Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2015, Article ID 259682, 18 pages
http://dx.doi.org/10.1155/2015/259682

Review Article

A Review of Differential Privacy in Individual Data Release

Jun Wang,[1,2] Shubo Liu,[1,2] and Yongkai Li[1,2]

[1]*School of Computer, Wuhan University, Wuhan 430072, China*
[2]*Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, Wuhan University, Wuhan 430072, China*

Correspondence should be addressed to Shubo Liu; liu.shubo@whu.edu.cn

Received 25 June 2015; Accepted 8 September 2015

Academic Editor: Chien-Lung Hsu

The rapid development of mobile technology has improved users' quality of treatment, and tremendous amounts of medical information are readily available and widely used in data analysis and application, which bring on serious threats to users' privacy. Classical methods based on cryptography and anonymous-series models fail due to their high complexity, poor controllability, and dependence on the background knowledge of adversaries when it comes to current mobile healthcare applications. Differential privacy is a relatively new notion of privacy and has become the de facto standard for a security-controlled privacy guarantee. In this paper, the key aspects of basic concepts and implementation mechanisms related to differential privacy are explained, and the existing research results are concluded. The research results presented include methods based on histograms, tree structures, time series, graphs, and frequent pattern mining data release methods. Finally, shortcomings of existing methods and suggested directions for future research are presented.

1. Introduction

With the further development of mobile technology, it is easy to attain, store, and analyze mobile healthcare information, such as physiological data, social interactions, consumer record, and track data, which all inevitably involve user's privacy. MHR (mobile healthcare record) has been questioned because of user's privacy disclosure since its inception due to the high sensitivity of health data [1, 2]. In 2015, MIT researchers (de Montjoye et al. [3]) reported that the so-called unimportant dates and locations of only four pieces of consumption records are enough to identify 90 percent of the people in a dataset recording three months of credit-card transactions by 1.1 million users. Based on concerns about user's privacy in mobile healthcare, researchers propose a series of privacy protection schemes mainly including data distortion, data encryption, and restrictive release [4].

The anonymity models based on restrictive release were proposed greatly to guarantee user's privacy in healthcare application [5], such as k-anonymity, l-diversity, and t-closeness. k-anonymity was first introduced by Sweeney and Samarati [6, 7] and it has the property that each record is indistinguishable from at least $k - 1$ records; however, it may disclose privacy information [8, 9], so l-diversity and t-closeness were proposed to enhance privacy security. But all these methods fail to offer quantifiable guarantee for user's privacy and are short of demarcation clearly for adversary's ability.

Avancha et al. [10] presented the privacy problems in mHealth systems and gave a conceptual privacy framework without specific technology; Francis et al. [11] proposed a privacy preserving model for the e-health system which uses the trusted authority to maintain the privacy of user's health records; Sadki and El Bakkali [12] proposed an integrated privacy module to protect user's privacy in mobile healthcare. However, existing privacy preserving models mainly depend on the background knowledge of adversaries who may harm the data by reidentifying it [1]. By reason of the dependence, the research of privacy protection falls into the strange circle; that is, a method is proposed to protect the privacy and is broken subsequently. Then, another method is proposed to solve this problem and is broken again, but differential privacy [13]

can protect the presence of an individual regardless of the adversary's background knowledge and give the demarcation of adversary's ability.

In 2006, differential privacy based on probability model was first proposed by Dwork. It could be classified as data distortion and uses random noise to ensure that the adversary fails to guess whether the tuple is present in the dataset with high probability even if he knows all the tuples except the current tuple, which is the theoretical upper limit of attacking capability of the adversary. Differential privacy is a strictly provable and security-controlled method and is introduced quickly and widely to other data application scenes since it was proposed in database domain [14–16]. The popularity of network applications [17, 18] will certainly promote the development of differential privacy.

Leoni [19] reviewed differential privacy, which contains noninteractive processing and is mainly classified under technical level, such as dimensionality reduction and filtering; Xiong et al. [20] gave a survey on differential privacy and applications and neglected time series and graph data release methods; Yang et al. [21] described the classification of differential privacy; however, the detailed statement is insufficient. As the classification of many existing surveys is incomplete and its account is not quite clear, this paper is going to summarize the state of the art in differential privacy research from data level, mainly including a variety of categories, histogram, tree structure, time series, graph, and frequent pattern mining data release methods, and then to discuss open problems by analysis and comparison of existing methods.

The rest of this paper is organized as follows. Section 2 provides the preliminaries on differential privacy and implementation mechanism. Section 3 introduces data processing models and releasing strategies of differential privacy. Section 4 presents data release methods of differential privacy. Finally, it comes to conclude the paper and presents research directions for future work in Section 5.

2. Preliminaries

Differential privacy is a relatively new notion of privacy and is one of the most popular privacy concepts as well. Starting from the basic concept of privacy, this section introduces the notion, implementation mechanism, and two important properties of differential privacy.

2.1. Base Definitions. Privacy is most often defined as the right to be left alone, free from intrusion or interruption. It is an umbrella term, encompassing elements such as physical privacy, communications privacy, and information privacy [22].

Intuitionally, differential privacy ensures that the removal or addition of a single database tuple does not affect the outcome of any analysis [23]. The definition of differential privacy is as follows.

Definition 1 (differential privacy [13]). Let D and D' be two neighboring datasets which differ on at most one tuple, denoted by $d(D, D') = 1$; given a randomized function $A : D \to R$, Range(A) is the set of all possible outputs of A in D and D', and given an arbitrary subset $S \subseteq$ Range(A), A is said to be ε-differential privacy if

$$\Pr[A(D) \in S] \leq \exp(\varepsilon) * \Pr\left[A\left(D'\right) \in S\right]. \quad (1)$$

The parameter ε is called privacy budget, which is used to control the ratio of output of function A in neighboring datasets D and D'. The smaller ε is, the more close ratio is to 1; that is to say, the output probability distributions of function A in neighboring datasets D and D' are approximately the same; meanwhile, the higher the security becomes, the lower the utility gets. The value of ε is small usually, such as 0.01, 0.1, 0.5, and 0.8.

The strong privacy guarantee of differential privacy comes at the price of much noise added to the results of queries and analyses [21]. Dwork et al. further proposed enhancing the utility of data by reducing the requirements of data security; then (ε, δ)-differential privacy was proposed to this issue.

Definition 2 ((ε, δ)-differential privacy [24, 25]). Let D and D' be two neighboring datasets which differ on at most one tuple, denoted by $d(D, D') = 1$; given a randomized function $A : D \to R$, Range(A) is the set of all possible outputs of A in D and D', and given an arbitrary subset $S \subseteq$ Range(A), A is said to be (ε, δ)-differential privacy if

$$\Pr[A(D) \in S] \leq \exp(\varepsilon) * \Pr\left[A\left(D'\right) \in S\right] + \delta. \quad (2)$$

No hard rule dictates what value of privacy budget ε and δ should be and they can be set according to the actual application requirements. The value of δ is small usually ($\delta \leq 10^{-4}$) and (ε, δ)-differential privacy equals ε-differential privacy when $\delta = 0$.

The output probabilities of randomized function A in neighboring datasets D and D' for ε-differential privacy are shown in Figure 1 [26, 27], where $c \leq |A(D) - A(D')|$; the larger c is, the wider the distribution will be and the higher the probability of adding more noise becomes.

Function A provides privacy guarantee by adding random noise to the output; that is, $A(D) = f(D) + noise$, where f is such query operations as average, count, and sum and *noise* is the random noise following Laplace distribution which will be introduced in Section 2.2. The smaller the difference between two curves is, the stronger the security received is.

A simple attack example is as follows: Table 1 is a health dataset and provides count service $f(x) = \text{count}(\cdot)$ for querying the number of people suffering from HIV. Without privacy protection strategy, the adversary just needs to query the number of four people containing Alice with HIV which is Yes and the number of three people excluding Alice also with HIV is Yes; then the adversary obtains that Alice is suffering from HIV only by count(4) − count(3) = 1. Obviously, practical application for health data release comes to failure if it fails to resist these attacks.

Now, we add a random noise to the output of count function f and let two count results fall into the same range with higher probability; for example, count(3) = $\{1, 1.5, 2\}$ and count(4) = $\{1, 1.5, 2\}$; by this time, the adversary fails

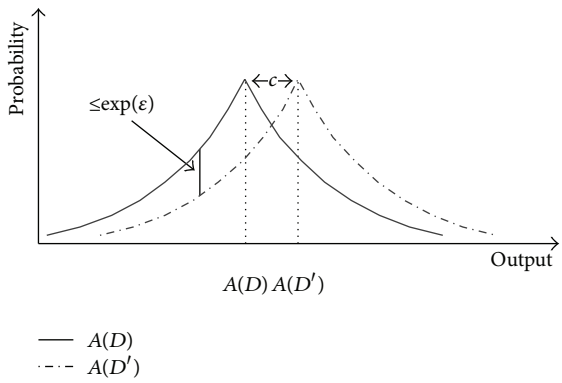


Figure 1: Output probabilities of randomized function A in neighboring datasets D and D'.

Table 1: Health records.

Name	HIV (0 is No; 1 is Yes)	Address
Tim	0	New York
Bob	1	Washington
Tony	0	New York
Alice	1	Hawaii
⋮	⋮	⋮

to get Alice's health information by the above method as both count(4) and count(3) may output 1.5 and count(4) − count(3) = 0. Differential privacy gives the maximum capacity of the adversary in this instance; that is to say, the adversary knows statistics whether the Alice is present in the dataset or not.

2.2. Noise Mechanism. The two main noise mechanisms are Laplace mechanism and exponential mechanism, and the noise magnitude refers to privacy budget and global sensitivity.

Definition 3 (global sensitivity [13]). Given a randomized function $f : D \to R^d$, the global sensitivity of function f is

$$\Delta f = \max_{D,D'} \left\| f(D) - f\left(D'\right) \right\|_1, \tag{3}$$

where D and D' are two neighboring datasets, R is the real space of mapping, d is the dimension, and $\|f(D) - f(D')\|_1$ is the first-order norm distance between $f(D)$ and $f(D')$ and is denoted by $L_1 = \|f(D) - f(D')\|_1$.

The global sensitivity for count function $\Delta f = 1$; however, Δf is larger for other operations. For instance, given dataset $D = \{1, 1, 1, 1, 21\}$, f is sum operation; then the global sensitivity $\Delta f = 21$ is larger than the sensitivity of count function. So Nissim et al. first proposed the notion of local sensitivity to reduce the sensitivity magnitude as the noise will become large with the increasing of global sensitivity.

Definition 4 (local sensitivity [41, 42]). Given a randomized function $f : D \to R^d$, the local sensitivity LS_f of function f is

$$\mathrm{LS}_f = \max_{D'} L_1. \tag{4}$$

For example, given the subset $D_1 = \{1, 1, 1, 1\}$ of D, the local sensitivity $\mathrm{LS}_f = 1$ is smaller than the global sensitivity 21 under adding operation. Local sensitivity may reduce the noise magnitude, but noise magnitude might reveal information about the dataset; for details see [41, Example 1]. So Nissim et al. further proposed the notion of β-smooth sensitivity to avoid disclosing privacy information; now the added noise is proportional to a smooth upper bound on the local sensitivity.

