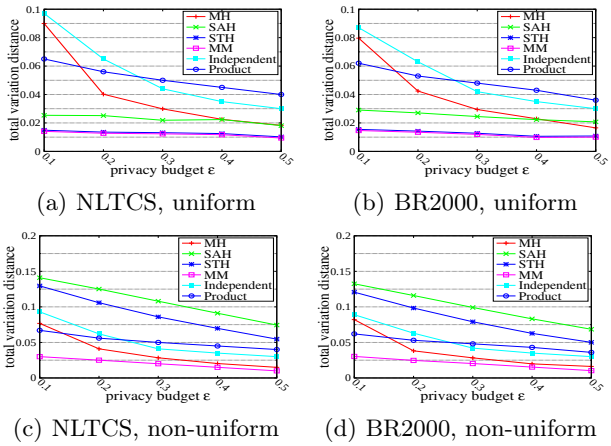
Parameters . There are three key parameters involved in our solutions:

- **Privacy preferences.** Following the setting in [18], we provide three kinds of privacy preferences for the parties i.e., ε_{min}, ε_{mid}, and ε_{max}. In particular, ε_{max} is always set to be 1, ε_{min} varies from 0.1 to 0.5, and ε_{mid} is set to be $\varepsilon_{mid} = \frac{\varepsilon_{min}+\varepsilon_{max}}{2}$.
- **Number of parties.** We let the number of parties vary from 2 to 10. According to the privacy preferences, the parties can be divided into three groups that with privacy preferences ε_{min}, ε_{mid}, and ε_{max}, respectively.
- **Fraction of users.** In the horizontally partitioned setting, the fraction of users that choose different privacy preferences will affect the utility of our solutions. We denote the fraction of each group to be f_{min}, f_{mid}, and f_{max}. The fraction can be set based on findings from several studies regarding user privacy attitudes (e.g., [1]). In particular, f_{mid} is always set to be 0.4, f_{min} varies from 0.1 to 0.5, and f_{max} is set to be $f_{max} = 1 - f_{min} + f_{mid}$.

Computing environment setup . All methods were implemented in Python. All methods were evaluated in a distributed environment using a cluster of nodes, which are connected by a 100Mbit network. Each node with an Intel Core i5-8300H processor and 16 GB of memory acts as either a curator or a party. The number of curator is 1, and the number of parties is up to 10.

7.2 Utility of Methods in the Horizontally Partitioned Setting

We first evaluate the impact of differences between private preferences by varying the value of the minimum privacy preference ε_{min}, where $\varepsilon_{max} = 1$ and



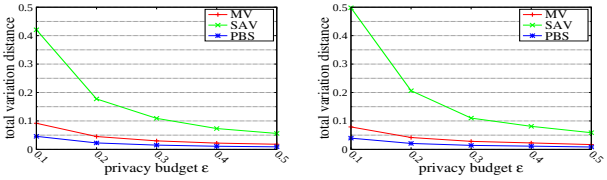
(a) NLTCS, uniform (b) BR2000, uniform

(c) NLTCS, non-uniform (d) BR2000, non-uniform

Fig. 1: Utility of methods in the horizontally partitioned setting.

$\varepsilon_{mid} = \frac{\varepsilon_{min}+\varepsilon_{max}}{2}$. Besides, the fraction of the group with privacy preference $\varepsilon_{max}/\varepsilon_{mid}/\varepsilon_{min}$ is set to be $f_{max} = 0.3$, $f_{mid} = 0.4$, $f_{min} = 0.3$, respectively.

Fig. 1 shows the effects of differences between private preferences on each approach over NLTCS and BR2000, where the marginals are 3-way marginals. Figs. 1(a)-1(b) show the total variation distance of each method when the local datasets follow identically distributed, and Figs. 1(c)-1(d) present the total variation distance of of each approach when the local datasets follow non-identically distributed. MM can always obtain the utility better than the others in all experiments. In particular, MM can obtain the utility better than Independent and Product. This is because Independent adds multiple shares of noise into the global marginals, reducing the utility of the results. In Product, it is assumed that the attributes are independent, this will incur a lot of precision loss. When the local datasets follow identically distributed, STH can obtain the utility as well as MM. But when the local datasets do not follow identically distributed, MM can obtain the utility better than STH. The reason lies in that, when the local datasets do not follow identically distributed, the calculated results in STH will be distorted. While by employing the Expectation-Maximization (EM) algorithm [10], MM can reconstruct the marginal distributions without distortion. Note that, ε_{max} is fixed at 1, ε_{mid} is set to be $\varepsilon_{mid} = \frac{\varepsilon_{min}+\varepsilon_{max}}{2}$. As ε_{min} increases, the difference between ε_{max}, ε_{mid}, and ε_{min} gradually narrows. The performance of MM, STH, and MH is all progressively close to that of the strategy that injects noise into the marginals according to a unified privacy budget. However, with the increase in the difference between private preferences, the superiority of the MM becomes more apparent, and MM can always obtain the utility better than the others. This is because, as the difference between private preferences increases, STH will stretch the data in the datasets that with the privacy preference ε_{min} by a smaller scaling factor. This will incur more distortion. And MH sets high-level global privacy to satisfy all the local datasets.



