Computer Science and Information Systems 16(3):915–938 https://doi.org/10.2298/CSIS180901033L

A Novel Self-adaptive Grid-partitioning Noise Optimization Algorithm Based on Differential Privacy

Zhaobin Liu[1], Haoze Lv[1], Minghui Li[1], Zhiyang Li[1], and Zhiyi Huang[2]

[1] School of Information Science and Technology, Dalian Maritime University
China
Correspondence: lizy0205@gmail.com
[2] Department of Computer Science, University of Otago
hzy@cs.otago.ac.nz

Abstract. As the development of the big data and Internet, the data sharing of users that contains lots of useful information are needed more frequently. In particular, with the widespread of smart devices, a great deal of location-based data information has been generated. To ensure that service providers can supply a completely optimal quality of service, users must provide exact location information. However, in that case, privacy disclosure accident is endless. As a result, people are paying attention to how to protect private data with location information. Of all the solutions of this problem, the differential privacy theory is based on strict mathematics and provides precise definition and quantitative assessed methods for privacy protection, it is widely used in location-based application. In this paper, we propose a self-adaptive grid-partitioning algorithm based on differential privacy for noise enhancement, providing more rigorous protection for location information. The algorithm first partitions into a uniform grid for spatial two dimensions data and adds Laplace noise with uniform scale parameter in each grid, then select the grid set to be optimized and recursively adaptively add noise to reduce the relative error of each grid, and make a second level of partition for each optimized grid in the end. Firstly, this algorithm can adaptively add noise according to the calculated count values in the grid. On the other hand, the query error is reduced, as a result, the accuracy of partition count query (the query accuracy of the differential private two-dimensional publication data) can be improved. And it is proved that the adaptive algorithm proposed in this paper has a significant increase in data availability through experiments.

Keywords: Data Publication, Privacy Protection, Differential Privacy, Noise Optimization.

1. Introduction

With the increasing development of location technology, location-based service has emerged for a long time [1]. User's geographic location information can be collected by spatial data set through mobile devices and then used for server analysis to allocate tasks and increase the efficiency of human works [2] [3]. For example, the takeaway service can select the optimal worker who is the best for this task (the distance is relatively short and the overhead is relatively small) by analyzing the location of takeaway task and the worker and calculating the distance between them. Online car-hiring service should determine the best match of a driver and a passenger. The weather application can infer the

climate in our city by using our location. Hence, the service providers can supply lots of services to help people better control their lives and make important decisions with user's location information collected by devices and platform. However, for many applications, people are told to contribute their exact location information, which may cause serious privacy disclosure problem. For example, when you submit your location to the application, the attacker knows your information such as religion, profession, age by analyzing the frequency of position where you appeared. In recent years, more and more attention are paid to the privacy protection that makes a challenge for us to protect user's private information when publishing data. Obviously, it is possible to achieve when the data is completely masked such as the function $f(x) = 0$ where lost its privacy. When the query is submitted by the analysis, the answer returns a value of 0 regardless of the count operation of any part of the data set. Meanwhile, although we can regard that we completely protect the privacy, we lose the availability of query data. As a result, it has become a hot issue for researchers to maximize the availability of published data. Because it's important to protect the user's two dimensions spatial data in privacy.

We consider the differential privacy (DP) [5] as a proper method of location-based privacy for issues we described. Differential privacy is a perfect model for privacy-preserving query and analyzes under the protection of user's privacy. Intuitively, it can ensure that for a single individual data included in a data set, the result of statistical query won't be significantly changed regardless of whether this individual is in the data set or not, which means the attacker cannot infer any individual data by the statistical result. DP focuses on two drawbacks compared with the traditional privacy protection model (such as encryption algorithm, k-anonymity [25] [5]). Firstly, DP proposes the definition of privacy protection based on strict mathematics that can provide privacy for the data sets in different levels. Secondly, DP model makes the maximum assumption for the background knowledge of attacker that it assumes the attacker knows all other records expect the target record, thus the differential privacy does not need to pay attention to the attacker's background knowledge. Above all, we argue the differential privacy model is the most suitable privacy-preserving method for location-based information.

For spatial data, such as individual location information in a certain area, we need to use a data publishing algorithm based on partition, which means Private Spatial Decomposition (PSD) [6]. Partition distribution is a form of differentially private data spatial publishing which basic idea is: firstly transform the original data, and then divide the data according to certain index construction rules, and publish data according to the index structure. Each index area is marked by a calculated count value, and noise is added for privacy protection.

There are two kinds of errors in the query result. The first type of error is called noise error [7]. In order to make area D to satisfy the differential privacy, we add noise to the area which makes the original area become D', and the error relative to D is counted as:

$$Error(P) = |Count(D) - Count(D')| \tag{1}$$

The second type is called uniform assumption error [7]. In the two-dimensional spatial data publication, it is often impossible to subdivide the two-dimensional space due to space limitations, and it is necessary to estimate the statistical value in one area. The commonly used estimation method is the assumption that the points within the area are uniformly distributed. In that case, the estimated value is returned according to the ratio

of the query area. The error due to uniform distribution estimation becomes a uniform assumption error.

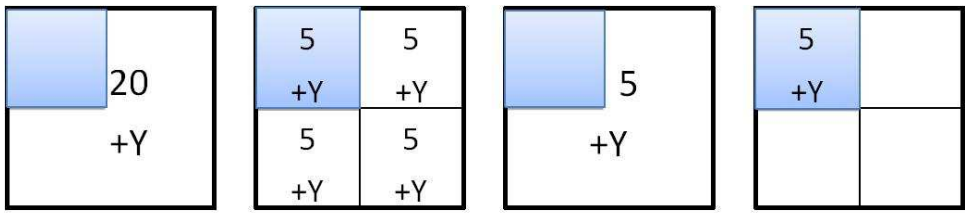

Fig. 1. Noise-added domain. (a) uniform non-partition. (b) uniform partition. (c) non-uniform non-partition. (d) non-uniform partition.

As shown in Fig 1, assume that (a) and (b) are uniformly distributed, (a) the noise is added to the area,(b) the noise is added after the area is partitioned. The numbers 5, 20 represent real statistics count for the area, +Y denotes the noise added to make this area satisfy differential privacy, and the blue district means the query area. Since the privacy budget is the same, the expected noise about two graphs (a) and (b) is the same. For the query result, (a) is (20+Y) / 4, and (b) is (5+Y). It is easy to see that the query error in the graph (b) is larger than in graph (a). However, for Figures (c) and (d), the query results are (5+Y) / 4 and (5+Y). Although Figure (c) reduces the noise error, it also increases the uniform assumption error. So under normal circumstances, the noise error and the uniform assumption error are opposite to each other. Utilizing the uniform assumption error estimate can reduce the noise error of the query result, but it will produce additional uniform assumption errors. However, subdivided regions can reduce the uniform assumption error of query results but increase the noise error. So how to balance the two kinds of errors in order to minimize the publishing error is a problem we need to discuss.

The methods described in the third section of this article all use tree structures to index sensitive data. However, the tree structure does not involve the issue of minimizing publishing errors. Literature [8] proposed uniform grid (UG) and adaptive grid (AG) to minimize publishing errors. The publishing errors mainly include the added Laplace noise error and the uniform assumption error. UG and AG quantify the two parts of the error and get the grid partition density. The UG method partitions the spatial data set equally. However,we argue that the same partition method for sparse data and dense data will bring greater uniform assumption error. On that basis, the author proposes an AG method to adaptively partition the grid according to the data density. The disadvantage of the UG and AG methods is that the same privacy budget is allocated for sparse data and dense data whose effect on the noise error of them are different. Further consideration should be given to the adaptive allocation of privacy budgets based on data point density.

We propose SGNO (Self-adaptive Grid-partitioning Noise Optimization algorithm) that focuses on geo-spatial data summarizing related domestic and foreign research status and existing algorithms. On one hand, the released data satisfies privacy protection, on

the other hand, the availability of published data(the similarity between original data and published data) is improved. We prove that the effectiveness of the algorithm is obtained in the experimental part compared with the relevant algorithm.

The rest of this paper is organized as follows: In section 2 we introduce the preliminaries and the related work. We propose our optimized algorithm for spatial decomposition and evaluate the algorithm compared with some of the previous ones in section 3 and 4. We conclude the paper in section 5.

