



Doctoral Thesis

**A Rigorous and Flexible Privacy Protection
Framework for Utilizing Personal Spatiotemporal
Data**

Yang Cao

February 2017

Department of Social Informatics
Graduate School of Informatics
Kyoto University

Doctoral Thesis
submitted to Department of Social Informatics,
Graduate School of Informatics,
Kyoto University
in partial fulfillment of the requirements for the degree of
DOCTOR of INFORMATICS

Thesis Committee: Masatoshi Yoshikawa, Professor
Katsumi Tanaka, Professor
Yasuo Okabe, Professor

A Rigorous and Flexible Privacy Protection Framework for Utilizing Personal Spatiotemporal Data*

Yang Cao

Abstract

Personal data is the new oil. Vast amounts of spatiotemporal data generated by individuals have been collected and analyzed, such as check-ins, trajectories, web browsing data and time-stamped medical records. These personal data are a valuable resource for business companies and also have the potential for significant social benefit through sharing and reusing. However, privacy concerns hinder the wider use of these personal data. To this end, dozens of privacy protection frameworks that attempt to define what data privacy is and how to achieve such privacy are proposed for privacy-preserving data publishing (PPDP). ϵ -Differential Privacy (DP) has emerged as a de facto privacy model for PPDP because of rigorous theoretical guarantees. However, traditional differential privacy may not be appropriate for utilizing spatiotemporal data because it is not flexible enough to deal with personalized privacy requirement and spatiotemporal correlations. We argue that there are two fundamental defects of differential privacy for utilizing personal spatiotemporal data as follows. (1) *One size for all*. ϵ -Differential privacy use a single parameter ϵ to represent the protection level for all users in the database, whereas the privacy requirements should be personalized. A user should have the right to choose his/her privacy protection level. (2) *Data independence assumption*. Differential privacy assumes that the tuples in the databases are independent, whereas the data collected from real-world tend to be temporally or spatially correlated. In this research, we designed a rigorous and flexible privacy framework that overcome these defects and be able to strikes a good balance between data privacy and utility for utilizing personal spatiotemporal data. In our framework, three novel privacy models and mechanisms are

*Doctoral Thesis, Department of Social Informatics, Graduate School of Informatics, Kyoto University, KU-I-DT6960-26-4383, February 2017.

proposed for achieving personalized differential privacy on correlated data. They can be used separately or in combination for different scenarios. We summarize the major contributions in this dissertation as follows.

First, we proposed a personalized privacy model to guarantee each users' ℓ -length trajectories to be protected under differential privacy. We call this model ℓ -trajectory privacy, which is an extension of differential privacy on streaming data. We formalized our privacy definition, proved its reachability, and designed efficient mechanisms to achieve it. This privacy model is personalized, i.e., different users can specify the length of being protected trajectory in the manner of sliding windows. On the other hand, this privacy model is rigorously defined because it is designed under the framework of differential privacy. We proposed a mechanism satisfying ℓ -trajectory privacy for *infinite* spatiotemporal data stream publishing, which is considered as an important and hard problem. The experiments show that our approach can achieve the same privacy protection level with fewer noises comparing with previous work.

Second, we tackle with the problem of temporally correlated data in DP, which is neglected by most of the previous work that attempts to apply DP in practice. Traditional DP mechanisms assume that the data are independent. However, the data continuously generated in the real-world tend to be temporally correlated. In this study, we demonstrate the privacy loss of a traditional DP mechanism on temporal data may increase over time. To address this issue, we propose a new privacy definition, temporal differential privacy, which is able to achieve differential privacy on temporally correlated data.

Third, we deal with another functional problem of DP that the data generated by different users may be correlated with each other. For example, if two users are colleagues and working in the same building, the location data generated by them may be correlated in a high probability. We call it user-user correlation. Such user-user correlations widely exist; however, traditional DP mechanisms assume that the data are independent. To this end, we analyze the potential privacy loss when an attacker has knowledge about such correlations, and proposed mechanisms to prevent such unexpected privacy loss so that to ensure differential privacy under user-user correlations.

Keywords: Differential Privacy, Privacy Preserving Data Publishing, Spatiotemporal data, Personal Data.

CONTENTS

1	Introduction	1
1.1	Motivation	1
1.2	Overview of This Research	7
2	Background	10
2.1	Differential Privacy	10
2.2	Differentially Private Sptiotemporal Data Publishing	12
2.3	Metrics of Data Utility	13
3	1	15
3.1	Motivation	15
3.2	Introduction	16
3.3	Related Work	19
3.3.1	Differential Privacy on Streams	19
3.3.2	Privacy of Trajectory Data	20
3.4	Preliminary	20
3.4.1	Differential Privacy	20
3.4.2	Mechanisms of Differential Pivacy	21
3.5	Problem Definition and Privacy Model	22
3.5.1	Data Model	22
3.5.2	Privacy Risk	24
3.5.3	Proposed Privacy Model	24
3.5.4	Methodology to Achieve ℓ -Trajectory Privacy	25
3.6	Algorithms	26
3.6.1	First-cut Solution	26

Contents

3.6.2	Proposed Framework	27
3.6.3	Dynamic Budget Allocation	29
3.6.4	Approximation Strategies	30
3.6.5	Private Publishing	33
3.7	Experiments	34
3.7.1	Privacy Evaluation	35
3.7.2	Utility Evaluation	36
3.7.3	Runtime	39
4	Temporal Differential Privacy	41
4.1	Motivation	41
4.2	Introduction	42
4.2.1	Contributions	45
4.3	Preliminaries	46
4.3.1	Privacy Leakage.	46
4.3.2	Problem Setting	47
4.4	Analyzing Privacy Leakage	47
4.4.1	Adversary with Knowledge of Temporal Correlations	48
4.4.2	Temporal Privacy Leakage	51
4.4.3	DP under Temporal Correlations and Its Composability	55
4.4.4	Discussion	57
4.5	Calculating Temporal Privacy Leakage	58
4.5.1	Problem formulation	58
4.5.2	Privacy Leakage Quantification Algorithm	61
4.6	Bounding Temporal Privacy Leakage	62
4.7	Experimental Evaluation	65
4.7.1	Runtime of Privacy Quantification Algorithms	66
4.7.2	Impact of Temporal Correlations on Privacy Leakage	68
4.7.3	Evaluation of Data Releasing Algorithms	68
4.8	Related Work	70
4.9	Conclusions	71
5	Differential Privacy under User-User Correlations	72
5.1	Introduction	72
5.2	Preliminaries	75
5.2.1	Bayesian Differential Privacy	75

Contents

5.2.2	Problem Setting	76
5.3	Adversary $\mathcal{ST}$	76
5.3.1	User-user correlation	77
5.3.2	Prior Knowledge and Attack Model	78
5.4	Privacy Leakage Analysis	80
5.4.1	Spatial Privacy Leakage (under user-user correlations) . . .	81
5.4.2	Composability	84
5.4.3	Discussion	86
5.5	Algorithms	87
5.5.1	Quantifying SPL	87
5.5.2	Achieving ϵ -DP $\mathcal{ST}$ against TPL and SPL	89
5.6	Experimental Evaluation	91
5.6.1	Evaluation of the Pruning Search	92
5.6.2	Impact of correlation on Privacy Loss	94
5.7	Related Work	95
6	Conclusion and Future work	97
6.1	Conclusion	97
6.2	Future work	98
	Acknowledgements	100
	References	102
	Selected List of Publications	110
	Appendix	111
A	Proof of Theorem 10	111
B	Proof of Theorem 12	112
C	Proof of Corollary 2	114
D	Proof of Theorem 13	114

LIST OF FIGURES

1.1	An illustration of Data Publishing.	3
1.2	An illustration of Privacy Preserving Data Publishing.	4
1.3	Input Perturbation v.s. Output Perturbation.	5
1.4	An overview of the research topics.	8
2.1	Differentially Private Continuous Aggregate Release under Temporal Correlations.	12
3.1	Raw data and aggregate information. Each row in Table (a) is a spatiotemporal data point.	16
3.2	Previous privacy model v.s. our proposed model. Trajectories in dashed ellipses in (b) means ongoing 3-trajectories.	18
3.3	Illustration of our proposed framework. The dark objects contain sensitive data.	28
3.4	Illustration of two approximation strategies.	31
3.5	Comparison of Private and Real Data.	36
3.6	MAE on four datasets by varying ℓ_u (10 to 100) and varying ϵ (0.0001 to 1).	37
3.7	MRE on four datasets by varying ℓ_u (10 to 100) and varying ϵ (0.0001 to 1).	37
3.8	MSE on four datasets by varying ℓ_u (10 to 100) and varying ϵ (0.0001 to 1).	38
3.9	KL-divergence on four datasets by varying ℓ_u (10 to 100).	39
3.10	Runtime of each algorithm on WorldCup98.	40

List of Figures

4.1	Differentially Private Continuous Aggregate Release under Temporal Correlations.	43
4.2	Examples of Temporal Correlations.	50
4.3	Example of Temporal Privacy Leakage of $Lap(1/0.1)$ at each time point.	50
4.4	Examples of the maximum BPL over time.	63
4.5	Runtime of Privacy Quantification Algorithms.	67
4.6	Evaluation of BPL.	69
4.7	Data Release Algorithms with $1\text{-DP}_{\mathcal{T}}$	69
4.8	Data utility of $2\text{-DP}_{\mathcal{T}}$ mechanisms.	70
5.1	Examples of user-user correlations.	73
5.2	Examples of User-user correlation.	79
5.3	The Spatial Privacy Leakage w.r.t. R_1 and R'_1 is the same.	82
5.4	Adversary $_{\mathcal{ST}}$ $R_1(P_1, G_1, 0)$	86
5.5	Example of a Pruning Search for $R_1(G_1, 2)$	88
5.6	Runtime and Accuracy of the Pruning Search	92
5.7	Evaluation of the SPL (on 100 graphs).	96

LIST OF TABLES

3.1	Summary of Notations.	23
3.2	Real-life datasets.	34
3.3	Ratios of risky w -trajectories under w -event privacy.	36
4.1	Summary of Notations.	48
4.2	The privacy guarantee of ϵ -DP mechanisms.	57
5.1	Summary of Notations.	77

CHAPTER 1

INTRODUCTION

1.1 Motivation

With the development of mobile communication and sensor technologies, people, the Internet and the physical world are being connected into one interconnected network. We are able to record our activities and the interactions with others in the physical world and Internet in the form of personal data, such as the visited locations, emails, daily health-related information, medical record, search log, purchase history and micro-blog. Personal data are collected on who we are, who we know, where we are, where we have been and where we plan to go. Mining and analyzing personal data give us the ability to understand and even predict the physical world and the human society.

Personal data In law, there are different definitions about personal data. For example, in Data Protection Acts 1998 of U.K., “personal data” is defined as the data relating to a living individual who is or can be identified either from the data or from the data in conjunction with other information that is in, or is likely to come into, the possession of the data controller. In EU Data Protection Directive (95/46/EC), “personal data” means any information relating to an identified or identifiable natural person (data subject); an identifiable person is one who can be identified, directly or indirectly, in particular by reference to an identification

number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity. Although there are some slightly differences between the definitions, it is widely accepted that personal data contains identifiable information. However, technically, it is hard to determine whether a given piece of information is identifiable or not because the answer depends on the background knowledge of adversaries. In the following paragraph of “privacy model”, we will describe several technical definitions about data privacy in literature. In this thesis, we call *personal spatiotemporal data* as a type of personal data that contain spatial and temporal properties generated in individuals’ daily life, such as people’s movements, search logs, shopping records, social activities and medical histories. We will give a more formal definition about such data in Chapter 2.

Personal data are extremely valuable and necessary to be shared and reused. Business companies like Google, Amazon and Facebook have created enormous wealth by building new services based on user generated data, e.g., search and click streams, purchase history and online social interactions. The demands of data sharing is not only because of commercial interests but also because of the potential for significant public good such as disease surveillance [1] and real-time traffic monitoring [2]. It is also driven by the regulations that require certain data to be published. For example, all licensed hospitals in California are required to submit demographic data on the patients discharged from their facility [3]. In Japan, all hospitals need to submit specific data on the cancer-related patients to National Cancer Center since January, 2016*.

Privacy Risks in Data Publishing However, several well-known privacy leakage cases happened in the attempts to publish and reuse personal data. Personal data are sensitive even if they are *anonymized*, e.g., removing or replacing identifier (name, ID, and so on) from the original data. One of the most well-known privacy leakage cases occurred in the Massachusetts Group Insurance Commission (GIC) medical dataset that includes anonymized medical records, which was opened in 1996 for medical research and public health. Latanya Sweeney [4] successfully identified the medical record of the governor of Massachusetts by combining the anonymized GIC database (which retained the

*The related law refer to (in Japanese): http://www.mhlw.go.jp/stf/seisakunitsuite/bunya/kenkou_iryuu/kenkou/gan/gan_toroku.html.



Figure 1.1: An illustration of Data Publishing.

birthdate, sex, and ZIP code of each patient) and voter registration records. In 2006, AOL company released more than 650 thousand users' anonymized search logs for research purpose, but researchers [5] found it is possible to de-identify some users by mining these timestamped records. Another privacy leakage cases happened in the datasets released by Netflix Prize for recommender algorithms contests. Researchers [6] verified that the datasets can be de-identified by linking them with IMDB dataset so that Netflix encountered in a lawsuit and had to stop this well-known competition in 2010. In Japan, East Japan Railway Company (JR) was faced with a scandal involving selling customer data which are collected through the use of Suica cards in 2013.

On the other hand, spatiotemporal data generated by individuals (such as trajectories, geo-tagged photos) are highly sensitive due to users' unique and rich mobility patterns. Study [7] [8] shows that only *four* spatiotemporal data points from anonymized mobile datasets and credit card records (with purchase time and location) are sufficient to identify 90% of individuals uniquely. Another study [9] demonstrated that it is possible to track user's mobile phone location by only monitoring battery consumption data (timestamp and battery left), which does not need the users' permission to be shared. Therefore, we need a *rigorous* privacy model to protect our data privacy.

Due to the conflict between increasing public concerns about data privacy and the demands of personal data publishing, privacy has become a critical obstacle in big data utilization. To this end, privacy preserving data publishing (PPDP) has

became an important research topic. In the following, we will briefly introduce the approaches of PPDP and the major privacy models used in PPDP.

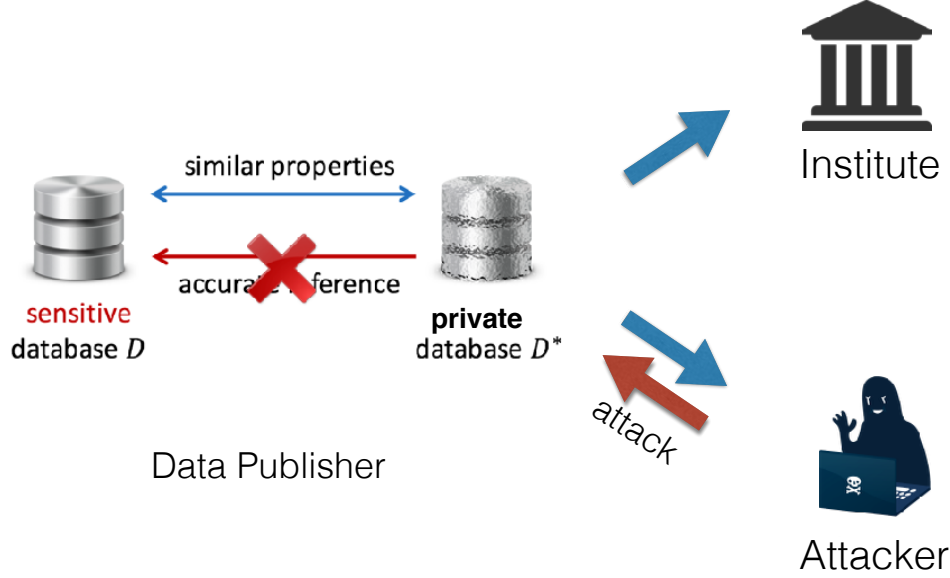


Figure 1.2: An illustration of Privacy Preserving Data Publishing.

Privacy Preserving Data Publishing (PPDP) Privacy Preserving Data Publishing [10] has attracted considerable attention in database community because of the increasing demands of personal data sharing and reusing. The goal of PPDP is to develop methods and tools for releasing a private database, i.e., a transformed database based on the original sensitive database, so that the released data remains practically useful while individual privacy is preserved. The usefulness of the released data is usually considered as the *similarity* between the transformed database and the original one with respect to a desired query over the database. This similarity can be formally defined by statistical measurements such as measure of average error (MAE). We will give more details in Chapter 2.

The scenario of data publishing is described as follows. The data publisher collects personal data from each individual, and releases the collected data to a data miner or to the public (as shown in Figure 1.1), called the data recipient, who will then conduct data mining on the published data. We note that, here “data mining” in the data recipient side has a broad sense, not necessarily restricted to pattern mining or model building. It could be anything from a simple count

query over the database to a sophisticated cluster analysis. There are two types of data publishers [11]. First, the *untrusted* publishers mean who is not trusted and may attempt to identify sensitive information from the collected personal data. Many cryptographic solutions [12]; anonymous communications [13] [14]; and statistical methods [15] were proposed to collect records anonymously from the data owners without revealing their identity. On the other hand, a *trusted* publishers indicate who is trustworthy and record owners are willing to provide their personal data to the data publisher; however, the trust is not transitive to the data recipient, i.e, a data recipient could be an attacker. In this study, we assume that the data publisher is trusted and consider privacy issues are in the data publishing phase instead of data collection phase.

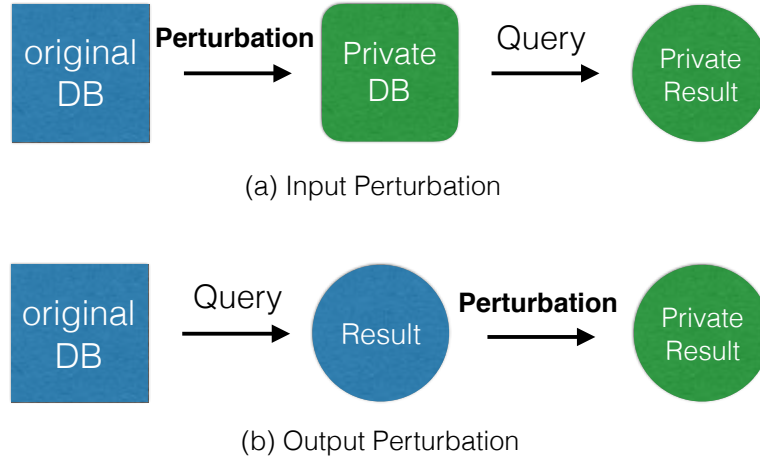


Figure 1.3: Input Perturbation v.s. Output Perturbation.

Privacy Models A privacy models provides the privacy protection goal in the PPDP process. It should also include mechanisms to achieve this goal. Then, what is the goal of privacy protection? Dalenius [16] provided a very strict definition: access to the published data should not enable the attacker to learn anything extra about any target victim compared to no access to the database, even with the presence of any attacker’s background knowledge obtained from other sources. Towards this goal, dozens of privacy models are proposed. *k-anonymity* [4] was proposed in 2002, which is the first privacy model widely accepted by researcher and practitioner. It attempts to hide a user in a group with size k . However, Machanavajjhala [17] pointed out a serious flaw of k -anonymity by demonstrating

homogeneity attack and background knowledge attack, and proposed an improved privacy model based on k-anonymity named ℓ -diversity. Consequently, due to the own defect of ℓ -diversity, it was criticized and improved by the following new privacy models such as t-closeness [18], M-invariance [19]. Researchers realized the essential defect of k-anonymity family is that their privacy guarantee depends on the specific background knowledge of adversaries. Dwork [20] showed that Dale-
 nius’s absolute privacy protection is impossible due to the presence of background knowledge. To this end, she proposed differential privacy [20] in 2006, which does not dependent on the adversarial background knowledge. Although both k-anonymity and differential privacy attempt to transform the original database to a private one, but a methodological difference is that k-anonymity is based on input perturbation while the differential privacy is based on output perturbation, as shown in Figure 1.3.

Differential privacy has received increasing attention because of its guarantee of being rigorous and mathematically provable. However, it is also criticized for performing too much distortion (adding random noises) and destroying the data utility. Hence, many research in this area are attempting to design advanced differentially private mechanisms that be able to achieve the same level of privacy protection (in term of differential privacy) with less noise. Most of these works are just based on traditional differential privacy model. However, we argue that there are two inherent defects of differential privacy may hinder its application in practice.

- ϵ -Differential privacy use a single parameter ϵ to represent the protection level of all users in the database, whereas the privacy requirements should be personalized. Although we can achieve the most strict privacy protection level among all users’ different privacy requirements, this simple approach will result in extra data distortion.
- Differential privacy assumes that the tuples in the databases are independent, whereas the data collected from real-world tend to be temporally or spatially correlated. Few previous studies has pointed out the protection level of differential privacy may degrade on correlated data, but none of them provides efficient solution. A naive solution is to protect the correlated tuples in a bundle no matter the strengthen of the correlation is; however, it results in excessive perturbation that makes the released data not very useful.

In this research, we show that, by creating new privacy models that solve these defects, we are able to design better data release mechanisms, which significantly improve the data utility than previous studies. Since the proposed privacy models are general for spatiotemporal data, it is also expected that they can be used in other privacy preserving data mining tasks.

1.2 Overview of This Research

In this study, to overcome the two fundamental drawbacks of differential privacy, we ask the following questions:

- How can we design personalized differentially private model for trajectory streaming data?
- How can we achieve differential privacy on *temporally* correlated data?
- How can we achieve differential privacy on *user-user* correlations?

Correlations in Spatiotemporal Data. We clarify the correlations defined in this paper below. There are two types of correlations in the spatiotemporal data. one is the correlation between user’s location at different time points, such as the probability of moving from *loc1* to *loc2* is 0.5. We call it *temporal correlation*. Another one is the correlation between different users. For example, if two users are colleagues and working in the same building, the location data generated by them may be correlated in a high probability. We call it *user-user correlation*, which indicates the geographical similarity between users in spatiotemporal data.

Our Approach. The approach used in this research is described as follows. Our goal is to design a flexible framework based on the rigorously defined differential privacy for utilizing spatiotemporal data. Although the first defect of differential privacy seems more general which is not only for spatiotemporal data, we will mainly focus on seeking an efficient personalized privacy model for infinite trajectory streams, which is considered as a challenging problem in PPDP. For the second defect that DP may not be valid on correlated data, we will discuss temporal correlations and user-user correlations, respectively. In our privacy protection framework, we designed three privacy models to overcome the defects

of differential privacy; to demonstrate the merits of our privacy models, we also provided efficient data release mechanisms for spatiotemporal data publishing.

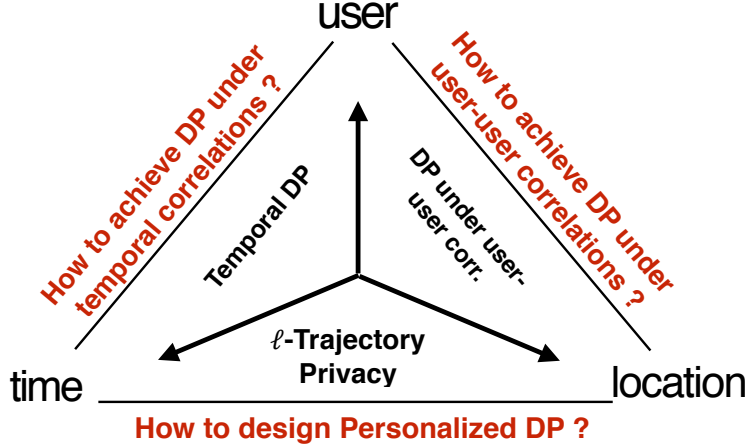


Figure 1.4: An overview of the research topics.

There are three topics in my doctoral study as illustrated in Figure 1.4. Basically, three aspects are involved in spatiotemporal data: *user*, *time* and *location*. First, we proposed ℓ -trajectory privacy for protecting each user's successive ℓ amount of $\langle time, location \rangle$ data points. Second, we model temporal correlations in the single user level, i.e., the data generated by a user may be temporally correlated. Hence, *temporal differential privacy* model defined between *user* and *time*. Finally, we model user-user correlations as the geographical similarity between users; hence the proposed *differential privacy under user-user correlation* is defined between the aspects of *user* and *location*.

Contributions First, we proposed a personalized privacy model to guarantee each users' ℓ -length trajectories to be protected under differential privacy. We call this model ℓ -trajectory privacy. We formalized our privacy definition, proved its reachability, and designed efficient mechanisms to achieve it. This privacy model is personalized, i.e., different users can specify different *length* of ℓ -trajectory according to their privacy requirements. On the other hand, this privacy model is rigorously defined because it is designed under the differential privacy guarantee. Based on ℓ -trajectory privacy, we proposed mechanisms for *infinite* spatiotemporal data stream release. The experiments show that our approach can achieve the same privacy protection level with fewer noises comparing with previous work.

Second, we tackle with the data independence assumption of DP, which may degrade the privacy significantly. Traditional DP mechanisms assume the data are independent. However, the data continuously generated in the real-world tend to be temporally correlated. In this study, we model temporal correlations using Markov model and demonstrate the risk of a traditional DP mechanism under such correlations. It shows that the privacy loss of a traditional DP mechanism may increase over time. To address this issue, we propose a new privacy definition, temporal differential privacy, extended differential privacy for temporally correlated data.

Third, we deal with another functional problem of DP that the data generated by different users may be correlated with each other. For example, if two users are colleagues, they may appear at the same location in a high probability because they may work in the same building. We call it user-user correlation (note that the above-mentioned *temporal correlation* is modeled in a single user level). Such user-user correlations widely exist; however, traditional DP mechanisms assume that the data are independent. In this study, we analyze the potential privacy loss when an attacker has knowledge about such correlation. First, we use a probabilistic graphical model, Gaussian Markov Random Field (GMRF), to represent user-user correlations (the nodes are users and the edges are the strength of geographical similarities between users), as the adversarial knowledge. Then, we quantified the privacy loss with respect to such attacker. We also analyzed the privacy guarantee of a DP mechanism under both temporal correlations and user-user correlations.

The rest parts of this thesis are arranged as follows. In Chapter 2, we provide background knowledge about differential privacy and the scenario of differentially privacy spatiotemporal data publishing. In Chapter 3, we design a flexible privacy model based on differential privacy for personalized privacy protection and mechanisms based on this privacy model. In Chapter 4, we design a privacy model extended differential privacy on temporally correlated data. In Chapter 5, we investigate how to guarantee differential privacy on spatially correlated data. In Chapter 6, we conclude that our approach is effective and efficient and discuss future research directions.

CHAPTER 2

BACKGROUND

2.1 Differential Privacy

Differential privacy [20] is a formal definition about data privacy. Let D be a database and D' be a copy of D that is different in any one tuple. D and D' are *neighboring databases*. A differentially private output from D or D' should exhibit little difference.

Definition 1 (ϵ -DP). $\mathcal{M}$ is a randomized mechanism that takes as input D and outputs $\mathbf{r}$, i.e., $\mathcal{M}(D) = \mathbf{r}$. $\mathcal{M}$ satisfies ϵ -differential privacy (ϵ -DP) if the following inequality is true for any pair of neighboring databases D, D' and all possible outputs $\mathbf{r}$.

$$\log \frac{\Pr(\mathbf{r}|D)}{\Pr(\mathbf{r}|D')} \leq \epsilon. \quad (2.1)$$

The parameter ϵ , called the *privacy budget*, represents the degree of privacy offered. Intuitively, a lower value of ϵ implies stronger privacy guarantee and a larger perturbation noise, and a higher value of ϵ implies a weaker privacy guarantee while possibly achieving higher accuracy.

Two commonly used methods to achieve ϵ -DP are Laplace mechanism and Exponential mechanism. Laplace mechanism adds random noise drawn from a calibrated Laplace distribution into the aggregates to be published. Exponential

mechanism returns the categorical data in a certain probability.

Theorem 1 (Laplace Mechanism [21]). *Let $Q : D \rightarrow \mathbb{R}$ be a statistical query on database D . The sensitivity of Q is defined as the maximum L_1 norm between $Q(D)$ and $Q(D')$, i.e., $\Delta = \max_{D, D'} \|Q(D) - Q(D')\|_1$. We can achieve ϵ -DP by adding Laplace noise with scale Δ/ϵ , i.e., $\text{Lap}(\Delta/\epsilon)$.*

Another way to obtain differential privacy is through the Exponential mechanism [22]. Given a quality function q that scores results of a specified query, where higher scores are better. Then returning results in the probability of exponentially proportional to the corresponding score. In other words, the result with a higher score is exponentially more likely to be chosen. This will ensure differential privacy.

Theorem 2 (Exponential Mechanism [22]). *Let D^n denote all possible databases in the universe, and $D, D' \in D^n$ are two instances of neighboring databases. Let $q : (D^n \times R) \rightarrow \mathbb{R}$ be a quality function that, R as a certain query's all possible sensitive outcome, given a database $D \in D^n$, assigns a score to each outcome $r \in R$. Then we define global sensitivity $GS(q)$ as $\max_r |q(D, r) - q(D', r)|, D, D' \in D^n$. Exponential mechanism M is obtained by returning the sensitive answer r with probability proportional to $\exp(\frac{\epsilon q(D, r)}{2GS(q)})$.*

Insufficiency of Differential Privacy In previous studies, there are two major families of privacy models: *k-anonymity* [4] and ϵ -differential privacy [20]. The former is an anonymity privacy model that is widely adopted in practice, but it is easy to be attacked. The latter is a statistical privacy model that provides theoretical privacy guarantee but it is criticized for adding too much perturbation so that it is hard to be applied in practice. We summarized three crucial weak points of differential privacy that hinder its extensive use in spatiotemporal data publishing.

- **One size for all.** ϵ -Differential privacy use a single parameter ϵ to represent the protection level of all users in the database, whereas the privacy requirements should be personalized.
- **Data independence assumption.** Differential privacy assumes that the tuples in the databases are independent, whereas the data collected from real-world tend to be temporally or spatially correlated.

To this end, we attempt to overcome the above fundamental shortcomings of differential privacy by rigorous and flexible privacy models for utilizing personal spatiotemporal data.

2.2 Differentially Private Sptiotemporal Data Publishing

In this thesis, we mainly focus on the scenario of spatiotemporal temporal data release as illustrated in Figure 2.1 and attempt to make more rigorous and flexible privacy model in this scenario.

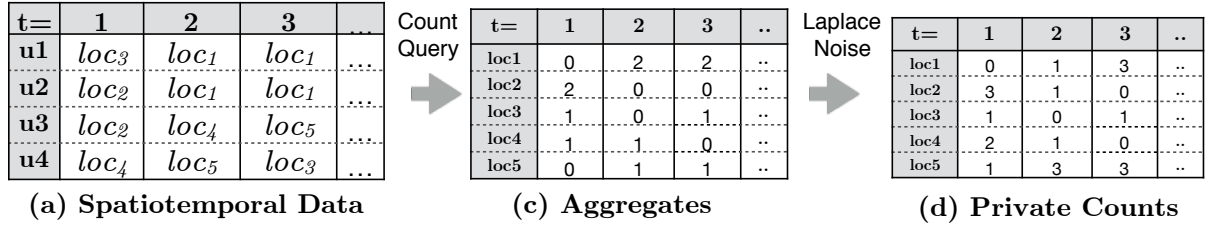


Figure 2.1: Differentially Private Continuous Aggregate Release under Temporal Correlations.

In this scenario, the database containing personal spatiotemporal data are generated by multiple users at each time point as shown in Figure 2.1(a). Essentially, the personal spatiotemporal data are triple tuple $\langle UID, location, time \rangle$. Taking first cell in Table (a) for example, it indicates that user $u1$ is at location $loc3$ at time $t1$. The cells can be other types of data with geographical information, such as users' activities or health information at a location. For simplicity, we only consider location data. The goal is to release information from the raw spatiotemporal data, i.e., the answers of statistical queries (e.g., count query, average query) over the database. Such answers are called as *aggregates* since they are generated by statistical queries. Because of privacy, we cannot directly release such raw aggregates (i.e., Table (c)) and need to make them private using differential privacy. Another notable feature of this scenario is that the private data are released over time since the spatiotemporal are generated continuously.

This setting is well researched in previous studies, as known as *differential privacy under continual observation* in the literature [23] [24] [25] [26] [27] [28] [29] [30] [31].

Previous studies just simply apply differential privacy model to design sophisticated mechanisms for different research problem. However, none of them tackle the two fundamental problems as we mentioned in Section 2.1. We will show that, by personalized privacy model, data utility of the released data can be improved significantly. On the other hand, data correlation poses a serious problem of unexpected privacy loss, which is neglected in previous studies. A naive solution is to add more perturbation at the cost of losing data utility. We will propose temporal and spatial differential privacy model that are able to achieve better data utility without unexpected privacy loss.

2.3 Metrics of Data Utility

Basically, data utility in PPDP is measured as the similarity between the released data and the original one. Since differential privacy is satisfied by adding randomness to the real data, the data utility of the released data is measured as the magnitude of random noises. We adopt Mean of Absolute Error (MAE) as shown below. Here, $\mathbf{r}_i$ and $\mathbf{n}_i$ as vectors of length $|\mathbf{locs}|$ are real and noisy data at timestamp i , respectively. $\mathbf{R}$ and $\mathbf{N}$ are real and noisy data of all timestamps, respectively. The error magnitude is measured as Equations (2.2) and (2.3).

$$MAE(\mathbf{r}_i, \mathbf{n}_i) = \frac{1}{|\mathbf{locs}|} \cdot \sum_{j=1}^{|\mathbf{locs}|} |\mathbf{r}_i[j] - \mathbf{n}_i[j]| \quad (2.2)$$

$$MAE(\mathbf{R}, \mathbf{N}) = \frac{1}{|T|} \cdot \sum_{i=1}^{|T|} MAE(\mathbf{r}_i, \mathbf{n}_i) \quad (2.3)$$

Similarly, Mean of Relative Error (MRE) and Mean of Squared Error (MSE) are defined as follows.

$$MRE(\mathbf{R}, \mathbf{N}) = \frac{1}{|T| * |\mathbf{locs}|} \cdot \sum_{i=1}^{|T|} \sum_{j=1}^{|\mathbf{locs}|} \frac{|\mathbf{r}_i[j] - \mathbf{n}_i[j]|}{\max\{\delta, \mathbf{r}_i[j]\}} \quad (2.4)$$

$$MSE(\mathbf{R}, \mathbf{N}) = \frac{1}{|T| * |\mathbf{locs}|} \cdot \sum_{i=1}^{|T|} \sum_{j=1}^{|\mathbf{locs}|} |\mathbf{r}_i[j] - \mathbf{n}_i[j]|^2 \quad (2.5)$$

2. Background

KL-divergence is widely used to measure the similarity of two probability distributions. It is a non-symmetric measure that computes the information lost when we use published noisy data $\mathbf{N}$ to estimate real data $\mathbf{R}$, as known as relative entropy. To get the probability distribution, we normalize the real and noisy data at each timestamp, which is denoted by $\tilde{\mathbf{r}}_i$ and $\tilde{\mathbf{n}}_i$ respectively.

$$D_{KL}(\mathbf{R}||\mathbf{N}) = \frac{1}{|T|} \cdot \sum_{i=1}^{|T|} \sum_{j=1}^{|\mathbf{locs}|} \ln \left(\frac{\tilde{\mathbf{r}}_i[j]}{\tilde{\mathbf{n}}_i[j]} \right) \tilde{\mathbf{r}}_i[j] \quad (2.6)$$

All the above mentioned metrics are the lower the better.

CHAPTER 3

ℓ -TRAJECTORY PRIVACY *

3.1 Motivation

Recent emerging mobile and wearable technologies make it easy to collect personal spatiotemporal data such as activity trajectories in daily life. Publishing real-time statistics over trajectory streams produced by crowds of people is expected to be valuable for both academia and business, answering questions such as “How many people are in Kyoto Station now?” However, analyzing these raw data will entail risks of compromising individual privacy. ϵ -Differential Privacy has emerged as a well-known standard for private statistics publishing because of its guarantee of being rigorous and mathematically provable. However, since user trajectories will be generated infinitely, it is difficult to protect every trajectory under ϵ -differential privacy. On the other hand, in real life, not all users require the same level of privacy. To this end, we propose a flexible privacy model of ℓ -trajectory privacy to ensure every desired length of trajectory under protection of ϵ -differential privacy. We also design an algorithmic framework to publish ℓ -trajectory private data in *real time*. Experiments using four real-life datasets show that our proposed algorithms are effective and efficient.