Definition 5 (smooth sensitivity [41]). For $\beta > 0$, given a randomized function $f : D \to R^d$, the β-smooth sensitivity $S_{f,\beta}(D)$ of function f is

$$S_{f,\beta}(D) = \max_{D'} \left(\mathrm{LS}_f\left(D'\right) \cdot e^{-\beta d(D,D')} \right). \tag{5}$$

2.2.1. Laplace Mechanism. The literature [42] proposed Laplace mechanism to achieve differential privacy by adding random noise drawn from the Laplace distribution to the output. Its definition is as follows.

Definition 6 (Laplace mechanism). Given a dataset D and the function $f : D \to R^d$, global sensitivity is Δf; random algorithm $A(D) = f(D) + noise$ satisfies ε-differential privacy if the noise obeys the Laplace distribution; that is, $noise \sim \mathrm{Lap}(\Delta f/\varepsilon)$; thereinto, location parameter is 0 and scale parameter is $\Delta f/\varepsilon$.

Let $\mathrm{Lap}(b)$ denote the Laplace distribution when location parameter is 0 and scale parameter is b, and its probability density function is $p(x) = \exp(-|x|/b)/2b$. From Figure 2, the larger the noise added to the output is, the larger b is and, meanwhile, the smaller ε becomes.

Let $\sigma(x)$ denote standard deviation; $D(x)$ denotes variance, and $noise \sim \mathrm{Lap}(b)$, $\sigma(x) = \sqrt{D(x)}$, $D(x) = 2b^2$, and $b = \Delta f/\varepsilon$; then we obtain the results $D(x) = 2(\Delta f/\varepsilon)^2 = 2\Delta f^2/\varepsilon^2$, $\sigma(x) = \sqrt{D(x)} = \sqrt{2\Delta f^2/\varepsilon^2} = \sqrt{2}\Delta f/\varepsilon$.

2.2.2. Exponential Mechanism. Laplace mechanism is used when the output is numerical, and exponential mechanism is another security-controlled scheme to satisfy differential privacy when the outputs are nonnumerical.

Intuitionally, exponential mechanism still ensures that the change of a single database tuple does not affect the outcome of the score function, and the definition is as follows.

Definition 7 (exponential mechanism [43]). Let D denote the input dataset; $r \in R$ denotes a potential answer, given a score function $u : D \times R \to R$; if a random algorithm A selects an answer based on the following probability, then algorithm A is said to satisfy ε-differential privacy:

$$A(D, u) = \left\{ r : \mid \Pr[r \in R] \propto \exp\left(\frac{\varepsilon u(D, r)}{2\Delta u} \right) \right\}, \tag{6}$$

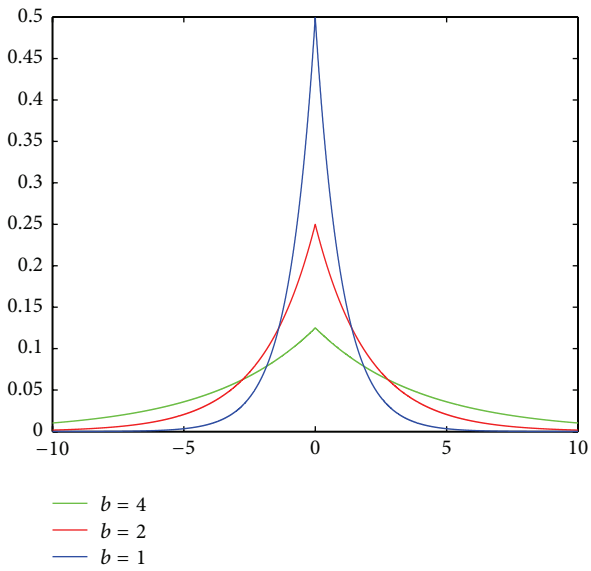


Figure 2: Probability density function.

where Δu denotes the sensitivity of score function u and is defined as

$$\Delta u = \max_{r \in R} \max_{\|D \Delta D'\| = 1} \left| u(D, r) - u\left(D', r\right) \right|. \quad (7)$$

The exponential mechanism can output nonnumerical results according to their values of score function. The output probability refers to privacy budget from the definition above, and the highest scored result is outputted with higher probability when ε is larger; meanwhile, when the difference between the output probabilities grows, the security becomes less; vice versa, the smaller ε is, the higher the security will be.

2.3. Combination Properties. Differential privacy has two important properties [44], as follows.

Property 1 (sequential composition). Set $A_1, A_2, \ldots, A_n$ satisfying differential privacy for $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$; multiple mechanism $A(A_1, A_2, \ldots, A_n)$ satisfies $\sum_{i=1}^{n} \varepsilon_i$-differential privacy for the same dataset D.

Sequential composition shows that privacy budget and error are cumulative linearly when we use multiple differential privacy to release data for the same dataset.

Property 2 (parallel composition). Set $A_1, A_2, \ldots, A_n$ satisfying differential privacy for $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$; multiple mechanism $A(A_1, A_2, \ldots, A_n)$ satisfies $\max \varepsilon_i$-differential privacy for the different datasets $D_1, D_2, \ldots, D_n$.

Parallel composition shows that the degree of privacy guarantee depends on the value of ε_i; when the value of ε_i grows, the security decreases gradually.

These two properties play a major role in proving whether an algorithm satisfies differential privacy and judging whether the budgeting strategy is reasonable [4]. For instance, given an algorithm A with n-step, each step has budget ε_i ($i = 1, 2, \ldots, n$); then the algorithm A satisfies $\sum_{i=1}^{n} \varepsilon_i$-differential privacy.

Differential privacy has great advantages compared to traditional privacy protection model; however, there will be some sacrifice for utility of the data to satisfy differential privacy. From Definition 2, the level of similarity in query result of neighboring datasets depends on the value of ε: the smaller ε is, the higher the level of similarity of query results becomes and the stronger privacy guarantee for raw data will become; at the same time, smaller ε leads to larger noise and poor data utility, and vice versa. In short, utility and security of data are a pair of contradictions, so the aim of privacy guarantee is to ensure very high level of privacy while simultaneously providing extremely accurate information about the database [13].

In order to balance the security and the utility of the data, we need to consider how to meet the maximum number of queries using the smallest budget before exhausting ε or how to enhance the accuracy of query under giving ε. Existing methods usually use relative error [31], absolute error [31], variance [30, 38], standard deviation [31], and false negative [40] to evaluate the rationality of algorithm.

3. Data Process Models and Release Strategies

There are two data process models of differential privacy, namely, noninteractive model and interactive model; the latter is also called online query model as data requester is only allowed to access the information through an interface provided by data owner. Similarly, noninteractive model is called offline query model as data requester can directly access the sanitized dataset released by data owner [20, 42].

Interactive model is shown in Figure 3; data owner provides a data query algorithm based on differential privacy, and data requester sends a query request; when the query algorithm receives the request, it gets the raw data from the original database and does sanitizing process of raw data; then the sanitized data is returned to the requester. In this model, query number is restricted by privacy budget ε; the more query number leads to smaller budget for each query if total budget ε is a constant and a larger noise is added to the query result; in the end, the query result becomes inutile. So how to design the query algorithm to provide the maximum number of queries under limited budget ε is the key to this model.

Noninteractive model is shown in Figure 4; data owner is a trusted curator and releases a sanitized dataset, and data requester sends a query request; when the sanitized dataset receives the request, the noisy result is returned to the requester. In this model, query number is unrestricted by privacy budget ε, so how to design the release algorithm with high efficiency to enhance the accuracy of query is the key to this model.

Difference privacy research is based on the two above models and there are roughly three data release strategies as follows.

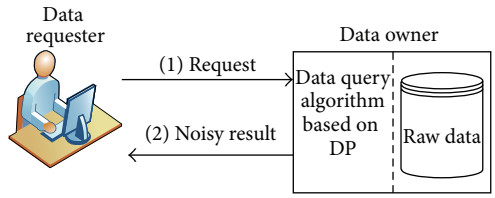


Figure 3: Interactive model.

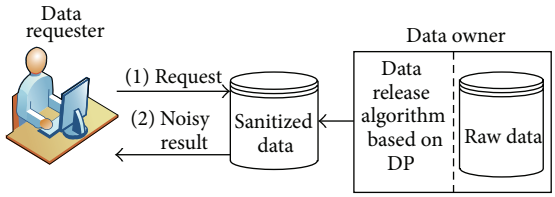


Figure 4: Noninteractive model.

Strategy 1 (see [45]). Noise $\mathrm{Lap}(\Delta f/\varepsilon)$ is added to raw data $D = \{x_1, x_2, \ldots, x_n\}$ and gets noisy data $\widetilde{D}$, which can be further optimized by postprocessing, for example, least square method, to enhance the query accuracy by the optimized data $\overline{D}$; then $\overline{D}$ is released. In this strategy, ε is usually uniform budgeting; its process is shown in Algorithm 1.

Algorithm 1.

Input. Raw data $D = \{x_1, x_2, \ldots, x_n\}$, privacy budget ε.

Output. Released data $\overline{D}$

(1) $\widetilde{D} = \{x_1 + \mathrm{Lap}(\Delta f/\varepsilon), x_2 + \mathrm{Lap}(\Delta f/\varepsilon), \ldots, x_n + \mathrm{Lap}(\Delta f/\varepsilon)\}$

(2) $\overline{D} = \mathrm{postprocess}(\widetilde{D})$.

Strategy 2 (see [45]). First, the conversion and the compression, which usually can reduce the sensitivity Δf of the function f, are applied to obtain a synthetic data $\overline{D}$; for example, the graph is converted into tree structure; then noise is added to $\overline{D}$ and noisy data $\widetilde{D}$ is gotten. In this strategy, ε is uniform budgeting; finally, $\widetilde{D}$ is released; the process of Strategy 2 is shown in Algorithm 2.

Algorithm 2.

Input. Raw data $D = \{x_1, x_2, \ldots, x_n\}$, privacy budget ε.

Output. Released data $\widetilde{D}$

(1) $\overline{D} = \mathrm{postprocess}(D)$

(2) $\widetilde{D} = \{\overline{x}_1 + \mathrm{Lap}(\Delta f/\varepsilon), \overline{x}_2 + \mathrm{Lap}(\Delta f/\varepsilon), \ldots, \overline{x}_n + \mathrm{Lap}(\Delta f/\varepsilon)\}$.

Strategy 3. Noise $\mathrm{Lap}(\Delta f/\varepsilon_i)$ is added to raw data $D = \{x_1, x_2, \ldots, x_n\}$ and gets noisy data $\widetilde{D}$; note that ε is nonuniform budgeting; that is, $\varepsilon_i \neq \varepsilon_j$; it uses the reasonable budgeting strategy to reduce the error of data query; the process of Strategy 3 is shown in Algorithm 3.

Algorithm 3.

Input. Raw data $D = \{x_1, x_2, \ldots, x_n\}$, privacy budget $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n, \varepsilon_i \neq \varepsilon_j$.

Output. Released data $\widetilde{D}$

(1) $\widetilde{D} = \{x_1 + \mathrm{Lap}(\Delta f/\varepsilon_1), x_2 + \mathrm{Lap}(\Delta f/\varepsilon_2), \ldots, x_n + \mathrm{Lap}(\Delta f/\varepsilon_n)\}$.

These three data release strategies can be used alone or in combination. For instance, given a data release method, Strategy 3 is first used to allocate the budget reasonably; then Strategy 1 is used to further enhance the query accuracy by postprocessing.

4. Data Release Categories

How to ensure the level of privacy guarantee while simultaneously providing highly utility of the data is a real concern; therefore, many methods were proposed in this section, and we will give our categories of differential privacy.

