UNIVERSITY OF CALIFORNIA,
IRVINE

Accuracy Aware Privacy Preserving Decision Support

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Sameera Ghayyur

Dissertation Committee:
Sharad Mehrotra, Chair
Xi He
Nalini Venkatasubramanian
Michael Goodrich

2022

DEDICATION

To my father, for being my biggest cheerleader!

پیارے ابو کے نام...

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

I want to take this opportunity to thank a lot of people who have been instrumental in completing my PhD.

First and foremost, my advisor, **Sharad Mehrotra**, for introducing me to the world of privacy and your valuable feedback and support. You have been there for me every step of the way throughout my PhD journey. Your passion for research is truly inspirational. I would like to thank **Xi He** for being a remarkable mentor and guiding my research work with your expertise in privacy.

I would like to thank my other collaborators for the research work presented in this thesis - **Dhrubajyoti Gosh, Roberto Yus, Yan Chen, Ashwin Machanavajjhala, Michael Hay, Gerome Miklau, Alisa Pankova, Peeter Laud**. I would also like to thank **Michael Goodrich** and **Nalini Venkatasubramanian** for being part of my thesis committee and your valuable feedback. I would also like to mention some of my mentors **Mumtaz Abbas, Basit Shafiq, Sadia Khan, Paul Gagliardi** and **Asim Farooqi** who always believed in me and have inspired me to pursue my dreams. I am thankful to my lab mates, specially **Dhrub, Primal, Roberto, Guoxi** and **Peeyush**, for your intellectual company and friendship.

I am extremely lucky and grateful to have wonderful friends who have been there for me, in person and from afar - **Anam, Sumaya, Wardah, Greta, Pedro, Maruf, Efi, Yaqoob, Ayesha** and many others that I may have missed.

Most importantly, I am grateful to my parents - **Razia Sultana** and **Muhammad Ghayyur Ahsan**, my siblings - **Zakiya, Mudeera** and **Hassan** and my grandfather - **Muhammad Awwal Shah** - for always believing in me and being supportive of my academic endeavors. I am thankful to my little nephews - **Rohaan** and **Sharoon**, for cheering me from afar with your cute little voice notes.

I am especially thankful to **Josh** for your kindness, love and support.

This work was partially funded by the research sponsored by DARPA under agreement number FA8750-16-2-0021, NSF Grants No. 1952247, 2133391, 2032525, 2008993.

VITA

Sameera Ghayyur

EDUCATION

Doctor of Philosophy in Computer Science **2022**
University of California, Irvine *Irvine, California*

Master of Science in Computer Science **2015**
Lahore University of Management Sciences *Lahore, Pakistan*

Bachelor of Science in Computer Science **2013**
Lahore University of Management Sciences *Lahore, Pakistan*

RESEARCH EXPERIENCE

Graduate Research Assistant **2017–2022**
University of California, Irvine *Irvine, California*

Research Assistant **2015–2016**
Lahore University of Management Sciences *Irvine, California*

TEACHING EXPERIENCE

Teaching Assistant **2016–2021**
University of California, Irvine *Irvine, California*

Lecturer **2015–2016**
Forman Christian College (A Chartered University) *Lahore, Pakistan*

REFEREED CONFERENCE PUBLICATIONS

MIDE: Accuracy Aware Minimally Invasive Data Exploration For Decision Support **Sep 2022**
48th International Conference on Very Large Databases (VLDB)

Designing privacy preserving data sharing middleware for internet of things **Nov 2020**
3rd International SenSys+BuildSys Workshop on Data: Acquisition to Analysis (DATA)

Semiotic: Bridging the semantic gap in iot spaces **Nov 2019**
6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys)

Towards accuracy aware minimally invasive monitoring (MiM) **Nov 2019**
ACM Conference on Computer and Communications Security, Theory and Practice of Differential Privacy Workshop (TPDP)

Iot-detective: Analyzing iot data under differential privacy **Jun 2018**
ACM Proceedings of the 2018 International Conference on Management of Data (SIGMOD)

REFEREED JOURNAL PUBLICATIONS

Composability verification of multi-service workflows in a policy-driven cloud computing environment **2015**
IEEE Transactions on Dependable and Secure Computing (TDSC)

BOOK CHAPTERS

A Privacy-Sensitive Collaborative Approach to Business Process Development **2015**
E-Business and Telecommunications, Communications in Computer and Information Science Series, Springer

ABSTRACT OF THE DISSERTATION

Accuracy Aware Privacy Preserving Decision Support

By

Sameera Ghayyur

Doctor of Philosophy in Computer Science

University of California, Irvine, 2022

Sharad Mehrotra, Chair

In this thesis, we study privacy in the context of Decision Support(DS) applications. DS applications utilize data collected from numerous sources to guide important decisions. However, such applications could face severe privacy challenges if the data contains sensitive information about individuals. While techniques such as differential privacy are suited for privacy-preserving data sharing, their usefulness in the context of decision support (DS) applications is limited due to privacy and utility trade-offs as these techniques do not offer any guarantees on the quality of results. DS tasks, in contrast, require guarantees on the output quality to avoid making misleading and inaccurate decisions.

We explore the concept of minimally invasive data exploration for decision support that attempts to minimize privacy loss while supporting bounded guarantees on accuracy. We build a demo application and conduct empirical studies to understand privacy utility trade-offs of different privacy techniques and to highlight the need for accuracy driven privacy preserving data analysis. We formally define decision support queries and their accuracy requirements and present privacy preserving algorithms to answer these queries that minimize the privacy loss while providing the required accuracy guarantees for decision support.

Chapter 1

Introduction

Decision Support (DS) applications [22, 53, 4] analyze data collected from numerous sources to produce valuable insights and allow timely and informed decision-making. These decision support applications are built on top of data analysis tools/ techniques such as OLAP(On-line analytical processing) and various data mining techniques e.g., decision trees, machine learning algorithms like neural networks, random forest trees.

On-line analytical processing (OLAP) and data warehousing are essential elements of decision support that are widely used in organizational decision making[46]. In a typical OLAP system, the data is aggregated over n-dimensional data cube and can be used to learn aggregated statistics e.g. how many copies of a particular book were sold in all northeast stores in the years 2021 and 2022? This kind of aggregated statistics can be used to make important business decisions e.g. deciding whether the company should keep selling a particular product. Similarly, data collected from Wifi access points can be used to learn the occupancy statistics inside a building. This data can be used to detect violation of fire code or efficient space utilization inside a building. Another example is of assisted living situations where one of the primary challenges is fall prevention of the elderly. The elderly could be monitored

using cameras and wearable devices to collect aggregated statistics e.g., number of sudden accelerations in a week. Such data can be used to make a decision about whether the elderly is a high fall risk and we can make a decision to monitor such an individual more closely.

While data driven Decision Support(DS) systems provide highly valuable information to guide important decisions, the data collected can be highly sensitive and can leak sensitive information about individuals. For example, prior work [32] has shown that occupancy data, with enough background knowledge, can lead to inferences about location of individuals, which, in turn, can leak sensitive information (e.g., faculty arriving late to classes, staff consistently leaving work early, smoking habits of individuals, etc.).

Ensuring user privacy in the data driven systems is a challenging and well-recognized research topic. Most of the commonly used privacy preserving technologies in the literature e.g. anonymization, obfuscation, randomization and differential privacy achieve privacy by adding noise or removing sensitive information. The techniques protect privacy at a cost of utility of the data and there is a direct trade-off between the utility and privacy. If the data is not accurate or noisy, the decisions made using privacy preserving decision support system can be misleading and inaccurate. Therefore, decision support systems not only need to incorporate privacy to protect individuals' data but also need insurance about the accuracy of the decision.

In this dissertation, we solve the problem of accuracy driven privacy preserving decision support applications. There are several privacy techniques that have been well studied in different application contexts for data analysis. Before we embark on developing privacy techniques for DS, we first explore existing state of the art techniques in the context of DS. Privacy loss is usually defined in the context of adversarial model i.e. ability of the adversary to learn sensitive information. Differential Privacy (DP) stands out among other techniques as it provides strong formal guarantees for privacy loss and protects against very strong adversaries, even ones who know the entire dataset except for one piece of information

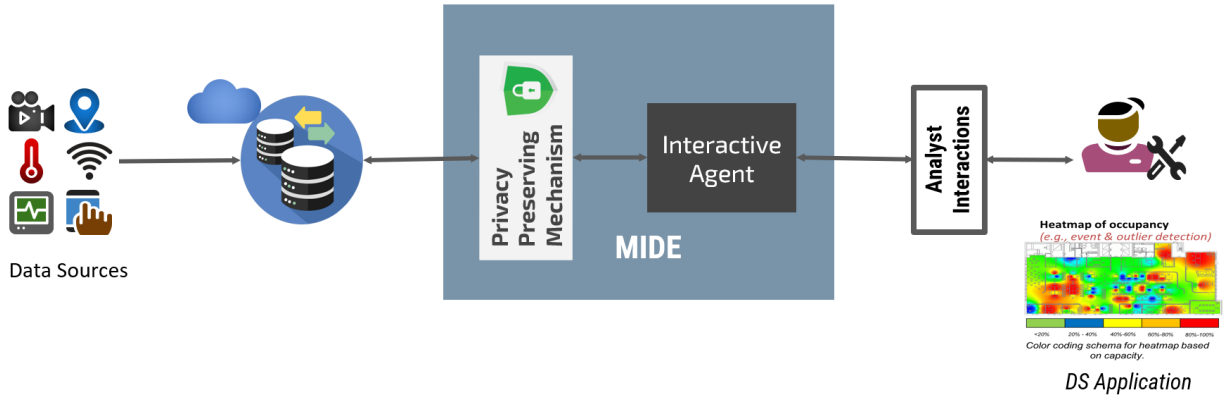


Figure 1.1: MIDE framework for minimally invasive data exploration for decision support applications

.

about one individual. In practical world, however, adversary might not be as strong and approach required may only need to protect against a limited adversary. We conduct two studies to understand the existing techniques both DP and techniques for different types of adversary models in the context of decision support. These studies demonstrate that same privacy levels for data analysis can result in vastly different utility depending on the dataset and highlight the need for accuracy centric approach to privacy preserving decision support as inaccurate results can be misleading and defeat the very purpose of decision support application.

We formally define decision support queries, their accuracy requirements and provide various algorithms to answer decision support queries that satisfy the required accuracy bounds while minimizing the privacy loss through our framework called '**MIDE**' (Minimally Invasive Data Exploration). MIDE uses privacy preserving mechanism to answer queries through an interactive agent that controls the privacy level necessary to correctly perform the decision support task as shown in Figure 1.1.

1.1 Thesis Organization

In Chapter 2, we present related work in the context of privacy preserving data analysis and decision support applications. To provide motivation for the need for accuracy guarantees for privacy preserving data driven solutions, we present two separate pieces of works (Chapter 3 and 4). This work highlights the importance of accuracy aware privacy preserving data analytics. It considers privacy preserving occupancy analysis for building management system as an example to highlight limitations of current privacy techniques and their privacy/ utility trade-offs. Chapter 3 presents an informal study based on a demo application presented at a conference(SIGMOD'19) to assess the usefulness of private streaming data in a real-world application setting in IoT domain. The demo consists of a game, in which participants carry out visual data analysis tasks on private data streams, earning points when they achieve results similar to those on the true data stream. The data collected from this demo helped us understand impact of privacy mechanisms on usefulness of data for DS. Chapter 4 presents a privacy case study of privacy mechanisms to publish occupancy data in the context of a smart building. The goal of the study was to explore the practical privacy implications for individuals' privacy of the release of occupancy counts. We proposed a methodology to compare across privacy techniques (e.g. randomization, differential privacy) with different privacy parameters and guarantees under realistic adversarial scenarios. This study highlights the trade-off between privacy and utility across various techniques.

Chapter 5 and 6 present accuracy aware privacy preserving data exploration algorithms for decision support systems. Most of the recent work on privacy preserving techniques focuses on maximizes accuracy given some privacy constraints. In decision support systems, accuracy is of prime importance as inaccurate results may render DS applications utterly useless. Our algorithms focus on accuracy first approach while minimizing privacy loss. Chapter 5 focuses on developing privacy preserving techniques for simple decision support queries where decisions are made based on a condition on single aggregate statistical value. These algo-

rithms satisfy given accuracy constraints and minimize the privacy loss. Chapter 6 present accuracy aware privacy preserving algorithms for more complex DS queries where decisions are made based on conjunction/disjunction of a set of conditions on multiple aggregated statistics. These algorithms also focus on providing required accuracy guarantees while minimizing privacy loss. In Chapter 7, we provide conclusion and possible future directions for our work.

Chapter 2

Preliminaries and Related Work

In this chapter, we provide background and related work in the context of privacy preserving decision support systems. First, we provide a privacy primer where we describe most commonly used privacy preserving technologies to release data. Since utility is of prime importance in the context of decision support application, we also provide an overview of the work in the context of privacy vs. utility trade offs and accuracy aware state of the art work for most widely used privacy technique differential privacy.

2.1 Privacy Primer

We live in a world that is becoming increasingly data driven. Large amounts of personal data is being collected by different organization (medical history, census berue,location data,) to facilitate and improve people's lives.

Much of the prior work on privacy has been motivated by the need for data sharing while ensuring privacy of sensitive data about individuals. Examples include privacy-preserving sharing of demographic data (e.g., US Census), medical data to support research (e.g., cancer

registries), or collecting click-stream data for vulnerability analysis (e.g., from browsers). Limiting the data release to only aggregated statistics does not guarantee privacy. An adversary could easily learn the confidential contents of a statistical database by creating a series of targeted queries and remembering the results.Lets consider the example of an adversary that asks the following set of aggregate queries on census database. Q1) Number of people with $gender =' M'$? Answer: 50. Q2) Number of people that have also $age > 50$? Answer: 10. Q3) Number of people that, in addition, have degree='eng'? Answer: 1. Now the adversary knows there is a single individual that satisfies the above constraints. The adversary can learn that individual's salary by using these constraints on the following query: Average salary of all individuals with $gender =' M'$ and $age > 50$ and $degree = eng$? This query will give the adversary exact salary of the individual person. Research shows that privacy properties in a database could only be preserved by considering each new query in light of all previous queries [8] which is shown to be an NP hard problem.

There are several privacy enhancing technologies that can be used to share valuable insights from user data. We provide some of the widely used techniques and their limitations as follows.

2.1.1 Data Suppression

Suppression techniques work by not answering queries if number of individuals are too small in order to prevent leakage of information about individuals. This technique does not prevent adversary from knowing confidential information about individuals. For example, lets say adversary knows some unique information about an individual e.g., the employee's age is more than 50, their gender is male and they have an engineering degree. Adversary can ask a series of questions 1) Average salary of all individuals, 2) Average salary of all individuals except whose $gender =' M'$, $age > 50$ and $degree = eng$? Based on these two queries

adversary can learn information about targeted employee salary.

2.1.2 De-Identification/Anonymization

De-identification or anonymization techniques work by replacing identifier fields (fields that contain information specific to an individual) with fictitious data such as characters or other data. There are several techniques that have been developed in the past e.g., K-anonymization, L-diversity, T-closeness. We provide details about more commonly used anonymization technique i.e. k-anonymization and discuss the possible privacy attacks.

K-Anonymization To achieve k-anonymity, the attributes of a dataset are suppressed/-generalized until all rows are identical with at least k-1 rows. Formally, an algorithm satisfies k-anonymization if it outputs only k-anonymous table. A table is considered k-anonymous if every record in the table has the same quasi-identifier values as $k - 1$ other records. These algorithms are found to be vulnerable towards different types of attackers [60, 48]. K-anonymity is a widely used privacy enhancing technology that can be used to release data to public. However it can be susceptible to multiple types of attack e.g., Homogeneity Attack, Background Knowledge Attack. Homogeneity Attack leverages the case where all the values for a sensitive attribute within a set of k records are same. In this case, even though the data was k-anonymized, it is possible to predict the exact sensitive value of the set of k records. Background Knowledge Attack leverages the association/relationship between one or more quasi-identifier values with the sensitive value to reduce the set of possible values for the sensitive value. There have been many such cases in the past that show that k-anonimization does not provide any guarantees for privacy. For example, it has been shown that publicly released Netflix Prize data set that contains anonymous movie ratings of about half a million subscribers of Netflix, an adversary can use deanonimazation techniques to identify this subscriber's record in the dataset with only a little bit of background knowledge about an

individual[49].

Similarly, privacy techniques such as l-diversity[40], t-closeness[36] which are refinement of anonymization techniques are also vulnerable in the face of a powerful adversary. We need data analysis algorithms that can mine aggregated personal data with provable guarantees of privacy for individuals even with strong and powerful adversary.

2.1.3 Differential Privacy

Differential privacy [13] has emerged as a widely used privacy definition with provable privacy guarantees. An algorithm is said to follow differential privacy given an input dataset $D \in \mathcal{D}$, if output of the algorithm does not change significantly, when a single tuple is added or removed from D. It is formally defined as follows:

Definition 2.1 (Differential Privacy (DP)). *A randomized mechanism $M : \mathcal{D} \rightarrow \mathcal{O}$ satisfies ϵ-differential privacy, if*

$$P[M(D) \in O] \leq e^{\epsilon} P[M(D') \in O] \tag{2.1}$$

for any set of outputs $O \subseteq \mathcal{O}$, and any pair of neighboring databases D, D' where D and D' differ by only one tuple, i.e., $\mid D \setminus D' \cap D' \setminus D \mid = 1$.

In this definition, ϵ is the privacy budget that controls the amount of privacy loss where $\epsilon \geq 0$. A higher ϵ value implies weaker privacy, whereas a lower ϵ value implies stronger privacy.

A Bayesian interpretation DP [26] is to bound the posterior odds of an adversary with respect to prior odds on whether a tuple x is in D and takes value $t \in \mathcal{T}$, where $\mathcal{T}$ is the domain of the tuples. The adversary's prior odds for the tuple x is defined as $\frac{P[x=t \wedge x \in D]}{P[x \notin D]}$, where

the numerator refers to the prior belief that x is in the database and takes value t and the denominator denotes the prior belief that x is not in the database. The posterior odds after observing an output o of the DP mechanism M, is expressed as $\frac{P[x=t \wedge x \in D|o]}{P[x \notin D|o]}$. As M satisfies ϵ-DP, we have the following guarantees, given non-zero prior beliefs for x and t,

$$|\ln(\frac{P[x = t \wedge x \in D|o]}{P[x \notin D|o]} / \frac{P[x = t \wedge x \in D]}{P[x \notin D]})| \leq \epsilon \tag{2.2}$$

Differential Privacy Properties

Differential privacy has important properties [33, 13] to allow the composition of multiple DP mechanisms.

Theorem 2.1 (Sequential Composition)**.** *Consider k algorithms $M_1, ..., M_k$ each satisfying ϵ_i-DP. The sequential execution of $M_1, ..., M_k$ satisfies $\sum_{i=1}^{k} \epsilon_i$-DP.*

Theorem 2.2 (Parallel Composition)**.** *Consider k algorithms $M_1, ..., M_k$, each satisfying ϵ_i-DP. The dataset D is partitioned into k disjoint parts and each M_i is executed on the i_{th} partition. Then the parallel execution of $M_1, ..., M_k$ satisfies $max(\epsilon_i)$-DP.*

Theorem 2.3 (Post-Processing)**.** *Let $M_1 : \mathcal{D} \to \mathcal{O}$ be an algorithm that satisfies ϵ-differential privacy. Then if an algorithm M_2 is applied to the output of M_1, then the overall mechanism $M_2 \circ M_1$ also satisfies ϵ-DP.*

Ex-Post Differential Privacy. It is possible to design DP algorithms that do not satisfy the differential privacy definition, but the output of such computations are private and can be quantified after the computation is finished. For example, consider an algorithm that repeatedly runs an ϵ-DP algorithm, until a stopping condition is met and the condition is defined by the output itself. This experiment does not satisfy ϵ-differential privacy for any fixed value of ϵ, since we do not know in advance how many times ϵ-DP algorithm will run. However, when the algorithm stops, we can see that it stopped after k runs so the ex-post

privacy loss for such algorithm will be $k\epsilon$. More details on Ex-Post differential privacy can be found in [37].

Differential Privacy Mechanisms

There are several DP mechanism proposed in the literature that can be used to achieve differential privacy. In the following, we present some of the most widely used mechanishms to achieve differential privacy.

Laplace Mechanism. The Laplace mechanism is one of the commonly used DP mechanisms and it achieves ϵ-DP by adding noise drawn from a Laplace distribution that is proportional to the *sensitivity* (Definition 2.2).

Theorem 2.4 (Laplace Mechanism (LM)). *Given a function $g : \mathcal{D} \rightarrow \mathbb{R}^d$, the Laplace Mechanism outputs g(D) + η, where η is a d-dimensional vector of independent random variables drawn from a Laplace distribution with the probability density function $p(x|\lambda) = \frac{1}{2\lambda}e^{-|x|/\lambda}$, where $\lambda = \Delta g/\epsilon$, and it satisfies ϵ-DP.*

Definition 2.2 (Sensitivity). *Given a function $g : \mathcal{D} \rightarrow \mathbb{R}^d$, the sensitivity of g is defined as the maximum L_1 distance between function outputs of any two neighboring databases D and D' that differ by only one tuple.*

$$\Delta g = \max_{\forall D, D'} \|g(D) - g(D')\|_1 \tag{2.3}$$

For instance, a counting query has a sensitivity of 1.

Exponential Mechanism. The exponential mechanism[42] can be used to provide differentially private answers to queries where responses of such queries are not numeric values. For instance "what colour of eyes is most common?" or "which country has the highest prevalence of heart disease?". It is also useful for constructing better mechanisms for numeric

computations like medians, modes, and averages. This work[42] provides formal definition of exponential mechanism and how to achieve ϵ-DP.

Randomized Response Mechanism. The randomized response is predecessor to differential privacy which was designed as a method used to be used in surveys[59] and can be used to achieve ϵ differential privacy. The basic concept of randomized response is that to answer a yes/no question, first, flip a coin. If the coin is heads, answer the question truthfully. If the coin is tails, flip another coin, if the second coin is heads, answer "yes"; if it is tails, answer "no". The randomization in this algorithm comes from the two coin flips. As in all other differentially private algorithms, this randomization creates uncertainty about the true answer, which is the source of privacy.

There are several other building blocks on differential privacy and readers may refer to [13] for more details. In the following, we describe important line of work in differential privacy research.

Answering Queries vs. Publishing Synthetic Data Depending analysis task at hand, a data analyst is either allowed to run multiple customized queries over the data in a differentially private manner[33, 34, 43] or differentially private synthetic data is generated once and all the analyst interactions are done over the differentially private data[19, 20]. Being able to answer multiple customized queries over data requires keeping track of ϵ. In this work, we consider the former method of analyzing the data in a differentially private manner.

Differential Privacy for Complex Data The standard definition of differential privacy is not well suited for data set that could be more complex and may have multiple types of entities that may be related to each other e.g., streaming data set where an individual may be represented by multiple rows in a table. There has been prior work that critically analyzes the privacy guarantees provided by differential private algorithms in terms of sensitive information disclosed to the adversary. These line of work (Pufferfish[26], Blowfish[21])

provides more generalized privacy definitions inspired by differential privacy which can be customized according to the complexity of data.

2.2 Related Work

As discussed above, most of the privacy preserving techniques work by adding noise/randomization or suppression of sensitive information. Hence, incorporating privacy can lead to inaccurate results when performing data analysis. In the context of Decision Support application, accuracy is of primary importance as without accuracy, the decision support applications will lead to incorrect decision i.e., defeating the very purpose of the application. In our work, we consider differential privacy(DP) to incorporate privacy in DS applications as DP provides strong privacy guarantees.

Accuracy-aware differentially private (DP) systems [47, 16, 37, 38] have been studied in the literature. These systems allow data analysts to specify their accuracy requirements for their queries/ applications while achieving bounded privacy loss. However, queries supported by these systems or their accuracy specifications do not directly match the need for decision support applications. GUPT [47] considers sample and aggregate framework which requires the query can be answered accurately from a data sample. Hence, it is not applicable to our aggregate threshold queries for decision support. APEx [16] and DPella [38] are able to answer aggregate threshold queries(basic decision support) with an accuracy guarantee, but this guarantee differs from the accuracy requirement of decision support query in terms of false negatives, and hence their algorithms also do not directly apply. The work by Ligett et al. [37] handles arbitrarily complex mechanisms and use empirical error of the mechanisms to pick epsilon, but the testing of the empirical errors requires additional privacy budget. Ligett et al. also introduced the notion of ex-post DP for accuracy-aware one-shot mechanisms. In our work, we extend this ex-post notion to predicate-wise DP, our fine-grained privacy

management framework for decision support.

Fine-grained privacy specifications have been considered previously at tuple level, like personalized DP[24] where each tuple has its own pre-set privacy budget; or at group level, like one-sided DP [29] that specifies a set of tuples are non-sensitive based on their values. Predicate-wise DP allows the tracking of the privacy budget at group level partitioned by the predicates. It generalizes one-sided DP, a case with only two groups. Though each tuple cannot leak its own privacy budget, the group-level (predicate-wise) budget can be leaked for budget accounting. Both personalized DP and one-sided DP do not have any accuracy-aware designed algorithms or a privacy metric for comparing group-level privacy budgets. Adapting them for decision support is not straightforward. Predicate-wise DP can also be treated as a development over the parallel composition property [43] of DP. This provides a formal framework for algorithm design over disjoint datasets.

In the context of privacy-preserving decision support using DP, Cuong et. al. [56] considered similar aggregate threshold queries. They focus on optimizing a fairness goal for resource allocation when observing that there are more disparities in decision errors for groups with closer aggregates to the thresholds. Hence, the algorithms do not apply to our queries. Furthermore, the paper did not take the accuracy-first approach, it is not clear how to set the privacy budget to achieve the desired fairness goal.

Chapter 3

IoT-Detective: Analyzing Differential Privacy For Decision Support in IoT domain

In this chapter, we study how state-of-the-art DP techniques perform for DS applicatons.

We focus on IoT(Internet of Things) applications in the context of a state-of-the-art IoT testbed (TIPPERS [45]) located at UC Irvine. In TIPPERS, data arrives in a stream and is used to develop dashboards based on the sensor data. These dashboard/applications are are can be used to drive decisions. We adapt a recently-proposed system, PeGaSus [5], which releases streaming data under the formal guarantee of differential privacy. We first describe Tippers and the application, and provide summary of PeGaSus and its privacy guarantees. We then discuss how we adapt PeGaSus for our context and conduct a study of its effectiveness in supporting DS applicaitons. In particular, we create a game called IoT-Detective, in which participants carry out visual data analysis tasks on private data streams, earning points when they achieve results similar to those on the true data stream.

We demonstrate the effectiveness of DP in such a task and privacy-utility trade-offs through demonstration study which we conducted at SIGMOD'18 conefrence.

3.1 Preliminaries

We briefly describe our IoT testbed, called TIPPERS, and the differentially private engine for releasing streaming data, PeGaSus.

3.1.1 TIPPERS

TIPPERS (**T**estbed for **I**oT-based **P**rivacy-preserving **PER**vasive **S**paces) is an experimental 6-story smart building testbed designed to study the numerous privacy challenges that result from fine-grained monitoring of building occupants and visitors using a diverse set of sensors [45]. To date, TIPPERS has installed 40 cameras, 64 WiFi APs, several hundred bluetooth beacons covering all major regions in the building, over a hundred smart plug meters to monitor energy consumption of connected devices, over six thousand HVAC sensors measuring airflow and ventilation as well as temperature at different parts of the building, and a large number of light and motion sensors. Data from these sensors flows through the TIPPERS system that fuses the underlying sensor data to produce mainly two higher-level data streams – PRESENCE, which monitors location of all individuals who are inside the building as a function of time, and ENERGY, which monitors energy usage at different spatial resolutions. The information managed by the TIPPERS database system is used to build a variety of applications from real-time awareness of resources, people, and events, to mechanisms to perform analytics on historical data.

The focus of the demo is on the *Building Analytics App*, shown in Figure 3.1. This app provides analytics about data gathered from multiple sensors in the building (e.g., occupancy,

temperature, and energy consumption). The user can view occupancy data for different time intervals and space granularities. The application is designed to gain an understanding of how the building is used as a function of time in order to better plan spaces and events, as well as to better control HVAC systems in order to be more energy efficient. For instance, patterns of building usage by occupants for different regions of the building could lead to customized HVAC settings that save energy without inconveniencing occupants. Likewise, occupancy data can also be used to determine if there are regions in the building that are under/over utilized and such information can lead to plans for better space management (e.g., understanding class rooms that are overflowing or underflowing or determining which lounge spaces are popular). The tasks we choose for our experimental game described as part of this demo are motivated by such real world needs of building analysts. For the context of this demo, the main focus is on occupancy data which is derived from PRESENCE data stream. The PRESENCE data stream has continuously been collected now for about two years, resulting in about 300 million location events since January 2016.

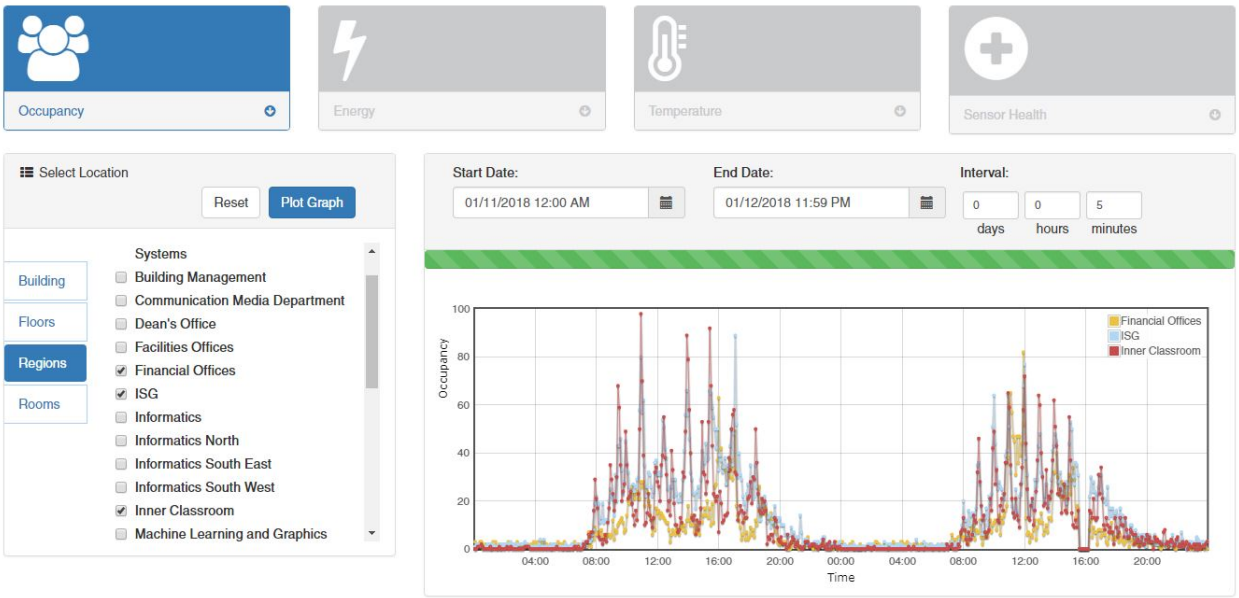


Figure 3.1: Screenshot of the Building Analytics app.

3.1.2 PeGaSus

PeGaSus is a novel system for releasing continuous query answers on real time streams under differential privacy [5]. PeGaSus assumes the input has been pre-processed into a stream of tuples (u, s, t) meaning user u was observed in state s at *logical* time t. The logical timestep captures a short window of time (e.g., 5 minutes). States correspond to events of the form "user u connected to a specific WiFi AP." Pre-processing ensures that, at each time t, a user can be in at most m states for some fixed and known m.

PeGaSus supports a variety of continuous queries over the data stream. The most basic query is the *unit counting query*, which corresponds to releasing the number of users in a given target state at each time point. It supports other queries over a single target state such as sliding window sum queries—which report aggregated counts over time windows—and event monitoring queries—which report whether or not a specific temporal event occurred (e.g., the number of connections exceeding a threshold). PeGaSus also supports queries over *multiple* target states (e.g., monitoring individual loads on each access point), and *aggregations* over states (e.g., monitoring loads aggregated over all access points on a floor of a building).

PeGaSus ensures event-differential privacy. Informally, this means that modifying the stream by adding or removing (up to m) tuples from a single user u at a single logical time t does not significantly change the output (quantified by privacy loss parameter ϵ). We refer the reader to the full paper [5] for a formal privacy statement, and its implications.

PeGaSus consists of three modules: a *Perturber*, which generates a stream of noisy counts, a *Grouper*, which privately partitions the stream into contiguous regions that have roughly uniform counts and a query specific *Smoother*, which combines the output of the *Perturber* and *Grouper* to generate the final estimate of the query answer at each time step. Only the *Perturber* and *Grouper* access the sensitive stream.

3.2 Demonstration Study

The demonstration study is based on IoT-Detective, a game where the demo participant plays the role of the building analyst and uses a tool similar to the Building Analytics app to explore the differentially private data and perform various analysis tasks. The objective of the game is to perform analysis as accurately as possible and achieve the highest score across all attendees.

The *target group* of this demonstration were the SIGMOD' 18 conference attendees. The players did not need to have any prior knowledge of differential privacy.

3.2.1 The IoT-Detective Game

The demo consists of a game where a player — in this case, an attendee — is challenged to identify a real world event or pattern using tools provided by TIPPERS on the differentially private data, much like a building manager might in a real-world deployment. This type of event monitoring can be used for making decision e.g. regarding better space utilization.

To play the game, the demo participant interacts with the IoT-Detective game (see Figure 3.2), which is very similar to the Building Analytics app, but has some additional game-specific features, such as a timer, leader board, etc. The game is played in rounds and a player can play as many rounds as possible in the allotted time. In each round, the player is given a specific task which requires answering a factual question about types of events during certain time periods (e.g., to identify the most likely time a weekly meeting occurs). The player can then use the app to navigate through the data to identify the relevant (differentially private) data streams and temporal windows and derive an estimate for the answer. The accuracy of the answer is measured in terms of the difference between the player's estimate and the correct answer on the true (non-differentially private) data.

Players will be rewarded with points after accurately accomplishing each task. The amount of points will depend on a combination of the accuracy of their estimate, the time taken to complete the task, and the number of tasks they have completed (to incentivize participants to play more than one round). The demo will track player points and maintain a leader board to encourage friendly competition.

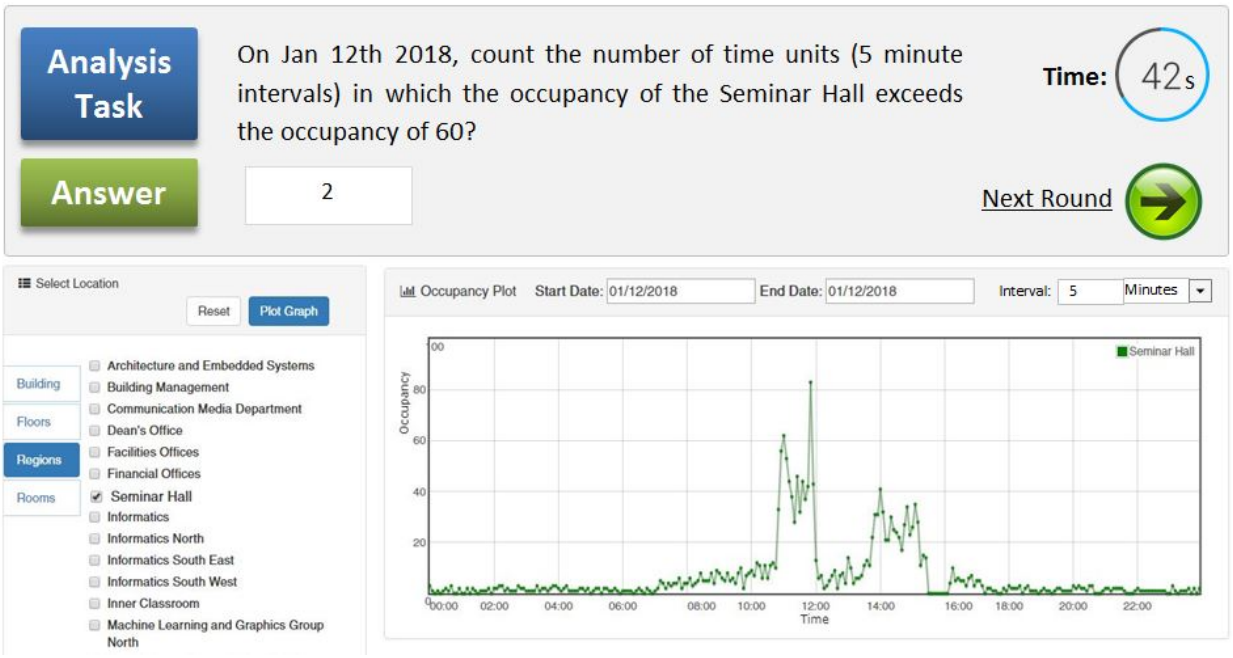


Figure 3.2: Screenshot of the IoT-Detective game interface.

The accuracy of a player's answer depends on two primary factors. First, it depends on the player's ability to successfully navigate the user interface—thus, the demo is serving as a valuable user test to see if the tool is intuitive and effective for these analytics tasks. Second, it depends on the amount of noise injected into the data stream by PeGaSus. By varying the privacy parameters across users and rounds, we gathered some preliminary data on how much noise is tolerable for varying tasks—thus exploring the practical viability of differential privacy in streaming data settings.

Example Tasks The Building Analytics Game app is initialized with a differentially private dataset that reports occupancy information at 5 minute intervals for each room in the

building.

An example of a task might be: "On [specific date], count the number of time units (5 minute intervals) in which the occupancy of the [main conference room] exceeds [60]." The parts in brackets can be varied to generate different versions of this task. The motivation for this task is that building managers may wish to detect when a room exceeds its maximum permitted occupancy under fire code regulations, or identify rooms/times in which space is heavily-utilized. Players will be asked to perform a variety of tasks. The following are additional illustrative examples:

- *High occupancy regions.* The rooms can be naturally organized into a fixed set of regions e.g., Facilities Offices, Department of Informatics, etc. This task is to identify which region is the most occupied at night (6pm to 6am) on [a particular day]. Most occupied could mean average number of people are highest during night time. The motivation for this task is better HVAC control at late hours when there are fewer occupants in the building. The accuracy measure can be the difference in rank between the user's choice and the true answer.
- *"Broken" sensors.* We presume here that when a sensor breaks, it no longer senses its environment and continuously reports a constant value, such as zero. Thus, we formulate the task as follows: identify the earliest point in time in which [a particular sensor] starts continuously reporting zero. This is motivated by the practical challenges that building managers face with equipment maintenance. The accuracy measure is distance to the actual time the sensor breaks (we will artificially modify the dataset to make a sensor appear broken).
- *Occupancy at routine events.* The task is to identify the start time of a regularly occurring event in a particular room e.g. start time of a lecture in a classroom. The motivation is to facilitate better scheduling or detecting events that deviate from a schedule. The accuracy measure is the distance between the player's estimate and the actual start time

of the event.