*Copyright ©2016 IEICE.

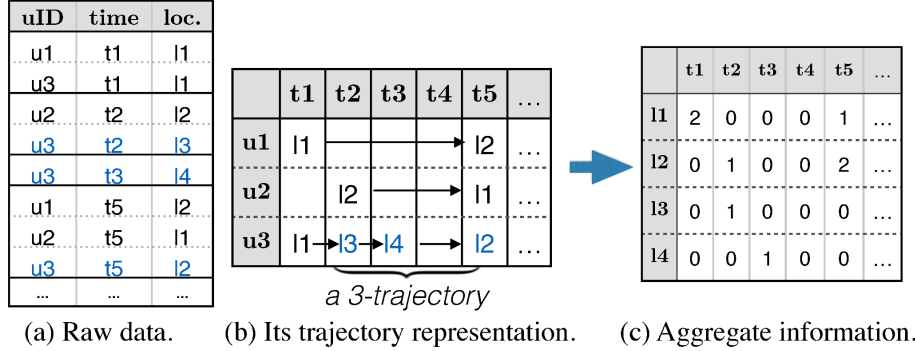


Figure 3.1: Raw data and aggregate information. Each row in Table (a) is a spatiotemporal data point.

3.2 Introduction

In recent years, personal data have been increasingly collected, stored, and analyzed. The information is sensitive, especially location data showing where one has visited. These successive points of interest (POIs), along with timestamps, constitute our moving trajectories in daily life. Releasing real-time statistics of trajectory streams produced by crowds of people is expected to be extremely valuable for data-based analysis and decision making. Consider the scenario illustrated in Fig.3.1. A trusted server is collecting many people’s moving trajectories continuously. The raw data are spatiotemporal points, as illustrated in Fig.3.1(a). User u ’s ℓ_u -trajectory is defined as ℓ_u successive (maybe not adjacently) spatiotemporal data points on the timeline, as illustrated in Fig.3.1(b). For example, $l1 \rightarrow l3 \rightarrow l4$ and $l3 \rightarrow l4 \rightarrow l2$ are $u3$ ’s two overlapped 3-trajectories. Then statistics at each timestamp such as “How many people are in location l_1 at the current time?” can be published such as Fig.3.1(c). These statistical answers are expected to be useful for marketing analysis[†], real-time traffic analysis [29], intelligent navigation systems [32], web browsing behavior mining [33] (users’ trajectories in the cyberspace) and so on. To protect privacy, such sensitive data should be anonymized. However, previous research has pointed out that even anonymized information can be used to identify a particular person. Several well-known privacy disclosure cases have arisen, such as an attack on the Netflix database [6] and medical records of the governor of Mas-

[†]Mobile spatial statistics (in Japanese), <https://www.nttdocomo.co.jp/corporate/disclosure/mobilespatialstatistics/>

sachusetts [4]. Such attacks are called “linkage attack” [34] by which an adversary uses auxiliary information to obtain sensitive data about a target. For example, if an attacker knows that $u3$ has stayed at some places at timestamps $t1, t2, t3$, then by the published data of Fig.3.1(c), the trajectory of $u3$ can be inferred as either $l1 \rightarrow l2 \rightarrow l4$ or $l1 \rightarrow l3 \rightarrow l4$. Especially, people’s moving trajectories are extremely vulnerable. Research [7] shows that only *four* spatiotemporal points from anonymized mobile datasets are sufficient to identify 95% of individuals uniquely. Therefore, a rigorous privacy model is necessary.

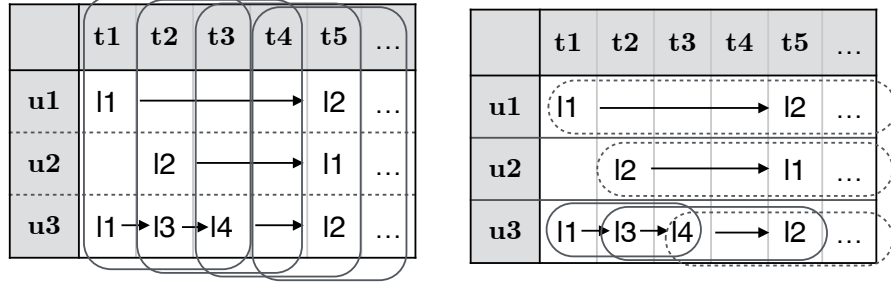
ϵ -Differential Privacy (DP), has emerged as a well-known standard for privacy preserving data publishing (PPDP) because of rigorous theoretical guarantees of robustness under arbitrary linkage attack [28]. ϵ is a positive parameter called *privacy budget* which is given in advance to control the privacy level. The larger ϵ , the lower the privacy level. Roughly speaking, under ϵ -DP, the modification (e.g., removing or changing the values) of any single user’s data is guaranteed to have a limited effect (bounded by ϵ) on the outcome of data publishing. In this way, the data privacy of any single user is protected from arbitrarily analyzing the published data (by attackers or users). However, DP is criticized by practitioners because of sacrificing too much data utility (e.g., adding too much random noises in order to achieve DP). There are only few real-world applications of differential privacy such as US census* private data publishing [35], iDASH project [36] for sharing differentially private data in biology and life-science research and Privacy Integrated Queries (Pinq) language† for differentially-private programming. In addition, it is hard to apply DP to real-time data publishing over infinite stream data in practice, because the user may produces data points infinitely and unpredictably.

To this end, a flexible privacy model of *w-event privacy* [30] based on ϵ -DP for infinite stream data was proposed recently. Instead of protecting the whole length of stream data, it focuses on protecting the data within any single sliding window. Besides the parameter ϵ , it introduces an additional parameter w to control the size (the number of contiguous timestamps) of sliding windows, to guarantee that the trajectories within any single sliding window will be protected under ϵ -DP‡. Regarding *3-event privacy* in Fig.3.2(a), it guarantees that the

* <http://onthemap.ces.census.gov/>

† <http://research.microsoft.com/en-us/projects/pinq/tutorial.aspx>

‡A trajectory being protected under ϵ -DP implies that removing or changing this trajectory should have a limited effect (bounded by ϵ) on the outcome of data publishing.



(a) w -event privacy [14] ($w=3$). (b) ℓ -trajectory privacy ($\ell=\langle 3, 3, 3 \rangle$).

Figure 3.2: Previous privacy model v.s. our proposed model. Trajectories in dashed ellipses in (b) means ongoing 3-trajectories.

following trajectories will be protected under ϵ -DP: $u1$'s 1-trajectories $l1$ and $l2$, $u2$'s 1-trajectories $l2$ and $l1$, $u3$'s 3-trajectory $l1 \rightarrow l3 \rightarrow l4$ (including all of the sub-trajectories such as $l1 \rightarrow l3$) and $u3$'s 2-trajectory $l4 \rightarrow l2$.

However, w -event privacy is not sufficient to provide *uniform* and *personalized* protection for trajectory streams. First, it is hard to uniformly protect *every* ℓ_u -trajectory for a given ℓ_u under ϵ -DP by its sliding window methodology. For example, regarding 3 -event privacy in Fig.3.2(a), the following trajectories are not guaranteed to be protected under ϵ -DP because they exceeds any single 3-length sliding window: $u1$'s 2-trajectory $l1 \rightarrow l2$, $u2$'s 2-trajectory $l2 \rightarrow l1$, $u3$'s 4-trajectory $l1 \rightarrow l3 \rightarrow l4 \rightarrow l2$. In general, there is no upper bound in the number of contiguous timestamps a ℓ_u -trajectory spans[§]. Second, different individuals may have different expectations for the privacy of their trajectories. For example, if $u1$ wants his/her every 10-trajectory to be protected, while $u2$ only wants every 3-trajectory to be protected, it is hard to achieve both of them by w -event privacy. Therefore, a fine-grained privacy model for trajectory streams is necessary.

For those reasons, we propose ℓ -trajectory privacy, where a vector ℓ is defined as $\langle \ell_1, \dots, \ell_{|U|} \rangle$. Let U be the set of all users, and ℓ_u is user u 's desired length of being protected trajectories, namely *privacy preference* of user u . Our model guarantees that every ℓ_u -trajectory is protected under ϵ -DP. Fig.3.2(b) illustrates the protected range of all users' 3-trajectory privacy.

In this chapter, we formulate ℓ -trajectory privacy, and prove that it can be

[§]In some cases, users' spatiotemporal data points are collected at each timestamps (dense trajectories). However, in most cases, only interesting attributes (e.g., Points of Interest, POIs) will be reserved after data cleaning.

achieved. Then, we design algorithms to achieve ℓ -trajectory privacy and boost data utility. The algorithm will be composed of sub-algorithms. Each one of them takes collected database at each timestamp as input, and outputs private data in real time. A salient challenge to achieve ℓ -trajectory privacy is that the upcoming spatiotemporal data points are unknown, hence, a naive method that pre-allocates the local budget will lead to inferior data utility. To this end, we propose an algorithmic framework that adopts dynamic budget allocation and approximate publishing. Then, two algorithms, GA+Adj and GA+MMD are implemented, as two approximation strategies.

The rest of this chapter is organized as follows. Section 3.3 briefly surveys the related works on differential privacy of data streams and private trajectory data publishing. Section 3.4 gives background knowledge of differential privacy and classical mechanisms to achieve it. Section 3.5 provides the problem formulation. Section 3.6 presents an overview of our framework, as well as the technical details of two proposed algorithms. Section 3.7 describes the data set and presents a set of experimental results.

3.3 Related Work

3.3.1 Differential Privacy on Streams

For stream data, *user-level* privacy (e.g., to protect the whole trajectory stream of a user) and *event-level* (e.g., to protect any single spatiotemporal data point) privacy have been proposed [28] as two different privacy goals. Previous work mainly focuses on event-level privacy on finite or infinite streams [26], [24, 30], and user-level privacy on finite streams [37], [29, 33]. Chan et al. [26] proposed scheme of p -sum to construct a full binary tree on the sequential data, where each node contains the sum of the sequential data in its subtree, plus noise with scale logarithmic in the length of the sequence. When the noise is accumulated up to a threshold p , it makes a release. Bolot et al. proposed a scheme of decayed privacy [24], which consider past data is not as sensitive as current data. Mir et al. [37] considers counters that may also decrease. Fan and Xiong [29, 33] proposed FAST framework for publishing time-series data in a user-level private way. FAST uses sampling and filtering components to reduce the noise; given a specified number of samples, the filtering component predicts the future data and corrects

its prior data by noisy samples. The authors report that their adaptively sampling scheme preserves high utility at the same privacy level. However, this scheme takes as input the total amount of timestamps $|T|$, which leads to inapplicability in our infinite scenario.

3.3.2 Privacy of Trajectory Data

The major work in previous research specifically examined techniques of anonymization [38–40] and scenarios of directly publishing trajectory data [41, 42]. Little work connects trajectory data and differential privacy. Chen et al. [42] proposed a data-dependent sanitation mechanism by grouping trajectories to produce a noisy prefix tree. Jiang et al. [43] also examined differentially publishing private trajectories by sampling suitable direction and distance of trajectory data. Ho et al. [44, 45] proposed a differentially private approach to mine interesting geographic location patterns from trajectory data. These works address the scenario of directly publishing user-level trajectories or the results of specific data mining tasks, which are different from publishing sequential statistics for sets of users' trajectories.

3.4 Preliminary

3.4.1 Differential Privacy

Differential privacy was proposed by Dwork et al. [20]. Roughly speaking, differential privacy attempts to limit the impact of an individual's record relative to the answer of a query on the database. In other words, whether the sensitive information is present or not in the database, the answers of a query should have little difference. In this way, an adversary cannot associate any piece of information with a specific individual from mining the answers of queries.

Definition 2 (Neighboring Database). *If D is a database, and D' is a copy of D but differing in one record, then we say D and D' are two neighboring databases.*

Definition 3 (ϵ -Differential Privacy, ϵ -DP). Λ is a randomized algorithm. $\mathbf{R}$ represents all possible outputs of Λ . Algorithm Λ satisfies ϵ -differential privacy,

if for any $r \in \mathbf{R}$ and any two neighboring databases D, D' , we have the following.

$$\Pr[\Lambda(D) = r] \leq \exp(\epsilon) \cdot \Pr[\Lambda(D') = r] \quad (3.1)$$

In Inequality (3.1), ϵ is a *privacy budget* given in advance. It is used to control the privacy level. We note that lower ϵ signifies higher privacy and more randomness, and vice versa. *Global sensitivity* (or *sensitivity* for short) $GS(Q)$ of query Q is the maximum L_1 distance between the query results for any two neighboring databases.

To avoid misunderstandings, we note that there are different ways to define differential privacy, based on the definition of neighboring databases that differ by adding/removing an individual's record (*unbounded*) [20], or modifying the content of an individual's record but keep the size of database (i.e., number of records) (*bounded*) [21]. Unless we explicitly state otherwise, we use *bounded* differential privacy in this chapter. Further discussion of these differences can be found in [46].

3.4.2 Mechanisms of Differential Privacy

Two widely used methods to achieve differential privacy is Laplace mechanism [21] which adds random noise to actual data to prevent the disclosure of sensitive information, and Exponential mechanism [22] which randomly returns the desired item by some calibrated probability.

Theorem 3 (Laplace Mechanism [21]). *ϵ -DP can be achieved by adding independent Laplace random noise x to the answer of query Q . In the following, n and D denote the noisy data and a given database, respectively.*

$$n = Q(D) + x \quad (3.2)$$

$$x \sim \Pr(x|\lambda) = \frac{1}{2\lambda} \cdot \exp(-|x|/\lambda) \quad (3.3)$$

where λ is a scale parameter of Laplace distribution, which equals to $GS(Q)/\epsilon$.

Another way to obtain differential privacy is through the Exponential mechanism [22]. Given a quality function q that scores results of a specified query, where higher scores are better. Then returning results in the probability of exponentially proportional to the corresponding score. In other words, the result

with a higher score is exponentially more likely to be chosen. This will ensure differential privacy.

Theorem 4 (Exponential Mechanism [22]). *Let D^n denote all possible databases in the universe, and $D, D' \in D^n$ are two instances of neighboring databases. Let $q : (D^n \times R) \rightarrow \mathbb{R}$ be a quality function that, R as a certain query's all possible sensitive outcome, given a database $D \in D^n$, assigns a score to each outcome $r \in R$. Then we define global sensitivity $GS(q)$ as $\max_r |q(D, r) - q(D', r)|, D, D' \in D^n$. Exponential mechanism M is obtained by returning the sensitive answer r with probability proportional to $\exp(\frac{\epsilon q(D, r)}{2GS(q)})$.*

The following sequential composition theory reveals the method of integrating multiple algorithms to achieve differential privacy.

Theorem 5 (Sequential Composition [47]). *Presuming that A_i satisfies ϵ_i -DP over database D , then an integrated algorithm Λ that consists of multiple A_i over the same database D satisfies $\sum \epsilon_i$ -DP.*

3.5 Problem Definition and Privacy Model

3.5.1 Data Model

We consider scenario of a trusted server that collects database D_i of users' spatiotemporal data points continuously at each timestamp $i \in [1, t]$. Let t denote the current timestamp. Each data point $\langle uID, time, loc \rangle$ is a row in D_i (Fig.3.1(a)). Assuming that **locs** is a set of all locations in which we were interested (POIs), then the server wishes to publish a vector $\mathbf{r}_i$ at each timestamp i as the counts of $loc \in \mathbf{locs}$ appeared in D_i (Fig.3.1(c)), i.e., the answer of count query $Q^c : D_i \rightarrow \mathbb{R}^{|\mathbf{locs}|}$. We assume that a user only appears at most *one* location at each timestamp, in other words, $\langle uID, time \rangle$ is the primary key in the whole database.

Infinite D_i constitutes trajectory streams. Prefixes of the infinite trajectory streams are all the database up to the current timestamp t , denoted by $S_t = \{D_1, \dots, D_t\}$.

Definition 4 (ℓ_u -trajectory). *A sequence of successive spatiotemporal data points produced by user u is a ℓ_u -trajectory if the number of data points is equal to a*

natural number ℓ_u . A ℓ_u -trajectory ends at timestamp k , as denoted by $\ell_{u,k} = \{\langle u, i, loc \rangle, \dots, \langle u, k, loc' \rangle\}$ where $loc, loc' \in \mathbf{locs}$. We say that the set of timestamps appeared in these data points (i.e., $\{i, \dots, k\}$), denoted by $\tau_{u,k}$, is dominated by $\ell_{u,k}$.

For convenience, we summarize the notations and symbols used in this chapter in Table 3.1.

Table 3.1: Summary of Notations.

ϵ	Total privacy budget
ε_i	Privacy budget variable at timestamp i
ℓ_u	The length of user u 's being protected trajectories
$\boldsymbol{\ell}$	A vector of all users' ℓ_u
t	The current timestamp
i, k	Variables of timestamps
D_i, D'_i	Collected database at timestamp i and the corresponding neighboring database
T	Set of all timestamps
U, us_i	Set of users in databases of the whole timeline and D_i respectively
$\ell_{u,k}$	A ℓ_u -trajectory of user u ends with timestamp k
$\tau_{u,k}$	Set of timestamps in $\ell_{u,k}$
$\mathbf{locs}$	Vector of POIs
j	Index variable on POIs
S_t, S'_t	Trajectory stream prefixes and the corresponding neighboring stream prefixes
Q^c	Count query
$\mathbf{R}, \mathbf{r}_i$	Real statistics of all timestamps and timestamp i respectively
$\mathbf{N}, \mathbf{n}_i$	Noisy statistics of all timestamps and timestamp i respectively
A_i	Sub-algorithm at timestamp i
Λ	An integrated algorithm consisting of $A_i, i \in [1, t]$

3.5.2 Privacy Risk

In our scenario, publishing the real statistics will make users' privacy at risk. As described above, even releasing anonymized information is maybe not safe because of the adversary's unforeseen background knowledge (linkage attack [34]). For instance, in the example of Fig.3.1, assuming that an informed adversary knows that u_3 has stayed at some places at timestamps t_1, t_2, t_3 , then by the published data Fig.3.1(c), the trajectory of u_3 can be inferred as either $l_1 \rightarrow l_2 \rightarrow l_4$ or $l_1 \rightarrow l_3 \rightarrow l_4$.

To avoid such risks, we will propose a privacy model based on differential privacy that does not make any assumption on the background knowledge of the attacker. We will give a formal definition of our privacy goal under ϵ -DP in Section 3.5.3.

3.5.3 Proposed Privacy Model

To define our model under ϵ -DP, we must first clarify the definition of neighboring databases under our setting.

Definition 5 (Neighboring databases at timestamp i). *If a database D_i is collected at timestamp $i \in [1, t]$, and D'_i is a copy of D_i but differing in one loc in a single record of an arbitrary user u , then we say that D_i, D'_i is a pair of neighboring databases with respect to u .*

Since we assume that a user only appears at most one *loc* at each timestamp, and modifying single *loc* means either change *loc* to another $loc' \in \mathbf{locs}$ or $loc' \notin \mathbf{locs}$, according to the definition of differential privacy, the sensitivity of Q^c is 2.

We define ℓ_u -trajectory neighboring stream prefixes as follows. Intuitively, they are trajectory streams differing in a ℓ_u -trajectory of user u .

Definition 6 (ℓ_u -trajectory neighboring stream prefixes). *Let S'_t be a copy of trajectory stream prefixes S_t but differing in ℓ_u neighboring databases D'_i . S_t and S'_t are ℓ_u -trajectory stream prefixes neighboring each other if one is obtained from another by modifying single or multiple locs in any one ℓ_u -trajectory $\ell_{u,k}$ (recall that a ℓ_u -trajectory is a set of ℓ_u spatiotemporal data points). We say that S_t and S'_t are neighboring with respect to $\ell_{u,k}$.*

Then we define ℓ -trajectory privacy where ℓ is a vector consisting of all user's privacy preferences as follows. It is attempted to cover the impact of any ℓ_u -trajectory relative to the answer of a sensitive query.

Definition 7 (ℓ -trajectory ϵ -differential privacy). *Let $\ell = \langle \ell_1, \dots, \ell_{|U|} \rangle$. Let Λ be an algorithm that takes prefixes of trajectory streams $S_t = \{D_1, \dots, D_t\}$ as inputs. Let $\mathbf{N}_t = \{\mathbf{n}_1, \dots, \mathbf{n}_t\}$ be a possible perturbed output stream of Λ . If for any ℓ_u -trajectory neighboring S_t and S'_t , the following holds,*

$$\Pr[\Lambda(S_t) = \mathbf{N}_t] \leq e^\epsilon \cdot \Pr[\Lambda(S'_t) = \mathbf{N}_t] \quad (3.4)$$

then we say that Λ satisfies ℓ -trajectory ϵ -differential privacy (simply, ℓ -trajectory privacy).

3.5.4 Methodology to Achieve ℓ -Trajectory Privacy

The next question is how to achieve ℓ -trajectory privacy. We prove Theorem 6. Roughly speaking, to satisfy ℓ -trajectory privacy, the sum of privacy budgets allocated to sub-algorithms at the timestamps of any single ℓ_u -trajectory must be less than or equal to the total privacy budget ϵ .

Theorem 6. *Let Λ be an integrated algorithm which takes stream prefixes $S_t = \{D_1, \dots, D_t\}$ as input, and $\mathbf{N}_t = \{\mathbf{n}_1, \dots, \mathbf{n}_t\}$ as output. Λ consists of a series of sub-algorithms $\{A_1, \dots, A_{j_t}\}$, where each A_i takes D_i as input, and outputs noisy data $\mathbf{n}_i$ with independent randomness.*

Presume A_i ensures ϵ_i -DP, and let $\tau_{u,k}$ be the set of timestamps dominated by an arbitrary trajectory $\ell_{u,k}$, then Λ satisfies ϵ -DP, if

$$\text{for } \forall u, \forall k, \sum_{i \in \tau_{u,k}} \epsilon_i \leq \epsilon \text{ holds.} \quad (3.5)$$

Proof. Presume the integrated algorithm Λ takes inputs as S_t and S'_t independently, which are two ℓ_u -neighboring stream prefixes with respect to $\ell_{u,t}$, and the outputs are $\mathbf{N}_t, \mathbf{N}'_t$ respectively. Since each A_i has independent randomness, the followings hold.

$$\Pr[\Lambda(S_t) = \mathbf{N}_t] = \prod_{i=1}^t \Pr[A_i(D_i) = \mathbf{n}_i] \quad (3.6)$$

$$\Pr[\Lambda(S'_t) = \mathbf{N}_t] = \prod_{i=1}^t \Pr[A_i(D'_i) = \mathbf{n}_i] \quad (3.7)$$

Dividing both side of Equation (3.6) by (3.7) respectively, we have the following.

$$\frac{\Pr[\Lambda(S_t) = \mathbf{N}_t]}{\Pr[\Lambda(S'_t) = \mathbf{N}_t]} = \frac{\prod_{i=1}^t \Pr[A_i(D_i) = \mathbf{n}_i]}{\prod_{i=1}^t \Pr[A_i(D'_i) = \mathbf{n}_i]} \quad (3.8)$$

Since each A_i are ε_i -DP, we derive the following.

$$\frac{\prod_{i=1}^t \Pr[A_i(D_i) = \mathbf{n}_i]}{\prod_{i=1}^t \Pr[A_i(D'_i) = \mathbf{n}_i]} \leq e^{\sum \varepsilon_i} \quad (3.9)$$

According to Definition 6 of ℓ_u -trajectory neighboring stream prefixes, the set of timestamps in Equation (3.8) is $\tau_{u,k}$. Therefore, by Equation (3.8) and Inequality (3.9) we obtain the following.

$$\frac{\Pr[\Lambda(S_t) = \mathbf{N}_t]}{\Pr[\Lambda(S'_t) = \mathbf{N}_t]} \leq e^{\sum \varepsilon_i}, i \in \tau_{u,k} \quad (3.10)$$

Finally, by Inequality (3.5), for any ℓ_u -trajectory neighboring stream prefixes S_t and S'_t , the following holds.

$$\frac{\Pr[\Lambda(S_t) = \mathbf{N}_t]}{\Pr[\Lambda(S'_t) = \mathbf{N}_t]} \leq e^{\sum \varepsilon_i} \leq e^\epsilon, i \in \tau_{u,k} \quad (3.11)$$

By Definition (7), Λ is ℓ -trajectory private. $\square$

3.6 Algorithms

Data utility is defined in Chapter 2. MAE can be considered as a utility goal in the our algorithm design.

3.6.1 First-cut Solution

According to the analysis in Section 3.5.4, the essential problem to achieve ℓ -trajectory privacy is how to allocate appropriate privacy budgets at each timestamp. The straightforward idea is to *uniformly* allocate the privacy budget of ϵ/ℓ_{max} at each timestamp, where ℓ_{max} is the maximum value in ℓ , as shown in Algorithm 1. The integrated algorithm UNIFORM includes UNIFORM_{*i*} of each timestamp $i \in [1, t]$.

Theorem 7. UNIFORM *satisfies ℓ -trajectory privacy.*

Algorithm 1: UNIFORM _{t}

Input: Database D_t ; Vector of trajectory length ℓ ; Privacy budget ϵ

Output: noisy data $\mathbf{n}_t$

- 1 Calculate the real statistics $\mathbf{r}_t \leftarrow Q^c(D_t)$
 - 2 Add scaled Laplace noises $\mathbf{n}_t \leftarrow \mathbf{r}_t + \langle \text{Lap}(2\ell_{\max}/\epsilon) \rangle^{|locs|}$
 - 3 **return** $\mathbf{n}_t$
-

Proof. As the global sensitivity of Q^c is 2, by Laplace mechanism, each UNIFORM _{i} , $i \in [1, t]$ satisfies $\epsilon/\ell_{\max}$ -DP. In addition, because of $\epsilon/\ell_{\max} \leq \epsilon/\ell_u$, the sum of privacy budgets $\epsilon/\ell_{\max}$ of UNIFORM _{i} at the timestamps of any *single* ℓ_u -trajectory will be at most ϵ . By Theorem 6, UNIFORM satisfies ℓ -trajectory privacy. $\square$

3.6.2 Proposed Framework

UNIFORM leaves us no space for optimization. To optimize privacy budget allocation, an idea is to dynamically allocate budgets at each timestamp according to the data points distribution. Besides ensuring ℓ -trajectory privacy, we also wish to spend more budget when it is “worth”, and vice versa. How to measure whether it is a worthwhile investment or not? This question leads us to develop approximation strategy cooperating with dynamic budget allocation component to raise data utility.

We propose the algorithmic framework shown in Fig.3.3 and Algorithm 2. It is composed of three components. *Dynamic budget allocation* component allocates budgets at each timestamp and ensures Inequality (3.5) (to satisfy ℓ -trajectory privacy). *Private approximation* component makes a decision of whether it is benefit to approximately publish or not, under a certain utility goal (such as MAE) and an approximation strategy (such as republishing). *Private publishing* component releases private (noisy) data by a certain differential privacy mechanism (such as Laplace mechanism). The latter two components need to be assigned privacy budgets, because they need to access the real statistics $\mathbf{r}_t$ to make a better decision. Note that the set of users who appear in D_t , denoted by us_t , is considered as non-sensitive data. The reason is that under the Definition 5 of neighboring databases, us_i will not be changed between any two neighboring databases.

Proposition 1. *Given ℓ and a total privacy budget ϵ at timestamp t , if (i) the*

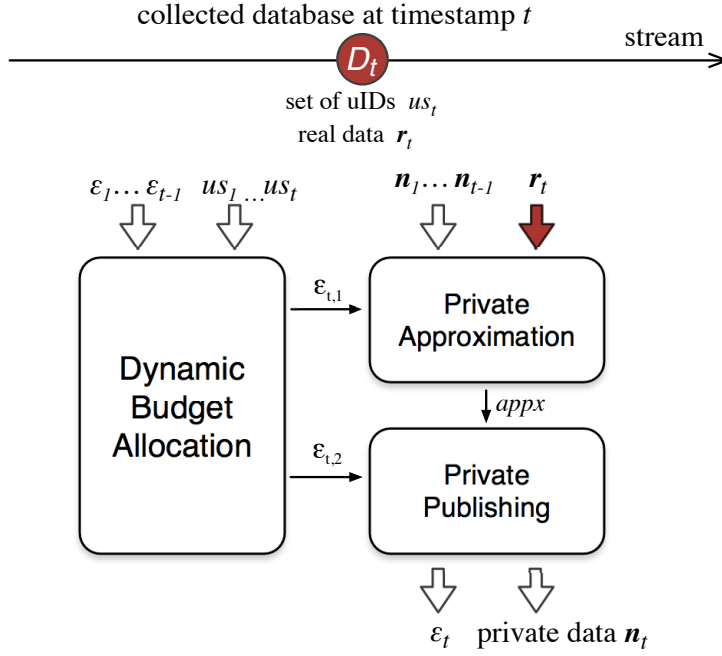


Figure 3.3: Illustration of our proposed framework. The dark objects contain sensitive data.

Algorithm 2: ℓ -Trajectory Private Data Publishing

Input: Real statistics $\mathbf{r}_t$; Users sets $us_i, i \in [1, t]$; Vector of privacy preferences ℓ ; Total privacy budget ϵ ; Previously allocated budget $\epsilon_i, i \in [1, t - 1]$; noisy data $\mathbf{n}_i, i \in [1, t - 1]$

Output: Noisy statistics $\mathbf{n}_t$, Allocated budget ϵ_t

- 1 $\epsilon_t \leftarrow \text{Dynamic Budget Allocation}_t$
 - 2 $appx \leftarrow \text{Private Approximation}_t$
 - 3 $\mathbf{n}_t \leftarrow \text{Private Publishing}_t$
-

Dynamic Budget Allocation component (row 1) allocating privacy budget ϵ_i at timestamp i ($i \in [1, t]$), ensures Inequality (3.5) of Theorem 6; and (ii) the *Private Approximation* component (row 2) and the *Private Publishing* component (row 3), achieve overall ϵ_t -DP. Then Algorithm 2 satisfies ℓ -trajectory privacy.

Proof. Since the *dynamic budget allocation* component only takes private data as input, the three components achieve ϵ_t -DP overall. According to Theorem 6, the conclusion holds. $\square$

In the following sections, we will design a greedy algorithm (GA) in the *dy-*

Algorithm 3: Dynamic Budget Allocation(GA_t)

Input: Users set $us_i, i \in [1, t]$; Previously allocated privacy budget $\varepsilon_i, i \in [1, t - 1]$; Vector of privacy preferences ℓ ; Total privacy budget ϵ

Output: privacy budget $\varepsilon_{t,1}, \varepsilon_{t,2}$

- 1 Allocate a fixed budget $\varepsilon_{t,1} \leftarrow \epsilon / (2 * \ell_{max})$
 - 2 Calculate spent budget $\varepsilon_t^s \leftarrow \max_{u \in us_t} \{\sum \varepsilon_i, i \in \tau_{u,t}\}$
 - 3 Calculate remainder budget $\varepsilon_t^a \leftarrow \epsilon / 2 - \varepsilon_t^s$
 // exponentially decreasing
 - 4 Allocate dynamic budget $\varepsilon_{t,2} \leftarrow \varepsilon_t^a / 2$
 - 5 **return** $\varepsilon_{t,1}, \varepsilon_{t,2}$
-

dynamic budget allocation component, and investigate two different approximation strategies which are republishing the *adjacent* (Adj) and *globally most similar* (MMD) noisy data based on utility metric MAE, then adopt Laplace mechanism in *private publishing* component. We note that the framework is orthogonal to its implementation. For example, we can define different utility goal (other than MAE), or adopt different approximation scheme and differential privacy mechanisms.

3.6.3 Dynamic Budget Allocation

The first question about privacy budget allocation is that how to decide the budget proportions of the left two components in Fig.3.3. We will allocate a fixed budget $\varepsilon_{t,1}$ to the *private approximation*, and dynamic budget $\varepsilon_{t,2}$ to the *private publishing* component. The reason is that, as the *private approximation* component will decide whether it is necessary to republish or not, it significantly affects the data utility. Therefore, we will allocate a fixed budget (a fixed level of randomness) in order to make it more stable.

Algorithm 3 describes how to dynamically allocate budget to the *private publishing* component in an exponentially decay fashion at the current timestamp t . First, it calculates the maximum available budget of the current timestamp under constraint of Inequality (3.5) by checking every user $\ell_{u,t}$ for every u . Then, as the algorithm may also need budgets at the future timestamps, it assigns half of the available amount to the current one.

Lemma 1. *Algorithm 3 (GA_t) ensures that the sum of privacy budgets on any*

one ℓ_u -trajectory is at most ϵ .

Proof. Two parts of budgets are allocated on each ℓ_u -trajectory: the fixed budgets $\varepsilon_{t,1}$ and the dynamic budgets $\varepsilon_{t,2}$. We will prove that both parts are at most $\epsilon/2$. In Line 1, it is uniformly allocated $\epsilon/(2*\ell_{max})$ at each timestamp. In other words, the sum of the fixed budgets on any ℓ_u -trajectory is at most $\epsilon/2$. In Line 2, since all of the previous budgets are known, it calculates the sums of previously spent budgets on each $\ell_{u,t}, u \in U$, and the maximum one is denoted by ε_t^s . Then the maximum available budget ε_t^a is calculated in Line 3. Finally, in Line 4, it saves half of the total available budget for the future data and assigns another half to $\varepsilon_{t,2}$. Hence, the worst case in *GA* is that a user produces data points at every timestamps. Then, the sum of *dynamically* allocated budgets on this user's ℓ_u -trajectory will be $\epsilon/2^2 + \epsilon/2^3 + \dots + \epsilon/2^{\ell_u+1}$, which is less than $\epsilon/2$. Therefore, the sum of all allocated budgets on any single ℓ_u -trajectory will be less than ϵ . $\square$

3.6.4 Approximation Strategies

Approximation strategies have been investigated in earlier research, such as histogram publishing [48, 49], and statistics on data stream publishing [29, 30]. Instead of directly adding noise to real data, they function by transformation of original data or a query structure to achieve better overall utility.

In our case, we will (i) choose an *appropriate* noisy data which was previously published (Fig.3.4), and (ii) republish it if it is “close to” the real statistics which we want to publish. The distance between the real and noisy data will be measured by MAE of Equation (2.3).

We adopt two schemes in the above mentioned step (i). The first one is to simply employ the adjacent noisy data, as shown in Algorithm 4 (Adj). As the principle of temporal locality, the last published data may appear again. The second scheme is to search the most similar noisy data on the timeline, as shown in Algorithm 5 (MMD). The observation is that the periodically repeating pattern[¶] is an intrinsic characteristic in real-life data. Now we will discuss the step (ii). This idea is the same in Lines 1 to 4 of Algorithm 4 and Lines 4 to 8 of Algorithm 5. Let us take the former as an example. In Line 1, it attempts to make a judgment on whether a republishing is benefit or not. Obviously, in

[¶]E.g., for people flow trajectory data, in general, the peak-hour counts appear around 8 a.m. or 6 p.m. everyday.

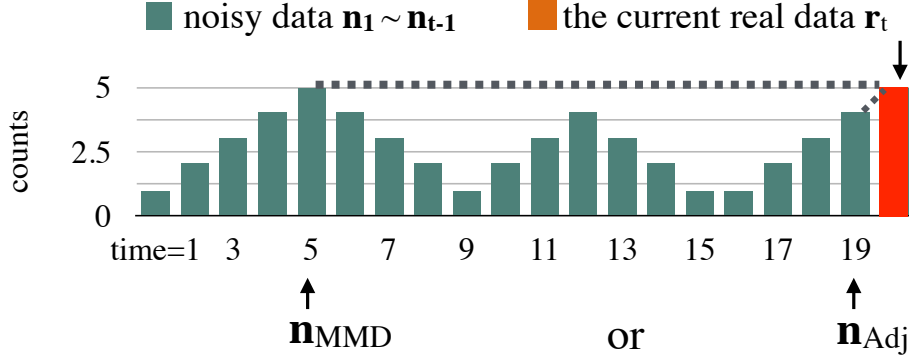


Figure 3.4: Illustration of two approximation strategies.