4.1. Histogram Release. Histogram is constructed by splitting the input dataset into mutually disjoint subsets called buckets or bins depends on a set of attributes. Any access to the raw data is conducted through the differential privacy interface when users send data query, and the differential privacy histogram is used to answer the user's queries; the process is shown in Figure 5 [46].

The most straightforward method is to add Laplace noise to each histogram bucket using privacy parameter ε, and the noise is $\mathrm{Lap}(1/\varepsilon)$ as count function sensitivity $\Delta f = 1$; this method is fit for unit length queries and short-range queries, and it will lead to larger query error under long-range (i.e., the number of bucket n is larger) queries according to noise variance formula $2n/\varepsilon^2$. In order to reduce the query error, multiple buckets are merged into one partition; now the number of tuples which fall into each partition is the average value of the number of tuples in these buckets, as shown in Figure 6.

The noise *noise* is added to each bucket before the merger, and the total noise is 7*noise*; after merge operation, the total noise becomes 3*noise*, which is smaller than the value before the merger. However, merge operation introduces an approximation error as the number of tuples in the partition is the average value of the number of tuples in multiple buckets.

We need to make the least possible number of partitions to reduce the noise error and make the number of tuples of buckets in a partition the same as much as possible to reduce the approximation error. In general, a finer-grained partitioning will introduce smaller approximation error but larger noise error, so finding the right balance between approximation error and noise error is a key question [46].

Xiao et al. [29, 46] proposed a partitioning strategy based on the *kd*-tree. It seeks to merge buckets which are close to uniform into a partition using a variance-like metric $U(p_0)$, which is defined as $U(p_0) = \sum_{c_i \in P_0} |x(c_i) - a_0|$, where p_0 is the current partition, c_i is the each bucket in p_0, $x(c_i)$ is

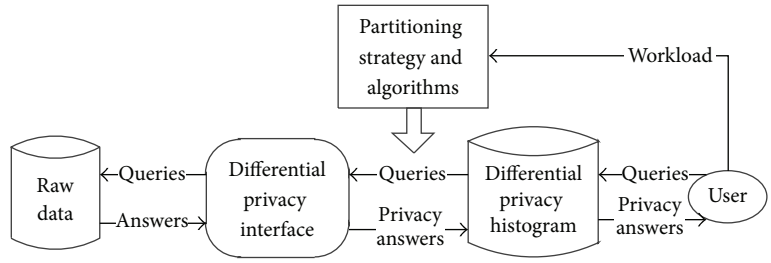


Figure 5: Different private histogram release.

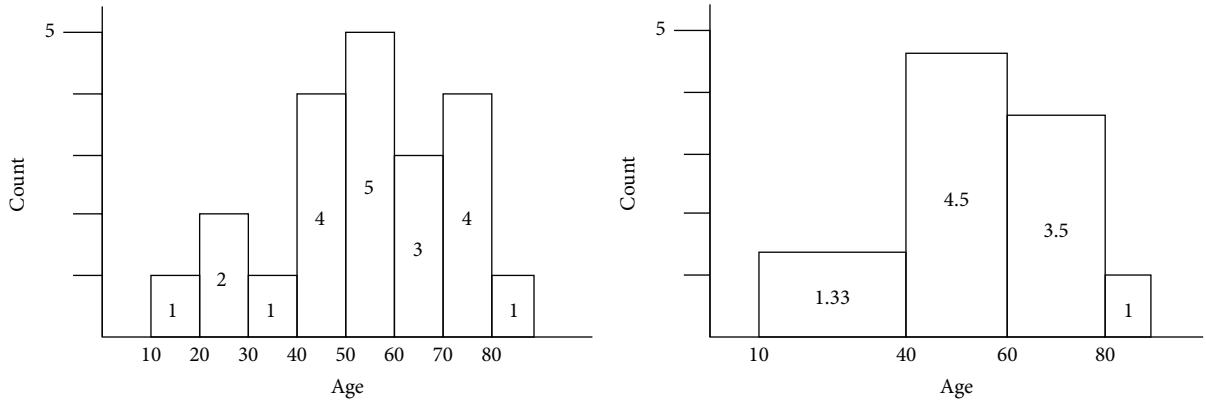


Figure 6: Example of buckets merger.

the number of tuples in c_i, and a_0 is the average value of buckets in p_0. If $U(p_0) > T$, where T is a giving threshold, then current partition p_0 is split into two partitions according to the split median of all the numbers of buckets in p_0. The process repeats until the whole domain is split into several partitions and each partition is close to uniform.

The process of kd-tree based histogram release is as follows. First, the original histogram is constructed according to the kth different attributes of raw data and the noise is added to original histogram using privacy budget $\varepsilon/2$ to generate a noisy k-dimensional dataset. Next, the noisy k-dimensional dataset is partitioned by kd-tree algorithm according to $U(P_0)$, getting final partitions. Then, the noise is added to each partition using the remaining privacy budget $\varepsilon/2$. Finally, the noisy histogram is released. The property of the algorithm is decided by the parameter $U(p_0)$ and the tree depth when partitioning or the number of tuples in each partition. kd-tree based histogram release algorithm is further used in health information DE-identification in Xiao et al.'s subsequent research and is named DPCube, which is based on Strategy 1 and kd-tree is used for postprocessing after adding noise. DPCube supports multidimensional data, long-range query and query error may be lower if the value of parameters selected properly.

In view of the right balance between the appropriation error and the noise error, Xu et al. [15] proposed the notion of Sum of Squared Error (SSE) in order to reduce the appropriation error and enhance the data query accuracy. SSE is defined as follows: $\mathrm{SSE}(H, D) = \sum_j \sum_{l_j \le i \le r_j} (x\mathrm{opt}_j - x_i)^2$, $D = \{x_1, x_2, \ldots, x_n\}$ is a series of counts of original histogram and x_i is the count of each bucket, $H = \{p_1, p_2, \ldots, p_k\}$ represents the partitioned histogram which merges neighboring counts into k partitions and p_j is the partition of the histogram, which contains an interval $[l_j, r_j] \subseteq [1, n]$, and $x\mathrm{opt}_j$ is the optimal value for p_j and is generally computed by the mean value of the counts in $[l_j, r_j]$; that is, $x\mathrm{opt}_j = \sum_{i=l_j}^{r_j} x_i/(r_j - l_j + 1)$.

Two algorithms NoiseFirst and StructureFirst are proposed dependent on SSE to measure the error between the histogram H and the original count data D. Strategy 1 based NoiseFirst constructs the optimal histogram adopting by dynamic programming method after adding noise and is fit for short-range query; Strategy 2 based StructureFirst generates the optimal histogram based on the original count sequence, which has the risk of privacy disclosure. StructureFirst first uses partly privacy budget ε_1 to guarantee the sensitive information in the histogram; then, the noise is added to the partitions using remaining privacy budget (ε-ε_1), and StructureFirst is fit for long-range query. StructureFirst and DPCube proposed by Xiao et al. are very similar; the maximal difference is that the two steps in DPCube separately allocated privacy budget $\varepsilon/2$, that is, uniform budgeting, while StructureFirst is a near-optimal budgeting strategy. The two above methods need to reconstruct the histogram; that is, the original histogram is reconstructed, and the number of partitions k is the key factor that affects the algorithm's performance.

In order to reduce the error in long-range query, Hay et al. [47] proposed optimizing histogram by hierarchical tree structure. Histogram is transformed into hierarchical tree

Table 2: Histogram release methods.

Method name	Strategy	Advantage	Defect	Budgeting
DPCube	Strategy 1	Supports multidimensional and long-range query	Parameter affects query accuracy heavily	Uniform budgeting
NoiseFirst	Strategy 1	Fit for short-range query	Histogram reconstruction is expensive	Uniform budgeting
Structure First	Strategy 2	Fit for long-range query	Histogram reconstruction is expensive	Near-optimal budgeting
Flat	Strategy 1	Fit for multidimensional data	Low-dimensional data query error is larger	Uniform budgeting

structure; then one can arrange all queried intervals into a tree, where the unit-length intervals are the leaves [48], and the query accuracy is enhanced by the consistency check between each layer node of the tree.

In view of the hierarchical method for optimizing histogram, Qardaji et al. [48] researched the factors which affect the mean squared error (MSE) of range query under the consistency check and different budgeting strategies. The hierarchical method in one dimension can reduce MSE when branching factor is more than 4; however, the hierarchical method fails to adapt multidimensional histogram when the dimensionality is greater than or equal to 3. The noise is directly added to each unit bucket in the histogram, which is called the Flat method by Qardaji et al., and Flat outweighs the hierarchical method. Detailed analysis of histogram release methods is shown in Table 2.

In a word, methods based on histogram release usually need to reconstruct the original histogram; when the number of partitions k grows, the approximation error gets less, but the noise error becomes larger; meanwhile, it will enlarge the time cost and make its application restricted. For instance, the time complexity of StructureFirst is $O(kN^2)$, $k = O(N)$; the value of k affects the time cost, so how to improve the efficiency of the algorithm and reduce the approximation error especially when data dimensionality is larger is one research direction of histogram. Moreover, many algorithms adopt uniform budgeting, and how to allocate the privacy parameter ε based on the requirements of application to enhance the query accuracy is another research direction. Note that Li et al. [49] first proposed DPSynthesizer, an open source visual toolkit for differentially private histogram data release, which can make the user see the one-dimensional or multidimensional differential privacy histogram data directly in terms of graph.

4.2. Tree Structure Release. In order to seek differential privacy budgeting strategy and reduce the query error, a series of methods based on tree-structure data split were proposed, and they are known as private spatial decompositions which divide the geospatial data into smaller regions and obtain statistics on the points within each region [30].

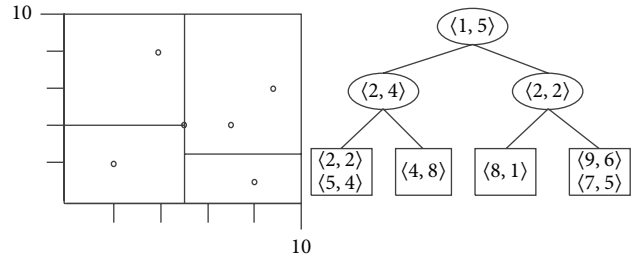


Figure 7: Example of private kd-tree.

If the partition discloses the sensitive information during spatial decomposition, it is called data-dependent decomposition; otherwise, it is called data-independent decomposition. For instance, kd-tree is recursively split through the median data value, which discloses the median value itself, so it is called data-dependent decomposition.

(1) Data-Dependent Decomposition. As shown in Figure 7, node $\langle 5, 4\rangle$ discloses itself during splitting; in order to mask the real information, the noise is added to the node. Let $D = \{x_1, \ldots, x_n\}$, $x_i \in [l, r]$; the data in D is sorted in ascending order and x_m is the median value; then the noise *noise* is added; that is, $M(D) = x_m + noise$, $noise \sim \mathrm{Lap}(\Delta f/\varepsilon)$; however, the noise median $M(D)$ may go beyond the range of the data D; that is, $M(D) \notin [l, r]$; as for this issue, Inan et al. [50, 51] proposed to adopt the mean instead of the median for numerical data, and the mean can be approximated as the ratio of the sum of noisy counts to the number of the counts, but this method fails to ensure the degree of approximation between the mean and the median.