2. Related Work

Differential privacy was originally proposed in [4] to protect the results of queries. To preserve the privacy of individuals, Mechanisms must guarantee that the contribution of each individual for a query result cannot disclose data. We introduce the related technology in this paper in detail, including the notion of differential privacy and the data publishing method based on partition.

2.1. Differential Privacy Model

The main idea of the differential privacy protection model is to perturb the data by random noise before it is published. Hence, even if an attacker knows all the other record information except the target record, the individual user's private information cannot be inferred through data mining and data analysis. In addition, differential privacy has a strict mathematical model, which can provide different degrees of privacy protection for data sets according to user requirements. This can protect users' privacy information in various types of background knowledge attacks. Hence, the differential privacy protection model has been studied and perfected by scholars since its proposed, and gradually has a rigorous theoretical system [9]. Since differential privacy is defined on neighbor data sets, it is necessary for us to introduce the definition of neighboring data sets first.

Definition 1 (Neighboring data set) For two data sets D_1 and D_2 with the same attribute structure, D_1 and D_2 are called neighboring data set, if and only if there is one different data record in D_1 and D_2.

We give the definition of differential privacy based on the neighboring data set.

Definition 2 (ε-differential privacy) A randomized algorithm A gives ε-differential privacy, for any pair of neighboring data sets D_1 and D_2 and for every set of outcomes O (O $\in$ Range(A)), A satisfies:

$$Pr[A(D_1) = O] \leq e^{\varepsilon} \cdot Pr[A(D_2) = O] \tag{2}$$

The ε in formula 2 is called privacy budget which represents the level of the privacy protection. The smaller the ε is, the higher the protection level is. In order to make the results of $A(D_1)$ and $A(D_2)$ have higher similarity, the data needs to be disturbed to a higher level, so the privacy protection level is improved and the availability of data is reduced [10]. On the contrary, the larger the value of ε, the less level of data disturbance will lead to better data availability, which may cause lower privacy protection. Nowadays, there is no good standard for the value of ε. Generally, the optimal value is constantly adjusted according to the level of privacy protection. It is always between $\ln 2$ and $\ln 5$ based on empirical evaluation.

The main idea of the differential privacy protection model is to hide the impact of a single data record on overall published data. It uses the idea of data perturbation to transform the single record by guaranteeing the invariance of the overall data on the probability to achieve the purpose of protecting the user's private data. Figure 2 shows a statistical model of differential privacy, which illustrates the results of the privacy protection model implemented on a neighboring data set.

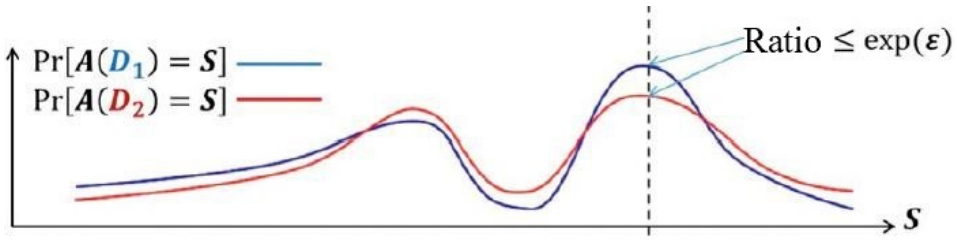

Fig. 2. Statistical Model of Differential Privacy

According to the definition of privacy we can know that it is used in data distribution mechanisms rather than directly applied to the data set. Intuitively, it can show that the behavior of differential privacy protection model on any two neighboring data sets are roughly the same in addition that the presence or absence of an individual does not affect the output of the algorithm. For example, there is a simplified medical record data set D_1 in which each record represents (name, diabetes), the second line is a boolean, 1 represents illness, and 0 represents no disease, as shown in Table 1. We assume that the opponent wants to know if Jack is sick or not and he also knows the number of rows in Jack's database. Suppose the opponent's query form is Q_i, which represents the sum of the first I rows of the $Diabetes$. In order to know Jack's prevalence, opponents perform queries Q_5 (D_1) and Q_4 (D_1) and then calculate their difference to know that Jack's disease status corresponds to 1. This example shows that personal information may be affected even if no specific personal information is queried. If we construct the data set D_2 by changing the last record to (Jack, 0) of table 1, the opponent can distinguish two neighboring data sets D_1 and D_2 by calculating Q_5-Q_4. If the opponent gets the query value Q_i through ε-differential privacy, he cannot distinguish between two neighboring data sets after selecting the appropriate value of ε.

Achieving differential privacy generally adopts two mechanisms: the Laplace mechanism and the exponential mechanism. These mechanisms contains the definition of sensitivity. To make these definition understood easily, we give an introduction of global sensitivity [11].

Definition 3 (Global sensitivity) For a function $f : D \to R^d$, for any neighboring data sets, the global sensitivity $\triangle f$ of function f is defined:

$$\triangle f = \max_{D_1 D_2} ||f(D_1) - f(D_2)||_1 \tag{3}$$

Where D refers to the data set, R_d refers to d-dimensional vector, d is a positive integer, $||f(D_1) - f(D_2)||_1$ represents the 1-order distance between $f(D_1)$ and $f(D_2)$.

Table 1. Medical Dataset of Diabetic Patients

Name	Diabetes
Ross	1
Bob	1
Phoebe	0
Carol	0
Jack	1

Global sensitivity represents the change in the output of the algorithm when changing any record in the data set.

Laplace Mechanism The Laplace mechanism achieves differential privacy by adding noise that obeys the Laplace distribution to the original real query value. As shown in Figure 3, the probability density function of the Laplace distribution is:

$$f(x|\mu, b) = \frac{1}{2b} \cdot e^{-\frac{|x-\mu|}{b}} \tag{4}$$

Where μ is position parameter, b refers to scale parameter, whose value is up to 0. When the parameter changes, the Laplace probability distribution function changes as the Figure shows.

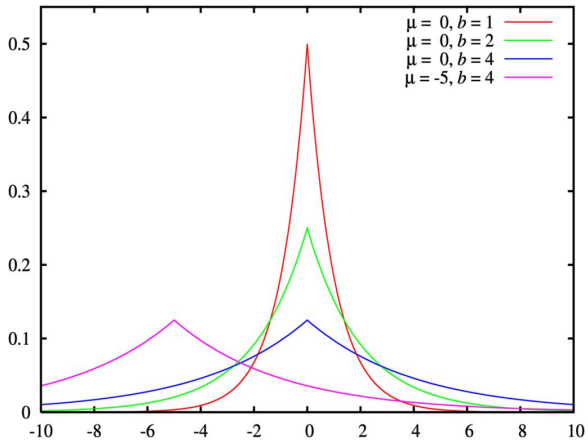

Fig. 3. Probability Density Function of Laplace Distribution

Definition 4 (Laplace mechanism) Given a function $f : D \to R^d$ over a data set D, a privacy budget ε and the global sensitivity $\triangle f$, f satisfies Laplace mechanism when:

$$L(D) = f(D) + Lap(\frac{\triangle f}{\varepsilon}) \tag{5}$$

Where $Lap(\lambda)$ means the position parameter of Laplace distribution is 0, and the scale parameter is λ, same as $\frac{\triangle f}{\varepsilon}$.

A Laplace mechanism is used for the numeric query result [12]. We can add noise to the output value of original data set to affect the result of query to protect the data set. We can know that the smaller ε corresponding larger noise value according to taking different parameter values. The noise size of the Laplace mechanism needs to achieve a good balance between the protection level and practical application according to actual requirements [13].

Composition Theorems We need to consider the privacy budget of the entire process to be controlled within ε, because a complex privacy protection scenario often requires multiple applications of the differential privacy protection model. As a result, we need to apply the two composition theorem of the differential privacy protection model [14,15]

Definition 5 (Sequential composition)

Assume a set of algorithms $A_1(D), A_2(D), ..., A_n(D)$ whose privacy budgets are $\varepsilon_1, \varepsilon_2, ..., \varepsilon_n$ respectively that satisfy differential privacy. For the data set D, a new algorithm A composed of the above algorithms $A(A_1(D), A_2(D), ..., A_n(D))$, gives $\sum_{i=1}^{n} \varepsilon_i$-differential privacy model. This theorem indicates that the privacy protection level of combined algorithms is the sum of all budgets in a differential privacy protection model sequences [12].