Algorithm 4: Approximation Strategy (Adj)

Input: Real statistics $\mathbf{r}_t$; Privacy budget $\varepsilon_{t,1}$

Output: timestamp of approximate data

```

1 if  $MAE(\mathbf{r}_t, \mathbf{n}_{t-1}) + Lap(2/(|\mathbf{locs}| * \varepsilon_{t,1})) \leq 2/(|\mathbf{locs}| * \varepsilon_{t,2})$  then
2   |  $appx \leftarrow t - 1$ 
3 else
4   |  $appx \leftarrow t$ 
5 return  $appx$ 
    
```

this inequality, the left part of MAE indicates the error of republishing $\mathbf{n}_{t-1}$ (the similarity between $\mathbf{r}_t$ and $\mathbf{n}_{t-1}$). We now examine the left part. When adding Laplace noise to real data, the error is only introduced by Laplace noises. It is easy to know that MAE between real and Laplace noisy data is numerically equal to the scale parameter of the Laplace noise, that is $2/(|\mathbf{locs}| * \varepsilon_{t,2})$. Hence, the right part of this inequality indicates Laplace error. Since the left part of MAE needs to access real data $\mathbf{r}_t$, it is necessary to add randomness to perturb the result of this judgment. Then, in the following parts of Algorithm 4, the index of noisy data with relatively lower error will be returned. Therefore, It trades off lower error between Laplace and approximation error.

Lemma 2. *Algorithm 4 (Adj) satisfies $\varepsilon_{t,1}$ -DP.*

Proof. Since the sensitive data $\mathbf{r}_t$ is accessed in Line 1 by MAE query which defined in Equation (2.3), it is necessary to add randomness to ensure differential privacy. By Definition 5 of neighboring database D_i, D'_i , the maximum difference (i.e., the sensitivity) of MAE between D_i, D'_i is $2/|\mathbf{locs}|$. Therefore, according to Laplace mechanism, by adding Laplace noises with scale parameter $2/(|\mathbf{locs}| * \varepsilon_{t,2})$

Algorithm 5: Approximation Strategy (MMD)

Input: Real statistics $\mathbf{r}_t$; Privacy budget $\varepsilon_{t,1}$

Output: timestamp of approximate data

```

1  $\varepsilon_{t,1}^m \leftarrow \frac{\varepsilon_{t,1}}{2}, \varepsilon_{t,1}^x \leftarrow \frac{\varepsilon_{t,1}}{2}$ 
2 Calculate  $score_i$  of quality function  $q : -MD(\mathbf{r}_t, \mathbf{n}_i), i \in [1, t-1]$ 
3  $\mathbf{n}_{MMD} \leftarrow \mathbf{n}_i \propto \exp(\frac{\varepsilon_{t,1}^m * score_i}{4}), i \in [1, t-1]$ 
4 if  $MAE(\mathbf{r}_t, \mathbf{n}_{MMD}) + Lap(2/(|\mathbf{locs}| * \varepsilon_{t,1}^x)) \leq 2/(|\mathbf{locs}| * \varepsilon_{t,2})$  then
5   |  $appx \leftarrow MMD$ 
6 else
7   |  $appx \leftarrow t$ 
8 return  $appx$ 
    
```

$\varepsilon_{t,1}$), it satisfies $\varepsilon_{t,1}$ -DP. From Lines 2 to 5, Algorithm 4 does not access any sensitive data, hence the conclusion holds. $\square$

On the other hand, under the utility goal of MAE, we observe that the “most similar” noisy data is equal to the one which holds the minimum Manhattan Distance (Equation (3.13)) to the current real data, denoted by $\mathbf{n}_{MMD}$. That is because Equations (2.2) and (3.12) are only different in a factor of $1/|\mathbf{locs}|$.

$$MD(\mathbf{r}_t, \mathbf{n}_k) = \sum_{j=1}^{|\mathbf{locs}|} |\mathbf{r}_t[j] - \mathbf{n}_k[j]| \quad (3.12)$$

$$MMD(\mathbf{r}_t, \mathbf{n}_i) = \min\{MD(\mathbf{r}_t, \mathbf{n}_i), i \in [1, t]\} \quad (3.13)$$

A challenge arises when we attempt to retrieve $\mathbf{n}_{MMD}$ because MMD query accesses $t-1$ times real data, which means if we adopt Laplace mechanism it will lead to an unacceptable scaled noise. By adopting Exponential mechanism (Theorem 4) and designing an appropriate quality function, we can reduce this noise within acceptable limits. Algorithm 5 implements this idea and uses the negative value of MD query as *quality function* of Exponential mechanism. In Line 1, it spent half of privacy budget of $\varepsilon_{t,1}$ to approximately retrieving $\mathbf{n}_{MMD}$. In Lines 2 and 3, it calculates the quality score and randomly returns the $\mathbf{n}_i$ according the probability of proportional to $\exp(\frac{\varepsilon_{t,1}^m * score_i}{4})$.

Lemma 3. *Algorithm 5 (MMD) satisfies $\varepsilon_{t,1}$ -DP.*

Proof. According to the above analysis, Exponential mechanism ensures that the result of Line 3 is $\varepsilon_{t,1}^m$ -DP, while the procedure of Lines 4 to 7 is $\varepsilon_{t,1}^x$ -differential

private (the proof is similar to Lemma 2, hence omitted here). By the sequential composition of Theorem 5, Algorithm 5 satisfies $(\varepsilon_{t,1}^x + \varepsilon_{t,1}^m)$ -DP, that is $\varepsilon_{t,1}$ -DP. $\square$

3.6.5 Private Publishing

We employ Laplace mechanism in the *privacy publishing* component as shown in Algorithm 6. It takes the real statistics and timestamp of approximate data as inputs, and outputs private statistics. As input, $appx$ is the timestamp of the approximate noisy data which will be republished. If $appx$ is equal to the current timestamp, then the private data will be published by adding Laplace noises. Otherwise, the noisy data at timestamp $appx$ will be republished.

Algorithm 6: Private Publishing

Input: Real statistics $\mathbf{r}_t$; Privacy budget $\varepsilon_{t,2}$; Timestamp of approximate data $appx$
Output: Noisy data $\mathbf{n}_t$

```

1 if  $appx = t$  then
    // add calibrated Laplace noises
2    $\mathbf{n}_t \leftarrow \mathbf{r}_t + \langle Lap(2/\varepsilon_{t,2}) \rangle^{|locs|}, \varepsilon_t \leftarrow \varepsilon_{t,1} + \varepsilon_{t,2}$ 
3 else
    // republish the approximate noisy data
4    $\mathbf{n}_t \leftarrow \mathbf{n}_{appx}, \varepsilon_t \leftarrow \varepsilon_{t,1}$ 
5 Return  $\mathbf{n}_t, \varepsilon_t$ 

```

Lemma 4. *Algorithm 6 satisfies $\varepsilon_{t,2}$ -DP.*

Proof. According to Laplace mechanism [21], adding Laplace noise with scale parameter $2/\varepsilon_{t,2}$ ensures $\varepsilon_{t,2}$ -DP under the sensitivity of 2. $\square$

Theorem 8. *$GA+Adj$ satisfies ℓ -trajectory privacy.*

Proof. In Proposition 1, the condition (i) is true according to Lemma 1, and the condition (ii) is true because of Lemmas 2 and 4. Hence, the conclusion follows. $\square$

Theorem 9. *$GA+MMD$ satisfies ℓ -trajectory privacy.*

Proof. This theorem can be proved in the same method as shown before. The conclusion follows from Lemmas 1, 3 and 4. $\square$

Table 3.2: Real-life datasets.

	PFlow	Geolife	T-Drive	WC98
$ T $	1,694	1,440	886	2,089
$ U $	11,406	170	2,698	12,390
$ locs $	18	56	21	4,016
Data Points	102,468	240,990	37,255	3,089,598
Time Interval	5min	1min	10min	1h
Time Domain	6 days	24 hours	>6 days	>87 days
Max len. traj.	39	1283	829	2085
Median len. traj.	9	36	91	165

3.7 Experiments

In this section, we will evaluate the privacy level and data utility of the proposed method on four real-life trajectory datasets. The real-life datasets include PeopleFlow(PFlow)[‡], Geolife^{**}, T-Drive^{††}, WorldCup98(WC98)^{‡‡}, summarized in Table 3.2. All of them are available online. The first three datasets are people moving trajectory data by diverse manner, such as walking, driving, by train, or by plane. The forth one is users’ click streams (i.e., from one webpage to another webpage), and it can be seen as people’s trajectory in the cyberspace (i.e., webpages are equivalent to locations). Since our basic setting is that each user is located at most one location at a single timestamp, we randomly remove data points who violate this assumption and sample data points according to intervals shown in Table 3.2.

As we introduced in Section 3.3, FAST[¶] is a sampling and filtering framework for privacy preserving time series data publishing. The authors design two kinds of algorithms based on FAST, that is FAST with adaptively sampling and with fixed sample rate repetitively. Since the former one needs to know the amount of total timestamps T in advance, it leads to inapplicability of infinite trajectory streams. Therefore, we use FAST with fixed sampling rate as our utility competitor, denoted by FAST_{fixed}. We configure it according to the original chapter [29].

[‡] <http://pflow.csis.u-tokyo.ac.jp/>

^{**} <http://research.microsoft.com/projects/geolife>

^{††} <http://research.microsoft.com/projects/tdrive>

^{‡‡} <http://ita.ee.lbl.gov/html/contrib/WorldCup.html>

[¶] <http://www.mathcs.emory.edu/aims/FAST/index.html>

ℓ -Trajectory privacy is featured as a personalized privacy model, which users can specify the length of the protected trajectories. However, since our competitors such as *w-event privacy* and $\text{FAST}_{\text{fixed}}$ inherently cannot provide protection with customized $\langle \ell_1, \dots, \ell_{|U|} \rangle$, we will set all ℓ_u as the same value in the following experiments. In addition, since the outputs of the algorithms are including randomness, each algorithm runs 50 times then outputs the average results.

3.7.1 Privacy Evaluation

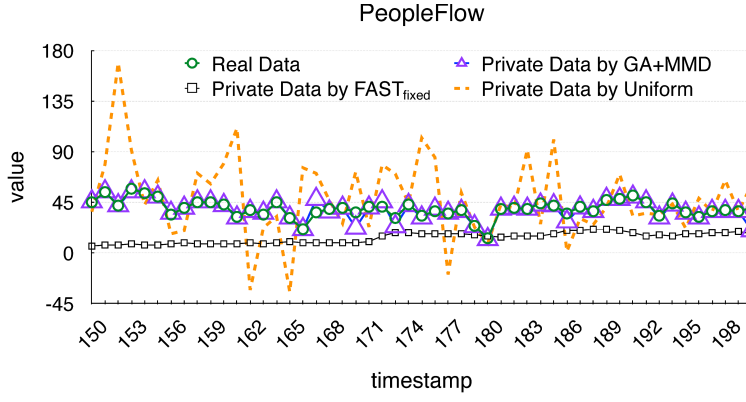
In this section, we will quantitatively evaluate the privacy risk of *w-event privacy* for achieving our privacy goal. As explained in Section 4.2, *w-event privacy* [30] cannot provide *personalized* and *uniform* protection. The former is true because essentially *w-event privacy* is an event-level, not a user-level privacy model. The latter is because there is no upper bound in the number of contiguous timestamps a ℓ_u -trajectory spans, which has been explained in Section 4.2. However, there is still a little chance for *w-event privacy* to achieve our privacy goal (i.e., satisfy Theorem 6). Therefore, we quantitatively measure the risk of *w-event privacy* for achieving our privacy goal. The metric is the percentage of *risky ℓ_u -trajectories* whose privacy budgets are greater than ϵ (violation of Theorem 4) among all users' ℓ_u -trajectories.

We run a private data publishing algorithm BD [30] of *w-event privacy*, and calculate the ratio of risky ℓ_u -trajectories. We set w as 10 to 90 with intervals of 20, and $\ell_u = w$. From the result shown in Table 3.3 (e.g., the first cell of 9.1% is the ratio of risky 10-trajectories under the protection of 10-event privacy), we can conclude that ℓ_u -trajectories cannot be uniformly protected by *w-event privacy* with a given ℓ_u and w , unless $w \gg \ell_u$ or dense trajectory (data points appear at each cells in Fig.3.2). For $w \gg \ell_u$, *w-event privacy* has a higher probability to protect every ℓ_u -trajectory. Two examples are the cases of $w = 70$ and 90 in dataset PFlow, and the actual protected length ℓ_u is 39 because the longest trajectory in PFlow is 39 as shown in Table 3.2. However, we do not know the appropriate w for different datasets to protect every ℓ_u -trajectory. For dense trajectory, *w-event privacy* can protect every ($\ell_u = w$)-trajectory; however, the users cannot specify different ℓ_u .

On the other hand, the ratios of risky trajectories of UNIFORM, GA+Adj and GA+MMD are all 0%. In other words, they exactly satisfy ℓ -trajectory privacy.

Table 3.3: Ratios of risky w -trajectories under w -event privacy.

$w =$	10	30	50	70	90
PFLow	9.1%	0.16%	0.004%	0%	0%
GeoLife	0.44%	9.3%	15.7%	19.5%	19.8%
T-Drive	1.16%	2.74%	3.47%	7.35%	11.66%
WC98	1.12%	1.99%	2.95%	3.96%	1.97%


Figure 3.5: Comparision of Private and Real Data.

3.7.2 Utility Evaluation

Fig.3.5 shows the real and noisy statistics which were published by different algorithms under the same privacy level of $\ell_u = 20$ and $\epsilon = 1$. We can find that the noisy data published by GA+MMD is more similar to the real data than others.

MAE, MRE and MSE

We quantitatively evaluate the data utility by the following metrics, Mean of Absolute Error (MAE), Mean of Relative Error (MRE), Mean of Square Error (MSE) and KL-divergence [33]. MAE has been given as Equation (2.3). In MRE, a user-specified constant δ is used to mitigate the effect of excessively small query results. This definition follows from previous work [50, 51]. We set δ as 1. MSE is more sensitive to larger errors. The results are shown in Fig.3.6, 3.7, 3.8 by varying ℓ_u (set $\epsilon = 1$) and varying ϵ (set $\ell_u = 20$) respectively. Longer ℓ_u or

3. ℓ -trajectory Privacy

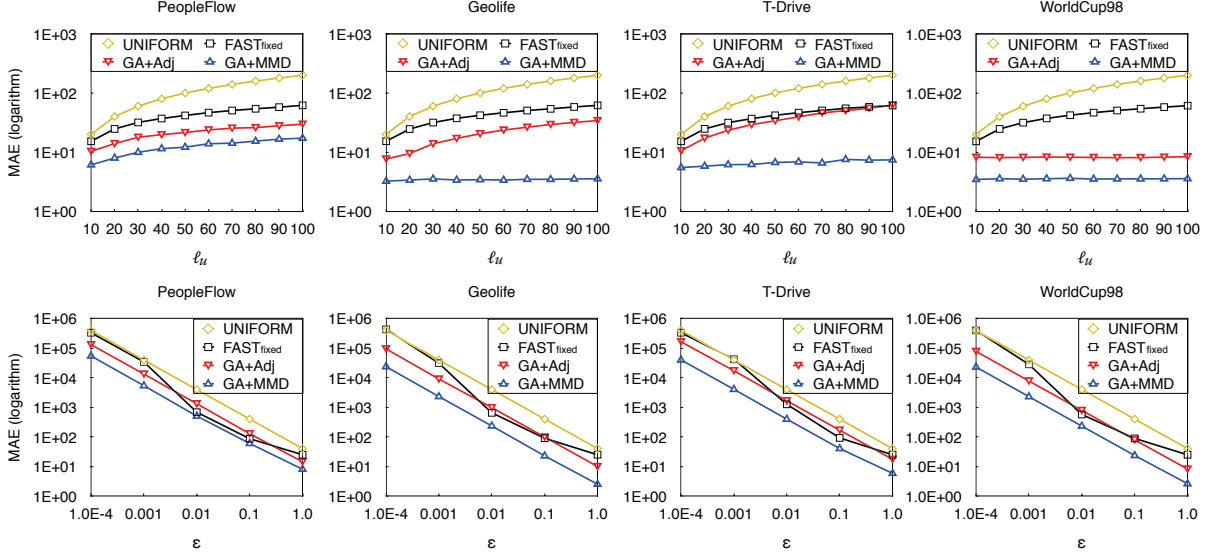


Figure 3.6: MAE on four datasets by varying ℓ_u (10 to 100) and varying ϵ (0.0001 to 1).

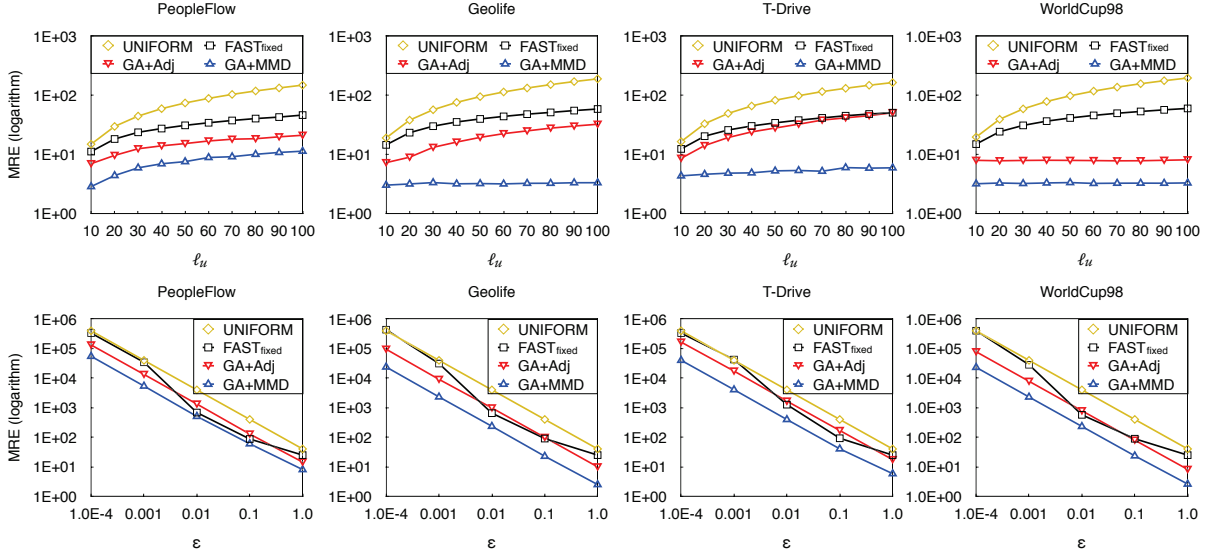


Figure 3.7: MRE on four datasets by varying ℓ_u (10 to 100) and varying ϵ (0.0001 to 1).

smaller ϵ higher the privacy level.

UNIFORM's MAE becomes larger almost linearly with the increasing of ℓ_u , and exponentially with decreasing of ϵ , because ℓ_u and ϵ are the only factors affecting UNIFORM's performance. For FAST_{fixed}, since the *adaptive* prediction and correction features are unavailable for achieving ℓ -trajectory privacy, it is not

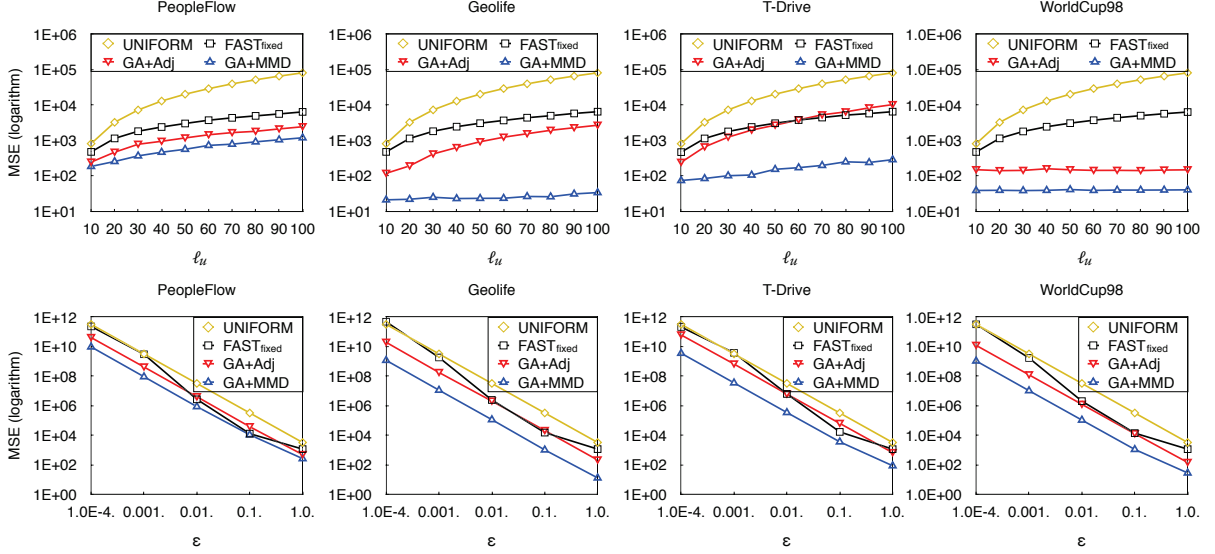


Figure 3.8: MSE on four datasets by varying ℓ_u (10 to 100) and varying ϵ (0.0001 to 1).

stable with different ϵ and ℓ_u . GA+Adj and GA+MMD are superior to others because of their data-dependent dynamic budget allocation and approximation. However, GA+Adj are susceptible to the similarity between each pair of the adjacent values on the timeline. This is one of the reasons that GA+Adj performs better in dataset WC98 which has the largest number of $|\mathbf{locs}|$ (the similarity is calculated with a factor of $1/|\mathbf{locs}|$; refer to Section 3.6.4). While, GA+MMD is more adaptive because it is capable of searching the most similar value all through the past timeline, not only the adjacent one, to republish it as the current value. Comparing the performances of GA+MMD among the four datasets, the one in PFlow is worst. One potential reason is that the dynamic budget allocation (GA) has a high probability of encountering with the worst case (refer to Section 3.6.3) because the trajectories in PFlow are quite “crowded” (more users but less locations). Thanks to approximation strategy MMD, that worst case (too small allocated budgets) can be eliminated by republishing the most similar data without spending any budget. Therefore, GA+MMD is the best among the four algorithms.

KL-divergence

Fig.3.9 shows the result of KL-divergence between noisy data and real data by varying ℓ_u . The lower the better. It shows that the noisy data published by

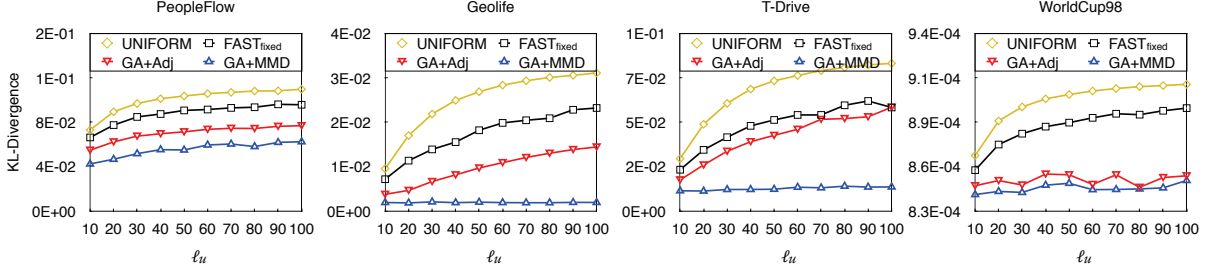


Figure 3.9: KL-divergence on four datasets by varying ℓ_u (10 to 100).

GA+MMD obtains lowest KL-divergence, which means its distribution is more close to original data. This is also a quantitative evidence of the visual comparison of Fig.3.5.

3.7.3 Runtime

It is easy to know the time complexity of UNIFORM_t is $O(|\mathbf{loc}|)$. For algorithm GA, since the complexity of Lines 1, 3, 4 and 5 are $O(1)$, and the complexity of Line 2 is $O(|us_i|)$ (to find the maximum value), the overall complexity is $O(|us_i|)$. For algorithm Adj, the time complexity is $O(|\mathbf{loc}|)$ because of computing MAE. For algorithm MMD, the most time-consuming operations are Lines 2 and 3. The calculation of the scores in Line 2 needs to access $|\mathbf{loc}|$ amount of data for total t timestamps, and the calculation in Line 3 involves sorting the scores. Therefore, the overall complexity of MMD are $O(t * |\mathbf{loc}| + t * \log t)$. We notice that if t is large, the complexity of MMD is high because of comparing r_t with n_i where $i \in [1, t - 1]$. One relaxation of MMD is that, by introducing a heuristic threshold t_{period} , we only need to compare r_t with n_i where $i \in [t - 1 - t_{\text{period}}, t - 1]$. t_{period} means the possible repetition interval which can be specified by experts. For example, if the expert knows that the data has a repeating pattern every week, then we can set t_{period} as a week. By this relaxation, t becomes a constant value t_{period} . Hence, the complexity of relaxed MMD reduces to $O(|\mathbf{loc}|)$.

Therefore, the time complexity of GA+Adj and GA+MMD are $O(|us_i| + |\mathbf{loc}|)$ and $O(|us_i| + t * |\mathbf{loc}| + t * \log t)$ respectively. The relaxed version of GA+MMD has the same complexity as GA+Adj. According to [29], real-time algorithm FAST_{fixed} has complexity $O(|\mathbf{loc}|)$.

Besides FAST_{fixed}, algorithms UNIFORM, GA+Adj and GA+MMD are capable

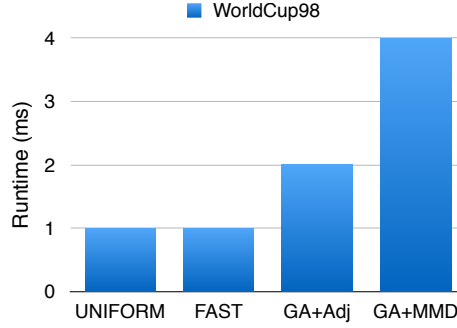


Figure 3.10: Runtime of each algorithm on WorldCup98.

of real-time processing and publishing because the algorithms are only need to be input data of the current timestamps t and the previous ones. In the above experiments, we simulated the generating process of real-time data. We make sure that the algorithms are input the data of each timestamp one by one, and output the corresponding private data one by one. Fig.3.10 shows the overall runtime of each algorithm on the largest dataset WorldCup98.

CHAPTER 4

TEMPORAL DIFFERENTIAL PRIVACY

4.1 Motivation

Differential Privacy (DP) has received increased attention as a rigorous privacy framework. Existing studies employ traditional DP mechanisms (e.g., the Laplace mechanism) as primitives, which assume that the data are independent, or that adversaries do not have knowledge of the data correlations. However, continuously generated data in the real world tend to be temporally correlated, and such correlations can be acquired by adversaries. In this chapter, we investigate the potential privacy loss of a traditional DP mechanism under temporal correlations in the context of continuous data release. First, we model the temporal correlations using Markov model and analyze the privacy leakage of a DP mechanism when adversaries have knowledge of such temporal correlations. Our analysis reveals that the privacy loss of a DP mechanism may *accumulate and increase over time*. We call it *temporal privacy leakage*. Second, to measure such privacy loss, we design an efficient algorithm for calculating it in polynomial time. Although the temporal privacy leakage may increase over time, we also show that its supremum may exist in some cases. Third, to bound the privacy loss, we propose

©2017 IEEE. Reprinted, with permission, from Yang Cao, Quantifying Differential Privacy Under Temporal Correlations, In Proceeding of the 33rd International Conference on Data Engineering (ICDE).

mechanisms that convert any existing DP mechanism into one against temporal privacy leakage. Experiments with synthetic data confirm that our approach is efficient and effective.

4.2 Introduction

With the development of wearable and mobile devices, vast amount of temporal data generated by individuals are being collected, such as trajectories and web page click streams. The continuous publication of statistics from these temporal data has the potential for significant social benefits such as disease surveillance [1], real-time traffic monitoring [2] and web mining [52]. However, privacy concerns hinder the wider use of these data. To this end, *differential privacy under continual observation* [23] [24] [25] [26] [27] [28] [29] [30] [31] has received increased attention because it provides a rigorous privacy guarantee. Intuitively, differential privacy (DP) [20] ensures that the modification of any single user’s data in the database has a “slight” (bounded in ϵ) impact on the change in outputs. The parameter ϵ is defined to be a positive real number to control the privacy level. Larger values of ϵ result in larger privacy leakage.

However, data independence, which is a potential assumption of the traditional DP mechanisms, was neglected by most previous works in differentially private continuous aggregate release. Recent studies [46] [53] [54] point out that traditional DP may not guarantee the expected privacy on correlated data. The following example shows that the temporal correlations may degrade the expected privacy guarantee of DP.

Example 1. *Consider the scenario of continuously aggregate release illustrated in Figure 4.1. A trusted server collects users’ locations at each time point in Figure 4.1(a) and continuously publishes aggregate (i.e., the counts of people at each location) in Figure 4.1(c) with differential privacy. Our goal is to achieve ϵ -DP at each time point t (event-level ϵ -DP [27] [28]) where $t \in [1, T]$. Suppose that each user appears at only one location at each time point. According to the Laplace mechanism [21], adding $\text{Lap}(1/\epsilon)$ noise* to perturb each count in Figure 4.1(c) can achieve ϵ -DP at each time point. However, the expected privacy guarantee may decay due to temporal correlations as follows. Using auxiliary information, such*

* $\text{Lap}(b)$ denotes a Laplace distribution with variance $2b^2$.

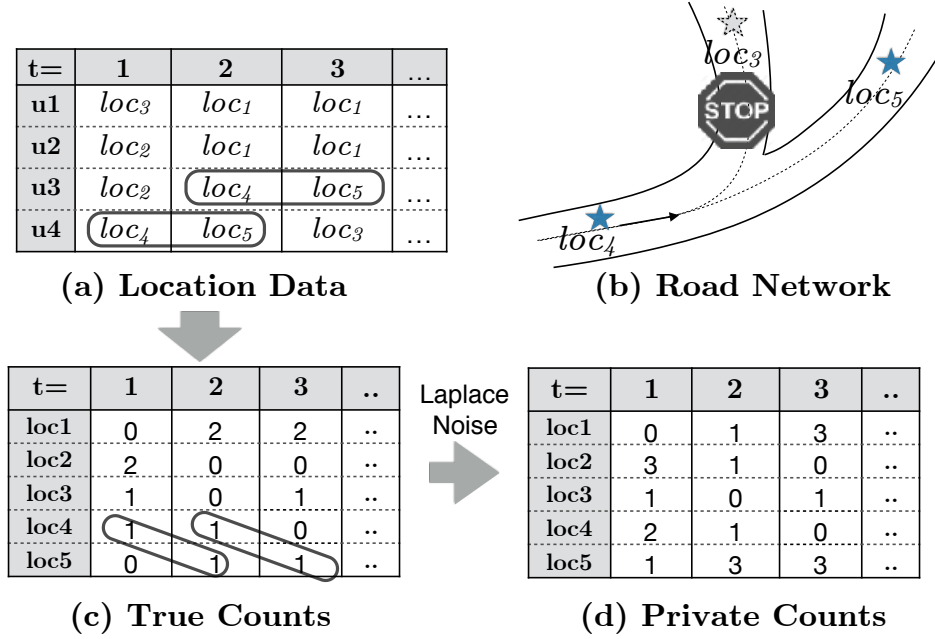


Figure 4.1: Differentially Private Continuous Aggregate Release under Temporal Correlations.

as road network constraints, an attacker may know users' mobility patterns, such as "always arriving at loc_5 after visiting loc_4 " (shown in Figure 4.1(b)), leading to the patterns illustrated in solid lines of Figure 4.1(c). The temporal correlation due to this road network can be represented as $\Pr(l^t = loc_5 | l^{t-1} = loc_4) = 1$ where l^t is the location of a user at time t . That is, given the previous counts of loc_4 , an attacker can derive the current count of loc_5 . Consequently, because an adversary can perform inference due to such correlations between the two consecutive time points (i.e., as if the same count is released two times), adding $Lap(1/\epsilon)$ noise to each count guarantees 2ϵ -DP at the time point. Furthermore, let us consider an extreme case of temporal correlation (e.g., a terrible traffic congestion) $\Pr(l^t = loc_4 | l^{t-1} = loc_4) = \Pr(l^t = loc_5 | l^{t-1} = loc_5) = 1$ (i.e., the counts of loc_4 and loc_5 will not change over time). Then, adding $Lap(1/\epsilon)$ noise to each count guarantees $T\epsilon$ -DP at time point T . *tor*

It is reasonable to consider that adversaries may obtain the temporal correlations, which commonly exist in our real life and are easily acquired from public information or historical data. In addition to road networks [55], there are countless factors that may cause temporal correlations such as the common patterns

characterizing human activities [56] and weather conditions [57].

Few studies in the literature investigated such potential privacy loss of event-level ϵ -DP under temporal correlations as shown in Example 1. A direct method (without finely utilizing the *probability* of correlation) involves amplifying the perturbation in terms of *group differential privacy* [58] [21], i.e., protecting the correlated data as a group. In Example 1, for temporal correlation $\Pr(l^t = loc_5 | l^{t-1} = loc_4) = 1$, we can protect the counts of loc_4 at time $t - 1$ and loc_5 at time t in a group (the sensitivity becomes 2) by increasing the scale of the perturbation to $Lap(2/\epsilon)$ at each time point; for temporal correlation $\Pr(l^t = loc_i | l^{t-1} = loc_i) = 1$, in order to guarantee ϵ -DP at time T , we need to add $Lap(T/\epsilon)$ noise at each time point because the privacy leakage accumulates over time. However, this technique is not suitable for *probabilistic* correlations to finely prevent privacy leakage and may over-perturb the data as a result. For example, regardless of whether $\Pr(l^t = loc_i | l^{t-1} = loc_i)$ is 1 or 0.1, it always protects the correlated data in a bundle.

Although a few studies investigated the issue of differential privacy under *probabilistic* correlations, they are not applicable for *continuous data release* because of the different problem settings. The following two works focused on one-shot data release and different types of correlations. Yang et al. [59] proposed Bayesian differential privacy (BDP), which measures the privacy leakage under probabilistic correlations between tuples, modeled by a Gaussian Markov Random Field without taking time factor into account. Liu et al. [60] proposed *dependent differential privacy* by introducing two parameters of *dependence size* and *probabilistic dependence relationship* between tuples. However, it is not clear whether we can specify them for temporally correlated data. Another line of work [61] [62] [63] has investigated adversaries with knowledge of temporal correlations. They focused on designing new mechanisms for protecting a single user’s location privacy extending DP, whereas we attempt to quantify the potential privacy loss of a traditional DP mechanism in the context of continuous aggregate release.

We call the adversary considered in traditional DP with additional knowledge of probabilistic temporal correlations as *adversary_T*. Rigorously quantifying and bounding the privacy leakage against *adversary_T* remains a challenge. Therefore, we attempt to solve the following problems in this chapter:

- How do we formalize *adversary_T* and define the privacy loss against *adversary_T*? (Section 4.4)

- How do we calculate the privacy loss against adversary $\mathcal{T}$? (Section 4.5)
- How do we bound the privacy loss against adversary $\mathcal{T}$? (Section 4.6)

4.2.1 Contributions

In this work, for the first time, we quantify and bound the privacy leakage of a DP mechanism due to temporal correlations. Our contributions are summarized as follows.

First, we rigorously define *adversary $\mathcal{T}$* with temporal correlations that are described by the commonly used Markov model. The temporal correlations include *backward* and *forward* correlations, i.e., $\Pr(l_i^{t-1}|l_i^t)$ and $\Pr(l_i^t|l_i^{t-1})$ where l_i^t denotes the value (e.g., location) of user i at time t . We then define *Temporal Privacy Leakage (TPL)* as the privacy loss of a DP mechanism at time t against adversary $\mathcal{T}$. TPL includes two parts: *Backward Privacy Leakage (BPL)* and *Forward Privacy leakage (FPL)* due to the existence of backward and forward temporal correlations. Our analysis shows that BPL may accumulates from previous privacy leakage and FPL increases with future release. Intuitively, BPL at time t is affected by previously released data and FPL at time t will be affected by future releases. We define *α -differential privacy under temporal correlation*, denoted as α -DP $\mathcal{T}$, to formalize the privacy guarantee of a DP mechanism against adversary $\mathcal{T}$, i.e., the temporal privacy leakage should be bounded in α . We prove a new form of sequential composition theorem for α -DP $\mathcal{T}$ (different from the traditional sequential composition [47] for ϵ -DP).