As for indexing structure based on kd-tree, Xiao et al. [29, 46] first use privacy budget $\varepsilon/2$ for original data and then construct kd-tree and compute the median by noisy data, finally guaranteeing the privacy of the median.

Cormode et al. [30] proposed EM method to judge the median according to exponential mechanism among the kd-tree; EM returns x with $\Pr[\mathrm{EM}(D) = x] \propto e^{-\varepsilon|\mathrm{rank}(x)-\mathrm{rank}(x_m)|/2}$, and $\mathrm{rank}(x)$ represents the rank of x in data D. Since the value x is approximately equal to the median

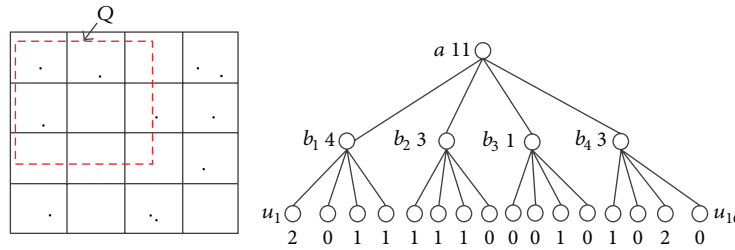


Figure 8: Example of private quadtree: the numbers are the actual counts.

x_m, the median is guaranteed. After the selection, using privacy budget $\varepsilon_{\text{median}}$ for x_m, using privacy budget $\varepsilon_{\text{count}}$ for count computation, and $\varepsilon = \varepsilon_{\text{median}} + \varepsilon_{\text{count}}$, a consideration is that larger value of $\varepsilon_{\text{median}}$ leads to more accurate median, but larger count error, while smaller value of $\varepsilon_{\text{median}}$ generates more accurate count computation, but larger median error, so how to allocate the budget ε for the count computation and the median computation and balance the errors on the count error and the median error is a concern. The above algorithm based on *kd*-tree and EM medians is called *kd*-standard.

(2) Data-Independent Decomposition. Cormode et al. [30] proposed the algorithm Quad-opt based on quadtree, which recursively divides the data space into equal quadrants, does not disclosure data information, and belongs to data-independent decomposition. As shown in Figure 8, the data a is divided into four quadrants b_i, $i \in [1, 4]$, and b_i would continue to divide until the predefined tree depth h is reached; h begins with 0. Compared to classical uniform budgeting, that is $\varepsilon_i = \varepsilon/(h+1)$, Cormode proposed a novel geometrical budgeting strategy (by a factor of $\sqrt[3]{2}$), which allocates ε_i for each level of the tree, thereto, $\varepsilon_i = 2^{(h-i)/3}\varepsilon(\sqrt[3]{2}-1)/(2^{(h+1)/3}-1)$, $i = [0, h]$, $\sum_{i=0}^{h}\varepsilon_i = \varepsilon$; then the least upper bound of error in query Q satisfies $\text{Err}(Q) = \text{Var}(Q) \leq 2^{h+7}/\varepsilon^2$; the query error is the sum of the nodes contained in Q variances. Quad-opt can handle the noisy data through the least square method during postprocessing and further enhance the query accuracy. For example, if the node a's budget is ε_1 and its children's budget is ε_2, in order to optimize the query error $\text{Err}(Q)$ when querying the noisy count in node a, we should let $Q(a) = (4\varepsilon_1^2/(4\varepsilon_1^2+\varepsilon_0^2))\widetilde{x}_a + (\varepsilon_0^2/(4\varepsilon_1^2+\varepsilon_0^2))\sum_{i=1}^{4}\widetilde{x}_{b_i}$, where $\widetilde{x}_a$ is the noisy count in node a itself and $\widetilde{x}_{b_i}$ is the noisy count of four children; then $\text{Err}(Q(a)) = \text{Var}(Q(a)) = 8/(4\varepsilon_1^2+\varepsilon_0^2) < 2/\varepsilon_1^2 = \text{Var}(\widetilde{x}_a)$.

If allocating all privacy budget to the leaf nodes, the algorithm is equivalent to directly dividing the data space into $w \times w$ cells. When the data is very sparse, adding noise to each cell will lead to poor information remaining; inspired by it, Fan et al. [32] proposed Spatial Estimation Algorithm (SEA), which merges similar cells into one group or partition to rise above the sparsity of the data. SEA first models different types of cells dependent on the domain knowledge before split; that is, each cell is sparse or dense, and the partition is called homogeneous if all the cells contained in this partition belong to the same category and need not to split; on the contrary, it is necessary to split until the partition is homogeneous or reaches the predefined tree depth. Then the noise *noise* is added to each partition p_i after the data splitting has completed, and the cell noise is $noise/p_i.\text{size}()$, where $p_i.\text{size}()$ is the number of cells in the partition p_i. Finally, SEA enhances the query accuracy while simultaneously providing very high level privacy guarantee by cells merge operation, essentially by reducing the added noise for cells.

The partition based on quadtree or *kd*-tree is fit for one-dimensional or two-dimensional data; as for multiple-dimensional data, Qardaji et al. [31] proposed method UG based on uniform grid which can be seen as an n-tree especially. There are two sources of error in the grid method: one is noise error introduced by adding random noise followed Laplace distribution; another is nonuniformity error introduced by assuming that the data points are distributed uniformly and nonuniformity error is proportional to the number of data points in the cells that fall on the border of the query rectangle [31]; then the query error $\text{Err}(Q)$ refers to the two errors above.

UG splits the space data into $w \times w$ cells; given a query Q, the query error $\text{Err}(Q) = \sqrt{2r}w/\varepsilon + \sqrt{r}N/wc_0$, where N is the number of data points, r is the ratio of the area of the query Q to the area of the whole domain, and c_0 is a constant. Note that UG gives the granularity w of the partition, which is the key factor effecting the accuracy of query results, and w is about $\sqrt{N\varepsilon/c}$, $(c = \sqrt{2}c_0)$.

UG treats all cells equally whether sparse cells or density cells. When the cell is very sparse, the noise error is too large; on the other hand, when the cell is very dense, the nonuniformity error is too large. Qardaji et al. further proposed the adaptive grids method AG to balance the two errors: when a region is sparse, a finer granularity is needed to reduce the noise error; when a region is dense, a coarse granularity is used to reduce the nonuniformity error.

AG first splits the space data into $w_1 \times w_1$ coarse cells. If a cell c_i is too dense, c_i can be further split into $w_2 \times w_2$ finer cells, and then issuing a count query for each coarse cell and finer cell using privacy budget $a\varepsilon$ and $(1-a)\varepsilon$ is carried out. In order to optimize the data query error, let $w_1 = \max(10, 0.25\lceil\sqrt{N\varepsilon/c}\rceil)$, $w_2 = \sqrt{2\widetilde{x}_i(1-\alpha)\varepsilon/\sqrt{2}c_0}$, where $\widetilde{x}_i$ is the noisy count in cell c_i. However, UG and AG also lack the adaptive judging criterion whether the cell is sparse or dense. At the same time, it is assumed that the nonuniformity error is proportional to the number of data points in the cells. In this model, the query rectangle is just square, which is not so consistent with the fact because the shape of query may be

Table 3: Tree structure release methods.

Method name	Strategy	Advantage	Defect	Budgeting
Quad-opt	Strategies 2 and 3	Computational efficiency is high; support any range query	Error is large	Geometric budgeting
SEA	Strategy 2	Noise error is small	Approximate error is large	Uniform budgeting
kd-standard	Strategy 2	Query accuracy is high	Only fit for low-dimensional data	Median and count use privacy budget together
UG	Strategy 1	Support any range query	Neglects the balance between noise error and nonuniformity error	Uniform budgeting
AG	Strategy 1	Balance the noise error and the nonuniformity error	Lack of the adaptive judging criterion	Uniform budgeting

rectangular, and the query results are influenced severely by the selection of c_0. Detailed analysis of tree structure release methods is shown in Table 3.

In conclusion, there is no privacy risk for tree structure in the data-independent decomposition, but how to reduce the noise error is the key for our special research. In Fan et al.'s data category, there are only sparse data and dense data, and the partition of data category is too coarse, so how to partition the data category based on the variance or information entropy to enhance the query accuracy is one of the research directions in the future. In data-dependent decomposition, some privacy budget is needed to guarantee the privacy for *kd*-tree itself disclosing the median information; thus, how to balance the allocation of the privacy budget to raise the query accuracy is one of the research directions in the future.

4.3. Time Series Release. There are many differential privacy real time data release, such as MHR, GPS, and track. The real time data with higher correlation between time stamps has a timescale, if the length of the time series is T and the noise $noise_t$ is added to the data x_t at time k, $noise \sim \mathrm{Lap}(T/\varepsilon)$, when T is large, the added noise gets large and leads to poor utility of the data.

As for time series data, in order to reduce the error, Rastogi and Nath [52] proposed algorithm DFT_k based on discrete Fourier transform. For time series D, DFT_k first executes the discrete Fourier transform, that is, $F = \mathrm{DFT}(D)$, and retains only the first k DEF coefficients; then the Laplace noise is added to those coefficients and the inverse Fourier transform is executed on the noisy coefficients $\widetilde{F}$, that is $\widetilde{D} = \mathrm{IDET}(\widetilde{F})$. Finally, the perturbed data $\widetilde{D}$ is released. However, the algorithm DFT_k needs to execute the DET() and IDET() operations with the full series, and it is not compatible to real time applications [32].

As for real time traffic monitoring data, in order to raise the execution efficiency and query accuracy of the algorithm in the real time environment, Fan et al. [32] proposed Temporal Estimation Algorithm (TEA) based on the time series model. Given the space G, TEA first splits the space into $w \times w$ cells and the time span T is discretized into time index, where $0 \le k < T$. The frequency series D^c of cell c is $\{x_k^c \mid 0 \le k < T\}$, where x_k^c is the frequency series of c at time stamp k and all frequency series D^G in G are $\{D^c \mid c \in G\}$. Then, the model is constructed dependent on domain knowledge, such as locations, population, and road networks; for each cell c, the process model is $x_{k+1}^c = x_k^c + \omega^c$, $p(\omega^c) \sim N(0, Q^c)$, where Q^c represents the degree of variation between neighbor time stamps, and the noise is added to the frequency series of c at time stamp k; that is, $\widetilde{x}_k^c = x_k^c + noise$, $noise \sim \mathrm{Lap}(0, 1/\varepsilon_0)$, where $\varepsilon_0 = \varepsilon/T$ and ε is uniform budgeting for the time span T. For estimation purpose, noise is replaced by white Gaussian noise; that is, $noise \sim N(0, R)$. Finally, the noisy data $\widetilde{D}^G$ is released. TEA utilizes the time series model and the educated guess to enhance the query accuracy; meanwhile, the computing complexity is low and time complexity is $O(kw^2)$, where $k = O(1)$.