Definition 6 (Parallel composition)

Assume a set of algorithms $A_1(D), A_2(D), ..., A_n(D)$ whose privacy budgets are $\varepsilon_1, \varepsilon_2, ..., \varepsilon_n$ respectively that satisfy differential privacy. For the data set D which can be partitioned into disjoint subcell data sets $D_1, D_2, ..., D_n$, a new algorithm A composed of the above algorithms $A(A_1(D), A_2(D), ..., A_n(D))$, gives $\max(\varepsilon_i)$-differential privacy model.

In a differential privacy protection model sequences, the privacy protection level of combined algorithms is determined by the algorithm with the largest privacy budget, where the data sets input by each algorithm are disjoint. These two kinds of composition theorems play an important role in the proof of the differential privacy protection model algorithm. At the same time, the algorithm that satisfies differential privacy can also be partitioned into sub-algorithms, and then the sub-algorithm gets the corresponding privacy budget. For example, algorithm A's privacy budget is ε. If A is divided into two processes, A_1 and A_2, then privacy budget ε can be divided into ε_1 and ε_2 and then assigned to A_1 and A_2 respectively. We only need to make A_1 and A_2 meet the privacy protection levels of ε_1 and ε_2 to satisfies differential privacy.

Metrics We usually use the degree of difference between the original and the noise-added data set to evaluate the algorithm. The commonly used error metrics are: relative error [16], absolute error [17], error variance [18], and Euclidean distance [19]. We use relative error for evaluations in this paper.

In addition, ε represents the degree of privacy protection, so it is also important to choose appropriate allocate strategies. Common allocation strategies include linear allocation, uniform distribution, exponential allocation, adaptive allocation, and mixed strategy allocation [20].

The previous partition method is mainly divided into two categories: tree-based spatial decomposition and grid-based spatial decomposition. We will introduce these two methods in detail as follows [21].

2.2. Tree-Based Spatial Decomposition

The tree-based data distribution method is a method for hierarchically decomposing spatial data. The data points are partitioned into leaf nodes so that the leaf nodes can contain a small number of data points or a small data range. Non-leaf nodes represent the sum of the counts of children's nodes. Unless otherwise specified, we assume that the tree structure is a complete binary tree that all root-to-leaf paths have the same length, and all internal nodes have the same output.

The tree-based data distribution methods can be roughly divided into two categories: data-independent partitioning and data-dependent partitioning. We define the data independent partitioning if the area is divided regardless of involving the basic data. On the contrary, when it is divided on the basis of the data, it is called the data-dependent partitioning, which may reveal data privacy [21].

Data-independent Tree Partitioning For the 2D spatial data, a representative example of the data-independent tree division is quadtree, which can be extended to high dimensions named octree and other structures to represent data. Their feature is to set the partitioning method in advance and based on the attribute domain.

Data-dependent Tree Partitioning The representative structures of data-dependent tree structure are kd-tree and Hilbert R-trees which mainly depends on the input. We focus on the construction of kd-tree and the process of combining it with differential privacy.

The main construction process of kd-tree is divided into two steps: (1) select the dimension k with the largest variance in the k-dimensional data set, and then select the median m in this dimension as the pivot to partition the data set to obtain two Subsets, while creating a tree node for storing the total calculated count values. (2) Repeat step (1) for the two subsets until all subsets can no longer be divided. We store the data of the subset to the leaf node until it can't be partitioned anymore [22].

The algorithm of using differential privacy based on kd-tree is called kd-standard [23]. Kd-standard's privacy budget is divided into two parts: First, we need to determine the median value. The segmentation line may leak the true value of the median value if you do not use differential privacy to protect the segmentation process. Then we add Laplace noise to each level calculated count value of the kd-tree. Kd-standard may also have two problems with the above-mentioned quadtree: inconsistent query results and non-uniform allocation of privacy budgets. The solution is the same as a quadtree.

Mixed Tree Partitioning The mixed tree combines the data-independent and data-dependent tree partitioning methods, in which kd-hybrid is representative. We use the tree-

dependent partitioning method in the previous l layer (l is set in advance). Then select the median with the maximum variance dimension as the pivot to divide recursively each time [24]. The rest of the tree uses a tree-independent partitioning method that sets up the partitioning process in advance. This algorithm makes the advantages and disadvantages of the two tree partitioning methods complementary, so the query results are more accurate.

2.3. Grid-Based Spatial Decomposition

In this section, we introduce the previous research achievements on the spatial data release and then gradually improve the method.

Uniform Grid Partition (UG) UG partitioning is a relatively simple way to partition the space. This method divides the data domain into $m * m$ equally sized grids, and then adds noise to a calculated count value for each grid, where m is obtained by minimizing the sum of the noise error and the uniform assumption error. The disadvantage of UG partitioning is that all areas in the data set are treated equally where dense and sparse areas are partitioned in the same way. If there are fewer data points in a region, this method will result in the region being divided too much, which increases the noise error and hardly reduces the uniform assumption error. In addition, if an area is dense, uniform grid partition can make this area too rough, which in turn leads to large uniform assumption errors. Therefore, when the area is dense, a fine-grained partitioning method should be used because the uniform assumption error in this area far exceeds the noise error. Similarly, if a region is sparse, coarse-grained partitioning is used. To overcome this problem, researchers proposed Adaptive Grids (AG) partitioning based on the uniform grid method.

Adaptive Grid Partition (AG) The AG first performs uniform grid partition in a smaller granularity which is set as $\max(10, \frac{1}{4}\lceil\frac{N\cdot\varepsilon}{c}\rceil)$ since there is a second grid division, and the privacy budget of the first layer is $\varepsilon_1 = \varepsilon \cdot \alpha$. Then, on the basis of the noise-added calculated count value N of the first layer of the grid, the AG further adaptively selects the division granularity of the second layer grid. We find that AG improves the accuracy compared with UG obviously on the second layer.

The advantage of AG partition method is that it can quantify two error sources on the basis of differential privacy to calculate the partition granularity. However, we find the drawback that AG does not adaptively allocate the privacy budget. According to the method proposed by Dwork [4], AG ignore the size of the query answer and add Laplace noise to each answer with the same scale parameters. The proposed method is more susceptible to the amount of noise which may lead to a large relative errors especially when the calculated count value of the query is small. For example, Figure 4 shows a part area of the AG partitioning results: The two numbers in each grid are the original calculated count value and the added noise value, where the noise follows the Laplace distribution with the same scale parameters. For the dashed line area which represents the user's query area, the corresponding noise added query result is 21.1 when the original query result is 11, so the query error can be calculated as $(21.1 - 11)/10 = 1.11$. Therefore, due to the accumulation of noise in the query area, the availability of published results is very low. We will focus on the shortcomings of AG and provide the corresponding solutions in Section 3.

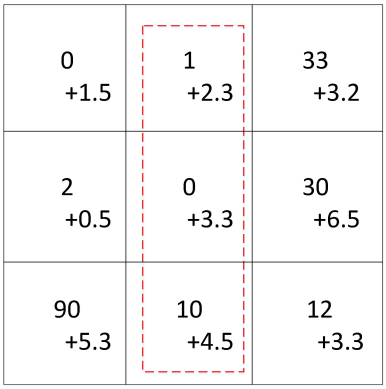

Fig. 4. Example of Injecting Noise into Grid Partition

3. Adaptive Grid-partitioning Noise Optimization Algorithm

In this section, we will introduce the improved algorithm in detail for the adaptive grid-partitioning. We introduced the quantification formulas for UG and AG, and illustrated their existing problems in related work. In that case, we proposed an optimized algorithm based on AG to solve these problems.

3.1. The Problem and Notions

The adaptive grid noise added publishing algorithm we proposed mainly addresses two aspects: (1) each grid adds Laplace noise with the same scale parameters (2) after the second-level grid is generated, the first level no longer provides useful information for the data set that may waste privacy budget. In response to the above problems, the corresponding solutions are presented as follows: First, to avoid receiving a larger relative error when querying a very small value. we propose a method to reduce the relative error by adjusting the noise scale parameter of the counter value. Second, in order to save the first-level grid privacy budget, we generate the value in second-level grid by sampling from first-level grid noise calculated count value.

Before proposing the overall steps, we first introduce some parameters and the calculation formula adjusted according to the differential privacy definition. $G = [N_1, ..., N_{m_1 * m_1}]$ is the set of count values for the first-level grid partitioning. $\Lambda = [\lambda_1, ..., \lambda_{m_1 \cdot m_1}]$ is a set of positive real numbers, corresponding to the noise scale parameter λ_i of each N_i,$(i \in [1, m_1 * m_1])$. $Y = [y_1, ..., y_{m_1 * m_1}]$ is a set of count values after adding noise to G in same scale parameter, where $y_i = g_i(G) + \eta_i$ and η_i is sampled from the Laplace distribution with scale parameter $i(i \in [1, m_1 * m_1])$. $G' = [Y'_1, ..., Y'_{m_1 * m_1}]$ is the result of optimized noise added method after the use of adaptive noise adding algorithm.