Second, we efficiently calculate the temporal privacy leakage under given backward and forward temporal correlations. We translate the calculation of temporal privacy leakage into finding an optimal solution of a *linear-fractional programming* problem. This type of problem can be solved using a simplex algorithm in exponential time. By exploiting the constraints, we propose a polynomial-time algorithm to finely quantify the temporal privacy leakage.

Third, we design private data release algorithms that can be used to convert a traditional DP mechanism into one satisfying α -DP $\mathcal{T}$. A challenge is that the temporal privacy leakage may increase over time so that α -DP $\mathcal{T}$ is hard to achieve when the length of release time T is unknown. In our first solution, we prove that the supremum of temporal privacy leakage may exist in some cases, and in these cases, we allocate appropriate privacy budgets to make sure the increased

temporal privacy leakage will never be greater than α , no matter how long the T is. However, when T is too short so that the accumulation of temporal privacy leakage have not resulted in a significant increase, we may over-perturb the data. The second solution is to exactly achieve α -DP $_{\mathcal{T}}$ at each time point by finely calculating the temporal privacy leakage.

Finally, Experiments with synthetic data confirm the efficiency and effective of our privacy leakage quantification algorithm. We also demonstrate the impact of different degree of temporal correlations on privacy leakage.

4.3 Preliminaries

4.3.1 Privacy Leakage.

Let us first discuss the adversaries tolerated by differential privacy, and then formalize privacy leakage w.r.t. such adversaries. Differential privacy is able to protect against the attackers who even have knowledge of all users' data in the database except the one of the targeted victim [64]. Let $i \in \mathbf{U}$ be a user in the database D . Let A_i be an adversary who targets user i and has knowledge of $D_{\mathcal{K}} = D - \{l_i\}$ where $l_i \in [loc_1, \dots, loc_n]$ denotes the data of user i . The adversary A_i observes the private output $\mathbf{r}$ and attempts to guess whether the possible value of user i is loc_j or loc_k where $loc_j, loc_k \in [loc_1, \dots, loc_n]$. We define the privacy leakage of a DP mechanism as follows.

Definition 8 (Privacy Leakage of a DP mechanism against A_i). *Let $\mathbf{U}$ be a set of users in the database. Let A_i be an adversary who targets user i and knows all the tuples in the database except the one of user i . The privacy leakage of a differentially private mechanism $\mathcal{M}$ against one A_i and all $A_i, i \in \mathbf{U}$ are defined, respectively, as follows in which l_i and l'_i are two different possible values of user i 's data.*

$$PL_0(A_i, \mathcal{M}) \stackrel{\text{def}}{=} \sup_{\mathbf{r}, l_i, l'_i} \log \frac{\Pr(\mathbf{r}|l_i, D_{\mathcal{K}})}{\Pr(\mathbf{r}|l'_i, D_{\mathcal{K}})}$$

$$PL_0(\mathcal{M}) \stackrel{\text{def}}{=} \max_{\forall A_i, i \in \mathbf{U}} PL_0(A_i, \mathcal{M}) = \sup_{\mathbf{r}, D, D'} \log \frac{\Pr(\mathbf{r}|D)}{\Pr(\mathbf{r}|D')}$$

In other words, the privacy budget of a DP mechanism can be considered as a metric of *privacy leakage*. The larger ϵ , the larger the privacy leakage. Hence, we

can say that $\mathcal{M}$ satisfies ϵ -DP if $PL_0(\mathcal{M}) \leq \epsilon$. We note that a ϵ' -DP mechanism automatically satisfies ϵ -DP if $\epsilon' < \epsilon$. For convenience, in the following parts of this chapter, when we say that $\mathcal{M}$ satisfies ϵ -DP we mean that $PL_0(\mathcal{M}) = \epsilon$.

4.3.2 Problem Setting

We attempt to quantify the potential privacy loss a DP mechanism under temporal correlations in the context of *continuous data release* (e.g., releasing private counts at each time as shown in Figure 4.1). Users in the database, denoted by U , are generating data continuously. Let $\mathbf{loc} = \{loc_1, \dots, loc_n\}$ be all possible values of user's data. We denote the value of user i at time t by l_i^t . A trusted server collects the data of each user into the database $D^t = \{l_1^t, \dots, l_{|U|}^t\}$ at each time t (e.g., the columns in Figure 4.1(a)). A DP mechanism $\mathcal{M}^t$ releases differentially private output $\mathbf{r}^t$ independently at each time t . Our goal is to quantify and bound the potential privacy loss of $\mathcal{M}^t$ against adversaries with knowledge of temporal correlations. We summarize the notations used in this chapter in Table 4.1. We note that while we use location data in Example 1, the problem setting is general for temporally correlated data.

Our problem setting is identical to *differential privacy under continual observation* in the literature [23] [24] [25] [26] [27] [28] [29] [30] [31]. In contrast to “one-shot” data release over a static database, the adversaries can observe multiple differentially private outputs, i.e., $\mathbf{r}^1, \dots, \mathbf{r}^t$. There are typically two different privacy goals in the context of continuous data release: *event-level* and *user-level* [27] [28]. The former protects each user's single data point at time t (i.e., the neighboring databases are D^t and $D^{t'}$), whereas the latter protects the presence of a user with all her data on the timeline (i.e., the neighboring databases are $\{D^1, \dots, D^t\}$ and $\{D^{1'}, \dots, D^{t'}\}$). In this work, we mainly study the privacy leakage at a single time point (event-level) under temporal correlations, and we also extend the discussion to user-level privacy by studying the composability of the privacy leakage.

4.4 Analyzing Privacy Leakage

In the following, we first formalize adversary with temporal correlations in Section 4.4.1. We then define and analyze *temporal privacy leakage* in Section 4.4.2.

Table 4.1: Summary of Notations.

$\mathbf{U}$	The set of users in the database
i	The i -th user where $i \in [1, \mathbf{U}]$
$\mathbf{loc}$	Value domain $\{loc_1, \dots, loc_n\}$ of all user's data
$l_i^t, l_i^{t'}$	The data of user i at time t , $l_i^t \in \mathbf{loc}$, $l_i^t \neq l_i^{t'}$
D^t	The database at time t , $D^t = \{l_1^t, \dots, l_n^t\}$
$\mathcal{M}^t$	Differentially private mechanism over D^t
$\mathbf{r}^t$	Differentially private output at time t
A_i	The adversary who targets user i , considered in traditional DP
$A_i^{\mathcal{T}}$	Adversary A_i with additional knowledge of temporal correlations
P_i^B	Transition matrix that represents $\Pr(l_i^{t-1} l_i^t)$, i.e., backward temporal correlation, known to $A_i^{\mathcal{T}}$
P_i^F	Transition matrix that represents $\Pr(l_i^t l_i^{t-1})$, i.e., forward temporal correlation, known to $A_i^{\mathcal{T}}$
$D_{\mathcal{K}}^t$	The subset of database $D^t - \{l_i^t\}$, known to $A_i^{\mathcal{T}}$

We provide a new privacy notion of α - $DP_{\mathcal{T}}$ against temporal privacy leakage and prove its composability in Section 4.4.3. Finally, we make a few important observations in Section 5.4.3.

4.4.1 Adversary with Knowledge of Temporal Correlations

Markov Chain for Temporal Correlations. The Markov chain (MC) is extensively used in modeling user mobility profiles [56] [65] [61] [55]. For a time-

homogeneous first-order MC, a user's current value only depends on the previous one. The parameter of the MC is the *transition matrix*, which describes the probabilities for transition between values. The sum of the probabilities in each row of the transition matrix is 1. A concrete example of transition matrix and time-reversed one for location data is shown in Figure 4.2. As shown in Figure 4.2(a), if user i is at loc_1 now (time t); then, the probability of coming from loc_3 (time $t - 1$) is 0.7, namely, $\Pr(l_i^{t-1} = loc_3 | l_i^t = loc_1) = 0.7$. As shown in Figure 4.2(b), if user i was at loc_3 at the previous time $t - 1$, then the probability of being at loc_1 now (time t) is 0.6; namely, $\Pr(l_i^t = loc_1 | l_i^{t-1} = loc_3) = 0.6$. We call the transition matrices in Figure 4.2(a) and (b) as backward temporal correlation and forward temporal correlation, respectively.

Definition 9 (Temporal Correlations). *The backward and forward temporal correlations between user i 's data l_i^{t-1} and l_i^t are described by transition matrices $P_i^B, P_i^F \in \mathbb{R}^{n \times n}$, representing $\Pr(l_i^{t-1} | l_i^t)$ and $\Pr(l_i^t | l_i^{t-1})$, respectively.*

It is reasonable to consider that the backward and/or forward temporal correlations could be acquired by adversaries. For example, the adversaries can learn them from user's historical trajectories (or the reversed trajectories) by well studied method such as Maximum Likelihood Estimation, or using more sophisticated Markov model [56], or utilizing road network constraints [55]. Also, if the initial distribution of l_i^1 is known (i.e., $\Pr(l_i^1)$), the backward temporal correlation (i.e., $\Pr(l_i^{t-1} | l_i^t)$) can be derived from the forward temporal correlation (i.e., $\Pr(l_i^t | l_i^{t-1})$) by the following Bayesian inference.

$$\Pr(l_i^{t-1} | l_i^t) = \frac{\Pr(l_i^t | l_i^{t-1}) * \Pr(l_i^{t-1})}{\sum_{l_i^{t-1}} \Pr(l_i^t | l_i^{t-1}) * \Pr(l_i^{t-1})}$$

Since estimating temporal correlations from data is beyond the scope of this work, we assume the adversaries' prior knowledge about temporal correlations is given in our framework.

We now define an “updated version” of A_i (in Definition 8) with knowledge of temporal correlations.

Definition 10 ($\text{Adversary}_{\mathcal{T}}$). *$\text{Adversary}_{\mathcal{T}}$ is a class of adversaries who have knowledge of (1) all other users' data $D_{\mathcal{K}}^t$ at each time t except the one of the targeted victim, i.e., $D_{\mathcal{K}}^t = D^t - \{l_i^t\}$, and (2) backward and/or forward temporal*

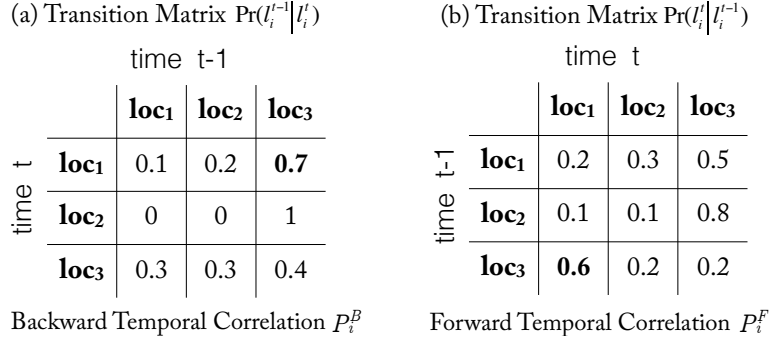
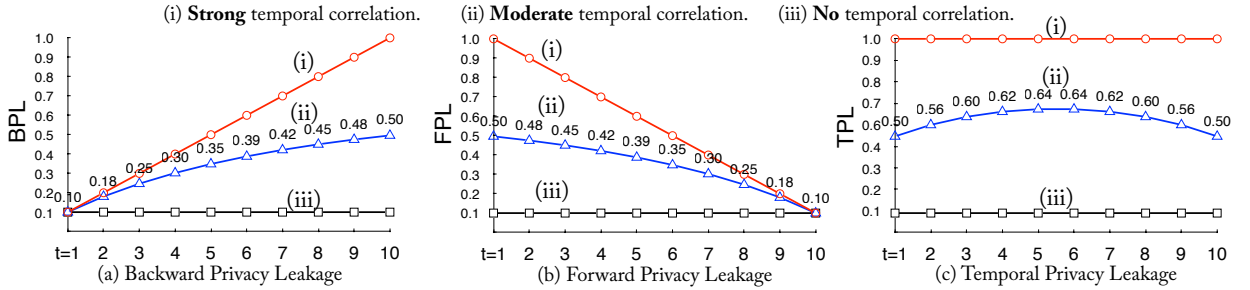


Figure 4.2: Examples of Temporal Correlations.


 Figure 4.3: Example of Temporal Privacy Leakage of $Lap(1/0.1)$ at each time point.

correlations represented as transition matrices P_i^B and P_i^F . We denote $\text{Adversary}_{\mathcal{T}}$ who targets user i by $A_i^{\mathcal{T}}(P_i^B, P_i^F)$.

There are three types of $\text{adversary}_{\mathcal{T}}$: (i) $A_i^{\mathcal{T}}(P_i^B, \emptyset)$, (ii) $A_i^{\mathcal{T}}(\emptyset, P_i^F)$, (iii) $A_i^{\mathcal{T}}(P_i^B, P_i^F)$, where $\emptyset$ denotes that the corresponding correlations are not known to the adversaries[†]. For simplicity, we denote types (i) and (ii) as $A_i^{\mathcal{T}}(P_i^B)$ and $A_i^{\mathcal{T}}(P_i^F)$, respectively. We note that $A_i^{\mathcal{T}}(\emptyset, \emptyset)$ is the same as the traditional DP adversary A_i without any knowledge of temporal correlations.

We now show what information $A_i^{\mathcal{T}}$ can derive from the temporal correlations.

Lemma 5. *The adversary $A_i^{\mathcal{T}}$ who has knowledge of P_i^B can derive $\Pr(D^{t-1}|D^t) = \Pr(l_i^{t-1}|l_i^t)$.*

Lemma 6. *The adversary $A_i^{\mathcal{T}}$ who has knowledge of P_i^F can derive $\Pr(D^t|D^{t-1}) = \Pr(l_i^t|l_i^{t-1})$.*

We omit the proofs of the lemmas due to space limitations.

[†]The adversaries of types (i) and (ii) will not “guess” the missing correlations; otherwise, they fall under type (iii).

4.4.2 Temporal Privacy Leakage

We now define the privacy leakage w.r.t. adversary $_{\mathcal{T}}$. For the convenience of analysis, let us assume the length of release time[‡] is T . The adversary $A_i^{\mathcal{T}}$ observes the differentially private outputs $\mathbf{r}^1, \dots, \mathbf{r}^t, \dots, \mathbf{r}^T$ and attempts to infer the value of user i 's data at time t , namely l_i^t . Similar to Definition 8, we define the privacy leakage in terms of differential privacy but in the context of *differential private under continual observation*, which is described in Section 5.2.2.

Definition 11 (Temporal Privacy Leakage, TPL). *Let $D^{t'}$ be a neighboring database of D^t . Let $D_{\mathcal{K}}^t$ be the tuple knowledge of $A_i^{\mathcal{T}}$. We have $D^{t'} = D_{\mathcal{K}}^t \cup \{l_i^t\}$ and $D^{t'} = D_{\mathcal{K}}^t \cup \{l_i^{t'}\}$ where l_i^t and $l_i^{t'}$ are two different values of user i 's data at time t . Temporal Privacy Leakage (TPL) of $\mathcal{M}^t$ w.r.t. a single $A_i^{\mathcal{T}}$ and all $A_i^{\mathcal{T}}, i \in \mathcal{U}$ are defined, respectively, as follows.*

$$TPL(A_i^{\mathcal{T}}, \mathcal{M}^t) \stackrel{\text{def}}{=} \sup_{l_i^t, l_i^{t'}, \mathbf{r}^1, \dots, \mathbf{r}^T} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | l_i^t, D_{\mathcal{K}}^t)}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | l_i^{t'}, D_{\mathcal{K}}^t)}. \quad (4.1)$$

$$TPL(\mathcal{M}^t) \stackrel{\text{def}}{=} \max_{\forall A_i^{\mathcal{T}}, i \in \mathcal{U}} TPL(A_i^{\mathcal{T}}, \mathcal{M}^t) \quad (4.2)$$

$$= \sup_{D^t, D^{t'}, \mathbf{r}^1, \dots, \mathbf{r}^T} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | D^t)}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | D^{t'})}. \quad (4.3)$$

We first analyze $TPL(A_i^{\mathcal{T}}, \mathcal{M}^t)$ (i.e., Equation (4.1)) because it is key to solve Equation (4.2) or (4.3). We can rewrite $TPL(A_i^{\mathcal{T}}, \mathcal{M}^t)$ as follows because $\mathbf{r}^1, \dots, \mathbf{r}^T$ are published independently by differentially private mechanism $\mathcal{M}^1, \dots, \mathcal{M}^T$.

$$\begin{aligned} \text{Eqn. (4.1)} &= \sup_{l_i^t, l_i^{t'}, \mathbf{r}^1, \dots, \mathbf{r}^T} \log \frac{\Pr(\mathbf{r}^1 | l_i^t, D_{\mathcal{K}}^t)}{\Pr(\mathbf{r}^1 | l_i^{t'}, D_{\mathcal{K}}^t)} * \dots * \frac{\Pr(\mathbf{r}^T | l_i^t, D_{\mathcal{K}}^t)}{\Pr(\mathbf{r}^T | l_i^{t'}, D_{\mathcal{K}}^t)} \\ &= \underbrace{\sup_{\substack{\mathbf{r}^1, \dots, \mathbf{r}^t, \\ l_i^t, l_i^{t'}}} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | l_i^t, D_{\mathcal{K}}^t)}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | l_i^{t'}, D_{\mathcal{K}}^t)}}_{\text{backward privacy leakage}} + \underbrace{\sup_{\substack{\mathbf{r}^t, \dots, \mathbf{r}^T, \\ l_i^t, l_i^{t'}}} \log \frac{\Pr(\mathbf{r}^t, \dots, \mathbf{r}^T | l_i^t, D_{\mathcal{K}}^t)}{\Pr(\mathbf{r}^t, \dots, \mathbf{r}^T | l_i^{t'}, D_{\mathcal{K}}^t)}}_{\text{forward privacy leakage}} \\ &\quad - \underbrace{\sup_{\mathbf{r}^t, l_i^t, l_i^{t'}} \log \frac{\Pr(\mathbf{r}^t | l_i^t, D_{\mathcal{K}}^t)}{\Pr(\mathbf{r}^t | l_i^{t'}, D_{\mathcal{K}}^t)}}_{PL_0(A_i^{\mathcal{T}}, \mathcal{M}^t)} \end{aligned} \quad (4.4)$$

[‡]In this chapter, we do not need to know the length of release time in advance.

It is clear that $PL_0(A_i^\mathcal{T}, \mathcal{M}^t) = PL_0(A_i, \mathcal{M}^t)$ because PL_0 indicates the privacy leakage w.r.t. one output $\mathbf{r}$ (refer to Definition 8). As annotated in the above equation, we define backward and forward privacy leakage as follows.

Definition 12 (Backward Privacy Leakage, BPL). *The privacy leakage of $\mathcal{M}^t$ caused by $\mathbf{r}^1, \dots, \mathbf{r}^t$ w.r.t. $A_i^\mathcal{T}$ is called backward privacy leakage, defined as follows.*

$$BPL(A_i^\mathcal{T}, \mathcal{M}^t) \stackrel{\text{def}}{=} \sup_{l_i^t, l_i^{t'}, \mathbf{r}^1, \dots, \mathbf{r}^t} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | l_i^t, D_K^t)}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | l_i^{t'}, D_K^t)}. \quad (4.5)$$

$$BPL(\mathcal{M}^t) \stackrel{\text{def}}{=} \max_{\forall A_i^\mathcal{T}, i \in \mathcal{U}} BPL(A_i^\mathcal{T}, \mathcal{M}^t). \quad (4.6)$$

Definition 13 (Forward Privacy Leakage, FPL). *The privacy leakage of $\mathcal{M}^t$ caused by $\mathbf{r}^t, \dots, \mathbf{r}^T$ w.r.t. $A_i^\mathcal{T}$ is called forward privacy leakage, defined by follows.*

$$FPL(A_i^\mathcal{T}, \mathcal{M}^t) \stackrel{\text{def}}{=} \sup_{l_i^t, l_i^{t'}, \mathbf{r}^t, \dots, \mathbf{r}^T} \log \frac{\Pr(\mathbf{r}^t, \dots, \mathbf{r}^T | l_i^t, D_K^t)}{\Pr(\mathbf{r}^t, \dots, \mathbf{r}^T | l_i^{t'}, D_K^t)}. \quad (4.7)$$

$$FPL(\mathcal{M}^t) \stackrel{\text{def}}{=} \max_{\forall A_i^\mathcal{T}, i \in \mathcal{U}} FPL(A_i^\mathcal{T}, \mathcal{M}^t). \quad (4.8)$$

By substituting Equation (4.5) and (4.7) into (4.4), we have

$$TPL(A_i^\mathcal{T}, \mathcal{M}^t) = BPL(A_i^\mathcal{T}, \mathcal{M}^t) + FPL(A_i^\mathcal{T}, \mathcal{M}^t) - PL_0(A_i^\mathcal{T}, \mathcal{M}^t). \quad (4.9)$$

Similarly, by expanding Equation (4.3) to one resembling Equation (4.4) and combining it with Equation (4.6) and (4.8), we have

$$TPL(\mathcal{M}^t) = BPL(\mathcal{M}^t) + FPL(\mathcal{M}^t) - PL_0(\mathcal{M}^t). \quad (4.10)$$

Intuitively, BPL and FPL are the privacy leakage w.r.t. the adversaries $A_i^\mathcal{T}(P_i^B)$ and $A_i^\mathcal{T}(P_i^F)$, respectively. TPL is the privacy leakage w.r.t. $A_i^\mathcal{T}(P_i^B, P_i^F)$. In Equation 4.10, we need to minus $PL_0(\mathcal{M}^t)$ because it is counted in both BPL and FPL. We will show more details in the following analysis.

BPL over time. For BPL, we first expand and simplify Equation (4.5) by Bayesian theorem and Lemma 5, $BPL(A_i^\mathcal{T}, \mathcal{M}^t)$ is equal to

$$\begin{aligned}
 & \sup_{\substack{l_i^t, l_i^{t'}, \\ \mathbf{r}^1, \dots, \mathbf{r}^{t-1}}} \log \frac{\sum_{l_i^{t-1}} \Pr(\mathbf{r}^1, \dots, \mathbf{r}^{t-1} | l_i^{t-1}, D_{\mathcal{K}}^{t-1}) \Pr(l_i^{t-1} | l_i^t)}{\sum_{l_i^{t-1'}} \underbrace{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^{t-1} | l_i^{t-1'}, D_{\mathcal{K}}^{t-1})}_{(i) BPL(A_i^T, \mathcal{M}^{t-1})} \underbrace{\Pr(l_i^{t-1'} | l_i^{t'})}_{(ii) P_i^B}} \\
 & + \sup_{l_i^t, l_i^{t'}, \mathbf{r}^t} \log \frac{\Pr(\mathbf{r}^t | l_i^t, D_{\mathcal{K}}^t)}{\underbrace{\Pr(\mathbf{r}^t | l_i^{t'}, D_{\mathcal{K}}^t)}_{(iii) PL_0(A_i^T, \mathcal{M}^t)}}. \tag{4.11}
 \end{aligned}$$

We now discuss the three annotated terms in the above equation. The first term indicates BPL at the previous time $t-1$, the second term is the backward temporal correlation determined by P_i^B , and the third term is equal to the privacy leakage w.r.t. adversaries in traditional DP (see Definition 8). Hence, BPL at time t depends on (i) BPL at time $t-1$, (ii) the backward temporal correlations, and (iii) the (traditional) privacy leakage of $\mathcal{M}^t$ (which is related to the privacy budget allocated to $\mathcal{M}^t$). By Equation (4.11), we know that if $t = 1$, $BPL(A_i^T, \mathcal{M}^1) = PL_0(A_i, \mathcal{M}^1)$; if $t > 1$, we have the following, where $\mathcal{L}^B(\cdot)$ is a *backward temporal privacy loss function* for calculating the accumulated privacy loss.

$$BPL(A_i^T, \mathcal{M}^t) = \mathcal{L}^B(BPL(A_i^T, \mathcal{M}^{t-1})) + PL_0(A_i, \mathcal{M}^t) \tag{4.12}$$

Equation (4.12) reveals that the BPL is calculated recursively and may *accumulate over time*, as shown in Example 2 (Fig.4.3(a)).

Example 2 (BPL due to previous releases.) Suppose that a DP mechanism $\mathcal{M}^t$ satisfies $PL_0(\mathcal{M}^t) = 0.1$ for each time $t \in [1, T]$, i.e., 0.1-DP at each time point. We now discuss BPL at each time point w.r.t. A_i^T with knowledge of backward temporal correlations P_i^B . In an extreme case, if P_i^B indicates the strongest correlation, say, $P_i^B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, then, at time t , A_i^T knows $l_i^t = l_i^{t-1} = \dots = l_i^1$, i.e., $D^t = D^{t-1} = \dots = D^1$ because of $D^t = \{l_i^t\} \cup D_{\mathcal{K}}^t$ for any $t \in [1, T]$. Hence, the continuous data release $\mathbf{r}^1, \dots, \mathbf{r}^t$ is equivalent to releasing the same database multiple times; the privacy leakage at each time point will accumulate from previous time points and increase linearly (Figure 4.3(a)(i)). In another extreme case, if there is no backward temporal correlation that is known to A_i^T (e.g., for the A_i in Definition 8 or $A_i^T(P_i^F)$ [§]), the backward privacy leakage at each time point is $PL_0(\mathcal{M}^t)$, as shown in Figure 4.3(a)(iii). Figure 4.3(a)(ii) depicts the backward

[§]Given the current l_i^t and P_i^F , i.e., $\Pr(l_i^t | l_i^{t-1})$, the adversary cannot derive $l_i^t = l_i^{t-1} = \dots = l_i^1$.

privacy leakage caused by $P_i^B = \begin{pmatrix} 0.8 & 0.2 \\ 0 & 1 \end{pmatrix}$, which can be finely quantified using our method (Algorithm 7) in Section 4.5.

FPL over time. For FPL, similar to the analysis of BPL, we expand and simplify Equation (4.5) by Bayesian theorem and Lemma 6, $FPL(A_i^T, \mathcal{M}^t)$ is equal to

$$\begin{aligned} & \sup_{\substack{l_i^t, l_i^{t'}, \\ \mathbf{r}^{t+1}, \dots, \mathbf{r}^T}} \log \frac{\sum_{l_i^{t+1}} \Pr(\mathbf{r}^{t+1}, \dots, \mathbf{r}^T | l_i^{t+1}, D_{\mathcal{K}}^{t+1}) \Pr(l_i^{t+1} | l_i^t)}{\sum_{l_i^{t+1}'} \underbrace{\Pr(\mathbf{r}^{t+1}, \dots, \mathbf{r}^T | l_i^{t+1}', D_{\mathcal{K}}^{t+1})}_{(i) FPL(A_i^T, \mathcal{M}^{t+1})} \underbrace{\Pr(l_i^{t+1}' | l_i^{t'})}_{(ii) P_i^F}} \\ & + \sup_{l_i^t, l_i^{t'}, \mathbf{r}^t} \log \frac{\Pr(\mathbf{r}^t | l_i^t, D_{\mathcal{K}}^t)}{\underbrace{\Pr(\mathbf{r}^t | l_i^{t'}, D_{\mathcal{K}}^t)}_{(iii) PL_0(A_i^T, \mathcal{M}^t)}}. \end{aligned} \quad (4.13)$$

By Equation (4.13), we know that if $t = T$, $FPL(A_i^T, \mathcal{M}^T) = PL_0(A_i, \mathcal{M}^T)$; if $t < T$, we have the following, where $\mathcal{L}^F(\cdot)$ is a *forward temporal privacy loss function* for calculating the increased privacy loss due to FPL at the next time.

$$FPL(A_i^T, \mathcal{M}^t) = \mathcal{L}^F(FPL(A_i^T, \mathcal{M}^{t+1})) + PL_0(A_i, \mathcal{M}^t) \quad (4.14)$$

Equation (4.14) reveals that FPL is calculated recursively and may *increase over time*, as shown in Example 3 (Fig.4.3(b)).

Example 3 (FPL due to future releases.) *Considering the same setting in Example 2, we now discuss FPL at each time point w.r.t. A_i^T with knowledge of forward temporal correlations P_i^F . In an extreme case, if P_i^F indicates the strongest correlation, say, $P_i^F = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, then, at time t , A_i^T knows $l_i^t = l_i^{t+1} = \dots = l_i^T$, i.e., $D^t = D^{t+1} = \dots = D^T$ because of $D^t = \{l_i^t\} \cup D_{\mathcal{K}}^t$ for any $t \in [1, T]$. Hence, the continuous data release $\mathbf{r}^t, \dots, \mathbf{r}^T$ is equivalent to releasing the same database multiple times; the privacy leakage at time t will increase when every time new release (i.e., $\mathbf{r}^{t+1}, \mathbf{r}^{t+2}, \dots$) happens. For example, we see that contrary to BPL, the FPL at time 1 is the highest (due to future releases at time 1 to 10) while FPL at time 10 is the lowest (since there is no future release with respect to time 10 yet). When $\mathbf{r}^{11}$ is released, all FPL at time $t \in [1, 10]$ will be updated. In another extreme case, if there is no forward temporal correlation that is known to A_i^T (e.g., for the A_i in Definition 8 or $A_i^T(P_i^B)$), then the forward privacy leakage at each time point is $PL_0(\mathcal{M}^t)$, as shown in Figure 4.3(b)(iii). Figure 4.3(b)(ii)*

depicts the forward privacy leakage caused by $P_i^F = \begin{pmatrix} 0.8 & 0.2 \\ 0 & 1 \end{pmatrix}$, which can be finely quantified using our method (Algorithm 7) in Section 4.5.

Remark 1. The extreme cases shown in Example 2 and 3 are the upper and lower bound of BPL and FPL. Hence, the backward temporal privacy loss function $\mathcal{L}^B(\cdot)$ in Equation (4.12) and the forward temporal privacy loss function $\mathcal{L}^F(\cdot)$ in Equation (4.14) satisfy $0 * \cdot \leq \mathcal{L}^B(\cdot) \leq 1 * \cdot$, where $\cdot$ is BPL at the previous time, and $0 * \cdot \leq \mathcal{L}^F(\cdot) \leq 1 * \cdot$, where $\cdot$ is FPL at the next time, respectively.

From Example 2 and 3, we know that: backward temporal correlation (i.e., P_i^B) does not affect FPL, and forward temporal correlation (i.e., P_i^F) does not affect BPL. In other words, adversary $A_i^T(P_i^B)$ only causes BPL; $A_i^T(P_i^F)$ only causes FPL; while $A_i^T(P_i^B, P_i^F)$ poses a risk on both BPL and FPL.

Figure 4.3(c) shows TPL, which is calculated using BPL and FPL (see Equation (4.10)). Given P_i^B and P_i^F , finely quantifying TPL is a challenge. We will design a novel algorithm to calculate them efficiently in Section 4.5.

4.4.3 DP under Temporal Correlations and Its Composability

In this section, we define α -DP $_{\mathcal{T}}$ to provide a privacy guarantee against temporal privacy leakage. We prove its sequential composition theorem and discuss the connection between α -DP $_{\mathcal{T}}$ and ϵ -DP in terms of event-level/user-level privacy [27] [28] and w-event privacy [30].

Definition 14 (α -DP $_{\mathcal{T}}$). For all user i in the database, if TPL of $\mathcal{M}^t$ (see Definition 11) is less than or equal to α , we say that $\mathcal{M}^t$ satisfies α -differential privacy under temporal correlation, denoted by α -DP $_{\mathcal{T}}$.

DP $_{\mathcal{T}}$ is an enhanced version of DP on temporal data. If the data are temporally independent (i.e., for all user i , both P_i^B and P_i^F are $\emptyset$), an ϵ -DP mechanism satisfies ϵ -DP $_{\mathcal{T}}$. If the data are temporally correlated (i.e., existing user i whose P_i^B or P_i^F is not $\emptyset$), an ϵ -DP mechanism satisfies α -DP $_{\mathcal{T}}$ where α is the increased privacy leakage and can be quantified using our framework.

One may wonder, for a sequence of DP $_{\mathcal{T}}$ mechanisms on the timeline, what is the overall privacy guarantee. Suppose that $\mathcal{M}^t$ satisfies ϵ_t -DP and poses risks of BPL and FPL as α_t^B and α_t^F , respectively. That is, $\mathcal{M}^t$ satisfies $(\alpha_t^B + \alpha_t^F - \epsilon_t)$ -DP $_{\mathcal{T}}$

at time t according to Equation (4.10). We formally define such overall privacy leakage based on Equation (4.3).

Definition 15 (TPL of a sequence of DP mechanisms). *The temporal privacy leakage of DP mechanisms $\{\mathcal{M}^t, \dots, \mathcal{M}^{t+j}\}$ where $j \geq 0$ is defined as follows.*

$$TPL(\{\mathcal{M}^t, \dots, \mathcal{M}^{t+j}\}) \stackrel{\text{def}}{=} \sup_{\substack{D^t, \dots, D^{t+j}, \\ D^{t'}, \dots, D^{t+j'}, \\ \mathbf{r}^1, \dots, \mathbf{r}^T}} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | D^t, \dots, D^{t+j})}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | D^{t'}, \dots, D^{t+j'})}$$

It is easy to see that, if $j = 0$, it is event-level privacy; if $t = 1$ and $j = |T - 1|$, it is user-level privacy.

Theorem 10 (Composition under Temporal Correlations). *A sequence of DP mechanism $\{\mathcal{M}^t, \dots, \mathcal{M}^{t+j}\}$ satisfies*

$$\begin{cases} (\alpha_t^B + \alpha_{t+1}^F) - DP_{\mathcal{T}} & j = 1 \\ (\alpha_t^B + \alpha_{t+j}^F + \sum_{k=1}^{k=j-1} \epsilon_{t+k}) - DP_{\mathcal{T}} & j \geq 2 \end{cases} \quad (4.15)$$

We omit the proofs of Theorems 10 due to space limitations.

When $t = 1$ and $j = T - 1$ in Theorem 10, we have the following corollary because $\alpha_1^B = BPL(\mathcal{M}^1) = PL_0(\mathcal{M}^1)$ and $\alpha_T^F = FPL(\mathcal{M}^T) = PL_0(\mathcal{M}^T)$.

Corollary 1. *The temporal privacy leakage of a combined mechanism $\{\mathcal{M}^1, \dots, \mathcal{M}^T\}$ is $\sum_{k=1}^{k=T} \epsilon_k$.*

It shows that temporal correlations do not affect the user-level privacy (i.e., protecting all the data on the timeline of each user), which is in line with the idea of group differential privacy: protecting all the correlated data in a bundle.

We now compare the privacy guarantee between DP and $DP_{\mathcal{T}}$. As we mentioned in Section 5.2.2, there are typically two privacy notions in continuous data release: *event-level* and *user-level* [27] [28]. Recently, *w*-event privacy [30] is proposed to merge the gap between event-level and user-level privacy. It protects the data in any *w*-length sliding window by utilizing the following sequential composition theorem of DP.

Theorem 11 (Sequential composition on independent data [47]). *Suppose that $\mathcal{M}^t$ satisfies ϵ_t -DP for each $t \in [1, T]$. A combined mechanism $\{\mathcal{M}^t, \dots, \mathcal{M}^{t+j}\}$ satisfies $\sum_{k=1}^{k=j} \epsilon_{t+k}$ -DP.*

Suppose that $\mathcal{M}^t$ satisfies ϵ -DP for each $t \in [1, T]$. According to Theorem 11, it achieves $T\epsilon$ -DP on user-level and $w\epsilon$ -DP on w -event level. We compare the privacy guarantee on independent data and temporally correlated data as follows.

Table 4.2: The privacy guarantee of ϵ -DP mechanisms.