For the time span T, if a real time system has higher sampling rate, larger noise will be introduced to satisfy the differential privacy; on the contrary, the approximate error is larger if the system is with lower sampling rate, so how to select the value of sampling rate is the key. Fan et al. proposed an adaptive sampling algorithm FAST [33, 34] to solve the problem in the subsequent research. FAST can adjust the sampling rate F automatically according to data dynamic and $F \le T$; the sampling rate is increased when the data changed rapidly and vice versa. The dynamic of current data can be described by the feedback error E_{k_n}, which is defined as $|\widehat{x}_{k_n} - \widehat{x}_{k_n}^-|/\max\{\widehat{x}_{k_n}, \delta\}$, where $\widehat{x}_{k_n}$ and $\widehat{x}_{k_n}^-$ are the posterior estimate and the prior state estimate for nth $(0 \le n < F)$ sampling point at time step k_n $(0 \le k_n < T)$ and δ is a parameter assigned by user and generally $\delta = 1$. Because of the data, the sampling frequency is not fixed; in order to achieve the optimal sampling rate in minimal time, FAST uses the PID controller to adjust the rate. Firstly, PID algorithm takes E_{k_n} as the input and gets the output Δ, that is, the PID error, which is defined as follows: $\Delta = C_p E_{k_n} + (C_i/T_i) \sum_{j=n-T_i+1}^{n} E_{k_j} +$

Table 4: Time series release methods.

Method name	Strategy	Advantage	Defect	Budgeting
DFT_k	Strategy 2	Supports long-range query	Parameter affects query accuracy	Uniform budgeting
TEA	Strategies 1 and 2	Computing complexity is low; query accuracy is high	Affected by data type	Uniform budgeting
FAST	Strategies 1 and 2	Query accuracy is high; adaptive data changes	Add the cost of feedback control	Uniform budgeting
U-KF	Strategy 2	Computing complexity is low; query accuracy is high	Only supports requests in signal server	Uniform budgeting

$C_d((E_{k_n} - E_{k_{n-1}})/(k_n - k_{n-1}))$, where C_p, C_i, and C_d are the proportional, integral, and derivative components of PID; then the new sampling rate I' is $\max\{1, I + \theta(1 - e^{(\Delta-\xi)/\xi})\}$, where θ and ξ are parameters specified by user. The adaptive sampling algorithm FAST reduces the noise error from $\Theta(T)$ to $\Theta(F)$.

As for web browsing behavior data, utilizing the rich correlation of the time series, Fan et al. proposed session-level differentially privacy algorithm U-KF [35] based on the user-level framework FAST. U-KF releases the differential privacy aggregates using state-space model in real time and adopts Kalman filter [53] during postprocessing. Given original data $\{x_k^i,\ i = 1, \ldots, m\}$, where x_k^i represents the number of sessions browsing page_i at time k. Firstly, in the prediction step, the prior estimate $\widehat{x}_k^{i-}$ is derived from the previous posterior estimate x_{k-1}^i, where "$\widehat{\ }$" represents state estimates and the superscript "$-$" represents variables related to the prior estimate [35]. Then, computing the perturbed value $\widetilde{x}_k^i$, that is $\widetilde{x}_k^i = x_k^i + noise_k^i$, $noise_k^i \sim \text{Lap}(0, l_{\max}/\varepsilon)$, where $l_{\max}$ is the query sensitivity. Next, in the correction step, the posterior estimate $\widehat{x}_k^i$ is computed according to the perturbed value $\widetilde{x}_k^i$ and the prior estimate $\widehat{x}_k^{i-}$, that is, $\widehat{x}_k^i = \widehat{x}_k^{i-} + K_k^i(\widetilde{x}_k^i - \widehat{x}_k^{i-})$, where K_k^i is the Kalman gain [53]. Finally, $\{\widehat{x}_k^i, i = 1, \ldots, m\}$ is released. U-KF is a univariate time series algorithm with higher query accuracy and its computing time is proportional to the number of web pages, that is, $O(m)$; however, U-KF only supports the browsing request in the single server, failing to support the request in different servers. Detailed analysis of time series release methods is shown in Table 4.

As for record data, in order to enhance the query accuracy, Xiao et al. [54, 55] proposed Privelet method based on wavelet transform. Privelet does not directly perturb the frequent matrix M of input data, but first transforms the M into the wavelet coefficient matrix C, which is perturbed by Laplace noise; then the noisy coefficient matrix is mapped into noisy frequent matrix $\widetilde{M}$; finally, $\widetilde{M}$ is released. It is known then that the differential privacy based on wavelet transform is feasible. As for real time data, such as smart meter, Ferrández-Pastor et al. [56] indicated that using wavelet transform to analyze the behavior of user is feasible, so the time series release combined with wavelet transform is one of the research directions in the future. At the same time, how to split the time span T is the key of time series release methods; the partition granularity of T should be approximate, combined with the features of data, such as the stepped data; how to adaptively spit the data is also one of the research directions in the future for time series.

4.4. Graph Release. Nowadays, digital health social network often contains user's sensitive information, and directly releasing these network data may harm individual privacy, so the network data should be sanitized before the release. Network data is different from general data, and it will be highly sensitive to small changes as the clustering coefficient is larger according to graph theory. For instance, adding random noise to a subgraph counting query for purpose of masking the presence or not of an edge, and larger noise implies poorer utility of the data. In order to enhance the utility of the sanitized data, a series of methods were proposed.

Hay et al. [57] first proposed edge differential privacy for networks data or differential privacy for short. Given graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, V_1 and V_2 are the set of vertices of graphs G_1 and G_2; E_1 and E_2 are the set of edges of graph G_1 and G_2; if $V_1 = V_2$, $E_1 = E_2$, and $|E_1| + 1 = |E_2|$, then G_1 and G_2 are neighbors which are denoted by $|G_1 \Delta G_2| = 1$.

Definition 8 (edge differential privacy [57]). Let G_1 and G_2 be two neighboring datasets which differ on at most one edge, denoted by $|G_1 \Delta G_2| = 1$; given a randomized function $A : D \to R$, $\text{Range}(A)$ is the set of all possible outputs of A in G_1 and G_2, and given an arbitrary subset $S \subseteq \text{Range}(A)$, A is said to be ε-edge differential privacy or ε-differential privacy if

$$\Pr[A(D) \in S] \leq \exp(\varepsilon) * \Pr\left[A\left(D'\right) \in S\right]. \quad (8)$$

Similarly, edge differential privacy can be expanded to multiple edges; if neighboring datasets differ on k edges, that is, $|G_1 \Delta G_2| = k$, then function A is said to be ε-k edge differential privacy. Furthermore, if neighboring datasets which differ up to all edges connected to one single node,

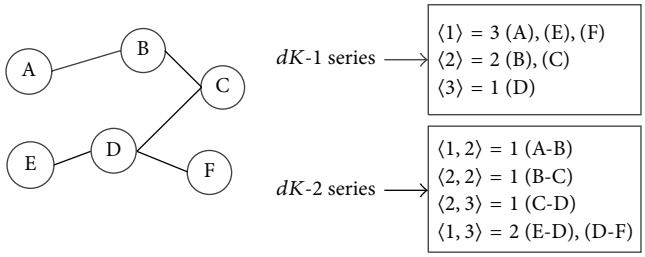


Figure 9: Example of dK-series.

that is, $|G_1 \Delta G_2| = \forall |k|$, $(k_i \in v)$, then A is said to be ε-node differential privacy, which is the ultimate aim desirable to achieve; however, it will introduce excessive noise and cause the worst utility of the data, in subsequent sections, A satisfies ε-edge differential privacy as the privacy standard.

In order to raise the utility of graph under the requested level of privacy, the graph's structure is needed to be captured first; then it would be converted back into a synthetic graph desired to be the same with original graph after adding noise. Sala et al. [58] proposed Pygmalion method, which captures the graph's structure according to dK-series graph model [59].

dK can capture the graph's structure into statistical number called dK-series, which is the number of nodes at different degree distribution for d-node subgraphs. As shown in Figure 9, when $d = 1$, the distribution is the node degree distribution; when $d = 2$, the distribution is the joint degree distribution for 2-node subgraphs; then using a matching generator, a synthetic graph is generated from the dK-series.

Pygmalion first captures the dK-series from the original graph and then sorts the dK-series and clusters the dK-series to get a set of subseries. Next, each subseries is perturbed by random noise according to local sensitivity. Finally, a synthetic graph is generated from the perturbed dK-series. However, local sensitivity might reveal sensitive information although it reduces the noise magnitude.

Wang and Wu [36] also used dK-series graph model to capture the graph's structure, in order to strictly achieve differential privacy and avoid revealing the privacy, and adopted the smooth sensitivity. For $2K$-graph, Wang and Wu proposed a method DP2$K(\varepsilon)$, which satisfies (ε, δ)-differential privacy. DP2$K(\varepsilon)$ first uses the parameter (ε, δ) to compute the value of (β, α), where $\alpha = \varepsilon/2$, $\beta = \varepsilon/2 \ln(2/\delta)$, and then obtains the β-smooth sensitivity $S_{f,\beta}(G)$ according to β and the sensitivity $\mathrm{LS}_f(G)$; next, it uses $S_{f,\beta}(G)$ to get the random noise which is added to the $2K$-graph series; finally, a new graph is generated by perturbed series.

Wang et al. [37] proposed another method, that is, LNPP, which uses spectral graph to guarantee the graph data. Spectral graph analysis mainly applies the graph's adjacent matrix to construct the graph's topological structure. For example, the adjacent matrix $A_{n\times n}$ represents the graph G, where $A_{i\times j} = 1$ if there is an edge between nodes i and j; $A_{i\times j} = 0$ if not. LNPP first decomposes matrix A and gets eigenvalues and eigenvectors; then, these values are perturbed based on Laplace mechanism, and the output eigenvectors are postprocessed by the vector orthogonalization technique [60] as the perturbed eigenvectors are no longer orthogonormal [37].

Adopting the spectral graph and the dK-graph model to guarantee privacy, the noise introduced by the former is proportional to $O(\sqrt{n})$, and the latter's sensitivity is $O(n)$, where n is the number of vertices in the input graph. In order to raise the utility of the data, Xiao et al. [38] proposed a novel method hrg-ε_1-e-ε_2, which is based on hierarchical random graph (HRG) model to reduce the perturbed noise magnitude.

Classical graph model is construed by considering the observed edges, and HRG uses the connection probabilities between vertices. In the HRG model, graph G can be represented by a hierarchical structure and the connection probabilities. The hierarchical structure of G is a rooted binary tree which has n leaf nodes corresponding to the n vertices of G [38]. Each internal node r has a probability p_r, and any two vertices i and j of G, their connection probability $p_{i,j} = p_r$, $p_r = e_r/n_{L_r} * n_{R_r}$, where r is two vertices' lowest common ancestor in T, R_r and L_r are the right and left subtrees of internal node r, n_{L_r} and n_{R_r} are the numbers of leaf nodes in left and right subtrees, and e_r is the number of edges in G whose endpoints are leaves of the two subtrees of r in T [38]. So, the hierarchical random graph can be defined as $(T, \{p_r\})$. As shown in Figure 10, original graph G can be represented by dendrogram T_1 and dendrogram T_2.

For graph G, in order to select a good dendrogram from numerous candidates, the notion likelihood is proposed to estimate how to represent G exactly by the selected dendrogram; that is, $L(T, \{p_r\}) = \prod_{r\in T} p_{p_r}^{e_r} (1 - p_r)^{n_{L_r} * n_{R_r} - e_r}$. In the above instance, firstly, the set of $\{p_r\}$ is calculated for each dendrogram; then the likelihoods of two dendrograms are computed; that is, $L(T_1) \approx 0.00165$ and $L(T_2) \approx 0433$; finally, $L(T_1)$ is compared to $L(T_2)$; one can see that T_2 is more exact to represent the original graph as $L(T_2)$ is larger.