Post-demo empirical evaluation The demo system recorded traces of the games of all participants. Based on immediate feedback to users on their success and analysis of the trace, this demo helped us to better understand the impact of the privacy mechanism on the usefulness of visually displayed stream data. Although a conference demo is inappropriate as the basis of a formal, controlled user study, the experience of gathering results gave us some idea of challenges in DS applications due to privacy-utility tradeoffs. The usefulness of privacy preserving data can vary depending on the underlying data set and the privacy level i.e. ϵ. If the privacy level is too high, the users were unable to perform the analysis tasks correctly. It is hard to determine what privacy level is too high to render the privacy preserving data useless as it can depend on the data and the analysis task. Hence, without setting appropriate privacy level, the decision support application may not work due to low utility of the data and setting the right privacy level to make accurate decisions is a challenging problem.

Chapter 4

Empirical Evaluation of Diverse PETs to Publish Smart Space Occupancy Data

This chapter considers an experimental evaluation of diverse privacy enhancing technologies (PETs) to publish occupancy data derived from continuous sensor streams in emerging smart buildings that can be highly valueable for decsion making. Ensuring individual's privacy in such a context, specially with formal privacy guarantees, is a hard challenge that has attracted significant research interest [25]. Different techniques that offer different levels of privacy guarantee have been proposed - e.g., techniques that are derivative of differential privacy define privacy goals (such as protect published data from revealing user's precise location or participation on an event at any small interval of time) and offer formal privacy guarantees (for the defined goals). Others, motivated by k-anonymity and techniques such as [2] offer practical privacy but do not offer formal privacy guarantees.

Our study evaluates the practical implications to individual's privacy, defined as the certainty

at which the location of an individual's location at a given point, of publishing a continuous occupancy map generated from streaming sensor data. We present our methodology for computing such adversarial guess based on aggregated data published while utilizing three different PETs. Additionally, we present realistic adversaries in our context (from a rogue building administrator to a group of students trying to stalk others). The empirical evaluation based on our methodology is performed in a real dataset containing 3 months of occupancy levels of a University building derived from connectivity events captured by the 64 WiFi access points within the building.

4.1 Introduction

The adoption of Internet of Things (IoT) devices (i.e., sensors and actuators) in our daily lives is transforming our spaces (e.g., homes, office buildings, cities) into *smart spaces*. This opens up endless opportunities to provide smart services based on data captured from the space itself and people within it that can result in benefits to users from the point of view of automation or comfort. In particular, a potentially useful smart space dataset that is currently being used in a plethora of applications is that of occupancy levels within the space. This data can be obtained from multiple different sources such as WiFi Access Points [9, 62], beacons [7], video cameras [55], etc. This dataset can be leveraged to, among others, optimize thermal comfort in the space [27] as well as demand response [28] or help in evacuating the space in case of an emergency [63]. Similarly, occupancy levels can enable inhabitants of the building optimize their activities by, for example, detecting which meeting rooms are available at the moment or which food court has the less number of people in queue. However, as such occupancy information is obtained by monitoring location of individuals there is a risk that analysis of occupancy data can violate individuals privacy [51]. For example, patterns of movement, affinity towards spaces and other people in the building

could be potentially learnt from such occupancy information.

In this chapter, we focus on the continuous publishing of statistical data: stream of occupancy counts of different regions within a building that are published continuously in order to generate a dynamic heatmap of occupancy. This heatmap is further utilized by users to understand different characteristics of the space[1]. Regarding the possible information disclosure, we do not consider disclosure of presence of an individual in the dataset as a problem since in scenarios such as ours (a University building) this information is typically publicly available (e.g., enrolled students, faculty, and staff members in the different departments). Instead, we focus on protecting the location of a user at time t. We consider this as a fundamental piece of private information that can be further utilized to extract information about the habits, patterns, social network, events in which the user participated, etc.

The most widely used privacy method to publish continuous statistical data is differential privacy [11] according to [25]. These methods differ in the specific level of protection they provide (e.g., any single event of an individual vs. all the events vs. sequence of event), the kind of attacks they protect against (e.g., complimentary release vs. data dependency attacks), category of data or publishing mode (e.g., batch vs. streaming). The use case of interest to us is the periodic generation of a heatmap of occupancy from which an attacker must not be able to learn the current location of an individual. We thus focus on approaches for protecting leakage of an event of an individual's from streaming/infinite data [3, 52, 35, 6] (see [25] for a comparison of those methods). In particular, PeGaSus [6], an algorithm for event-level diffentially private stream processing, has been utilized in a similar context to our use case to release occupancy counts in a smart building [17]. In addition to differential privacy methods, another mechanism traditionally used in publishing of location data is k-anonymity [54] which aims at obfuscating the identity of the individuals in the dataset

[1]The specific use of the heatmap is beyond the scope of this work as different users/profiles might have very different use cases in mind (e.g., a student might be looking for a study room available, whereas the building administrator might be monitoring whether fire code violations are taking place).

and can potentially be used before performing the aggregation to compute occupancy levels. Similarly, methods based on *mixed zones* [2], which anonymize user identity by restricting the positions where users can be located, has been explored in the context of location privacy preservation.

The previous techniques (a representative but not comprehensive set of traditional PETs used for this purpose) have different implications. First, is the issue of utility of the published dataset after applying such PETs. Addition of noise to the occupancy counts or anonymization of source data before computing occupancy might impact applications (in some situations severely –e.g., if the goal is to perform automatic fire code violation control–). However, we can assume that different PETs can be tuned (e.g., by means of their parameters) to achieve a specific utility requirement when the final application requirement is clear. Second, each technique has a different impact on individuals privacy. From formal privacy guarantees, as in the case of DP, to weaker guarantees, as in the case of anonymization. Additionally, different PETs have different underlying assumptions about possible attack models. For instance, DP techniques are aimed to consider attackers with an almost unlimited knowledge which for release of data to a limited/controlled population could result in an overestimation of the adversarial strength. Third, in some situations the priors computed depending on the underlying context could be so strong that would make the influence of the PET small (e.g., imagine an office space with the same daily set of people in which their trajectories are highly predictable). Fourth, the robustness of the data published by non-deterministic PETs (e.g., based on differential privacy) will depend on the specific perturbation performed which, even if on average remains constant, in specific situations can be better or worse. All of the previous points make it challenging to understand the implications of each data releasing technique for each specific context.

In this chapter, we present an empirical evaluation of the practical privacy implications of three privacy preserving methods (two techniques based on differential privacy and a

technique based on k-anonymity and mixed zones). The study is perform in the context of a real building which captures connectivity data streams from WiFi access points and publishes continuous occupancy data (see Sec. 4.2). For the purpose of evaluation, we propose a methodology to compute privacy loss of individuals (see Sec. 4.3). Typically, privacy analysis in the literature is done in the context of differential private mechanisms which are compared based on their epsilon parameter as a bound on privacy loss. In our case, since we want to compare also techniques which are not based on differential privacy, we need a privacy metric to represent real privacy loss of individuals. We also present a posterior computation method (see Sec. 4.4) which considers both the computation of posteriors with and without privacy mechanisms. This is required to compare the privacy loss when publishing real occupancy counts with the loss when publishing data with each of the PETs analyzed. Based on the previous we present a set of experiments (see Sec. 4.5) and show the results per privacy technique considering realistic adversaries in our context such as a rogue administrator or a group of student trying to stalk others (see Sec. 4.6).

4.2 Background

In this section, we describe the dataset and the evaluated privacy techniques for this study.

4.2.1 Data

The data used in this study comprises occupancy levels of different spaces along time. We have used a fragment of the dataset collected by Tippers[44] for building Donald Bren Hall at University of California, Irvine. This dataset contains Wifi connectivity data of 3 months (Feburary to April 2018) containing 36,436 number of devices connecting to 64 Wifi access points. Over the 3 months period, these devices generated 3,895,732 connectivity events.

System S generates occupancy data based on connectivity events captured by standard WiFi Access Points (APs) using SNMP traps when WiFi-enabled devices (e.g., smartphones) connect to any of the 64 APs in the building. An event occurs when the device initially connects with a specific AP, and then such events continue to occur periodically. Device re-connection triggers if the signal strength between the device and the connected AP decreases. It also occur stochastically even when devices are stationary and the signal strength is stable depending upon the type of device, device manufacturer, type of OS, and network characteristics. Such events are stored in a table to which we will refer as "Observation" with the schema $< MAC_{device}, MAC_{AP}, timestamp >$ where MAC_{device} and MAC_{AP} represent the MAC address of the user device and of the WiFi AP, respectively. The first step in the processing of the events is to enrich them to generate a table to which we will refer as "Presence" with the schema $< UserID, LocationID, StartTS, EndTS >$. Where $UserId$ is an id used to anonymize the MAC address of the device, $LocationID$ is the id of the region of the building covered by MAC_{AP}, $StartTS$ represents the time when the event was generated (and therefore is equal to $timestamp$), and $EndTS$ is an attribute used to represent the validity of the event (initialized as $timestamp + 10$ where 10 minutes is a validity obtained experimentally after analyzing connectivity patterns of different devices). In the following, we will assume that the anonymization process involves hashing the MAC address deterministically (a same MAC receives the same hash) unless another technique is explicitly mentioned. Indeed, we will show when describing the privacy techniques that some of them apply a different type of anonymization. When a new event arrives at time t ($< UserID_i, MAC_{AP_i}, t >$) we look for the last presence tuple generated for that same user ($< UserID_i, MAC_{AP_j}, t - k >$) and update its $EndTS$ to t if $k \leq t - k + 10$. Finally, there is a table to which we refer as "Occupancy" which stores the occupancy of different locations with the schema $< LocationID, count, timestamp >$. Every five minutes (i.e., time t) a query is executed in the "Presence" table to count the number of distinct $UserID$ in each $LocationID$ where $StartTS < t < EndTS$ and the results are inserted into the

"Occupancy" table.

Occupancy Heatmap. This stream of occupancy counts of different regions within a building that are released using PETs to allow the user to generate a dynamic heatmap of occupancy. The heatmap assigns colors ranging from $c_1, c_2, ..., c_n$ (in order of low to high occupancy) where $n = 10$ to each location l at a time t. We compute each color c_i based on the distance from average occupancy μ of the place l at the time t as observed in the past data. If the occupancy is above $\mu + 2\sigma$, the occupancy is the highest i.e., c_{10}. Similarly, if it is below $\mu - 2\sigma$, it gets the lowest color value c_1. The rest of the colors are assigned based on dividing the range ($\mu - 2\sigma$,$\mu + 2\sigma$) equally.

4.2.2 Techniques

We evaluate the following techniques to release privacy preserving aggregate level data (occupancy).

Differential Privacy: The goal of a differential private algorithm is to prevent and adversary from telling whether a particular individual's information was used to derive its answer [11]. In our set up, a differential private mechanism is used to output occupancy counts based on the presence data. Therefore, the output is a noisy version of the occupancy levels of each space. Differential private algorithms offer strong privacy guarantees, under certain assumptions. However, the utility of the output data directly depends of the privacy level (controlled through their ϵ parameter). In particular we use two differentially private mechanisms. First, we use a standard Laplace mechanism to add Laplace noise to the counts of occupancy in the occupancy table (we will refer to this technique as simply *Laplace* in the rest of the chapter). Second, we use the PeGaSus [6] mechanism which is an instance of DP for streaming data.PeGaSus ensures event-differential privacy which means modifying the stream by adding or removing tuples from a single user u at a single time t does

not significantly change the output of the mechanism. PeGaSus consists of a Perturber, a Grouper and a Smoother. The Perturber generates a stream of noisy counts based on laplace noise. The grouper privately partitions the stream into contiguous regions that have roughly uniform counts and the Smoother combines the output of Perturber and Grouper to generate the final estimate of the count at each time t. PeGaSus just like any Differential Privacy mechanism quantifies the privacy loss parameter in terms of ϵ. (we will refer to this technique as *PeGaSus* in the rest of the chapter).

k-anonymity: MAC randomization at the client side, which generates a new MAC address for a device when connecting to a network, to prevent tracking has been shown to fail as it becomes stable when connected to it [41]. The goal of this technique is to perform such randomization at the server side when the connectivity events are captured. Instead of a deterministic hashing of the MAC address, in this technique the hashing is defined to be salted with a salt that changes periodically every k minutes (thus each salt has a TTL –Time To Live–). This implies that for a user that remains connected to the same WiFi AP for a longer period than the TTL if two connectivity observations are captured and the difference between their timestamps is greater than the TTL, then the $UserID$ associated to them will be different even if the MAC address of the device is the same. The rest of the process to translate connectivity events to presence and occupancy is the same as explained before. This technique is aimed at linkage attacks to protect user trajectory information. Notice that the occupancy data will be less accurate given that people can be counted several times depending on the TTL. This technique has no formal guarantees of privacy w.r.t. released counts, as counting does not depend on the object identities anyway.(we will refer to this technique as TTL in the rest of the chapter).

4.3 Methodology

In this study, we are targeting systems where location of individuals is collected, but only occupancy data is available through the queries. For this class of systems, we want to evaluate the privacy implications of a given technique following the methodology summarized in Figure 4.1. Notice that the methodology is general and can be used in other contexts too. Therefore, along with the explanation of each component we explain how they have been instantiated in our scenario.

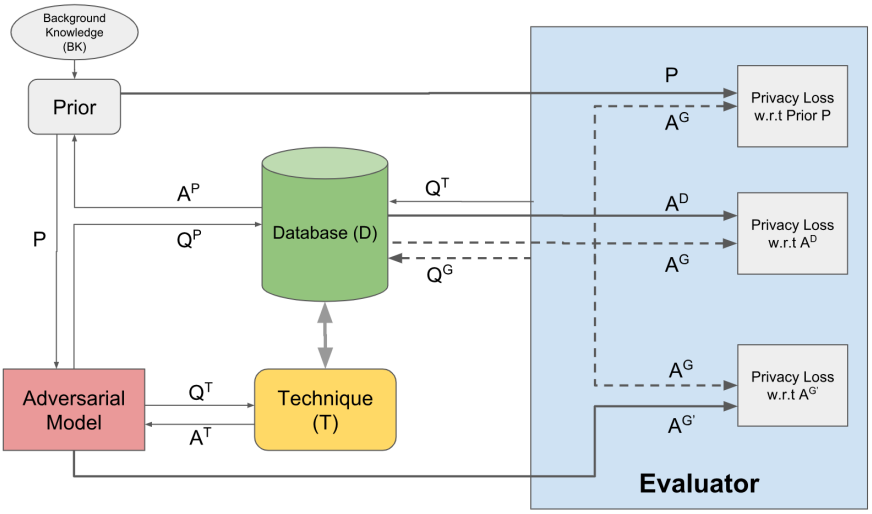


Figure 4.1: Methodology defined.

We define a database D that contains all the data captured by the smart building (see Sec. 4.2.1). The data, as well as its schema, is fixed. We define $\mathcal{T}$ to be the set of Privacy Enhancing Techniques utilized for answering the adversary's allowed queries (see Sec. 4.2.2). Thus, we consider that a technique $T \in \mathcal{T}$ answers the query Q^T as A^T in a privacy preserving manner after consulting D.

Adversarial Model. As an adversary, we consider a tuple of algorithms whose goal is to retrieve/guess information about individuals in the dataset. We consider that such information cannot be directly accessed by the adversary by posing queries on the dataset. Also, we consider that the adversary can pose some queries on the dataset. More formally, we

define such tuple of algorithms for: 1) Creating the prior P by asking questions Q^P, that the adversary can pose on D, in addition to his/her background knowledge, for prior creation. 2) Computing the guess $A^{G'}$ to the question he/she wants to retrieve/guess from D by posing questions Q^T to the technique and using answer to Q^T (i.e., A^T) and the prior P. The goal A^G that the adversary wants to retrieve/guess from D that can be retrieved by posing questions Q^G may be specified as a set of SQL queries against the database D. Notice that Q^G cannot be posed on D by adversary.

In the context of our study, we consider that the adversarial prior P is created by observing past occupancy as well as presence data Q^P. Also, the adversary aims at guessing (A^G) the location of individuals at a time t. The adversary may have access to some background knowledge about some users location at time t (see Sec. 4.5 for more details about the specific background knowledge of different attackers). The adversary is allowed to pose queries (Q^T) to obtain occupancy counts (A^T) through technique T at the given time t.

Privacy Metric. As the goal is to determine the practical privacy implications of technique T, we define a measure based on privacy loss. This is defined as the number of people for which the adversary can guess the correct location at time t with a specific confidence class (in our experiments we defined six levels of confidence classes). First, we compute the adversary's guess based on prior P. This measure indicates how good is the adversary's guess of A^G based on just the prior. Then, we computed the adversary's guess A^D based on answers to Q^T without using any technique. This measure indicates how good is the adversary's guess of A^G if the adversary was allowed to run query Q^T over database D without any privacy preserving mechanism T. Finally, we compute the adversary's guess $A^{G'}$ based on prior P and answers to Q^T using technique T. This measure indicates how good is the adversary's guess of A_G based on the prior (P) and answers to queries Q^T using privacy preserving mechanism T.

4.4 Posterior Computation

In this section we present our method to compute posterior for an adversary with and without privacy mechanisms.

4.4.1 Posterior without Privacy Techniques

Let $\mathcal{U}$ be the set of users, $\mathcal{L}$ the set of locations, and $\mathcal{T}$ the set of observed timepoints. For all $u \in \mathcal{U}$, $l \in \mathcal{L}$, $t \in \mathcal{T}$, let p_{ul}^t denote the prior probability of the user u being in the location l at the timepoint t. Now fix a particular timepoint t. Assume that, for each location l, the attacker has obtained the total count c_l^t of users that are in location l at time t. We want to see how the probability p_{ul}^t changes after the attacker observes the counts $(c_\ell^t)_{\ell \in \mathcal{L}}$. Let C_ℓ^t be the random variable corresponding to occupancy of room ℓ at time t. Define an event $C := \bigwedge_{\ell \in \mathcal{L}} (C_\ell^t = c_\ell^t)$. Let the notation "$u \in l@t$" denote the event "the user u was in location l at time t". Using Bayesian inference, we get

$$\mathbf{Pr}[u \in l@t \mid C] = \frac{1}{1 + \frac{\sum_{l \neq \ell \in \mathcal{L}} \mathbf{Pr}[u \in \ell@t \wedge C]}{\mathbf{Pr}[u \in l@t \wedge C]}} \ . \tag{4.1}$$

To compute (4.1), we need to estimate $\mathbf{Pr}[u \in l@t \wedge C]$ for all $l \in \mathcal{L}$. This can be done as follows. The user u is assigned the location l. The remaining $|\mathcal{U}| - 1$ users need to be distributed to locations in such a way that the resulting occupancies would satisfy C. That is, the users are partitioned into $|\mathcal{L}|$ disjoint sets S_ℓ ($\ell \in \mathcal{L}$), where $|S_l| = c_l^t - 1$, and $|S_\ell| = c_\ell^t$ for $\ell \neq l$. The quantity $\mathbf{Pr}[u \in l@t \wedge C]$ can be computed by summing up the probabilities of all possible partitionings of users to such sets, which gives us

$$\mathbf{Pr}[u \in l@t \wedge C] = p_{ul}^t \cdot \sum_{\substack{(|S_\ell| = c_\ell^t)_{\ell \neq l}, |S_l| = c_l^t - 1 \\ \forall k, \ell : S_k \cap S_\ell = \emptyset, \forall \ell : u \notin S_\ell}} \prod_{\ell \in \mathcal{L}} \prod_{v \in S_\ell} p_{v\ell}^t \ .$$

We could directly compute the sum over all possible partitionings, but it would be computationally too expensive. We need to make some additional assumptions to simplify the computation.

Non-individualized distributions.

Let us assume that we have $p^t_{ul} = p^t_{vl}$ for all users $u, v \in \mathcal{U}$, i.e., the distribution of locations does not depend on the individuality of a particular user. This is reasonable in the case where the potential attacker only knows how an average user behaves in general, but does not distinguish between them, e.g., the attacker learned this by a prior observation of occupancy counts for a certain period of time. Let n^t_u be the total number of users recorded at time t. The rooms are occupied according to multinomial distribution: if we order the users and assume that the first c^t_1 go to S_1, the next c^t_2 got to S_2, etc, there are $(n^t_u - 1)!$ possibilities to rearrange the users, and since the ordering inside S_j does not matter, we get $\frac{(n^t_u-1)!}{c^t_1!\cdots(c^t_l-1)!\cdots c^t_{n_l}!}$ possible partitionings for $n_l = |\mathcal{L}|$ rooms. We get

$$\mathbf{Pr}[u \in l@t \wedge C] = c^t_l \cdot \frac{(n^t_u - 1)!}{c^t_1! \cdots c^t_{n_l}!} \prod_{\ell \in \mathcal{L}} (p^t_\ell)^{c^t_\ell} \ , \tag{4.2}$$

which gives us the posterior probability

$$\mathbf{Pr}[u \in l@t \mid C] = \frac{1}{1 + \frac{\sum_{l \neq \ell \in \mathcal{L}} c^t_\ell}{c^t_l}} = \frac{c^t_l}{n^t_u} \ . \tag{4.3}$$

This is quite an intuitive result, since if all users are treated equally, then any user will most likely be located in the most popular place, even if the prior probability of being there is very small. In fact, the posterior probability does not depend on the prior probability at all, but only on the counts, as the prior probability only defines the distribution of room occupancy, which is overridden by actual counts. More detailed derivations of Eq. 4.2 and

Eq. 4.3 are given in App. B.1.

Increasing attacker's knowledge.

If the attacker only knows the counts, his/her probability of guessing will be quite low even if no privacy mechanism is used. In reality, it is unlikely that the attacker has no other information at all. It is quite possible that he/she already knows the location of some people at the time of the attack. For example, some locations may be observed by the attacker directly (e.g., if the attacker is physically located in the space, he/she could observe who is around him/her) or through security cameras (e.g., for an attacker with access to the security camera system). We will refer to those areas for which the attacker has information about who is located inside of it at the time of the attack as *open regions*. Considering such open regions reduces the total number of locations where the victim could potentially be, so the posterior probability increases. The prior probability will also change, and will be scaled according to the number of open regions.

4.4.2 Posterior with a Privacy Technique

Let $\mathbf{X}$ be the random variable representing attacker's opinion about the input x, and $\mathbf{C}$ about the true output c (without noise). Let $\mathbf{Y}$ be the random variable representing the noisy output y, and $f_Y(\cdot)$ its probability density function. We let x denote the part of the input, guessing which is the attacker's goal, i.e., the location of the victim. Let X be the total space of possible values of x, i.e., all possible locations. The particular timepoint that we consider is implicit, and we do not use it in the notation.

In Sec. 4.4.1, we have shown how to compute $\mathbf{Pr}[u \in l \mid \mathbf{C} = c]$ for a count histogram $c = (c^t_\ell)_{\ell \in \mathcal{L}}$. In this case, $\mathbf{X} = x$ iff $u \in l$, so we can use the results of Sec. 4.4.1 to compute

$\mathbf{Pr}[\mathbf{X} = x|\mathbf{C} = c]$. We can use these results also to compute $\mathbf{Pr}[\mathbf{X} = x|\mathbf{C} = c, \mathsf{A}]$ where A is the additional knowledge that comes from opening some regions to the attacker. We want to estimate $\mathbf{Pr}[\mathbf{X} = x|\mathbf{Y} = y, \mathsf{A}]$.

A worst-case bound for Laplace mechanism

First of all, let us discuss a known upper bound on posterior probability for Laplace mechanism, taken from [31]. Assume that the attacker already knows the location of all other users except the victim. Let X be the set of possible choices for the attacker, i.e., locations of the victim. Laplace mechanism parametrized by ϵ gives us an upper bound $f_Y(y \mid \mathbf{X} = x) \leq e^{\epsilon \cdot |x-x'|} f_Y(y \mid \mathbf{X} = x')$ for all $x, x' \in X$. Using Bayesian inference, for all $y \in Y$, we can write

$$\begin{aligned}
\mathbf{Pr}[x|\mathbf{Y} = y] &= \frac{f_Y(y \mid \mathbf{X} = x)\mathbf{Pr}[x]}{f_Y(y)} \\
&= \frac{1}{1 + \frac{\sum_{x' \in X \setminus \{x\}} f_Y(y \mid \mathbf{X}=x')\mathbf{Pr}[x']}{f_Y(y \mid \mathbf{X}=x)\mathbf{Pr}[x]}} \\
&\leq \frac{1}{1 + e^{-\epsilon} \frac{\sum_{x' \in X \setminus \{x\}} \mathbf{Pr}[x']}{\mathbf{Pr}[x]}} \quad ,
\end{aligned}$$

so, in our case study an upper bound on posterior guessing probability is $1/(1 + e^{-\epsilon} \cdot (1 - p^t_{ul})/p^t_{ul})$, where p^t_{ul} is the prior probability of user u being in location l at time t.

One disadvantage of the obtained upper bound is that we assumed a very strong attacker who already knows locations of all other users. We could try to increase the number of unknown users, which changes the definition of X. This would decrease the prior, but at the same time we would get $e^{-\epsilon k}$ instead of $e^{-\epsilon}$, where k is the number of unknown users. The

exponent grows too fast with k. Hence, we cannot experiment with adversarial knowledge parameter A to get smaller levels of posterior probability.

Another problem is that the obtained upper bound is very generic and does not depend on the true query output $q(x)$, so it approaches 1 as noise approaches 0, and we cannot use it to evaluate a *particular query output*. Intuitively, if the privacy mechanism releases $q(x) + \eta$ for some randomly sampled η, then the attacker cannot get more advantage in guessing x than from observing $q(x)$. Hence, we are looking for other approaches that would give us an upper bound $\mathbf{Pr}[x \mid q(x)]$ on posterior guessing advantage.

Posterior for a particular noisy output instance

Let us try to evaluate the quantity $\mathbf{Pr}[x|y, \mathsf{A}] = \mathbf{Pr}[x|\mathbf{Y} = y, \mathsf{A}]$ directly. Let C be the set of all possible true outputs (e.g., count histograms). Using chain rule, we can write it out as

$$\mathbf{Pr}[x|y, \mathsf{A}] \;=\; \sum_{z \in C} \mathbf{Pr}[x|y, \mathbf{C} = z, \mathsf{A}] \cdot \mathbf{Pr}[z|y, \mathsf{A}] \;\; .$$

This equality can be viewed as an attacker making a guess z about the real output $q(x)$ and checking how likely this z could be obtained from the noisy output y. After the attacker has selected z according to y, it makes it guess purely from z and the additional knowledge A, so $\mathbf{Pr}[x|y, \mathbf{C} = z, \mathsf{A}] = \mathbf{Pr}[x|\mathbf{C} = z, \mathsf{A}]$. To estimate $\mathbf{Pr}[z|y, \mathsf{A}]$, the attacker takes into account the likelihood of the noise that would turn z into y, as well as the probability of z itself. It can be done using Bayesian inference $\mathbf{Pr}[z|y, \mathsf{A}] = \frac{f_Y(y|\mathbf{C}=z,\mathsf{A}) \cdot \mathbf{Pr}[z \mid \mathsf{A}]}{f_Y(y \mid \mathsf{A})}$. Since A only contains knowledge about the data, and not the distribution, we have $f_Y(y|\mathbf{C} = z, \mathsf{A}) = f_Y(y|\mathbf{C} = z)$, which can be computed from the noise distribution. The quantity $\mathbf{Pr}[z \mid \mathsf{A}]$ can be computed from prior probabilities, taking into account the additional knowledge. From these two

quantities, we can in turn compute $f_Y(y \mid \mathsf{A}) = \sum_{z \in C} f_Y(y|\mathbf{C} = z)\mathbf{Pr}[z \mid \mathsf{A}]$. We get

$$\mathbf{Pr}[x|y, \mathsf{A}] = \frac{\sum_{z \in C} \mathbf{Pr}[x|\mathbf{C} = z, \mathsf{A}] \cdot f_Y(y|z)\mathbf{Pr}[z \mid \mathsf{A}]}{\sum_{z \in C} f_Y(y|z)\mathbf{Pr}[z \mid \mathsf{A}]} \ . \tag{4.4}$$

Intuitively, we want that our estimated posterior probability would stay between the prior $\mathbf{Pr}[x \mid \mathsf{A}]$ and the probability $\mathbf{Pr}[x \mid c, \mathsf{A}]$ of guessing from the true count c. We state and prove this property in App. B.2.

Posterior for a particular true output instance

Fixing a particular $y \in Y$ can make the attacker seem too successful or too unlucky, depending on the $y \in Y$ that we have got. Knowing a particular distribution on Y, we may estimate how much the attacker may guess in average for a particular output $c := q(x) \in C$. First of all, we could directly compute the average posterior probability for all possible outcomes y as

$$\mathbf{Pr}[x|\mathsf{A}] = \int_Y \mathbf{Pr}[x|y, \mathsf{A}] \cdot f_Y(y|c) \ dy \ . \tag{4.5}$$

This approach is good if the resulting integral has closed form, or at least can be approximated efficiently. However, in practice it may be computationally hard to compute the integral precisely. Alternatively, we can *empirically* compute the posterior probability on many instances of randomly generated noise.

4.5 Experiments

The experiments are performed on presence and occupancy records for 3 months (February, March, April, 2018), which comprises $N = 89$ days. We use the first $N - 1$ days of presence

data for constructing prior probabilities. We then use the last N-th day of the occupancy table to compute posterior probabilities, showing how attacker's guesses improve compared to guessing from prior. Our analysis consists of the following steps.

1. We split a day into 10-min spans. This gives us $T = 144$ time units per day.

2. For each 10-min span t of a day, for each location l and each user u, we compute prior probabilities $P_{prior(u,t,l)}$ from the first $N-1$ days.

3. Based on the prior probabilities $P_{prior(u,t,l)}$ and the noisy occupancy counts generated by a particular privacy mechanism $\mathcal{M}$ on the N-th day, we compute the posterior probabilities $P^{\mathcal{M}}_{noisyOcc(u,t,l)}$. Among other mechanisms, we estimate $\mathcal{M}(x) = x$ (guessing from true occupancy counts) and $\mathcal{M}(x) = \perp$ (guessing just from prior).

4. Let $P_{true(u,t,l)} \in \{0, 1\}$ be the actual user locations, i.e., $P_{true(u,t,l)} = 1$ iff u was in location l at time t. Compute the following for each user u and time t:

$$P^{\mathcal{M}}_{guess(u,t)} = \sum_{l \in \mathcal{L}} P^{\mathcal{M}}_{noisyOcc(u,t,l)} \cdot P_{true(u,t,l)} \ .$$

5. Plot aggregate privacy metric: how many people have been localized correctly from $P^{\mathcal{M}}_{guess(u,t)}$ with a probability within certain range, excluding those who have been localized with similar confidence purely from prior.

4.5.1 Prior distribution

For a fixed timepoint t, the attacker receives a prior distribution of location of an "average user", expressed as $p^t_l \in [0..1]$ for all $l \in \mathcal{L}$, where $\sum_{l \in \mathcal{L}} p^t_l = 1$. The values p^t_l are computed from the training period using counting. That is, for each time of day t, we count the total number of users m^t_l recorded in region j at time t, and define $p^t_l = m^t_l / \sum_{l \in \mathcal{L}} m^t_l$. Hence, the

prior defines an expected distribution over region counts for different times of the day. Here we use the meta-knowledge that similar pattern repeats periodically. The priors would be more precise if we generated, for instance, a separate prior for each weekday, or found some more interesting meta-data like exceptional holidays which should be discarded as outliers. The problem is that, the more we partition the prior, the less data we have to estimate it. Also, too strong prior may nullify guessing advantage, as the attacker would learn too much already from the prior.

4.5.2 Posterior distribution

The attacker receives noisy occupancy counts $(y_l^t)_{l \in \mathcal{L}}$ of all regions at timepoint t. Depending on the attacker type, certain regions in the buildings are opened. If the opening does not reveal the location of u immediately, it modifies the priors as $p_{ul}^t = \frac{p_{ul}^t}{\sum_{\ell \in \mathcal{L}_{closed}} p_{u\ell}^t}$ for $l \in \mathcal{L}_{closed}$, where $\mathcal{L}_{closed}$ is the set of regions that remained closed.

For true outputs, the victim's location depends solely on the counts of the current timepoint. This is however different for privacy enhancing mechanisms, where sequential timepoints may leak information about each other's randomness. For example, if the attacker knows that the counts most likely do not change during a 1.5 hour span (e.g., if the region is a classroom and it is a lecture time), then the noise will be essentially applied to exactly the same counts, and multiple outputs help in undoing it. Hence, let us only estimate how much the attacker learns from the output of single timepoint. That is, while we still report results for all timepoints t, we assume a separate attacker for each reported timepoint.

Let m be the total number of objects, and $p := p_\ell^t$ be the probability of each object being in region l. The question is now how to efficiently compute Eq. 4.4. Since z is not a single count, but a vector of region occupancies, summing up all combinations $z_1, \ldots, z_n$ is infeasible. Hence, let us estimate how much the attacker learns from observing the occupancy

of one region without taking into account the others. For guessing from true counts, we only need to know z_l and m, so similarly to the issue with different timepoints, taking into account more z_j-s can only help in undoing the noise, e.g., if the attacker knows that some regions are occupied simultaneously.

While these two constraints are fine for guessing for true counts, they give us only a lower bound for general privacy mechanisms, showing how much an attacker can learn at least. Similarly to composition theorems of differential privacy, we can extend our results to several outputs as described in App. B.3, but it may give us too rough upper bounds. As a result, we are comparing different privacy mechanisms based on observing a single output. It may be that some mechanism scales better with the number of observed outputs than another, which remains out of scope of this chapter.

We are now ready to estimate the posterior probability. Using Eq. 4.4, we get

$$\mathbf{Pr}[x|\mathbf{Y} = y] = \frac{\sum_{z=1}^{m} \mathbf{Pr}[x|\mathbf{C} = z] \cdot f_Y(y|z)\mathbf{Pr}[z]}{\sum_{z=1}^{m} f_Y(y|z)\mathbf{Pr}[z]} \ ,$$

where y is the occupancy of *one particular region*, and the probabilities are instantiated as follows.

- $\mathbf{Pr}[x|\mathbf{C} = z] = \frac{z}{m}$.
- $\mathbf{Pr}[z] = \binom{m}{z} p^z (1-p)^z$.
- $f_Y(y|z)$ depends on the particular analyzed privacy mechanism.

The quantity $f_Y(y|z)$ is either given in advance (for a known noise distribution like Laplace), or approximated from (y, z) points of training data using *kernel density estimation* (KDE).

This is a quite standard density approximation technique, which can be viewed as assigning to each sample point a bell-shaped curve centered at that point, and then summing all curves up, scaling the result to get a probability distribution. We use Gaussian kernel of Python `scipy` library [58]. By default, the Gaussian KDE bandwidth parameter in `scipy` library is $n^{-1/(d+4)}$, where n the number of data points and d the number of dimensions. In our case, n comes from the training data (88 days), and since we are computing a separate kernel for each true count z (the approximated probability density is conditional), we have $d = 1$. We note that attacker's success may depend on the bandwidth parameter, and choosing one that approximates the noise distribution most precisely is out of scope of this work.

In general, we do not know which parameters we should take into account in training. E.g., even if we include all occupancies of all regions to predict occupancy of a single region, some meta-knowledge like day of the week or time of the day can actually affect the noise distribution. For DP, we actually know that the noisy output depends only on the randomness and the true output, but in general we do not know all dependencies in advance. To simplify the training process, in our experiments we compute the noise distribution for a single location at a single timepoint, as if the randomness was sampled independently.

The posterior confidence of the attacker may be erroneous due to improperly computed noise distribution, or improperly computed priors. The latter may happen even if we use a well-defined DP mechanism. Hence, we add an important condition to our privacy metric. We model a particular attacker who actually *makes a particular guess* about victim's location. We then check whether that guess has been correct or not, and nullify the estimated advantage if the guess was incorrect.

4.6 Results

The results of our experiments are presented as plots (a sample plot is given in Figure 4.2). For each of the T timepoints, we count the total number of people whom the attacker managed to localize correctly with certain *confidence*, defined as the posterior probability of being in the room where the user has actually been according to the presence table. On each plot, the x-axis denotes the timepoint, and the y-axis is the number of localized people. The colors, ranging from light blue to dark red, correspond to localization confidence p, where light blue is the lowest confidence class ($0.0 < p \leq 0.1$), and dark red is the largest confidence class ($0.9 < p \leq 1$). Notice that the plot shows for each time point the total amount of people localized in the building broken into different confidence classes. This means that, for instance, out of the 123 people located in the building at 11:40am of the particular day in Figure 4.2, 22 are localized with the lowest confidence $0.0 < p \leq 0.1$, 55 with confidence $0.1 < p \leq 0.25$, 26 with confidence $0.25 < p \leq 0.5$, and 20 with confidence $0.5 < p \leq 0.75$. There are no red and dark red areas for 11:40am, so there have been no people localized with confidence $p > 0.75$. The plot format will be the same for all plots in this chapter, so we will avoid repeating labels and legends on further plots to conserve space.

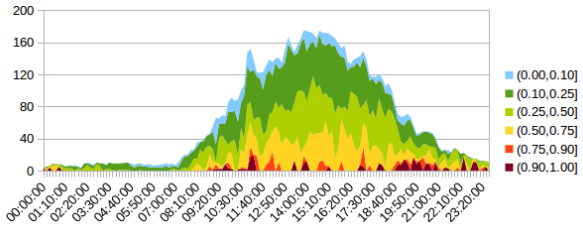


Figure 4.2: A sample plot: x-axis is the timepoint, y-axis the total number of people localized within each confidence class.

4.6.1 Exact vs Predicted noise distribution

First of all, we need to estimate how well predicting $f_Y(y|z)$ using kernel density estimation from training data works compared to true noise distribution. We do it on the example of Laplace noise, for which we already know the true distribution of noise $f_Y(y|z) = \frac{\epsilon}{2} \cdot e^{\epsilon \cdot |y-z|}$.

Prior

The plots of prior probabilities are given in Figure 4.3. The three columns correspond to the initial knowledge of the attacker, where he knows $\delta \in \{0\%, 50\%, 90\%\}$ of people locations. In this experiment, the regions have been opened randomly. We compute the posterior probability for each user on the condition that their location has not been revealed to the attacker directly, so formally for each potential victim we consider a separate attacker who knows δ of the *other* users. This is why we do not observe that δ of the graph is dark red.