Privacy Notion \ Data	independent	temporally correlated
event-level [27] [28]	ϵ -DP	α -DP $_{\mathcal{T}}$ ($\alpha \geq \epsilon$)
w -event [30]	$w\epsilon$ -DP	see Theorem 10
user-level [27] [28]	$T\epsilon$ -DP	$T\epsilon$ -DP $_{\mathcal{T}}$ (by Corollary 1)

It reveals that temporal correlations may blur the boundary between event-level privacy and user-level privacy. In an extreme case, the temporal privacy leakage of an ϵ -DP mechanism on event-level can be $T\epsilon$, i.e., $T\epsilon$ -DP $_{\mathcal{T}}$. Considering the examples shown in Figure 4.3, under the strongest temporal correlations, $\mathcal{M}^{10}$ satisfies 1-DP $_{\mathcal{T}}$ on event-level and a combined mechanism $\{\mathcal{M}^1, \dots, \mathcal{M}^{10}\}$ also satisfies 1-DP $_{\mathcal{T}}$ on user-level. Essentially, it is because the adversaries may infer $\{D^1, \dots, D^T\}$ (user-level) from D^t (event-level) using temporal correlations.

4.4.4 Discussion

We make a few important observations regarding our privacy analysis.

First, the temporal privacy leakage is defined in a personalized way. That is, the privacy leakage may be different for users with distinct temporal patterns (i.e., P_i^B and P_i^F). We define the overall temporal privacy leakage as the maximum one for all users, so that α -DP $_{\mathcal{T}}$ is compatible with the traditional ϵ -DP mechanism (using one parameter to represent the overall privacy level) and we can convert a traditional DP mechanism to bound the temporal privacy leakage. On the other hand, our definitions also can be compatible with personalized differential privacy (PDP) mechanisms [66], in which the personalized privacy budgets, i.e., a vector $[\epsilon_1, \dots, \epsilon_n]$, are allocated to each user. In other words, we can convert a PDP mechanism to bound the temporal privacy leakage for each user.

Second, in this chapter, we focus on the temporally correlated data and assume that the adversary has knowledge of temporal correlations modeled by Markov chain. However, it is possible that the adversary has knowledge about more sophisticated temporal correlation model or other types of correlations, such as

user-user correlations modeled by Gaussian Markov Random Field in [59]. Our contributions in this work can serve as primitives for quantifying the privacy risk under more advanced adversarial knowledge.

4.5 Calculating Temporal Privacy Leakage

In this section, we design algorithms for computing backward privacy leakage (BPL) and forward privacy leakage (FPL). We first show that both of them can be translated into the optimal solution of a *linear-fractional programming* problem [67]. Traditionally, this type of problem can be solved by simplex algorithm [68] in exponential time. By exploiting the constraints in this problem, we then design a method to solve it in polynomial time.

4.5.1 Problem formulation

According to the privacy analysis of BPL and FPL in Section 4.4.2, we need to solve the backward and forward temporal privacy loss functions $\mathcal{L}^B(\cdot)$ and $\mathcal{L}^F(\cdot)$ in Equations (4.12) and (4.14), respectively. By observing the structure of the first term in Equations (4.11) and (4.13), we can see that the calculations for recursive functions $\mathcal{L}^B(\cdot)$ and $\mathcal{L}^F(\cdot)$ are virtually in the same way. They calculate the increment of the input values (the previous BPL or the next FPL) based on temporal correlations (backward or forward). Although different degree of correlations result in different privacy loss functions, the methods for analyzing them are the same.

We now quantitatively analyze the temporal privacy leakage. In the following, we demonstrate the calculation of $\mathcal{L}^B(\cdot)$. The first term of Equation (4.11), i.e., $\mathcal{L}^B(BPL(A_i^T, \mathcal{M}^{t-1}))$ is as follows.

$$\sup_{\substack{l_i^t, l_i^{t'}, \\ \mathbf{r}^1, \dots, \mathbf{r}^{t-1}}} \log \frac{\sum_{l_i^{t-1}} \Pr(\mathbf{r}^1, \dots, \mathbf{r}^{t-1} | l_i^{t-1}, D_{\mathcal{K}}^{t-1}) \Pr(l_i^{t-1} | l_i^t)}{\sum_{l_i^{t-1'}} \underbrace{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^{t-1} | l_i^{t-1'}, D_{\mathcal{K}}^{t-1})}_{BPL(A_i^T, \mathcal{M}^{t-1})} \underbrace{\Pr(l_i^{t-1'} | l_i^{t'})}_{P_i^B}} \quad (4.16)$$

We now simplify the notations in the above formula. Let two arbitrary (different) rows in P_i^B be vectors $\mathbf{q} = (q_1, \dots, q_n)$ and $\mathbf{d} = (d_1, \dots, d_n)$. For example, suppose that $\mathbf{q}$ is the first row in the transition matrix of Figure 4.2(b). Then, the elements in $\mathbf{q}$ are: $q_1 = \Pr(l_i^{t-1} = loc_1 | l_i^t = loc_1)$, $q_2 = \Pr(l_i^{t-1} = loc_2 | l_i^t = loc_1)$,

$q_3 = \Pr(l_i^{t-1} = loc_3 | l_i^t = loc_1)$, etc. Let $\mathbf{x} = (x_1, \dots, x_n)^T$ be a vector whose elements indicate $\Pr(\mathbf{r}^1, \dots, \mathbf{r}^{t-1} | l_i^{t-1}, D_{\mathcal{K}}^{t-1})$ with distinct values of $l_i^{t-1} \in \mathbf{loc}$, e.g., x_1 denotes $\Pr(\mathbf{r}^1, \dots, \mathbf{r}^{t-1} | l_i^{t-1} = loc_1, D_{\mathcal{K}}^{t-1})$. We obtain the following by expanding $l_i^{t-1}, l_i^{t-1'} \in \mathbf{loc}$ in (4.16).

$$\begin{aligned} \mathcal{L}^B(BPL(A_i^T, \mathcal{M}^{t-1})) &= \sup_{\mathbf{q}, \mathbf{d} \in P_i^B} \log \frac{q_1 x_1 + \dots + q_n x_n}{d_1 x_1 + \dots + d_n x_n} \\ &= \sup_{\mathbf{q}, \mathbf{d} \in P_i^B} \log \frac{\mathbf{q}\mathbf{x}}{\mathbf{d}\mathbf{x}} \end{aligned}$$

Next, we formalize the problem and constraints. Suppose that $BPL(A_i^T, \mathcal{M}^{t-1}) = \alpha_{t-1}^B$. According to the definition of BPL (as the supremum), for any $x_j, x_k \in \mathbf{x}$, we have $e^{-\alpha_{t-1}^B} \leq \frac{x_j}{x_k} \leq e^{\alpha_{t-1}^B}$. Considering $\mathbf{x}$ as the variable vector and $\mathbf{q}, \mathbf{d}$ as the coefficient vectors, $\mathcal{L}^B(\alpha_{t-1}^B)$ is equal to the logarithm of the objective function (4.17) in the following maximization problem (4.17)~(4.19).

$$\text{maximize } \frac{\mathbf{q}\mathbf{x}}{\mathbf{d}\mathbf{x}} \quad (4.17)$$

$$\text{subject to } e^{-\alpha_{t-1}^B} \leq \frac{x_j}{x_k} \leq e^{\alpha_{t-1}^B}, \quad (4.18)$$

$$0 < x_j < 1 \text{ and } 0 < x_k < 1, \quad (4.19)$$

where $x_j, x_k \in \mathbf{x}$, $j, k \in [1, n]$.

The above is a form of *linear-fractional programming* [67], where the objective function is a ratio of two linear functions and the constraints are linear inequalities or equations. A linear-fractional programming problem can be converted into a sequence of linear programming problems [67] and then solved using the simplex algorithm [68] in time $O(2^n)$. When n is large, the computation is time consuming.

Bounding the objective function by constraints. We now investigate a more efficient method to solve this problem by exploiting the structure of constraints. From Inequalities (4.18) and (4.19), we know that the feasible region of the constraints are not empty and bounded; hence, an optimal solution exists. By exploiting the constraints, we prove the following theorem, which enables the optimal solution to be found in time $O(n^2)$.

We define some notations that will be frequently used in the following parts of this chapter. Suppose that the variable vector $\mathbf{x}$ consists of two parts (subsets): $\mathbf{x}^+$ and $\mathbf{x}^-$. Let the corresponding coefficients vectors be $\mathbf{q}^+, \mathbf{d}^+$ and $\mathbf{q}^-, \mathbf{d}^-$. Let $q = \sum \mathbf{q}^+$ and $d = \sum \mathbf{d}^+$. For example, suppose that $\mathbf{x}^+ = [x_1, x_3]$ and

$\mathbf{x}^- = [x_2, x_4, x_5]$. Then, we have $\mathbf{q}^+ = [q_1, q_3]$, $\mathbf{d}^+ = [d_1, d_3]$, $\mathbf{q}^- = [q_2, q_4, q_5]$, and $\mathbf{d}^- = [d_2, d_4, q_5]$. In this case, $q = q_1 + q_3$ and $d = d_1 + d_3$.

Theorem 12. *If the following Inequalities (4.20) and (4.21) are satisfied, the maximum value of the objective function in the problem (4.17)~(4.19) is $\frac{q(e^{\alpha_{t-1}^B} - 1) + 1}{d(e^{\alpha_{t-1}^B} - 1) + 1}$.*

$$\frac{q_j}{d_j} > \frac{q(e^{\alpha_{t-1}^B} - 1) + 1}{d(e^{\alpha_{t-1}^B} - 1) + 1}, \quad \forall j \in [1, n] \text{ where } q_j \in \mathbf{q}^+, d_j \in \mathbf{d}^+ \quad (4.20)$$

$$\frac{q_k}{d_k} \leq \frac{q(e^{\alpha_{t-1}^B} - 1) + 1}{d(e^{\alpha_{t-1}^B} - 1) + 1}, \quad \forall k \in [1, n] \text{ where } q_k \in \mathbf{q}^-, d_k \in \mathbf{d}^- \quad (4.21)$$

Proof. See Appendix B. □

As we mentioned previously, the calculations of $\mathcal{L}^B(\cdot)$ and $\mathcal{L}^F(\cdot)$ are identical. Therefore, given a transition matrix P_i^B (or P_i^F) and the previous BPL (or the next FPL), the increment of the backward (or forward) privacy loss is the maximum value in the above theorem for any two rows $\mathbf{q}$ and $\mathbf{d}$ in P_i^B (or P_i^F). We denote $FPL(A_i^T, \mathcal{M}^{t+1})$ by α_{t+1}^F .

$$\mathcal{L}^B(\alpha_{t-1}^B) = \max_{\mathbf{q}, \mathbf{d} \in P_i^B} \log \frac{q(e^{\alpha_{t-1}^B} - 1) + 1}{d(e^{\alpha_{t-1}^B} - 1) + 1} \quad (4.22)$$

$$\mathcal{L}^F(\alpha_{t+1}^F) = \max_{\mathbf{q}, \mathbf{d} \in P_i^F} \log \frac{q(e^{\alpha_{t+1}^F} - 1) + 1}{d(e^{\alpha_{t+1}^F} - 1) + 1} \quad (4.23)$$

Corollary 2. *If Inequalities (4.20) and (4.21) are satisfied, we have $q_j > d_j$ in which $q_j \in \mathbf{q}^+$ and $d_j \in \mathbf{d}^+$.*

We omit the proof due to space limitations.

Now, we simply examine the $\mathcal{L}^B(\cdot)$ and $\mathcal{L}^F(\cdot)$ in Equations (4.22) and (4.23). First, we have $0 \leq \mathcal{L}^B(\alpha_{t-1}^B)$ and $0 \leq \mathcal{L}^F(\alpha_{t-1}^B)$ because of $q > d$. Second, when q and d have the largest difference (e.g., $\mathbf{q} = (1, 0)$, $\mathbf{d} = (0, 1)$ and hence $q = 1, d = 0$), it follows that $\mathcal{L}^B(\alpha_{t-1}^B) \leq \alpha_{t-1}^B$ and $\mathcal{L}^F(\alpha_{t+1}^F) \leq \alpha_{t+1}^F$. Therefore, it is in accordance with Remark 1. The advantage of Equations (4.22) and (4.23) is being able to finely quantify BPL and FPL w.r.t. arbitrary P_i^B and P_i^F .

4.5.2 Privacy Leakage Quantification Algorithm

The next question is how do we find q and d (or $\mathbf{q}^+$ and $\mathbf{d}^+$) that gives the maximum objective function. Inequalities (4.20) and (4.21) in Theorem 12 are sufficient conditions for obtaining such optimal value. Corollary 2 gives necessary conditions for satisfying Inequalities (4.20) and (4.21). Based on the above analysis, we design Algorithm 7 for computing BPL or FPL .

Algorithm 7: Finding BPL or FPL

Input: P_i (P_i^B or P_i^F); α (α_{t-1}^B or α_{t+1}^F); ϵ_t (i.e., $PL_0(\mathcal{M}^t)$).
Output: BPL at time t or FPL at time t

```

1  $\mathcal{L}_i \leftarrow 0$  //the value of Equation (4.22) or (4.23)
2 foreach two rows  $\mathbf{q}, \mathbf{d} \in P_i$  do
3     foreach  $q_j \in \mathbf{q}, d_j \in \mathbf{d}$  do
4         if  $q_j > d_j$  then add  $q_j$  to  $\mathbf{q}^+$ ; add  $d_j$  to  $\mathbf{d}^+$ 
5      $update \leftarrow false$ ;
6     do //find  $q, d$  by Theorem 12 and Corollary 2
7          $\mathbf{q} \leftarrow \sum \mathbf{q}^+; \mathbf{d} \leftarrow \sum \mathbf{d}^+$  //satisfy Inequality (4.21)
8         foreach  $q_j \in \mathbf{q}^+, d_j \in \mathbf{d}^+$  do
9             //for satisfying Inequality (4.20)
10            if  $q_j/d_j \leq (q * (e^\alpha - 1) + 1) / (d * (e^\alpha - 1) + 1)$ 
11            then  $\mathbf{q}^+ \leftarrow \mathbf{q}^+ - q_j; \mathbf{d}^+ \leftarrow \mathbf{d}^+ - d_j; update \leftarrow true$ ;
12    while  $update$ 
13    if  $\mathcal{L}_i < \log \frac{q * (e^\alpha - 1) + 1}{d * (e^\alpha - 1) + 1}$  then  $\mathcal{L}_i \leftarrow \log \frac{q * (e^\alpha - 1) + 1}{d * (e^\alpha - 1) + 1}$ 
13 return  $\mathcal{L}_i + \epsilon_t$  //by Equation (4.12) or (4.14)
    
```

Computing BPL or FPL by solving the linear-fractional programming. According to the definition of BPL and FPL, we need to return the maximum privacy leakage (Line 12) w.r.t. any two rows in the given transition matrix (Lines 2). Lines 3~11 are to solve one linear-fractional programming problem (4.17)~(4.19) w.r.t two specific rows in the transition matrix. In Lines 3 and 4, we divide the variable vector $\mathbf{x}$ into two parts according to Corollary 2, which gives the necessary condition for finding the maximum solution: if the coefficients $q_j \leq d_j$, they are not in $\mathbf{q}^+$ and $\mathbf{d}^+$ that satisfy Inequalities (4.20) and (4.21). In other words, if $q_j > d_j$, they are “candidates” in $\mathbf{q}^+$ and $\mathbf{d}^+$ that gives the maximum objective function. In Lines 5 to 11, we further check $\mathbf{q}^+$ and $\mathbf{d}^+$ whether they satisfy Inequalities (4.20) and (4.21). According to Line 7, it is clear that any subset of $\mathbf{q}^+$ and $\mathbf{d}^+$ automatically satisfy Inequality (4.21). In Lines 8–10, we remove the pairs $q_j \in \mathbf{q}^+$ and $d_j \in \mathbf{d}^+$ that do not satisfy Inequality

(4.20). Note that the values of q and d (recall that $q = \sum \mathbf{q}^+$ and $d = \sum \mathbf{d}^+$) will be recalculated due to such “update” (deletion in Line 10).go If q and d are updated, we need to recheck each pair of q_j and d_j in the current set of $\mathbf{q}^+$ and $\mathbf{d}^+$ until every pair of them satisfies Inequality (4.20).

A subtle question may arise regarding such “update”. In Lines 8~10, if *several* pairs of q_j and d_j do not satisfy Inequality (4.20), say, $\{q_1, d_1\}$ and $\{q_2, d_2\}$, one may wonder if it is possible that, after removing $\{q_1, d_1\}$ from $\mathbf{q}^+$ and $\mathbf{d}^+$, Inequality (4.20) can be satisfied for $\{q_2, d_2\}$ due to the update of q and d , i.e., $\frac{q_2}{d_2} > \frac{(q-q_1)*(e^\alpha-1)+1}{(d-d_1)*(e^\alpha-1)+1}$. We show that this is not possible. If $\frac{q_1}{d_1} \leq \frac{q*(e^\alpha-1)+1}{d*(e^\alpha-1)+1}$, we have $\frac{q*(e^\alpha-1)+1}{d*(e^\alpha-1)+1} \leq \frac{(q-q_1)*(e^\alpha-1)+1}{(d-d_1)*(e^\alpha-1)+1}$. Hence, $\frac{q_2}{d_2} \leq \frac{q*(e^\alpha-1)+1}{d*(e^\alpha-1)+1} \leq \frac{(q-q_1)*(e^\alpha-1)+1}{(d-d_1)*(e^\alpha-1)+1}$. Therefore, we can remove multiple pairs of q_j and d_j that do not satisfy Inequality (4.20) at one time (Lines 8~10).

It is easy to know that the update will be terminated with non-empty $\mathbf{q}^+$ and $\mathbf{d}^+$. If only one element is left in each of $\mathbf{q}^+$ and $\mathbf{d}^+$, say, q_n and d_n , respectively, then Inequality (4.20) is true (i.e., $\frac{q_n}{d_n} > \frac{q_n*(e^\alpha-1)+1}{d_n*(e^\alpha-1)+1}$) because of $q_n > d_n$.

Algorithm complexity. The time complexity for solving one linear-fractional programming problem (Lines 3~11) w.r.t. two specific rows of the transition matrix is $O(n^2)$ because Line 9 may iterate $n * (n - 1)$ times in the worst case. The overall time complexity of Algorithm 7 is $O(n^4)$.

4.6 Bounding Temporal Privacy Leakage

In this section, we design private data release algorithms that can be used to convert a traditional DP mechanism into one satisfying α -DP $_{\mathcal{T}}$ by allocating calibrated privacy budgets.

We first investigate the upper bound of BPL and FPL. We have demonstrated that BPL and FPL may accumulate and increase over time in Figure 4.3. A natural question is that: is there a limit of BPL and FPL over time. For $\mathcal{M}^t$ that satisfies ϵ -DP at each $t \in [1, T]$, we know that a loose upper bound of BPL or FPL over time T is $T\epsilon$ according to Remark 1. When T is unknown, giving the upper bound of BPL or FPL is a challenge.

Theorem 13. *Given a transition matrix P_i^B (or P_i^F) representing temporal correlation, let q and d be the ones that give the maximum value in Equation (4.22) (or Equation (4.23)). For $\mathcal{M}^t$ that satisfies ϵ -DP at each $t \in [1, T]$, there are four cases regarding the supremum of BPL (or FPL) over time.*

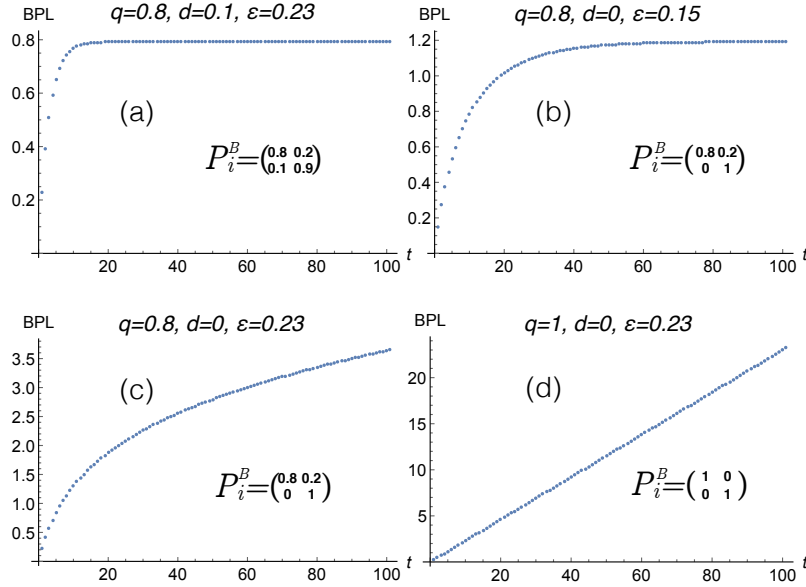


Figure 4.4: Examples of the maximum BPL over time.

$$\begin{cases} \log \frac{\sqrt{4de^\epsilon(1-q)+(d+qe^\epsilon-1)^2}+d+qe^\epsilon-1}{2d} & d \neq 0 \\ \log \frac{(1-q)e^\epsilon}{1-qe^\epsilon} & d = 0 \text{ and } q \neq 1 \text{ and } \epsilon \leq \log(1/q) \\ \text{not exist} & d = 0 \text{ and } q \neq 1 \text{ and } \epsilon > \log(1/q) \\ \text{not exist} & d = 0 \text{ and } q = 1 \end{cases}$$

We omit the proof due to space limitations.

The above theorem is applicable for both BPL and FPL because the calculation of BPL and FPL is the same. According to the previous analysis, we can consider that the growth of BPL and FPL is in the same manner but in the reversed directions on the timeline (see Figure 4.3(a)(b)).

Example 4 (The supremum of the increased BPL over time.). Suppose that $\mathcal{M}^t$ satisfies ϵ -DP at each time point. In Figure 4.4, we demonstrate the maximum BPL w.r.t. different ϵ and different transition matrices that represents P_i^B . In (a) and (b), the supremum does not exist. In (c) and (d), we can directly calculate the supremum using Theorem 13. The results are in line with the ones that compute BPL step by step at each time point using Algorithm 7.

Achieving α -DP $_{\mathcal{T}}$ by limiting upper bound. We now design a data release algorithm utilizing Theorem 13 to bound TPL. Theorem 13 tells us that, if it is not the strongest temporal correlation (i.e., $d = 0$ and $q = 1$), we may bound BPL

or FPL within a desired value by allocating an appropriate privacy budget to a traditional DP mechanism at each time point. A problem is that, in Theorem 13, the q and d are assumed to be the ones that give the maximum value of the objective function; however, they are initially unknown. According to our analysis of Algorithm 7, such q and d depend on not only the given transition matrix but also the previous BPL (or the next FPL); however, the “previous BPL” is not clear when BPL achieves its supremum at some time point. To solve this problem, we can consider that, if T is close to infinite, BPL at time T and $T+1$ are both supremum, so that we can find q and d that give the maximum objective function using Algorithm 7 by setting the “previous BPL” to such supremum. Now, we can find an appropriate ϵ to bound BPL based on Theorem 13. For example, if $d = 0$ and $q \neq 1$, we can solve an equation with one variable ϵ : $\log \frac{(1-q)e^\epsilon}{1-qe^\epsilon} = \alpha^B$ (it is easy to prove that a positive solution always exists) where α^B is a desirable privacy level. Similarly, we can restrict FPL within a given value. We use this idea to bound both BPL and FPL, as shown in Algorithm 8.

Algorithm 8: Releasing Data with α -DP $_{\mathcal{T}}$ by upper bound

Input: P_i^B and P_i^F , $i \in \mathcal{U}$; α (desired privacy level).
Output: private data satisfying α -DP $_{\mathcal{T}}$

```

1 foreach user  $i \in \mathcal{U}$  do
2   Initialize  $\alpha^B \in (0, \alpha]$  as the supremum of BPL;
3   Find  $q^B$  and  $d^B$  that give the maximum value in Eq.(4.22) using Algorithm 7 with
   input  $\alpha_{t-1}^B = \alpha^B$ ;
4   Find  $\epsilon_i^B$  using Theorem 13 with the above  $q^B$ ,  $d^B$  and  $\alpha^B$ ;
5    $\alpha^F \leftarrow \alpha - \alpha^B + \epsilon_i^B$ ; //see Equation (4.9)
6   Find  $q^F$  and  $d^F$  using Algorithm 7 with input  $\alpha_{t+1}^F = \alpha^F$ 
7   Find  $\epsilon_i^F$  using Theorem 13 with the above  $q^F$ ,  $d^F$  and  $\alpha^F$ ;
8   if  $\epsilon_i^B < \epsilon_i^F$  then goto Line 2, initialize a larger  $\alpha^B$ ;
9   else if  $\epsilon_i^B > \epsilon_i^F$  then goto Line 2, initialize a smaller  $\alpha^B$ ;
10  else  $\epsilon_i \leftarrow \epsilon_i^B$ 
11  $\epsilon \leftarrow \min\{\epsilon_i, i \in \mathcal{U}\}$ ;
12 return  $\epsilon$ -DP data at each time point

```

Achieving α -DP $_{\mathcal{T}}$ by privacy leakage quantification. Algorithm 8 guarantees that the temporal privacy leakage never exceeds the given value, no matter how long the T is. However, when T is too short so that the accumulation of temporal privacy leakage have not resulted in a significant increase, we may not take the full advantage of the privacy budgets. Our observation is that, the DP mechanisms at the first and last time points should be allocated more budgets because

they are relatively more “influential” in term of privacy loss. For example, BPL of $\mathcal{M}^t, t \in [2, T]$ is affected by the first mechanism $\mathcal{M}^1$, and FPL of $\mathcal{M}^t, t \in [1, T-1]$ is affected by the last mechanism $\mathcal{M}^T$. Our idea is to allocate more privacy budgets to $\mathcal{M}^1$ and $\mathcal{M}^T$ so that both BPL and FPL are bounded in given values at each time point. For example, if we want that BPL at every time points are exactly the same value α^B , i.e., $BPL(\mathcal{M}^1) = \dots = BPL(\mathcal{M}^T) = \alpha^B$, then we need to make sure: (i) $PL_0(\mathcal{M}^1) = \alpha^B$ and (ii) $\mathcal{L}^B(\alpha^B) + \epsilon_m^B = \alpha^B$ in which ϵ_m^B is the privacy budget allocated in the “middle” of the timeline, i.e., from 2 to $T-1$. We can solve the above equations to obtain ϵ_m^B ensuring $BPL(\mathcal{M}^1) = \dots = BPL(\mathcal{M}^T) = \alpha^B$. Similarly, we can bound FPL in a α^F by finding another ϵ_m^F . If $\epsilon_m^B \neq \epsilon_m^F$, we can assign ϵ_m as $\min\{\epsilon_m^B, \epsilon_m^F\}$ to ensure both BPL and FPL are bounded in $\min\{\alpha^B, \alpha^F\}$. It is easy to know that, when $\epsilon_m^B = \epsilon_m^F$, we can exactly achieve α -DP $_{\mathcal{T}}$ at each time point.

Algorithm 9: Releasing Data with α -DP $_{\mathcal{T}}$ by quantification

Input: P_i^B and P_i^F , $i \in \mathcal{U}$; α (desired privacy level).
Output: private data satisfying α -DP $_{\mathcal{T}}$

```

1 foreach  $user\ i \in \mathcal{U}$  do
2   Initialize  $\alpha^B \in (0, \alpha]$  as the supremum of BPL,  $\epsilon_{i,1} = \alpha^B$ ;
3   Find  $q^B$  and  $d^B$  using Algorithm 7 with input  $\alpha_{t-1}^B = \epsilon_{i,1}$ 
4   Find  $\epsilon_{i,m}^B$  by solving  $\mathcal{L}^B(\epsilon_{i,1}) + \epsilon_{i,m}^B = \epsilon_{i,1}$  with  $q^B, d^B$ ;
5    $\epsilon_{i,T} \leftarrow \alpha - \epsilon_{i,1} + \epsilon_{i,m}^B$ ; //see Equation (4.9)
6   Find  $q^F$  and  $d^F$  using Algorithm 7 with input  $\alpha_{t+1}^F = \epsilon_{i,T}$ 
7   Find  $\epsilon_{i,m}^F$  by solving  $\mathcal{L}^F(\epsilon_{i,T}) + \epsilon_{i,m}^F = \epsilon_{i,T}$  with  $q^F, d^F$ ;
8   if  $\epsilon_{i,m}^B < \epsilon_{i,m}^F$  then goto Line 2, initialize a larger  $\alpha^B$ ;
9   else if  $\epsilon_{i,m}^B > \epsilon_{i,m}^F$  then goto Line 2, initialize a smaller  $\alpha^B$ ;
10  else  $\epsilon_{i,m} \leftarrow \epsilon_{i,m}^B$ 
11  $\epsilon_1 \leftarrow \min\{\epsilon_{i,1}, i \in \mathcal{U}\}$ ,  $\epsilon_t \leftarrow \min\{\epsilon_{i,m}, i \in \mathcal{U}\}$ ,  $\epsilon_T \leftarrow \min\{\epsilon_{i,T}, i \in \mathcal{U}\}$ ;
12 return  $\epsilon_t$ -DP data at  $t \in [1, T]$ 
    
```

We note that, the initializations of both α^B in Algorithm 8 and 9 are nontrivial: too large or too small α^B results in more iterations to converge to $\epsilon_{i,m}^B = \epsilon_{i,m}^F$. We can prove that $\epsilon_{i,m}^B = \epsilon_{i,m}^F$ always can be achieved. We delegate the detailed descriptions and proofs to the long version of our chapter.

4.7 Experimental Evaluation

In this section, we design experiments for the following: (1) verifying the runtime and correctness of our privacy leakage quantification algorithm (Algorithm 7),

(2) investigating the impact of the temporal correlations on privacy leakage and (3) evaluating the data release Algorithms 8 and 9. We implemented all the algorithms in Java and conducted the experiments on a machine with an Intel Core i7 2.8GHz CPU and 16 GB RAM running OSX El Capitan.

The setting of temporal correlations. To evaluate if our privacy loss quantification algorithms can perform well under diverse circumstances, we need different degrees of temporal correlations. Although there are well studied methods to estimate the temporal correlations, in our experiments, we *generate* the correlations (transition matrices) directly to eliminate the effect of different estimation algorithms or datasets.

We now present a method for obtaining different degrees of temporal correlations. First, we generate a transition matrix indicating the “strongest” correlation that contains a cell with probability 1.0 at each row but for different columns (this type of transition matrix will lead to an upper bound of TPL, as shown in Examples 2 and 3). Then, we perform *Laplacian smoothing* [69], which is a method originally used to smooth a polygonal mesh, to *uniformize* the probabilities of P_i in different degree. Let p_{jk} be an element at the j th row and k th column of the matrix P_i . The new probabilities $\hat{p}_{jk}$ are generated using Equation (4.24), where s is a positive parameter that controls the degrees of smoothing. A smaller s results in a stronger temporal correlation.

$$\hat{p}_{jk} = \frac{p_{jk} + s}{\sum_{u=1}^n (p_{ju} + s)} \quad (4.24)$$

We note that, the degrees of correlation with s are only comparable with each other under the same n (i.e., $|\mathbf{loc}|$).

4.7.1 Runtime of Privacy Quantification Algorithms

In this section, we compare the runtime of our algorithm with `Gurobi`[¶] and `lp_solve`[‡], which are two well-known softwares for solving optimization problems, e.g., the linear-fractional programming problem (4.17)~(4.19) in our setting. We run our privacy quantification algorithm 30 times, and run `Gurobi` and `lp_solve` 5 times (because they are very time-consuming), and then calculate the average runtime for each of them. At each time, we randomly generate a transition

[¶]<http://www.gurobi.com/>. Commercial software. We use version 6.5.

[‡]<http://lpsolve.sourceforge.net/>. Open source software. We use version 5.5.

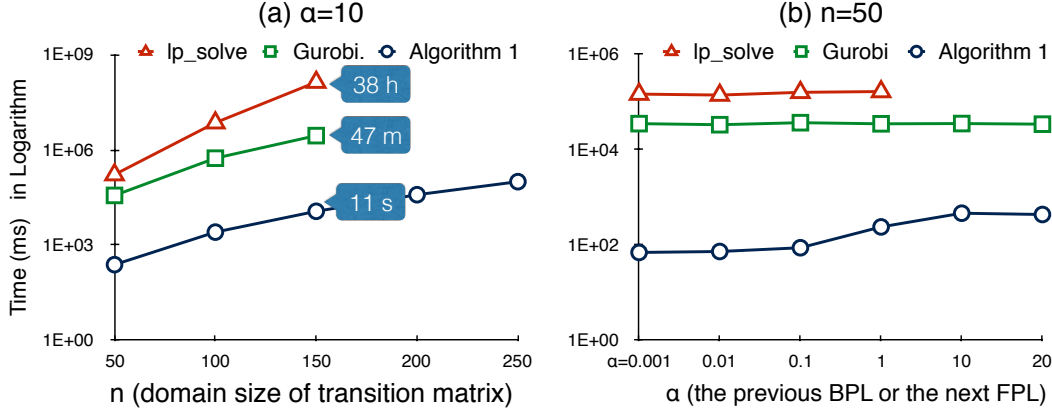


Figure 4.5: Runtime of Privacy Quantification Algorithms.

matrix P_i whose elements are uniformly drawn from $[0, 1]$. We verified that the optimal solution returned by the three algorithms are the same. In the following, we describe two factors that may affect the runtime: α as BPL at the previous time point or at the next time point (i.e., one input of Algorithm 7), n as the domain size of transition matrix. The results are shown in Figure 4.5.

Runtime vs. n . In Figure 4.5(a), we show the runtime of the three algorithms with inputs of $\alpha = 10$ and a $n \times n$ random probability matrix P_i . The runtime of all algorithms increase along with n because n is the number of variables in our linear-fractional program. Algorithm 7 significantly outperforms `Gurobi` and `lp_solve`. For example, in Figure 4.5(a), when $n = 150$, Algorithm 7 only spends 11 seconds, whereas the runtime of `Gurobi` and `lp_solve` are about 47 minutes and 38 hours, respectively. Since `Gurobi` and `lp_solve` spend tremendous time when $n > 150$, we omit to present them.

Runtime vs. α . In Figure 4.5(b), we show that, a larger previous BPL (or the next FPL), i.e., α , may lead to higher runtime of Algorithm 7, whereas `Gurobi` and `lp_solve` are stable for varying α . The reason is that, when α is large, Algorithm 7 may take more time in Lines 9 and 10 for updating each pair of $q_j \in \mathbf{q}^+$ and $d_j \in \mathbf{d}^+$ to satisfy Inequality (4.20). An update in Line 10 is more likely occur due to a large α because $\frac{q(e^\alpha - 1) + 1}{d(e^\alpha - 1) + 1}$ is increasing with α . However, such growth of runtime along with α will not last so long because the update happens $n - 1$ times in the worse case (according to our previous analysis, the update will be terminated if only one element is left in $\mathbf{q}^+$). As shown in Figure 4.5(b), when $\alpha > 10$, the runtime of Algorithm 7 becomes stable. We only obtain a part of the runtime for `lp_solve` because a precision problem occurs when $\alpha \geq 10$ due to the

design of `lp_solve`.

4.7.2 Impact of Temporal Correlations on Privacy Leakage

In this section, for the convenience of explanation, we only present the impact of temporal correlations on BPL because the growth of BPL and FPL are in the same way but in the reversed directions on the timeline. We examined s values in Equation (4.24) ranging from 0.005 to 1. We set n to 50 and 200. Let ε be the privacy budget of $\mathcal{M}^t$ at each time point. We test $\varepsilon = 1$ and 0.1. The results are shown in Figure 4.6 and are summarized as follows.

Privacy Leakage vs. s . Figure 4.6 shows that the privacy leakage caused by a non-trivial temporal correlation will increase over time, and such growth first increases sharply and then remains stable because the increment is calculated recursively. The increase caused by a stronger temporal correlations (i.e., smaller s) is steeper, and the time for the increase is longer. Consequently, stronger correlations result in higher privacy leakage.

Privacy Leakage vs. ε . Comparing Figures 4.6(a) and (b), we found that 0.1-DP significantly delayed the growth of privacy leakage. Taking $s = 0.005$, for example, the noticeable increase continues for almost 8 timestamps when $\varepsilon = 1$ (Figures 4.6(a)), whereas it continues for approximately 80 timestamps when $\varepsilon = 0.1$ (Figures 4.6(b)). However, after a sufficient long time, the privacy leakage in the case of $\varepsilon = 0.1$ is not substantially lower than that for the case of $\varepsilon = 1$ under stronger temporal correlations because although the privacy leakage is eliminated at each time point by setting a small privacy budget, the adversaries can eventually learn sufficient information from the continuous releases.