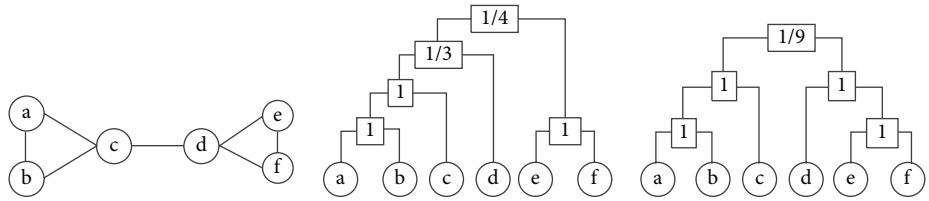
In hrg-ε_1-e-ε_2, privacy budget ε is divided into two parts; firstly, exponential mechanism is used; a Markov chain Monte Carlo procedure is designed to select a good dendrogram T_{sample}, in which only one probability would be affected with a single edge change. This step consumes privacy budget, that is, ε_1 and score function is $\log L(T, \{p_r\}) = -\sum_{r\in T} n_{L_r} n_{R_r} h(p_r)$, where $h(p_r) = -p_r \log p_r - (1-p_r) \log(1-p_r)$; then, Laplace mechanism is adopted; the set of $\{p_r\}$ of T_{sample} is perturbed using remaining budget ε_2; finally, the sanitized graph $\widetilde{G}$ is reconstructed by HRG model and is released. The global sensitivity of hrg-ε_1-e-ε_2 is $O(\log n)$ and reduces the added noise magnitude. Detailed analysis of graph release methods is shown in Table 5.

To sum up, directly adding noise to graph will lead to poor utility of the data as high sensitivity of graph, and it requires to project the data to other data types, such as spectral domain, dK-series, and connection probabilities; therefore, how to select a good data type with a lower sensitivity is one of the research directions in the future.

4.5. Pattern Mining Release. Frequent pattern is a subset of items and is frequently present in dataset, such as itemset, subsequence, and substructure, and it is the basis of

Table 5: Time series release methods.

Method name	Strategy	Advantage	Defect	Budgeting
Pygmalion	Strategy 2	Reducing the noise by subsection	Having the risk of privacy disclosure using local sensitivity	Uniform budgeting
DP2K(ε)	Strategy 2	Guaranteeing the privacy by smooth sensitivity	Noise is large	Uniform budgeting
LNPP	Strategies 1 and 2	Enhancing the utility by postprocessing	Noise is large	Uniform budgeting
hrg-ε_1-e-ε_2	Strategy 2	Utility of the data is high	Result affected by different budgeting strategies	Sampling and perturbation use privacy budget together



(a) Original graph G (b) Dendrogram T_1 (c) Dendrogram T_2

Figure 10: Example of the HRG model.

mining tasks, that is, classification, clustering, and so forth. However, it may reveal individual's privacy during frequent pattern mining, so how to identify the useful patterns in the data while simultaneously guarantee user's privacy is a new research direction of data mining.

Frequent pattern mining algorithm is usually measured by support degree [28, 39, 40], occurrence [61], and stay point [62, 63], given data $D = \{\text{tran}_1, \text{tran}_2, \ldots, \text{tran}_n\}$, $\text{tran}_i \in T$, where T is the transaction set constituted by all transactions, and each transaction contains some items; that is, $\text{tran}_i = \{\text{item}_1, \text{item}_2, \ldots, \text{item}_m\}$, $\text{item}_m \in I$, where I is the item space constituted by all items and itemset $itemset_j$ is the subset of item space I. if transaction tran_n contains itemset $itemset_j$, we say that tran_n supports $itemset_j$, and the ratio of the number of transactions supporting $itemset_j$ to the total number of transactions is the support degree for $itemset_j$. Then, it could be said that $itemset_j$ is a frequent itemset if the support degree is more than the giving threshold.

In terms of personal location information, in order to mine the useful pattern, for example, people's interesting geographic location, Ho and Ruan [62] proposed BuildDPQuadTree method based on quadtree and density-based clustering algorithm (DBSCAN). How to define the interesting region when we query the location history database is the focus, and the notion of stay point is proposed. Given the trajectory data $\text{traj}_i^{k_i} = \{x_1, x_2, \ldots, x_{k_i}\}$, $x_j = (\text{Lat}_j, \text{Lon}_j, t_j)$, where t_j is a timestamp, Lat_j is the latitude, and Lon_j is the longitude. If a trajectory stays in a circle region with a ρ radius for at least a time period ΔT, the center (Lat, Lon) of the circle is a stay point; given a set of trajectories $TJ = \{\text{traj}_1^{k_1}, \text{traj}_2^{k_2}, \ldots, \text{traj}_s^{k_s}\}$, the region is called interesting location if it has more than r stay points in it.

BuildDPQuadTree first recursively divides the data space G into small regions, given stay point set S, threshold $T_{\text{threshold}}$, and privacy budget ε_{qt}; if $|S| + \text{Lap}(3T_{\text{threshold}}/\varepsilon_{\text{qt}}) \leq 3T_{\text{threshold}}$, stop partition and go to partition if not. After getting the set of spatial partitions, each cluster G_j, which is the set of stay points of the form $(\text{Lat}_j, \text{Lon}_j)$, is obtained by density-based spatial clustering of application with noise clustering method [64]. Given the threshold r, privacy budget ε_{cg}, and ε_{cts}, if $|G_j| + \text{Lap}(\Delta f_{\text{cts}}^j/\varepsilon_{\text{cts}}) \geq r$, where $\Delta f_{\text{cts}}^j = \max_{i \in D} \#\{s \in G_j \mid s \text{ is a stay point for individual } i\}$ and D is the set of all the individuals who have records in the cluster G_j [62], then G_j is called interesting region, and its center location is defined as $[\sum_{k=1}^{|G_j|}(\text{Lat}_k, \text{Lon}_k)]/|G_j| + \text{Lap}(\Delta f_{\text{cg}}^j/\varepsilon_{\text{cts}})$, where $\Delta f_{\text{cg}}^j = \max \text{length}(\text{point}_x, \text{point}_y)/2$, $\text{point}_i \in G_j$, and $\varepsilon = \sum_{i=1}^{h} \varepsilon_{\text{qt}} + \varepsilon_{\text{cg}} + \varepsilon_{\text{cts}}$, where h is the quadtree depth. BuildDPQuadTree provides a novel method identifying the interesting region by stay point; however, its noise magnitude is easily affected by threshold $T_{\text{threshold}}$. In the Ho and Ruan's subsequent research, the algorithm DP-ILD [63] adopting β-smooth sensitivity was proposed and it satisfies (ε, δ)-differential privacy.

The sensitivity of itemset query depends on the dimensionality of itemset during Top-k frequent pattern mining. In order to reduce the sensitivity, the data usually is projected into a lower dimensional space. Li et al. [39] proposed PrivBasis method to search Top-k frequent patterns. In the initialization step, PrivBasis tries to reduce the search space

by finding the minimal sets of items I_B, which contains the top k frequent itemsets [39]; then the supports of all subsets of I_B are derived by applying binary itemset support counting [65]; however, the final frequent patterns may be imprecise.

In order to raise the utility of the data, Lee and Clifton [40] proposed the NoiseCut method based on the sparse vector technique and FP-tree. NoiseCut first identifies all frequent itemsets L; in this step, sparse vector technique is adopted to reduce consumption of privacy budget, that is, given a noisy threshold $\widehat{\tau}$ and a count query q, which only pays privacy budget if $q(D) + \mathrm{Lap}(2k/\varepsilon) \geq \widehat{\tau}$, where $\widehat{\tau} = \sigma_k + \mathrm{Lap}(\cdot)$, σ_k is the support of the kth most frequent itemset. It is noted that NoiseCut regards the itemset as frequent if its noisy support $(\tau + \alpha)$ is bigger than $\widehat{\tau}$. Then the noisy FP-tree is built according to L, and the supports of itemsets and Top-k frequent itemsets can be derived from the FP-tree. In NoiseCut, given an itemset whose support is $\tau + \alpha$, its false negative is $\exp(-\alpha\varepsilon/2)(\alpha\varepsilon/2 + 2)/4$; the smaller the false negative is, the higher obtaining correct frequent patterns becomes.

As for graph data, Shen and Yu [28] proposed frequent graph pattern mining method Diff-FPM based on Markov chain Monte Carlo random walk technology. Diff-FPM obtains the Top-k frequent subgraph mainly according to random walk; then, the real counts of subgraphs are perturbed by Laplace random noise. The key to Diff-FPM is whether random walk can reach steady state or not. Diff-FPM satisfies ε-differential privacy if steady state is reached; if not, Diff-FPM only satisfies (ε, δ)-differential privacy, in which the security of the data maybe is affected.

As for sequential data, such as DNA sequences and trajectories, Luca and Li [61] indicated that there are some disadvantages using support to mine frequent sequential patterns; for example, one pattern may fail to be regarded as frequent pattern if it repetitively occurs in the same transaction, which may issue in a loss of information patterns. In addition, patterns are represented as a subset of an universe of items and may not well describe the sequentiality of important events in reality [61]. In order to successfully mine the frequent sequential patterns from the sequential data, Bonomi et al. proposed the notion of occurrence which can identify repetitive patterns as frequent. Given a pattern $p = a_0 a_1 \cdots a_{n-1}$, $a_i \in \Sigma$, and a string $x = x_0 x_1 \cdots x_{m-1}$, if there is an integer i $(0 \leq i \leq m - n)$ making $x_{i+j} = a_j$, $j = 0, \ldots, n-1$, it says that the pattern p presents in position i of the string x, and $f_x(p)$ represents all number of occurrences of p in x. Given the dataset $D = \{x^0, x^1, \ldots, x^{N-1}\}$, $F_D(p) = \sum_{i=0}^{N-1} f_{x^i}(p)$ denotes the number of occurrences of the p in D, and p is called frequent sequential pattern if $F_D(p)$ is larger than the giving threshold. Note that adding or removing one string x in the dataset D may affect the count of a pattern up to $O(|x|)$ and lead to higher sensitivity; in order to reduce the sensitivity, it usually uses a prefix tree structure [66, 67] to split the input data and uses the frequency of the prefixes instead of the frequency of the patterns [61]. Detailed analysis of pattern mining release methods is shown in Table 6.

Luca and Li pointed out the existing differently privacy frequent pattern mining methods mainly based on the exact data; however, the real data usually are noisy, and some patterns may be lost [61]. There are less research about how to mine the frequent patterns in the noisy data, which is a future research direction. At the same time, how to raise the utility of the data while simultaneously providing privacy guarantee is an issue for further research.

For different data types, related researchers proposed a series of differential privacy data release methods, and the methods summarized in this paper only are a part of the whole; there are many methods which the paper fails to describe, such as batch query methods based on matrix [68–70] and data release methods based on machine learning [71].

4.6. Discussion. Data release methods under different strategies are the main research contents for differential privacy and are the theoretical basis for practical applications. Classification and comparison of the differential privacy data release methods are shown in Table 7, from which we can see that there are few research achievements in this new research area, and there still exists many urgent issues to be addressed and some new study directions to be explored.

(1) Highly Sensitive Problem. As for multiple-dimensional or graph data, whose sensitivity is so high that it may lead to poor utility of the data, how to reduce the sensitivity of function f and enhance the utility require deep-going research.

(2) Time Series Problem. As far as time series data is concerned, how to balance the utility and the security when the time T is increasing is a question deserving research, and privacy guarantee for online data is another question worth studying as well.