Posterior from Exact Noise Distribution.

For posteriors we will only show those probabilities that have been *improved compared to the prior*, thus demonstrating the *advantage*. The results are given in Figure 4.4. The rows of the plot matrices correspond to $\epsilon \in \{0.1, 1.0, 5.0, \infty\}$, where ∞ is guessing from true outputs. We can see how confidence increases with ϵ. We see that $\epsilon \geq 5.0$ already gives us a plot very similar to guessing from true outputs, so it does not make sense to consider larger epsilons. For smaller epsilons, we indeed get smaller confidence, which converges to 0 as $\epsilon \to 0$. While the posterior *probability* always increases with δ, we see that the *advantage* may sometimes be larger for smaller δ, which means that the attacker guesses so poorly from prior that even a very noisy answer gives some benefits.

We compare obtained results with the worst-case upper bound estimate considered in Sec. 4.4.2,

which does not depend neither on δ nor the particular counts, and holds for any ϵ-DP mechanism. The results are given in Figure 4.5 for $\epsilon \in \{0.1, 1.0, 5.0\}$. We see that, for larger ϵ, the upper bound gets larger than the probabilities of guessing from true counts, so the upper bound is too rough for our type of attacker. These bounds nevertheless seem to be good for small ϵ.

Table 4.1 shows the times of computing the posterior probabilities. The first row corresponds to computation of the worst case bound of Sec. 4.4.2. The second row corresponds to precise computation of posterior (Eq. 4.4). In general, the precise computation of posterior is $O(n)$ times slower for n rooms since we are evaluating a sum over n terms. In our example, it is ca 9 times slower for $n = 64$ rooms. The preprocessing time is spent on the bookkeeping related to loading data from the database. We do not count the time spent on generating the occupancy tables.

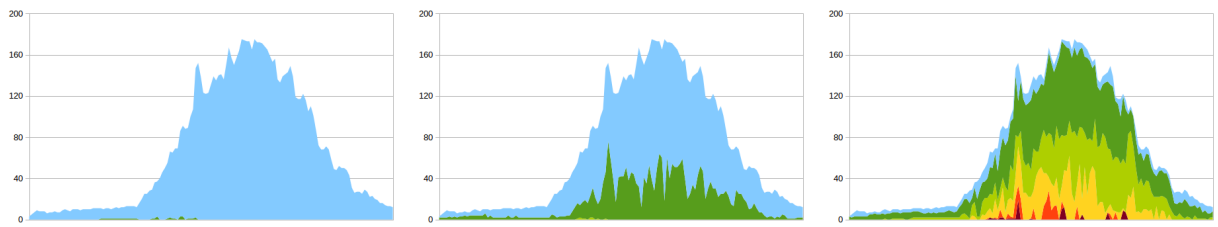


Figure 4.3: Prior guessing probabilities for $\delta = 0\%$ (left), $\delta = 50\%$ (middle), $\delta = 90\%$ (right)

Posterior from Predicted Noise Distribution.

Since the trained noise distribution does not depend on δ anyway, let us only consider $\delta = 0.9$.

First of all, we repeat the experiment with known DP noise distributions, filtering out the guesses that have actually been correct (top row of Figure 4.6). We then repeat a similar experiment with noise distribution learned from prior data, using Gaussian kernel with default parameters (bottom row of Figure 4.6). The plots are shown for $\epsilon \in \{0.1, 1.0, 5.0\}$ (left to right). We see that the guesses based on trained distribution perform similarly to the true

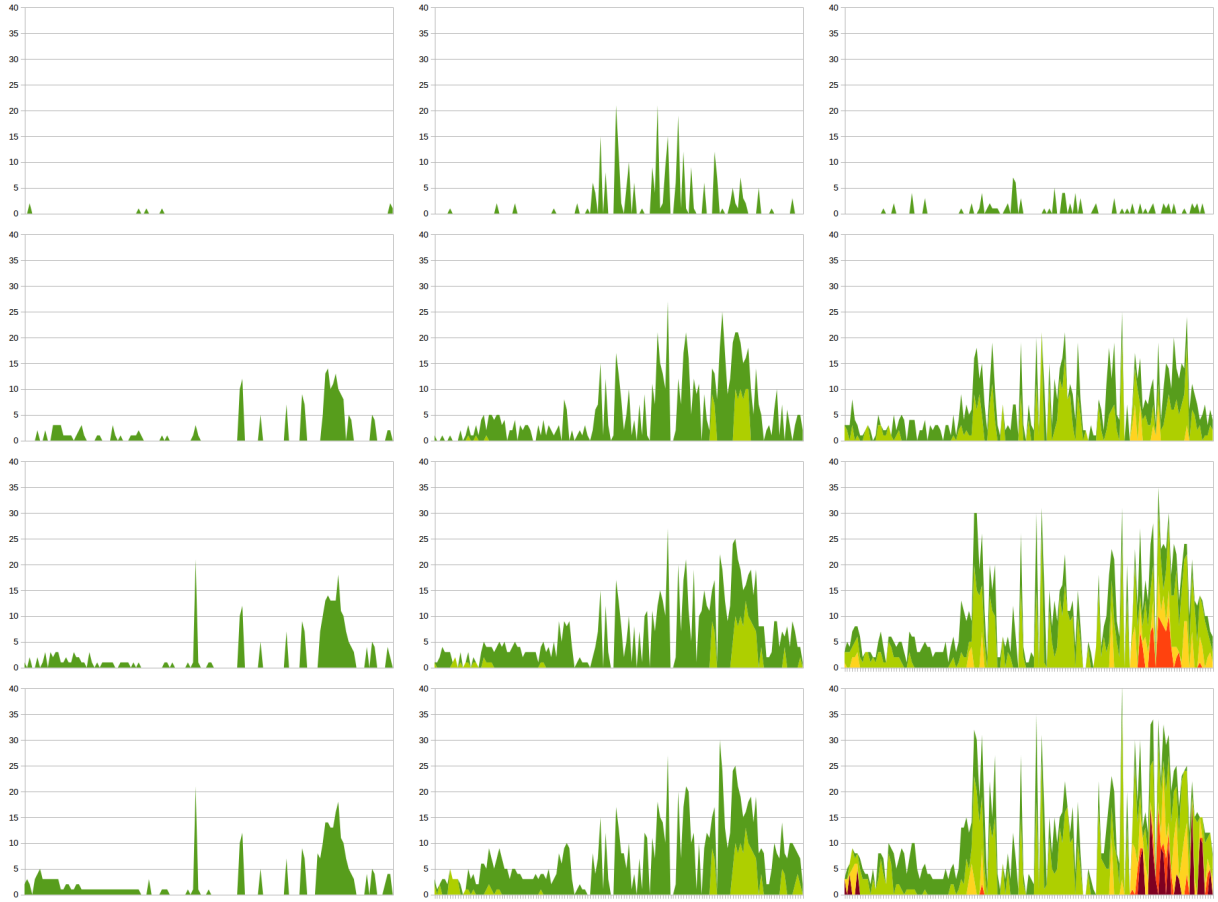


Figure 4.4: Posterior guesses from noisy Laplace counts for $\delta = 0\%$ (left), $\delta = 50\%$ (middle), $\delta = 90\%$ (right), $\epsilon \in \{0.1, 1.0, 5.0, \infty\}$ (top to bottom)

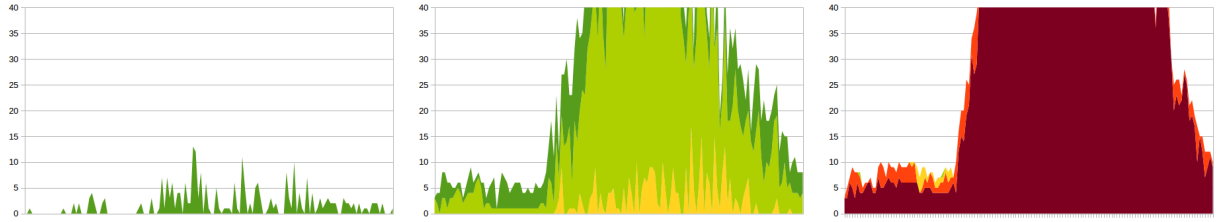


Figure 4.5: Posterior guesses for worst-case DP, $\delta = 90\%$, $\epsilon \in \{0.1, 1.0, 5.0\}$ (left to right)

distribution of Laplace noise.

The third row of Table 4.1 shows the times for the trained distribution experiment. We see that the testing time is a bit higher, which is due to computing $f_Y(y|z)$ from kernel. In addition, there is now also some time spent on one training to compute the kernel itself. We note that for DP experiments, we need a separate training for each ϵ.

Table 4.1: Running times (in seconds) of computing posterior probabilities for Laplace noise.

	Preprocess	Training	Posterior evaluation
worst-case DP bound	4	0	0.6
$\mathbf{Pr}[x\|y]$ for known $f_Y(y\|z)$	4	0	5.4
$\mathbf{Pr}[x\|y]$ for unknown $f_Y(y\|z)$	4	6.9	7.3

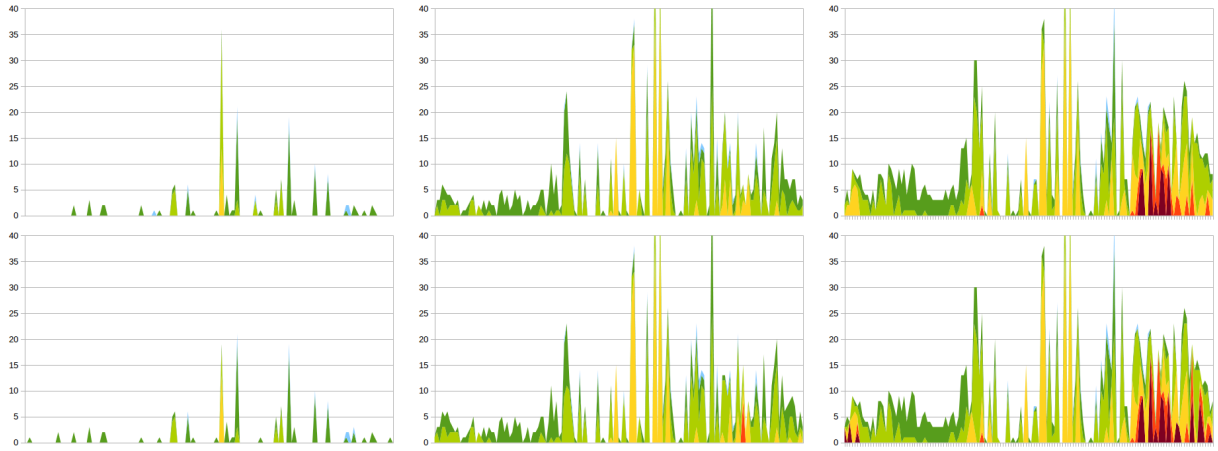


Figure 4.6: Successful guesses from noisy Laplace counts with $\delta = 90\%$ using true distribution (top) and scaled kernel density estimation (bottom), $\epsilon \in \{0.1, 1.0, 5.0\}$ (left to right)

4.6.2 Comparing different privacy techniques

We now compare the techniques in Sec. 4.2.2 w.r.t. the attacker's success in breaking user privacy for the same level of utility. The noise density functions of all methods is approximated from training data using Gaussian kernel with scaling 0.1. The parameter 0.1 has been chosen empirically as the one for which the attacker was more successful. We note that a different scaling parameter can be preferred for different mechanisms, and that the goodness scaling in turn depends on the size of training data sample: the more datapoints we have, the more we want to narrow the kernels to get a more precise estimate.

TTL (described in Sec. 4.2.2) changes the unique identifiers of the objects every k seconds for a fixed parameter k. However, counting does not depend on the identifiers anyway. The only way in which TTL can affect released counts is that the same user may be recorded

multiple times if his identity has been updated within the 10-minute span for which the count is computed. If the attacker knows the value k, it can just divide all counts by the expected number of repetitions to get the true counts back. We see that there are no provable security guarantees at all. However, in practice the number of repetitions turns out to less predictable, as the identities are not being updated "for all users at once", but depend on the time when the user has connected to the system. While there is no randomness, the non-determinism of user movement can be viewed as a random variable whose distribution is difficult to estimate theoretically.

Setting up Privacy Parameters

To fairly compare privacy loss across diverse PETs, their utility should be similar as there is an inherent privacy vs. utility tradeoff. We have developed a tool which, given a specific task, obtains the configuration parameters per PET (i.e., epsilon and TTL) that will satisfy a given utility requirement. The tool takes as input the WiFi connectivity data, the PET to apply, a function that computes utility, and a requirement for the utility. Then, it tries different values for the privacy parameter until it finds one that satisfies the utility requirement. In our set up, we define the task to be that of generating a heatmap (as described in Sec. 4.2.1) given the occupancy data (real data or data generated through a PET). Our utility metric is computed as a percentage difference between the heatmap color assigned to the occupancy value generated by the PET and the heatmap color assigned to the real occupancy value (100% utility means real data and data generated by the PET are assigned same color). Then, we average this utility across time and space to get the utility value for a given dataset.

We computed parameters for PeGaSus, Laplace, and TTL that give us the utility 75% and 90%. These parameters are summarized in Table 4.2. We compute these parameters over 5 runs for non-deterministic techniques (i.e., Laplace, PeGaSus). The utility in each run lies

within $75 \pm 0.2\%$ and $90 \pm 0.2\%$ for the given parameters. Since the privacy parameters for the same utility are very different for day and night time, we also extract privacy parameters for day and night times and perform different experiments. Since TTL tends to map empty rooms to empty in most cases, for TTL we get high utility at night when the most true counts are 0, which allows us to introduce more noise, i.e., refresh the user identities more frequently. To get 90% utility for PeGaSus for day time, ϵ turns out to be very high i.e. 15. For high values of utility, we see a very small increase (only 2%) in utility from $\epsilon = 2$ to $\epsilon = 15$ which is possible due to error introduced by grouping/smoothing of contiguous similar occupancy counts. Since the utility is averaged over time and space, we also show average variance in utility over time and space σ_u (which turns out to be similar across techniques).

Table 4.2: Parameters achieving the same utility for different privacy mechanisms.

	utility	Laplace		PeGaSus		TTL	
		σ_u	ϵ	σ_u	ϵ	σ_u	T
day	75%	23.7	0.1	24.1	0.04	20.4	$1sec$
	90%	14.5	0.66	16.9	15	14.9	$2min$
night	75%	26.6	0.18	26.6	0.00001	19.5	$1sec$
	90%	19.3	1.675	19.5	4	19.5	$1sec$

Results per Attacker Type

Instead of opening regions to the attacker randomly as in Sec. 4.6.1, we consider in the following certain types of realistic attackers in the context of the dataset. There are 64 regions in the building, each having typically granting access to different profiles of people (e.g., students, professors, staff). We will consider three types of attackers based on such information.

1. An external attacker who is not present in the building and thus, does not have access

to any region (i.e., 0 open regions).

2. A student (or group of student) who coordinate an attack and gain access to all regions where a student can enter (i.e., 37 open regions). This includes classrooms and public areas.

3. A building administrator who has access to the security camera system and thus has access to all the regions covered by cameras (i.e., 39 open regions). This set of spaces includes public areas as well as corridors near offices.

Figures 4.7-4.9 compare different privacy techniques in different settings for the previous attackers. The columns correspond to the four types of experiments (day/night, 75%/90%-utility), and the rows to different privacy mechanisms, including guessing from true counts (the last row). In each graph, the X-axis is time (7am-7pm for day and 7pm-7am for night in intervals of 10 minutes) and the Y-axis is the number of people correctly localized at each confidence level[2]. As the noise added by DP techniques will be different in different executions, each experiment has been repeated n=30 times, and for each posterior probability class we took the average number of people that has been guessed with that probability. Finally, we consider that in the case of these realistic adversaries an open region implies that the adversary knows exactly who is inside of it. Thus, in the following we consider such information to be prior and in the plots we focus on how the different PETS affect the guessability of those individuals in closed rooms. Let us now discuss the results for different attackers.

External attacker. In Figure 4.7, we see the results for an external attacker who has a weaker prior as he/she does not have access to real location information for any user at the time of the attack. Given the true counts, the external attacker can localize some users from, albeit with confidence $p \leq 50\%$. In particular, during the day time and early morning,

[2]Note that the scaling of Y-axis for day and night time are different, as the total number of people in the building is very different for them.

when the occupancy of the building reaches its peak, the external attacker can perform the most successful attack by correctly guessing the localization of 50 people with a confidence of $0.25 \leq p \leq 50\%$. Notice that during the night, the number of people in the building is small and the most successful attack occurs in the early morning with 10 individuals localized with low confidence. Using privacy mechanisms eliminates most of these localizations. Indeed, with parameters satisfying 75% utility, which increase the privacy protection, the localization possible when publishing real counts is completely eliminated. Notice that when a higher utility is required (90%), the mechanisms might publish data closer to the real occupancy and therefore there is some leakage. Notice that the results are averages from different runs which means that mechanisms based on differential privacy, which are not deterministic, might perform better or worse in specific situations. We explore this aspect further in App. B.4. From the plots, we cannot determine whether one mechanism is better than the other one, as the number of localizations is small for all of them.

Student attacker. A student attacker has access to 37 of the 64 regions. Hence, the attacker knows the real location of all the individuals in those regions at the time of the attack. For the remaining set of users in closed regions, in Figure 4.8, we see that a student can localize more people than an external attacker, indeed in the strongest attack the student attacker is able to localize 10 people during day time and 20 during the night time with a high confidence $0.75 \leq p \leq 0.9$. This happens at the end/beginning of the working day when the building is less occupied vs. the middle of the day when most of the classrooms are full. This attacker has some success even if privacy mechanisms are applied, specially for the published dataset at daytime to achieve 90% utility. Both mechanisms based on differential privacy offer slightly better practical privacy than TTL until the middle of the day in the plot corresponding to daytime. The Laplace mechanism gives the most privacy in this situation when compared to PeGaSus and TTL. Nevertheless, notice that in this situation the difference between mechanisms is small as the number of users that are localized either

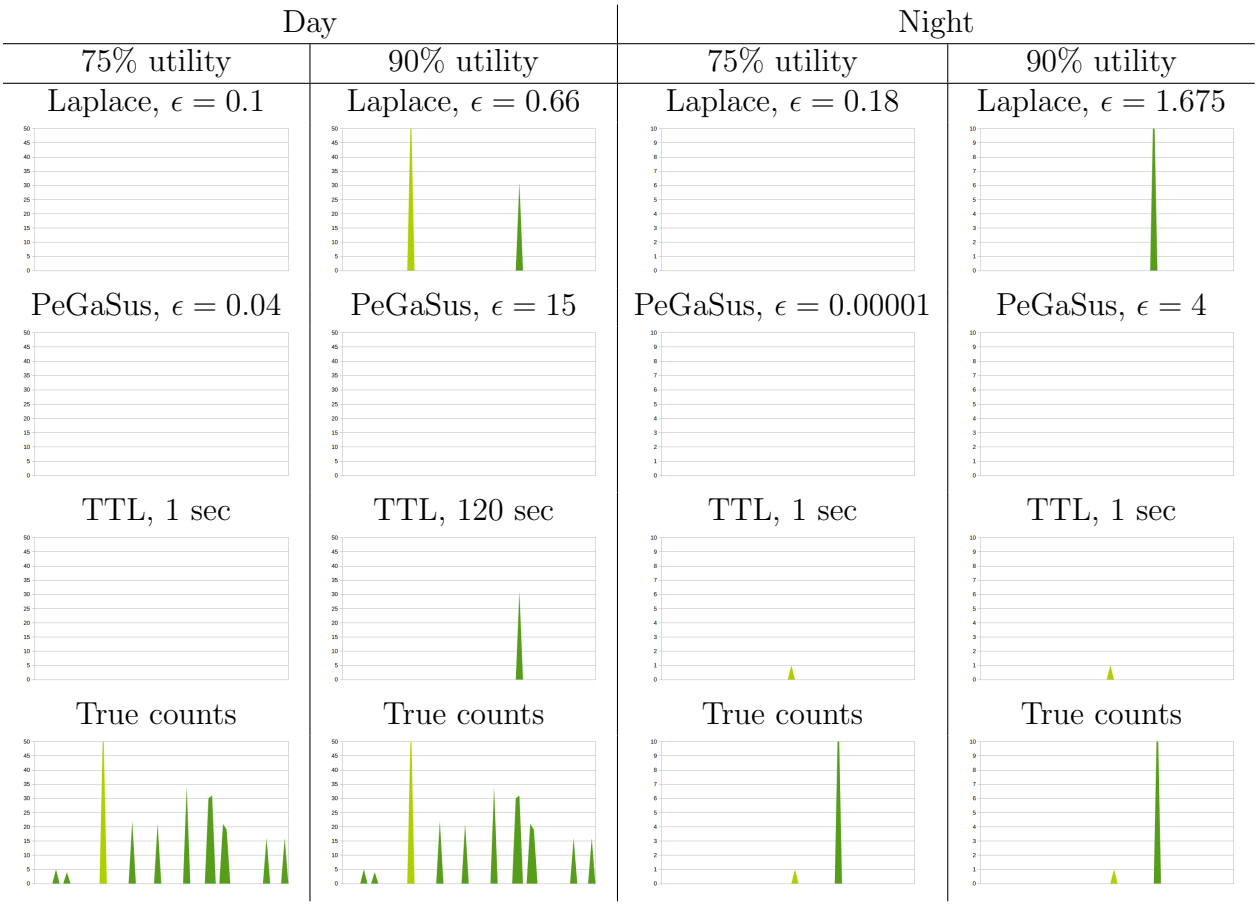


Figure 4.7: Comparison of different mechanisms for the external visitor attacker.

because of they are located in an open room or because the attacker's prior is large.

Administrator attacker. An administrator is given access to 39 of the 64 regions. This situation is similar to the previous but the set of open regions is different including those that contain offices. The results are also similar to those of the student attacker, although in general there is more privacy loss across techniques and for the true counts. This occurs because this adversary has access to real location of more people as the open regions cover a higher amount of the building's population. As in the case of student attacker, both the techniques based on differential privacy perform slightly better than the TTL technique. Similarly, the Laplace technique performs slightly better than PeGaSus. When comparing TTL and PeGaSus for the daytime and 90% utility we notice how both perform very similarly

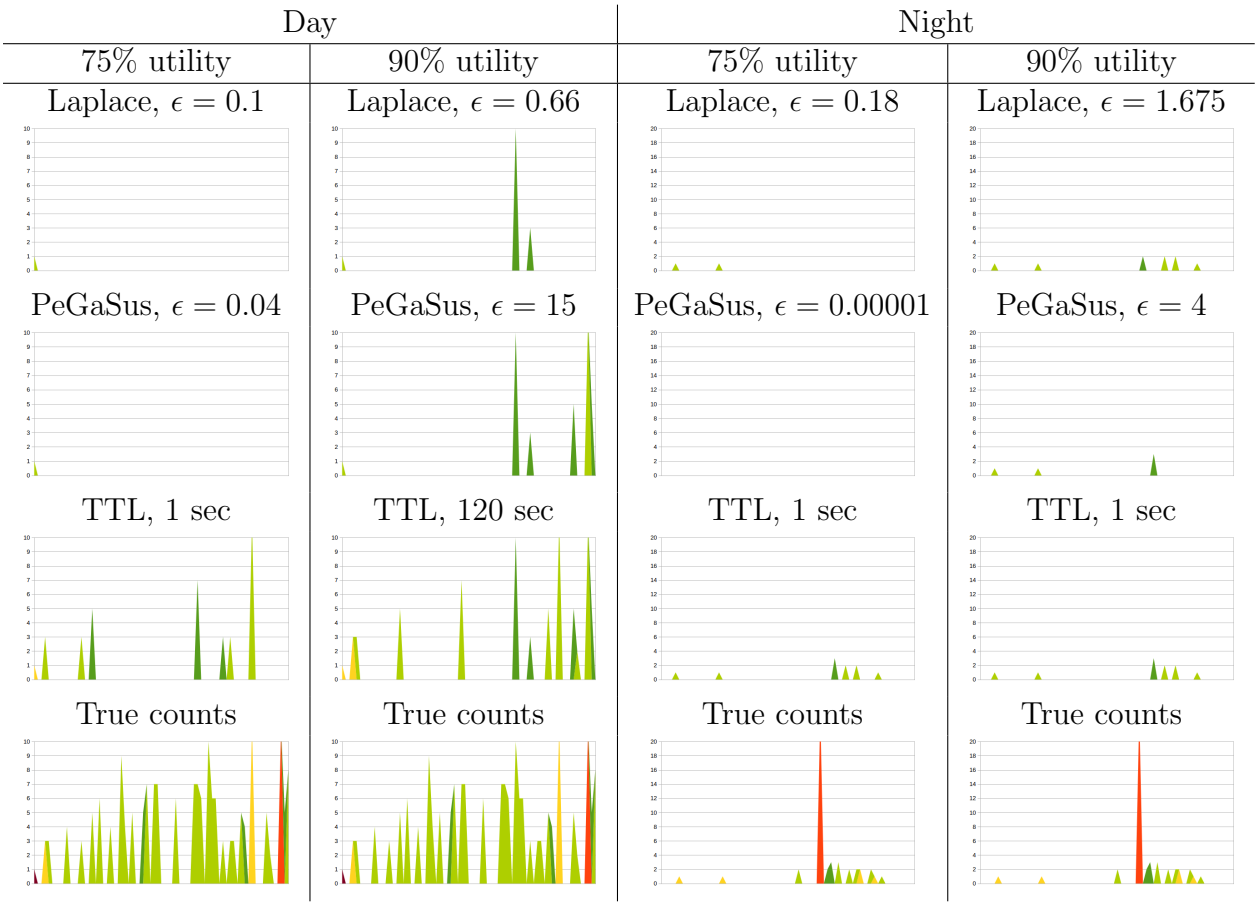


Figure 4.8: Comparison of different mechanisms for the student attacker.

in the afternoon when the building is less occupied. Also, when focusing on the nighttime, at 90% utility all the techniques perform very similarly.

Summary

Differential privacy in general consider very strong attackers that have access to almost unlimited information. In our set up with more realistic adversaries, we have seen that when the adversary is weaker (e.g., our external attacker). The practical privacy offered by Laplace and PeGaSus is almost the same than the one offered by TTL. Even when the practical privacy for Laplace and PeGaSus is similar, for higher utility values the formal privacy guarantee for PeGaSus is less than Laplace during the daytime and nighttime, whereas for

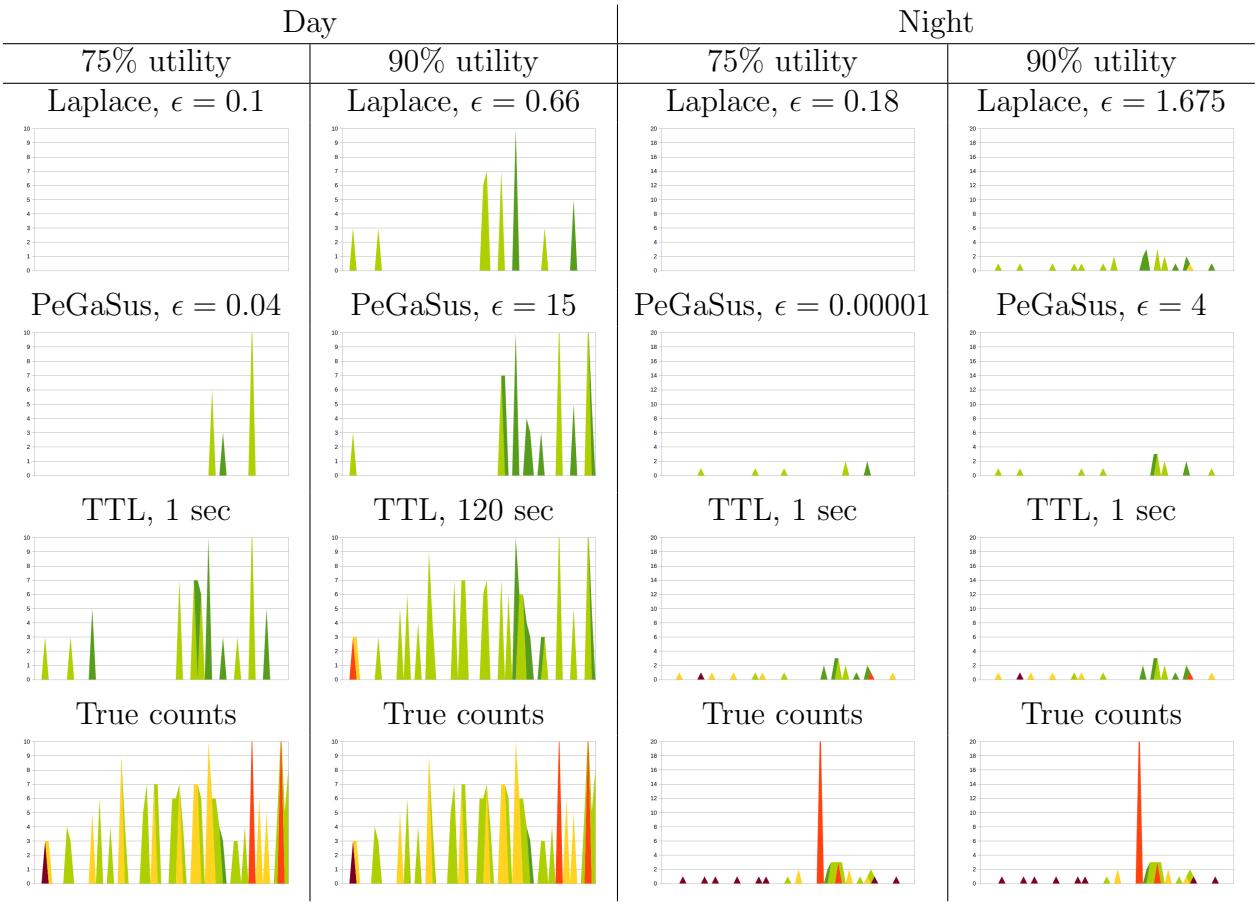


Figure 4.9: Comparison of different mechanisms for the administrator attacker.

lower utility values it is the opposite. When the adversaries become stronger (e.g., the student or administrator attacker), Laplace and PeGaSus offer more practical privacy than TTL, as it is expected. However, in some specific situations (e.g., in the afternoon when the building is less occupied) all the techniques behave similarly. Additionally, with stronger attackers the privacy loss due to the prior and adversarial knowledge at the time of the attack is high already. This means that effectively, in such situations the privacy of most of the individuals would be already compromised. Therefore, the difference between the differential privacy based techniques and TTL in terms of number of people being localized is small.

We would like to highlight that even when in terms of practical privacy the techniques behave similarly, TTL lacks of formal privacy guarantees which means that stronger attackers

using more sophisticated attack methods could potentially result in higher privacy loss. Additionally, the results for Laplace and PeGaSus, which are the average over 30 counts, could potentially be worse depending on the noisy count generated in a single run at publishing time. In App. B.4, we present a sample analysis of Laplace and PeGaSus for some particular noisy counts. However, comparing the *distributions* of attacker's success for different privacy mechanisms remains out of scope of this work.

4.7 Conclusion

We have performed a case privacy study of privacy mechanisms to publish occupancy data in the context of a smart building. The goal of the study was to explore the practical privacy implications for individuals privacy of the release of occupancy counts. We proposed a methodology to compare across privacy techniques with different privacy parameters and guarantees. Additionally, we presented a posterior computation method to perform an attack on data output by a privacy technique to infer, with certain confidence level, where an individual is located at a point of time. In our study, we performed experiments based on a real dataset containing connectivity events captured at a smart building for a period of three months. The results show that the practical privacy for individuals, under our attack model, offered by the Laplace, PeGaSus, and TTL mechanisms is similar with the former two performing slightly better than the latter. When comparing the two differential privacy mechanisms in terms of their formal guarantees, PeGaSus provides better formal guarantees for lower utility values whereas Laplace provides better guarantees for higher utility values. This highlights that while generic methods of differential privacy analysis give some bounds to strong types of attacker, when facing weaker and more realistic attackers, techniques with no formal guarantees can still offer similar practical privacy.

Chapter 5

MIDE: Accuracy Aware Minimally Invasive Data Exploration For Decision Support

5.1 Introduction

Decision-support (DS) applications [22, 53, 4] allow timely and informed decision-making and planning based on analyzing data, but such applications could face severe privacy challenges if the data analyzed contains personally identifiable information about individuals. For instance, a building management system may maintain the occupancy statistics (like in Figure 5.1) to detect violation of fire code, adherence to the CDC (Center For Disease Control) guideline in the context of COVID-19, or for better space utilization. If the location of interest has an aggregated occupancy that is higher than a threshold, an alarm is raised, but this aggregated statistics can leak sensitive information about users [18]. For example, prior work [32] has shown, with enough background knowledge, occupancy data can lead to

inferences about the location of individuals, which, in turn, can leak sensitive information (*e.g.*, in an office building staff consistently leaving work early, smoking habits of individuals). As another example, consider assisted living situations where one of the primary challenges is fall prevention [57] of the elderly and the goal is to balance safety with privacy. We could monitor someone invasively using a camera, but such invasiveness is not necessary if the person is not a high fall risk. To make a decision about using invasive means of monitoring, wearables can be used to collect aggregated statistics *e.g.*, number of sudden accelerations in a week. Sudden accelerations exceeding a threshold could be interpreted to mean high fall risk and we can make a decision to monitor such an individual more invasively. The commonality in such DS applications is that the aggregated statistics are collected and compared to a preset threshold that classifies objects as either satisfying the predicate (*i.e.*, true), or as not satisfying the predicate (*i.e.*, false). Simply releasing the aggregated statistics, however, can lead to privacy violation of individuals , *i.e.*, reconstruction attack as shown in [14, 12, 10, 32].

Much of the prior work on privacy has been motivated by the need for data sharing while ensuring the privacy of sensitive data. Examples include privacy-preserving sharing of demographic data (*e.g.*, US Census), medical data to support research (*e.g.*, cancer registries), or collecting click-stream data for vulnerability analysis (*e.g.*, from browsers). Over the past decade, differential privacy [13] has emerged as one of the most popular privacy notions. It provides a formal mathematical guarantee that individual records are hidden even with the release of aggregate statistics and it is possible to bound the information leakage by a total privacy budget across multiple data releases. This has led to a wide range of adoption of differential privacy in a number of products at the US Census Bureau [15], Google [39], and Uber [23].

While differential privacy is suited for privacy-preserving sharing, its usefulness in the context of decision support (DS) applications is limited. DS tasks require guarantees on the

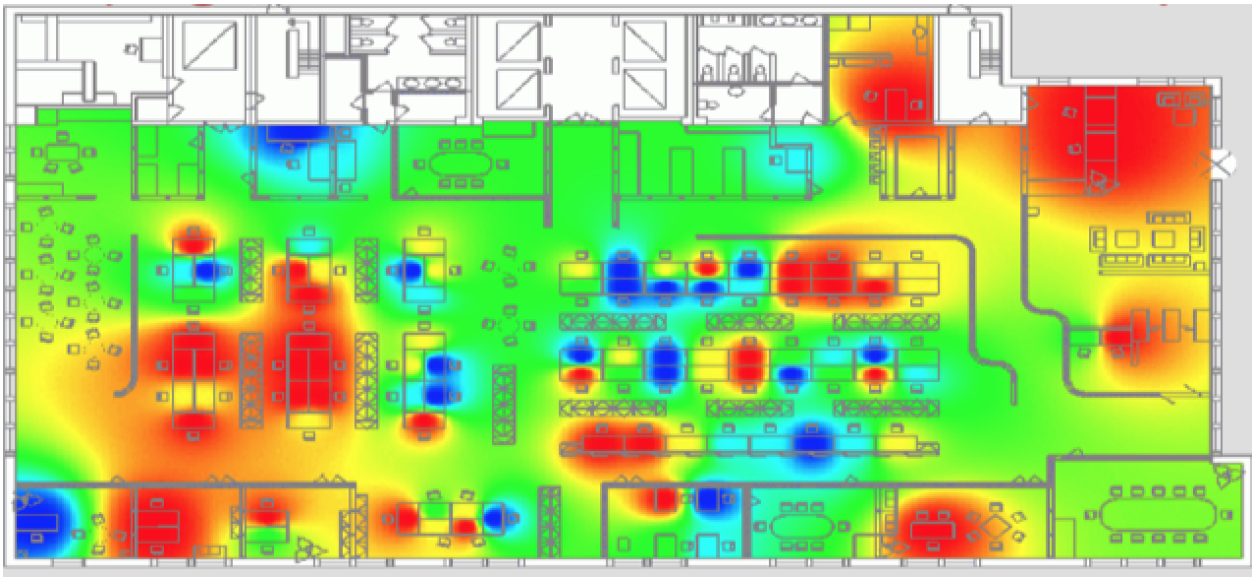


Figure 5.1: Occupancy Heatmap of a Building in UCI.

output quality, especially, for false negatives that may result due to the addition of noise to aggregated statistics. Such false negatives may result in events of interest/anomalies not being detected. For instance, in the elderly fall prevention example, a false negative may cause increased fall risk (from aggregated statistics of the number of accelerations) to go unnoticed preventing timely escalation and intervention. False positives are also not desirable, e.g., in the elderly fall prevention scenario, it may result in unnecessary escalation by using more invasive camera technology and wasted resources of video processing. Likewise, in the example of a fire code violation in a building, false positives on highly occupied spaces in the building may result in a heightened investigation of the region. While one would desire effective bounds on both false negatives and positives, in DS applications, increased false negatives are far more debilitating (compared to false positives) since they effectively defeat the very purpose of decision support. Thus, in DS applications, we desire to have bounded guarantees on false negatives without significantly increasing the number of false positives. [1]

[1]If we ignored false positives and only considered false negatives, a trivial algorithm would be to simply ignore the query condition and return all the objects. This will meet the bounded requirement of false negatives and will have zero false negatives. But that also defeats the purpose of decision support applications.

Traditionally, DP-based approaches focus on providing formal privacy guarantees (in the form of a privacy parameter) while trying to maximize utility. These techniques do not offer guarantees on the quality of data outputted. Recent studies have addressed this challenge by designing accuracy aware DP techniques where the goal is to provide provable bounds on utility, e.g., [16, 38, 47]. Such approaches, however, are unsuitable for DS for several reasons: first, such approaches do not differentiate between false positives and false negatives, and offer a symmetric guarantee on both which makes them suboptimal in the DS context. Furthermore, the guarantee such approaches offer have a *region of uncertainty* around the threshold such that bounded guarantees (on either false positives or negatives) do not apply to data that falls in that region. This makes the techniques unsuitable for DS applications that require a tight guarantee on (at least) the false negatives.