The steps of the algorithm are as follows: In first step we perform a two-dimensional data set into first-level uniform grid partition with partition granularity $m_i = \max(10, \frac{1}{4}\sqrt{\frac{N \cdot \varepsilon}{c}})$. The choice of the value c mainly depends on the uniformity of the data set. Then we add the Laplace noise in a same uniform scale parameter to

the grid set G generated in the first step to obtain the noise-added grid calculated count value set Y. Next the grid sets to be optimized from Y are selected recursively and optimized, where we use a noise optimization algorithm to satisfy the difference privacy guarantee in the meanwhile. It construct an optimized noise calculated count value set G'. Finally we perform adaptive grid partition on the basis of G' with the granularity of $m_2 = \lceil \sqrt{\frac{Y' \cdot \varepsilon_2}{c_2}} \rceil$, where $c_2 = \frac{c}{2}$. The calculation process of m_2 and c_2 are as follows: When the grid in G' is further divided into second $m_2 * m_2$ layer grids, only those queries whose query boundaries pass through the first layer grids are affected. These queries may contain 0, 1, 2... $m_2 - 1$ rows (or columns) of the second-level grids, and therefore correspond to 0, m_2, $2m_2$...$(m_2 - 1)m_2$ grids in second-level. When the query contains more than half of the second-level grid, the query is answered using Constraint Reasoning, which uses the calculated count value of the first layer minus the calculated count value in the second-level grid that is not included in the query. Therefore, the query uses an average of $\frac{1}{m_2}(\sum_{i=0}^{m_2-1} \min(i, m_2 - 1)) * m_2 \approx \frac{m_2^2}{4}$ second-level grids, which means that the average noise error is approximate $\sqrt{\frac{m_2^2}{4}} * \frac{\sqrt{2}}{\varepsilon_2}$. In addition, the mean value of the uniform assumption error is approximately $\frac{Y'}{c_0 * m_2}$. Finally, we minimize the sum of the average noise error and the uniform hypothesis error to get the minimum value of m_2 is $\lceil \sqrt{\frac{Y' \cdot \varepsilon_2}{c_2}} \rceil$, where $c_2 = \frac{c}{2}$.

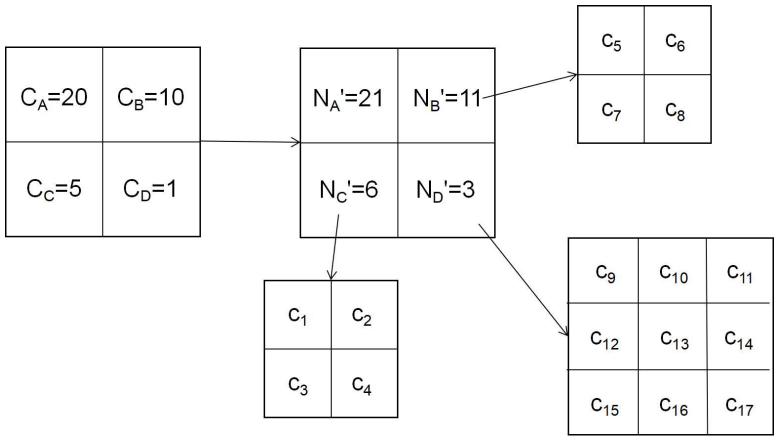

Fig. 5. Example of SGNO Partition (ε=0.5,α=0.5)

Figure 5 shows the partitioning process of SGNO, where A, B, C, and D represent the four first-level grids. Constructing differential privacy SGNO requires three steps: First, calculate the partition granularity of the first-level grid, where $c = 10$, and then add the Laplace noise with the scale parameter to the original values of A, B, C, and D to obtain the noise calculated count value N'. Secondly, we select the grid sets to be optimized from N' recursively using a noise optimization algorithm satisfying the differential privacy guarantee. The selected grid sets construct an optimized noise calculated count value set G'. In figure 5, after the second step, the N'_A is not included in the grid sets G', Next,

calculate the granularity of the second-level grid on the basis of G', where $c_2 = 5$. We use the above formulas to obtain different partition granularity for different noise-added count values N', and add Laplace noise with scale parameter to each second-level grid. Finally, SGNO structure is published with noise count.

3.2. Noise Optimized Partitioning

The principle of the NOP (Noise Optimization Partitioning) [16] is to add a uniform scale parameter $\lambda_i (1 \leq i \leq m_i * m_i)$ to the second layer of the grid to obtain a set of noise-added counts Y, and then adjust the noise scale parameter set $\Lambda = [\lambda_1, ..., \lambda_{m_1 * m_1}]$ continuously under the constraint of differential privacy to generate updated set of noise count Y'. The key question of this algorithm is whether Y' is to add noise to the original value $N_i (1 \leq i \leq m_i * m_i)$ through a new set of scale parameters, or to adjust the noise value set to generate Y' on the basis of Y, so next we consider for both cases separately.

If the first and the second level of the grid noise adding process are independent, then the total privacy budget is $2/\lambda + 2/\lambda'$, where the privacy budget of the second layer grid is $2/\lambda'$. However, the result of the first level grid partitioning is useless after the generating of the second layer grid, which means the privacy budget used by the first level grid is wasted. To solve the above problems, we argue that the noise-added value of the second level grid obeys Laplace distribution and is sampled from the distribution that depends on the first level grid, which can save part of the privacy budget.

We use formula derivation to explain the process of reducing the privacy budget: First, for the data set T_1, the first layer of partitioning produces a result of Y, and the second produces a result of Y'. Y's result is useless after Y' is generated which can quantify as the following formula:

$$Pr[T = T_1 | Y = y, Y' = y'] = Pr[T = T_1 | Y' = y'] \tag{6}$$

Formula 6 allows us to use the privacy budget more efficiently. When Equation 6 is satisfied, we can apply the Bayesian equation twice to derive the following derivation for two neighbor data sets T_1 and T_2:

$$\begin{aligned} & Pr[Y' = y', Y = y | T = T_i] \\ =& Pr[T = T_i | Y' = y', Y = y] \cdot \frac{Pr[Y' = y', Y = y]}{Pr[T = T_i]} \\ =& Pr[T = T_i, | Y' = y'] \cdot \frac{Pr[Y' = y', Y = y]}{Pr[T = T_i]} \\ =& \frac{Pr[Y' = y' | T = T_i] Pr[T = T_i]}{Pr[Y' = y']} \cdot \frac{Pr[Y' = y', Y = y]}{Pr[T = T_i]} \\ =& Pr[Y' = y' | T = T_i] \cdot Pr[Y = y, Y' = y'] \end{aligned} \tag{7}$$

Next, we consider the case where the Y and the Y' generated by the first-level and second-level grid with noise added are dependent. For a count query $q, q(T_1) - q(T_2) \in \{-1, 0, 1\}$. Y' is a random variable and satisfies two conditions: First, it obeys the Laplace

distribution with the positional parameter $q(T)$ and the scale parameter λ'. Second, it is dependent on Y. Equation 6 and 7 get the following derivation process:

$$\begin{aligned}&\frac{Pr[Y'=y', Y=y|T=T_1]}{Pr[Y'=y', Y=y|T=T_2]}\\ =&\frac{Pr[Y'=y'|T=T_1]\cdot Pr[Y=y|Y=y']}{Pr[Y'=y'|T=T_2]\cdot Pr[Y=y|Y=y']}\\ =&\frac{Pr[Y'=y'|T=T_1]}{Pr[Y'=y'|T=T_2]} \leq \exp(2/\lambda')\end{aligned} \tag{8}$$

From equation 8 we can get an upper bound of the privacy budget. which means we can achieve Y' on the basis of Y. The total privacy budget is only $2/\lambda'$ to ensure no privacy budget is wasted on Y.