Privacy Leakage vs. n . Under the same s , TPL is smaller when n (dimension of the transition matrix) is larger, as shown in the lines $s = 0.005$ with $n = 50$ and $n = 200$ of Figure 4.6. This is because the transition matrices tend to be uniform (weaker correlations) when the dimension is larger.

In conclusion, the experiments reveal that our quantification algorithms can flexibly respond to different degrees of temporal correlations.

4.7.3 Evaluation of Data Releasing Algorithms

In this section, we first show a visualization of privacy allocation of Algorithms 8 and 9, then we compare the data utility in terms of Laplace noise.

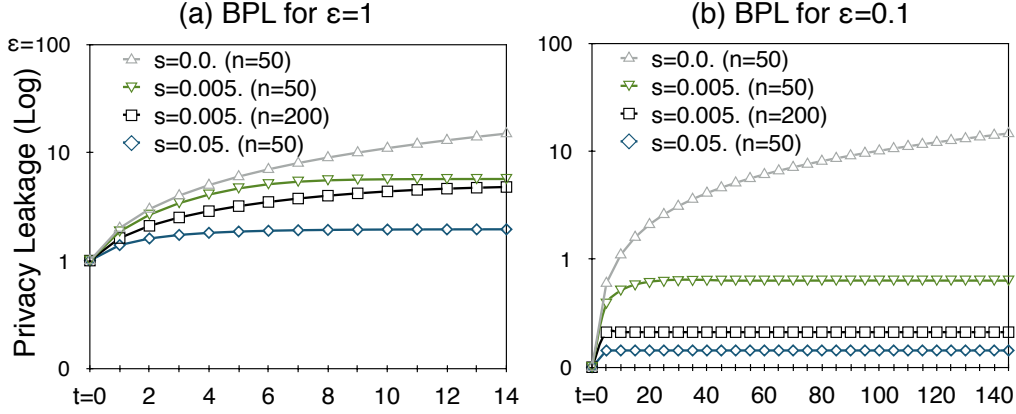


Figure 4.6: Evaluation of BPL.

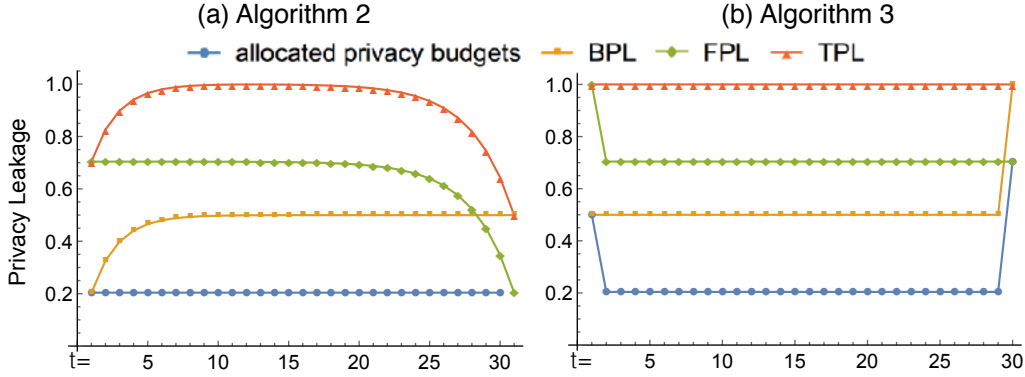

 Figure 4.7: Data Release Algorithms with 1-DP_T .

Figure 4.7 shows an example of budget allocation, w.r.t. $P_i^B = \begin{pmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{pmatrix}$ and $P_i^F = \begin{pmatrix} 0.8 & 0.2 \\ 0.1 & 0.9 \end{pmatrix}$. The goal is 1-DP_T . It is easy to see that Algorithm 9 has better data utility because it exactly achieves the desired privacy level.

Figure 4.8 shows the data utility of Algorithms 8 and 9 with 2-DP_T . We calculate the absolute value of the Laplace noise with the allocated budgets (as shown in Figure 4.7). Higher value of noise indicate low data utility. In Figure 4.8(a), we test the data utility under backward and forward temporal correlation both with parameter $s = 0.001$, which means relatively strong correlation. It shows that, when T is short, Algorithm 9 outperforms Algorithm 8. In other words, whenever how long T is, Algorithm 8 perturb data in the same way. In Figure 4.8(b), we investigate the data utility under different degree of correlations. The dash line indicates the absolute value of Laplace noise if no temporal correlation exists (privacy budget is 2). It is easy to see that the data utility significantly

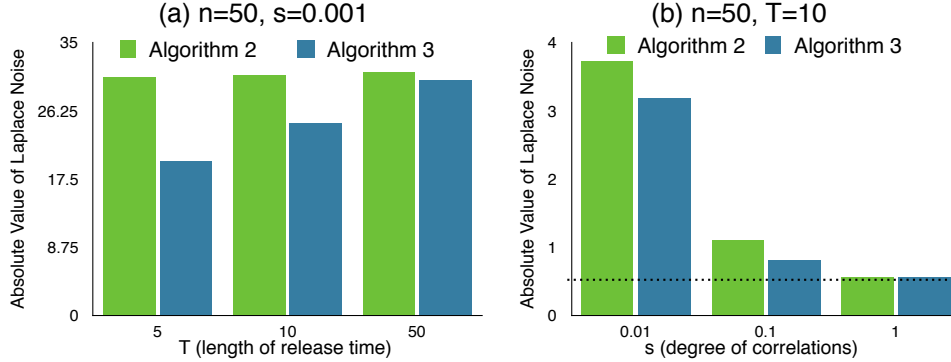


Figure 4.8: Data utility of 2-DP_T mechanisms.

decays under strong correlation $s = 0.01$.

4.8 Related Work

Several studies have questioned whether differential privacy is valid for correlated data. Kifer and Machanavajjhala [46] [53] [54] first raised the important issue that differential privacy may not guarantee privacy if adversaries know the data correlations. In their line of work, they [46] argued that it is not possible to ensure any utility in addition to privacy without making assumptions about the data-generating distribution and the background knowledge available to an adversary. To this end, they proposed a general and customizable privacy framework called *PufferFish*, in which the potential secrets, discriminative pairs, and data generation need to be explicitly defined. Yang et al. [59] further investigated differential privacy on correlated tuples described using a proposed Gaussian correlation model. The privacy leakage w.r.t. adversaries with specified prior knowledge can be efficiently computed. The major difference between correlation $\mathbb{G}$ in this work and Gaussian correlation model $\mathcal{G}$ in [59] is that the latter requires a connected graph (i.e., any user in the database should have at least one correlated user), whereas the $\mathbb{G}$ is not necessarily connected (and is thus more practical).

Zhu et al. [70] proposed correlated differential privacy by redefining the sensitivity of queries on correlated data; however, the privacy guarantee provided by this definition for spatio-temporal data is unclear. Very recently, Liu et al. [60] proposed dependent differential privacy by introducing dependence coefficients for analyzing the sensitivity of different queries under probabilistic dependences

between tuples. However, such dependence coefficients do not easily account for the spatio-temporal correlations.

Dwork et al. first studied *differential privacy under continual observation* and proposed event-level/user-level privacy [27] [28]. The previous studies in this setting focused on the problems of high dimension [23] [71] [31], infinite sequence [72] [26] [30], data sparsity [71], sliding window queries [73], and real-time publishing [29]. However, none of them addressed the problem of privacy loss under temporal correlations.

To the best of our knowledge, no study has reported the risk of differential privacy under temporal correlations for the continuous aggregate release setting. Although a few studies [61] [63] have considered a similar adversarial model in which the adversaries have prior knowledge of temporal correlations represented by Markov chains, they focused on location privacy in the single-user setting. Shokri et al. [61] proposed an evaluation framework for location privacy protection, assuming that the adversary knows the transition probabilities of each user. Xiao et al. [63] proposed a mechanism extended DP for single user location sharing under temporal correlations modeled by Markov chains. In contrast, the scenario in this chapter focuses on quantifying the privacy loss of traditional DP mechanisms under temporal correlations for continuous aggregate release setting.

4.9 Conclusions

In this chapter, we quantified the risk of differential privacy under temporal correlations by formalizing, analyzing and calculating the privacy loss against adversary who has varying degrees of temporal correlations. This work opens up interesting future research directions, such as modeling temporal correlations with other type of correlations (e.g. tuple-wise correlations), and combining our methods with the previous studies that neglected the effect of temporal correlations in order to bound the temporal privacy leakage.

CHAPTER 5

DIFFERENTIAL PRIVACY UNDER USER-USER CORRELATIONS

5.1 Introduction

As we mentioned in Chapter 4, data independence, which is a potential assumption of the DP mechanisms, has been neglected by most previous works that attempted to apply DP in practice. Recent studies [46] [53] [54] point out that traditional DP may not guarantee the expected privacy on correlated data. The following example shows that the user-user correlations may degrade the privacy guarantee of DP.

Example 5. *Consider the scenarios of aggregate location data release and medical data release illustrated in Figure 5.1. A trusted server releases the counts of people at each location at each time in Table5.1(b), or release the counts of patients with the disorders d_1 and d_2 in Table5.1(d). Traditionally, according to the Laplace mechanism [21], adding $\text{Lap}(1/\epsilon)$ noise* to perturb each count can achieve ϵ -DP for protecting individual privacy if the users are independent of each other (i.e., modification of a single user has no impact on others). However, the expected privacy guarantee may decay due to user-user correlations. For example*

* $\text{Lap}(b)$ denotes a Laplace distribution with variance $2b^2$.

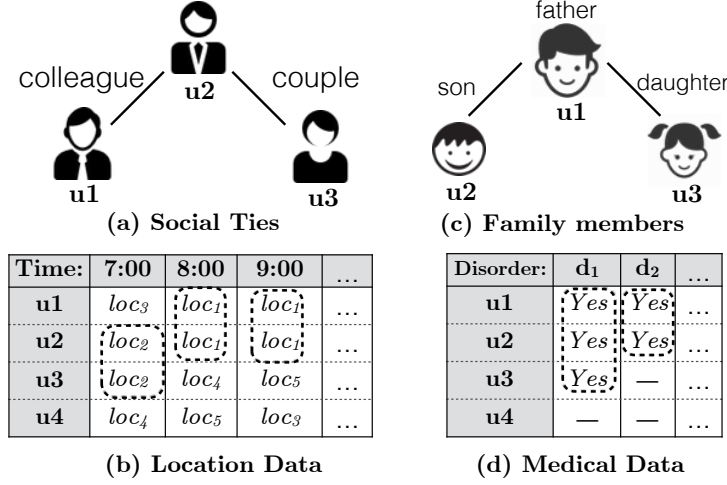


Figure 5.1: Examples of user-user correlations.

in location data release, an attacker may have auxiliary information as shown in Figure 5.1(a): (1) Users u_1 and u_2 are always at the same location during working hours (e.g., they are colleagues); (2) Users u_2 and u_3 are always at the same location during leisure time (e.g., they are newlyweds). There clear pattern are shown by the dashed lines of Figure 5.1(c). Due to a higher sensitivity caused by such patterns (i.e., modification of a single user impacts all correlated users), adding $\text{Lap}(1/\epsilon)$ noise only achieves 2ϵ -DP at each time (recall that larger values of ϵ result in larger privacy leakage). Similarly, in medical data release, if an attacker has knowledge about genealogical information as shown in Figure 5.1(c), he/she may know: (1) Users u_1 , u_2 and u_3 have the same value for disorder d_1 (e.g., a generic disease, spontaneously hypertensive, which exhibits familial aggregation) (2) User u_1 and u_2 have the same value for d_2 (e.g., color blindness which the causing gene is carried on the X chromosome). Adding $\text{Lap}(1/\epsilon)$ noise only achieve 3ϵ -DP because modification of a single user impacts 3 users in the worst case for such medical data.

It is reasonable to consider that adversaries may obtain user-user correlations, which commonly exist in our real life and are easily acquired from public information or historical data. In addition to social ties and genealogical information, there are countless factors that may cause user-user correlations which make differential privacy vulnerable.

Few studies in the literature pointed out this problem and none of them can provided an actionable and practical method in differentially private data release.

Given the user correlations, the direct method for protecting against the adversaries with knowledge of user correlations involves amplifying the perturbation in terms of *group differential privacy* [58] [21], i.e., protecting the correlated user as a group. However, this technique is not suitable for partial or probabilistic correlation to finely prevent privacy leakage and may over-perturb the data as a result. For example, regardless of whether the probability of the correlations between u_1 and u_2 is 1 or 0.5, it always protects the correlated users in a bundle. Very recently, Liu et al. proposed *dependent differential privacy* (DDP) with *dependence size* L (any tuple in D is dependent on at most $L - 1$ other tuples) and *probabilistic dependence relationship* R (similar to user-user correlations). However, it remains a challenge for data publisher to define L and R .

Although a recent study [59] provide a method for modeling user correlations, the setting of adversaries is still inadequate in practice. Yang et al. proposed a generalization of DP, called Bayesian differential privacy (BDP) [59], to measure the privacy leakage under probabilistic correlation. They described the correlation among users as a connected graph using a proposed *Gaussian correlation model* and assumed that each adversary has full knowledge about such correlation. We argue that the setting of adversaries with *full* knowledge of correlations may not be practical. In reality, the database could be quite large, and an adversary may only know the relationships among a subset of the users. Moreover, the Gaussian correlation model requires a *connected* correlation graph, i.e., any user in the database should have *at least one* correlated user; otherwise, their privacy leakage analysis may not work. From the perspective of data publisher, this technique requires specifying not only the correlation graph but also the disclosed tuples that is known to adversaries so that the privacy leakage can be properly measured. However, in practice, it is hard to know which user's data has been disclosed to the adversaries.

In this chapter, we first explicitly model user correlations using Gaussian Markov Random Field (GMRF), and analyze the extra privacy leakage of a traditional DP mechanism when such GMRF is known by the adversaries. Second, we quantify the extra privacy leakage under different degree of GMRFs and propose mechanisms that prevent the extra privacy leakage. Third, we evaluate the practicality of our approach in differentially private data publishing by testing the impact of different level of user correlations and the runtime of our mechanisms on real-world datasets and synthetic data.

5.2 Preliminaries

5.2.1 Bayesian Differential Privacy

Yang et al. proposed Bayesian differential privacy [59] by extending Pufferfish privacy [53] [54] to explicitly model the adversaries' prior knowledge of the data correlation. They proposed the *Gaussian correlation model*, denoted by $\mathcal{G}$, which is a connected correlation graph including all n users in the database. $A_i(\mathcal{G}, D_{\mathcal{K}})$ indicates an adversary targeting user i with knowledge of the correlation $\mathcal{G}$ and the true value of the tuples $D_{\mathcal{K}}$. Let x_i be the tuple of user i , and let $D_{\mathcal{U}}$ be the unknown tuples in D except $D_{\mathcal{K}} \cup \{x_i\}$. The attack is performed in a Bayesian manner. The adversaries first attempt to infer the unknown tuples $D_{\mathcal{U}}$ based on their knowledge of $D_{\mathcal{K}}$ and on a guessed value of x_i , using the correlation $\mathcal{G}$, and then distinguish D from its neighboring database D' based on the released $\mathbf{r}$.

Definition 16 (ϵ -BDP [59]). *A DP mechanism $\mathcal{M}$ takes as input D and outputs $\mathbf{r}$. Let D' be a copy of D but with the tuple x_i replaced by a distinct value of x'_i . Then, the privacy leakage of $\mathcal{M}$ against A_i is defined as follows.*

$$\begin{aligned} PL(A_i, \mathcal{M}) &\stackrel{\text{def}}{=} \sup_{\mathbf{r}, x_i, x'_i, D_{\mathcal{K}}} \log \frac{\Pr(\mathbf{r}|x_i, D_{\mathcal{K}})}{\Pr(\mathbf{r}|x'_i, D_{\mathcal{K}})} \\ &= \sup_{\mathbf{r}, x_i, x'_i, D_{\mathcal{K}}} \log \frac{\sum_{D_{\mathcal{U}}} \Pr(\mathbf{r}|D) \Pr(D_{\mathcal{U}}|x_i, D_{\mathcal{K}})}{\sum_{D_{\mathcal{U}}} \Pr(\mathbf{r}|D') \Pr(D_{\mathcal{U}}|x'_i, D_{\mathcal{K}})} \end{aligned} \quad (5.1)$$

$\mathcal{M}$ satisfies ϵ -Bayesian Differential Privacy (ϵ -BDP), if for all $A_i(\mathcal{G}, D_{\mathcal{K}}), i \in [1, n]$ it follows that $PL(A_i, \mathcal{M}) \leq \epsilon$.

ϵ -BDP is a generalization of ϵ -DP. If the data are independent, Equation (5.1) is equal to Equation (3.1) because $\Pr(D_{\mathcal{U}}|x_i, D_{\mathcal{K}}) = \Pr(D_{\mathcal{U}}|x'_i, D_{\mathcal{K}}) = \Pr(D_{\mathcal{U}})$. Hence, regardless of the prior knowledge of $D_{\mathcal{K}}$ that the adversaries have, the privacy leakage caused by them is at the same level. In other words, DP guarantees privacy against arbitrary adversaries *without* knowledge of the data correlation. In contrast, if the adversaries know the data correlation, the privacy leakage may increase. For an adversary $A_i(\mathcal{G}, D_{\mathcal{K}})$, they proposed an algorithm with time complexity $O(n^3)$ to compute the *leakage coefficient* ℓ_{A_i} in the following equation.

$$PL(A_i, \mathcal{M}) = \ell_{A_i} * PL_0(\mathcal{M}) \quad (5.2)$$

5.2.2 Problem Setting

We attempt to quantify the potential privacy loss under *continuous aggregate release with DP* (e.g., private counts release as shown in Figure 5.1(c) and (d)). There are n users in the database, and a user only appears at *one* location at most at each time point. We denote the location of user i by a variable x_i . Let l_i^t be the value of x_i at time t . Let $\mathbf{loc} = \{loc_1, \dots, loc_{|\mathbf{loc}|}\}$ be all possible locations in the spatial domain. A trusted server collects the locations of each user into the database $D^t = \{l_1^t, \dots, l_n^t\}$ at each time t (e.g., the columns in Figure 5.1(c)) and calculate the raw aggregate at each location at each time t (e.g., the columns in Figure 5.1(d)). A DP mechanism $\mathcal{M}^t$ releases differentially private output $\mathbf{r}^t$ at each time t . Our goal is to quantify and prevent the potential privacy loss of $\mathcal{M}^t$ against adversary $_{\mathcal{ST}}$ with knowledge of spatial and temporal correlation of the data. We summarize the notations used in this chapter in Table 5.1.

We note that while we use location data as an example, the problem also has other applications such as web mining (e.g. releasing number of users visiting different web pages at each time t). The setting is identical to *differential privacy under continual observation* [23] [25] [28] [29] [74] [30] [75] [31]. In contrast to “one-shot” data release over a static database, the adversaries can observe multiple differentially private outputs, i.e., $\mathbf{r}^1, \dots, \mathbf{r}^t$. There are typically two different privacy goals in the literature: *event-level* and *user-level* [27] [28]. The former protects a single event (i.e., the neighboring databases are D^t and $D^{t'}$), whereas the latter protects all the events of any single user (i.e., the neighboring databases are $\{D^1, \dots, D^t\}$ and $\{D^{1'}, \dots, D^{t'}\}$). In this work, we restrict the discussion to quantifying event-level privacy because it can be seen as a primitive component of user-level privacy.

5.3 Adversary $_{\mathcal{ST}}$

In this section, we formalize the definition of adversary $_{\mathcal{ST}}$. In Section 5.3.1, we study a commonly used models to describe user-user correlation that could be acquired by adversaries. In Section 5.3.2, combining with the defined temporal correlation in Chapter 4, we define the prior knowledge and attack model of adversary $_{\mathcal{ST}}$. We then define ϵ -DP $_{\mathcal{ST}}$, a generalization of ϵ -DP and ϵ -BDP, to prevent the privacy leakage of a DP mechanism against adversary $_{\mathcal{ST}}$.

Table 5.1: Summary of Notations.

$\mathbf{loc}$	The set of locations $\{loc_1, \dots, loc_{ \mathbf{loc} }\}$
x_i	A variable denoting the location of user i , where $i = 1, 2, \dots, n$.
l_i^t	The location of user i at time t
A_i	Adversary who targets x_i in previous study [59]
ℓ_{A_i}	<i>leakage coefficient</i> of adversary A_i
R_i	$Adversaries_{ST}$ who target x_i in this work
G_i	GMRF (Gaussian Markov Random Field) known to R_i
D^t	The database at time t , $\{l_1^t, \dots, l_n^t\}$
$D_{\mathcal{K}}^t, D_{\mathcal{U}}^t$	Two subsets of the database at time t , which are known and unknown to R_i , respectively. $D^t = D_{\mathcal{K}}^t \cup D_{\mathcal{U}}^t \cup \{l_i^t\}$.
m	The size of disclosed tuples $D_{\mathcal{K}}^t, m \in [1, n - 1]$
$\mathcal{M}^t$	Differentially private mechanism over D^t
$\mathbf{r}^t$	Differentially private output at time t

5.3.1 User-user correlation

In the following, we model diverse types of adversaries with different level of knowledge about user-user correlation. We assume that the user-user correlations are given as parameters in our framework.

GMRF for User-User correlation. The user-user correlation in spatiotemporal data indicate the *geographical similarities* between users. In the following, we first clarify our assumption about a user’s spatial pattern (Figure 5.2(a)), and then describe the user-user correlation between two users (Figure 5.2(b)) and multiple users (Figure 5.2(c)). In our setting, we assume that the frequencies of locations that user i may visit, approximately follow a Gaussian distribution (as shown in Figure 5.2(a)). More specifically, we use a Gaussian random variable x_i , denoting the possible location that user i visits, to depict the spatial pattern of user i . Note that the order of these locations on the x-axis of Figure 5.2(a) should be the same for all users to measure the similarity between different users. Then, the user-user correlation between any two users can be described using a *bivariate Gaussian distribution*. For example, a bivariate Gaussian of x_1 and x_2 illustrated in Figure 5.2(b) indicates the geographical similarity between them. The dots are pairs of locations that x_1 and x_2 have visited at the same time, which can

be sampled from users' trajectories such as in Figure 4.2(b). Furthermore, the Gaussian variable vector $[x_1, \dots, x_n]$ forms a multivariate Gaussian distribution, which can be represented by a probabilistic graphical model called the *Gaussian Markov Random Field (GMRF)* [76] [59], illustrated in Figure 5.2(c). The *nodes* are Gaussian random variables (users), and an *edge* indicates that the two connected nodes are (spatially) correlated. The absence of an edge denotes that the two variables are independent. The edges can be weighted to describe the strength of the correlation.

Definition 17 (User-user Correlation). *Let GMRF $\mathbb{G}$ represent the correlation graph including all users in the database. When given the true value of $\{x_i, D_K\}$, the user-user correlation between a set of users D_U and the remaining users $\{x_i, D_K\}$ is described by a joint conditional probability $\Pr(D_U|x_i, D_K)$.*

Note that the modeled user-user correlation is independent of time factor. In other words, we can say that the user-user correlation are the same at every time. We assume the GMRF $\mathbb{G}$ is given (as prior knowledge of the adversaries) in our framework. Learning a GMRF from the data is an independent topic in the research on probabilistic graphical models [77] [76]. A major difference between the correlation graphs $\mathcal{G}$ (see Section 5.2.1) in [59] and $\mathbb{G}$ in this work is that the former is assumed to be *connected*, whereas the latter can be unconnected, as shown in Figure 5.2(c).

5.3.2 Prior Knowledge and Attack Model

In our setting, the adversaries' prior knowledge is tunable in the following two respects.

Prior knowledge of the correlation. There are three combinations of correlation as follows: “only temporal”, “only spatial”, or “both temporal and spatial”. Each correlation can be specified in a flexible manner. For temporal correlation, the transition matrices known to the adversaries may be less than $|\mathbf{loc}|$ in dimension.[†] We designate the probabilities of unknown dimensions to be 0. Let P_i be the temporal correlation known to the adversaries who target user i , namely, $P_i \subseteq \mathbb{P}_i$. For user-user correlation, the GMRF learned by an adversary does not always include *all* users because the correlation may be sparse or the

[†]For example, an adversary learning such an incomplete transition matrix from a segment of the victim's trajectory.

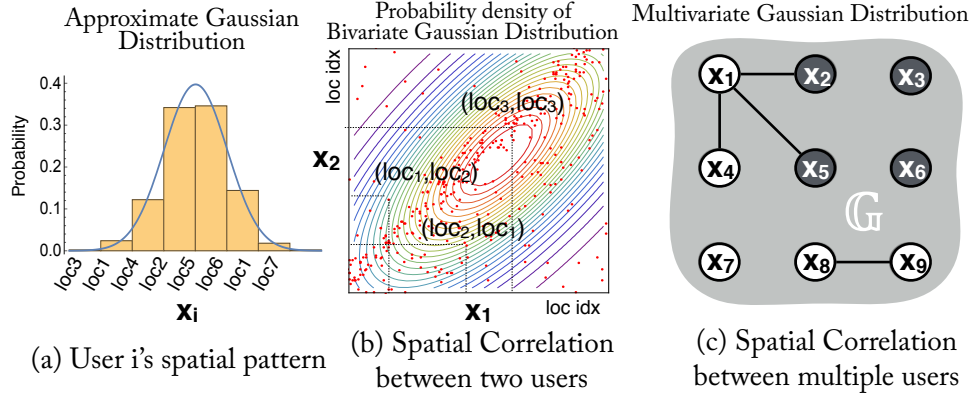


Figure 5.2: Examples of User-user correlation.

adversaries may only learn part of the correlation from incomplete data. Let G_i be the user-user correlation that is known to the adversaries who target user i , namely, $G_i \subseteq \mathbb{G}$.

Prior knowledge of the tuples. The adversaries may have knowledge about the true value of some tuples that we call the *disclosed tuples*. As shown in Definition 16, a DP mechanism can protect against adversaries with an arbitrary number of disclosed tuples; however, this may not be true if the adversaries also know the correlation between the tuples. We use the *number* of the disclosed tuples, denoted by m , as a measure of the adversaries' prior knowledge. We have $m \in [1, n - 1]$ because the “strongest” adversary knows all the tuples except that of the victim. One may wonder why we do not directly specify a set of tuples as the disclosed tuples. The reason is that in practice, it is difficult to determine *which* users are disclosed; however, it is easier to estimate *how many* users are at risk (e.g., from historical data). There are $\binom{n-1}{m}$ such adversaries[‡]. Protecting against all of them is safer than only against one of them.

Definition 18 ($\text{Adversaries}_{\mathcal{ST}}$). *A class of adversaries who target user i and have knowledge about temporal correlation (refer to Chapter 4), user-user correlation and disclosed tuples are called $\text{adversaries}_{\mathcal{ST}}$, denoted by $R_i(P_i, G_i, m)$. Basically, there are three types of $\text{adversary}_{\mathcal{ST}}$: (1) $R_i(P_i, \emptyset, m)$, (2) $R_i(\emptyset, G_i, m)$, (3) $R_i(P_i, G_i, m)$, where $\emptyset$ denotes that the corresponding correlation are not known to the adversaries[§].*

[‡]They may cause different privacy leakages due to the non-uniform distribution of data correlation.

[§]The adversaries of types (1) and (2) will not “guess” the missing correlation; otherwise,

For simplicity, we denote types (1) and (2) as $R_i(P_i, m)$ and $R_i(G_i, m)$, respectively.

Attack model. The attack by adversary $_{\mathcal{ST}}$ is Bayesian in nature. Recall that given the class of adversaries $R_i(P_i, G_i, m)$, there are $\binom{n-1}{m}$ such adversaries with distinct tuple knowledge. For each such an adversary, let $D_{\mathcal{K}}^t \subset D^t$ be the tuple knowledge and $D_{\mathcal{U}}^t$ be the unknown tuples at time t , i.e., $D^t = D_{\mathcal{K}}^t \cup D_{\mathcal{U}}^t \cup \{l_i^t\}$. The adversaries attempt to infer the location of user i at time t . To this end, they first infer $D_{\mathcal{U}}^t$ based on $D_{\mathcal{K}}^t$ and a guessed value of l_i^t and then attempt to distinguish the difference between the two neighboring databases D^t and $D^{t'}$, where $D^{t'}$ is a neighboring database with another guessed value of $\{l_i^{t'}\}$, i.e., $D^{t'} = D_{\mathcal{K}}^t \cup D_{\mathcal{U}}^t \cup \{l_i^{t'}\}$.

Definition 19 (ϵ -DP $_{\mathcal{ST}}$). A DP mechanism $\mathcal{M}^t$ takes as input D^t and outputs $\mathbf{r}^t$ at each time t . The privacy leakage of $\mathcal{M}^t$ against $R_i(P_i, G_i, m)$ is defined as follows.

$$\begin{aligned} PL(R_i, \mathcal{M}^t) &\stackrel{\text{def}}{=} \sup_{\substack{\forall |D_{\mathcal{K}}^t|=m, l_i^t, l_i^{t'} \\ \mathbf{r}^1, \dots, \mathbf{r}^t}} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | l_i^t, D_{\mathcal{K}}^t)}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | l_i^{t'}, D_{\mathcal{K}}^t)} \\ &= \sup_{\substack{\forall |D_{\mathcal{K}}^t|=m, l_i^t, l_i^{t'} \\ \mathbf{r}^1, \dots, \mathbf{r}^t}} \log \frac{\sum_{D_{\mathcal{U}}^t} \Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | D^t) \Pr(D_{\mathcal{U}}^t | l_i^t, D_{\mathcal{K}}^t)}{\sum_{D_{\mathcal{U}}^t} \Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | D^{t'}) \Pr(D_{\mathcal{U}}^t | l_i^{t'}, D_{\mathcal{K}}^t)} \end{aligned} \quad (5.3)$$

$\mathcal{M}^t$ satisfies ϵ -DP $_{\mathcal{ST}}$ (i.e., DP under spatiotemporal correlation) if for all $R_i, i \in [1, n]$ it follows that $PL(R_i, \mathcal{M}^t) \leq \epsilon$.

DP $_{\mathcal{ST}}$ is a generalization of DP and BDP because the adversary $_{\mathcal{ST}}$ is a generalization of the adversary considered in traditional DP and BDP. When both P_i and G_i are $\emptyset$ and m is equal to $n - 1$, R_i is equivalent to the adversary considered in traditional DP. When $P_i = \emptyset$, $G_i = \mathcal{G}$ and $m = |D_{\mathcal{K}}|$ (see Definition 16), a certain adversary in R_i is equivalent to $A_i(\mathcal{G}, D_{\mathcal{K}})$ in BDP.

5.4 Privacy Leakage Analysis

In this section, we analytically measure the privacy leakage w.r.t. different types of adversary $_{\mathcal{ST}}$. We define *Spatial Privacy Leakage (SPL)* w.r.t. user-user correlations they fall under type (3).

lations in Section 5.4.1 (*Temporal Privacy Leakage (TPL)* w.r.t. temporal correlations was defined in Section 4.4.2). We then discuss the composability of TPL and SPL in Section 5.4.2. Finally, we make a few important observations in Section 5.4.3.

5.4.1 Spatial Privacy Leakage (under user-user correlations)

In the following, we consider adversary type $R_i(G_i, m)$ who has knowledge of user-user correlation only (no temporal correlation) and tuple knowledge. We will show that only *partial* prior knowledge affects the privacy leakage, called *Spatial Privacy Leakage*, because the user-user correlation indicates the spatial similarity between users' data.

First, we clarify the attack of $R_i(G_i, m)$. We have $\mathbf{r}^1, \dots, \mathbf{r}^{t-1} \perp\!\!\!\perp D^t, D^{t'}$ because no temporal correlation is available to $R_i(G_i, m)$. Hence, the privacy leakage defined in Equation (5.3) is equal to the following.

Definition 20 (SPL). *A DP mechanism $\mathcal{M}^t$ takes as input D^t and outputs $\mathbf{r}^t$ at each time t . The Spatial Privacy Leakage (SPL) of $\mathcal{M}^t$ w.r.t. $R_i(G_i, m)$ is defined as follows.*

$$\begin{aligned} \text{SPL}(R_i, \mathcal{M}^t) &\stackrel{\text{def}}{=} \sup_{\forall |D_{\mathcal{K}}^t|=m, \mathbf{r}^t, l_i^t, l_i^{t'}} \log \frac{\Pr(\mathbf{r}^t | l_i^t, D_{\mathcal{K}}^t)}{\Pr(\mathbf{r}^t | l_i^{t'}, D_{\mathcal{K}}^t)} \\ &= \sup_{\forall |D_{\mathcal{K}}^t|=m, \mathbf{r}^t, l_i^t, l_i^{t'}} \log \frac{\sum_{D_{\mathcal{U}}^t} \Pr(\mathbf{r}^t | D_{\mathcal{U}}^t) \Pr(D_{\mathcal{U}}^t | l_i^t, D_{\mathcal{K}}^t)}{\sum_{D_{\mathcal{U}}^t} \underbrace{\Pr(\mathbf{r}^t | D_{\mathcal{U}}^{t'})}_{PL_0(\mathcal{M}^t)} \underbrace{\Pr(D_{\mathcal{U}}^t | l_i^{t'}, D_{\mathcal{K}}^t)}_{\text{spatial corr.}}}. \end{aligned} \quad (5.4)$$

According to Equation (5.4), we have the following, in which $\mathcal{S}$ denotes a *spatial privacy loss function*.

$$\text{SPL}(R_i, \mathcal{M}^t) = \mathcal{S}(PL_0(\mathcal{M}^t)) \quad (5.5)$$

If the G_i known to $R_i(G_i, m)$ is empty, $\mathcal{S}(PL_0(\mathcal{M}^t)) = PL_0(\mathcal{M}^t)$ which is the privacy leakage against a traditional adversary (no knowledge of correlation) considered in ϵ -DP.

In the following example, we show that only partial prior knowledge is involved in the calculation of SPL.

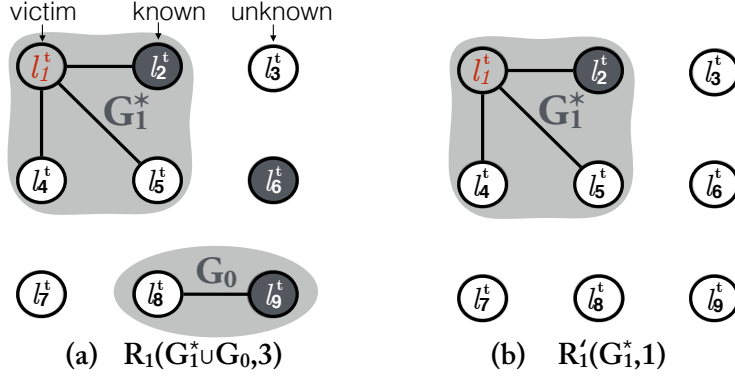


Figure 5.3: The Spatial Privacy Leakage w.r.t. R_1 and R_1' is the same.

Example 6. Figure 5.3 shows two adversaries $\mathcal{S}_T$ both targeting user 1 but with distinct prior knowledge. Compared to the adversary in (b), the one in (a) has more prior knowledge that includes the correlation between users 8 and 9 (i.e., G_0) and the values of l_6^t and l_9^t . However, the privacy leakage w.r.t. the two adversaries is the same because informally their knowledge related to l_1 is the same.

We define G_i^* and $D_{\mathcal{K}}^{t*}$ as follows. If G_i consists of more than one connected components (e.g., G_1^* and G_0 in Figure 5.3(a)) and that G_i^* denotes the one that includes the victim (e.g., G_1^*). The intersection set between $D_{\mathcal{K}}^t$ and the tuples in G_i^* is denoted by $D_{\mathcal{K}}^{t*}$ (e.g., $\{l_2^t\}$ in Figure 5.3(a)).

Lemma 7. For $R_i(G_i, m)$, only G_i^* and $D_{\mathcal{K}}^{t*}$ are involved in the computation of $\Pr(D_{\mathcal{U}}^t | l_i^t, D_{\mathcal{K}}^t)$ and $\Pr(D_{\mathcal{U}}^t | l_i^{t'}, D_{\mathcal{K}}^t)$ in Equation (5.4).