In this work, we explore a utility-aware technique that provides (probabilistically) bounded guarantee on utility (in terms of asymmetric bounds on false negatives that are guaranteed to remain lower than a limited number) while minimizing privacy loss using differential privacy. The key intuition is to *modify* the DS query appropriately (before adding noise) so as to control the trade-off between false positives and false negatives and supports guaranteed utility in terms of false negatives. In particular, we generalize the query condition (e.g., replacing a query condition $X > \tau$ by $X > \tau'$, where $\tau' < \tau$) to admit a larger number of false positives but reduce the probability of data being wrongly classified as a false negative.

While a scheme that offers a bounded guarantee on false negatives can be designed by weakening the query condition, a proper design leads to subtle complexities. As will become evident, the (probabilistic) guarantee on false negatives, the weakening of the query condition, and the amount of privacy loss (ϵ in differential privacy terms) are interrelated. In particular, the weaker we make the query condition (i.e., over-generalization), the lower the privacy loss (smaller ϵ), while maintaining a bound on the false negatives. However, the weaker the query condition, the more the number of false positives. Ideally, we would like

to weaken the condition as much as possible, as long as it does not cause false positives to arbitrarily increase. This depends upon the data distribution. Imagine, for instance, that there is almost no data (or very little data) around the threshold specified in the query — such would be the case, for instance, for outlier queries. In such a case, weakening the query condition significantly would be desirable since that would allow us to reduce privacy loss without increasing false positives, while still ensuring the required bounds on false negatives.

In this chapter, we explore the design space of solutions alluded to in the discussion above. We first explore a single-step approach that minimally weakens/generalizes the query condition to achieve the bounded guarantee. We then explore a multi-step approach, wherein we aggressively make a decision to significantly weaken the query condition, and then, based on the outcome (*i.e.*, possibility of too many false positives) progressively refine the condition at the cost of loss of privacy (*i.e.*, larger ϵ), while maintaining false negative bounds. Like prior multi-step approaches of Apex [16], our multi-step approach also offers Ex-Post Differential Privacy [37] where the final privacy budget spent is determined after the completion of algorithm. Finally, we explore a data dependent version[2] of the multi-step algorithm that exploits the knowledge of data distribution learnt in previous steps to minimize the privacy loss.

In our algorithms, different objects/entities can be processed (*i.e.*, tested for threshold satisfaction) at different levels of privacy (ϵ). In the initial steps, the objects are processed at smaller ϵ (*i.e.*, higher privacy), and as the algorithm proceeds, some of the objects may be processed more invasively at higher values of ϵ with the goal of reduce the overall privacy loss. We, thus, refer to our approach as *Minimally Invasive Data Exploration (MIDE).*

The idea of different entities having different privacy levels has been studied in several pieces of prior work e.g. Personalized Differential Privacy [24], One-sided Privacy [29]. However,

[2]Data dependent algorithms have been studied in the context of differential privacy setting where privacy is fixed and we need to optimize utility [33, 61]

these works do not explore or provide a metric for overall privacy loss.

In summary, our contributions in this work are as follows:

- We introduce and formally define the problem of accuracy aware privacy-preserving decision support that has wide applicability in privacy preserving applications.
- We introduce Predicate-wise Differential Privacy (referred to as PWDP) which is suited for a data dependent approach to accuracy aware privacy-preserving analysis. We formally define the associated privacy metric for PWDP.
- We develop multiple efficient algorithms for the problem of accuracy aware privacy preserving decision support, including a multi-step algorithm and its data dependent variant.
- We show the applicability of our approach in a detailed study of several real-world scenarios.

The organization of this chapter is as follows: Section 5.2 defines the decision support queries, accuracy requirements of such queries, and our problem statement. This section also provides a new privacy definition of Predicate-wise Differential Privacy (PWDP) and defines a new privacy metric to measure the privacy loss. We use this to minimize privacy loss for our accuracy aware differentially private decision support algorithms in Section 5.3. Section 5.4 provides an algorithm to compute the new privacy loss metric. In section 5.5, we evaluate our algorithms using multiple real datasets. This chapter contains several theorems and lemmas, the proofs of which can be found in the Appendix.

5.2 Privacy in Decision Support

Decision support applications such as violation detection of the fire code based on the occupancy statistics or fall prevention based on weekly movement statistics, can be supported by a class of *aggregate threshold queries*. Such a query checks whether the aggregated values computed on a subset of tuples pass the thresholds or not.

Formally, an aggregate threshold query, denoted by $Q^{\Lambda}_{\mathsf{g}(.)>C}$, consists of (i) an aggregate function $\mathsf{g}(.)$; (ii) a set of predicates $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_k\}$; and (iii) a set of corresponding thresholds $C = \{c_1, c_2, ..., c_k\}$. Each predicate λ_i takes in a tuple and outputs *True* or *False* based on the value of the tuple. Let D_{λ_i} be the set of tuples in D that evaluate λ_i to be *True*. This query returns all the predicates that have an aggregate $g(D_{\lambda_i})$ greater than their respective threshold c_i, i.e.,

$$Q^{\Lambda}_{\mathsf{g}(.)>C}(D) = \{\lambda_i \in \Lambda \mid \mathsf{g}(D_{\lambda_i}) > c_i\} \tag{5.1}$$

For example, consider a location dataset inside a building with schema $Location\_Data(person, location, timestamp)$, a decision support application would like to learn which locations have more people than their maximum capacity. In this example, the predicate is conditioned on the location of a tuple, the aggregate is the number of people for a given location, and the threshold is the maximum capacity of that location. Another way to look at the problem is that the whole database could be viewed as points in a multi-dimensional space, and each predicate defines a subspace or a region. Given a set of such non-overlapping regions, the goal is to find the regions that contain points more than a certain threshold.

Answering such an aggregate threshold query with differential privacy guarantees has been considered in prior work [43, 38, 16], but these solutions may fail the accuracy requirements of a decision support application or demand an unnecessarily large privacy budget. Next, we

will describe and formalize the accuracy requirement and privacy requirement for decision support queries.

Accuracy Requirement. Two types of errors can be made by a randomized mechanism that answers a decision support query defined in Eqn. (6.1): (*i*) *false positives*, predicates that have smaller aggregate values than the thresholds but appear in the output; (*ii*) *false negatives*, predicates that have bigger aggregate values than the thresholds but are not outputted. While both false negatives and positives impact the effectiveness of the decision support application, preventing false negatives is far more crucial than false positives. A false negative may prevent timely intervention (*e.g.*, in the context of fall detection, or room code violation) which might be the very purpose of the decision support application. False positives, on the other hand, may result in false alarms that might have negative consequences in terms of wasted resources and/or violation of privacy (*e.g.*, as in more invasive monitoring in the fall detection example mentioned earlier). While one would like to minimize both, bounding false negative is far more crucial in decision support compare to false positives.

We formalize this accuracy requirement as follows.

Definition 5.1 (Accuracy Requirement (β-False Negative Rate))**.** *We say a mechanism $M : \mathcal{D} \to O$ satisfies β-false negative rate for an aggregate threshold query $Q^{\Lambda}_{\mathbf{g}(.)>C}$ if for any database $D \in \mathcal{D}$, we have*

$$\forall \lambda_i \in \Lambda, P[\lambda_i \notin M(D) | \lambda_i \in Q^{\Lambda}_{\mathbf{g}(.)>C}(D)] \leq \beta \tag{5.2}$$

Prior DP mechanisms such as the Laplace mechanism (Theorem 2.4) add noise from zero-mean distribution to the aggregate and compare it with the threshold, which place equal weights on false positives and false negatives. This approach can fail to bound both errors together by setting the privacy budget too small (large noise); or have guarantees on both false positives and false negatives, but with a high privacy cost. This symmetrical guarantee

will be illustrated in Section 5.3.1. To bound the false negative rate without incurring additional privacy cost, we design a class of mechanisms that generalizes the thresholds in the query. For example, for an aggregate threshold query where we are checking $X > c$ for an aggregate X, we generalize the query threshold to $X > c - \alpha$. This type of generalization allows us to achieve trade-off between false negatives and false positives that helps us achieve β-false negative rate with a minimal privacy cost. This generalization parameter α and the accuracy parameter β are translated to privacy cost ϵ. We will present these algorithms in Section 5.3.

Privacy Requirement. The privacy budget (ϵ) of a DP mechanism depends on the accuracy specification (e.g. β in Def. 5.1). Furthermore, if the DP mechanism is data-dependent, then the minimum privacy budget to achieve the accuracy requirement also varies among the data and depends on the output. This privacy loss is known as *ex-post DP* [37]. If running the DP mechanism on the disjoint part of the data (based on the predicates) in parallel, each part of the data may end up with different ex-post privacy loss. For example, to achieve the same β-false negative rate, a predicate with an aggregate value that is far from the threshold can tolerate a large generalization parameter α and result in a small privacy loss; while another predicate that is close to the threshold requires a big privacy budget. To capture this predicate-wise privacy loss for DP applications, we propose a new framework Predicate-wise Differential Privacy to generalize DP and ex-post DP. This framework allows the decision support application to attain the required level of utility while using higher privacy levels for some predicates and lower privacy levels for other predicates.

5.2.1 Predicate-wise Differential Privacy

Consider a set of mutually exclusive predicates $\{\lambda_1, \lambda_2, ..., \lambda_k\}$ that they can partition a dataset D into disjoint parts $\{D_{\lambda_1}, D_{\lambda_2}, ..., D_{\lambda_k}\}$. We define the new privacy as follows. In

this new framework, there is a privacy parameter ϵ_i associated with each predicate λ_i.

Definition 5.2 (Predicate-wise Differential Privacy (PWDP)). *Given* $\Theta = \{(\lambda_1, \epsilon_1), (\lambda_2, \epsilon_2), ..., (\lambda_k, \epsilon_k)\}$, *a set of mutually exclusive predicates that partition the full domain of the database and their corresponding privacy budgets, we say a randomized mechanism* M *satisfies* Θ*-Predicate-wise DP if for all* i*, for any neighboring databases* D *and* D' *differing in a record that satisfies* λ_i*, denoted by* $D \sim_i D'$ *i.e.,* $\mid (D_{\lambda_i} \setminus D'_{\lambda_i}) \cap (D'_{\lambda_i} \setminus D_{\lambda_i}) \mid = 1$ *and* $D_{\lambda_j} = D'_{\lambda_j}$ *for all* $j \neq i$*, the following condition holds:*

$$Pr[\mathcal{M}(D) \in O] \leq e^{\epsilon_i} \times Pr[\mathcal{M}(D') \in O] \tag{5.3}$$

In this new definition, the neighboring databases still differ by a single record (adding/removing a record), but the output distribution ratio depends on the value of the record. For example, for a location dataset inside a building with schema $Location\_Data(person, location)$, if it consists of only two predicates $\lambda_1 = (location = room_1)$ and $\lambda_2 = (location = room_2)$. Adding or removing a tuple $(person_1, room_1)$ will only affect the aggregate for only one of the above predicates (i.e. λ_1) as predicates are mutually exclusive. If this record takes a value t that satisfies one of the predicates λ_i and hence fails other predicates, then output distribution ratio is bounded by e^{ϵ_i}. A simple approach to achieve a predicate-wise DP is to run an ϵ_i-DP mechanism on a data partition D_{λ_i}.

Theorem 5.1. *Given* $\Theta = \{(\lambda_1, \epsilon_1), (\lambda_2, \epsilon_2), ..., (\lambda_k, \epsilon_k)\}$, *a set of mutually exclusive predicates and their corresponding privacy budgets, running* ϵ_i*-DP mechanism* M_i *over* D_{λ_i} *in parallel for* $i = 1, \ldots, k$*, achieves* Θ*-predicate-wise DP.*

It is also easy to see that a Θ-predicate-wise DP mechanism satisfies ϵ-DP, where $\epsilon = \max_{\epsilon_i \in \Theta} \epsilon_i$ by parallel composition of DP.

Predicate-wise DP also has the following composition properties. If two mechanisms consider

different sets of mutually exclusive predicates, then the composed guarantee will create a new set of mutually exclusive predicates to partition the dataset further. If a new partition has participated in only one mechanism, it takes the privacy budget of that mechanism, and if it has participated in both mechanisms, it takes the sum of the two privacy budgets.

Theorem 5.2. *Let M_1 and M_2 be predicate-wise DP mechanisms with $\Theta_1 = \{(\lambda_1, \epsilon_1), ..., (\lambda_{k_1}, \epsilon_{k_1})\}$, and $\Theta_2 = \{(\lambda'_1, \epsilon'_1), ..., (\lambda'_{k_2}, \epsilon'_{k_2})\}$, respectively. Let $M = f(M_1(D), M_2(D))$, then M is Θ-predicate-wise DP with the following predicates and their respective privacy budgets:*

$$\Theta = \{(\lambda_i \wedge \lambda'_j, \epsilon_i + \epsilon'_j) \mid \forall (\lambda_i, \epsilon_i) \in \Theta_1, (\lambda'_j, \epsilon'_j) \in \Theta_2, \lambda_i \wedge \lambda'_j \neq \emptyset\} \tag{5.4}$$

where $\lambda_i \wedge \lambda'_j \neq \emptyset$ denotes that the two predicates overlap. We exclude the conjunctions of non-overlapping predicate pairs. The resulted predicate set is mutually exclusive and partitions the full domain.

Last, we provide the ex-post version of predicate-wise DP, that generalizes the ex-post DP [37]. We will use it for our data dependent algorithms.

Definition 5.3 (Ex-Post Predicate-wise DP)**.** *Let $\mathcal{E} : \mathcal{O} \to \mathbb{R}^{|\Theta|}$ be a function on the output space of a Θ-predicate-wise DP mechanism $M : \mathcal{D} \to \mathcal{O}$. We say M satisfies $\mathcal{E}(o)$-Ex-post predicate-wise DP if for all $o \in \mathcal{O}$, and any neighboring database D and D' differing in a record that satisfy λ_i,*

$$\max_{D,D':D \sim_i D'} \ln \frac{P[M(D) = o]}{P[M(D') = o]} \leq \mathcal{E}_i(o), \tag{5.5}$$

where $\mathcal{E}_i(o)$ denotes the ith entry of $\mathcal{E}(o)$, the ex-post privacy cost for predicate λ_i.

Theorem 5.3. *A PWDP mechanism $\mathcal{M}$ with $\Theta = \{(\lambda_1, \epsilon_1), ..., (\lambda_k, \epsilon_k)\}$ satisfies ϵ-DP with $\epsilon = \max_i \epsilon_i$. A mechanism $\mathcal{M}$ with an ex-post PWDP loss $\mathcal{E}(o)$ has an $\epsilon(o)$-ex-post DP with*

$\epsilon(o) = \max_i \mathcal{E}_i(o)$.

PWDP can be used to track privacy loss in a more fine-grained manner (even without knowing the exact mechanisms) and result in a lower privacy loss even in terms of DP loss . Consider a database that only consists of two predicates λ_1, λ_2 to partition the domain. Consider two mechanisms M_1 and M_2, where the PWDP cost for M_1 is $\epsilon_{M_1,\lambda_1} = 0.1$, $\epsilon_{M_1,\lambda_2} = 0.5$ and the cost for M_2 is $\epsilon_{M_2,\lambda_1} = 0.5$, $\epsilon_{M_2,\lambda_2} = 0.1$. Keeping track of the fine grained epsilon loss per predicate using PWDP results in ex-post DP loss of 0.6. However, if we used DP, M_1 has a privacy loss of 0.5, and M_2 has a privacy loss of 0.5, and hence, the overall ϵ DP loss would be 1 by sequential/parallel composition. Hence, a fine-graiend tracking of privacy loss allows a tigheter privacy analysis, and more queries to be answered with the same DP loss.

PWDP and its ex-post privacy can also be interpreted as providing bounds on adversarial posterior odds ratio just like DP. After observing an output o of a PWDP mechanism M, the adversary can not successfully distinguish whether a tuple x is in D and takes a value t that satisfies λ_i, denoted by t_{λ_i}or the tuple x is not in D. Given adversary's prior odds ratio *i.e.*, $P[x \in D \wedge x = t_{\lambda_i}]/P[x \notin D]$, the bounds on adversary's posterior odds ratio *i.e.*, $P[x \in D \wedge x = t_{\lambda_i}|o]/P[x \notin D|o]$ is as follows:

$$|\ln(\frac{P[x \in D \wedge x = t_{\lambda_i}|o]}{P[x \notin D|o]} / \frac{P[x \in D \wedge x = t_{\lambda_i}]}{P[x \notin D]})| \leq \epsilon_i \tag{5.6}$$

Similarly, the ratio is bounded by $\mathcal{E}_i(o)$ for ex-post privacy.

5.2.2 Min-Entropy based Privacy Metric

Traditionally, DP mechanisms quantify privacy loss using ϵ. However, in predicate-wise DP, entities have different ϵ values. Comparing scenarios of different sets of epsilon values is

non-trivial. For example, consider $(\epsilon_1 = 0.1, \epsilon_2 = 0.5, \epsilon_3 = 1)$ v.s. $(\epsilon_1 = 0.2, \epsilon_2 = 0.4, \epsilon_3 = 1)$ for three predicates, it is not obvious which scenario has a lower overall privacy loss as both have the same maximum epsilon value (1.0) and the same averaged epsilon value (0.53).

This section introduces our privacy metric for predicate-wise DP using *entropy*. In information theory, entropy is a well known metric for measuring uncertainty of a random variable. Given a discrete random variable X with possible outcomes of $x_1, ..., x_k$, with occurrence probabilities of $P(x_1), ..., P(x_k)$, the entropy of X is defined as: $-\sum_{i=1}^{k} P(x_i) \log P(x_i)$. In the context of predicate-wise DP, the adversary is guessing which predicate from the given set $\{\lambda_1, .., \lambda_k\}$ a record $x \in D$ can satisfy based on the output of a predicate-wise DP mechanism o. We use $\hat{p}_i$ to denote the posterior belief that x takes t_{λ_i}, a value satisfies λ_i. This posterior is proportional to $\bar{p}_i = \sum_{t_{\lambda_i}} (\frac{P[x \in D \wedge x = t_{\lambda_i} | o]}{P[x \in D | o]})$ and hence $\hat{p}_i = \bar{p}_i / \sum_i \bar{p}_i$. Then, the entropy over $\{\hat{p}_1, \ldots, \hat{p}_k\}$ can measure how uncertain the adversary' belief about the value of x.

There is no direct information for the posterior beliefs, but based on the predicate-wise DP guarantee (Eqn. (5.6)), we can derive a lower and upper bound for each posterior belief $\hat{p}_i$.

Lemma 5.1. *Given a Θ-Predicate-wise DP mechanism M with output o, where $\Theta = \{(\lambda_1, \epsilon_1), (\lambda_2, \epsilon_2), ..., (\lambda_k, \epsilon_k)\}$, each adversarial posterior guess $\hat{p}_i \propto \sum_{t_{\lambda_i}} \frac{P[x \in D \wedge x = t_{\lambda_i} | o]}{P[x \in D | o]}$ is bounded:*

$$\frac{e^{-\epsilon_i}}{\sum_i e^{\epsilon_i}} \leq \hat{p}_i \leq \frac{e^{\epsilon_i}}{\sum_i e^{-\epsilon_i}}, \tag{5.7}$$

when priors $p_i \propto \sum_{t_{\lambda_i}} \frac{P[x \in D \wedge x = t_\lambda]}{P[x \in D]}$ are the same for $i \in [1, k]$.

This lemma assumes that the priors are the same for all predicates, which is possible when the adversary does not know the person. We also present the extended lemma for general priors in the appendix. Under these bounds, the largest entropy can always be attained when setting $\hat{p}_i$ the same for all the predicates. Hence, we consider the least uncertainty (min-entropy) as the privacy metric for predicate-wise DP.

Definition 5.4. *[Min-Entropy of PWDP] The privacy metric (Min-Entropy) of a Θ-Predicate-wise DP with $\Theta = \{(\lambda_1, \epsilon_1), ..., (\lambda_k, \epsilon_k)\}$ is defined as follows:*

$$\gamma(\Theta) = \min \sum_{i=1}^{k} -\hat{p}_i \log \hat{p}_i \tag{5.8}$$
$$s.t. \frac{e^{-\epsilon_i}}{\sum_i e^{\epsilon_i}} \leq \hat{p}_i \leq \frac{e^{\epsilon_i}}{\sum_i e^{-\epsilon_i}} \quad \forall i \in [1, k], \quad and \quad \sum_i \hat{p}_i = 1$$

Our privacy metric measures the lower bound on entropy, *i.e.*, the least uncertainty in the adversarial guess as $\gamma(\Theta)$. A high value of $\gamma(\Theta)$ means lower privacy loss, as the least uncertainty in adversarial guess is higher. Whereas, a low $\gamma(\Theta)$ means a higher privacy loss. We use this metric to compare the privacy loss of different Θs with the same set of predicates Λ. More details about an algorithm to compute this min-entropy metric are provided in §5.4.

5.2.3 Problem Definition

Consider the accuracy and privacy requirements defined above for decision support applications, we formalize our Accuracy Aware **M**inimally **I**nvasive **D**ata **E**xploration problem (or MIDE in short) as follows. Given an aggregate threshold query $Q^{\Lambda}_{\mathsf{g}(.)>C}$ on a dataset D, we want to develop a set of differentially private mechanisms that answer the query with β-false negative rate guarantee (Def. 5.1) and minimal privacy loss in terms of ex-post privacy loss (Def. 5.3) and min-entropy (Def. 5.4). Among these mechanisms, we want to choose the DP mechanism with the minimal privacy loss.

5.3 Algorithms for MIDE

In the section, we propose three algorithms that solve the MIDE problem. Recall that a decision support query $Q^{\Lambda}_{\mathsf{g}(.)>c}(D)$ consists of a set of predicates $\Lambda = \{\lambda_1, ..., \lambda_k\}$, an aggregate function $g(.)$ and a set of thresholds $C = \{c_1, c_2, ..., c_k\}$. In this chapter, we consider that the predicates in Λ are mutually exclusive and the aggregate function $g(.)$ is a counting function with sensitivity of 1. Extensions to other predicates and aggregates are discussed in the end.

All algorithms aims to satisfy the accuracy requirement of decision support query *i.e.*, the bound on β false negative rate (Definition 5.1). Our first algorithm is based on the modification of a previous work in the literature: APEx [16]. The second algorithm uses the concept of Predicate-wise DP (as introduced in §5.2.1) by iteratively increasing the privacy budget ϵ for each predicate till it reaches its accuracy bound. The third algorithm is a data dependent method that increases the privacy budget adaptively for different predicates in each iteration based on the outcome of the previous iterations.

5.3.1 Threshold-shift Laplace Mechanism

The Laplace Mechanism (Definition 2.4) can be used directly to answer the decision support query of $Q^{\Lambda}_{\mathsf{g}(.)>C}$ in a privacy preserving manner. However, a naive application of this mechanism for this query can result in a large number of false positives and false negatives. We will first illustrate this limitation below, and then introduce an improved application, named as *Threshold-shift Laplace mechanism*, that achieves the required β-false negative rate.

Naive Laplace Mechanism. This mechanism adds a noise η_i to the aggregated count for each predicate λ_i, *i.e.*, $g(D_{\lambda_i})$, where $\eta_i \sim \mathit{Laplace}(0, 1/\epsilon)$. All predicates with noisy aggregate counts that are greater than the query thresholds *i.e.*, $g(D_{\lambda_i}) + \eta_i > c_i$ are returned as the query result. This randomized mechanism makes two types of errors in the output: (i)

false positives which are the predicates with true aggregate $g(D_{\lambda_i}) \leq c_i$ but noisy aggregate $g(D_{\lambda_i}) + \eta_i > c_i$, ; (ii) *false negatives* which have true aggregate $g(D_{\lambda_i}) > c_i$ but noisy aggregate $g(D_{\lambda_i}) + \eta_i \leq c_i$.

If setting the privacy budget for Laplace Mechanism like prior work APEx [16] by $\epsilon = \frac{\ln(1/(2\beta))}{\alpha}$, we can achieve the following accuracy guarantees: with a small probability β, a predicate λ_i with a true aggregate $g(D_{\lambda_i}) > c_i + \alpha$ will have a noisy aggregate smaller than c_i (false negative); a predicate λ_i with a true aggregate $g(D_{\lambda_i}) < c_i - \alpha$ will have a noisy aggregate bigger than c_i (false positive). These guarantees are illustrated in Figure 5.2(i). However, no accuracy are guaranteed (bounded false positive/negative rates) for the predicates with true aggregates falling into the region of $[c_i - \alpha, c_i + \alpha]$. If most of the predicates have aggregates falling in to this uncertain region, the naive Laplace mechanism would output many predicates falsely and fail the accuracy requirement of decision support queries. One approach is to increase the privacy budget to shrink this uncertain region and hence reduce both false positives and false negatives. However, the decision support applications place more importance on the false negatives. We propose the following mechanism to bound the false negatives without increasing the privacy cost.

Threshold Shift Laplace Mechanism. This mechanism aims to achieve a bounded false negative rate for all the predicates (Definition 5.1) unlike the previous naive mechanism. Instead of comparing the noisy aggregates with the initial threshold C in the query $Q^{\Lambda}_{\mathbf{g}(.)>C}$, this mechanism compares each noisy aggregate $g(D_{\lambda_i}) + \eta_i$ with a shifted threshold $c_i - \alpha$, where α is a generalized parameter and noise η_i is based on a privacy budget $\epsilon = \frac{\ln(1/(2\beta))}{\alpha}$. This mechanism then returns all the predicates that have noisy aggregates greater than the shifted thresholds, i.e. $g(D_{\lambda_i}) + \eta_i > c_i - \alpha$.

Figure 5.2(ii) illustrates the guarantees of the new mechanism. Due to the generalization of the threshold from c to $c - \alpha$, the uncertain region with no accuracy guarantees shifts from $[c - \alpha, c + \alpha]$ to $[c - 2\alpha, c]$. This ensures that all the predicates with true aggregates

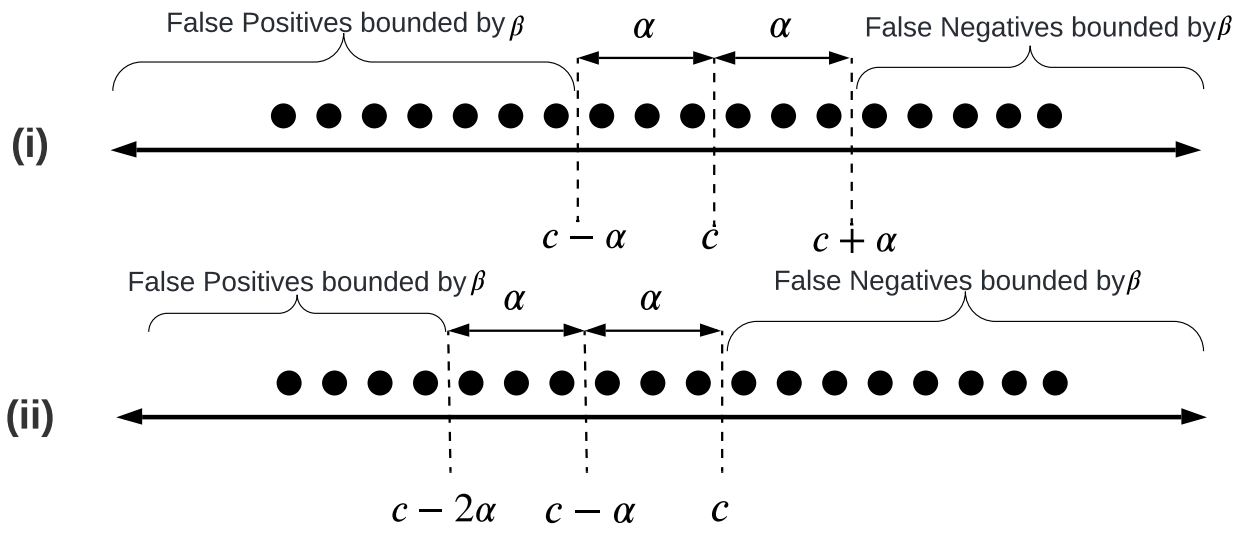


Figure 5.2: The figure shows accuracy guarantees of (i) Naive Laplace Mechanism: noisy aggregates are compared with threshold c (ii) Threshold Shift Laplace Mechanism: noisy aggregates are compared with shifted threshold $c-\alpha$. The dots represent aggregates on the predicates. By shifting the threshold to $c-\alpha$, (ii) achieves β-False Negative Rate (Definition 5.1) as compared to (i) where there is no guarantee on false negatives in the region $[c, c+\alpha]$

Algorithm 1 Threshold Shift Laplace Mechanism.

procedure ThresholdShiftLM($Q^{\Lambda}_{\mathsf{g}(.)>C}, D, \alpha, \beta, \epsilon_{max}$)
  $\epsilon \leftarrow \frac{\ln(1/(2\beta))}{\alpha}$
  if $\epsilon \leq \epsilon_{max}$ **then**
    $O \leftarrow \{\lambda_i \in \Lambda \mid \mathsf{g}(D_{\lambda_i}) + \eta_i > c_i - \alpha, \eta_i \sim Lap(0, 1/\epsilon)\}$
    return O, ϵ
  end if
  return 'Query Denied'
end procedure

greater than the original thresholds are in a guaranteed region, where they would have noisy aggregates smaller than the shifted thresholds and become false negatives with a small probability β.

This mechanism achieves β-false negative rate without increasing the privacy budget compared to the naive Laplace mechanism. Note that in this strategy, the false negative guarantee is independent of the choice of α, but such a guarantee comes at the cost of a potential increase of the false positives, which are the predicates with aggregates falling in the new

uncertain region $[c - 2\alpha, c]$. These predicates should not appear in the output as their true aggregate is smaller than the original thresholds, but their noisy aggregates are very likely greater than the shifted thresholds to output them. We name this region $[c - 2\alpha, c]$ as α-*uncertain region of false positives* for all mechanisms that use a threshold-shift approach. A larger generalization parameter α leads to a larger uncertain region, and can result in more false positives. We will use this generalized parameter α to limit the false positives.

Definition 5.5. *(Uncertainty Region) For each predicate $\lambda_i \in \Lambda$, the Uncertainty Region is based on the threshold $c_i \in C$ and the query generalization parameter α. It is defined the interval $[c_i - 2\alpha, c_i]$. If the predicate λ_i's aggregate value lies in this interval, the algorithm does not provide any bound on probability of λ_i to be in the output to the query as false positive.*

The Threshold-shift Laplace Mechanism is summarized in Algorithm 1. Given the β-false negative rate and α-uncertain region of false post as input, this algorithm first computes the minimal privacy budget to achieve these accuracy requirements, denoted by ϵ (line 2). It also takes the maximum privacy budget allowed for the query ϵ_{max} as input. If the budget is sufficient, then the algorithm proceeds with perturbing the aggregate for each predicate $g(D_{\lambda_i}) + \eta_i$ and returns the ones with noisy aggregates greater than the shifted thresholds $c_i - \alpha$ (line 4); otherwise, the query is denied (line 7). The guarantees of this algorithm are stated as follows.

Theorem 5.4. *Algorithm 1 satisfies ϵ_{max}-DP and β-false negative rate. If the query is not denied, its ex-post DP cost is $\epsilon = \frac{\ln(1/(2\beta))}{\alpha}$.*

5.3.2 Progressive Predicate-wise Laplace Mechanism

If we know that the aggregate value for a predicate λ_i is significantly smaller than its threshold, *i.e.*, $g(D_{\lambda_i}) << c_i$, then having a larger generalization α (which results in a smaller

privacy loss) will still allow this predicate to stay out of the uncertain region of false positive, *i.e.*, $g(D_{\lambda_i}) < c_i - 2\alpha$.

Example 5.1. *Consider two predicates* λ_1, λ_2 *with aggregates* $g(D_{\lambda_1}) = 10$ *and* $g(D_{\lambda_2}) = 150$*, which are smaller than their thresholds* $c_1 = c_2 = 200$*. To achieve* $\beta = 0.01$*-false negative rate using the Threshold Shift Laplace Mechanism, if generalizing the threshold from* 200 *to* 120 *by* $\alpha = 80$ *(which results in* $\epsilon = \ln(1/2(0.01))/(80) = 0.049$*), the first predicate with aggregate value* 10 *is still out of the uncertain region of false positives* $[200 - 2 \cdot 80, 200]$ *and it should be reported correctly with a high probability.However, the aggregate value of the second predicate falls into this* $\alpha = 80$*-uncertain region, and hence it requires a tighter generalization parameter, e.g.* $\alpha' = 40$ *to be in a region with guarantees, which leads to a larger privacy cost* $\epsilon = \ln(1/2(0.01))/(40) = 0.098$*.*

This observation motivates us to design an algorithm that provides different generalizations for the given predicates based on their aggregate values. Since the aggregate values $g(D_{\lambda_i})$ are unknown at first, we start each predicate with a large generalization parameter (and a small privacy budget), and incrementally tightens the generalization parameter (increases the privacy budget) till the predicate can be outputted or pruned with a high certainty. We name this algorithm *Progressive Predicate-wise Laplace Mechanism*, summarized in Algorithm 2.

Besides the same input as the Threshold Shift Laplace mechanism, Algorithm 2 takes in an initial privacy budget of ϵ_1 for the initial generalization and the number of iterations m. As each predicate can be tested at most m times, we aim β/m-false negative rate for each iteration to ensure that the overall false negative rate is bounded by β (Theorem 5.5). First, we estimate the total ϵ_m needed to satisfy the accuracy guarantee over m iterations. If the privacy budget is sufficient, $\epsilon_m < \epsilon_{max}$ (Line 2), we proceed the algorithm; otherwise, the query is denied.

The algorithm starts with ϵ_1 and its corresponding generalization α_1 in the first iteration

Algorithm 2 Progressive Predicate-wise Laplace Mechanism

1: **procedure** PROGRESSIVEPWLM($Q^{\Lambda}_{\mathsf{g}(.)>C}, D, \alpha, \beta, \epsilon_{max}, \epsilon_1, m$)
2: set final privacy cost $\epsilon_m \leftarrow \frac{\ln(1/(2\beta/m))}{\alpha}$
3: set $\epsilon_j \leftarrow \epsilon_1 \cdot \omega^{j-1}$ and $\alpha_j \leftarrow \frac{\ln(1/(2\beta/m))}{\epsilon_j}$ for $j = 1, \ldots, m$, where $\omega = (\frac{\epsilon_m}{\epsilon_1})^{1/(m-1)}$
4: **if** $\epsilon_m \leq \epsilon_{max}$ **then**
5: $[\eta_1, \ldots, \eta_{|\Lambda|}] \leftarrow Lap(1/\epsilon_1)^{|\Lambda|}$
6: $O_d \leftarrow \{\lambda_i \in \Lambda \mid \mathsf{g}(D_{\lambda_i}) + \eta_i > c_i + \alpha_1\}$
7: $O_u \leftarrow \{\lambda_i \in \Lambda \mid \mathsf{g}(D_{\lambda_i}) + \eta_i > c_i - \alpha_1 \wedge \lambda_i \notin O_d\}$
8: **for** $j = 2, \ldots, m$ **do**
9: **if** $O_u = \emptyset$ **then return** O_d, ϵ_{j-1}
10: **end if**
11: $[\eta_1, \ldots, \eta_{|\Lambda|}] =$ PRIVRELAX$(\epsilon_{j-1}, \epsilon_j, [\eta_1, \ldots, \eta_{|\Lambda|}])$
12: $O_d \leftarrow O_d \cup \{\lambda_i \in O_u \mid \mathsf{g}(D_{\lambda_i}) + \eta_i > c_i + \alpha_j\}$
13: $O_u \leftarrow \{\lambda_i \in O_u \mid \mathsf{g}(D_{\lambda_i}) + \eta_i > c_i - \alpha_j \wedge \lambda_i \notin O_d\}$
14: **end for**
15: **return** $O_u \cup O_d, \epsilon_m$
16: **end if**
17: **return** 'Query Denied'
18: **end procedure**

(Lines 5-7). The algorithm increments ϵ_j in each iteration geometrically by a factor of $\omega = (\frac{\epsilon_m}{\epsilon_1})^{\frac{1}{m-1}}$ (Line 3), and the corresponding generalization parameter in the j-th iteration decreases by the same ratio. We consider geometric increments instead of arithmetic increments as smaller increments in the earlier iterations (*i.e.*, using smaller epsilon values) have a higher chance of achieving lower privacy loss. At the j-th iteration, the algorithm adds Laplace noise to the aggregate per predicate based on ϵ_j using Laplace mechanism or using PrivRelax [30]. PrivRelax generates noises for the next iteration j (noises based on ϵ_j) by drawing correlated noises based on the noise drawn in the previous iteration (noises generated using ϵ_{j-1}). This correlated noise ensures that the total privacy loss over the m iterations is bounded by ϵ_m.