From the above process, we learned that if Y obeys the Laplace distribution and samples from the Y-dependent distribution, it can save some privacy overhead. So we define the conditional probability distribution function for Y as follows:

$$\begin{aligned}&f_{\mu,\lambda,\lambda'}(y'|Y=y)=\frac{\lambda}{\lambda'}\cdot\frac{\exp(-\frac{|y'-\mu|}{\lambda'})}{\exp(-\frac{|y-\mu|}{\lambda})}\cdot\gamma(\lambda',\lambda,y',y)\\ &\gamma(\lambda',\lambda,y',y)=\frac{1}{4\lambda}\cdot\frac{1}{\cosh(\frac{1}{\lambda'})-1}\cdot(2\cosh(\frac{1}{\lambda'}\cdot\exp(-\frac{|y-y'|}{\lambda})-\exp(-\frac{|y-y'-1|}{\lambda})-\exp(-\frac{|y-y'+1|}{\lambda}))\end{aligned} \tag{9}$$

The probability density function of Y' can be further obtained from the conditional probability density function. When $\mu \leq y$, $\xi = \min\{\mu, y-1\}$. We can obtain the following formula according to the conditional probability density function for $y' \leq \xi$.

$$\begin{aligned}&f(y')=e^{y'/\lambda'}\cdot\gamma(\lambda',\lambda,y',y)\cdot\frac{\lambda}{\lambda'}\cdot\exp(\frac{-\mu}{\lambda'}+\frac{y-\mu}{\lambda})\\ &\gamma(\lambda',\lambda,y',y)=e^{y'/\lambda}\cdot\frac{1}{4\lambda}\cdot\frac{1}{\cosh(\frac{1}{\lambda'})-1}\cdot(2\cosh(\frac{1}{\lambda'}\cdot\exp(-\frac{-y}{\lambda})-\exp(-\frac{1-y}{\lambda})-\exp(-\frac{-1-y}{\lambda}))\end{aligned} \tag{10}$$

Formula 7 can be simplified as $f(y') \propto \exp(y'/\lambda + y'/\lambda')$, meanwhile, we can obtain $y' \in (\xi, y-1]$, $f(y') \propto \exp(y'/\lambda - y'/\lambda')$ and $y' \in (y+1, +\infty]$, $f(y') \propto \exp(-y'/\lambda - y'/\lambda')$. Based on this, the random variables $\theta_1, \theta_2, \theta_3$ that follow the probability density function f can be calculated and their formula is as follows:

$$\begin{aligned}\theta_1 =&\int_{-\infty}^{\xi} f(y')dy'\\ =&\frac{\lambda\cdot(\cosh(\frac{1}{\lambda'})-\cosh(\frac{1}{\lambda})\cdot\exp(\frac{1}{\lambda'}+\frac{1}{\lambda})\cdot(\xi-\mu))}{2(\lambda'+\lambda)\cdot(\cosh(\frac{1}{\lambda'})-1)}\end{aligned} \tag{11}$$

$$\begin{aligned}\theta_2 =&\int_{\xi}^{y-1} f(y')dy'\\ =&\frac{\lambda\cdot(\cosh(\frac{1}{\lambda'}))\cdot(e^{\frac{1}{\lambda'}}-e^{\frac{1}{\lambda}})\cdot(1-e^{-\frac{1}{\lambda'}-\frac{1}{\lambda}})}{4(\lambda-\lambda')\cdot(\cosh(\frac{1}{\lambda'})-1)}\cdot(1-\exp((\frac{1}{\lambda'}-\frac{1}{\lambda})\cdot(\xi-y+1))\end{aligned} \tag{12}$$

$$\begin{aligned}\theta_3 &= \int_{y+1}^{+\infty} f(y')dy' \\ &= \frac{\lambda \cdot (\cosh(\frac{1}{\lambda'}) - \cosh(\frac{1}{\lambda}) \cdot \exp(\frac{\mu-y-1}{\lambda'} - \frac{\mu-y+1}{\lambda}))}{2(\lambda'+\lambda) \cdot (\cosh(\frac{1}{\lambda'}) - 1)}\end{aligned} \tag{13}$$

For the remaining space $(y-1, y+1)$, we obtain its probability density function through the standard importance sampling function. We will introduce the specific process in the algorithm. The formula of φ in the algorithm 1 is as follows:

$$\varphi = \frac{1}{2\lambda'} \cdot \frac{\cosh(\frac{1}{\lambda'}) - \exp(-\frac{1}{\lambda})}{\cosh(\frac{1}{\lambda'} - 1)} \cdot \exp(\frac{y-\mu}{\lambda} - \frac{\max\{0, y-\mu-1\}}{\lambda'}) \tag{14}$$

The algorithm 1 represents a specific process of NOP, where the input is original calculated count value μ, a noise-added calculated count value y, an original scale parameter λ, and an adjusted scale parameter λ', and the output is an updated y.

3.3. Self-adaptive Grid-partitioning Noise Optimization Algorithm

We use the algorithm 2 to describe the grid-based adaptive noise-added publishing method in detail. The pseudocode of this algorithm is shown in Algorithm 2. The input of the algorithm is data set T, privacy budget ε, initial privacy budget λ_{max}, and privacy budget variance λ_Δ. The output is Adaptive Grid AG. The algorithm first averages the data set to get a uniform grid set UG and an original privacy set Λ. Then the grid to be optimized is selected recursively from the UG. We adjust the scale parameters and complete the noise optimization process with former algorithm. If this process does not satisfy the differential privacy, the changes made to the noise scale parameters are restored. Finally, AG is adaptively partitioned with the updated UG set.

We need to explain the process of selecting the grid to be optimized. Ideally, running a noise optimization algorithm on a selected set of grids may reduce the overall more error and make slightly lower privacy overhead, so the criteria for PickQueries function selection grids is to maximize the ratio between the value of overall error reduction and privacy budget increase value. First, calculate the privacy budget increase value. Each grid has the same scale parameter $\lambda_i (i \in [1, m_1 * m_1])$ at first with the global sensitivity $g_1 = \sum_{i \in [1, m_1 * m_1]} \frac{2}{\lambda_i}$. Then we use the noise optimized algorithm for the selected $j-th$ grid, and the corresponding global sensitivity is changed to $g_2 = \frac{2}{\lambda_j - \lambda_\Delta} + \sum_{i \in [1, m_1 * m_1] \wedge i \neq j} \frac{2}{\lambda_i}$, the privacy budget applied by the noise optimized algorithm is $g_2 - g_1 = \frac{2}{\lambda_j - \lambda_\Delta} - \frac{2}{\lambda_j}$. Second, we calculate the total error reduction value. The relative error of each grid is $\frac{\lambda_i}{max\{y_i, \delta\}} (i \in [1, m_1 * m_1])$. Then, the relative error of grid UG is $\sum_{i \in [1, m_1 * m_1]} \frac{\lambda_i}{max\{y_i, \delta\}}$. After applying the noise optimized algorithm, the total relative error becomes $\frac{\lambda_j - \lambda_\Delta}{max\{y_i, \delta\}} + \sum_{i \in [1, m_1 * m_1] \wedge i \neq j} \frac{\lambda_i}{max\{y_i, \delta\}}$, and the change of total relative error is obtained to $\frac{2\lambda_j - \lambda_\Delta}{max\{y_i, \delta\}}$, so that the average relative error variation is $\frac{1}{m_1 \cdot m_1} \cdot \frac{2\lambda_j - \lambda_\Delta}{max\{y_i, \delta\}}$. Therefore, the ratio between the overall error reduction value and the privacy budget increase value is $\frac{1}{m_1 \cdot m_1} \cdot \frac{2\lambda_j - \lambda_\Delta}{max\{y_i, \delta\}} / (\frac{2}{\lambda_j - \lambda_\Delta} - \frac{2}{\lambda_j})$. So we choose the grid that can maximize this ratio as the set of grids to be optimized.