(a) NLTCS (b) BR2000

Fig. 2: Utility of methods in the vertically partitioned setting.

This will introduce a large amount of noise into the marginals, resulting in poor utility.

We also evaluate the impact of group fractions by varying the fraction f_{min} of the local dataset, and find that, MM can obtain the utility better than the others, and with the increase in f_{min}, the superiority of the MM becomes more apparent. Due to the space limitation, we do not show the experiments.

7.3 Utility of Methods in the Vertically Partitioned Setting

In the vertically partitioned setting, we compare PBS with MV and SAV. We evaluate the impact of differences between private preferences by varying the value of the minimum privacy preference ε_{min}, where $\varepsilon_{max} = 1$ and $\varepsilon_{mid} = \frac{\varepsilon_{min}+\varepsilon_{max}}{2}$. Note that, all of the local datasets are over the same set of individuals, we need not consider the impact of group fractions in the vertically partitioned setting. In addition, to conveniently evaluate the performance of each method, in this experiment, we select one attribute from each local dataset and calculate 3-way marginals. Generally, for a given attribute set $\mathcal{X}$, if there exist multiple attributes contained in the local dataset D_k, denoted $X_1^k, \ldots, X_l^k$, then these attributes can be treated as a new attribute, whose domain is $\Omega_{X_1^k} \times \cdots \times \Omega_{X_l^k}$, where $\Omega_{X_i^k}$ denotes the domain of one attribute X_i^k.

Fig. 2 shows the effects of differences between private preferences on each approach over NLTCS and BR2000. In all experiments, PBS can obtain the utility better than the others. The reason lies in that, by the privacy budget segmentation and composition, the PBS method can make full use of the privacy budget of each party. In addition, it is interesting that the sapling-based method SAV performs even worse than the Minimum method MV. This confirms our analysis in Section 5.2, i.e., the sampling-based method leads to a sparse sampled global dataset, which will reduce the utility of the calculated marginal distributions.

8 Conclusions

In this paper, we studied the problem of multi-party marginal distribution calculation under personalized differential privacy. We proposed the mixture-of-multinomials-based approach for the horizontally partitioned setting and the

privacy budget segmentation method for the vertically partitioned setting. We formally proved that these approaches guarantee multi-party personalized differential privacy for each local dataset. Extensive experiments on real datasets demonstrated that our solution offers high data utility.

Acknowledgments

The work was supported by the National Key R&D Program of China under Grant No. 2020YFB1710200, National Natural Science Foundation of China under Grant No. 62002203, No. 61872105, No. 62072136, Shandong Provincial Natural Science Foundation No. ZR2020QF045, No. ZR2020MF055, No. ZR2021LZH007, No. ZR2020LZH002, the New Engineering Disciplines Research and Practice Project under Grant No. E-JSJRJ20201314, and Young Scholars Program of Shandong University.

References

1. Acquisti, A., Grossklags, J.: Privacy and rationality in individual decision making. S&P **3**(1), 26–33 (2005)
2. Alaggan, M., Gambs, S., Kermarrec, A.: Heterogeneous differential privacy. J. Priv. Confidentiality **7**(2) (2016)
3. Alhadidi, D., Mohammed, N., Fung, B.C.M., Debbabi, M.: Secure distributed framework for achieving ε-differential privacy. In: PETS (2012)
4. Bater, J., He, X., Ehrich, W., Machanavajjhala, A., Rogers, J.: Shrinkwrap: Efficient SQL query processing in differentially private data federations. VLDB **12**(3), 307–320 (2018)
5. Beimel, A., Nissim, K., Omri, E.: Distributed private data analysis: Simultaneously solving how and what. In: CRYPTO (2008)
6. Chen, R., Li, H., Qin, A.K., Kasiviswanathan, S.P., Jin, H.: Private spatial data aggregation in the local setting. In: ICDE (2016)
7. Chen, R., Reznichenko, A., Francis, P., Gehrke, J.: Towards statistical queries over distributed private user data. In: NSDI (2012)
8. Cheng, X., Tang, P., Su, S., Chen, R., Wu, Z., Zhu, B.: Multi-party high-dimensional data publishing under differential privacy. TKDE **32**(8), 1557–1571 (2020)
9. Cramer, R., Damgård, I., Nielsen, J.B.: Multiparty computation from threshold homomorphic encryption. In: EUROCRYPT (2001)
10. Do, C.B., Batzoglou, S.: What is the expectation maximization algorithm? Nature Biotechnology **26**, 897–899 (2008)
11. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: TCC (2006)
12. Goethals, B., Laur, S., Lipmaa, H., Mielikäinen, T.: On private scalar product computation for privacy-preserving data mining. In: ICISC (2004)
13. Goryczka, S., Xiong, L.: A comprehensive comparison of multiparty secure additions with differential privacy. TDSC **14**(5), 463–477 (2017)
14. Gu, X., Li, M., Xiong, L., Cao, Y.: Providing input-discriminative protection for local differential privacy. In: ICDE (2020)