We categorize the predicates into three categories: (*i*) *decided*, denoted by O_d, which include predicates with noisy aggregates greater than the generalized thresholds and they are always outputted by the mechanism; (*ii*) *undecided*, denoted by O_u, which include the predicates with noisy aggregates in the range of $[c_i - \alpha_j, c_i + \alpha_j]$, and they are passed to the next

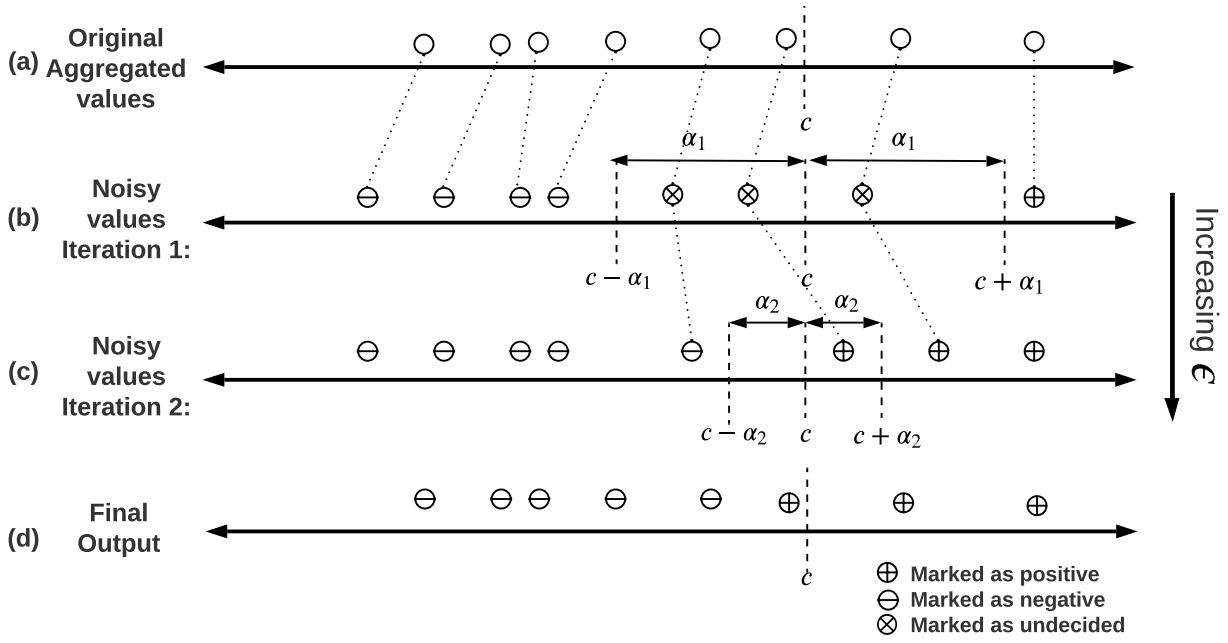


Figure 5.3: PPWLM with 2 iterations. (a) shows the original aggregated counts and the threshold c. (b) and (c) show the noisy aggregated values for each predicate for iteration 1 and iteration 2. In iteration 1, predicates with noisy aggregates $< c - \alpha_1$ are outputted as negatives, those with noisy aggregates $> c - \alpha_1$ are outputted as positives, the remaining are undecided and continue in iteration 2. Iteration 2 outputs all predicates with noisy aggregates $> c - \alpha_2$ as positives.

iteration; and (*iii*) *eliminated*, which are the predicates with noisy aggregates lower than $c_i - \alpha_j$, and they are not considered in the next step or the output of the query. The union of O_d and O_u for each iteration is always a solution that achieves β-false negative rate like the Threshold Shift Laplace Mechanism, but by an iterative tightening of the generalization factor, the number of false positives are improved with a minimal privacy loss. The algorithm terminates when the set O_u is empty *i.e.*, the algorithm has made decisions for all the predicates (Line 9). Otherwise, the algorithm terminates when it has spent the privacy budget of ϵ_m which satisfies the accuracy guarantees of α and β (Line 15). In this situation, the algorithm returns O_u as the answer of the query. The privacy loss in terms of ex-post DP or ex-post PWDP is dependent on the input data and releasing it breaks ϵ_{max}- DP. It is crucial that the ex-post (PW)DP loss is not released to the data analyst (adversary), as it will violate the ϵ_{max}-DP guarantees.

Theorem 5.5. *Algorithm 2 satisfies ϵ_{max}-DP and β-false negative rate. If the query is not denied, its ex-post DP cost is less than $\epsilon_m = \frac{\ln(1/(2\beta/m)}{\alpha}$.*

Figure 5.3 demonstrates the benefits of using this multiple step approach using $m = 2$. Figure 5.3(a) shows the true aggregated values of all predicates and the threshold c In the first iteration, the noisy aggregates (indicated by the position of the dots in Figure 5.3(b)) by spending ϵ_1 are compared against the corresponding generalized threshold $c - \alpha_1$. Four predicates marked negative have smaller noisy aggregates than $c - \alpha_1$ and are eliminated from the next iteration. Among the four predicates with noisy aggregates greater than $c - \alpha_1$, one of them has a noisy aggregate greater than $c + \alpha_1$ and hence it is directly outputted as a positive, while the other three continue to the next iteration. This iteration guarantees that there is a low probability $\beta/2$ for a predicate with true aggregate greater than c to be eliminated. In the second iteration, the newly perturbed aggregates with a larger privacy budget ϵ_2 (Figure 5.3(c)) are compared with a less generalized threshold $c - \alpha_2$. One additional predicate gets eliminated as its noisy aggregate is smaller than $c - \alpha_2$. The final output include 3 predicates. In this example, the final result does not contain any false negatives. Also, five predicates end up using ϵ_1 and three undecided predicates after iteration 1, end up using ϵ_2 privacy budgets. In some cases, the overall privacy loss can be smaller than the previous Threshold Shift Laplace mechanism, if we measure the privacy loss using ex-post Predicate-wise Differential Privacy and min-entropy $\gamma(\Theta)$ as described in §5.2.2.

5.3.3 Data Dependent Mechanism

The algorithm of previous section, (*i.e.*, Algorithm 2) used a fixed number of iterations and updated the privacy parameter and generalization parameter in a geometric manner. This section makes the case that this choice may not be optimal all the time. If the algorithm has knowledge about the data distribution, it can perform better in terms of privacy loss. Since

we are using a multi-step algorithm, we can make use of the noisy aggregated values from the previous iteration to determine the number of iterations and the privacy/generalization parameters for the subsequent steps. We call this algorithm *Data Dependent Progressive Laplace Mechanism*, summarized in Algorithm 3. The privacy loss in terms of ex-post DP or ex-post PWDP is data dependent just like PPWLM so the ex-post (PW)DP loss is not released to the data analyst (adversary) in order to achieve ϵ_{max} DP guarantee.

Algorithm 3 first plans the privacy budgets (Lines 2 - 3), denoted by a vector B of m entries, in a way similar to Algorithm 2. In the first iteration, it still starts with ϵ_1 and stores the noisy aggregates G. Based on the noisy aggregates, the predicates are classified into three categories, decided positives O_d, undecided ones O_u, and decided negatives $(\Lambda - O_d - O_u)$. For all the predicates with a confident decision (*i.e.*, decided positives and decided negatives), their ex-post privacy cost stop at ϵ_1 and are saved in a vector E while the others in O_u are temporarily set to be the final cost ϵ_m (Line 9). In the next iteration, rather than using the planned privacy budget stored in B, we use the noisy aggregates G and the temporary ex-post privacy cost E to estimate the best privacy level that maximizes the min-entropy $\gamma(\Theta)$.

The estimation of the best privacy level for the next iteration is presented in Algorithm 4. It searches the privacy level ϵ_{next} for the next iteration in the remaining privacy levels in B and for each privacy level in B, it also further divides the intervals into m_f number of fine-grained steps (Line 4). The algorithm aims to find an ϵ_{next} that can lead to a predicate-wise privacy loss E' with a largest min-entropy; hence, the algorithm will be able to skip all the privacy levels before ϵ_{next} (Lines 5- 10). The algorithm removes the unused privacy levels from the budget plan B and updates the corresponding β for the next iteration (Line 12).

We cannot compute the exact predicate-wise privacy loss without running the algorithm. To estimate this privacy loss, the algorithm first uses the noisy aggregates G to compute how many of the undecided predicates from previous iteration O_u will still remain undecided

if a privacy level of ϵ_{next} is used in the current iteration. For each predicate $\lambda_i \in O_u$, the algorithm estimates its probability of remaining undecided (*i.e.*, its new noisy aggregate $g(D_{\lambda_i}) + \eta_i'$ falls into the range of $[c_i - \alpha_j, c_i + \alpha_j]$) by using its noisy aggregate $G[i]$ which was perturbed by η_i at a privacy level ϵ_{j-1} from the previous iteration; and then sum them up as an expected number for the undecided predicates:

$$\begin{aligned} n_u &= \sum_{\lambda_i \in O_u} P(g(D_{\lambda_i}) + \eta_i' \in [c_i - \alpha_j, c_i + \alpha_j]) \\ &\approx \sum_{\lambda_i \in O_u} \int_{c_i - \alpha_j}^{c_i + \alpha_j} \int_{-\infty}^{\infty} \frac{\epsilon_{j-1}}{2} e^{-|x - G[i]|\epsilon_{j-1}} \times \frac{\epsilon_j}{2} e^{-|z-x|\epsilon_j} dx dz \end{aligned} \tag{5.9}$$

Theorem 5.6. *Algorithm 3 satisfies ϵ_{max}-DP and β-false negative rate. If the query is not denied, its ex-post DP cost is $max(E)$.*

This data dependent algorithm comes at computation cost as we choose ϵ in each iteration based on min-entropy. In the worst case scenario, the cost of computing min-entropy can be exponential in terms of number of predicates; hence it may incur high computation overhead the when number of predicates are very high. We present an efficient algorithm to compute this cost next.

5.4 Computing Privacy Loss

We use an entropy based privacy metric for PWDP to compute the privacy loss of our multi-step algorithms (*i.e.*, PPWLM and DPPWLM). Furthermore, we use this metric to estimate the optimal ϵ values in each iteration to minimize the privacy loss in DPPWLM.

Our privacy metric for PWDP measures the lower bound on entropy, *i.e.*, the least un-

Algorithm 3 Data Dependent Progressive Mechanism DPPWLP

procedure DPPWLM($Q^{\Lambda}_{\mathbf{g}(.)>C}, D, \alpha, \beta, \epsilon_{max}, \epsilon_1, m, m_f$)
  set final privacy cost $\epsilon_m \leftarrow \frac{\ln(1/(2\beta/m))}{\alpha}$
  set $B[j] = \epsilon_1 \omega^{j-1}$ for $j \in [1, m]$, where $\omega = (\frac{\epsilon_m}{\epsilon_1})^{1/(m-1)}$
  if $\epsilon_m \leq \epsilon_{max}$ **then**
    $[\eta_1, \ldots, \eta_{|\Lambda|}] \leftarrow Lap(1/\epsilon_1)^{|\Lambda|}$
    set $G[i] = g(D_{\lambda_i}) + \eta_i$ for $\lambda_i \in \Lambda$ and $\alpha_1 = \frac{\ln(1/(2\beta/m))}{\epsilon_1}$
    $O_d \leftarrow \{\lambda_i \in \Lambda \mid G[i] > c_i + \alpha_1\}$
    $O_u \leftarrow \{\lambda_i \in (\Lambda - O_d) \mid G[i] > c_i - \alpha_1\}$
    Initialize predicate epsilon $E[i] = \epsilon_1$ if $\lambda_i \in (\Lambda - O_u)$; for the other predicates, it with final cost $E[i] = \epsilon_m$
    initalize $j \leftarrow 1$
    while $\epsilon_j \leq \epsilon_m$ and $O_u \neq \emptyset$ **do**
      $j \leftarrow j + 1$
      $\epsilon_j, \beta_j, B = \textsc{NextStepParams}(E, G, B, O_u, \beta, m, m_f)$
      $[\eta_1, \ldots, \eta_{|\Lambda|}] = \textsc{PrivRelax}(\epsilon_{j-1}, \epsilon_j, [\eta_1, \ldots, \eta_{|\Lambda|}])$
      set $G[i] = g(D_{\lambda_i}) + \eta_i$ for $\lambda_i \in O_u$, $\alpha_j = \frac{\ln(1/(2\beta_j))}{\epsilon_j}$
      $O_d \leftarrow O_d \cup \{\lambda_i \in O_u \mid G[i] > c_i + \alpha_j\}$
      $O'_u \leftarrow O_u$, $O_u \leftarrow \{\lambda_i \in (O_u - O_d) \mid G[i] > c_i - \alpha_j\}$
      set predicate epsilon $E[i] = \epsilon_j$ if $\lambda_i \in (O'_u - O_u)$
    end while
    return $O_u \cup O_d, \epsilon_j$
  end if
  return 'Query Denied'
end procedure

Algorithm 4 Estimated Epsilon for next step in DPPWLM

procedure $\textsc{NextStepParams}(E, G, B, O_u, \beta, m, m_f)$
  initalize $ent_{max} = 0$, $\epsilon_{next} = B[0]$,
  $r_{opt} = 1$, $\epsilon_{opt} = \epsilon_{next}$
  for $r \in [1, \ldots, |B| - 1]$ and $s \in [1, \ldots, m_f]$ **do**
    $\epsilon_{next} \leftarrow \epsilon_{next} + \frac{(B[r+1] - B[r])}{m_f}$
    $E' \leftarrow E$ and choose $(|O_u| - n_u)$ number of predicates from O_u and set their $E'[\lambda_i] = \epsilon_{next}$
    $ent_{next} = \textsc{MinEnt}(b_{E'})$ ▷ $b_{E'}$ are bounds on $\hat{p}_i$ (Eq A.2) based on E'.
    if $ent_{max} \leq ent_{next}$ **then**
      $ent_{max} \leftarrow ent_{next}$, $r_{opt} \leftarrow r$, $\epsilon_{opt} \leftarrow \epsilon_{next}$
    end if
  end for
  return $(\epsilon_{opt}, \frac{\beta \cdot r_{opt}}{m}, B[r_{opt} + 1 :])$
end procedure

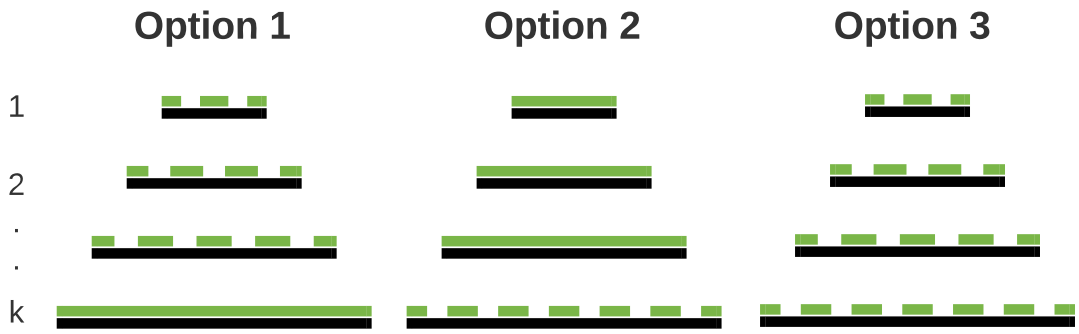


Figure 5.4: Possible options at k-th step of MinEnt algorithm. Option 1 distributes as much slack as possible to $\hat{p}_k$ (solid green line) and the rest to $\hat{p}_1, \ldots, \hat{p}_{k-1}$ (dotted green line). Option 2 distributes as much slack as possible to $\hat{p}_1, ..., \hat{p}_{k-1}$ and the rest to $\hat{p}_k$. Option 3 distributes slack to $\hat{p}_1, \ldots, \hat{p}_{k-1}$ and $\hat{p}_k$ instead of distributing as much as possible to either.

certainty in the adversarial guess as follows: $\gamma(\Theta) = min(\ \sum_{i=1}^{k} -\hat{p}_i \log \hat{p}_i)$, subject to $\frac{e^{-\epsilon_i}}{\sum_i e^{\epsilon_i}} \leq \hat{p}_i \leq \frac{e^{\epsilon_i}}{\sum_i e^{-\epsilon_i}}$ and $\sum_i \hat{p}_i = 1$. This is a concave optimization problem with constraints. Finding the global minima with constraints for a concave function is computationally difficult since the function may have several local minimas [50]. However, finding the minima of the sum of entropy functions is a tractable problem, since the shape of entropy function is known and simple (*i.e.*, with only one maxima instead of multiple local maxima). We leverage this idea to develop a dynamic programming based algorithm that finds the global minima of the sum of entropy functions, *i.e.*, to compute $\gamma(\Theta)$.

Given Θ, *i.e.*, a set of k predicates with their epsilons, the algorithm first computes their corresponding lower bounds ($l_i = \frac{e^{-\epsilon_i}}{\sum_i e^{\epsilon_i}}$) and upper bounds ($u_i = \frac{e^{\epsilon_i}}{\sum_i e^{-\epsilon_i}}$) and then sort them based on their upper bounds in ascending order as an input to Algorithm 5. For simplicity, we assume that $u_1 \leq u_2 \cdots \leq u_k$ without introducing new indices.

If we start by allocating each $\hat{p}_i$ with its lower bound l_i, there is a remaining amount $s = (1 - \sum_{i=1}^{k} l_i)$ which has to be distributed to among $\hat{p}_i$s to ensure $\sum_i \hat{p}_i = 1$ and $\hat{p}_i \leq u_i$ while

minimizing the entropy function. We call this remaining amount *slack*. The maximum slack that can be distributed to $\hat{p}_i$ is bounded by $\Delta_i = u_i - l_i$. We consider three options that cover all possible distributions of the slack s among the k predicates:

- **Option 1**. Distribute as much slack as possible to the $\hat{p}_k$ (the one with the largest upper bound).
- **Option 2**. Distribute as little slack as possible to the $\hat{p}_k$, and hence distribute as much slack as possible to $\hat{p}_1, \ldots, \hat{p}_{k-1}$.
- **Option 3**. Unlike the previous two options, here the slack is divided between $\hat{p}_k$ and the sub-problem of size $k-1$ *i.e.*, $\hat{p}_{k-1}, ..., \hat{p}_1$ without fully allocating to either of them.

These three options are illustrated in Figure 5.4. The figure represents the interval of $[l_i, u_i]$ from $i = 1, \ldots, k$. Note that a lower ϵ_i value will have a higher l_i and a lower u_i value. For option 1, if the slack $s > \Delta_k$, there is still remaining slack to be distributed among the $k-1$ predicates. This gives a sub-problem of size $k-1$, i.e. distributing the new slack $s' = (s - \Delta_k)$ among the first $(k-1)$ predicates. For option 2, if the slack $s < \sum_{i=1}^{k} \Delta_i$, then the remaining slack will be added to $\hat{p}_k$; otherwise, we need to solve a sub-problem of size $k-1$, i.e., distributing the full slack s among the first $(k-1)$ predicates. We don't need to solve additional sub-problem. For option 3, we can show that it always results in a poorer solution than the solution coming from option 1 or option 2.

Theorem 5.7. *Given a set of intervals of posterior probabilities* $\{(l_i, u_i) \mid i = 1, 2, ...k\}$ *and a slack* s *to be distributed among the intervals, the option 3 always performs worse than either the strategies of option 1 or option 2 in terms of minimizing entropy.*

Hence, Algorithm 5 considers only option 1 and option 2 and only option 1 requires solving a sub-problem with a smaller number of predicates. At the base case when $k = 1$, all the slack is allocated to this predicate (Line 2). When $k > 1$, we consider option 1 and option 2 described above. For option 1, the solution is stored in $p1$ (Lines 5-6) which requires solving

Algorithm 5 Minimize Entropy

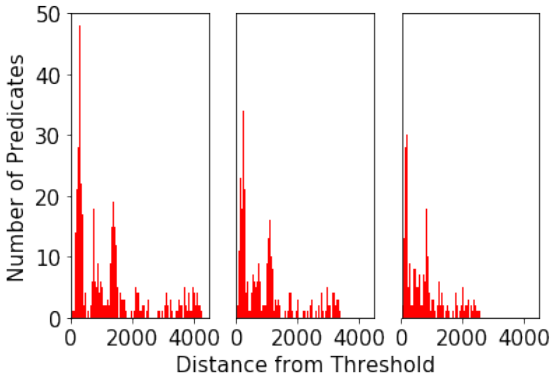
1: **procedure** MinEnt($\{(l_i, u_i) \forall i \in \{1, 2, ..., k\}\}, s$) ▷ sorted by u_i in ascending order. Initially, $s = (1 - \sum_{i=1}^{k} l_i)$ is a slack variable.
2: **if** $k = 1$ **then return** $[l_1 + s]$
3: **end if**
4: $\Delta_i = u_i - l_i, \forall i \in \{1, 2, ..., k\}$
5: $p1[k] = l_k + min(\Delta_k, s)$
6: $p1[1 : k-1] =$MinEnt$([l_i, u_i] \forall i \in \{1, ..., k-1\}, s - min(\Delta_k, s))$
7: $p2[1 : k-1] =$MinEnt$([l_i, u_i] \forall i \in \{1, ..., k-1\}, min(\sum_{i=1}^{k-1} \Delta_i, s))$
8: $p2[k] = l_k + s - min(\sum_{i=1}^{k-1} \Delta_i, s)$
9: **if** CalEntropy$(p1)$ ¡ CalEntropy$(p2)$ **then return** $p1$
10: **else return** $p2$
11: **end if**
12: **end procedure**
13: **procedure** CalEntropy$(\ p)$
14: **return** $-\sum_{i=1}^{|p|} p[i] \log(p[i])$
15: **end procedure**

a sub-problem for the first $(k-1)$ predicates with the remaining slack $s - min(\Delta_k, s)$. For option 2, the solution is stored in $p2$ (Lines 7-8) which requires solving a sub-problem for the first $(k-1)$ predicates with the full slack s. The solution with higher entropy is returned.

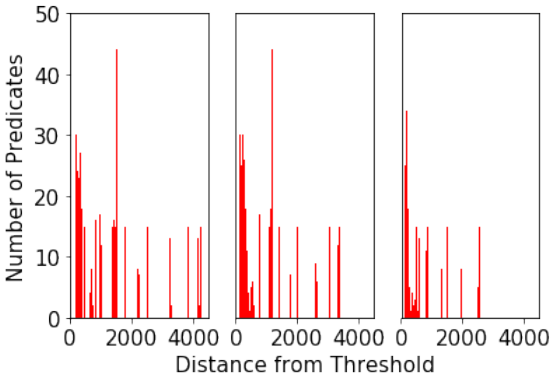
Theorem 5.8. *Algorithm 5 outputs the optimal solution to the min-entropy problem* $\gamma(\Theta)$.

5.5 Experiments

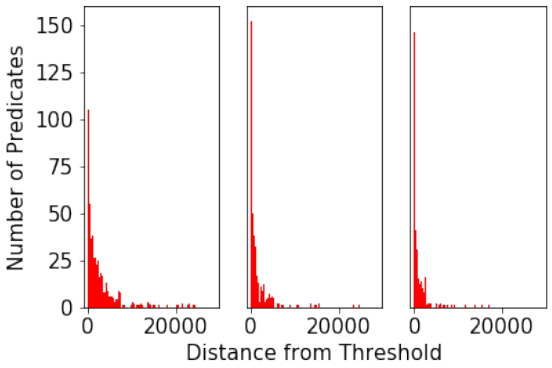
This section evaluates our algorithms (Algorithms 1,2, and 3) for MIDE using various queries taken from real life scenarios and over real datasets. This is to show that all the algorithms effectively achieve their accuracy guarantees in terms of bounded false negative rate; among them, the data dependent mechanism (Algorithm 3) obtains the lowest minimal privacy cost over most of the queries.



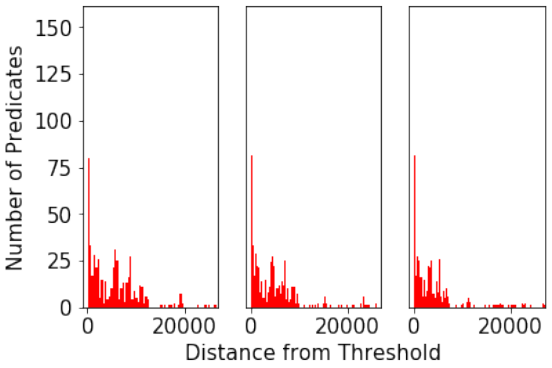
(a) Q1H/M/L



(b) Q2H/M/L

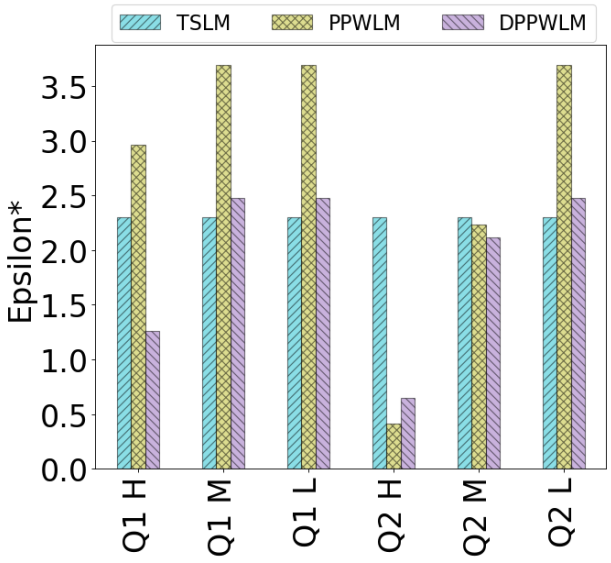


(c) Q3H/M/L

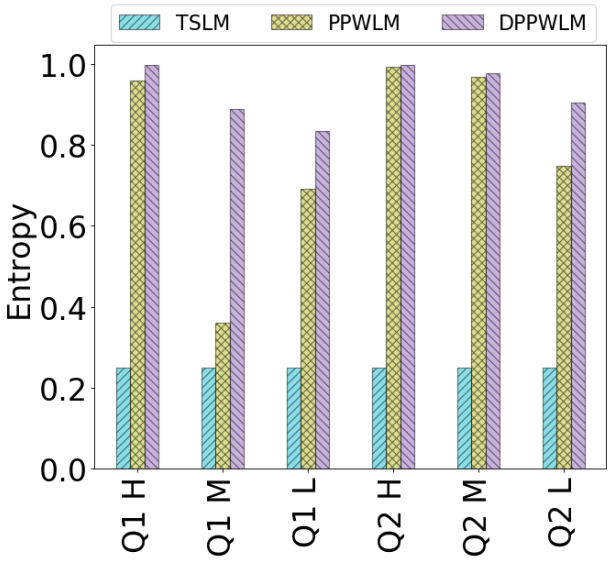


(d) Q4H/M/L

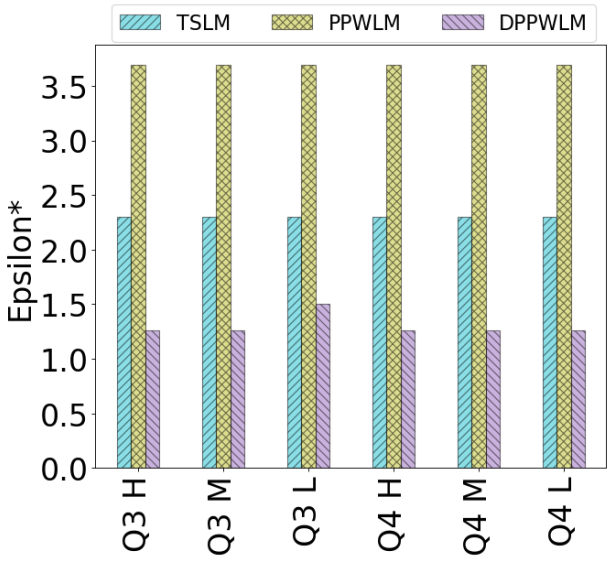
Figure 5.5: This figure shows the distribution of the distances from the thresholds for all aggregates for $Q1, Q2, Q3, Q4$ with thresholds = High (H), Medium (M) and Low (L).



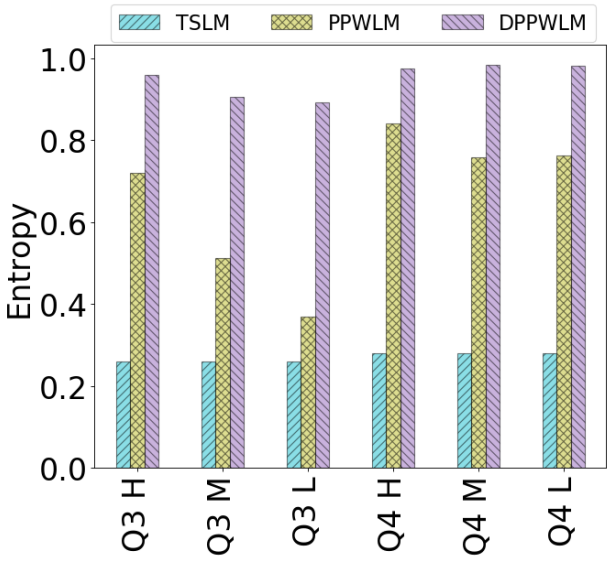
(a) UCI data $(Q1, Q2)$ Epsilon



(b) UCI data $(Q1, Q2)$ Entropy



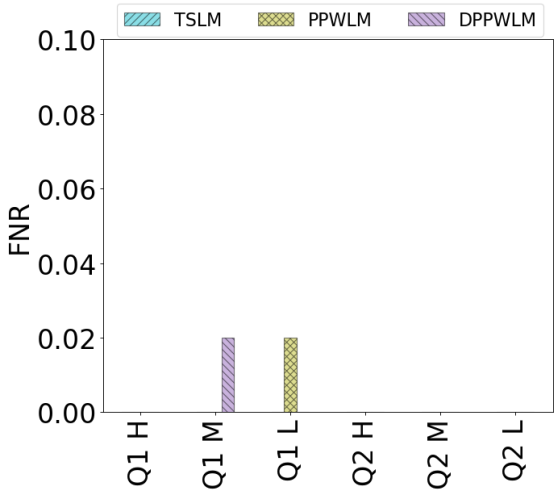
(c) NYTaxi data $(Q3, Q4)$ Epsilon



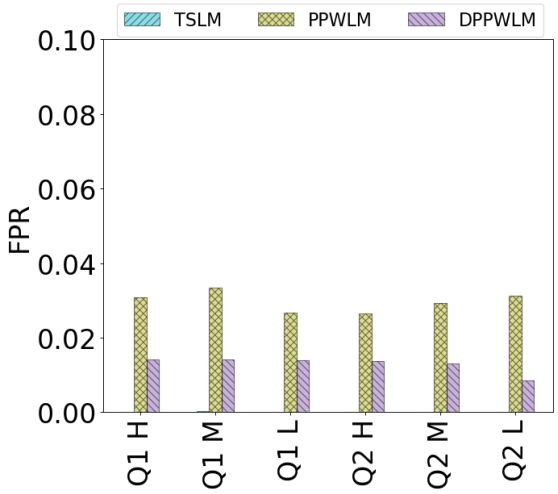
(d) NYTaxi data $(Q3, Q4)$ Entropy

Figure 5.6: Privacy loss in terms of ϵ^*(Ex-Post DP) and Min-Entropy $\gamma(\Theta)$ for Q1, Q2, Q3, Q4 with threshold = High (H),Medium (M), Low (L) at $\beta = 0.05$, $\alpha = 1$

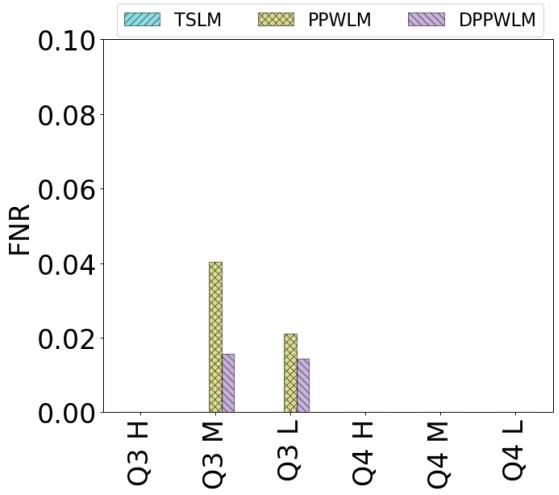
.



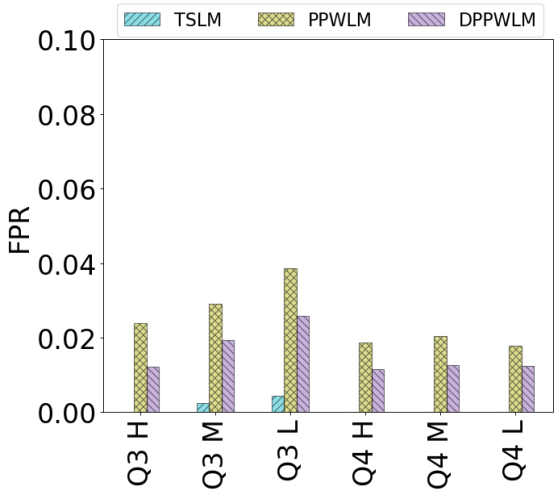
(a) UCI data $(Q1, Q2)$ FNR



(b) UCI data $(Q1, Q2)$ FPR

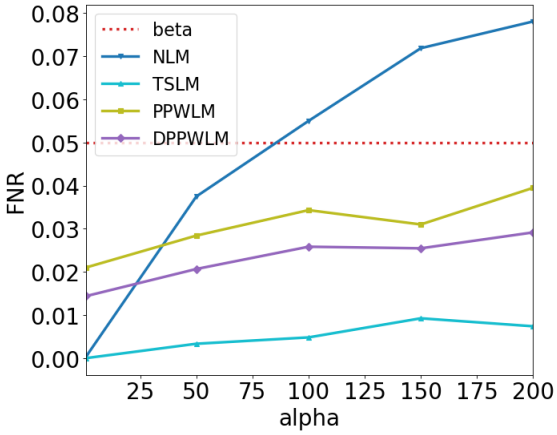


(c) NYTaxi data $(Q3, Q4)$ FNR

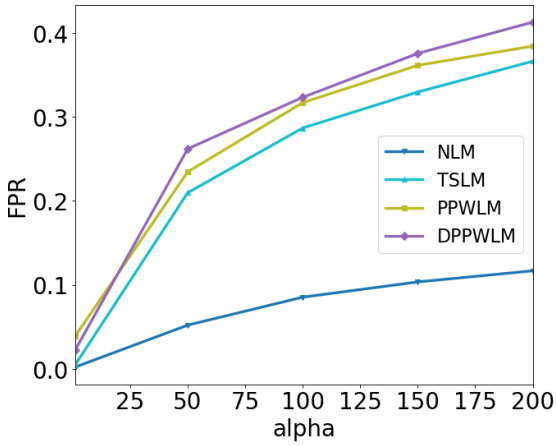


(d) NYTaxi data $(Q3, Q4)$ FPR

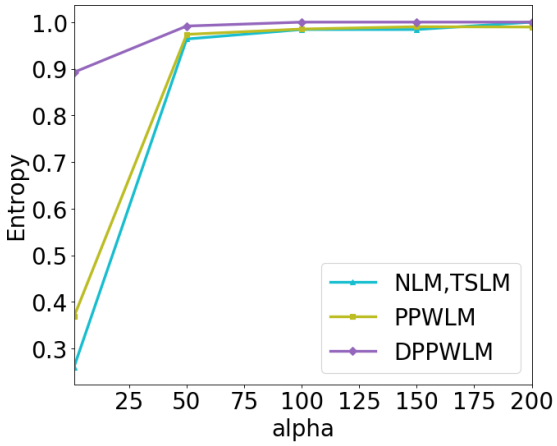
Figure 5.7: Accuracy in terms of False Negative Rate (FNR) and False Positive Rate (FPR) for Q1, Q2, Q3, Q4 with threshold = High (H), Medium (M), Low (L) at $\beta = 0.05$, $\alpha = 1$.



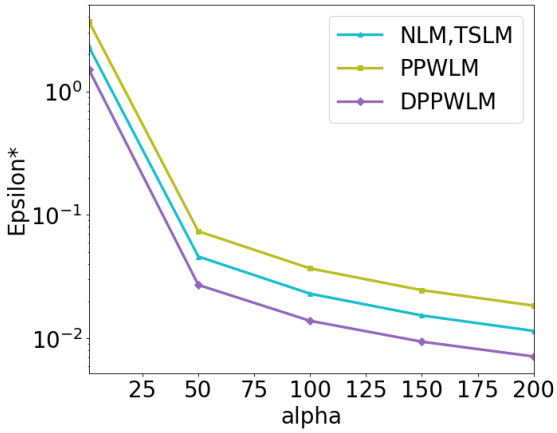
(a) FNR



(b) FPR



(c) Min-Entropy $\gamma(\Theta)$



(d) Ex-Post Privacy Loss ϵ^*

Figure 5.8: Accuracy (FNR,FPR) and Privacy (ϵ^*,$\gamma(\Theta)$) for $Q3$ (NYTaxi data) with threshold = Low over varying α.

5.5.1 Setup

Datasets & Queries. We used two real-world datasets and designed queries for the evaluation as described below.

UCI Dataset. This dataset contains the occupancy data of 24 different buildings of University of California, Irvine campus collected in 2018 October [44]. The data consists of 3 million records where attributes are userID, location, time. The DS queries find out the anomalous incidents (*e.g.*, violation of fire safety norm setup by the California fire department), *i.e.*, buildings with occupancy (number of individuals) that was higher than their capacity. We run 2 queries: $Q1$ on a weekday (Oct 09) and $Q2$ on a weekend (Oct 13) that has different data distributions. Both queries check every hour between 7 a.m. to 10 p.m. if a building's occupancy is exceeding the threshold. Total number of predicates for both $Q1$ and $Q2$ are 15(number of hours)$\times$24(number of buildings)$=$ 420. $Q1$ and $Q2$ are also coupled with three levels of thresholds (high, medium, low), set as $1, 0.8$, and 0.6 times of the building capacities.

NYTaxi Dataset. This dataset records taxi trips in New York City in 2020 [1], consisting of 15.7 million records with 18 attributes, *e.g.*, pick-up and drop-off locations and their timestamps. We group the pickup locations into 34 different regions and run queries to find out the regions and timestamps that had anomalous pickup counts. We run two queries: $Q3$ is run on March (1-14) (before the lockdown); and $Q2$ is run on March (15-30) (after the lockdown). Both queries check for each day in the corresponding time range if a regions's pickup count was higher than the threshold for all 34 regions. Total number of predicates for $Q3$ are 34 (regions)$\times$14(days)$=$ 476 and for $Q4$ are 34 (regions)$\times$16(days)$=$ 544. For each predicate, we use the maximum number of pickups from Jan and Feb times a multiplicative factors of 1, 0.8, 0.6 as the high, medium, low thresholds.

We display the distributions of the absolute distance of the aggregates in each query from their corresponding thresholds in Figure 5.5. We use uniform priors for these datasets to

compute min-entropy.

Algorithms & Parameters. We consider three MIDE algorithms: Threshold Shifted Laplace Mechanism (TSLM), Progressive Predicate-wise Laplace Mechanism (PPWLM), and Data Dependent PPWLM (DPPWL). The naive Laplace Mechanism (NLM) is evaluated at the same privacy cost as TSLM as a baseline for accuracy.

Our accuracy requirements is defined in terms of two parameters: β-false negative rate and α-uncertain region of false positives. We consider values for $\beta \in \{0.01, 0.02, \ldots, 0.1\}$ and $\alpha \in \{1, 10, 20, \ldots, 200\}$. The default values are $\beta = 0.05$ and $\alpha = 1$. For algorithms with multiple iterations including PPWLM and DPPWLM, we set the starting epsilon ϵ_1 be 0.00001, the total number of iterations to be $m = 4$, the maximum value without exceeding $\epsilon_{max} = 4$ at the default choice for α and β. For DPPWLM, we set the fine grained steps $m_f = 3$. We run each algorithm 100 times and report their averaged privacy or utility metrics.

5.5.2 Experimental Results

Privacy Results. We compare the algorithms based on two privacy metrics: ex-post DP, denoted by ϵ^*, and min-entropy for predicate-wise DP, $\gamma(\Theta)$. For TSLM, all predicates end with the same epsilon values, and hence the same lower and upper bounds for the posteriors to compute the min-entropy (Definition 5.4) using Algorithm 5. The privacy results for 4 queries ($Q1$-$Q4$) with their corresponding threshold levels (denoted by H,M,L) are presented in Figure 5.6 when setting the accuracy parameters $\beta = 0.05$ and $\alpha = 1$.