Algorithm 1: NOP (μ, y, λ, λ')

Input: $\mu, y, \lambda, \lambda'$
Output: y
Initial: $mark = true$
if $\mu > y$ **then**
    μ=-μ,y=-y
    $mark = false$
$\xi = \min\{\mu, y - 1\}$
generate a random variable u uniformly distributed in $[0, 1]$
if $\mu \in [0, \theta_1]$ **then**
    $f(y') \propto \exp(y'/\lambda + y'/\lambda')$ // y' *is a random variable generated from* $(-\infty, \xi]$;
    else
        if $\mu \in [\theta_1, \theta_1 + \theta_2]$ **then**
            $f(y') \propto$
            $\exp(y'/\lambda' - y'/\lambda)$ // y' *is a random variable generated from* $(\xi, y - 1]$;
    else
        if $\mu \in [1 - \theta_3, 1]$ **then**
            $f(y') \propto \exp(-y'/\lambda -$
            $y'/\lambda')$ // y' *is a random variable generated from* $(y + 1, \infty)$;
    else
        while $true$ **do**
            generate a random variable y' uniformly distributed in (y-1,y+1)
            generate a random variable μ' uniformly distributed in [0,1]
            if $\mu' \leq f(y')/\varphi$ **then**
                $break$;
if $mark = true$ **then**
    return y'
    else
        return $-y'$

3.4. Privacy Analysis

We propose self-adaptive grid-partitioning noise optimization algorithm after the first-level partition. The algorithm first adds Laplace noise with uniform scale parameters to each grid. Then continue the process if the grid satisfies the ε-differential privacy, otherwise return empty set. Next, the grid to be optimized is recursively selected in the grid set, and the corresponding noise scale parameters are changed. We judge the condition that whether the ε-differential privacy is satisfied. If it is satisfied, the noise optimized algorithm is continued to be called; Otherwise, the changes made to the noise scale parameters are restored to satisfy the ε-differential privacy requirements. In summary, the proposed algorithm generally satisfies ε-differential privacy.

Algorithm 2: $SGNO\ (T, N, \delta, \varepsilon, \lambda_{max}, \lambda_{\Delta}, \alpha)$

Input: $Dataset\ T$, $the\ sanity\ bound\ \delta$, $privacy\ budget\ \varepsilon$, $constant\ \lambda_{max}, \lambda_{\Delta}$,
Output: $Adaptive\ Grids\ AG$
Initial: $let\ T\ be\ partitioned\ uniformly\ with\ m_1 = \max(10, \frac{1}{4}\lceil\frac{N\varepsilon}{c}\rceil)$
$let\ m = m_1 * m_1\ and\ g_i\ be\ the\ i-th(i \in [1, m,])\ grid\ in\ UG$
$initialize\ \Lambda = [\lambda_1, ..., \lambda_{m_1 \cdot m_1}], such\ that\ \lambda_i = \lambda_{max}$
if $GS(UG, \Lambda) > \varepsilon$ **then**
    $return\ \emptyset$
$Y = LaplaceNoise(UG, \Lambda)$ // *Add Laplace noise to every grid in UG*
$Let\ UG' = UG$
while $UG' \neq \emptyset$ **do**
    $U_{\Delta} = PickQuries(UG', Y, \Lambda, \delta)$
    for $i\ from\ 1\ to\ m$ **do**
        if $g_i \in U_{\Delta}$ **then**
            λ_i=λ_i-λ_{Δ}
    if $GS(UG, \Lambda) \leq \varepsilon$ **then**
        for $i\ from\ 1\ to\ m$ **do**
            if $g_i \in U_{\Delta}$ **then**
                y_i=$NOP(g_i, y_i, \lambda_i + \lambda_{\Delta}, \lambda_i)$
        else
            for $i\ from\ 1\ to\ m$ **do**
                if $g_i \in U_{\Delta}$ **then**
                    λ_i=λ_i+λ_{Δ}
        $UG' = UG \setminus U_{\Delta}$
$Update\ UG\ by\ Y$
$let\ UG\ be\ partitioned\ by\ m_2 = \lceil\frac{Y'(1-\alpha)\varepsilon}{C_2}\rceil\ to\ get\ AG$
Return AG;

4. Evaluation

In this section we compare the effectiveness of the algorithm proposed in section 4 with some previous methods. We introduce the process and results in detail.

4.1. Environment

Experiment Platform Table 2 shows the relevant configuration information of the experiment platform. We have implemented the encoding algorithm proposed in section 4 in the Linux operating system.

Experiment Database We chose three data sets for spatial data and conducted experiments separately because our two data publishing algorithms are applied to different forms of data sets.

Table 2. Configuration of Experiment Platform

Hardware and software	Configuration
Processor	Intel@ Xeon@ CPU E5-2670 2.27 GHz
Internal storage	32GB
Hardware	1TB Mechanical hard disk
Network card	Intel 82551 10M/100M Adaptive network card
OS	Ubuntu Server 16.04.1 LTS

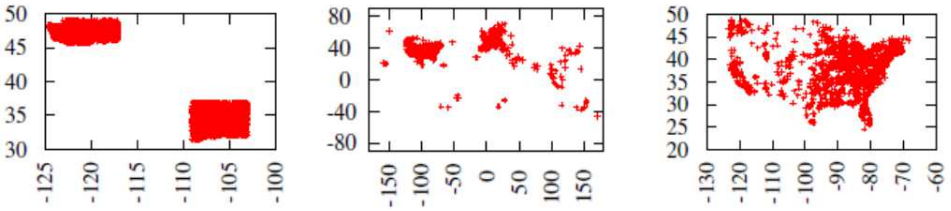

Fig. 6. Illustration of datasets

In order to verify our proposed grid-based adaptive noise-added publishing algorithm, we used three datasets shown in Figure 6.

(1)$Road$: This data set consists of the GPS coordinates of road intersections in Washington State and New Mexico and is derived from the 2006 U.S. census data in the TIGER/Line data set. There are approximately 1.6 million data points in the data set, roughly corresponding to human activities. As shown in the first picture of Figure 6, the distribution of data points is somewhat special. Two data points are dense where is distributed in two states, and almost no data points in there.

(2)$Checkin$: This data set consist of the check-in data of the location-based social network Gowalla which records the time and location of the user's check-in from February 2009 to October 2010. We use the location information for evalution. There are approximately one million data points in the data set. As shown in the second picture of Figure 6, the data distribution is sparse.

(3)$Landmark$: This data set contains information on the location of landmarks such as schools, post offices, shopping malls, construction sites, and train stations in 48 states in the United States. It originated from the 2010 Census TIGER. The data set contains approximately $900,000$ data points. As shown in the third picture of Figure 6, the data distribution is relatively uniform.

Table 3 gives the detailed information of the three data sets, including the number of data points, the size of the domain, and the size of the query area used in the evaluation of the experiment, where q_1 is the smallest query area and q_6 is the largest query area.

Experimental Process For spatial data, we propose a self-adaptive grid-based algorithm for adding noise, which is based on AG and adaptively adds noise according to the number of data points in each grid. In addition to comparing it with AG, we also compare it with

Table 3. Information on Datasets

Dataset	Number of data points	Size of the domain	Size of the query area q_1	Size of the query area q_6
Road	1.6M	25*20	0.5*0.5	16*16
Checkin	1M	360*150	6*3	192*96
Landmark	0.9M	60*40	1.25*0.625	40*20

the mixed-tree partitioning algorithm [22] which belongs to the same general-distributed class publishing algorithm to verify the effectiveness of the grid-partitioning algorithm.

Evaluation Metrics The differential privacy protection mechanism is mainly obtained by adding noise to the original calculated count value. This mechanism has two purposes. On one hand, it should to protect the privacy of each user. On the other hand, it should obtain that the published results will be still usable. Therefore, we quantify these two aspects through various index parameters, hoping to reach a balance.

For grid-based spatial data publishing algorithms, we measure the accuracy of published data by calculating relative errors.

For a query r, we use $A(r)$ to represent the correct answer for r. For the method M and a query r, we use to represent the query r which is answered using an index structure constructed by the method M. The formula for the relative error is:

$$RE_M(r) = \frac{|Q_M(r) - A(r)|}{max\{A(r), \rho\}} \tag{15}$$

where the query r has 6 kinds of sizes, q_1 is the smallest, and the length and width of q_{i+1} are respectively increased by 2 times on the basis of q_i, and q_6 is maximal and covers $1/4$ to $1/2$ of the entire space. The specific information is shown in Table 3. We randomly generate 200 queries for each query size and calculate their relative errors.

ρ is set to $0.001|D|$, where D represents the total number of data points in the data set. The reason why we maximize the denominator is to prevent $A(r) = 0$. When the query r is medium in size, $RE_M(r)$ tends to be the largest. When the query is large, it may be small because $A(r)$ is large.