Due to space limitations, we omit the proof here. Intuitively, the lemma holds because only G_i^* and $D_{\mathcal{K}}^{t*}$ have correlation with the target victim.

According to Lemma 7, we only need to focus on G_i^* and $D_{\mathcal{K}}^{t*}$ for calculating SPL. Although G_i can be given arbitrarily, without loss generality, we suppose that G_i is the *connected component* in $\mathbb{G}$ including x_i , namely, $G_i = G_i^*$. Additionally, m is the number of disclosed tuples in G_i^* , namely, $m \in [1, |G_i| - 1]$.

Next, we translate SPL w.r.t. $R_i(G_i, m)$ into the privacy leakage w.r.t. all possible $A_i(\mathcal{G}, D_{\mathcal{K}}^t)$ with $|D_{\mathcal{K}}^t| = m$ (recall that there are $\binom{n-1}{m}$ adversaries A_i with distinct $D_{\mathcal{K}}$ in R_i). A significant difference between R_i and A_i is that the former indicates a class of adversaries, whereas the latter denotes one adversary. Additionally, the Gaussian correlation model $\mathcal{G}$ known to A_i is assumed to

be *connected* (to guarantee that the joint conditional probability distribution remains Gaussian), whereas the correlation graphs G_i known to R_i are arbitrary. In Lemma 7, we prove that even if the correlation graph known to R_i is partial and not connected, the privacy leakage is only related to the connected component G_i that includes victim i , so that SPL under G_i can be calculated by computing $\ell_{A_i(G_i, D_{\mathcal{K}}^t)}$ (see Equation (5.2)) for all $|D_{\mathcal{K}}^t| = m$ (replacing $\mathcal{G}$ with G_i). However, SPL under G_i may not be equal to SPL under $\mathbb{G}$ because the $\mathbf{r}^t$ in Definition 20 is generated from the tuples in $\mathbb{G}$ rather than the tuples in G_i . Theorem 14 proves such equivalence.

Theorem 14. $SPL(R_i, \mathcal{M}^t) = \max_{\forall |D_{\mathcal{K}}^t|=m} \ell_{A_i} * PL_0(\mathcal{M}^t)$ where ℓ_{A_i} is the leakage coefficient[¶] of the adversary A_i .

Proof. Let D^{t*} be the tuples of G_i (recall that G_i is a connected correlation graph containing user i). Let $\mathbf{r}^{t*}$ be the released data that are generated from D^{t*} . Denote $R_i^0(G_i, D_{\mathcal{K}}^t)$ where $|D_{\mathcal{K}}^t| = m$ as an adversary in $R_i(G_i, m)$. According to the definition of adversary_{ST}, R_i^0 is the same as $A_i(G_i, D_{\mathcal{K}}^t)$ but only by observing $\mathbf{r}^{t*}$ instead of $\mathbf{r}^t$. By Lemma 7, we know that even if R_i^0 has extra knowledge of the raw data $D^t - D^{t*}$, the privacy leakage w.r.t. R_i^0 remains the same. In other words, R_i^0 can derive $\mathbf{r}^t$ combining the raw data $D^t - D^{t*}$ with $\mathbf{r}^{t*}$. Hence, R_i^0 is equivalent to $A_i(G_i, D_{\mathcal{K}}^t)$ where $|D_{\mathcal{K}}^t| = m$. According to Equation (5.2), the privacy leakage caused by A_i is $\ell_{A_i} * PL_0(\mathcal{M}^t)$. The theorem follows because SPL is defined as a supremum. $\square$

Now, we present the properties of SPL. According to the result in [59], a weaker adversary (smaller $|D_{\mathcal{K}}^t|$) poses the same or a larger privacy leakage^{||}, and $\ell_{A_i(G_i, D_{\mathcal{K}}^t)}$ is less than or equal to $|G_i| - |D_{\mathcal{K}}^t|$.

Corollary 3. *The SPL w.r.t. $R_i(G_i, m)$ is greater than or equal to the SPL w.r.t. $R_i(G_i, m + 1)$.*

Corollary 4. *The SPL of $\mathcal{M}^t$ w.r.t. $R_i(G_i, m)$ is less than or equal to $(|G_i| - m) * PL_0(\mathcal{M}^t)$.*

[¶]Yang et al. [59] proposed an $O(n^3)$ algorithm to compute ℓ_{A_i} . We see that algorithm as a black-box.

^{||}This implies that a stronger adversary (with higher prior) has less information gain than a weaker adversary (with lower prior).

The $|G_i| - m$ in Corollary 4 is the size of the unknown tuples (including the victim). It indicates that the sensitivity is $|G_i| - m$ times as large as the case of no spatial correlation.

Remark 2. *The lower and upper bound of SPL are $PL_0(\mathcal{M}^t)$ and $(|G_i| - m) * PL_0(\mathcal{M}^t)$ respectively. Hence, the spatial privacy loss function $\mathcal{S}$ satisfies that $x \leq \mathcal{S}(x) \leq (|G_i| - m) * x$, where the variable x is $PL_0(\mathcal{M}^t)$.*

The exact calculation of SPL is computationally intractable because $\binom{|G_i|-1}{m}$ adversaries A_i with distinct $D_{\mathcal{K}}^t$ exist in R_i and we need to compute ℓ_{A_i} of them. To obtain a tradeoff between runtime and utility, we design a polynomial-time algorithm to return the approximate SPL in Section 5.5.1.

5.4.2 Composability

In this section, we prove that P_i and G_i are independent of each other when computing privacy leakage w.r.t. $R_i(P_i, G_i, m)$ so that they are composable for quantifying the privacy leakage.

Lemma 8. *Lemmas 5 and 6 remains true when $R_i(P_i, m)$ obtains spatial correlation G_i .*

Proof. We have $\Pr(D^{t-1}|D^t) = \Pr(D_{\mathcal{U}}^{t-1}, l_i^{t-1}, D_{\mathcal{K}}^{t-1}|D_{\mathcal{U}}^t, l_i^t, D_{\mathcal{K}}^t)$. Since G_i only gives $\Pr(D_{\mathcal{U}}^{t-1}|l_i^{t-1}, D_{\mathcal{K}}^{t-1})$, we still have $D_{\mathcal{U}}^{t-1} \perp\!\!\!\perp D_{\mathcal{U}}^t$ and $D_{\mathcal{K}}^{t-1} \perp\!\!\!\perp D_{\mathcal{K}}^t$. Hence, $\Pr(D^{t-1}|D^t) = \Pr(l_i^{t-1}|l_i^t)$. $\square$

In other words, the temporal privacy loss function $\mathcal{T}$ will be same for $R_i(P_i, m)$ and $R_i(P_i, G_i, m)$.

Lemma 9. *Lemma 7 remains true when $R_i(G_i, m)$ obtains temporal correlation P_i .*

Proof. We only need to prove that the spatial correlation $\Pr(D_{\mathcal{U}}^t|l_i^t, D_{\mathcal{K}}^t)$ will not change when given temporal correlation $\Pr(l_i^{t-1}|l_i^t)$ or $\Pr(l_i^t|l_i^{t+1})$. This is true because l_i^t is given in the conditional part of $\Pr(D_{\mathcal{U}}^t|l_i^t, D_{\mathcal{K}}^t)$. $\square$

In other words, the spatial privacy loss function $\mathcal{S}$ will be same for $R_i(G_i, m)$ and $R_i(P_i, G_i, m)$.

Theorem 15. *The privacy leakage w.r.t. $R_i(P_i, G_i, m)$ is*

$$PL(R_i, \mathcal{M}^t) = \begin{cases} \mathcal{S}(PL_0(\mathcal{M}^1)) & t = 1 \\ \mathcal{T}(PL(R_i, \mathcal{M}^{t-1})) + \mathcal{S}(PL_0(\mathcal{M}^t)) & t > 1 \end{cases}$$

Proof. We expand Equation (5.3) as follows.

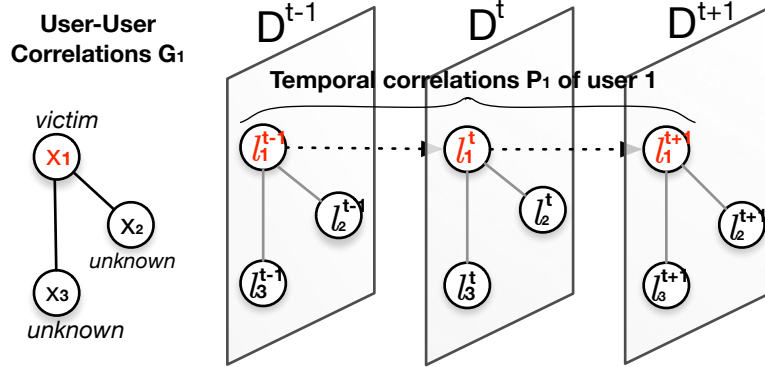
$$\begin{aligned} & \underbrace{\sup_{\substack{\forall |D_{\mathcal{K}}^{t-1}|=m, \\ \mathbf{r}^1, \dots, \mathbf{r}^t, l_i^t, l_i^{t'}}} \log \frac{\sum_{l_i^{t-1}, D_{\mathcal{K}}^{t-1}} \Pr(\mathbf{r}^1, \dots, \mathbf{r}^{t-1} | l_i^{t-1}, D_{\mathcal{K}}^{t-1}) \Pr(l_i^{t-1} | l_i^t)}{\sum_{l_i^{t-1'}, D_{\mathcal{K}}^{t-1}} \underbrace{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^{t-1} | l_i^{t-1'}, D_{\mathcal{K}}^{t-1})}_{PL(R_i, \mathcal{M}^{t-1})} \underbrace{\Pr(l_i^{t-1'} | l_i^{t'})}_{\text{temporal corr.}}}}_{\mathcal{T}(PL(R_i, \mathcal{M}^{t-1}))} \\ & + \underbrace{\sup_{\forall |D_{\mathcal{K}}^t|=m, \mathbf{r}^t, l_i^t, l_i^{t'}} \log \frac{\Pr(\mathbf{r}^t | l_i^t, D_{\mathcal{K}}^t)}{\Pr(\mathbf{r}^t | l_i^{t'}, D_{\mathcal{K}}^t)}}_{\mathcal{S}(PL_0(\mathcal{M}^t))} \end{aligned} \quad (5.6)$$

The above equation can be interpreted as the case of $R_i(P_i, m)$ with extra knowledge of G_i ; hence, we can obtain it by replacing $PL_0(\mathcal{M}^t)$ with $\mathcal{S}(PL_0(\mathcal{M}^t))$ and replacing $TPL(R_i, \mathcal{M}^{t-1})$ with $PL(R_i, \mathcal{M}^{t-1})$. When $t = 1$, it is clear that $PL(R_i, \mathcal{M}^1) = \mathcal{S}(PL_0(\mathcal{M}^1))$. When $t > 1$, we have $PL(R_i, \mathcal{M}^t) = \mathcal{T}(PL(R_i, \mathcal{M}^{t-1})) + \mathcal{S}(PL_0(\mathcal{M}^t))$. $\square$

Example 7. *We demonstrate the privacy leakage w.r.t.*

$R_1(P_1, G_1, 0)$ (as shown in Figure 5.4). Suppose that the temporal correlation P_1 results in $\mathcal{T}(x) = 1 * x$ (see Remark 1) and that the spatial correlation G_1 leads to $\mathcal{S}(x) = 3 * x$ (see Remark 2). If $\mathcal{M}^t$ adds $\text{Lap}(1/\epsilon)$ noises to the aggregates of each time point, we have $PL_0(\mathcal{M}^1) = \epsilon$. The privacy leakage w.r.t. R_1 over time is listed in the following table.

$t =$	1	2	3	4	...
$\mathcal{T}(PL(R_i, \mathcal{M}^{t-1}))$	—	3ϵ	6ϵ	9ϵ	...
$\mathcal{S}(PL_0(\mathcal{M}^t))$	3ϵ	3ϵ	3ϵ	3ϵ	...
$PL(R_i, \mathcal{M}^t)$	3ϵ	6ϵ	9ϵ	12ϵ	...


 Figure 5.4: $\text{Adversary}_{ST} R_1(P_1, G_1, 0)$.

5.4.3 Discussion

We make a few important observations regarding our privacy analysis. First, our contributions in this work can serve as primitives for quantifying the privacy risk under more sophisticated adversarial knowledge. For example, the temporal correlation P_i can be n -th order Markov model. Then, we need to slightly modify Lemmas 5, 6 and 8 for quantifying privacy leakage under this situation. Although (for simplicity) we assume G_i and m are fixed at each time, our framework can also handle different P_i , G_i and m at different time points; then, Lemma 8 and 9 need some modifications. On the other hand, the privacy leakage also has an effect on the adversarial knowledge. It is an interesting future research question to quantify how P_i , G_i and m will change over time due to the leakage.

Second, we assume that the correlation P_i and G_i are given as *parameters* in our framework. In practice, a data publisher can learn these correlation from the raw data using standard methods (see Section 5.3.1) or from public information such as the common pattern of human activities [56], or weather conditions [57]. We note that, given a standard adversary who does not have knowledge of the data correlation, using these correlation in the data publishing process may leak these correlation to them and they will eventually evolve to adversaries_{ST} . However, the privacy leakage w.r.t. such evolved standard adversary, namely, adversary_{ST} , is still bounded by the quantified value.

5.5 Algorithms

We first design efficient algorithms for computing SPL in Sections 5.5.1 respectively. We then present a framework for achieving ϵ - $DP_{\mathcal{ST}}$ preventing TPL and SPL against adversary $_{\mathcal{ST}}$ in Section 5.5.2.

5.5.1 Quantifying SPL

In the following, we propose three methods to find SPL. BFS returns exact SPL but suffers from an impractical computation time. NoSearch and PS return approximate SPL.

Procedure 10: FindSPL: BFS (Brute Force Search)

Input: $G_i; m; \mathcal{M}^t$.
Output: SPL w.r.t $R_i(G_i, m)$

```

1  $SPL_i \leftarrow 0$ 
   //  $D_{G_i}^t$  denotes the tuples in  $G_i$ 
2 forall the  $D_K^t \subset D_{G_i}^t, i \notin D_K^t$  and  $|D_K^t| = m$  do
3   if  $SPL_i < \text{privacy leakage of } A_i(G_i, D_K^t)$  then
4      $SPL_i \leftarrow \text{privacy leakage of } A_i(G_i, D_K^t)$            //by Theorem 14
4 return  $SPL_i$ 
    
```

Brute Force Search. We first present a Brute Force Search (BFS) method for quantifying SPL. According to Theorem 14, we need to check the privacy leakage w.r.t. $\binom{|G_i|-1}{m}$ possible adversaries. We can employ the algorithm in [59] (seen as a black-box) as a sub-procedure to compute each privacy leakage w.r.t. A_i in time $O(n^3)$. BFS is shown in Procedure 10, and its time complexity is $O(\binom{|G_i|-1}{m} * n^3)$.

NoSearch. Then, we show Procedure 11, which quantifies SPL without searching (NoSearch). It directly returns the upper bound of SPL w.r.t. $R_i(P_i, G_i, m)$ according to Corollary 4. Obviously, this may lead to large perturbations.

Procedure 11: FindSPL: NoSearch

Input: $G_i; m; \mathcal{M}^t$.
Output: SPL w.r.t $R_i(G_i, m)$

```

1 return  $SPL_i \leftarrow (|G_i| - m) * PL_0(\mathcal{M}^t)$            //by Corollary 4
    
```

Pruning Search. To obtain a better tradeoff between runtime and utility, we design Pruning Search (PS) as shown in Procedure 12. We organize the

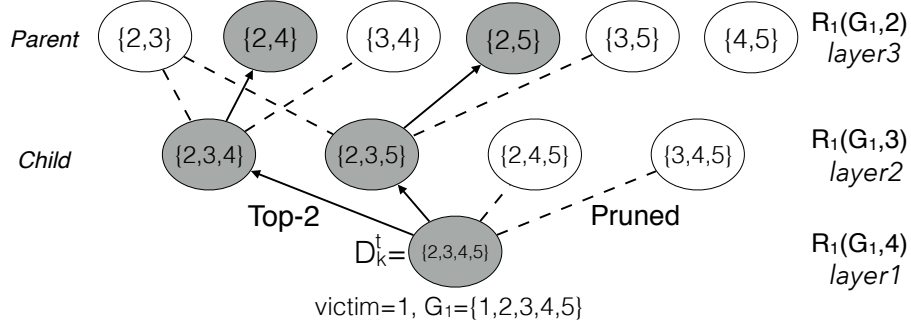


Figure 5.5: Example of a Pruning Search for $R_1(G_1, 2)$.

adversaries into a lattice as shown in Figure 5.5. The nodes indicate adversaries $A_i(G_i, D_K^t)$ with distinct D_K^t . The adversaries on the same layer have the same $|D_K^t|$, and those on the upper layer are “parents” of those on the lower layer called “children” if they differ in exactly one tuple. Hence, the goal of FindSPL is to retrieve the adversary who poses the largest privacy leakage on a targeted layer. Instead of brute-force searching (BFS) all adversaries on the targeted layer, we adopt a bottom-up search that starts from the adversary on the bottom layer and ends at the targeted layer. Our idea is that if A_i and A'_i are on the same layer and A_i poses a larger privacy leakage than A'_i , then the parents/children of A_i are more likely to pose larger privacy leakage than those of A'_i (hence, we may prune A'_i). In other words, “like father, like son”.

Our heuristic pruning strategy is inspired by Corollary 3. Corollary 3 tells us that the adversaries on the upper layer produce a larger privacy leakage than those on the lower layer, i.e., the parents cause more risk. When an adversary obtains a new tuple (becomes a “parent”), the privacy leakage either increases or remains the same (Corollary 3). This monotonic property suggests that the “larger” adversaries most likely have larger parents/children, though this is not always true. If A'_i is pruned and their parent is the largest, PS may return an incorrect SPL (i.e., not the supremum). To obtain a tradeoff between the runtime and utility, we provide a tunable parameter k (in Line 11) to control the pruning rate. During the bottom-up search, only the top- k largest adversaries are preserved and others are pruned. When setting k as $\binom{|G_i|-1}{m}$, PS is equivalent to BFS because no adversary is pruned. When setting k as a smaller number, say $|G_i|$, the computation time of PS will be much less than BFS; however, the search will be prone to error.

Example 8. The goal is to find the SPL w.r.t. $R_1(G_1, 2)$, as shown in Figure 5.5. For convenience, we use $A_{i,j}$ to denote an adversary with $D_K^t = \{i, j\}$. We adopt a bottom-up search that starts from the adversary on layer 1 who knows all the tuples except the victim. We compute the privacy leakage of their parents and select the top-2, say, $A_{2,3,4}, A_{2,3,5}$. Next, based on the selected adversaries on layer 2, we choose the top-2 adversaries on layer 3 among their parents, i.e., $A_{2,4}, A_{2,5}$. Finally, we return the max privacy leakage between them as SPL_1 w.r.t. $R_1(G_1, 2)$.

The above example describes a strategy of pruning parents from the bottom to the top. Similarly, we can prune children in a *top-down* fashion if m is greater than $\frac{|G_i|}{2}$ because it searches over fewer layers than that encountered when pruning in a bottom-up way. We present PS, a two-way pruning search, in Procedure 12. The time complexity is $O(\frac{|G_i|}{2} * k * n^3)$ because the calculation traverses $\frac{|G_i|}{2}$ layers in the worst case.

Procedure 12: FindSPL: PS (Pruning Search)

Input: $G_i; m; \mathcal{M}^t$.
Output: SPL w.r.t $R_i(G_i, m)$

- 1 **if** $m \leq \frac{|G_i|}{2}$ **then** $start \leftarrow 1$ **else** $start \leftarrow (|G_i| - 1)$
- 2 Initialize Adv_s as a set of A_i on layer s
- 3 **for** layer $l = start$ **to** m **do**
- 4 Set $NextAdv_s \leftarrow \emptyset$
- 5 **foreach** $A_i \in Adv_s$ **do**
- 6 **if** $start = 1$ **then**
- 7 $NextAdv_s \leftarrow Adv_s \cup \{\text{parents of } A_i\}$
- 8 **else**
- 9 $NextAdv_s \leftarrow Adv_s \cup \{\text{children of } A_i\}$
- 10 sort $NextAdv_s$ by *privacy leakage of* A_i
- 11 $Adv_s \leftarrow \text{top-}k \in NextAdv_s$ //heuristic pruning
- 12 **return** max *privacy leakage w.r.t.* $A_i, A_i \in Adv_s$

5.5.2 Achieving ϵ - DP_{ST} against TPL and SPL

We present Algorithm 13 for achieving ϵ - DP_{ST} by calibrating the privacy budget of a DP mechanism $\mathcal{M}^t$. Because traditional DP mechanisms use a single pri-

vacy budget to denote the privacy of all users**, we must prevent the maximum TPL and SPL for all users. Based on *FindTPL* and *FindSPL*, we calculate an appropriate privacy budget for a traditional DP mechanism to prevent privacy leakage against adversary $_{\mathcal{ST}}$. In the following, $\mathcal{M}_O$ and $\mathcal{M}_I$ are 0-DP and 1-DP mechanisms: $PL_0(\mathcal{M}_O) = 0$ and $PL_0(\mathcal{M}_I) = 1$.

Algorithm 13: PREVENT TPL AND SPL

Input: ϵ (expected privacy level); $P_i, i \in [1, n]$; $G_i, i \in [1, n]$; m ; ϵ_{t-1}
 (privacy leakage at time $t - 1$).
Output: Privacy budget of $\mathcal{M}^t$

```

1 foreach  $P_i, i \in [1, n]$  do
2    $TPL_i \leftarrow FindTPL(P_i, \epsilon_{t-1}, \mathcal{M}_O)$            //FindTPL: Cascading
3  $maxTPL \leftarrow \max\{TPL_i, i \in [1, n]\}$ 
4 foreach  $G_i, i \in [1, n]$  do
5    $SPL_i \leftarrow FindSPL(G_i, m, \mathcal{M}_I)$            //FindSPL: BFS, NoSearch or PS
6  $maxSPL \leftarrow \max\{SPL_i, i \in [1, n]\}$ 
7 return  $(\epsilon - maxTPL)/maxSPL$                        //by Theorem 15
```

Theorem 16. *Algorithm 13 satisfies ϵ -DP $_{\mathcal{ST}}$ when $maxTPL$ (Line 3) and $maxSPL$ (Line 6) are greater than or equal to the supremum of TPL and SPL respectively.*

Proof. We only need to prove that the privacy leakage of a traditional DP mechanism $\mathcal{M}^t$ with budget $\frac{\epsilon - maxTPL}{maxSPL}$ (Line 7) is less than or equal to ϵ w.r.t. R_i , i.e., $PL(R_i, \mathcal{M}^t) \leq \epsilon$. According to Theorem 15, $PL(R_i, \mathcal{M}^t) = \mathcal{T}(\epsilon_{t-1}) + \mathcal{S}(PL_0(\mathcal{M}^t))$ where ϵ_{t-1} is the previous privacy leakage at time $t - 1$ and $PL_0(\mathcal{M}^t) = \frac{\epsilon - maxTPL}{maxSPL}$. Now, we analyze $\mathcal{T}$ and $\mathcal{S}$. First, because each TPL_i is calculated based on $\mathcal{M}_O$ (Line 2) and because $maxTPL$ is greater than or equal to each TPL_i , we have $maxTPL \geq \mathcal{T}(\epsilon_{t-1}) + PL_0(\mathcal{M}_O)$; hence, $\mathcal{T}(\epsilon_{t-1}) \leq maxTPL$. Second, because each SPL_i is calculated based on $\mathcal{M}_I$ (Line 5) and because $maxSPL$ is greater than or equal to each SPL_i , we have $maxSPL \geq \mathcal{S}(PL_0(1))$ by Equation (5.5); then, we have $\mathcal{S}(PL_0(\mathcal{M}^t)) \leq maxSPL * \frac{\epsilon - maxTPL}{maxSPL}$. Therefore,

$$PL(R_i, \mathcal{M}^t) = \mathcal{T}(\epsilon_{t-1}) + \mathcal{S}(PL_0(\mathcal{M}^t)) \leq maxTPL + maxSPL * \frac{\epsilon - maxTPL}{maxSPL} \leq \epsilon. \quad \square$$

For *FindSPL*, the PS may not exactly satisfy ϵ -DP $_{\mathcal{ST}}$ because PS may return a value that is less than the supremum of SPL. We will examine the accuracy of PS in Section 5.6.1.

**Our methods are compatible with recently proposed personalized differentially private mechanisms [66].

Overall time complexity. The proposed algorithm includes **PS** with $O(|G| * k * n^3)$ runtime for calculating SPL and **Cascading** with $O(|\text{loc}|^2)$ runtime for calculating TPL. Note that **PS** only needs to run *once* at time $t = 1$ to determine the coefficient of the spatial privacy loss function $\mathcal{S}$. The $\mathcal{S}$ does not change over time because there is no time factor involved in the spatial correlation G_i . In contrast, the **Cascading** for computing TPL needs to run at each time point $t > 1$ because it involves solving a linear-fractional programming program whose constraints change over time (see Equation (4.18)). Hence, when the data are published continuously over a long period of time, the overall time complexity is dominated by **Cascading**, which is capable of handling large datasets.

5.6 Experimental Evaluation

In this section, we design experiments for the following: (1) verifying the runtime and accuracy of heuristic algorithm **PS** for computing SPL compared with **BFS** and (2) investigating the impact of the temporal and spatial correlation on privacy leakage. We omit the present the experimental verification of **Cascading** (for computing TPL) due to space limitations (the correctness has been theoretically proven in Theorem 12). We implemented all the algorithms in Java and conducted the experiments on a machine with an Intel Core i7 2.8GHz CPU and 16 GB RAM running OSX El Capitan.

The setting of temporal and spatial correlation. To evaluate if our privacy loss quantification algorithms can perform well under diverse circumstances, we need different degrees of temporal and spatial correlation. We will not *estimate* the correlation from data. Instead, we *generate* the correlation to eliminate the effect of different correlation estimation algorithms and different datasets.

Spatial correlation. We generate spatial correlation by correlation graphs^{††} (see Figure 5.2(c)) from both the real-world dataset Brightkite^{‡‡} and synthetic graphs. Brightkite was introduced in [78], which shows that users' friendships may affect their mobility patterns, e.g., people more likely appear where their friends have been. Their findings also contribute to our motivations of the spatial correlation in Example 1. Brightkite contains 4,491,143 check-ins along with user IDs and timestamps and includes geo-social networks with 58,228 nodes (users)

^{††}The GMRF can be constructed from a Laplacian matrix of the correlation graphs [59].

^{‡‡} <http://snap.stanford.edu/data/loc-brightkite.html>

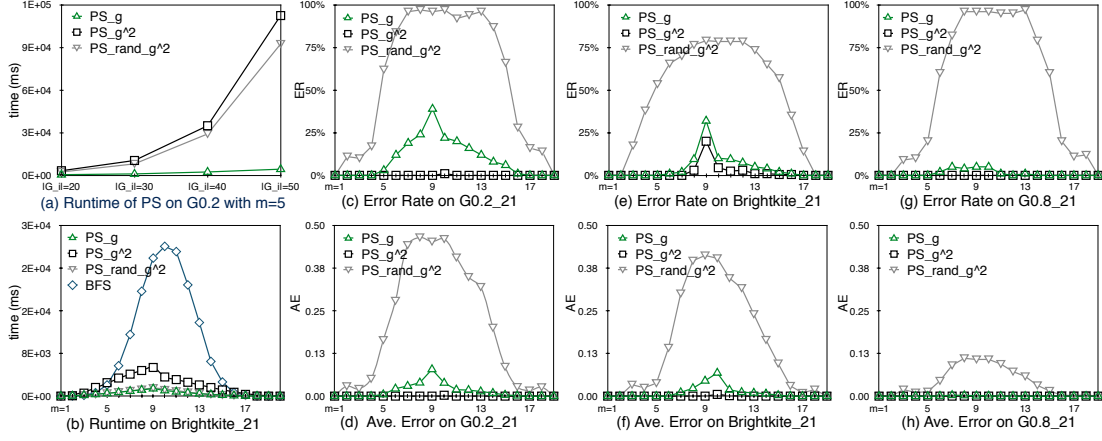


Figure 5.6: Runtime and Accuracy of the Pruning Search.

and 214,078 edges (friendships). We sampled friendship networks as follows. From 281 users who have 20 friends, we randomly choose 100 users and their friendship networks, i.e., 100 correlation graphs with 21 nodes. We call this dataset `Brightkite_21`. Similarly, we choose all 36 users who have 49 friends for `Brightkite_50`. On the other hand, to better understand the influence of different correlation graphs, we also generate random graphs with a controlled density parameter ρ , which denotes the probability of the existence of an edge between any two nodes. We denote a generated dataset that includes 100 graphs with $\rho = 0.2$ and 21 nodes in each graph as `G0.2_21`, and we denote a dataset with the same properties but with $\rho = 0.8$ as `G0.8_21`.

5.6.1 Evaluation of the Pruning Search

We describe the setup for testing the runtime and accuracy of PS (Pruning Search) compared with BFS (Brute Force Search). We examined different strategies for pruning. In algorithms `PS_g` and `PS_g2`, the pruning parameter k is set as $|G_i|$ and $|G_i|^2$, respectively. Algorithm `PS_rand_g2` also has $k = |G_i|^2$; however, it randomly chooses the parents or children instead of the top- k . The results shown in Figure 5.6 confirm the effectiveness of our heuristic pruning strategy.

We present the following factors affecting the runtime, as shown in Figure 5.6 (a) and (b).

Runtime vs. $|G_i|$. In Figure 5.6(a), we show the runtime of PS with different

k on datasets G0.2 and $m = 5$. The run time increases with $|G_i|$ because a larger $|G_i|$ results in more adversaries (i.e., $\binom{|G_i|-1}{m}$). The figure shows that the pruning parameter k significantly affects the runtime scalability of PS. Algorithms PS_g and PS_g² denote PS with $k = |G_i|$ and $|G_i|^2$, respectively. PS_g scales well with runtime increasing linearly with $|G_i|$, whereas the run time of PS_g² is increasing exponentially. PS_g² has slightly higher runtime than PS_rand_g² because the former incorporates top- k queries, whereas the latter makes randomly selection.

Runtime vs. m . Both PS and BFS present a pattern of first increasing and then decreasing with increasing m . For PS, this occurs because of the two-way pruning strategy. For BFS, this occurs because of the symmetry of $\binom{|G_i|-1}{m}$. In Figure 5.6(b), when m ranges from 1 to 10, the runtime of PS is larger than that of the other half because the computation time [59] for the leakage coefficient (i.e., ℓ_{A_i}) of a *single* adversary is proportional to $|G_i| - m$.

Runtime vs. k . Figure 5.6(a) and (b) exhibit the obvious pattern that the computation time is proportional to k . The runtimes of two algorithms with $k = |G_i|^2$ are the same and are several orders of magnitude larger than that for $k = |G_i|$.

Next, we discuss the accuracy of PS measured by the Error Rate (ER) and Average Error(AE) as follows.

$$\begin{aligned} ER &= \frac{\text{COUNT}(SPL_{\text{BFS}} > SPL_{\text{PS}})}{\text{runtime}} \\ AE &= \frac{\sum_{t=1}^{\text{runtime}} (SPL_{\text{BFS}} - SPL_{\text{PS}})}{\text{runtime}} \end{aligned}$$

NoSearch incurred an ER of almost 100% and an AE in the range of $0.4 \sim 11.9$ on G0.8_21; thus, we do not include them in the figures for better readability. We present the following results running on 100 graphs.

Accuracy vs. ρ . We found that the accuracy of PS is susceptible to the density ρ of the correlation. In Figure 5.6 (c), (e), (g) and (d), (f), (h), the errors for the three datasets are decreasing from left to right (the average density of Brightkite_21 is approximately 0.6). This is because, for the same m , the privacy leakage in the sparse graphs tends to be greater than that for the dense graphs (we will present evidence of this in Section 5.6.2). This means that, for sparse graphs, the SPL on each layer will increase more rapidly from the bottom to top such that PS is prone to high errors.

Accuracy vs. m . The error reaches its peak when $m = |G_i|/2$ because the

number of adversaries is highest compared to the other layers. Another reason is that the error of PS will accumulate from the start layer.

Accuracy vs. k . The result shows that our heuristic pruning strategy is effective. The top- k pruning algorithms significantly outperform the random pruning of PS_rand_g². Additionally, the accuracy will increase when k increases because of fewer adversaries being pruned.

5.6.2 Impact of correlation on Privacy Loss

In this section, we evaluate TPL and SPL separately to better show the impact of different parameters related to spatial and temporal correlation.

We examined s values in Equation (4.24) ranging from 0.005 to 1. We set $|\mathbf{loc}|$ to 50 and 200. Let ε be the privacy budget of $\mathcal{M}^t$ at each time point. We test $\varepsilon = 1$ and 0.1. The results are shown in Figure 4.6 and are summarized as follows.

TPL vs. $|\mathbf{loc}|$. Under the same s , TPL is smaller when $|\mathbf{loc}|$ (dimension of the transition matrix) is larger, as shown in the lines $s = 0.005$ with $|\mathbf{loc}| = 50$ and $|\mathbf{loc}| = 200$ of Figure 4.6. This is because the transition matrices tend to be uniform (weaker correlation) when the dimension is larger.

In the following, we present the impact of spatial correlation (correlation graphs) on SPL. Because SPL is directly proportional to $PL_0(\mathcal{M})$ (see Theorem 14), i.e., ε , we do not examine different ε and set it as 1. We also know SPL increases when m increase according to Corollary 3. Hence, we only examine the impact of the density ρ and size $|G_i|$ of correlation graphs on SPL. Due to the impractical computational time of BFS, we use PS with $k = |G_i|^2$ to compute the approximate SPL.

SPL vs. ρ . Figure 5.7(a) shows that, for the same m , the privacy leakage on a sparser graphs tend to be higher. This can be explained by Remark 2, which tells us that the privacy leakage reaches the upper bound if the sensitivity is $|G_i| - m$, e.g., the $|G_i| - m$ tuples that are unknown to adversaries are correlated and not affected by (no edge to) other m tuples. When the correlation graph is sparse, this can be observed easily.

SPL vs. $|G_i|$. Figure 5.7(b) shows that, for the same m , the privacy leakage on larger correlation graphs tends to be higher because, when $|G_i|$ is larger, there are more types of adversaries (because of $\binom{|G_i|}{m}$) and a greater chance to find a $D_{\mathcal{K}}^t$ that results in the upper bound of SPL (recall that we define SPL as a supremum

of privacy leakage w.r.t. all those adversaries whose prior knowledge $|D_K^t| = m$).

In conclusion, the experiments reveal that our quantification algorithms can flexibly respond to different degrees of temporal and spatial correlation.

5.7 Related Work

Several studies have questioned whether differential privacy is valid for correlated data. Kifer and Machanavajjhala [46] [53] [54] first raised the important issue that differential privacy may not guarantee privacy if adversaries know the data correlation. In their line of work, they [46] argued that it is not possible to ensure any utility in addition to privacy without making assumptions about the data-generating distribution and the background knowledge available to an adversary. To this end, they proposed a general and customizable privacy framework called *PufferFish*, in which the potential secrets, discriminative pairs, and data generation need to be explicitly defined. Yang et al. [59] further investigated differential privacy on correlated tuples described using a proposed Gaussian correlation model. The privacy leakage w.r.t. adversaries with specified prior knowledge can be efficiently computed. The major difference between spatial correlation $\mathbb{G}$ in this work and Gaussian correlation model $\mathcal{G}$ in [59] is that the latter requires a connected graph (i.e., any user in the database should have at least one correlated user), whereas the $\mathbb{G}$ is not necessarily connected (and is thus more practical).

Zhu et al. [70] proposed correlated differential privacy by redefining the sensitivity of queries on correlated data; however, the privacy guarantee provided by this definition for spatio-temporal data is unclear. Very recently, Liu et al. [60] proposed dependent differential privacy by introducing dependence coefficients for analyzing the sensitivity of different queries under probabilistic dependences between tuples. However, such dependence coefficients do not easily account for the spatio-temporal correlation.