DPPWLM achieves a privacy cost that is near to the lowest or the lowest for all the queries. As it uses a multi-step approach, it allows earlier stop and hence a smaller ex-post DP cost than a single-step method TSLM for Q1H,Q2H/M, Q3H/M/L,Q4H/M/L, as shown in

Figures 5.6a and 5.6c. DPPWLM does not always have an earlier stop, which depends on data distribution. For Q1M/L and Q2L, the distances of the counts from the thresholds shown in Figures 5.5a and 5.5b are relative small for most of the predicates, i.e., the counts are closer to thresholds. For such a case, all predicates need to consume a high privacy budget to be accurately decided and incur a slightly higher ex-post privacy than TSLM due to the division of the β among multiple steps. However, it is better than the other multi-step approach PPWLM, because DPPWLM uses learned data distribution to determines the number of iterations and hence budget allocation adaptively. Furthermore, as DPPWLM optimizes min-entropy, we observe that it achieves the highest min-entropy for all the queries as shown in Figures 5.6b and 5.6d.

Accuracy Results. For each run of the algorithm, we measured the number of false negatives n_{fn} and the number of false positives n_{fp}. Then we report the averaged false negative rate (FNR) as n_{fn}/n_p and the averaged false positive rate (FPR) as n_{fp}/n_n over multiple runs, where n_p and n_n are the number of positives and the number of negatives respectively. The results are presented in Figure 5.7 when $\beta = 0.05$ and $\alpha = 1$.

Figures 5.7a and 5.7c show that all the MIDE algorithms achieve *a bounded FNR* lower than $\beta = 0.05$, which is the key accuracy requirement of DS. Note that the multi-step approach DPPWLM can make different decisions in each step (e.g., epsilon values) depending on the randomness of the algorithm and the data distribution, so there is no guarantee that DPPWLM will always win PPWLM in terms of utility (e.g., DPPWLM has a lower FNR than PPWLM for all queries except Q1M), but both of them have a bounded FNR. The trade-off of FNR in terms of FPR is relatively low, less than 0.04, for all MIDE algorithms and queries shown in Figures 5.7b and 5.7d.

Accuracy-Privacy Tradeoffs. TSLM achieves a better utility (FPR and FNR, and FPR/FNR tradeoff) than multi-step algorithms, but at a privacy cost. Since DPPWLM

performs better than PPWLM (in both privacy cost and utility), we focus on the tradeoff comparisons between DPPWLM and TSLM. The privacy goal of DPPWLM is to optimize min-entropy (a higher min-entropy is preferred). The utility goal is to achieve a bounded FNR and optimize FPR (a smaller FPR is preferred). We compare its min-entropy (Figures 5.6b/5.6d) and its FPR (Figures 5.7b/5.7d) with TSLM. On average, DPPWLM improves the min-entropy of TSLM from 0.25 to a value above 0.8 in Figures 5.6b/5.6d, while it sacrifices the FPR of TSLM from ˜0 to a value at most 0.034 in Figures 5.7b/5.7d for all the queries.

Comparison with Naive Laplace Mechanism. We use $Q3$ with threshold = 'low' for the comparison between the naive laplace mechanism (NLM) and our algorithms in Figure 5.8 by changing the accuracy parameter α. As there is no guideline for setting the parameter of NLM to achieve β-FNR, we use the same privacy budget for NLM as TSLM. When α increases, the privacy budget becomes smaller. Figure 5.8a shows that NLM does not satisfy β-false negative guarantee as α increases while the other algorithms still have a bounded FNR. Figure 5.8b shows that the trade off in terms of false positives for false negatives is data dependent. If many true negatives lie close to thresholds (most of our datasets), then the trade-off cost is high. The NLM has the same ex-post privacy loss and min-entropy as our TSLM as both algorithm use the same privacy budget. The results for our privacy metric (Figure 5.8c, 5.8d) show that our DPPWLM has the lowest privacy loss across different values of α. Similar results are observed when changing β.

Varying Parameters for Multi-step Algorithms. We evaluated our multi-step algorithms with varying starting epsilon values $\epsilon_1 \in \{10^{-5}, 10^{-4}, \ldots, 10^{-1}\}$ and varying number of steps $m \in \{2, 4, \ldots, 12\}$. Due to space constraint, we leave the plots in Appendix and summarize the results here.

As ϵ_1 increases, PPWLM and DPPWLM have a larger privacy loss (both ex-post DP and

min-entropy). If ϵ_1 is too small, all predicates may be undecided in the first step in both approaches. However, DPPWLM chooses appropriate epsilons in the subsequent steps to effectively classify the predicates. When changing ϵ_1, there are no significant differences in utility and fulfilling the required accuracy bounds. The utility improves slightly if DPPWLM ends with a relatively higher privacy loss due to the data distribution and choice of ϵ and β in the intermediate steps.

Our experiments show that increasing the number of steps by more than 4 can result in a higher ex-post DP loss for both PPWLM and DPPWLM as ϵ_m for the last step will exceed $\epsilon_{max} = 4$. On the other hand, choosing a smaller number of steps may not result in an optimal solution as a data dependent algorithm becomes limited in the optimal choice of epsilon. The DPPWLM does better in min-entropy than PPWLM with a larger m as DPPWLM optimizes the choice of ϵ and β to maximize the min-entropy. The utility satisfies the required bound and varies slightly depending on the data distribution and the choice of ϵ and β across multiple steps.

5.6 Conclusion

In conclusion, we presented minimally invasive data exploration for decision support applications for basic form of decision support queries (aggregate threshold queries). We formally defined the accuracy requirement and presented three different privacy preserving algorithms that aim to minimize privacy loss while providing accuracy guarantees. We provided a new privacy definition to capture predicate-wise privacy loss and an entropy based privacy metric. Our results show that our data-dependent algorithm is robust and minimizes privacy loss for different data distributions.

Chapter 6

Accuracy Aware Privacy Preserving Decision Support with Complex Queries

6.1 Introduction

In this chapter, we study privacy in the context of complex decision support queries. Previous work in DS queries (e.g. MIDE) deals with simple DS query that classifies objects either true or false based on the condition on a single aggregate statistic. This work considers more complicated DS queries based on conjunctive/disjunctive conditions on multiple aggregate statistics on an object. We formally define such queries and their accuracy requirement and provide algorithms that attempt to minimize privacy loss while supporting bounded guarantee on the accuracy. Our preliminary experiments show that our algorithms satisfy accuracy guarantees and minimize privacy loss.

On-line analytical processing (OLAP) and data warehousing are essential elements of decision

support that are widely used in organizational decision making. For example, in OLAP (On Line Analytic Processing), the analyst can use the information in database to guide strategic business decisions. In a typical OLAP system, the data is aggregated over n-dimensional data cube and can be used to learn aggregated statistics e.g. how many items of a particular clothing were sold in all northeast stores in the years 2021 and 2022? This kind of statistics can be used to make important business decisions e.g. deciding whether the company should keep selling a particular product. The commonality in such DS applications is that the aggregated statistics are collected and compared to a preset threshold that classifies objects as either satisfying the predicate (i.e. true), or as not satisfying the predicate (i.e. false). The aggregated statistics used in decision making are derived from databases which may contain sensitive information about individuals. Most widely used privacy preserving techniques for query analytics (e.g. Differential Privacy [13]) protect user privacy at the cost of utility. If the data is not accurate or noisy, the decisions made using privacy preserving decision support system can be misleading and inaccurate. Therefore, the goal in answering decision support queries is to preserve privacy of individual while ensuring required accuracy level. Previous work (Chapter 5) presents **MIDE** that can be used to answer Decision Support queries in a privacy preserving way while ensuring required accuracy level. However, MIDE only deals with most basic form of Decision Support query where we have a single statistical measure that is compared to a corresponding threshold. Decision Support queries can be a combination (conjunction/disjunction) of multiple conditions where each condition consists of a statistical value compared to it's corresponding threshold. For example, in sales DS systems, to gauge if a product is profitable in North America, we may want to evaluate if profits in USA exceed a corresponding threshold and profits in Canada exceed a corresponding threshold (conjunction of two conditions). Similarly, we may want to provide an alert when a specific group of items reaches the minimum profit or exceeds the amount of sales from past quarter (disjunction of two conditions).

In business applications, KPIs (Key Performance Indicators) provide an analytical basis for

such decision making queries. KPIs are defined as business metrics that help gauge the progress towards an intended goal and play a crucial role in monitoring and evaluating the processes carried out by the company. A KPI handles information about a set goal, the performance recorded in the data cube in OLAP, and a status value. Typically, to make a decision, analyst can use multiple conditions over the boolean status values of multiple KPIs.

In this chapter, we formally define complex decision support queries and provide algorithm to evaluate such queries in a privacy preserving manner that results in minimal privacy loss and satisfies the accuracy requirement for such queries. The organization of this chapter is as follows. In Section 6.2, we first define the complex decision support queries, provide accuracy requirements for such queries and describe our problem statement. In Section 6.4, we provide differentially private algorithms that can be used to answer complex DS queries while satisfying accuracy guarantees and minimizing privacy loss. In Section 6.5, we provide preliminary experiments to evaluate our algorithm using real data set. In Section 6.6, we discuss conclusion and future work.

6.2 Query Definition

In this section, we present complex decision support queries. First, we generalize the basic decision support query (aggregate threshold query) as described in Chapter 5 and then define more complex DS queries i.e. conjunction and disjunction queries. We generalize the aggregate threshold query by adding an optional filter clause to the query. Such a query checks whether the aggregated values computed on a subset of tuples pass the thresholds or not. Formally,

Aggregate Threshold Query. An aggregate threshold query, denoted by $Q^{\Lambda,f}_{\mathbf{g}(.)\theta C}$, consists

of (i) an aggregate function $\mathtt{g}(.)$; (ii) a set of predicates $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_k\}$; (iii) a operator θ (iv) a set of corresponding thresholds $C = \{c_1, c_2, ..., c_k\}$ and (v) an optional filter f. The filter f takes in a tuple and outputs *True* or *False* based on the value of the tuple. We use D^f to denote all the tuples that satisfy the filter. Each predicate λ_i takes in a tuple from filtered tuples D^f and outputs *True* or *False* based on the value of the tuple. We let $D^f_{\lambda_i}$ be the set of tuples in D that evaluate λ_i to be *True*. This query returns all the predicates that have the aggregate $g(D^f_{\lambda_i})$ satisfy the condition $g(D^f_{\lambda_i})\theta c_i$ where θ is a comparison operator such as ¿,¡,= and c_i is their respective threshold. Formally,

$$Q^{\Lambda,f}_{\mathtt{g}(.)\theta C}(D) = \{\lambda_i \in \Lambda \mid \mathtt{g}(D^f_{\lambda_i})\theta c_i\} \tag{6.1}$$

This is equivalent to group-by-having query in SQL. Given a sales dataset with schema $SALES\_DATA(product\_id, location, profit, timestamp)$, the following is an example of aggregate threshold query.

```
SELECT product_id FROM SALES_DATA
Where location = 'USA'
GROUP BY product_id HAVING count(*) > c
```

The *where* clause $age > 25$ is an example of a filter f, $product\_id = p_1, product\_id = p_1, ...product\_id = p_k$ are examples set of k predicates i.e. Λ, $count()$ is the aggregate function $g(.)$ and threshold is same for all the predicates i.e. c. In the rest of this chapter, we focus on $\theta = '>'$ and discuss other operators in generalized version of the query.

Accuracy for Aggregate Threshold Query. The following definition provides accuracy definitions for an aggregate threshold query.

Definition 6.1 (β-False Negative Rate)**.** *We say a mechanism $M : \mathcal{D} \to O$ satisfies β-false*

negative rate for an aggregate threshold query $Q^{\Lambda,f}_{\mathsf{g}(.)>C}$ *if for any database* $D \in \mathcal{D}$*, we have*

$$\forall \lambda_i \in \Lambda, P[\lambda_i \notin M(D) | \lambda_i \in Q^{\Lambda,f}_{\mathsf{g}(.)>C}(D)] \leq \beta \tag{6.2}$$

Definition 6.2 (α-False Positive Rate)**.** *We say a mechanism* $M : \mathcal{D} \rightarrow O$ *satisfies* α*-false positive rate for an aggregate threshold query* $Q^{\Lambda,f}_{\mathsf{g}(.)>C}$ *if for any database* $D \in \mathcal{D}$*, we have*

$$\forall \lambda_i \in \Lambda, P[\lambda_i \in M(D) | \lambda_i \notin Q^{\Lambda,f}_{\mathsf{g}(.)>C}(D)] \leq \alpha \tag{6.3}$$

In the following, we define complex decision support queries where instead of a having one clause that compares one aggregate value with its corresponding threshold , they consist of conjunction or disjunction of multiple clauses where each clause compares an aggregated value with it's corresponding threshold. We provide formal definition of Query Conjunction and Query Disjunction and their corresponding accuracy definitions as follows.

6.2.1 Query Conjunction

Consider a query Q which is conjunction of n aggregate threshold queries $Q_1, Q_2, \ldots, Q_n$ where we have same predicates for all queries with different filters, aggregates and thresholds i.e. $Q = Q_1 \cap Q_2 \cap, \ldots, \cap Q_n$. The query Q returns all the predicates that satisfy all the n queries.

Definition 6.3 (Query Conjunction)**.** *Given a set of* n *aggregate threshold queries* $Q_1, Q_2, \ldots, Q_n$ *where* $Q_j = Q^{\Lambda,f_j}_{\mathsf{g_j}(.)>C_j}$*, the conjunction of these Queries returns all predicates* λ_i *such that all the aggregates* $g_j(D^{f_j}_{\lambda_i})$ *are greater than their respective threshold* $c_{j,i}$ *for all* $j \in \{1, 2, \ldots, n\}$

and $i \in \{1, 2, \ldots, k\}$. *More formally,*

$$Q_1 \cap Q_2 \cap, \ldots, \cap Q_n(D) = \{\lambda_i \in \Lambda \mid \mathsf{g_1}(D_{\lambda_i}^{f_1}) > c_{1,i}\} \cap$$
$$\{\lambda_i \in \Lambda \mid \mathsf{g_2}(D_{\lambda_i}^{f_2}) > c_{2,i}\} \cap, \ldots, \cap \{\lambda_i \in \Lambda \mid \mathsf{g_n}(D_{\lambda_i}^{f_n}) > c_{n,i}\} \tag{6.4}$$

This is equivalent to using intersect operator between multiple group-by-having queries in SQL. Given a sales dataset with schema $SALES\_DATA($ *product_id*, *location*, *profit*, *timestamp*), the following is an example of conjunction of two aggregate threshold queries.

```
( SELECT product_id FROM SALES_DATA
Where location = 'USA'
GROUP BY product_id HAVING count(*) > c1)
INTERSECT
( SELECT product_id FROM SALES_DATA
Where location = 'Canada'
GROUP BY product_id HAVING count(*) > c2)
```

In this query, $product\_id = p_1, product\_id = p_2, ...product\_id = p_k$ are set of k predicates i.e. Λ. The query uses INTERSECT operator between two aggregate threshold queries Q_1 and Q_2. Q_1 returns all the predicates (product id) for which number of sales in 'USA' are greater than a certain threshold. Q_2 returns all the predicates (product id) for which number of sales in 'Canada' are greater than a certain threshold. The conjunction query returns all the predicates that satisfy both Q_1 and Q_2.

Query Conjunction Accuracy. Let query Q be conjunction of n such aggregate threshold queries i.e. $Q = Q_1 \cap Q_2 \cap, \ldots, \cap Q_n$. The accuracy in terms of false negative rate and false positive rate for Q can be derived as follows:

Definition 6.4 (β-False Negative Rate for query conjunction)**.** *We say a mechanism* $M : \mathcal{D} \to O$ *satisfies* β*-false negative rate for a conjunction query* $Q = Q_1 \cap Q_2 \cap, \ldots, Q_n$ *if for any database* $D \in \mathcal{D}$*, we have*

$$\forall \lambda_i \in \Lambda, P[\lambda_i \notin M(D) | \lambda_i \in Q_1(D) \cap Q_2(D) \cap, \ldots \cap Q_n(D)] \leq \beta \tag{6.5}$$

Definition 6.5 (α-False Positive Rate for query conjunction)**.** *We say a mechanism* $M : \mathcal{D} \to O$ *satisfies* α*-false positive rate for a conjunction query* $Q = Q_1 \cap Q_2 \cap, \ldots, Q_n$ *if for any database* $D \in \mathcal{D}$*, we have*

$$\forall \lambda_i \in \Lambda, P[\lambda_i \in M(D) | \lambda_i \notin Q_1(D) \cap Q_2(D) \cap, \ldots \cap Q_n(D)] \leq \alpha \tag{6.6}$$

6.2.2 Query Disjunction

Consider a query Q which is disjunction of n aggregate threshold queries $Q_1, Q_2, \ldots, Q_n$ where we have same predicates for all queries with different filters, aggregates and thresholds i.e. $Q = Q_1 \cup Q_2 \cup, \ldots, \cup Q_n$. The query Q returns all the predicates that satisfy at least one of the n queries.

Definition 6.6 (Query Disjunction)**.** *Given a set of* n *aggregate threshold queries* $Q_1, Q_2, \ldots, Q_n$ *where* $Q_j = Q^{\Lambda, f_j}_{\mathsf{g_j}(.) > C_j}$*, the disjunction of these queries returns all predicates* λ_i *such that at least one of the aggregates* $g_j(D^{f_j}_{\lambda_i})$ *is greater than their respective threshold* $c_{j,i}$ *for all* $j \in \{1, 2, \ldots, n\}$ *and* $i \in \{1, 2, \ldots, k\}$*. More formally,*

$$\begin{aligned} Q_1 \cup Q_2 \cup, \ldots, \cup Q_n(D) = \{\lambda_i \in \Lambda \mid \mathsf{g_1}(D^{f_1}_{\lambda_i}) > c_{1,i}\} \cup \\ \{\lambda_i \in \Lambda \mid \mathsf{g_2}(D^{f_2}_{\lambda_i}) > c_{2,i}\} \cup, \ldots, \cup \{\lambda_i \in \Lambda \mid \mathsf{g_n}(D^{f_n}_{\lambda_i}) > c_{n,i}\} \end{aligned} \tag{6.7}$$

This is equivalent to using union operator between multiple group-by-having queries in SQL. Given a sales dataset with schema $SALES\_DATA($ *product_id*, *location*, *profit*, *timestamp*), the following is an example of disjunction of two aggregate threshold queries.

```
( SELECT product_id FROM SALES_DATA
Where location = 'USA'
GROUP BY product_id HAVING count(*) > c1)
UNION
( SELECT product_id FROM SALES_DATA
Where location = 'Canada'
GROUP BY product_id HAVING count(*) > c2)
```

In this query, $product\_id = p_1, product\_id = p_2, ...product\_id = p_k$ are set of k predicates i.e. Λ. The query uses UNION operator between two aggregate threshold queries Q_1 and Q_2. Q_1 returns all the predicates (product id) for which number of sales in 'USA' are greater than a certain threshold. Q_2 returns all the predicates (product id) for which number of sales in 'Canada' are greater than a certain threshold. The disjunction query returns all the predicates that satisfy either Q_1 or Q_2 or both.

Query Disjunction Accuracy. Let query Q be conjunction of n such aggregate threshold queries i.e. $Q = Q_1 \cup Q_2 \cup, \ldots, \cup Q_n$. The accuracy in terms of false negative rate and false positive rate for Q can be derived as follows:

Definition 6.7 (β-False Negative Rate for query conjunction)**.** *We say a mechanism* $M : \mathcal{D} \rightarrow O$ *satisfies* β*-false negative rate for a disjunction query* $Q = Q_1 \cup Q_2 \cup, \ldots, Q_n$ *if for any database* $D \in \mathcal{D}$*, we have*

$$\forall \lambda_i \in \Lambda, P[\lambda_i \notin M(D) | \lambda_i \in Q_1(D) \cup Q_2(D) \cup, \ldots \cap Q_n(D)] \leq \beta \tag{6.8}$$

Definition 6.8 (α-False Positive Rate for query disjunction)**.** *We say a mechanism* $M : \mathcal{D} \rightarrow O$ *satisfies* α*-false positive rate for a disjunction query* $Q = Q_1 \cup Q_2 \cup, \ldots, Q_n$ *if for any database* $D \in \mathcal{D}$*, we have*

$$\forall \lambda_i \in \Lambda, P[\lambda_i \in M(D) | \lambda_i \notin Q_1(D) \cup Q_2(D) \cup, \ldots \cup Q_n(D)] \leq \alpha \tag{6.9}$$

We can easily generalize the above definitions to consider more general queries based on any logical expression with conjunction and disjunction by appropriately combining the intersections and unions. Further note that our query condition $g(.)\theta c$ does not need to consider negation as $\sim g(.)\theta c$ can always be written as $g(.)\theta' c$ where θ' is dual of θ. For example $\sim g(.) < c$ can be written as $g(.) > c$. In the following, we focus on conjunctive and disjunctive queries and devise accuracy aware privacy preserving algorithms and later generalize the full logical expression using conjunctive and disjunctive query algorithms.

6.3 Privacy and Accuracy Guarantees For Complex DS Queries

Chapter 5 demonstrated that DS tasks require guarantees on the output quality, especially, for false negatives that may result due to the addition of noise to aggregated statistics. Such false negatives may result in events of interest/anomalies not being detected. False positives are also not desirable. While one would desire effective bounds on both false negatives and positives, in DS applications, increased false negatives are far more debilitating (compared to false positives) since they effectively defeat the very purpose of decision support. Thus, in DS applications, we desire to have bounded guarantees on false negatives without significantly

increasing the number of false positives. [1]

DP mechanisms such as the Laplace mechanism add noise from zero-mean distribution to the aggregate and compare it with the threshold, which place equal weights on false positives and false negatives. This approach can fail to bound both errors together by setting the privacy budget too small (large noise); or have guarantees on both false positives and false negatives, but with a high privacy cost.

To bound the false negative rate without incurring additional privacy cost, we design a class of mechanisms that generalizes the thresholds in the query for each aggregate value. For example, for an aggregate threshold query where we are checking $X > c$ for an aggregate X, we generalize the query threshold to $X > c - u$. This type of generalization allows us to achieve trade-off between false negatives and false positives that helps us achieve β-false negative rate with a minimal privacy cost. This generalization parameter u and the accuracy parameter β are translated to privacy cost ϵ.

6.3.1 Problem Definition

Considering the accuracy and privacy requirements defined above for decision support queries, we formalize our problem definition as follows. Given complex decision support query Q on a dataset D, we want to develop a set of differentially private mechanisms that answer the query with β-false negative rate guarantee (Def. 6.7,6.7) and minimal privacy loss in terms of ϵ.

In this section, we first present mechanisms to answer complex decision support queries which consist of conjunction or disjunction of 2 aggregate threshold queries and their accuracy and privacy guarantees.

[1] If we ignored false positives and only considered false negatives, a trivial algorithm would be to simply ignore the query condition and return all the objects. This will meet the bounded requirement of false negatives and will have zero false negatives. But that also defeats the purpose of decision support applications.

M_1	M_2	$M_1 \cap M_2$	$M_1 \cup M_2$
TP	TP	TP	TP
TP	TN	TN	TP
TP	FP	FP	TP
TP	FN	FN	TP
TN	TP	TN	TP
TN	TN	TN	TN
TN	FP	TN	FP
TN	FN	TN	FN

M_1	M_2	$M_1 \cap M_2$	$M_1 \cup M_2$
FP	TP	FP	TP
FP	TN	TN	FP
FP	FP	FP	FP
FP	FN	TN	TP
FN	TP	FN	TP
FN	TN	TN	FN
FN	FP	TN	TP
FN	FN	FN	FN

Figure 6.1: The figure shows output of intersection and union of M_1 and $M2$ and highlights FP(in green) and FN(in red) cases where M_1 and M_2 are output of mechanism to answer Q_1 and Q_2.

6.3.2 Query Conjunction Mechanism

The following mechanism provides a way to answer conjunctive queries Q where $Q = Q_1 \cap Q_2$ based on mechanism to answer individual aggregate threshold queries.

Definition 6.9 (Query Conjunction Mechanism)**.** *Let mechanism $M_i : \mathcal{D} \to O_i$ satisfy α_i-false positive rate and β_i-false negative rate for aggregate threshold query Q_i. We can answer query Q which is a conjunction of 2 aggregate threshold queries Q_1 and Q_2 using mechanism M where $M(D) = M_1(D) \cap M_2(D)$.*

To understand the intuition of how we support bounded false negatives, lets consider a conjunctive query $Q = Q_1 \cap Q_2$. We answer queries Q_1 and Q_2 with differential private mechanism $M_1(D)$ and $M_2(D)$. Note that output of these mechanisms can be one of the following true positive (TP), true negative (TN), false positive (FP) and false negative (FN).

Given the above cases, we can ascertain the output of mechanism $M(D) = M_1(D) \cap M_2(D)$ as described in Figure 6.1 which gives us 3 cases when mechanism M will result in FNs.

In the following, we use this intuition to provide bounds on false negative rate for such a mechanism M based on FNR and FPR of each M_i in case of $n = 2$.

Theorem 6.1. *Given mechanism M where $M(D) = M_1(D) \cap M_2(D)$ and β_i and α_i are respective FNR and FPR rates for M_i, $\forall i = 1, 2$. FNR for M is bounded by β if $(1-\alpha_1)\beta_2 + (1-\alpha_2)\beta_1 + \beta_1\beta_2 < \beta$*

Proof. Given mechanism $M_i : \mathcal{D} \to O_i$ satisfy α_i-false positive rate and β_i-false negative rate for aggregate threshold query Q_i. We can derive the β-FNR for mechanism M where

$M(D) = M_1(D) \cap M_2(D)$ to answer conjunctive query Q in terms of α_i and β_i as follows.

$$\forall \lambda_i \in \Lambda,$$

$$FNR = P[\lambda_i \notin M_1(D) \cap M_2(D) | \lambda_i \in Q_1(D) \cap Q_2(D)]$$

$$\text{Let } A = P[\lambda_i \in M_1(D)], B = P[\lambda_i \in M_2(D)], C = P[\lambda_i \in Q_1(D)],$$

$$\text{and } D = P[\lambda_i \in Q_1(D)]$$

$$FNR = P[\overline{AB}|CD] = P[\overline{A} \cup \overline{B}|CD] = P[\overline{A}B \cup A\overline{B} \cup \overline{AB}|CD]$$

$$= \frac{P[(\overline{A}B \cup A\overline{B} \cup \overline{AB}) \cap CD]}{P[CD]}$$

$$= \frac{P[(\overline{A}B \cap CD) \cup (A\overline{B} \cap CD) \cup (\overline{AB} \cap CD)]}{P[CD]}$$

Mutually exclusive events

$$= \frac{P[\overline{A}B \cap CD] + P[A\overline{B} \cap CD] + P[\overline{AB} \cap CD]}{P[CD]}$$

Assuming A, C are independent of B, D

$$= \frac{P[\overline{A}C]P[BD] + P[AC]P[\overline{B}D] + P[\overline{A}C]P[\overline{B}D]}{P[C][D]}$$

$$= P[\overline{A}|C]P[B|D] + P[A|C]P[\overline{B}|D] + P[\overline{A}|C]P[\overline{B}|D]$$

$$= P[\lambda_i \notin M_1(D) | \lambda_i \in Q_1(D)]P[\lambda_i \in M_2(D) | \lambda_i \in Q_2(D)]$$

$$+P[\lambda_i \in M_1(D) | \lambda_i \in Q_1(D)]P[\lambda_i \notin M_2(D) | \lambda_i \in Q_2(D)]$$

$$+P[\lambda_i \notin M_1(D) | \lambda_i \in Q_1(D)]P[\lambda_i \notin M_2(D) | \lambda_i \in Q_2(D)]$$

$$= (1 - \alpha_1)\beta_2 + \beta_1(1 - \alpha_2) + \beta_1\beta_2 \tag{6.10}$$

6.3.3 Query Disjunction Mechanism

The following mechanism provides a way to answer disjunctive queries Q where $Q = Q_1 \cup Q_2$ based on mechanism to answer individual aggregate threshold queries.

Definition 6.10 (Query Disjunction Mechanism)**.** *Let mechanism $M_i : \mathcal{D} \to O_i$ satisfy α_i-*

false positive rate and β_i-false negative rate for aggregate threshold query Q_i. We can answer query Q which is a disjunction of 2 aggregate threshold queries Q_1 and Q_2 using mechanism M where $M(D) = M_1(D) \cup M_2(D)$.

To understand the intuition of how we support bounded false negatives, lets consider a disjunctive query $Q = Q_1 \cup Q_2$. We answer queries Q_1 and Q_2 with differential private mechanism $M_1(D)$ and $M_2(D)$. Note that output of these mechanisms can be one of the following true positive (TP), true negative (TN), false positive (FP) and false negative (FN). Given the above cases, we can ascertain the output of mechanism $M(D) = M_1(D) \cup M_2(D)$ as described in Figure 6.1 which gives us 3 cases when mechanism M will result in FNs.

In the following, we use this intuition to provide bounds on false negative rate for such a mechanism M based on FNR and FPR of each M_i in case of $n = 2$.

Theorem 6.2. *Given mechanism M where $M(D) = M_1(D) \cup M_2(D)$ and β_j and α_j are respective FNR and FPR rates for M_j, $\forall j = 1, 2$. FNR for mechanism M is bounded by β if $\frac{(1-\alpha_1)\beta_2 q_1' q_2 + \beta_1(1-\alpha_2) q_1 q_2' + \beta_1\beta_2 q_1 q_2}{q_1' q_2 + q_1 q_2' + q_1 q_2} < \beta$ where q_1 and q_2 are selectivity of the query i.e. $q_j = P[\lambda_i \in Q_j(D)]$ and $q_j' = 1 - q_j$.*

Proof. Given mechanism $M_j : \mathcal{D} \to O_j$ satisfy α_j-false positive rate and β_j-false negative rate for aggregate threshold query Q_j. We can derive the β-FNR for mechanism M where $M(D) = M_1(D) \cup M_2(D)$ to answer disjunctive query Q in terms of α_j and β_j as follows.

$$\forall \lambda_i \in \Lambda,$$

$$FNR = P[\lambda_i \notin M_1(D) \cup M_2(D) | \lambda_i \in Q_1(D) \cup Q_2(D)]$$

$$\text{Let } A = P[\lambda_i \in M_1(D)], B = P[\lambda_i \in M_2(D)], C = P[\lambda_i \in Q_1(D)],$$

$$\text{and } D = P[\lambda_i \in Q_1(D)]$$

$$FNR = P[\overline{A \cup B} | C \cup D] = P[\overline{A} \cap \overline{B} | C \cup D] = P[\overline{A} \cap \overline{B} | \overline{C}D \cap C\overline{D} \cap CD]$$

$$= \frac{P[(\overline{A} \cap \overline{B}) \cap (\overline{C}D \cap C\overline{D} \cap CD)]}{P[\overline{C}D \cap C\overline{D} \cap CD]}$$

$$= \frac{P[(\overline{A} \cap \overline{B} \cap \overline{C}D) \cup (\overline{A} \cap \overline{B} \cap C\overline{D}) \cup (\overline{A} \cap \overline{B} \cup CD)]}{P[\overline{C}D \cap C\overline{D} \cap CD]}$$

$$= \frac{P[\overline{A} \cap \overline{C}]P[\overline{B} \cap D] + P[\overline{A} \cap C]P[\overline{B} \cap \overline{D}] + P[\overline{A} \cap C]P[\overline{B} \cap D]}{P[\overline{C}D \cap C\overline{D} \cap CD]}$$

$$= \frac{P[\overline{A}|\overline{C}]P[\overline{B}|D]P[\overline{C}D] + P[\overline{A}|C]P[\overline{B}|\overline{D}]P[C\overline{D}] + P[\overline{A}|C]P[\overline{B}|D]P[CD]}{P[\overline{C}D \cap C\overline{D} \cap CD]}$$

$$= \frac{(1-\alpha_1)\beta_2 q_1' q_2 + \beta_1(1-\alpha_2) q_1 q_2' + \beta_1\beta_2 q_1 q_2}{q_1' q_2 + q_1 q_2' + q_1 q_2} \tag{6.11}$$

To provides β bound on FNR for mechanism M, we need to satisfy the following condition.

$$\frac{(1-\alpha_1)\beta_2 q_1' q_2 + \beta_1(1-\alpha_2) q_1 q_2' + \beta_1\beta_2 q_1 q_2}{q_1' q_2 + q_1 q_2' + q_1 q_2} < \beta$$

□

6.4 Accuracy Aware Privacy Preserving Algorithms For Complex DS Queries

In Section 6.3, we provided mechanism to answer conjunction/disjunction of two queries and their accuracy and privacy guarantees. In this section, we provide algorithms to answer these queries using the concept of threshold shift algorithm from Chapter 5 to achievβ-FNR guarantees. We describe how these algorithm can be generalized conjunction/disjunction of n variables. We add a discussion on more decision support queries that can be a combination of arbitrary number of conjunctions/disjunctions.

6.4.1 Algorithms for Query Conjunction

In Section 6.3, we presented mechanism M (Definition 6.9) to answer conjunction of two queries. To provides β bound on FNR for mechanism M, we need to satisfy the following condition.

$$(1-\alpha_1)\beta_2 + (1-\alpha_2)\beta_1 + \beta_1\beta_2 < \beta$$

□

The following algorithms use the concept of threshold shifting from Chapter 5 to answer conjunction query in a privacy preserving manner and satisfy the β-false negative guarantee. We use M_1 and M_2 to be threshold shift laplace mechanism (TSLM) from Chapter 5. We know for TSLM, False positive rate $\alpha_j > \beta_j$. We can simplify the condition to bound FNR

(from Theorem 6.1) as follows:

$$\beta_1 + \beta_2 - \beta_1\beta_2 < \beta$$

We can further simplify it by using upper bound on FNR.

$$\beta_1 + \beta_2 < \beta \tag{6.12}$$

In the following, we provide two algorithms for query conjunction. We first provide a naive threshold shift algorithm for query conjunction that achieves β-false negative guarantee and then provide optimized threshold shift algorithm that not only satisfies the β-false negative guarantee but also minimizes the privacy loss.

Naive threshold shift algorithm for Query Conjunction.

To satisfy the β-FNR accuracy requirement, we want to set β_1 and β_2 to be any value that satisfies $\beta_1 + \beta_2 < \beta$ (Equation 6.12). For naive algorithm, we simply choose $\beta_1 = \beta_2$ that satisfy the FNR constraint on query conjunction. The FNR constraint (Equation 6.12) can be rewritten as following:

$$\begin{aligned} \beta_1 + \beta_1 &\leq \beta \\ \beta_1 &\leq \beta/2 \end{aligned} \tag{6.13}$$

Algorithm 6 Query Conjunction based on Threshold Shift Laplace Mechanism.

procedure NAIVETSACONJUNCTION($Q_1, Q_2, D, u_1, u_2, \beta, \epsilon_{max}$)
    $\beta_1 = \beta/2$
    $\beta_2 = \beta/2$
    $\epsilon_j \leftarrow \frac{\ln(1/(2\beta_j))}{u_j}$ for j =1,2
    if $\sum_j \epsilon_j \leq \epsilon_{max}$ **then**
        $O_j \leftarrow \{\lambda_i \in \Lambda \mid \mathsf{g_j}(D_{\lambda_i}) + \eta_i > c_{i,j} - u_j, \eta_i \sim Lap(0, 1/\epsilon_j)\}$
        return $O_1 \cap O_2, \epsilon_1 + \epsilon_2$
    end if
    return 'Query Denied'
end procedure

We can set $\beta_1 = \beta_2 = \beta/2$ to achieve β-false negative rate.

We present the Naive Threshold Shift Algorithm for query conjunction in Algorithm box 6. This algorithm compares each noisy aggregate for Q_j, $g_j(D_{\lambda_i}) + \eta_i$ with a shifted threshold $c_{i,j} - u_i$, where u_i is a generalized parameter and noise η_i is based on a privacy budget $\epsilon_j = \frac{\ln(1/(2\beta_i))}{u_j}$ where $\beta_1 = \beta/2$ and $\beta_2 = \beta/2$. The set O_j consists the predicates that have noisy aggregates greater than the shifted thresholds for Q_j, i.e. $g_j(D_{\lambda_i}) + \eta_i > c_{i,j} - u_j$. The algorithm returns intersection of O_j. This algorithm satisfies over all $\beta - FNR$ guarantee and the privacy loss for this algorithm is $\epsilon = \epsilon_1 + \epsilon_2$.

Optimized Threshold-shift Algorithm For Query Conjunction.

This algorithm aims to minimize privacy loss and achieve a bounded false negative rate for all the predicates (Equation 6.12) using concept of threshold shift from Chapter 5. We want to choose β_1 and β_2 that satisfy the bound in Equation 6.12 while minimizing the privacy loss.

Overall privacy loss for conjunction of two query can be written as follows

$$
\begin{aligned}
\epsilon &= \epsilon_1 + \epsilon_2 \\
&= \frac{\ln(1/(2\beta_1))}{u_1} + \frac{\ln(1/(2\beta_2))}{u_2} \\
&\text{We want to minimize } \epsilon, \\
&= \ln(1/2\beta_1)^{\frac{1}{u_1}} + \ln(1/2\beta_2)^{\frac{1}{u_2}} \\
&= -\ln(2\beta_1)^{\frac{1}{u_1}} - \ln(2\beta_2)^{\frac{1}{u_2}} \\
&\approx (\beta_1)^{\frac{1}{u_1}}(\beta_2)^{\frac{1}{u_2}}
\end{aligned} \tag{6.14}
$$

To choose β_1 and β_2 that minimizes the privacy loss and satisfies the accuracy guarantees, we minimize ϵ i.e. minimize $(\beta_1)^{\frac{1}{u_1}}(\beta_2)^{\frac{1}{u_2}}$ subject to $FNR < \beta$ i.e. $\beta_1 + \beta_2 < \beta$. In the following, we solve this optimization problem.

Using Langrange Method,

$$\frac{d}{d\beta_2}((\beta_1)^{\frac{1}{u_1}}(\beta_2)^{\frac{1}{u_2}}) = \lambda\frac{d}{d\beta_2}(\beta_1+\beta_2)$$

$$(\beta_1)^{\frac{1}{u_1}}(\beta_2)^{\frac{1}{u_2}-1}/(u_2) = \lambda \tag{6.15}$$

Similarly,

$$(\beta_1)^{\frac{1}{u_1}-1}(\beta_2)^{\frac{1}{u_2}}/(u_1) = \lambda \tag{6.16}$$

From Equation 6.15 and 6.16

$$\beta_1 = \frac{u_2\beta_2}{u_1}$$

Substitute in optimization equality constraint

$$\beta_2 + (\frac{u_2\beta_2}{u_1}) = \beta$$

$$\beta_2 = u_1\beta/(u_1+u_2)$$

$$\text{Similarly, } \beta_1 = u_2\beta/(u_1+u_2) \tag{6.17}$$

These values for β_1, β_2 for min-privacy Threshold Shift Algorithm for query conjunction satisfy the overall β-FNR. Since these values are derived based on optimization of ϵ, Optimized Threshold Shift Algorithm for query conjunction minimizes the privacy loss.