4.2. Evaluation

We proposed a grid-based spatial partitioning data publishing algorithm for publishing geo-spatial data based on a differential privacy protection model. In the experimental results, K_{hy} represents the kd-hybrid algorithm proposed in [24], UG represents uniform grid partition, AG represents adaptive grid partition, and SGNO represents our proposed grid-based advanced noise-added self-adaptive grid publishing algorithm. When $\varepsilon = 0.1$, the granularity of the Road, Checkin, and Landmark datasets is 126, 100, and 95, respectively calculated by the UG method by the formula. When $\varepsilon = 1$, the granularity of the Road, Checkin, and Landmark data sets is 400, 316, and 300, respectively. When

$\varepsilon = 0.1$, the first level of the AG method divides the granularity of the reference [26] recommended value of the experiment, which means the granularity of the Road, Checkin, and Landmark data sets is 16, 32, and 32 respectively. When $\varepsilon = 0.1$, the granularity of the Road, Checkin, and Landmark data sets is 32, 64, and 64 respectively. UG and AG take the corresponding $c = 10$, $c_2 = 5$, $\beta = 0.5$, in the granularity formula. Figure 7 to Figure 12 give the average curve of the relative error of the random query region for the four algorithms on the three data sets when the value of ε is 0.1 or 1 respectively. In order to make the experimental results more general, we will run the four algorithms 50 times, and finally take the average of them.

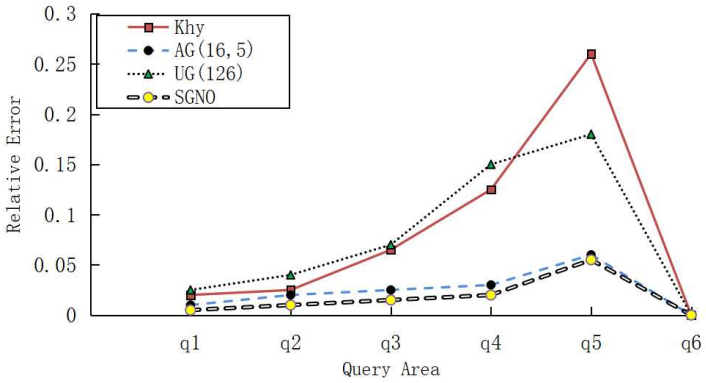

Fig. 7. Experiment Result on Dataset Road with $\varepsilon = 0.1$

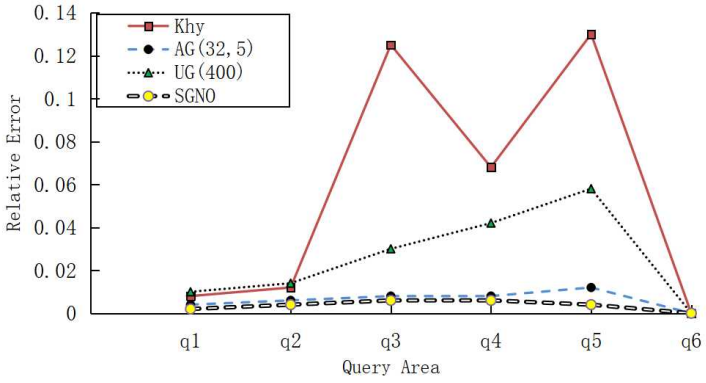

Fig. 8. Experiment Result on Dataset Road with $\varepsilon = 1$

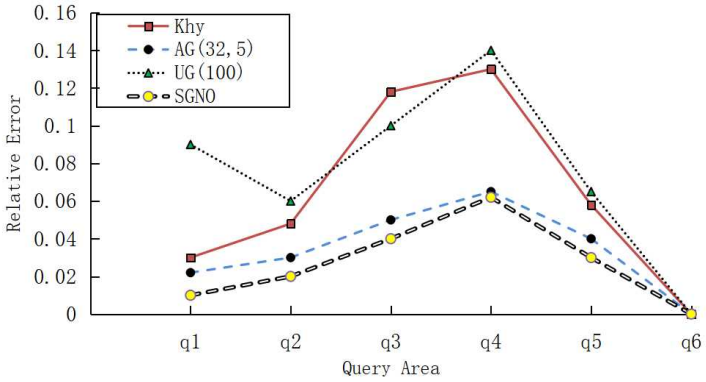

Fig. 9. Experiment Result on Dataset Checkin with $\varepsilon = 0.1$

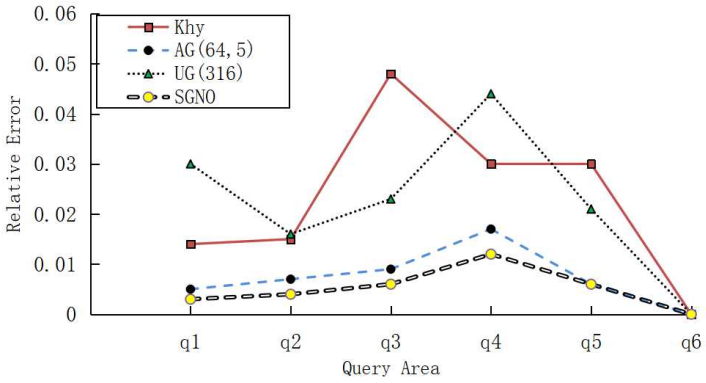

Fig. 10. Experiment Result on Dataset Checkin with $\varepsilon = 1$

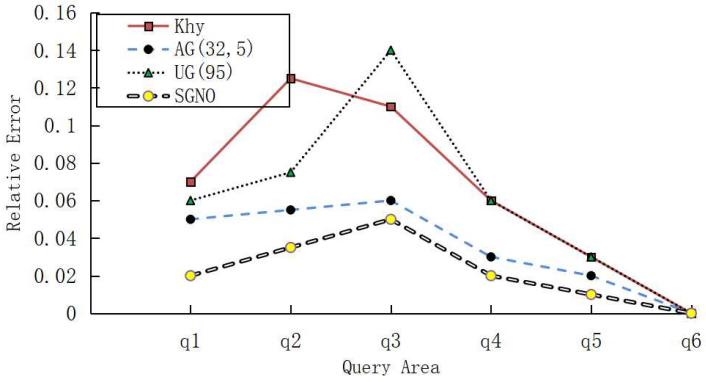

Fig. 11. Experiment Result on Dataset Landmark with $\varepsilon = 0.1$

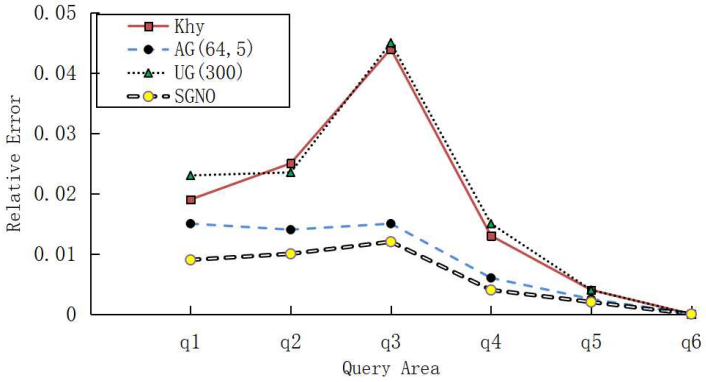

Fig. 12. Experiment Result on Dataset Landmark with $\varepsilon = 1$

Figure 7 and Figure 8 show the experimental results for the data set Road at $\varepsilon = 0.1$ and $\varepsilon = 1$ respectively. It can be seen that the relative error of the four algorithms increases with the increase of the query area, and the relative error at the query q_5 reaches the maximum, at q_6 decreases sharply and is almost close to zero. The reason for this trend is that the query area q_6 occupies between $1/4$ and $1/2$ of the entire query space, making the true answer value large, thus the relative error is small.

When $\varepsilon = 0.1$, the relative errors of the first four queries of kd-hybird and UG are relatively close, but the kd-hybird changes more greatly in the latter two queries. The AG and our method SGNO are superior to kd-hybird and UG on any query. The relative error of the first four queries of SGNO is better than AG, and the query result of q_5 is close to AG. The reason is that when the query range is relatively large, the actual query value is also relatively large, which makes the effect of noise optimization cannot be clearly displayed. When $\varepsilon = 1$, the relative error of the corresponding four algorithms is reduced since the degree of privacy protection is reduced thus the amount of added noise is reduced too.

Figure 9 and Figure 10 show the experimental results of the data set Checkin when $\varepsilon = 0.1$ and $\varepsilon = 1$ respectively. It can be seen that the relative error of the four algorithms increases with the increase of the query area, and the relative error at the query q_4 reaches a maximum when decreasing sharply at q_5 and q_6.

