Differential Privacy on Spatio-Temporal Data

Yi Li [a], Bo Ning [b, *], Mei Bai [c], Yawen Zheng [d] and Yu Wang [e]
School of Information Science and Technology, Dalian Maritime University, Dalian116026, China;
[a]yilidlmu7@163.com, [b]ningbo@dlmu.edu.cn, [c]baimei861221@163.com,
[d]18041154873@163.com, [e]wangyu_dlmu@163.com

Keywords: spatio-temporal; differential privacy; score coefficient; sensitivity; prefix tree.

Abstract. In this paper, we consider the location and time factor together, and we give different location and time to add the corresponding score coefficient to distinguish between the sensitivity of location and time. We will be under strict differential privacy model study of spatio-temporal data. We propose an efficient data processing method, based on the hybrid granularity prefix tree structure. At the same time, several novel differential privacy budget allocation schemes are proposed.

1. Introduction

Now a lot of works have shown that it is possible to treat sensitive data while ensuring strong privacy guarantees under differential privacy[1]. Spatio-temporal data can be considered as a special kind of trajectory data. There have been some existing researches[2][3] on protecting location-based queries. Euclidean distance is used as the metric function of trajectory clustering in [4].

Common differential privacy histogram publishing algorithm uses a tree structure to add noise, such as, k-tree[7]. And many interval trees based differential privacy histogram distribution use the same variance method[6]. DP-tree is proposed in [5], multidimensional data can be released.

Our contributions are as follows. We combine location and time to consider under differential privacy. And we propose different privacy budget allocation methods and get the noise to distinguish each node of sensitivity. We create an empty node in the case which the time is not continuous.

2. Preliminaries

Let L={Loc_1t_1,Loc_2t_2 •••Loc_kt_n} be the universe of locations at different time, where $t_1 \le t_2 \le t_n$. The time factor is discretized into intervals at different levels of granularity, e.g., minute. Each record in a sequential database consists of a sequence of time-ordered locations. Formally, a sequence S is an ordered list of locations S= $Loc_1t_1 \rightarrow Loc_2t_2 \rightarrow \cdots \rightarrow Loc_kt_n$, where $Loc_kt_n \in L$. A location can occur multiple times in S, and may occur consecutively in S. Table 1 presents a sample sequential database with L={Loc_1t_1,Loc_2t_2,Loc_3t_3,Loc_3t_1,Loc_1t_3,Loc_4t_3,Loc_1t_4,Loc_1t_2}.

Table 1. Sample sequential database

Rec. #	Sequential database
1	Loc1t1→ Loc2t2→ Loc3t3
2	Loc1t1→ Loc2t2
3	Loc3t1→ Loc2t2→ Loc1t3
4	Loc1t1→ Loc2t2→ Loc4t3
5	Loc1t1→ Loc2t2→ Loc3t3
6	Loc3t1→ Loc2t2→ Loc1t4
7	Loc1t1→ Loc2t2→ Loc4t3→ Loc1t4
8	Loc3t1→ Loc1t2

2.1 Prefix Tree.

Definition 1 (prefix tree). A prefix tree PT is a triplet PT = (*V*, E, Root), where *V* is the labeled with location and time, each corresponding to a unique prefix in D; E is the set of edges, representing transitions between nodes; Root$\in V$ is the virtual root of PT. The unique prefix represented by a node $v\in V$, denoted by prefix (v, PT), is a sequence of spatio-temporal property starting from Root to v.

Each node $v\in V$ keeps the form of (tr(v), c(v)), where tr(v) having prefix(v, PT), and c(v) is a noisy version of |tr(v)| (e.g., |tr(v)| plus Laplace noise), tr(Root) contains all sequences in D. We call the set of all nodes of PT at a given depth i a level of PT, denoted by level (i, PT). Root is at depth zero. But the sequential data is often discrete at time. So in our prefix tree, we put the nodes with time t_i at the level of i. For example, if a sequence S= $Loc_3t_1\rightarrow Loc_2t_2\rightarrow Loc_1t_4$, a null node is created to put at the lever of Loc_2t_2 and Loc_1t_4. Fig. 1 illustrates the prefix tree of the sample database in Table 1.

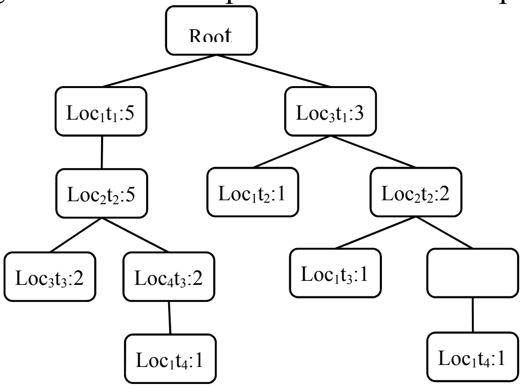

Fig. 1 The prefix tree of the sequential database

2.2 Differential Privacy

Definition 2 (ε-Differential Privacy). Consider that ε > 0 be an arbitrarily-small real constant and any neighboring databases D_1 and D_2 which are differing on at most one record, a privacy mechanism A gives ε-differential privacy for any possible sanitized dataset $\overline{D}\subseteq$ Range(A),

$\Pr[A(D_1)=\overline{D}]\leq e^{\varepsilon}\times\Pr[A(D_2)=\overline{D}]$

Where Pr is a probability distribution over the randomness of the algorithm.