15. Hardt, M., Nath, S.: Privacy-aware personalization for mobile advertising. In: CCS (2012)
16. Hong, D., Jung, W., Shim, K.: Collecting geospatial data with local differential privacy for personalized services. In: ICDE (2021)
17. Jiang, W., Clifton, C.: A secure distributed framework for achieving k-anonymity. VLDB J. **15**(4), 316–333 (2006)
18. Jorgensen, Z., Yu, T., Cormode, G.: Conservative or liberal? personalized differential privacy. In: ICDE (2015)
19. Kasiviswanathan, S.P., Lee, H.K., Nissim, K., Raskhodnikova, S., Smith, A.D.: What can we learn privately? In: FOCS (2008)
20. Kotsogiannis, I., Doudalis, S., Haney, S., Machanavajjhala, A., Mehrotra, S.: One-sided differential privacy. In: ICDE (2020)
21. McSherry, F., Talwar, K.: Mechanism design via differential privacy. In: FOCS (2007)
22. Mironov, I., Pandey, O., Reingold, O., Vadhan, S.P.: Computational differential privacy. In: CRYPTO (2009)
23. Mohammed, N., Alhadidi, D., Fung, B.C.M., Debbabi, M.: Secure two-party differentially private data release for vertically partitioned data. TDSC **11**(1), 59–71 (2014)
24. Mohammed, N., Fung, B.C.M., Debbabi, M.: Anonymity meets game theory: secure data integration with malicious participants. VLDB J. **20**(4), 567–588 (2011)
25. Narayan, A., Haeberlen, A.: Djoin: Differentially private join queries over distributed databases. In: OSDI (2012)
26. Nie, Y., Yang, W., Huang, L., Xie, X., Zhao, Z., Wang, S.: A utility-optimized framework for personalized private histogram estimation. TKDE **31**(4), 655–669 (2019)
27. Niu, B., Chen, Y., Wang, B., Cao, J., Li, F.: Utility-aware exponential mechanism for personalized differential privacy. In: WCNC (2020)
28. Qardaji, W.H., Yang, W., Li, N.: Priview: practical differentially private release of marginal contingency tables. In: SIGMOD (2014)
29. Song, H., Luo, T., Wang, X., Li, J.: Multiple sensitive values-oriented personalized privacy preservation based on randomized response. TIFS **15**, 2209–2224 (2020)
30. Su, S., Tang, P., Cheng, X., Chen, R., Wu, Z.: Differentially private multi-party high-dimensional data publishing. In: ICDE (2016)
31. Tang, P., Chen, R., Su, S., Guo, S., Ju, L., Liu, G.: Differentially private publication of multi-party sequential data. In: ICDE (2021)
32. Tang, P., Cheng, X., Su, S., Chen, R., Shao, H.: Differentially private publication of vertically partitioned data. TDSC **18**(2), 780–795 (2021)
33. Tsybakov, A.B.: Introduction to Nonparametric Estimation. Springer (2009)
34. Wagh, S., He, X., Machanavajjhala, A., Mittal, P.: Dp-cryptography: marrying differential privacy and cryptography in emerging applications. Commun. ACM **64**(2), 84–93 (2021)
35. Wu, D., Wu, X., Gao, J., Ji, G., Xu, X., Qi, L., Dou, W.: A personalized preservation mechanism satisfying local differential privacy in location-based services. In: SPDE (2020)
36. Xiao, X., Tao, Y.: Personalized privacy preservation. In: SIGMOD (2006)
37. Xue, Q., Zhu, Y., Wang, J.: Mean estimation over numeric data with personalized local differential privacy. FCSC **16**(3), 163806 (2022)