(3) Efficiency Problem. Computational complexity of many differential privacy algorithms is relatively high, so how to ensure privacy while simultaneously improving algorithm efficiency is a question deserving research.

(4) New Research Direction for Differential Privacy. Among existing differential privacy data release methods, the data usually are stored in one party; if the data are stored in multiple parties, therefore how to share the data and guarantee user's privacy between multiple parties is a question deserving further research for distributed differential privacy [72, 73], which is also called security multiparty computation problem. It mainly faces the following difficulties: (1) excessive communication overhead makes the algorithm infeasible if the data and participants are large; (2) how to design an algorithm and make it satisfy differential privacy is another obstacle. So an efficient and safe algorithm with lower communication overhead is a question deserving research, for example, how to design a differential privacy algorithm for such distributed real time streaming data as financial data.

(5) The Definition Expansion of Differential Privacy. Kearns et al. [74] introduced the differential privacy into game theory to guarantee player's privacy. The paper points out that the equilibria of complete information games can be implemented in

Table 6: Pattern mining release methods.

Method name	Strategy	Advantage	Defect	Budgeting
BuildDPQuadTree	Strategy 2	Identify the interesting location using stay point	Noise is large	Partition and cluster use the privacy budget together
DP-ILD	Strategy 2	Utility of the data is high	Only supports offline data	Uniform budgeting
PrivBasis	Strategy 2	Mining speed is fast	The final frequent patterns may be imprecise	Uniform budgeting
NoiseCut	Strategies 1 and 2	Utility of the data is high	Results are affected by different budgeting strategies	Privacy budget is allocated for two steps of the algorithm
Diff-FPM	Strategy 2	Query accuracy is high	The drop in utility with the increase of the number of outputs [28]	Sampling and perturbation use the privacy budget together

Table 7: Classification and comparison of differential privacy data release methods.

Classification	Advantage	Defect	Representative methods	Typical Applications
Histogram	Supports any range query	The noise is large in high-dimensional data	DPCube [29] NoiseFirst [15] Structure First [15]	Statistical analysis of disease and search history
Tree structure	Supports multiple-dimensional data and data-dependent or data-independent query	The noise is large in high-dimensional data	Quad-opt [30] AG [31] SEA [32]	Location query of user and device, transport planning
Time series	Supports time series query	It is hard to balance the utility and security in high-dimensional data	TEA [32] FAST [33, 34] U-KF [35]	Real time traffic, disease surveillance
Graph	Supports the analysis and query of graph data	Sensitivity is high, and node-differential privacy is difficult to achieve	DP2K(ε) [36] LNPP [37] hrg-ε_1-e-ε_2 [38]	Relationship analysis of user in health social network
Pattern mining	Supports differential privacy pattern mining, and original data can be constructed by frequent patterns	Long pattern leads to large noise	PrivBasis [39] NoiseCut [40] Diff-FPM [28]	User behavior, DNA sequences, trajectory and disease trend analysis, recommended system

setting of incomplete information under certain conditions (the number of players is large). Any algorithm that estimates a correlated equilibrium of a complete information game while satisfying a variant of differential privacy is called joint differential privacy. By this time, no group of the players can derive "much" about the type of any player outside the group, even if these players collude in an arbitrary way [74].

Definition 9 (joint differential privacy [74]). Given a randomized function $A : D^n \rightarrow R^n$, let D and D' be i-neighboring datasets which differ only in their ith coordinate, $i \in \{1, \ldots, n\}$; for every event $S \subseteq R^{n-1}$, A is said to be (ε, δ)-joint differential privacy if

$$\Pr\left[A\left(D\right)_{-i} \in S\right] \leq \exp\left(\varepsilon\right) * \Pr\left[A\left(D'\right)_{-i} \in S\right] + \delta, \quad (9)$$

where $A(D) = (r_1, \ldots, r_n)$, $A(D)_i = r_i$, $A(D)_{-i} = (r_1, \ldots, r_{i-1}, r_{i+1}, \ldots, r_n)$, and (ε, δ)-joint differential privacy equals ε-joint differential privacy when $\delta = 0$.

In joint differential privacy, the input date has n data elements and the output also is a vector of n messages; that

is, the input $D = (x_1, \ldots, x_n) \in D^n$ and the output is $r = (r_1, \ldots, r_n) \in R^n$. Each output element r_i is corresponding to a input element x_i of one player, and it can guarantee ith player's privacy even if other players collude; that is, they know all information except input x_i and output r_i of ith player $(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$ and $r_1, \ldots, r_{i-1}, r_{i+1}, \ldots, r_n)$ but fail to learn much about ith player's input x_i.

Joint differential privacy is an appropriate solution concept for problems whose solution can be partitioned amongst the n players who provide the privacy data to the algorithm, such as equilibrium computation [74, 75] in game theory and agent's privacy guarantee in maximum matching problem [76].

5. Conclusion and Future Work

Differential privacy has become the de facto standard to guarantee privacy and is one of the hot research topics in the field of healthcare applications. In this paper, the methods of differentially private data releasing in recent years' literatures are classified, and a series of data release methods are summarized for privacy problem, which provide certain references for differential privacy in further study. In Section 4, as for different categories, different research directions are proposed; for example, it is pointed out that time series release combined with wavelet transform is one of the research directions to solve the privacy of real time streaming data. However, there are still some other questions which are worth studying. They are summarized as follows.

(1) The Privacy Budgeting for Tree Structure. Privacy budgeting strategy is usually uniform budgeting in tree structure release, and Cormode et al. [30] presented a novel nonuniform budgeting method based on complete quadtree to improve query accuracy. The key of the method is geometric budgeting strategy, and its theoretical analysis is conducted by the total variance of the all node variances with a great theoretical significance, but the effect is not optimal as fewer nodes may be touched in actual queries. Inspired by it, the error of the data query is small if privacy budget properly tilts toward to the below level nodes of the tree, so how to design an optimal privacy budgeting strategy is a great challenge.

(2) The Balance of the Noise Error and Nonuniformity Error. As for geospatial data, Qardaji et al. [31] constructed a differentially private synopsis by choosing the right partition granularity over the data domain to balance the noise error and nonuniformity error, which is proportional to the number of data points in the cells that intersect with the border of the query rectangle. However, nonuniformity error is not only connected with the point number, but also refers to the area of the cells that intersect with the border of the query rectangle, as the less area of intersected portion is, the less nonuniformity error will be. Nonuniformity error especially is 0 when the area of intersected portion is 0, so how to choose the optimal partition granularity for geospatial data when nonuniformity error is connected with both point number and area is a great challenge as well.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 41371402); the National Basic Research Program of China (973 Program) (Grant no. 2011CB302306); the Fundamental Research Funds for the Central Universities under Grant no. 2015211020201.

References

[1] F. K. Dankar and K. El Emam, "The application of differential privacy to health data," in *Proceedings of the Joint EDBT/ICDT Workshops*, pp. 158–166, ACM, Berlin, Germany, March 2012.

[2] T. Francis, M. Madiajagan, and V. Kumar, "Privacy issues and techniques in E-health systems," in *Proceedings of the ACM SIGMIS Conference on Computers and People Research (SIGMIS-CPR '15)*, pp. 113–115, ACM, Newport Beach, Calif, USA, June 2015.

[3] Y.-A. de Montjoye, L. Radaelli, V. K. Singh, and A. S. Pentland, "Unique in the shopping mall: on the reidentifiability of credit card metadata," *Science*, vol. 347, no. 6221, pp. 536–539, 2015.

[4] Y. Li, W. Wen, and G.-Q. Xie, "Survey of research on differential privacy," *Application Research of Computers*, vol. 29, no. 9, pp. 3201–3211, 2012.

[5] H. Ye and E. S. Chen, "Attribute utility motivated k-anonymization of datasets to support the heterogeneous needs of biomedical researchers," *AMIA Annual Symposium Proceedings*, vol. 2011, pp. 1573–1582, 2011.

[6] L. Sweeney, Ed., *Computational Disclosure Control: A Primer on Data Privacy Protection*, Massachusetts Institute of Technology, Cambridge, Mass, USA, 2001.

[7] P. Samarati, "Protecting respondents' identities in microdata release," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, no. 6, pp. 1010–1027, 2001.

[8] A. Machanavajjhala, D. Kifer, and J. Gehrke, "L-diversity: privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, article 3, 2007.

[9] L. Ninghui, L. Tiancheng, and S. Venkatasubramanian, "t-Closeness: privacy beyond k-anonymity and I-diversity," in *Proceedings of the 23rd International Conference on Data Engineering (ICDE '07)*, pp. 106–115, IEEE, Istanbul, Turkey, April 2007.

[10] S. Avancha, A. Baxi, and D. Kotz, "Privacy in mobile technology for personal healthcare," *ACM Computing Surveys*, vol. 45, no. 1, article 3, 2012.

[11] T. Francis, M. Madiajagan, and V. Kumar, "Privacy issues and techniques in e-health systems," in *Proceedings of the ACM SIGMIS Conference on Computers and People Research*, pp. 113–115, ACM, Newport Beach, Calif, USA, June 2015.

[12] S. Sadki and H. El Bakkali, "Enhancing privacy on mobile health: an integrated privacy module," in *Proceedings of the 5th International Conference on Next Generation Networks and Services (NGNS '14)*, pp. 245–250, IEEE, Casablanca, Morocco, May 2014.

[13] C. Dwork, "Differential privacy," in *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP '06)*, pp. 1–12, Venice, Italy, July 2006.

[14] G. Yuan, Z. Zhang, M. Winslett, X. Xiao, Y. Yang, and Z. Hao, "Low-rank mechanism: optimizing batch queries under differential privacy," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, pp. 1352–1363, 2012.

[15] J. Xu, Z. Zhang, X. Xiao, Y. Yang, G. Yu, and M. Winslett, "Differentially private histogram publication," *The VLDB Journal*, vol. 22, no. 6, pp. 797–822, 2013.

[16] V. Rastogi and S. Nath, "Differentially private aggregation of distributed time-series with transformation and encryption," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '10)*, pp. 735–746, ACM, Indianapolis, Ind, USA, June 2010.

[17] R. Zhu, W. Shu, T. Mao, and T. Deng, "Enhanced MAC protocol to support multimedia traffic in cognitive wireless mesh networks," *Multimedia Tools and Applications*, vol. 67, no. 1, pp. 269–288, 2013.

[18] K. Zhu, R. Zhu, H. Nii, H. Samani, and B. Jalaeian, "PaperIO: a 3D interface towards the internet of embedded paper-craft," *IEICE Transactions on Information and Systems*, vol. 97, no. 10, pp. 2597–2605, 2014.

[19] D. Leoni, "Non-interactive differential privacy: a survey," in *Proceedings of the 1st International Workshop on Open Data (WOD '12)*, pp. 40–52, ACM, Nantes, France, May 2012.

[20] P. Xiong, T.-Q. Zhu, and X.-F. Wang, "A survey on differential privacy and applications," *Chinese Journal of Computers*, vol. 37, no. 1, pp. 101–122, 2014.

[21] Y. Yang, Z. Zhang, G. Miklau et al., "Differential privacy in data publication and analysis," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '12)*, pp. 601–606, Scottsdale, Ariz, USA, May 2012.

[22] International Organization for Standardization, *Information Technology—Business Operational View—Part 1: Operational Aspects of Open-Edi for Implementation*, ISO/IEC 15944-1, International Organization for Standardization, London, UK, 2nd edition, 2011.