We can convert Naive Threshold-shift Algorithm (Algorithm 6) to Optimize Threshold-shift Algorithm by modifying line 2 and 3 and setting β_1, β_2 according to Equation 6.17. The rest of the algorithm remains the same.

6.4.2 Algorithms for Query Disjunction

In Section 6.3, we presented mechanism M (Definition 6.10) to answer disjunction of two queries. To provides β bound on FNR for mechanism M, we need to satisfy the following condition.

$$\frac{(1-\alpha_1)\beta_2 q_1' q_2 + \beta_1(1-\alpha_2) q_1 q_2' + \beta_1\beta_2 q_1 q_2}{q_1' q_2 + q_1 q_2' + q_1 q_2} < \beta$$

The following algorithms use the concept of threshold shifting from Chapter 5 to answer disjunction query in a privacy preserving manner and satisfy the β-false negative guarantee. We use M_1 and M_2 to be threshold shift laplace mechanism (TSLM) from Chapter 5. We know for TSLM, False positive rate $\alpha_j > \beta_j$. We can simplify the condition to bound FNR (from Theorem 6.2) as follows:

$$\begin{aligned} FNR &\leq \frac{\beta_2 q_2 - \beta_2 q_1 q_2 + \beta_1 q_1 - \beta_1 q_1 q_2 - \beta_1\beta_2 q_1 - \beta_1\beta_2 q_2 + 3\beta_1\beta_2 q_1 q_2}{q_1 + q_2 - q_1 q_2} \\ &\leq \frac{\beta_2(q_2 - q_1 q_2) + \beta_1(q_1 - q_1 q_2) - \beta_1\beta_2(q_1 + q_2 - q_1 q_2) + 2\beta_1\beta_2 q_1 q_2}{q_1 + q_2 - q_1 q_2} \\ &\leq \beta_2 + \beta_1 - \beta_1\beta_2 + \frac{-\beta_1 q_2 - \beta_2 q_1 + 2\beta_1\beta_2 q_1 q_2}{q_1 + q_2 - q_1 q_2} \end{aligned} \tag{6.18}$$

We can further simplify it by using upper bound on FNR.

$$\beta_1 + \beta_2 < \beta \tag{6.19}$$

In the following, we provide two algorithms for query disjunction. We first provide a naive threshold shift algorithm for query disjunction that achieves β-false negative guarantee and then provide optimized threshold shift algorithm that not only satisfies the β-false negative guarantee but also minimizes the privacy loss.

Naive threshold shift algorithm for Query Disjunction.

To satisfy the β-FNR accuracy requirement, we want to set β_1 and β_2 to be any value that satisfies $\beta_1 + \beta_2 < \beta$ (Equation 6.19). For naive algorithm, we simply choose $\beta_1 = \beta_2$ that satisfy the FNR constraint on query disjunction. The FNR constraint (Equation 6.19) can be rewritten as following:

$$\begin{aligned} \beta_1 + \beta_1 &\leq \beta \\ \beta_1 &\leq \beta/2 \end{aligned} \tag{6.20}$$

We can set $\beta_1 = \beta_2 = \beta/2$ to achieve β-false negative rate.

We present the Naive Threshold Shift Algorithm for query disjunction in Algorithm box 7. This algorithm compares each noisy aggregate for Q_j, $g_j(D_{\lambda_i}) + \eta_i$ with a shifted threshold $c_{i,j} - u_i$, where u_i is a generalized parameter and noise η_i is based on a privacy budget

Algorithm 7 Query Disjunction based on Threshold Shift Laplace Mechanism.

1: **procedure** NAIVETSADISJUNCTION($Q_1, Q_2, D, u_1, u_2, \beta, \epsilon_{max}$)
2: $\beta_1 = \beta/2$
3: $\beta_2 = \beta/2$
4: $\epsilon_j \leftarrow \frac{\ln(1/(2\beta_j))}{u_j}$ for j =1,2
5: **if** $\sum_j \epsilon_j \leq \epsilon_{max}$ **then**
6: $O_j \leftarrow \{\lambda_i \in \Lambda \mid \mathrm{g_j}(D_{\lambda_i}) + \eta_i > c_{i,j} - u_j, \eta_i \sim Lap(0, 1/\epsilon_j)\}$
7: **return** $O_1 \cup O_2, \epsilon_1 + \epsilon_2$
8: **end if**
9: **return** 'Query Denied'
10: **end procedure**

$\epsilon_j = \frac{\ln(1/(2\beta_i))}{u_j}$ where $\beta_1 = \beta/2$ and $\beta_2 = \beta/2$. The set O_j consists the predicates that have noisy aggregates greater than the shifted thresholds for Q_j, i.e. $g_j(D_{\lambda_i}) + \eta_i > c_{i,j} - u_j$. The algorithm returns union of O_j. This algorithm satisfies over all $\beta - FNR$ guarantee and the privacy loss for this algorithm is $\epsilon = \epsilon_1 + \epsilon_2$.

Optimized Threshold-shift Algorithm For Query Disjunction.

This algorithm aims to minimize privacy loss and achieve a bounded false negative rate for all the predicates (Equation 6.19) using concept of threshold shift from Chapter 5. We want to choose β_1 and β_2 that satisfy the bound in Equation 6.18 while minimizing the privacy loss.

Notice that privacy loss for disjunction of two querys is the same as conjunction of two queries as $\epsilon = \epsilon_1 + \epsilon 2$ and the bound in Equation 6.18 to satisfy β-FNR is also the same as conjunction of two queries i.e. minimize $(\beta_1)^{\frac{1}{u_1}}(\beta_2)^{\frac{1}{u_2}}$ subject to $FNR < \beta$ i.e. $\beta_1 + \beta_2 < \beta$.

These values for β_1, β_2 for min-privacy Threshold Shift Algorithm for query disjunction satisfy the overall β-FNR. Since these values are derived based on optimization of ϵ, Optimized Threshold Shift Algorithm for query disjunction minimizes the privacy loss.

We can convert Naive Threshold-shift Algorithm (Algorithm 7) to Optimize Threshold-shift Algorithm for query conjunction by modifying line 2 and 3 and setting β_1, β_2 according to

Equation 6.17 i.e. $\beta_1 = u_2\beta/(u_1+u_2), \beta_2 = u_1\beta/(u_1+u_2)$. The rest of the algorithm remains the same.

6.4.3 Generalized Conjunction/Disjunction Query

In this section, we present generalized conjunction/disjunction queries. First, we generalize how to select β_j for each Q_j in case where the DS query consists of n aggregate threshold queries $Q_1, Q_2, ..., Q_n$. Generalization to conjunction/disjunction to n queries is described as follows.

Query conjunction/disjunction with n aggregate threshold queries. Consider conjunction of n aggregate threshold queries i.e. $Q = Q_1 \cap Q_2 \cap, ..., \cap Q_n$, where Q_j can be defined as $Q^{\Lambda,f_j}_{\mathsf{g_j}(.)>C_j}(D) = \{\lambda_i \in \Lambda \mid \mathsf{g_j}(D^{f_j}_{\lambda_i}) > c_{j_i}\}$. All Q_j have same $\Lambda = \lambda_i, \lambda_2, ..., \lambda_k$ and corresponding filter f_j, aggregate function $g_j(.)$ and threshold $C_j = c_{j_1}, ..., c_{j_k}$

Algorithm 6 can be modified to answer generalized version of conjunction query which is an intersection of n aggregate threshold queries as described above. To ensure β-FNR and minimize the privacy loss, we modify Algorithm 6 and set β_j as follows,

$$\beta_j = \frac{\beta \prod_{x=1}^{n,x\neq j}(u_x)}{\sum_{y=1}^{n}\prod_{x=1}^{n,x\neq y}(u_x)}, \forall j = \{1, 2, ..., n\} \tag{6.21}$$

This expression is derived from optimizing the more general solution to optimization problem of minimizing epsilon expression as decribed in Equation 6.14 i.e. $(\beta_1)^{\frac{1}{u_1}}(\beta_2)^{\frac{1}{u_2}}...(\beta_n)^{\frac{1}{u_n}}$. subject to having $FNR < \beta$ which translates to

$$\beta_1 + \beta_2 + ... + \beta_n < \beta \tag{6.22}$$

Similarly, we can modify Algorithm 7 to answer generalized version of disjunction query which is the union of n aggregate threshold queries i.e. $Q = Q_1 \cup Q_2 \cup, ..., \cup Q_n$. To ensure β-FNR guarantee and minimize privacy loss in terms of ϵ, we set β_j according to Equation 6.21 in Algorithm 7.

DS Query With Combination Of Conjunction And Disjunction.

We will illustrate by an example that algorithm for combination of conjunction/disjunction of queries is similar to either conjunction or disjunction of n queries.Consider an example query $Q = Q_1 \cup (Q_2 \cap Q_3)$. We can represent the query tree for query Q as presented in Figure 6.2 (a). In this query tree, we know how to answer sub-tree of this query as it just conjunction of two queries. Lets Q_{23} be the sub-tree where $Q_{23} = Q_2 \cap Q_3$ (as shown in Figure 6.2)(b). Lets say we can answer the conjunction of Q_2 and Q_3 with β_{23}-FNR guarantee with ϵ_{23} differential privacy. We need to satisfy the following bounds on $\beta_2 3$,

$$\beta_2 + \beta_3 < \beta_{23} \tag{6.23}$$

where β_2 and β_3 are respective FNR bound on individual queries Q_2 and Q_3. The privacy loss for such query Q_{23} would be $\epsilon_{23} = \epsilon_2 + \epsilon_3$.

Similarly, to answer query Q which is a disjunction of two queries $Q = Q_1 \cup Q_{23}$ with β-FNR guarantee, we need to satisfy the following bounds,

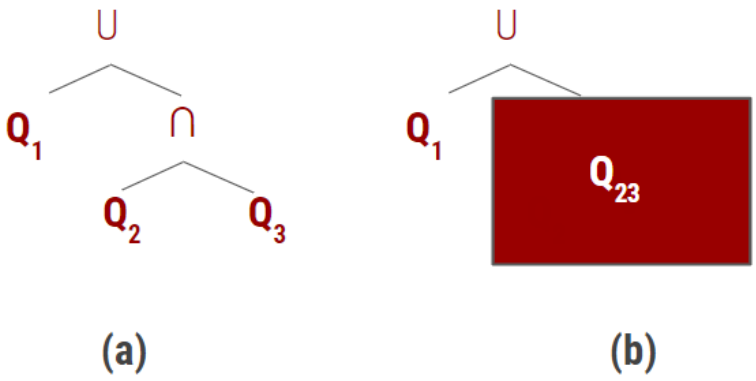


Figure 6.2: The figure shows query tree for (a) $Q = Q_1 \cup (Q_2 \cap Q_3)$, (b) $Q = Q_1 \cup Q_{23}$ where $Q_{23} = Q_2 \cap Q_3$

$$\beta_{23} + \beta_1 < \beta \tag{6.24}$$

where β_{23} and β_1 are FNR of individual queries Q_{23} and Q_1. The privacy loss for query Q would be $\epsilon = \epsilon_{23} + \epsilon_1$.

From equation 6.23 and 6.25, we can write the bounds for query $Q = Q_1 \cup (Q_2 \cap Q_3)$ as follows,

$$\beta_1 + \beta_2 + \beta_3 < \beta \tag{6.25}$$

where β_1,β_2 and β_3 are FNR guarantees of individual query Q_1, Q_2 and Q_3. The privacy loss $\epsilon = \epsilon_1 + \epsilon 2 + \epsilon_3$. Notice that this combination of conjunction and disjunction of query has same bounds that we need to satisfy for either conjunction or disjunction of 3 queries.

We can use the same optimization for combination of conjunction and disjunction of n queries as we presented for either conjunction or disjunction of n queries in this section. The only difference is that in combination of conjuction/disjunction, the order of conjunction matter. We can answer these type of queries by performing intersection and union of results of each individual queries in the order they appear in the original queries.

We can represent a combination of conjunction and disjunction of n queries in multiple ways. For example, $Q = Q_1 \cup (Q_2 \cap Q_3)$ can also be written as $Q = (Q_1 \cup Q_2) \cap (Q_1 \cup Q_3)$. We leave the selection of most optimized query tree that minimizes the privacy loss for a combination of conjunction and disjunction of n queries for future work.

6.5 Preliminary Experiments

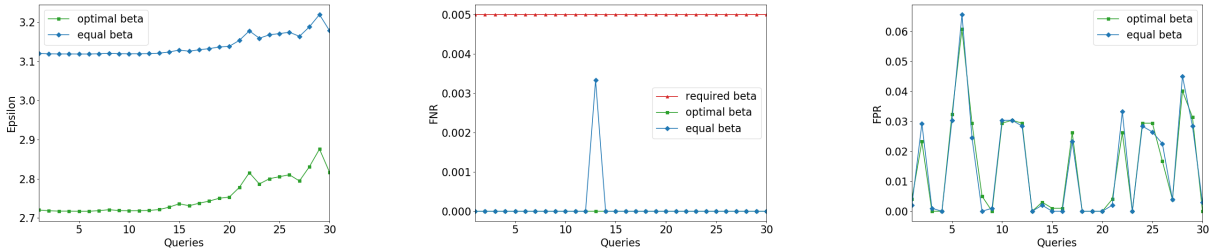
This section evaluates our Threshold-shift and naive Mechanisms for Query Conjunction and Query Disjunction as presented in Section 6.3.2 and 6.4.2 for various complex DS queries over real datasets. This is to show that all the algorithms effectively achieve their accuracy guarantees in terms of bounded false negative rates. Algorithm to minimize privacy loss i.e. Threshold-shift Mechanism for both query conjunction and disjunction performs better than naive mechanism in terms of privacy loss.

NYTaxi Dataset. This dataset records taxi trips in New York City in 2020 [1], consisting of 15.7 million records with 18 attributes, e.g. pick-up and drop-off locations, fare amount and timestamps. We group the pickup locations into 34 different regions and run queries to find out the regions that had anomalous pickup counts and or anomalous average fare amount each day in the month of March. We run 62 queries, 30 queries for (one per day) for conjunctive condition (pickup counts greater than certain threshold and average fare amount is also greater than threshold) and 30 queries (one per day) for disjunctive condition

(pickup counts greater than certain threshold or average fare amount is also greater than threshold). Total number of predicates for all queries are 34 (number of regions). For each condition(e.g. count¿t), we set threshold such that the selectivity (ratio of predicates satisfying the condition) of the query is 0.3. We set uncertain region u_1, u_2 for aggregates f_1, f_2 (number of trips, average fare amount) to be $0.05 * (max(g_j(.)) - min(g_j(.)))$. We use accuracy requirement on FNR $\beta = 0.005$ and $\epsilon_{max} = 4$.

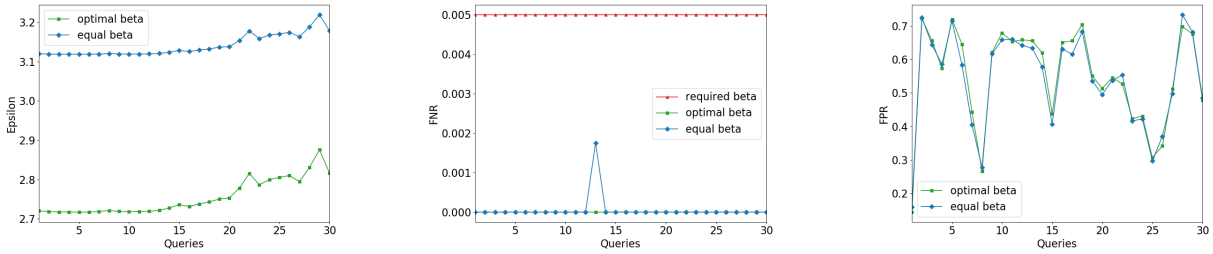
Experimental Evaluation. Figure 6.3 and 6.4 show results for conjunction and disjunction of queries on NYTaxi Dataset. The x-axis shows the query indexes (1 query for each day in the month of march 2020). We evaluate Threshold Shift Algorithm and Naive Algorithm for both query conjunction and disjunction. We run each algorithm for 100 iterations and present privacy loss and accuracy of each query.

Figure 6.3a and 6.4a shows privacy loss in terms of ϵ on y-axis. The privacy loss is minimized for Threshold Shift Algorithm for query conjunction and disjunction as compared to the naive method. The privacy loss is lower because Threshold Shift Algorithm selects β_1 and β_2 based on optimizing ϵ loss as compared to naive algorithm that chooses $\beta_1 = \beta_2 = \beta/2$. For accuracy analysis, we measured the number of false negatives n_{fn} and the number of false positives n_{fp} for each run of the algorithm. Then we report the averaged false negative rate (FNR) as n_{fn}/n_p and the averaged false positive rate (FPR) as n_{fp}/n_n over multiple runs, where n_p and n_n are the number of positives and the number of negatives respectively. The results are presented in Figure 6.3c,6.4c, 6.3b and 6.4b.In terms of accuracy, both naive and Threshold Shift Algorithm satisfy the β-FNR and have a similar FPR which is data dependent. The disjunction of queries has higher FPR as compared to conjunctive query.



(a) Epsilon (b) FNR (c) FPR

Figure 6.3: Privacy loss in terms of epsilon, accuracy in terms of False Negative Rate (FNR) and False Positive Rate (FPR) $\beta = 0.005$ for query conjunction.



(a) Epsilon (b) FNR (c) FPR

Figure 6.4: Privacy loss in terms of epsilon, accuracy in terms of False Negative Rate (FNR) and False Positive Rate (FPR) $\beta = 0.005$ for query disjunction.

6.6 Conclusion and Future Work

In this chapter, we presented accuracy-aware privacy preserving algorithms to answer complex decision support queries. We formally defined the complex queries and their accuracy requirement and presented differentially private algorithms that aim to minimize privacy loss while providing accuracy guarantees. Our preliminary results show that both naive and optimized algorithms achieve β-FNR guarantee and our optimize algorithm achieves lower privacy loss as compared to naive algorithm for different data distributions. We also present algorithms that optimize privacy loss for more general DS queries that consists of combination of conjunction and disjunction queries. In the future work, we would like to explore using progressive and data dependent algorithm from MIDE to answer conjunction/disjunction of n queries to optimize privacy loss and perform an extensive experimental study to evaluate our algorithms. We would also like to explore different version of query tree to explore the most optimized representation that minimizes privacy loss.

Chapter 7

Conclusion and Future Work

In this thesis, we formally define decision support queries and their accuracy requirement and present privacy preserving algorithms for decision support that satisfy the accuracy requirements of the decision support application while trying to provide maximum privacy using differential privacy. Privacy in the context of decision support query is a complex problem and this work is a precursor for a lot of future directions. In this work, we limit the scope to binary classifiers using simple/complex DS queries. One possible future direction for this work is to consider more general class classifiers as generalizing the classifier trade-off between false positive/false negatives applies to a broad class of classifiers. In the context of complex DS queries, using progressive and data dependent algorithm from MIDE to answer conjunction/disjunction of n queries to optimize privacy loss could be another future direction. Similarly, different version of query tree can also be used to explore the most optimized representation that minimizes privacy loss. Another possible future direction is to generalize minimally invasive architecture to broader class of SQL queries (e.g. queries with overlapping predicates). Other types of advance DP mechanism such as hierarchical mechanism [33] and exponential mechanism [13] can also be used to answer such queries. It would be interesting to generalize the β-FNR guarantee and accounting for privacy loss in

terms of predicate-wise DP framework. Fairness in the context of Predicate-wise DP is an interesting problem to solve as entities end with different privacy loss depending on the data distribution. Another future direction to extend this work is to consider decision support applications/queries in the context of streaming dataset.

Bibliography

[1] Tlc trip record data. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page, 2020. Accessed: 2021-12-31.

[2] A. R. Beresford and F. Stajano. Mix zones: user privacy in location-aware services. In *2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops*, pages 127–131, 2004.

[3] T.-H. H. Chan, E. Shi, and D. Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3), 2011.

[4] P. Chatterjee, L. J. Cymberknop, and R. L. Armentano. Iot-based decision support system for intelligent healthcare — applied to cardiovascular diseases. In *2017 7th International Conference on Communication Systems and Network Technologies (CSNT)*, pages 362–366, 2017.

[5] Y. Chen, A. Machanavajjhala, M. Hay, and G. Miklau. PeGaSus: Data-adaptive differentially private stream processing. In *ACM Conf. on Computer and Communications Security (CCS)*, pages 1375–1388, 2017.

[6] Y. Chen, A. Machanavajjhala, M. Hay, and G. Miklau. Pegasus: Data-adaptive differentially private stream processing. In *2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 1375–1388, 2017.

[7] A. Corna, L. Fontana, A. A. Nacci, and D. Sciuto. Occupancy detection via ibeacon on android devices for smart building management. In *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 629–632, 2015.

[8] D. E. Denning, P. J. Denning, and M. D. Schwartz. The tracker: A threat to statistical database security. *ACM Trans. Database Syst.*, 4(1):76–96, mar 1979.

[9] S. Depatla, A. Muralidharan, and Y. Mostofi. Occupancy estimation using only wifi power measurements. *IEEE Journal on Selected Areas in Communications*, 33(7):1381–1393, 2015.

[10] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 202–210, 2003.

[11] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *3rd Conference on Theory of Cryptography*, TCC'06, page 265–284, 2006.

[12] C. Dwork, F. McSherry, and K. Talwar. The price of privacy and the limits of lp decoding. In *Proceedings of the thirty-ninth annual ACM Symposium on Theory of Computing*, pages 85–94, 2007.

[13] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, Aug. 2014.

[14] C. Dwork and S. Yekhanin. New efficient attacks on statistical disclosure control mechanisms. In *Annual International Cryptology Conference*, pages 469–480. Springer, 2008.

[15] U. Erlingsson et al. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *ACM SIGSAC*, 2014.

[16] C. Ge et al. Apex: Accuracy-aware differentially private data exploration. SIGMOD, 2019.

[17] S. Ghayyur, Y. Chen, R. Yus, A. Machanavajjhala, M. Hay, G. Miklau, and S. Mehrotra. Iot-detective: Analyzing iot data under differential privacy. In *Proceedings of the 2018 International Conference on Management of Data, SIGMOD Conference 2018, Houston, TX, USA, June 10-15, 2018*, pages 1725–1728, 2018.

[18] S. Ghayyur et al. Iot-detective: Analyzing iot data under differential privacy. SIGMOD '18, pages 1725–1728, New York, NY, USA, 2018. ACM.

[19] M. Hardt, K. Ligett, and F. McSherry. A simple and practical algorithm for differentially private data release. *ArXiv*, abs/1012.4763, 2012.

[20] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially-private histograms through consistency. *Proceedings of the VLDB Endowment*, 3, 04 2009.

[21] X. He, A. Machanavajjhala, and B. Ding. Blowfish privacy: Tuning privacy-utility tradeoffs using policies. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, page 1447–1458, New York, NY, USA, 2014. Association for Computing Machinery.

[22] N. T. N. Hien and P. Haddawy. A decision support system for evaluating international student applications. In *2007 37th Annual Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports*, pages F2A–1–F2A–6, 2007.

[23] N. Johnson et al. Towards practical differential privacy for sql queries. *Proc. VLDB Endow.*, 11(5):526–539, Jan. 2018.

[24] Z. Jorgensen, T. Yu, and G. Cormode. Conservative or liberal? personalized differential privacy. In *2015 IEEE 31st International Conference on Data Engineering*, pages 1023–1034, 2015.

[25] M. Katsomallos, K. Tzompanaki, and D. Kotzinos. Privacy, space and time: a survey on privacy-preserving continuous data publishing. *Journal of Spatial Information Science*, 2019(19):57–103, 2019.

[26] D. Kifer and A. Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Trans. Database Syst.*, 39(1), Jan. 2014.

[27] W. Kleiminger, F. Mattern, and S. Santini. Predicting household occupancy for smart heating control: A comparative performance analysis of state-of-the-art approaches. *Energy and Buildings*, 85:493 – 505, 2014.

[28] C. D. Korkas, S. Baldi, I. Michailidis, and E. B. Kosmatopoulos. Occupancy-based demand response and thermal comfort optimization in microgrids with renewable energy sources and energy storage. *Applied Energy*, 163:93 – 104, 2016.

[29] I. Kotsogiannis, S. Doudalis, S. Haney, A. Machanavajjhala, and S. Mehrotra. One-sided differential privacy. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 493–504, 2020.

[30] F. Koufogiannis et al. Gradual release of sensitive data under differential privacy. *CoRR*, abs/1504.00429, 2015.

[31] J. Lee and C. Clifton. How much is enough? choosing ϵ for differential privacy. In X. Lai, J. Zhou, and H. Li, editors, *Information Security, 14th International Conference, ISC 2011, Xi'an, China, October 26-29, 2011. Proceedings*, volume 7001 of *Lecture Notes in Computer Science*, pages 325–340. Springer, 2011.

[32] P. Lee, E.-J. Shin, V. Guralnik, S. Mehrotra, N. Venkatasubramanian, and K. T. Smith. Exploring privacy breaches and mitigation strategies of occupancy sensors in smart buildings. In *Proceedings of the 1st ACM International Workshop on Technology Enablers and Innovative Applications for Smart Cities and Communities*, TESCA'19, page 18–21, New York, NY, USA, 2019. Association for Computing Machinery.

[33] C. Li, M. Hay, G. Miklau, and Y. Wang. A data-and workload-aware algorithm for range queries under differential privacy. *Proceedings of the VLDB Endowment*, 7(5):341–352, 2014.

[34] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing linear counting queries under differential privacy. In *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '10, page 123–134, New York, NY, USA, 2010. Association for Computing Machinery.

[35] F. Li, J. Sun, S. Papadimitriou, G. A. Mihaila, and I. Stanoi. Hiding in the crowd: Privacy preservation on evolving streams through correlation tracking. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 686–695, 2007.

[36] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115, 2007.

[37] K. Ligett, S. Neel, A. Roth, B. Waggoner, and Z. Wu. Accuracy first: Selecting a differential privacy level for accuracy-constrained erm. *Journal of Privacy and Confidentiality*, 9, 05 2017.

[38] E. Lobo-Vesga, A. Russo, and M. Gaboardi. A programming framework for differential privacy with accuracy concentration bounds. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 411–428, Los Alamitos, CA, USA, may 2020. IEEE Computer Society.

[39] A. Machanavajjhala et al. Privacy: Theory meets practice on the map. In *2008 IEEE 24th International Conference on Data Engineering*, pages 277–286, April 2008.

[40] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. L-diversity: privacy beyond k-anonymity. In *22nd International Conference on Data Engineering (ICDE'06)*, pages 24–24, 2006.

[41] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, E. C. Rye, and D. Brown. A study of MAC address randomization in mobile devices and when it fails. *PoPETs*, 2017(4):365–383, 2017.

[42] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103, 2007.

[43] F. D. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. SIGMOD '09, page 19–30, New York, NY, USA, 2009. Association for Computing Machinery.

[44] S. Mehrotra et al. Tippers: A privacy cognizant iot environment. In *2016 IEEE PerCom Workshops*, March 2016.

[45] S. Mehrotra, A. Kobsa, N. Venkatasubramanian, and S. R. Rajagopalan. TIPPERS: A privacy cognizant iot environment. In *IEEE Int. Conf. on Pervasive Computing and Communication Workshops (PerCom Workshops)*, pages 1–6, 2016.

[46] J. Mohammed Dahr, A. Khalaf, I. Najm, and M. Ahmed. Implementing sales decision support system using data mart based on olap, kpi, and data mining approaches. *Journal of Engineering Science and Technology*, 17:0275, 02 2022.

[47] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler. Gupt: privacy preserving data analysis made easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 349–360, 2012.

[48] Y.-A. Montjoye, C. Hidalgo, M. Verleysen, and V. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3:1376, 03 2013.

[49] A. Narayanan and V. Shmatikov. Robust de-anonymization of large datasets (how to break anonymity of the netflix prize dataset). 11 2006.

[50] P. M. Pardalos and J. B. Rosen. Methods for global concave minimization: A bibliographic survey. *Siam Review*, 28(3):367–379, 1986.

[51] R. A. Popa, A. J. Blumberg, H. Balakrishnan, and F. H. Li. Privacy and accountability for location-based aggregate statistics. In *18th ACM Conference on Computer and Communications Security*, CCS '11, page 653–666, 2011.

[52] D. L. Quoc, M. Beck, P. Bhatotia, R. Chen, C. Fetzer, and T. Strufe. Privapprox: Privacy-preserving stream analytics. In *2017 USENIX Conference on Usenix Annual Technical Conference*, USENIX ATC '17, page 659–672, 2017.

[53] R. Shirsath, N. Khadke, D. More, P. Patil, and H. Patil. Agriculture decision support system using data mining. In *2017 International Conference on Intelligent Computing and Control (I2C2)*, pages 1–5, 2017.

[54] L. Sweeney. K-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, 2002.

[55] R. Tomastik, Y. Lin, and A. Banaszuk. Video-based estimation of building occupancy during emergency egress. In *2008 American Control Conference*, pages 894–901, 2008.

[56] C. Tran, F. Fioretto, P. Van Hentenryck, and Z. Yao. Decision making with differential privacy under a fairness lens. 05 2021.

[57] A. Ungar, M. Rafanelli, I. Iacomelli, M. A. Brunetti, A. Ceccofiglio, F. Tesi, and N. Marchionni. Fall prevention in the elderly. *Clinical Cases in mineral and bone metabolism*, 10(2):91, 2013.

[58] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 2020.

[59] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[60] R. C. wing Wong, A. W.-C. Fu, K. Wang, and J. Pei. Minimality attack in privacy preserving data publishing. In *VLDB*, 2007.

[61] J. Zhang, X. Xiao, and X. Xie. Privtree: A differentially private algorithm for hierarchical decompositions. In *Proceedings of the 2016 International Conference on Management of Data*, SIGMOD '16, page 155–170, New York, NY, USA, 2016. Association for Computing Machinery.

[62] H. Zou, Y. Zhou, J. Yang, W. Gu, L. Xie, and C. Spanos. Freedetector: Device-free occupancy detection with commodity wifi. In *2017 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops)*, pages 1–5, 2017.

[63] I. A. Zualkernan, F. A. Aloul, V. Sakkia, H. A. Noman, S. Sowdagar, and O. A. Hammadi. An iot-based emergency evacuation system. In *2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)*, pages 62–66, 2019.

Appendix A

MIDE Appendix

A.1 Proofs

A.1.1 Properties of PWDP

Proof for Lemma 5.1. Given a Θ-Predicate-wise DP mechanism M with output o, where $\Theta = \{(\lambda_1, \epsilon_1), (\lambda_2, \epsilon_2), ..., (\lambda_k, \epsilon_k)\}$, each adversarial posterior guess $\hat{p}_i \propto \sum_{t_{\lambda_i}} \frac{P[x \in D \wedge x = t_{\lambda_i} | o]}{P[x \in D | o]}$ is bounded:

$$\frac{e^{-\epsilon_i}}{\sum_i e^{\epsilon_i}} \leq \hat{p}_i \leq \frac{e^{\epsilon_i}}{\sum_i e^{-\epsilon_i}}, \tag{A.1}$$

when priors $p_i \propto \sum_{t_{\lambda_i}} \frac{P[x \in D \wedge x = t_\lambda]}{P[x \in D]}$ are the same for $i \in [1, k]$.

Proof.

Let x be any tuple and

$r_i = P[x \in D \wedge x = t_{\lambda_i}]$ and $r_0 = P[x \notin D] = 1 - \sum_{i=1}^{k} r_i$

$q_i = r_i / r_0$,Assuming uniform priors: $\forall i,\ q_i = q$

$p_i = P[x \in D \wedge x = t_{\lambda_i} \mid o]$ and $p_0 = P[x \notin D | o]$

$\sum_{i=1}^{k} p_i + p_0 = 1$

Adversary's posterior guess: $\hat{p_i} = p_i / (1 - p_0)$

From the definition of Predicate-wise differential privacy Equation A.2:

$e^{-\epsilon_i} q \leq p_i / p_0 \leq e^{\epsilon_i} q$

$$\sum_j e^{-\epsilon_j} q p_0 + p_0 \leq 1 \leq \sum_j e^{\epsilon_j} q p_0 + p_0$$
$$\frac{1}{1 + \sum_j q e^{\epsilon_j}} \leq p_0 \leq \frac{1}{1 + \sum_j q e^{-\epsilon_j}}$$
$$\frac{1 + \sum_j q e^{\epsilon_j}}{\sum_j q e^{\epsilon_j}} \leq \frac{1}{1 - p_0} \leq \frac{1 + \sum_j q e^{-\epsilon_j}}{\sum_j q e^{-\epsilon_j}}$$
$$\frac{1}{\sum_j q e^{\epsilon_j}} \leq \frac{p_0}{1 - p_0} \leq \frac{1}{\sum_j q e^{-\epsilon_j}}$$
$$\frac{q e^{-\epsilon_i} p_0}{1 - p_0} \leq \frac{p_i}{1 - p_0} \leq \frac{q e^{\epsilon_i} p_0}{1 - p_0}$$
$$\frac{e^{-\epsilon_i} q}{\sum_j q e^{\epsilon_j}} \leq \hat{p_i} \leq \frac{e^{\epsilon_i} q}{\sum_j q e^{-\epsilon_j}}$$
$$\frac{e^{-\epsilon_i}}{\sum_j e^{\epsilon_j}} \leq \hat{p_i} \leq \frac{e^{\epsilon_i}}{\sum_j e^{-\epsilon_j}}$$

□

Bounds on adversarial posterior guess in case of non-uniform priors. The following lemma provides bounds on adversay's posterior guess in case of non uniform priors. We

update the constraints in the optimization of min entropy using these bounds to calculate the min-entropy metric.

Lemma A.1. *Given a Θ-Predicate-wise DP mechanism M with output o, where $\Theta = \{(\lambda_1, \epsilon_1), (\lambda_2, \epsilon_2), ..., (\lambda_k, \epsilon_k)\}$, and adversarial prior $q_i \propto \sum_{t_{\lambda_i}} \frac{P[x \in D \wedge x = t_{\lambda_i}]}{P[x \in D]}$, each adversarial posterior guess $\hat{p}_i \propto \sum_{t_{\lambda_i}} \frac{P[x \in D \wedge x = t_{\lambda_i} | o]}{P[x \in D | o]}$ is bounded:*

$$\frac{e^{-\epsilon_i} q_i}{\sum_j q_j e^{\epsilon_j}} \leq \hat{p}_i \leq \frac{e^{\epsilon_i} q_i}{\sum_j q_j e^{-\epsilon_j}}, \tag{A.2}$$

Proof.

Let x be any tuple and

$r_i = P[x \in D \wedge x = t_{\lambda_i}]$ and $r_0 = P[x \notin D] = 1 - \sum_{i=1}^{k} r_i$

$q_i = r_i / r_0$

$p_i = P[x \in D \wedge x = t_{\lambda_i} \mid o]$ and $p_0 = P[x \notin D | o]$

$\sum_{i=1}^{k} p_i + p_0 = 1$

Adversary's posterior guess: $\hat{p}_i = p_i/(1 - p_0)$

From the definition of Predicate-wise DP Equation A.2 :

$e^{-\epsilon_i} q_i \leq p_i/p_0 \leq e^{\epsilon_i} q_i$

$\sum_j e^{-\epsilon_j} q_j p_0 + p_0 \leq 1 \leq \sum_j e^{\epsilon_j} q_j p_0 + p_0$

$$\frac{1}{1 + \sum_j q_j e^{\epsilon_j}} \leq p_0 \leq \frac{1}{1 + \sum_j q_j e^{-\epsilon_j}}$$

$$\frac{1 + \sum_j q_j e^{\epsilon_j}}{\sum_j q_j e^{\epsilon_j}} \leq \frac{1}{1 - p_0} \leq \frac{1 + \sum_j q_j e^{-\epsilon_j}}{\sum_j q_j e^{-\epsilon_j}}$$

$$\frac{1}{\sum_j q_j e^{\epsilon_j}} \leq \frac{p_0}{1 - p_0} \leq \frac{1}{\sum_j q_j e^{-\epsilon_j}}$$

$$\frac{q_i e^{-\epsilon_i} p_0}{1 - p_0} \leq \frac{p_i}{1 - p_0} \leq \frac{q_i e^{\epsilon_i} p_0}{1 - p_0}$$

$$\frac{e^{-\epsilon_i} q_i}{\sum_j q_j e^{\epsilon_j}} \leq \hat{p}_i \leq \frac{e^{\epsilon_i} q_i}{\sum_j q_j e^{-\epsilon_j}}$$

□

Please note that our scheme provides a bounded epsilon post-DP guarantees, just like any DP-based mechanism. The assumption of uniform prior does not influence those guarantees. However, the uniform prior assumption does influence the min-entropy metric and hence

also the data dependent algorithm which tries to maximize the min-entropy. In Algorithm 4, we take priors as input, and hence instead of computing the posterior bounds based on uniform prior (line 7 Algorithm 4), we can compute the bounds based on the generalized lemma (Lemma 13) that establishes generalized priors without changing the algorithm.

Proof for Theorem 5.1. Given $\Theta = \{(\lambda_1, \epsilon_1), (\lambda_2, \epsilon_2), ..., (\lambda_k, \epsilon_k)\}$, a set of mutually exclusive predicates and their corresponding privacy budgets, running ϵ_i-DP mechanism M_i over D_{λ_i} in parallel for $i = 1, \ldots, k$, achieves Θ-predicate-wise DP.

Proof. Consider the differing tuple x in D and D' is in D_{λ_j}

$$
\begin{aligned}
&P[M(D) \in O] = \Pi_{i=1}^{k} P[M_i(D_{\lambda_i}) \in O_i | O_1, ..., O_{i-1}] \\
&\leq \Pi_{i!=j}^{k} P[M_i(D_{\lambda_i}) \in O_i | O_1, ..., O_{i-1}] e^{\epsilon_j} P[M_j(D'_{\lambda_j}) \in O_j | O_1, ..., O_{j-1}] \\
&\leq \Pi_{i!=j}^{k} P[M_i(D'_{\lambda_i}) \in O_i | O_1, ..., O_{i-1}] e^{\epsilon_j} P[M_j(D'_{\lambda_j}) \in O_j | O_1, ..., O_{j-1}] \\
&\leq e^{\epsilon_j} P[M(D') \in O]
\end{aligned}
$$

□

Proof for Theorem 5.3. A PWDP mechanism $\mathcal{M}$ with $\Theta = \{(\lambda_1, \epsilon_1), ..., (\lambda_k, \epsilon_k)\}$ satisfies ϵ-DP with $\epsilon = \max_i \epsilon_i$. A mechanism $\mathcal{M}$ with an ex-post PWDP loss $\mathcal{E}(o)$ has an $\epsilon(o)$-ex-post DP with $\epsilon(o) = \max_i \mathcal{E}_i(o)$.