Theorem 1. [1] For any function $f : D\rightarrow R^d$ over an arbitrary domain D, the mechanism A

$A(D)=f(D)+\text{Laplace}(\Delta f/\varepsilon)$

Gives ε-differential privacy.

Theorem 2. (Sequential Composition). Suppose there are n random algorithms, let A_i be a set of algorithm such that each provides ε_i – differential privacy, and running in sequence all algorithms A_i provides $\sum_{i=1}^{n}\varepsilon_i$ - differential privacy.

Theorem 3. (Parallel Composition). Suppose D_i are disjoint subsets of the original database, and A_i is a set of analyses each providing ε_i - differential privacy, then applying each analysis A_i on partition D_i provides max (ε_i)- differential privacy.

3. Sanitization Algorithm

3.1 Location Sensitive Segmentation Algorithm

When the same position on the same path, which itself will reduce the sensitivity of the location, that is to say, the sensitivity of a location sensitivity not only depends on the site itself, but also depends on the number of it appears on the same path.

On a path, we divide the original location sensitivity by n which is the number of the same location points appear on a path to get the final location sensitivity. That is, if a location point appears repeatedly on a path, then its score sensitivity decreases to the original st_i / n. When a node has at least two child nodes, the score coefficient sensitivity of a node selects smaller. For example, in Figure 1, we respectively set the score coefficient of and Loc_1 to Loc_4 as 0.1, 0.15, 0.05, 0.2, t_1 to t_4 as 0.05, 0.15, 0.2, 0.1. Loc_1 repeats two times on the path of $Loc_1t_1\rightarrow Loc_2t_2\rightarrow Loc_4t_3\rightarrow Loc_1t_4$, on this path, the score coefficient of Loc_1 is 0.1/2=0.05, Others remain unchanged. And the score coefficient of node Loc_1t_1=0.05+0.05=0.1, Loc_2t_2=0.15+0.15=0.3, Loc_4t_3=0.2+0.2=0.4, Loc_1t_4=0.05+0.1=0.15.

Loc_1t_1 on the two paths. For $Loc_1t_1 \rightarrow Loc_2t_2 \rightarrow Loc_3t_3$, the score coefficient sensitivity of Loc_1t_1 is 0.1+0.05=0.15. And for $Loc_1t_1 \rightarrow Loc_2t_2 \rightarrow Loc_4t_3 \rightarrow Loc_1t_4$, the score coefficient sensitivity of Loc_1t_1 is 0.1/2+0.05=0.1. So in the final process, 0.1 will be chosen. Algorithm 1 presents the details.

Algorithm 1 location sensitive segmentation algorithm

```
Input: raw sequential dataset D, privacy budget $\varepsilon$, score coefficient $sl_k$, $st_i$, taxonomy tree T, height of the prefix tree h
Output: noisy hybrid-granularity prefix tree PT
1: i=0;
2: Construct a prefix tree PT with a virtual root Root;
3: tr(Root) ← all records in D, node with time $t_i$ in level(i);
4: if no node with time $t_i$ in level(i)
5: level(i) ← Null node;
6: end if;
7: for i < h do
8:   for non empty node v level(i) do
9:      $sl_{k1}=sl_k/n$;
10:     $\varepsilon_v=(sl_{k1}+st_i)\varepsilon$;
11:   if node v on multiple paths
12:     the minimum coefficient will be chosen;
13:   end if;
14:     c(v)=Noisecount((|tr(u)|,$\varepsilon_v$);
15:       add c(v) to PT;
16:   end for;
17: i++;
18:  end for;
19: return PT;
```

Algorithm 2 Location Sensitivity Segmentation Base On Time Sensitivity Algorithm

```
Input: raw sequential dataset D, privacy budget $\varepsilon$, score coefficient $sl_k$, $st_i$, $st'_i$, taxonomy tree T, height of the prefix tree h
Output: noisy hybrid-granularity prefix tree PT
1: i=0;
2: Construct a prefix tree PT with a virtual root Root;
3: tr(Root) ← all records in D, node with time $t_i$ in level(i);
4: if no node with time $t_i$ in level(i)
5: level(i) ← Null node;
6: end if;
7: for i < h do
8:   for non empty node v level(i) do
9:      $sl_{k1}= sl_{k*} st_i/( st_i+ st'_i)$;
10:     $\varepsilon_v=(sl_{k1}+st_i)\varepsilon$;
11:   if node v on multiple paths
12:     the minimum coefficient will be chosen;
13:   end if;
14:     c(v)=Noisecount((|tr(u)|,$\varepsilon_v$);
15:       add c(v) to PT;
16:   end for;
17: i++;
18:  end for;
19: return PT;
```

3.2 Location Sensitivity Segmentation Base On Time Sensitivity Algorithm

In our article, a node contains the location and time of the two factors, so a path contains repeated location nodes, we just consider the number of repeating location of nodes, without considering the impact of time sensitivity on location sensitivity, and it is not comprehensive.