[23] C. Dwork, "Differential privacy: a survey of results," in *Theory and Applications of Models of Computation: 5th International Conference, TAMC 2008, Xi'an, China, April 25–29, 2008. Proceedings*, vol. 4978 of *Lecture Notes in Computer Science*, pp. 1–19, Springer, Berlin, Germany, 2008.

[24] C. Dwork, "Differential privacy," in *Encyclopedia of Cryptography and Security*, pp. 338–340, 2nd edition, 2011.

[25] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: privacy via distributed noise generation," in *Advances in Cryptology—EUROCRYPT: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28–June 1, 2006. Proceedings*, vol. 4004 of *Lecture Notes in Computer Science*, pp. 486–503, Springer, Berlin, Germany, 2006.

[26] C. Dwork, "A firm foundation for private data analysis," *Communications of the ACM*, vol. 54, no. 1, pp. 86–95, 2011.

[27] J. Hsu, M. Gaboardi, A. Haeberlen et al., "Differential privacy: an economic method for choosing epsilon," in *Proceedings of the IEEE 27th Computer Security Foundations Symposium (CSF '14)*, pp. 398–410, IEEE, Vienna, Austria, July 2014.

[28] E. Shen and T. Yu, "Mining frequent graph patterns with differential privacy," in *Proceedings of the 19th ACM SIGKDD International Conference*, pp. 545–553, Chicago, Ill, USA, August 2013.

[29] Y. Xiao, J. Gardner, and L. Xiong, "DPCube: releasing differentially private data cubes for health information," in *Proceedings of the IEEE 28th International Conference on Data Engineering (ICDE '12)*, pp. 1305–1308, IEEE, Washington, DC, USA, April 2012.

[30] G. Cormode, C. Procopiuc, D. Srivastava, E. Shen, and T. Yu, "Differentially private spatial decompositions," in *Proceedings of the 28th IEEE International Conference on Data Engineering (ICDE '12)*, pp. 20–31, April 2012.

[31] W. Qardaji, W. Yang, and N. Li, "Differentially private grids for geospatial data," in *Proceedings of the 29th International Conference on Data Engineering (ICDE '13)*, pp. 757–768, Brisbane, Australia, April 2013.

[32] L. Fan, L. Xiong, and V. Sunderam, "Differentially private multi-dimensional time series release for traffic monitoring," in *Data and Applications Security and Privacy XXVII*, vol. 7964 of *Lecture Notes in Computer Science*, pp. 33–48, Springer, Berlin, Germany, 2013.

[33] L. Fan, L. Xiong, and V. Sunderam, "FAST: differentially private real-time aggregate monitor with filtering and adaptive sampling," in *Proceedings of the ACM SIGMOD Conference on Management of Data*, pp. 1065–1068, New York, NY, USA, June 2013.

[34] L. Fan and L. Xiong, "An adaptive approach to real-time aggregate monitoring with differential privacy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 9, pp. 2094–2106, 2014.

[35] L. Fan, L. Bonomi, L. Xiong, and V. Sunderam, "Monitoring web browsing behavior with differential privacy," in *Proceedings of the 23rd International Conference on World Wide Web (WWW '14)*, pp. 177–187, Seoul, Republic of Korea, April 2014.

[36] Y. Wang and X. Wu, "Preserving differential privacy in degree-correlation based graph generation," *Transactions on Data Privacy*, vol. 6, no. 2, pp. 127–145, 2013.

[37] Y. Wang, X. Wu, and L. Wu, "Differential privacy preserving spectral graph analysis," in *Advances in Knowledge Discovery and Data Mining: 17th Pacific-Asia Conference, PAKDD 2013, Gold Coast, Australia, April 14–17, 2013, Proceedings, Part II*, vol. 7819 of *Lecture Notes in Computer Science*, pp. 329–340, Springer, Berlin, Germany, 2013.

[38] Q. Xiao, R. Chen, and K.-L. Tan, "Differentially private network data release via structural inference," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*, pp. 911–920, New York, NY, USA, August 2014.

[39] N. Li, W. Qardaji, D. Su, and J. Cao, "Privbasis: frequent itemset mining with differential privacy," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, pp. 1340–1351, 2012.

[40] J. Lee and C. W. Clifton, "Top-k frequent itemsets via differentially private FP-tree," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*, pp. 931–940, New York, NY, USA, August 2014.

[41] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC '07)*, pp. 5–84, ACM, 2007.

[42] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4–7, 2006. Proceedings*, vol. 3876

of *Lecture Notes in Computer Science*, pp. 265–284, Springer, Berlin, Germany, 2006.

[43] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *Proceedings of the 48th Annual Symposium on Foundations of Computer Science (FOCS '07)*, pp. 94–103, Providence, RI, USA, October 2007.

[44] F. McSherry, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," *Communications of the ACM*, vol. 53, no. 9, pp. 89–97, 2010.

[45] X.-J. Zhang and X.-F. Meng, "Differential privacy in data publication and analysis," *Chinese Journal of Computers*, vol. 37, no. 4, pp. 927–949, 2014.

[46] Y. Xiao, L. Xiong, and C. Yuan, "Differentially private data release through multidimensional partitioning," in *Secure Data Management: 7th VLDB Workshop, SDM 2010, Singapore, September 17, 2010. proceedings*, vol. 6358 of *Lecture Notes in Computer Science*, pp. 150–168, Springer, Berlin, Germany, 2010.

[47] M. Hay, V. Rastogi, G. Miklau, and D. Suciu, "Boosting the accuracy of differentially private histograms through consistency," in *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 1021–1032.

[48] W. Qardaji, W. Yang, and N. Li, "Understanding hierarchical methods for differentially private histograms," *Proceedings of the VLDB Endowment*, vol. 6, no. 14, pp. 1954–1965, 2013.

[49] H. Li, L. Xiong, L. Zhang, and X. Jiang, "DPSynthesizer: differentially private data synthesizer for privacy preserving data sharing," *Proceedings of the VLDB Endowment*, vol. 7, no. 13, pp. 1677–1680, 2014.

[50] A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino, "Private record matching using differential privacy," in *Proceedings of the 13th International Conference on Extending Database Technology (EDBT '10)*, pp. 123–134, ACM, Lausanne, Switzerland, March 2010.

[51] A. Inan, M. Kantarcioglu, G. Ghinita, and E. Bertino, "A hybrid approach to private record matching," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 5, pp. 684–698, 2012.

[52] V. Rastogi and S. Nath, "Differentially private aggregation of distributed time-series with transformation and encryption," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 735–746, ACM, Indianapolis, Ind, USA, June 2010.

[53] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[54] K. Xiao, G. Wang, and J. Gehrke, "Differential privacy via wavelet transforms," in *Proceedings of the 26th IEEE International Conference on Data Engineering (ICDE '10)*, pp. 225–236, IEEE, Long Beach, Calif, USA, March 2010.

[55] X. Xiao, G. Wang, and J. Gehrke, "Differential privacy via wavelet transforms," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 8, pp. 1200–1214, 2011.

[56] F.-J. Ferrández-Pastor, J.-M. García-Chamizo, M. Nieto-Hidalgo, V. Romacho-Agud, and F. Flórez-Revuelta, "Using wavelet transform to disaggregate electrical power consumption into the major end-uses," in *Ubiquitous Computing and Ambient Intelligence. Personalisation and User Adapted Services*, vol. 8867 of *Lecture Notes in Computer Science*, pp. 272–279, Springer, 2014.

[57] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *Proceedings of the 9th IEEE International Conference on Data Mining (ICDM '09)*, pp. 169–178, IEEE, Miami, Fla, USA, December 2009.

[58] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao, "Sharing graphs using differentially private graph models," in *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC '11)*, pp. 81–97, Berlin, Germany, November 2011.

[59] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat, "Systematic topology analysis and generation using degree correlations," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '06)*, pp. 135–146, ACM, Pisa, Italy, September 2006.

[60] P. H. Garthwaite, F. Critchley, K. Anaya-Izquierdo, and E. Mubwandarikwa, "Orthogonalization of vectors with minimal adjustment," *Biometrika*, vol. 99, no. 4, pp. 787–798, 2012.

[61] B. Luca and X. Li, "Mining frequent patterns with differential privacy," *Proceedings of the VLDB Endowment*, vol. 6, no. 12, pp. 1422–1427, 2013.

[62] S.-S. Ho and S. Ruan, "Differential privacy for location pattern mining," in *Proceedings of the 4th ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS (SPRINGL '11)*, pp. 17–24, ACM, Chicago, Ill, USA, November 2011.

[63] S.-S. Ho and S. Ruan, "Preserving privacy for interesting location pattern mining from trajectory data," *Transactions on Data Privacy*, vol. 6, no. 1, pp. 87–106, 2013.

[64] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the International Conference on Knowledge Discovery & Data Mining (KDD '96)*, pp. 226–231, 1996.

[65] J. Chen and K. Xiao, "BISC: a bitmap itemset support counting approach for efficient frequent itemset mining," *ACM Transactions on Knowledge Discovery from Data*, vol. 4, no. 3, article no. 12, 2010.

[66] L. Bonomi and L. Xiong, "A two-phase algorithm for mining sequential patterns with differential privacy," in *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM '13)*, pp. 269–278, Burlingame, Calif, USA, November 2013.

[67] L. Bonomi, L. Xiong, R. Chen, and B. C. M. Fung, "Frequent grams based embedding for privacy preserving record linkage," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM '12)*, pp. 1597–1601, November 2012.

[68] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor, "Optimizing linear counting queries under differential privacy," in *Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '10)*, pp. 123–134, ACM, Indianapolis, Ind, USA, June 2010.

[69] C. Li and G. Miklau, "An adaptive mechanism for accurate query answering under differential privacy," *Proceedings of the VLDB Endowment*, vol. 5, no. 6, pp. 514–525, 2012.

[70] G. Yuan, Z. Zhang, M. Winslett, X. Xiao, Y. Yang, and Z. Hao, "Low-rank mechanism: optimizing batch queries under differential privacy," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, pp. 1352–1363, 2012.

[71] Z. Ji, X. Jiang, S. Wang, L. Xiong, and L. Ohno-Machado, "Differentially private distributed logistic regression using private

and public data," *BMC Medical Genomics*, vol. 7, supplement 1, article S14, 2014.

[72] J. Liu, J. Z. Huang, J. Luo, and L. Xiong, "Privacy preserving distributed DBSCAN clustering," *Transactions on Data Privacy*, vol. 6, no. 1, pp. 69–85, 2013.

[73] A. Friedman, I. Sharfman, D. Keren, and A. Schuster, "Privacy-preserving distributed stream monitoring," in *Proceedings of the Network and Distributed System Security Symposium*, pp. 1–12, San Diego, Calif, USA, February 2014.

[74] M. Kearns, M. M. Pai, A. Roth, and J. Ullman, "Mechanism design in large games: incentives and privacy," *American Economic Review*, vol. 104, no. 5, pp. 431–435, 2014.

[75] A. Roth, "Differential privacy, equilibrium, and efficient allocation of resources," in *Proceedings of the 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton '13)*, pp. 1593–1597, International Society for Optics and Photonics, Monticello, Ill, USA, October 2013.

[76] J. Hsu, Z. Huang, A. Roth, T. Roughgarden, and Z. S. Wu, "Private matchings and allocations," in *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC '14)*, pp. 21–30, ACM, New York, NY, USA, May-June 2014.