Dwork et al. first studied *differential privacy under continual observation* and proposed event-level/user-level privacy [27] [28]. The previous studies in this setting focused on the problems of high dimension [23] [71] [31], infinite sequence [72] [26] [30], data sparsity [71], sliding window queries [73], and real-time publishing [29]. However, none of them addressed the problem of increased privacy loss under temporal and spatial correlation.

To the best of our knowledge, no study has reported the risk of differential

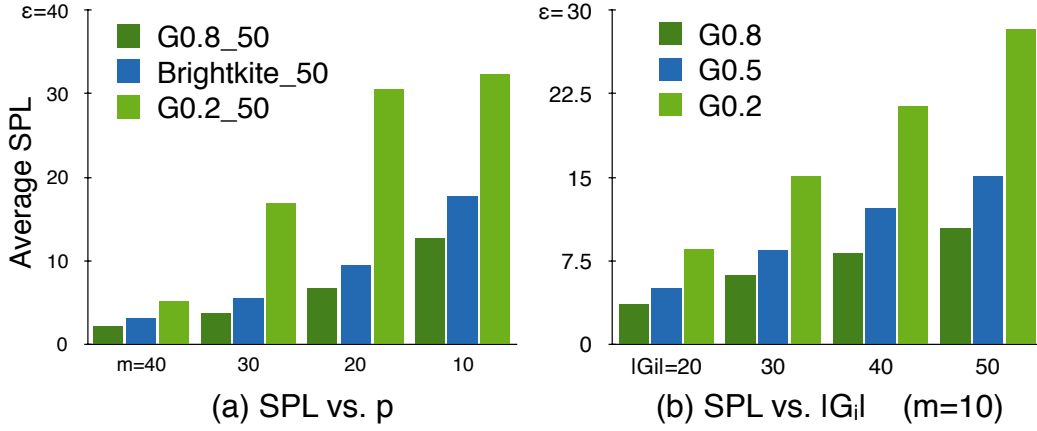


Figure 5.7: Evaluation of the SPL (on 100 graphs).

privacy under temporal and spatial correlation. Although a few studies [61] [63] have considered a similar adversarial model in which the adversaries have prior knowledge of temporal correlation represented by Markov chains, they focused on location privacy in the single-user setting. Shokri et al. [61] proposed an evaluation framework for location privacy protection, assuming that the adversary knows the transition probabilities of each user. Xiao et al. [63] proposed a mechanism for single user location sharing with differential privacy under temporal correlation with Markov chains. In contrast, the scenario in this chapter focuses quantifying the privacy loss of traditional DP mechanisms under temporal and spatial correlation.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this work, we explored the potential of flexible and rigorous privacy models in spatiotemporal data publishing. In our privacy protection framework for spatiotemporal data, we proposed novel privacy models that tackle with the fundamental drawbacks of differential privacy and designed novel mechanisms that be able to release infinite streaming data and correlated data. The contributions of this dissertation are summarized below:

- We provided a formal definition of ℓ -trajectory privacy as a personalized and customizable model. It is a user-level privacy model which allows different users to specify their own ℓ_u , even varying ℓ_u at different times. It can be easily applied to other time-sequenced data, such as medical records or health-related streaming data.
- We proposed an algorithmic framework for publishing ℓ -trajectory private statistics in real time. Algorithms GA+Adj and GA+MMD implemented the framework to achieve ℓ -trajectory privacy in a data-dependent way.
- We conducted a formal analysis that the privacy of a traditional DP mechanism may degrade under temporal or user-user correlations. An efficient algorithm is proposed to precisely measure such extra privacy loss.

- We provide *temporal differential privacy* and *differential privacy under user-user correlations* that enable formal privacy guarantee on correlated data. These models add adaptive noises with respect to the strength of the correlations, to prevent extra privacy loss.
- We designed a data releasing mechanism that be able to convert a traditional DP mechanism into one preventing unexpected privacy due to data correlations. In other words, our mechanism is a remedial tool for previous studies that ignore the effect of data correlation on differential privacy.
- We provide a software toolkit that implement the mechanisms for releasing infinite streaming data and temporally correlated data*.

6.2 Future work

Although the results in this dissertation demonstrated effectiveness of the proposed methods for privacy preserving spatiotemporal data release, they can be extended in a number of ways:

Extending ℓ -trajectory privacy and the proposed algorithmic framework for other data mining task. The algorithms in Chapter 3 were optimized towards a specified utility goal of MAE. However, the proposed framework can be implemented towards other utility goals or data mining tasks, such as frequent sequential pattern mining. On the other hand, in Chapter 3, we design algorithm for releasing the result of count query over infinite streams. It is an interesting question how can we design a releasing algorithm for a complex query (the data mining task can be seen as a combination of simple queries) that satisfies ℓ -trajectory privacy.

Extending differential privacy for complex correlations In Chapters 4 and 5, we model temporal correlations using Markov model and user-user correlations using GMRF. We assume these data correlations are given in our framework because the (trusted) data publisher can easily learn them from the raw data. One question is, for untrusted data publisher who cannot access the raw data, how can

*The source codes are available online: https://github.com/szhan41/TPL_Demo.

he/she learns these correlation in a differentially private manner. Another interesting question is that how can we combine the two model for spatiotemporally correlated data.

Designing advanced data releasing mechanisms for differential privacy on correlated data. We have proposed the first mechanism that be able to convert any existing DP mechanism into one prevent extra privacy loss on temporally or spatially correlation data. This mechanism takes advantage of privacy loss quantification algorithm to allocate appropriate privacy budgets for a traditional DP mechanism. It assumes that the data correlations are available for attackers. However, under such situation, the data publisher should also be able to take advantage of data correlations in the data releasing process. For example, we may not need to add any noise into the data at the next time point if we can predict it using data correlations.

ACKNOWLEDGEMENTS

First and foremost, I offer my sincerest gratitude to my advisor, Professor Masatoshi Yoshikawa, who led me to data privacy research area and supported me during my doctoral study while allowing me the room to work on my own. Without him, this dissertation would not have been completed. I could not wish for a friendlier and more understanding advisor. My committee members, Professor Katsumi Tanaka and Professor Yasuo Okabe have provided me with much advices and insights. Their time and effort are highly appreciated. I would like to thank Associate Professor Qiang Ma, Associate Professor Yasuhito Asano and Assistant Professor Toshiyuki Shimizu in Yoshikawa-Ma lab, they always provide me helpful comments in the seminars and gave me tremendous help in daily research life. I thank my external advisors: Professor Jun Sakuma (University of Tsukuba), Professor Keishi Tajima, Associate Professor Naoto Kume, Associate Professor Hiroshi Tamura. During the course of my research, they helped me to verify my research ideas and made great suggestions. My special thanks goes to Professor Li Xiong (Emory University), who provided me a visiting research assistant opportunity in AIMS (Assured Information Management and Sharing) group at the department of Math&CS of Emory university from May 2016 to March 2017. Thanks to her constructive advices and insightful comments, our discussion on differential privacy is always efficient and productive. I am grateful for the opportunity to work with Professor Xiong.

Besides my advisors, I would like to thank all the members in Yoshikawa-Ma lab at Kyoto university. Secretary Nakajima San saved a lot of time for me on administrative procedure and kindly keep my mails during my stay at Emory. Secretary Nakahara San is alway helpful for providing us enjoyable research environment. My fellow labmates: Ling Xu, Purnomo Husnul Khotimah, Chenyi

Acknowledgements

Zhuang and Yiling Dai, they are cooperative and caring. We always learn from and encourage each other (especially, when our paper got rejected). Thanks to the members of data privacy research group in Yoshikawa-Ma lab, Rachana San and Asada San, our RinKou and weekly discussion on personal data market and trajectory privacy are visionary and insightful.

I would also like to thank all the members in AIMS group at Emory university: Yousef, Daniel, Yonghui, Jinfei, Layla, Jining and Tom. During my visiting at Emory, they does not only provided me extremely valuable feedback on my research, but also helped me to settle down in Atlanta. Especially, I would like to thank Professor Vaidy Sunderam for providing advices on Indian Conference Visa application for attending VLDB 2016, and thank Mr. Derek Baran and Ms. Terry Ingram for helping me on J1 visa and other administrative procedure.

Last but not the least, I would like to thank my parents for giving birth to me at the first place and supporting me spiritually throughout my life; thank my wife for taking care of my life without any complaints; thank my son for bringing me joy and hope. They are my source of energy and really make me love my life.

Yang Cao, February 2017

REFERENCES

- [1] Colleen A. Bradley, H. Rolka, D. Walker, and J. Loonsk. BioSense: implementation of a national early event detection and situational awareness system. *MMWR supplements*, 54:11–19, 2005.
- [2] Federal Highway Administration (FHWA). Traffic monitoring guide, 2013.
- [3] Office of Statewide Health Planning and Development. California inpatient data reporting manual, medical information reporting for california (7th edition). Technical report, 2016.
- [4] Latanya Sweeney. K-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, 2002.
- [5] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. Wherefore art thou r3579x?: Anonymized social networks, hidden patterns, and structural steganography. In *WWW*, pages 181–190, 2007.
- [6] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *SE&P ’08*, pages 111–125, 2008.
- [7] Yves-Alexandre de Montjoye, César A. Hidalgo, Michel Verleysen, and Vincent D. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific Reports*, 3, 2013.
- [8] Yves-Alexandre de Montjoye, Laura Radaelli, Vivek Kumar Singh, and Alex Sandy Pentland. Unique in the shopping mall: On the reidentifiability of credit card metadata. *Science*, 347(6221):536–539, 2015.

- [9] Yan Michalevsky, Gabi Nakibly, Gunaa Arumugam Veerapandian, Dan Boneh, and Gabi Nakibly. PowerSpy: location tracking using mobile device power analysis. In *SEC*, pages 785–800, 2015.
- [10] Benjamin C. M. Fung, Ke Wang, Rui Chen, and Philip S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Comput. Surv.*, 42(4):14:1–14:53, 2010.
- [11] J. Gehrke. Models and methods for privacy-preserving data analysis and publishing. In *22nd International Conference on Data Engineering (ICDE’06)*, pages 105–105, 2006.
- [12] Zhiqiang Yang, Sheng Zhong, and Rebecca N. Wright. Anonymity-preserving data collection. In *KDD*, pages 334–343, 2005.
- [13] David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90, 1981.
- [14] Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *Proceedings of the 11th USENIX Security Symposium*, pages 339–353, 2002.
- [15] Stanley L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.
- [16] Tore Dalenius. Towards a methodology for statistical disclosure control. *Statistik Tidskrift*, 15:429–444, 1977.
- [17] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1):3–es, 2007.
- [18] Ninghui Li, Tiancheng Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *IEEE 23rd International Conference on Data Engineering, 2007. ICDE 2007*, pages 106–115, 2007.
- [19] Xiaokui Xiao and Yufei Tao. M-invariance: Towards privacy preserving republication of dynamic datasets. In *SIGMOD*, pages 689–700, 2007.

- [20] Cynthia Dwork. Differential privacy. In *ICALP*, pages 1–12, 2006.
- [21] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.
- [22] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, 2007.
- [23] Gergely Acs and Claude Castelluccia. A case study: Privacy preserving release of spatio-temporal density in paris. In *KDD*, pages 1679–1688, 2014.
- [24] Jean Bolot, Nadia Fawaz, S. Muthukrishnan, Aleksandar Nikolov, and Nina Taft. Private decayed predicate sums on streams. In *ICDT*, pages 284–295, 2013.
- [25] T.-H. Hubert Chan, Mingfei Li, Elaine Shi, and Wenchang Xu. Differentially private continual monitoring of heavy hitters from distributed streams. In *PETS*, pages 140–159, 2012.
- [26] T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, 2011.
- [27] Cynthia Dwork and Adam Smith. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2):2, 2010.
- [28] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In Leonard J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 715–724, 2010.
- [29] Liyue Fan, Li Xiong, and Vaidy Sunderam. FAST: differentially private real-time aggregate monitor with filtering and adaptive sampling. In *SIGMOD*, pages 1065–1068, 2013.
- [30] Georgios Kellaris, Stavros Papadopoulos, Xiaokui Xiao, and Dimitris Papadias. Differentially private event sequences over infinite streams. *Proc. VLDB Endow.*, 7(12):1155–1166, 2014.

- [31] Yonghui Xiao, J. Gardner, and Li Xiong. DPCube: releasing differentially private data cubes for health information. In *ICDE*, pages 1305–1308, 2012.
- [32] Xinpeng Zhang, Hiroki Kitabayashi, Yasuhito Asano, and Masatoshi Yoshikawa. A health-aware pedestrian navigation system by analysis of spatiotemporal vital sign data. 2014.
- [33] Liyue Fan, Luca Bonomi, Li Xiong, and Vaidy Sunderam. Monitoring web browsing behavior with differential privacy. In *WWW*, pages 177–188, 2014.
- [34] Cynthia Dwork. A firm foundation for private data analysis. *Commun. ACM*, 54(1):86–95, 2011.
- [35] A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, and L. Vilhuber. Privacy: Theory meets practice on the map. In *IEEE 24th International Conference on Data Engineering, 2008. ICDE 2008*, pages 277–286, 2008.
- [36] Lucila Ohno-Machado, Vineet Bafna, Aziz A. Boxwala, Brian E. Chapman, Wendy W. Chapman, Kamalika Chaudhuri, Michele E. Day, Claudiu Farcas, Nathaniel D. Heintzman, Xiaoqian Jiang, Hyeoneui Kim, Jihoon Kim, Michael E. Matheny, Frederic S. Resnic, Staal A. Vinterbo, Team, and the iDASH. iDASH: integrating data for analysis, anonymization, and sharing. *Journal of the American Medical Informatics Association*, 19(2):196–201, 2012.
- [37] Darakhshan Mir, S. Muthukrishnan, Aleksandar Nikolov, and Rebecca N. Wright. Pan-private algorithms via statistics on sketches. In *PODS*, pages 37–48, 2011.
- [38] Osman Abul, Francesco Bonchi, and Mirco Nanni. Never walk alone: Uncertainty for anonymity in moving objects databases. In *ICDE*, pages 376–385, 2008.
- [39] Anna Monreale, Gennady Andrienko, Natalia Andrienko, Fosca Giannotti, Dino Pedreschi, Salvatore Rinzivillo, and Stefan Wrobel. Movement data anonymity through generalization. *Trans. Data Privacy*, 3(2):91–121, 2010.
- [40] Roman Yarovoy, Francesco Bonchi, Laks VS Lakshmanan, and Wendy Hui Wang. Anonymizing moving objects: How to hide a mob in a crowd? In *Pro-*

- ceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pages 72–83, 2009.
- [41] Manolis Terrovitis and Nikos Mamoulis. Privacy preservation in the publication of trajectories. In *Mobile Data Management, 2008. MDM'08. 9th International Conference on*, pages 65–72, 2008.
- [42] Rui Chen, Benjamin C. M. Fung, Noman Mohammed, Bipin C. Desai, and Ke Wang. Privacy-preserving trajectory data publishing by local suppression. *Information Sciences*, 231:83–97, 2013.
- [43] Kaifeng Jiang, Dongxu Shao, Stéphane Bressan, Thomas Kister, and Kian-Lee Tan. Publishing trajectories with differential privacy guarantees. In *SSDBM*, pages 12:1–12:12, 2013.
- [44] Shen-Shyang Ho and Shuhua Ruan. Differential privacy for location pattern mining. In *SPRINGL*, pages 17–24, 2011.
- [45] Shen-Shyang Ho and Shuhua Ruan. Preserving privacy for interesting location pattern mining from trajectory data. *Transactions on Data Privacy*, 6(1):87–106, 2013.
- [46] Daniel Kifer and Ashwin Machanavajjhala. No free lunch in data privacy. In *SIGMOD*, pages 193–204, 2011.
- [47] Frank D. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 19–30, 2009.
- [48] Jia Xu, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, and Ge Yu. Differentially private histogram publication. In *ICDE*, pages 32–43, 2012.
- [49] X. Zhang, R. Chen, J. Xu, X. Meng, and Y. Xie. Towards accurate histogram publication under differential privacy. In *Proceedings*, pages 587–595. 2014.
- [50] Minos Garofalakis and Amit Kumar. Wavelet synopses for general error metrics. *ACM Trans. Database Syst.*, 30(4):888–928, 2005.
- [51] Xiaokui Xiao, Gabriel Bender, Michael Hay, and Johannes Gehrke. iReduct: differential privacy with reduced relative errors. In *Proceedings of the 2011*

- ACM SIGMOD International Conference on Management of data*, pages 229–240, 2011.
- [52] Raymond Kosala and Hendrik Blockeel. Web mining research: A survey. *SIGKDD Explor. Newsl.*, 2(1):1–15, 2000.
- [53] Daniel Kifer and Ashwin Machanavajjhala. A rigorous and customizable framework for privacy. In *PODS*, pages 77–88, 2012.
- [54] Daniel Kifer and Ashwin Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Trans. Database Syst.*, 39(1):3:1–3:36, 2014.
- [55] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. Driving with knowledge from the physical world. In *KDD*, pages 316–324, 2011.
- [56] Sebastien Gambs, Marc-Olivier Killijian, and Miguel Nunez del Prado Cortez. Next place prediction using mobility markov chains. In *MPM*, pages 3:1–3:6, 2012.
- [57] Teerayut Horanont, Santi Phithakkitnukoon, Tuck W. Leong, Yoshihide Sekimoto, and Ryosuke Shibasaki. Weather effects on the patterns of people’s everyday activities: A study using GPS traces of mobile phone users. *PLoS ONE*, 8(12):e81153, 2013.
- [58] Rui Chen, Benjamin C. Fung, Philip S. Yu, and Bipin C. Desai. Correlated network data publication via differential privacy. *VLDBJ*, 23(4):653–676, 2014.
- [59] Bin Yang, Issei Sato, and Hiroshi Nakagawa. Bayesian differential privacy on correlated data. In *SIGMOD*, pages 747–762, 2015.
- [60] Dependence makes you vulnerable: Differential privacy under dependent tuples.
- [61] Reza Shokri, George Theodorakopoulos, Jean-Yves Le Boudec, and Jean-Pierre Hubaux. Quantifying location privacy. In *SP*, pages 247–262, 2011.
- [62] George Theodorakopoulos, Reza Shokri, Carmela Troncoso, Jean-Pierre Hubaux, and Jean-Yves Le Boudec. Prolonging the hide-and-seek game:

- Optimal trajectory privacy for location-based services. In *WPES*, pages 73–82, 2014.
- [63] Yonghui Xiao and Li Xiong. Protecting locations with differential privacy under temporal correlations. In *CCS*, pages 1298–1309, 2015.
- [64] Cynthia Dwork and Aaron Roth. *The Algorithmic Foundations of Differential Privacy*, volume 9. 2013.
- [65] Wesley Mathew, Ruben Raposo, and Bruno Martins. Predicting future locations with hidden markov models. In *UbiComp*, pages 911–918, 2012.
- [66] Z. Jorgensen, Ting Yu, and G. Cormode. Conservative or liberal? personalized differential privacy. In *ICDE*, pages 1023–1034, 2015.
- [67] Erik B. Bajalinov. *Linear-Fractional Programming Theory, Methods, Applications and Software*, volume 84. 2003.
- [68] George Bernard Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1998.
- [69] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *SGP*, pages 175–184, 2004.
- [70] Tianqing Zhu, Ping Xiong, Gang Li, and Wanlei Zhou. Correlated differential privacy: Hiding information in non-IID data set. *IEEE Trans. Inf. Forensics Security.*, 10(2):229–242, 2015.
- [71] Liyue Fan, Li Xiong, and Vaidy Sunderam. Differentially private multi-dimensional time series release for traffic monitoring. In *DBSec*, pages 33–48, 2013.
- [72] Yang Cao and Masatoshi Yoshikawa. Differentially private real-time data publishing over infinite trajectory streams. *IEICE Trans. Inf. & Syst.*, E99-D(1), 2016.
- [73] Jianneng Cao, Qian Xiao, Gabriel Ghinita, Ninghui Li, Elisa Bertino, and Kian-Lee Tan. Efficient and accurate strategies for differentially-private sliding window queries. In *EDBT*, pages 191–202, 2013.

- [74] Arik Friedman, Izchak Sharfman, Daniel Keren, and Assaf Schuster. Privacy-preserving distributed stream monitoring. 2014.
- [75] D.J. Mir, S. Isaacman, R. Cáceres, M. Martonosi, and R.N. Wright. DP-WHERE: differentially private modeling of human mobility. In *IEEE Big-Data*, pages 580–588, 2013.
- [76] Havard Rue and Leonhard Held. *Gaussian Markov Random Fields: Theory and Applications*. CRC Press, 2005.
- [77] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. 2009.
- [78] Eunjoon Cho, Seth A. Myers, and Jure Leskovec. Friendship and mobility: User movement in location-based social networks. In *KDD*, pages 1082–1090, 2011.
- [79] Werner Dinkelbach. On nonlinear fractional programming. *Management Science*, 13(7):492–498, 1967.

SELECTED LIST OF PUBLICATIONS

- **Journals**

- [1] Yang Cao, Masatoshi Yoshikawa. Differentially Private Real-time Data Publishing over Infinite Trajectory Stream. IEICE Transactions on Information and Systems, Vol.E99–D, No.1, Jan. 2016.

- **International Conferences and Workshops**

- [2] Yang Cao, Masatoshi Yoshikawa, Yonghui Xiao, Li Xiong. Quantifying Differential Privacy under Temporal Correlations. International Conference on Data Engineering (ICDE) 2017.
- [3] Yang Cao. Rigorous and Flexible Privacy Models for Utilizing Personal Spatiotemporal Data. International Conference on Very Large Data Bases (VLDB) 2016, PhD Workshop.
- [4] Yang Cao, Masatoshi Yoshikawa. Differentially Private Real-Time Data Release over Infinite Trajectory Streams. In Proceedings of the 2015 16th IEEE International Conference on Mobile Data Management (MDM), Volume 02, Washington, DC, USA, 2015, pp. 68–73.
- [5] Yang Cao, Si Zhang, Masatoshi Yoshikawa, Yonghui Xiao, Li Xiong. Release Temporally Correlated Data with Differential Privacy. Submitted to an International Conference.

- **Domestic Conferences**

- [6] Yang Cao, Masatoshi Yoshikawa. Differentially Data Release over Trajectory Streams (in Japanese). The 7th Forum on Data Engineering and Information Management (DEIM2015).

APPENDIX

A Proof of Theorem 10

Proof. According to Definition 11 of TPL, the overall privacy leakage of $\mathcal{M}^t$ and $\mathcal{M}^{t+1}$ in terms of $\text{DP}_{\mathcal{T}}$ are

$$\begin{aligned}
& \sup_{D^t, D_{\mathcal{M}}^t, \dots, D_{\mathcal{M}}^{t+j'}, \mathbf{r}^1, \dots, \mathbf{r}^T} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | D^t, D^{t+1})}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | D^{t'}, D^{t+1'})} \\
&= \sup_{\substack{D^t, D^{t'}, \\ \mathbf{r}^1, \dots, \mathbf{r}^t}} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | D^t)}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | D^{t'})} + \sup_{\substack{D^t, D^{t'}, \\ \mathbf{r}^{t+1}, \dots, \mathbf{r}^T}} \log \frac{\Pr(\mathbf{r}^{t+1}, \dots, \mathbf{r}^T | D^{t+1})}{\Pr(\mathbf{r}^{t+1}, \dots, \mathbf{r}^T | D^{t+1'})} \\
&= BPL(\mathcal{M}^t) + FPL(\mathcal{M}^{t+1}) \\
&= \alpha_t^B + \alpha_{t+j}^F.
\end{aligned}$$

Therefore, the theorem follows when $j = 1$.

According to Definition 15, the overall privacy leakage of $\mathcal{M}^t, \dots, \mathcal{M}^{t+j}$ in terms of $\text{DP}_{\mathcal{T}}$ is as follows.

$$\begin{aligned}
& \sup_{\substack{D^t, \dots, D^{t+j}, \\ D^{t'}, \dots, D^{t+j'}, \\ \mathbf{r}^1, \dots, \mathbf{r}^T}} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | D^t, \dots, D^{t+j})}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^T | D^{t'}, \dots, D^{t+j'})} \\
&= \sup_{\substack{D^t, D^{t'}, \\ \mathbf{r}^1, \dots, \mathbf{r}^t}} \log \frac{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | D^t)}{\Pr(\mathbf{r}^1, \dots, \mathbf{r}^t | D^{t'})} + \sup_{\substack{D^{t+j}, D^{t+j'}, \\ \mathbf{r}^{t+j}, \dots, \mathbf{r}^T}} \log \frac{\Pr(\mathbf{r}^{t+j}, \dots, \mathbf{r}^T | D^{t+j})}{\Pr(\mathbf{r}^{t+j}, \dots, \mathbf{r}^T | D^{t+j'})} \\
&+ \sup_{\substack{D^{t+1}, \\ D^{t+1'}, \\ \mathbf{r}^{t+1}}} \log \frac{\Pr(\mathbf{r}^{t+1} | D^{t+1})}{\Pr(\mathbf{r}^{t+1} | D^{t+1'})} + \dots + \sup_{\substack{D^{t+j-1}, \\ D^{t+j-1'}, \\ \mathbf{r}^{t+j-1}}} \log \frac{\Pr(\mathbf{r}^{t+j-1} | D^{t+j-1})}{\Pr(\mathbf{r}^{t+j-1} | D^{t+j-1'})} \\
&= BPL(\mathcal{M}^t) + FPL(\mathcal{M}^{t+j}) + PL_0(\mathcal{M}^{t+1}) + \dots PL_0(\mathcal{M}^{t+j-1}) \\
&= \alpha_t^B + \alpha_{t+j}^F + \sum_{k=1}^{k=j-1} \epsilon_{t+k}.
\end{aligned}$$

Therefore, the theorem follows when $j \geq 2$. $\square$

B Proof of Theorem 12

We need Dinkelbach's Theorem and Lemma 10 in our proof.

Theorem 17 (Dinkelbach's Theorem [79]). *In a linear-fractional programming problem, suppose that the variable vector is $\mathbf{x}$ and the objective function is represented as $\frac{Q(\mathbf{x})}{D(\mathbf{x})}$. Vector $\mathbf{x}^*$ is an optimal solution if and only if*

$$\max\{Q(\mathbf{x}) - \lambda * D(\mathbf{x})\} = 0 \text{ where } \lambda = \frac{Q(\mathbf{x}^*)}{D(\mathbf{x}^*)}. \quad (6.1)$$

Lemma 10. *For the following maximization problem ($k_1, \dots, k_n \in \mathbb{R}$) with the same constraints as the ones in the linear-fractional programming (4.17)~(4.19),*

$$\begin{aligned}
& \text{maximize } k_1 x_1 + \dots + k_n x_n \\
& \text{subject to } e^{-\alpha_{t-1}^B} * x_k \leq x_j \leq e^{\alpha_{t-1}^B} * x_k, \\
& \quad 0 < x_j < 1 \text{ and } 0 < x_k < 1, \\
& \quad \text{where } x_j, x_k \in \mathbf{x}, \quad j, k \in [1, n].
\end{aligned}$$

an optimal solution is as follows: if $k_i > 0$, let $x_i = e^{\alpha_{t-1}^B} m$ where m is a positive real number; if $k_i \leq 0$, let $x_i = m$.

Proof. Without loss of generality, we suppose that the smallest value in the optimal solution is x_n . Let y_j be $\frac{x_j}{x_n}$ for $j \in [1, n-1]$; then, $1 \leq y_j \leq e^{\alpha_{t-1}^B}$. Replacing x_j with y_j and setting $x_n = m$, we have a new objective function $\frac{1}{m} * (k_1 y_1 + \dots + k_{n-1} y_{n-1} + k_n)$ whose solution is equivalent to the original one. Because the only constraint is $1 \leq y_j \leq e^{\alpha_{t-1}^B}$, the following is an optimal solution for the maximum objective function: if $k_j > 0$, let $y_j = e^{\alpha_{t-1}^B}$; if $k_j \leq 0$, let $y_j = 1$. $\square$

Proof of Theorem 12. We first prove that, under the conditions shown in Theorem 12, i.e., Inequalities (4.20) and (4.21), an optimal solution of the problem (4.17)~(4.19) is:

$$\mathbf{x}^* = \begin{cases} x_j = e^{\alpha_{t-1}^B} * m & x_j \in \mathbf{x}^+ \\ x_k = m & x_k \in \mathbf{x}^- \end{cases}, \quad (6.2)$$

where m is a positive real number.

For convenience, we rewrite our objective function as $\frac{Q(\mathbf{x})}{D(\mathbf{x})}$ in which $Q(\mathbf{x}) = \mathbf{q}\mathbf{x}$ and $D(\mathbf{x}) = \mathbf{d}\mathbf{x}$. Substituting $\mathbf{x}^*$ of Equation (6.2) into $Q(\mathbf{x})$ and $D(\mathbf{x})$, we have $Q(\mathbf{x}^*) = q(e^{\alpha_{t-1}^B} - 1) + 1$ and $D(\mathbf{x}^*) = d(e^{\alpha_{t-1}^B} - 1) + 1$ (recall that $q = \sum \mathbf{q}^+$ and $d = \sum \mathbf{d}^+$). Then, we can rewrite Inequalities (4.20) and (4.21) in Theorem 12 as follows.

$$\frac{q_j}{d_j} > \frac{Q(\mathbf{x}^*)}{D(\mathbf{x}^*)}, \quad \forall j \in [1, n] \text{ where } q_j \in \mathbf{q}^+, d_j \in \mathbf{d}^+ \quad (6.3)$$

$$\frac{q_k}{d_k} \leq \frac{Q(\mathbf{x}^*)}{D(\mathbf{x}^*)}, \quad \forall k \in [1, n] \text{ where } q_k \in \mathbf{q}^-, d_k \in \mathbf{d}^- \quad (6.4)$$

According to Dinkelbach's Theorem, to prove $\mathbf{x}^*$ in (6.2) is an optimal solution, we only need to prove the following equation because of $D(\mathbf{x}^*) > 0$.

$$\text{maximize } \{D(\mathbf{x}^*)Q(\mathbf{x}) - Q(\mathbf{x}^*)D(\mathbf{x})\} = 0. \quad (6.5)$$

We expand the above equation as follows.

$$\begin{aligned} \text{Eqn. (6.5)} &= D(\mathbf{x}^*)(\mathbf{q}^+ \mathbf{x}^+ + \mathbf{q}^- \mathbf{x}^-) - Q(\mathbf{x}^*)(\mathbf{d}^+ \mathbf{x}^+ + \mathbf{d}^- \mathbf{x}^-) \\ &= (D(\mathbf{x}^*)\mathbf{q}^+ - Q(\mathbf{x}^*)\mathbf{d}^+) \mathbf{x}^+ + (D(\mathbf{x}^*)\mathbf{q}^- - Q(\mathbf{x}^*)\mathbf{d}^-) \mathbf{x}^- \end{aligned} \quad (6.6)$$

By Equations (6.3) and (6.4), we have $D(\mathbf{x}^*)\mathbf{q}^+ - Q(\mathbf{x}^*)\mathbf{d}^+ > 0$ and $D(\mathbf{x}^*)\mathbf{q}^- - Q(\mathbf{x}^*)\mathbf{d}^- \leq 0$. Hence, according to Lemma 10, we can obtain the maximum value

in Equation (6.5) by setting $\mathbf{x}^+ = [e^{\alpha_{t-1}^B} * m]$ and $\mathbf{x}^- = [m]$ where m is a positive real number. Now, we obtain the maximum value in Equation (6.5).

$$\begin{aligned} & ((D(\mathbf{x}^*)q - Q(\mathbf{x}^*)d)e^\varepsilon m + (D(\mathbf{x}^*)(1-q) - Q(\mathbf{x}^*)(1-d)))m \\ &= (D(\mathbf{x}^*)(qe^\varepsilon + (1-q)) - Q(\mathbf{x}^*)(de^\varepsilon + (1-d)))m \\ &= (D(\mathbf{x}^*)Q(\mathbf{x}^*) - Q(\mathbf{x}^*)D(\mathbf{x}^*))m = 0 \end{aligned}$$

Therefore, by Dinkelbach's Theorem, $\mathbf{x}^*$ is an optimal solution for the problem (4.17)~(4.19). Substituting them into the objective function (4.17), we obtain the maximum value $\frac{q(e^{\alpha_{t-1}^B}-1)+1}{d(e^{\alpha_{t-1}^B}-1)+1}$. $\square$

C Proof of Corollary 2

Proof. The proof is by contradiction.

First, assume that $q = d$, i.e., $\sum \mathbf{q}^+ = \sum \mathbf{d}^+$. That is, there exist $q_j \in \mathbf{q}^+$ and $d_j \in \mathbf{d}^+$ satisfying $\frac{q_j}{d_j} \leq 1$. This leads to a contradiction to Inequality (4.20) because of $\frac{q_i}{d_i} > \frac{q(e^{\alpha_{t-1}^B}-1)+1}{d(e^{\alpha_{t-1}^B}-1)+1} = 1$. Hence, $q = d$ is false.

Second, assume that $q < d$, i.e., $\sum \mathbf{q}^+ < \sum \mathbf{d}^+$. Since $\sum (\mathbf{q}^+ + \mathbf{q}^-) = \sum (\mathbf{d}^+ + \mathbf{d}^-) = 1$, we have $\sum \mathbf{q}^- > \sum \mathbf{d}^-$. That is, there exist $q_k \in \mathbf{q}^-$ and $d_k \in \mathbf{d}^-$ satisfying $\frac{q_k}{d_k} > 1$. This leads to a contradiction to Inequality (4.21) because of $\frac{q_k}{d_k} < \frac{q(e^{\alpha_{t-1}^B}-1)+1}{d(e^{\alpha_{t-1}^B}-1)+1} < 1$. Hence, $q < d$ is false.

Therefore, $q > d$ is true. Then, we have $\frac{q(e^{\alpha_{t-1}^B}-1)+1}{d(e^{\alpha_{t-1}^B}-1)+1} > 1$. By Inequality (4.20), it follows that $\frac{q_j}{d_j} > \frac{q(e^{\alpha_{t-1}^B}-1)+1}{d(e^{\alpha_{t-1}^B}-1)+1} > 1$ in which $q_j \in \mathbf{q}^+$ and $d_j \in \mathbf{d}^+$. $\square$

D Proof of Theorem 13

Proof. We denote the supremum of BPL or FPL over time by α . If $d = 0$ and $q = 1$, it is easy to see that $\mathcal{L}^B(\cdot) = \mathcal{L}^F(\cdot) = 1 * \cdot$. In other words, BPL and FPL will increase linearly without limit; hence, $\mathcal{F}$ does not exist. If BPL (or FPL) has a limit, since the current privacy leakage should be calculated based on the previous one (or the next one for FPL), we have $\alpha = \mathcal{L}^B(\alpha) + \epsilon$ (see Equations (4.12) or (4.14)), namely $\alpha = \log \frac{q(e^\alpha-1)+1}{d(e^\alpha-1)+1} + \epsilon$. By expanding this equation and letting $\mathcal{F}$ be e^α , we have an equation with a variable $\mathcal{F}$, which $\mathcal{F} > 1$.

$$d\mathcal{F}^2 + (qe^\epsilon + d - 1)\mathcal{F} + (qe^\epsilon - e^\epsilon) = 0 \quad (6.7)$$

Hence, the logarithm of the solution (if it exists) in the above equation is the supremum of BPL or FPL. If $d = 0$ and $q \neq 1$, we have $\mathcal{F} = \frac{(1-q)e^\epsilon}{1-qe^\epsilon}$. To ensure a positive $\mathcal{F}$, because of $(1-q)e^\epsilon > 0$, we need $1 - qe^\epsilon > 0$, i.e., $\epsilon < \log(1/q)$. Therefore, when $d = 0$, $q \neq 1$ and $\epsilon \geq \log(1/q)$, the supremum does not exist; when $d = 0$, $q \neq 1$ and $\epsilon < \log(1/q)$, we have the supremum $\log \frac{(1-q)e^\epsilon}{1-qe^\epsilon}$. If $d \neq 0$, by solving the quadratic equation (6.7), we can obtain a positive solution: $\frac{\sqrt{4de^\epsilon(1-q)+(d+qe^\epsilon-1)^2+d+qe^\epsilon-1}}{2d}$. it is easy to verify that this solution always exist and is greater than 1. Therefore, the theorem follows. $\square$