In this phase, we reassigned the location score coefficient sensitivity according to the time. The reassigned score coefficient, equals the ratio of time score coefficient of this node to the sum of time score coefficients with the same location, and multiplied by the predefined coefficient. To show our intentions more clearly, we also use Figure 1 as an example, and set the score coefficient as the same as 3.1. Because Loc_1 repeats two times on $Loc_1t_1 \rightarrow Loc_2t_2 \rightarrow Loc_4t_3 \rightarrow Loc_1t_4$, the score coefficient of Loc_1 in Loc_1t_1 is 0.1*0.05/(0.05+0.1)=1/30, that is to say, the node of Loc_1t_1 is 1/30+0.05, and the Loc_1 in Loc_1t_4 is 0.1*0.1/(0.05+0.1)=1/15, that is to say, the node of Loc_1t_4 is 1/15+0.1. And the score coefficient of Loc_2t_2 is 0.15+0.15=0.3, Loc_4t_3 is 0.2+0.2=0.4. Algorithm 2 presents the details.

3.3 Sanitization Database Release Generation

A prefix tree exists two sets of consistency constraints, one is for any root-to-leaf path p, $\forall v_i \in p$, $|tr(v_i)| \le |tr(v_{i+1})|$, where v_i is a child of v_{i+1}. The other is for each node v, $|tr(v)| \ge \sum_{u \in children(v)} |tr(u)|$.

We compute the consolidated intermediate estimate $c_0(v)$ of v as the mean of the estimates. And we can computer the consistent estimate $c_1(v)$ of v in a top-down fashion, that is:

$$c_0(v) = \begin{cases} c_1(v) & if\ v \in leave(1, PT) \\ c_1(v) + \min\left(0, \frac{c_1(p) - \sum_{u \in children(w)} c_0(u)}{|children(p)|}\right) & otherwise \end{cases}$$

p is the parent of v. And in this way, the sum of children noise is most equal to the parent's noise.

4. Privacy Analysis

Because differential privacy has a strict definition, we should use it carefully. We now proof that our algorithm satisfies ε-differential privacy.

Proof. According to the parallel composition, the entire privacy budget needed for a level is bounded by the worst case, that is, the node privacy budget which is the biggest sum of score coefficient. We get the score coefficient is between 0 and 1, so the biggest privacy budget is between 0 and ε.

According to the sequential composition, we set the sum of score coefficients not more than 1, so each path of the privacy budget not more than ε. So the actual cost of the privacy budget is less than ε.

The procedure of generating the sanitized database not accesses the underlying database. Traversal doesn't destroy the relative nature of differential privacy, so the depth first search processing results remains differential privacy. So our algorithm satisfies ε-differential privacy.

5. Conclusion

In this paper, we put our database on prefix tree to deal with, and we proposed the concept of score coefficient of location and time. Then we proposed two privacy allocation algorithms, and each node's privacy budget does not depend on layers, only with the score coefficient. At the same time, we presented the sanitization database release generation, it guarantees the satisfaction of the consistency constraints. Finally, we did privacy analysis to prove our algorithms to satisfy differential privacy.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No. 61371090, 61272171), the Natural Science Foundation of Liaoning Province(Grant No. 201602094), the Fundamental Research Funds for the Central Universities(No. 3132017118).

References

[1]. C. Dwork, F. McSherry, K. Nissim, and A.Smith.Calibrating noise to sensitivity in private data analysis. In TCC, 2006.

[2]. G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan. Private queries in location based services: anonymizers are not necessary. In SIGMOD, pages 121–132, 2008.

[3]. M. F. Mokbel, C.-Y. Chow, and W. G. Aref. The New Casper: Query Processing for Location Services without Compromising Privacy. In Proc. of VLDB, 2006.

[4]. Abul, O., Bonchi, F., & Nanni, M. (2008). Never walk alone: uncertainty for anonymity in moving objects databases. 376-385.
[5]. Peng, Shangfu, et al. "DP-tree: indexing multi-dimensional data under differential privacy." Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. ACM, 2012.
[6]. Acs, Gergely, Claude Castelluccia, and Rui Chen. "Differentially private histogram publishing through lossy compression." Data Mining (ICDM), 2012 IEEE 12th International Conference on. IEEE, 2012.
[7]. Hay, Michael, et al. "Boosting the accuracy of differentially private histograms through consistency." Proceedings of the VLDB Endowment 3.1-2 (2010): 1021-1032.