When $\varepsilon = 0.1$, the results of kd-hybrid and UG are mostly similar. The overall trend of UG and SGNO is relatively close, but the relative error of SGNO is generally lower than AG, especially when the query area is relatively small. The reason is that SGNO optimizes the amount of noise added in each grid. In that case, we reduce the relative error of each grid so the entire error is improved. However, when the query area is relatively large, the real calculated count value is relatively large, making the optimization effect difficult to show. When $\varepsilon = 1$, kd-hybrid and UG results change more drastically. Figure 11 and Figure 12 show the results of the Landmark data sets when $\varepsilon = 0.1$ and $\varepsilon = 1$ respectively. With the increase of the query area, the four algorithms increase the relative error, and because the data points of the data set are relatively evenly distributed, the relative error reaches the maximum at the relatively small q_3 in the query area. When

$\varepsilon = 0.1$ and $\varepsilon = 1$, the overall trend of UG and SGNO is relatively close. SGNO is mostly better than the other three algorithms.

From the figure, we can observe our SGNO methods have performed better than other methods. The performance of the UG method is similar to that of the kd-hybrid method. Above all, the results of SGNO in most queries are better than the other three algorithms.

5. Conclusion

This article is based on application scenarios for the partition based data distribution algorithm. For the partition-based data distribution method, we first uniformly partitions the original spatial data, add a Laplace noise with uniform scale parameters, and then select the set of grids to be optimized in a standard way that is based on the maximum ratio between the value of overall error reduction and privacy budget increase value, then operate noise optimized algorithm for the grid. This process is recursive until all grids have been optimized. This grid-based self-adaptive noise-added publishing algorithm solves the problem of the noise scale parameters added uniformly to each grid and the waste of the first-level grid privacy budget.

At the same time, for the above differential privacy data publishing algorithm, problem how to support dynamic data partitioning is the future research direction.

Acknowledgements Supported by the Double-class Construction Innovation Project 014-3190518. Supported by the National Nature Science Foundation of China 61370198, 61370199, 61672379 and 61300187. Supported by the Liaoning Provincial Natural Science Foundation of China NO. 2019-MS-028.

References

1. Xiong, P., Zhu, T. Q., Meng, X. F.: A Survey on Differential Privacy and Applications, Chinese Journal of Computers, 37(1), 101-120.(2014)
2. Gherari, M., Amirat, A., Laouar, R., Oussalah, M.: A smart mobile cloud environment for modelling and simulation of mobile cloud applications, International Journal of Embedded Systems, 9(5), 426-443.(2017)
3. Chen, P., Chen, J., Huang, J.: Multi-user location-dependent skyline query based on dominance graph, International Journal of Computational Science and Engineering, 13(3), 209-218.(2016)
4. Dwork, C.: Differential privacy[J], Lecture Notes in Computer Science, 26(2), 1-12.(2006)
5. Li, N., Li, T., Venkatasubramanian, S.: T-closeness: privacy beyond k-anonymity and l-diversity[C], Proceeding of the IEEE 23rd International Conference on Data Engineering (ICDE), 106-115.(2007)
6. Cormode, G., Procopiuc, C., Srivastava, D., Shen, E., Yu, T.: Differential private spatial decompositions, IEEE International Conference on Data Engineering (ICDE), 20-31.(2012)
7. Zhang, X. J., Meng, X. F., Chen, R.: Differentially Private Set-Valued Data Release against Incremental Updates, International Conference on Database Systems for Advanced Applications, 392-406.(2013)
8. Qardaji, W., Yang, W., Li, N.: Differentially private grids for geospatial data[C], Proceedings of IEEE 29th International Conference on Data Engineering (ICDE), 757-768.(2012)
9. Ebadi, H., Sands, D., Schneider, G.: Differential privacy: now it's getting personal[C], Proceedings of the 42nd Annual Symposium on Principles of Programming Languages, 69-81.(2015)

10. Gruska, D. P.: Differential privacy and security[J], Fundamenta Infomaticae, 143(1), 73-87.(2016)
11. Dwork, C., McSherry, F., Nissim, K. et al.: Calibrating noise to sensitivity in private data analysis, Theory of Cryptography, 265-284.(2006)
12. Mcsherry, F., Talwar, K.: Mechanism design via differential privacy, IEEE Symposium on Foundations of Computer Science, 94-103.(2007)
13. Geng, Q., Viswanath, P.: The optimal noise-adding mechanism in differential privacy, IEEE Transactions on Information Theory, 62(2), 925-951.(2016)
14. Kairouz, P., Oh, S., Viswanath, P.: The composition theorem for differential privacy, Proceedings of the 32nd International Conference on Machine Learning, 63(6), 4037-4049.(2015)
15. Mcsherry, F. D., Meng, X. F.: Privacy integrated queries: an extensible platform for privacy-preserving data analysis, Communications of the ACM, 53(9), 89-97.(2015)
16. Xiao, XX., Bender, G., Hay, H. et al.: iReduct:differential privacy with reduced relative errors, ACM SIGMOD International Conference on Management of Data, 229-240.(2011)
17. Li, Y. D., Zhang, Z., Winslett, M. et al.: Compressive mechanism:utilizing sparse representation in differential privacy, Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society(WPES), 177-182.(2011)
18. Peng, S., Yang, Y., Zhang, Z. et al.: DP-tree:indexing multi-dimensional data under differential privacy, Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, 864-864.(2012)
19. Hardt, M., Talwar, K.: On the geometry of differential Privacy, Proceedings of the 42nd Annual ACM Symposium on Theory of Computing(STOC), 705-714.(2010)
20. Zhang, X. L., Wu, Y. J., Wang, X. D.: Differential Privacy Data Release through Adding Noise on Average ValuE, International Conference on Network and System Security, 37(1), 417-429.(2012)
21. Dwork, C.: Differential privacy: a Survey of results, International Conference on Theory and Applications of MODELS of Computation, 1-19.(2008)
22. Cormode, G., Procopiuc, C., Srivastava, D. et al.: Differentially private spatial decompositions, Proceedings of IEEE 28th International Conference on Data Engineering (ICDE), 41(4), 21-31.(2011)
23. Kamel, I., Faloutsos, C.: Hilbert R-tree: an improved R-tree using fractals, International Conference on Very Large Data Bases, 500-509.(1994)
24. Roy, I., Setty, S. T. V., Kilzer, A. et al.: Airavat: security and privacy for MapReduce, Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation(NSDI), 297-312.(2010)
25. Machanavajjhala, A., Kifer, D., Gehrke, J. et al.: L-diversity: privacy beyond k-anonymity, ACM Transactions on Knowledge Discovery from Data(TKDD), 1(1), 3-14.(2007)
26. Xiao, Y., Xiong, L., Yuan, C.: Differentially private data release through multidimensional partitioning, Proceedings of the 7th VLDB Workshop on Secure Data Management (SDM), 150-168.(2011)

Zhaobin Liu received the Ph.D degree in computer science from Huazhong University of Science and Technology, China, in 2004. He is currently a Professor in the school of information science and technology, Dalian Maritime University, China. He has been a Senior Visiting Scientist at The University of Auckland, New Zealand, in 2017, and a visiting scholar at University of Central Florida, USA, in 2014 and University of Otago, New Zealand in 2008 respectively. His research interests include big data, cloud computing and data privacy.

Haoze Lv is currently in school with the Department of Computer Science in Dalian Maritime University. His research mainly focuses on Big data and privacy protection.

Minghui Li is currently pursuing the master's degree with the Department of Computer Science in Dalian Maritime University. Her research mainly focuses on Big data and privacy protection.

Zhiyang Li (corresponding author) is currently an associate professor at the Information Science and Technology College, Dalian Maritime University, China. He received the Ph.D. degree in computation mathematics from Dalian University of Technology, China in 2011. His research interests include computer vision, cloud computing and data mining.

Zhiyi Huang received the BSc degree in 1986 and the PhD degree in 1992 in computer science from the National University of Defense Technology (NUDT) in China. He is an Associate Professor at the Department of Computer Science, University of Otago. He was a visiting professor at EPFL and Tsinghua University in 2005, a visiting scientist at MIT CSAIL in 2009, and a visiting professor at Shanghai Jiao Tong University in 2013. His research fields include parallel/distributed computing, multicore architectures, operating systems, green computing, cluster/grid/cloud computing, high-performance computing, and computer networks. He has more than 130 publications in peer-reviewed conferences and journals, many of which are top ranked.

Received: September 1, 2018; Accepted: September 12, 2019.