Proof. A PWDP mechanism with $\Theta = \{(\lambda_1, \epsilon_1), ..., (\lambda_k, \epsilon_k)\}$ is equivalent to running ϵ_i-DP mechanism M_i over D_{λ_i} in parallel for $i = 1, 2, ..., k$ as running ϵ_i-DP mechanism M_i over D_{λ_i} in parallel for $i = 1, 2, ..., k$, achieves Θ-PWDP (Theorem 5.1). In PWDP, D_{λ_i} are mutually exclusive, hence by parallel composition property of DP (Definition 2.2), a PWDP mechanism $\mathcal{M}$ satisfies ϵ-DP with $\epsilon = \max_i \epsilon_i$. □

A.1.2 Guarantees of MIDE Algorithms

Proof for Theorem 5.4. Algorithm 1 satisfies ϵ_{max}-DP and β-false negative rate. If the query is not denied, its ex-post DP cost is $\epsilon = \frac{\ln(1/(2\beta))}{\alpha}$..

Proof. The algorithm satisfies ϵ_{max}-DP according to Definition 2.4 as it adds noise from Laplace Distribution with mean of 0 and standard deviation of $1/\epsilon$ where $\epsilon \leq \epsilon_{max}$ as checked in line 5.

In the following, we show that the false negative rate is bounded by β. For $\lambda_i \in \Lambda$

$$
\begin{aligned}
& P[\lambda_i \notin O | \lambda_i \in Q^{\Lambda}_{\mathsf{g}(.)>C}(D)] \\
= & P[\mathsf{g}(D_{\lambda_i}) + \eta_i \leq c_i - \alpha | \mathsf{g}(D_{\lambda_i}) > c_i] \\
\leq & P[\eta_i < -\alpha] \leq \frac{e^{-\ln(\frac{1}{2\beta})}}{2} \leq \beta
\end{aligned}
$$

Since the probability that the noise η_i is less than $(-\alpha)$ is bounded by β, shifting the threshold from c_i to $c_i - \alpha$ ensures that the predicates for which $g(D_{\lambda_i}) > c_i$ are eliminated after the noise addition with a high probability of $(1 - \beta)$. Hence, it guarantees that the probability of a predicate to be false negative is bounded by β. □

Proof for Theorem 5.5. Algorithm 2 satisfies ϵ_{max}-DP and β-false negative rate. If the query is not denied, its ex-post DP cost is less than $\epsilon_m = \frac{\ln(1/(2\beta/m)}{\alpha}$.

Proof. Algorithm 2 uses m iterations with $\epsilon_1, ...\epsilon_m$ privacy budget for each iteration. Any predicate $\lambda_i \in \Lambda$ ends up using one of the ϵ_j Let $\eta_1, \eta_2, \cdots, \eta_m$ be the noise values drawn from a Laplace distribution with the following variances: $\frac{1}{\epsilon_1}$, $\frac{1}{\epsilon_2}$, $\cdots$, $\frac{1}{\epsilon_m}$.

$$
\begin{aligned}
&P[\lambda_i \in False\ \ Negative] = \\
&P[g(D_{\lambda_i}) + \eta_1 \leq (c - \alpha_1) | g(D_{\lambda_i}) > c] + \\
&P[g(D_{\lambda_i}) + \eta_1 > (c - \alpha_1) | g(D_{\lambda_i}) > c] \cdot P[g(D_{\lambda_i}) + \eta_2 < c - \alpha_2 | g(D_{\lambda_i}) > c] \\
&+ \cdots + \prod_{j=1}^{m-1} P[g(D_{\lambda_i}) + \eta_j > (c - \alpha_j) | g(D_{\lambda_i}) > c] \\
&\cdot P[g(D_{\lambda_i}) + \eta_m < c - \alpha_m | g(D_{\lambda_i}) > c]
\end{aligned}
$$

In this equation $P[g(D_{\lambda_i}) + \eta_1 \leq (c - \alpha_1) | g(D_{\lambda_i}) > c] \leq \beta_1$ as the noise η_1 is drawn from Laplace distribution with parameter $\frac{1}{\epsilon_1}$ where $\epsilon_1 = \frac{\ln(\frac{1}{2\beta_1})}{\alpha_1}$. Similarly, $P[g(D_{\lambda_i}) + \eta_2 < c - \alpha_2 | g(D_{\lambda_i}) > c] \leq \beta_2$. The coefficient of the second term, *i.e.*, $P[g(D_{\lambda_i}) + \eta_1 > (c - \alpha_1) | g(D_{\lambda_i}) > c] > (1 - \beta_1)$. As the maximum value of $(1 - \beta_1)$ is 1, Equation **??** can be re-written as follows:

$$
\begin{aligned}
&\beta_1 + (1 - \beta_1)\beta_2 + (1 - \beta_1)(1 - \beta_2)\beta_3 + ... + \prod_{j=1}^{m-1}(1 - \beta_j)\beta_m \\
&\leq \beta_1 + \beta_2 + ... + \beta_m
\end{aligned}
\tag{A.3}
$$

As we use $\beta_j = \beta/m$ in Algorithm 2, $P[\lambda_i \in FalseNegative] \leq \sum_{j=1}^{m} \beta_j \leq \beta$ and hence it saitisfies the β-False Negative Rate accuracy requirement.

The algorithm satisfies ϵ_{max}-differential privacy according to Definition 2.4 as it adds noise from Laplace Distribution with mean of 0 and standard deviation of $1/\epsilon_j$ in each step j (Line 11) where $\epsilon_j \leq \epsilon_m$ and $\epsilon_m \leq \epsilon_{max}$ as checked in Line 4. By the definition of ex-post DP loss, the algorithm's ex-post DP cost is at most ϵ_m as the algorithm does not use $\epsilon > \epsilon_m$ for any iteration. □

Proof for Theorem 5.6. Algorithm 3 satisfies ϵ_{max}-DP and β-false negative rate. If the query is not denied, its ex-post DP cost is $max(E)$.

Proof. Similar to Algorithm 2, this algorithm satisfies β-false negative guarantees as $\sum_{j=1}^{m} \beta_j \leq \beta$ where β_j is β used in iteration j.

The algorithm satisfies ϵ_{max}-differential privacy according to Definition 2.4. In each step j, it adds noise from Laplace Distribution with mean of 0 and standard deviation of $1/\epsilon_j$ (Line 14) where $\epsilon_j \leq \epsilon_m \leq \epsilon_{max}$ (Line 4). Hence the algorithm satisfies ϵ_{max} DP. By the definition of ex-post DP loss, the algorithm's ex-post DP cost is at most ϵ_m as the algorithm does not use $\epsilon > \epsilon_m$ for any iteration. Algorithm 3 satisfies $\Theta = E$- PWDP, hence the ex-post DP cost is $max(E)$ as described in Theorem 5.3. □

Expected Number of Undecided Predicates. To calculate the expected number of undecided predicates in iteration j, we estimate whether a predicate λ_i is still undecided in j-th iteration (*i.e.*, to be reconsidered in j+1-th iteration), *i.e.*, $P(g(D_{\lambda_i})+\eta'_i \in [c_i-\alpha_j, c_i+\alpha_j]$. Since our algorithm does not know the actual value of $g(D_{\lambda_i})$, we use a distribution on $g(D_{\lambda_i})$ based on the noisy count $G[i] = (g(D_{\lambda_i}) + \eta_i)$ where $\eta_i \sim Laplace(1/\epsilon_{j-1})$ from the previous iteration $j-1$. The distribution on aggregated counts $g(D_{\lambda_i})$ can be thought of as $X \sim Laplace(\mu_x = G[i], 1/\epsilon_{j-1})$. Using the distribution on $g(D_{\lambda_i})$ *i.e.*, X, and the distribution on η'_i *i.e.*, $Y \sim Laplace(\mu_y = 0, \sigma_y = 1/\epsilon_j)$, we calculate the combined distribution of $Z = X+Y$ is the distribution of noisy aggregates in iteration j *i.e.*, the distribution of $g(D_{\lambda_i})+\eta'_i$. Given two random distributions of X and Y that are independent, the cumulative density function (CDF) of $Z = X + Y$ can be written as follows:

$$
\begin{aligned}
f_{Z=X+Y}(z) &= \int_{-\infty}^{\infty} f_X(x) f_Y(z-x) dx \\
f_{Z=X+Y}(z) &= \int_{-\infty}^{\infty} \frac{\epsilon_{j-1}}{2} e^{-|x-\mu_x|\epsilon_{j-1}} \times \frac{\epsilon_j}{2} e^{-|z-x|\epsilon_j} dx \\
&\text{for } Z > \mu_x \\
f_{Z=X+Y}(z) &= \frac{\epsilon_{j-1}\epsilon_j}{4} \times e^{(\mu_x - z)} \Big(\frac{e^{\epsilon_j} + e^{\epsilon_{j-1}}}{\epsilon_j + \epsilon_{j-1}} + \frac{e^{\epsilon_j} - e^{\epsilon_{j-1}}}{\epsilon_{j-1} - \epsilon_j} \Big) \\
&\text{for } Z \leq \mu_x \\
f_{Z=X+Y}(z) &= \frac{\epsilon_{j-1}\epsilon_j}{4} \times e^{(z - \mu_x)} \Big(\frac{e^{\epsilon_j} + e^{\epsilon_{j-1}}}{\epsilon_j + \epsilon_{j-1}} + \frac{e^{\epsilon_j} - e^{\epsilon_{j-1}}}{\epsilon_{j-1} - \epsilon_j} \Big)
\end{aligned}
\tag{A.4}
$$

We can calculate the Expected number of undecided predicated based on distribution of noisy aggregates $g(D_{\lambda_i}) + \eta_i'$ from iteration j *i.e.*, f_Z as follows

$$
\begin{aligned}
n_u &= \sum_{\lambda_i \in O_u} P(g(D_{\lambda_i}) + \eta_i' \in [c_i - \alpha_j, c_i + \alpha_j]) \\
&\approx \sum_{\lambda_i \in O_u} \int_{c_i - \alpha_j}^{c_i + \alpha_j} f_Z dz \\
&\approx \sum_{\lambda_i \in O_u} \int_{c_i - \alpha_j}^{c_i + \alpha_j} \int_{-\infty}^{\infty} \frac{\epsilon_{j-1}}{2} e^{-|x - G[i]|\epsilon_{j-1}} \times \frac{\epsilon_j}{2} e^{-|z-x|\epsilon_j} dx dz
\end{aligned}
\tag{A.5}
$$

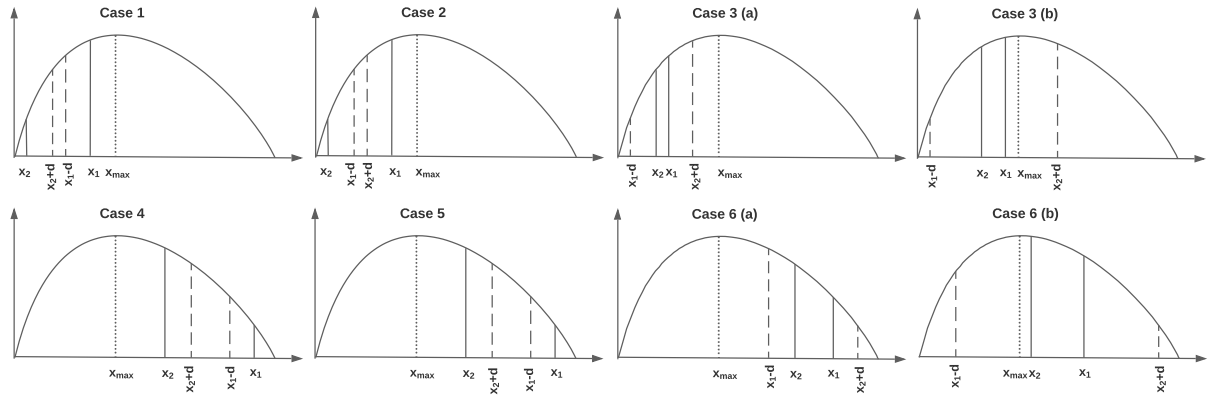


Figure A.1: Different cases for x_1, x_2 and d in LemmaA.1.3.

A.1.3 Correctness of MinEnt Algorithm

In this section, we provide the proof of Theorem 5.7. The proof of the theorem is dependent on a lemma as described below.

Lemma A.2. *Let $x_1, x_2, d \in \mathcal{R}^+$ where $x_1 > x_2$, and let $f(x) = -xlog(x)$, then $f(x_1) + f(x_2) < f(x_1 - d) + f(x_2 + d)$.*

Proof. The proof of this lemma depends on the exact values of x_1, x_2, and d. We consider the different values of the variables as different scenarios (see Figure A.1) and show that the Lemma A.1.3 holds in all scenarios. In the following, we refer to x_{max} as the value of $x \in \mathcal{R}^+$ at which the entropy function attains its maximum value.

Case 1. $x_2 < x_1 < x_{max}$ **and** $d < (x_2 - x_1)/2$**.** The entropy function is a concave function with a monotonically increasing pattern within 0 and x_{max}. Hence, between any two points x_1 and $x_1 - d$ both less than x_{max}, the following relationship of $f(x)$ holds: $\frac{f(x_1)-f(x_1-d)}{d} < f'(x_1-d)$, where $f'(x_1-d)$ is the slope of $f(x)$ at (x_1-d). The reason is that the tangent at point x_1 will have higher slope as compared to the straight line connecting $(x_1, f(x_1))$ and $(x_1-d, f(x_1-d))$. Similarly for the points x_2+d and x_2 following condition holds: $\frac{f(x_2+d)-f(x_2)}{d} > f'(x_2+d)$, since (x_2+d) is the larger among the points of x_2 and

$(x_2 + d)$. Simplifying them, we derive the following condition:

$$
\begin{aligned}
& f(x_1) - f(x_1 - d) > d(f'(x_1 - d)), f(x_2 + d) - f(x_2) < d(f'(x_2 + d)) \\
& \Rightarrow f(x_1) - f(x_1 - d) > d(f'(x_1 - d)), \\
& f(x_2) - f(x_2 + d) > -d(f'(x_2 + d)) \\
& \text{Adding these two terms,} \\
& \Rightarrow [f(x_1) - f(x_1 - d)] - [f(x_2 + d) - f(x_2)] < \\
& d(f'(x_1 - d) - f'(x_2 + d))
\end{aligned}
\tag{A.6}
$$

Since $f'(x_1 - d) - f'(x_2 + d) < 0$ when $x_1 > x_2$, the L.H.S. of the equation in Lemma A.1.3 will be less than zero.

Case 2. $x_2 < x_1 < x_{max}$ **and** $d > (x_2 - x_1)/2$ **and** $d < (x_2 - x_1)$**.** We use the similar arguments as the previous case here. Only difference from the previous case is that the value of $(x_1 - d)$ becomes lower than the value of $(x_2 + d)$. The ordering of the points become as follows: $x_2 < x_1 - d < x_2 + d < x_1$. Hence, if we choose the pair of points x_1 and $(x_2 + d)$, the following condition holds: $\frac{f(x_1) - f(x_2 + d)}{x_1 - x_2 - d} < f'(x_2 + d)$. Similarly, considering the pair of points x_2 and $(x_1 - d)$, the following condition holds: $\frac{f(x_1 - d) - f(x_2)}{x_1 - d - x_2} < f'(x_1 - d)$. Hence, we derive the following condition:

$$
\begin{aligned}
& [f(x_1) - f(x_2 + d)] - [f(x_1 - d) - f(x_2)] \\
& < (x_1 - d - x_2)(f'(x_2 + d) - f'(x_1 - d)) < 0
\end{aligned}
\tag{A.7}
$$

Hence the equation in the Lemma is satisfied for this case.

Case 3. $x_2 < x_1 < x_{max}$ **and** $d > (x_2 - x_1)$**.** In this case, the value of $(x_1 - d)$ becomes lower than x_2 and the value of x_2 becomes higher than x_1. The ordering of the four points are as follows: $x_1 - d < x_2 < x_1 < x_2 + d$. If the slope of the line joining x_1 and $x_2 + d$ stays above zero, then the proof of the lemma stays the same as Case 2, since Equation A.8 is

satisfied. If the slope of the line becomes negative(*i.e.*, $\frac{f(x_1)-f(x_2+d)}{x_1-x_2-d} < 0$), then the following condition holds: $\frac{f(x_1)-f(x_2+d)}{x_1-d-x_2} < f'(x_1)$, since $f'(x_1) > 0$. Hence, we derive the following condition:

$$\begin{aligned}&[f(x_1) - f(x_2 + d)] - [f(x_1 - d) - f(x_2)] \\ &< (x_1 - d - x_2)(f'(x_1) - f'(x_1 - d)) < 0\end{aligned} \tag{A.8}$$

Hence the equation in the Lemma is satisfied for this case.

Case 4. $x_{max} < x_2 < x_1$ **and** $d < (x_2 - x_1)/2$**.** In this part of the domain, the entropy function is a monotonically decreasing function. The ordering of the four points are as follows: $x_2 < x_2 + d < x_1 - d < x_1$. Considering the points x_1 and $x_1 - d$, we derive the following condition: $\frac{f(x_1)-f(x_1-d)}{d} < f'(x_1 - d)$. This is true because the value of $f'(x_1 - d)$ is more negative as compared to the slope of line joining x_1 and $x_1 - d$. Similarly the condition of $\frac{f(x_2+d)-f(x_2)}{d} > f'(x_2 + d)$ holds. Hence, proceeding the same way as Case 1, we observe that Equation A.6 and the lemma holds.

Case 5. $x_{max} < x_2 < x_1$ **and** $d > (x_2 - x_1)/2$ **and** $d < (x_2 - x_1)$**.** The ordering of the points are as follows: $x_2 < x_1 - d < x_2 + d < x_1$. Considering the pair of points x_1 and $x_2 + d$, the following condition holds: $\frac{f(x_1)-f(x_2+d)}{x_1-x_2-d} < f'(x_1)$. Similarly the condition of $\frac{f(x_1-d)-f(x_2)}{x_1-d-x_2} > f'(x_2)$ is true. Hence the following condition can be derived from them:

$$\begin{aligned}&[f(x_1) - f(x_2 + d)] - [f(x_1 - d) - f(x_2)] \\ &< (x_1 - d - x_2)(f'(x_2) - f'(x_1)) < 0\end{aligned} \tag{A.9}$$

Case 6. $x_{max} < x_2 < x_1$ **and** $d > (x_2 - x_1)$**.** The ordering of the point is as follows:

$x_1 - d < x_2 < x_1 < x_2 + d$. If the value of d is such that the slope of the line joining x_2 and $x_1 - d$ remain negative then the proof follows the same as Case 5. If the slope of $\frac{f(x_2)-f(x_1-d)}{x_1-d-x_2}$ becomes positive, then $\frac{f(x_1-d)-f(x_2)}{x_1-d-x_2} > f'(x_1 - d)$ holds since $f'(x_1 - d)$ is positive. Hence we can derive the following:

$$\begin{aligned} &[f(x_1) - f(x_2 + d)] - [f(x_1 - d) - f(x_2)] \\ &< (x_1 - d - x_2)(f'(x_2) - f'(x_1 - d)) < 0 \end{aligned} \tag{A.10}$$

From the above condition, we can conclude that the L.H.S. of the equation in Lemma A.1.3 is less than the R.H.S. of the equation.

□

Proof of Theorem 5.7. As a first step, we prove that option 3 is worse than either option 1 or option 2 when the number of predicates k is 2. Using this result, we later generalize it to the scenario of $k > 2$.

Let $\{l_1, u_1\}, \{l_2, u_2\}$ be a set of two ranges of posterior probability values of the adversary p_1, p_2 such that $u_1 \leq u_2$ and $l_1 \geq l_2$. Let s be the total amount of slack that needs to be distributed among these two ranges and $\Delta_1 = u_1 - l_1$ and $\Delta_2 = u_2 - l_2$. Let $f(x) = -x \log(x)$, our optimization function for Option 1, Option 2 and Option 3 can be written as follows

$$\begin{aligned} &Opt1 : f(l_2 + min(\Delta_2, s)) + f(l_1 + (s - min(\Delta_2, s)) \\ &Opt2 : f(l_1 + min(\Delta_1, s) + f(l_2 + (s - min(\Delta_1, s)) \\ &Opt3 : f(l_2 + \alpha\Delta_2) + f(l_1 + (s - \alpha\Delta_2)) \text{ where } 0 < \alpha < 1, \\ &s - \alpha\Delta_2 < \Delta_1 \end{aligned} \tag{A.11}$$

We show it for all possible cases:

Case 1. $s < \Delta_2$ **and** $s < \Delta_1$. For this case, Opt1 and Opt2 will be as follows:

$$\begin{aligned} &Opt1 : f(l_2 + s) + f(l_1) \\ &Opt2 : f(l_1 + s) + f(l_2) \end{aligned} \tag{A.12}$$

if $l_2 + s > l_1$, Opt1 is better than Opt3 by Lemma A.1.3 for $x_1 = l_2 + s, x_2 = l_1, d = s - \alpha\Delta_2$. Similarly, if $l_1 \geq l_2 + s$, Opt2 is better than Opt3 by Lemma A.1.3.

Case 2.$s < \Delta_2$ **and** $s \geq \Delta_1$. For this case, Opt1, and Opt2 will be as follows:

$$\begin{aligned} &Opt1 : f(l_2 + s) + f(l_1) \\ &Opt2 : f(l_1 + \Delta_1) + f(l_2 + s - \Delta_1) \end{aligned} \tag{A.13}$$

if $l_2 + s > l_1$, Opt1 is better than Opt3 by Lemma A.1.3 for $x_1 = l_2 + s, x_2 = l_1, d = s - \alpha\Delta_2$. Similarly, if $l_1 \geq l_2 + s$, Opt2 is better than Opt3 by Lemma A.1.3.

Case 3. $s \geq \Delta_2$ **and** $s \geq \Delta_1$. For this case, Opt1, Opt2, Opt3 will be as follows:

$$\begin{aligned} &Opt1 : f(l_2 + \Delta_2) + f(l_1 + s - \Delta_2) \\ &Opt2 : f(l_1 + \Delta_1) + f(l_2 + s - \Delta_1) \end{aligned} \tag{A.14}$$

As $l_2 + \Delta_2 > l_1 + s - \Delta_2$, Opt1 is always better than Opt3 by Lemma A.1.3 for $x_1 = l_2 + \Delta_2, x_2 = l_1 + s - \Delta_2, d = \Delta_2 - \alpha\Delta_2$.

Hence, based on all possible cases (Case 1, Case 2, Case 3) either option 1 or option 2 are always better than option 3 for k=2.

For $k > 2$, we prove the theorem as follows: we divide the overall slack into two parts: s' and $(s - s')$. The slack of s' is divided among two ranges of p_m and p_n and the slack of $(s - s')$ distributed among the remaining intervals of $\{p_i \mid i \neq m, i \neq n\}$. We represent the overall objective function as $\sum_{i=1}^{k} f(p_i)$, where p_i is the solution chosen for i-th interval. We assume that the slack of s' is optimally distributed among the intervals other than m-th and n-th interval. We rewrite the objective function as follows $F_1(p_i) + f(p_m) + f(p_n)$, where function $F_1(p_i)$ is the value of the summation function when slack $(s - s')$ is optimally distributed among the intervals other than m-th and n-th intervals.

In order to distribute slack s' among p_m and p_n, we have already shown in the previous proof that the option 1 or option 2 performs better than option 3.

A.1.4 Complexity of MinEnt Algorithm.

The time complexity of the algorithm is $\mathcal{O}(2^k)$ where k is the number of predicates, $k = |\Lambda|$. To limit the exponential time complexity, we exploit several edge cases containing conditions (presented in Algorithm 8) on the solutions of $p1$ and $p2$ to restrict the recursion tree. For example, let us consider the first condition, where the slack s is greater than both $\sum_{i=1}^{k-1} \Delta_i$ and Δ_k in iteration i. For the solution of the subproblem of size $k - 1$ for option 2 *i.e.*, $p2$, we assign u_i for all $\hat{p}_i$s and the the rest of the slack is added to $p2[k]$ instead of calling the MinEnt() again. Similarly, for condition 2, when s in smaller than both $\sum_{i=1}^{k-1} \Delta_i$ and Δ_k, option 1 has a simpler solution that is to assign all slack to $p2[k]$ and terminate option 1. This results in calling MinEnt() function only once. There is only one edge case that calls MinEnt() twice. Using these conditions, we are able to reduce some of the complexity.

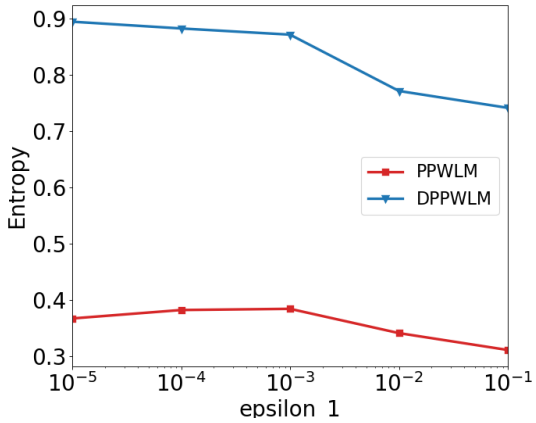
Another optimization that we do is to round of slack and upper and lower bound on $\hat{p}_i$ to 3 decimal points. This also results in faster termination of MinEnt() algorithm with a good estimate on min- entropy.

Algorithm 8 Conditions

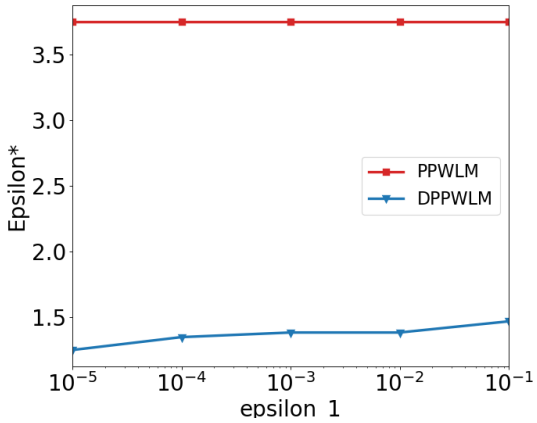
if $s \geq max(\sum_{i=1}^{k-1} \Delta_i, \Delta_k)$ **then**
  $p1[k] = u_k$
  $p1[i] =$MinEnt$([l_i, u_i] \forall i \in \{1, ..., k-1\}, s - \Delta_k)$
  $p2[i] = u_i, \forall i \in \{1, 2, ..., k-1\}$
  $p2[k] = l_k + s - \sum_{i=1}^{k-1} \Delta_i$
else if $s \leq min(\sum_{i=1}^{k-1} \Delta_i, \Delta_k)$ **then**
  $p1[k] = l_k + s$
  $p1[i] = l_i, \forall i \in \{1, 2, ..., k-1\}$
  $p2[i] =$MinEnt$([l_i, u_i] \forall i \in \{1, 2, ..., k-1\}, s)$
  $p2[k] = l_k$
else if $s \geq \Delta_k$ **and** $s \leq \sum_{i=1}^{k-1} \Delta_i$ **then**
  $p1[k] = u_k$
  $p1[i] =$MinEnt$([l_i, u_i] \forall i \in \{1, ..., k-1\}, s - \Delta_k)$
  $p2[i] =$MinEnt$([l_i, u_i] \forall i \in \{1, 2, ..., k-1\}, s)$
  $p2[k] = l_k$
else if $s \leq \Delta_k$ **and** $s \geq \sum_{i=1}^{k-1} \Delta_i$ **then**
  $p1[k] = l_k + s$
  $p1[i] = l_i, \forall i \in \{1, 2, ..., k-1\}$
  $p2[i] = u_i, \forall i \in \{1, 2, ..., k-1\}$
  $p2[k] = l_k + s - \sum_{i=1}^{k-1} \Delta_i$
end if

A.1.5 Additional Experiment Results

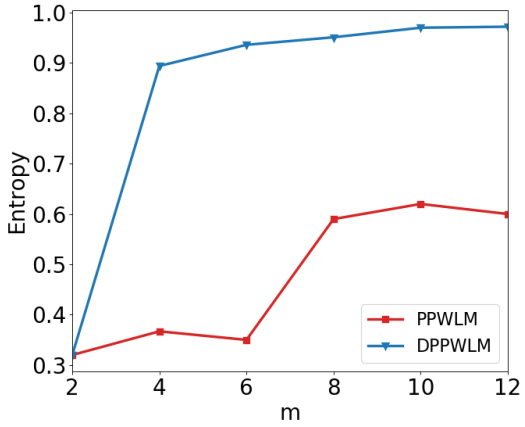
Epsilon per predicate. Figure A.3 shows the cumulative number of predicates that have a final ex-post privacy cost smaller than a value for a sample query $Q3$ with threshold = 'Medium' . In TSLM, all the predicates ended with a high ϵ value of 1.2. In contrast, both the multi-step algorithms PPWLM and DPPWLM made decision for a large number of predicates at a much lower ϵ value than TSLM. For example, DPPWLM had up to 380 predicates with an ϵ value lower than 0.01, which leads to a much smaller min-entropy than the other algorithms. This example shows the progressive algorithms have an advantage over



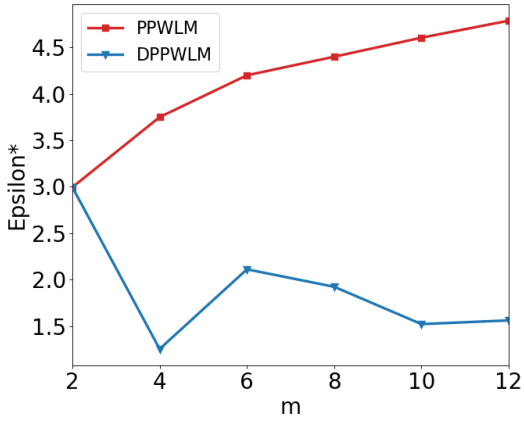
(a) Min-Entropy $\gamma(\Theta)$



(b) Ex-Post Privacy Loss ϵ^*



(c) Min-Entropy $\gamma(\Theta)$



(d) Ex-Post Privacy Loss ϵ^*

Figure A.2: Privacy Loss (ϵ^*,$\gamma(\Theta)$) for $Q3$ (NYTaxi data) with threshold = Low over varying start epsilon ϵ_1 (a),(b) and over varying number of steps m (c),(d)

TSLM to achieve lower privacy loss per predicate.

Start epsilon for progressive algorithms. We choose a small epsilon as the starting epsilon ($\epsilon_1 = 0.00001$) in our experiments. We present the results of privacy loss in terms of min-entropy and ex-post dp in Figure A.2a (a) and (b) of DPPWLM over varying values of starting epsilon for one of the queries Q3 on NYTaxi data with threshold = LOW. The results show that a larger value of starting epsilon may result in overall higher privacy loss in terms of both min-entropy and ex-post DP loss as the algorithm misses the opportunity of using smaller values of epsilons in the beginning. If epsilon is too small, all predicates may

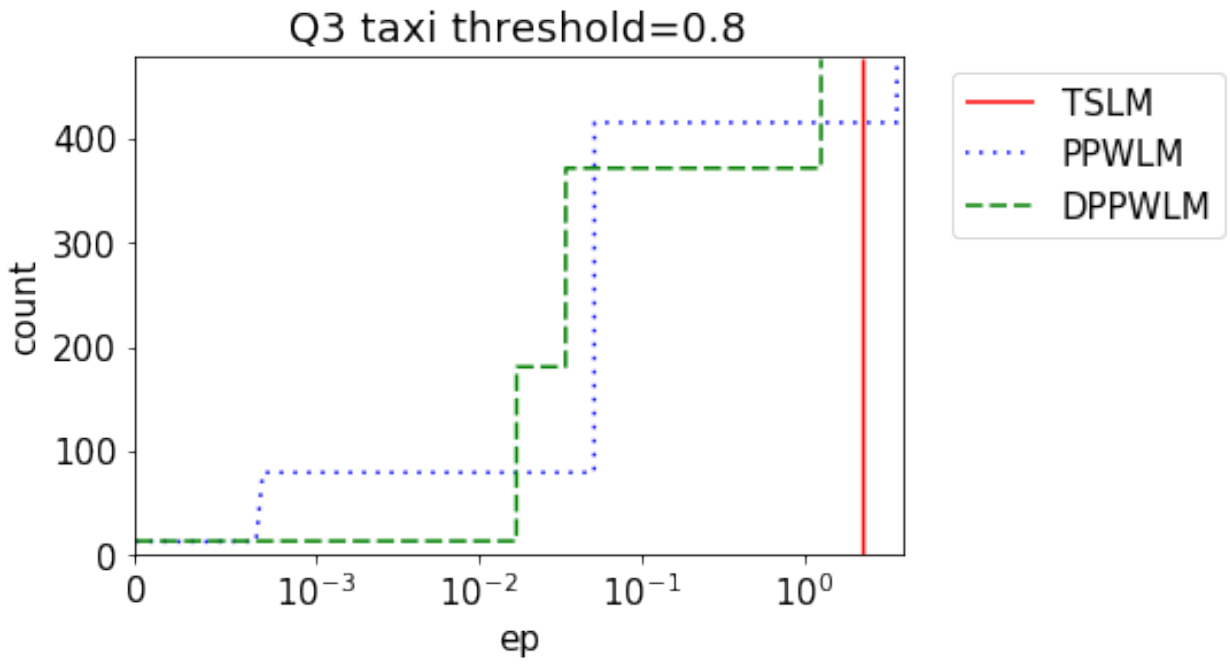


Figure A.3: Cumulative distribution of epsilon per predicate in TSLM, PPWLM, DPPWLM for Q3 with threshold = Medium.

end up undecided in the first step. This can be data dependent because if all aggregates lie close to the threshold choosing a very small epsilon is not ideal as it will make the uncertain region to be large enough that none of the predicates get eliminated. Our data dependent algorithm takes care of it, it may end up resulting in all predicates to be undecided in the first step however it will choose appropriate epsilons in the next step such that some predicates are classified the results show that the best starting epsilon for DPPWLM is 0.00001 and this results holds for other queries too.

Number of steps We choose number of steps $m = 4$ without exceeding our $\epsilon_{max} = 4$ requirement. We perform the experiments for one of the queries Q3 on NYTaxi data with threshold = LOW and present the result in Figure A.2 (c) and (d). The plots shows the privacy metrics min-entropy and ex-post privacy loss with varying number of steps. We can see that for PPWLM, as we increase the number of steps more than 4, ex-post privacy loss exceeds our DP requirement of $\epsilon_{max} = 4$ as there was no early stop for this particular query.

This mean there was at-least one predicate with did not get classified until the last step. For number of steps less than $m = 4$, both PPWLM and DPPWLM under perform and have higher ex-post dp cost as there are not enough number of steps to exploit the data distribution. The ex-post dp cost for DPPWLM can be different than PPWLM based on data distribution and at which step the algorithm stops. The classification of a predicate as decided (positive or negative) or undecided (needs to be reclassified in the next step) depends on the selection of *epsilon* and β for intermediate steps. In this particular query, the ex-post dp cost never exceeds ϵ_{max}. However it is possible that it exceeds for DPPWLM just like it did for PPWLM depending upon the data distribution. Our experiments show that it is best to select max number of steps (m=4) to ensure that we satisfy our $\epsilon_{max} = 4$ requirement and our data dependent algorithm DPPWLM has more options to select the best ϵ and β in each step. The DPPWLM does better in term of min-entropy as the number of steps increases as compare to PPWLM as DPPWLM is optimizes the choice of ϵ and β to maximize the min entropy.

Appendix B

Empirical Study Appendix

B.1 Derivations of some probabilities

$\mathbf{Pr}[u \in l@t \mid C]$

$$
\begin{aligned}
&= \frac{\mathbf{Pr}[u \in l@t \wedge C]}{\mathbf{Pr}[C]} \\
&= \frac{\mathbf{Pr}[u \in l@t \wedge C]}{\mathbf{Pr}[u \in l@t \wedge C] + \mathbf{Pr}[u \notin l@t \wedge C]} \\
&= \frac{1}{1 + \frac{\mathbf{Pr}[u \notin l@t \wedge C]}{\mathbf{Pr}[u \in l@t \wedge C]}} = \frac{1}{1 + \frac{\sum_{l \neq \ell \in \mathcal{L}} \mathbf{Pr}[u \in \ell@t \wedge C]}{\mathbf{Pr}[u \in l@t \wedge C]}} \ .
\end{aligned}
$$

$\mathbf{Pr}[u \in l@t \wedge C]$

$$
\begin{aligned}
&= p^t_{ul} \cdot \sum_{\substack{(|S_\ell|=c^t_\ell)_{\ell\neq l}, |S_l|=c^t_l-1 \\ \forall k,\ell: S_k\cap S_\ell=\emptyset, \forall \ell: u\notin S_\ell}} \prod_{\ell\in\mathcal{L}} \prod_{v\in S_\ell} p^t_{v\ell} \\
&= p^t_l \cdot \sum_{\substack{(|S_\ell|=c^t_\ell)_{\ell\neq l}, |S_l|=c^t_l-1 \\ \forall k,\ell: S_k\cap S_\ell=\emptyset, \forall \ell: u\notin S_\ell}} \prod_{\ell\in\mathcal{L}} \prod_{v\in S_\ell} p^t_{\ell} \\
&= p^t_l \cdot \sum_{\substack{(|S_\ell|=c^t_\ell)_{\ell\neq l}, |S_l|=c^t_l-1 \\ \forall k,\ell: S_k\cap S_\ell=\emptyset, \forall \ell: u\notin S_\ell}} \prod_{\ell\in\mathcal{L}} (p^t_\ell)^{|S_\ell|} \\
&= p^t_l \cdot \frac{(n^t_u-1)!}{c^t_1!\cdots(c^t_l-1)!\cdots c^t_{n_l}!} (p^t_l)^{c^t_l-1} \prod_{l\neq\ell\in\mathcal{L}} (p^t_\ell)^{c^t_\ell} \\
&= c^t_l \cdot \frac{(n^t_u-1)!}{c^t_1!\cdots c^t_{n_l}!} \prod_{\ell\in\mathcal{L}} (p^t_\ell)^{c^t_\ell} \ .
\end{aligned}
$$

$\mathbf{Pr}[u \in l@t \mid C]$

$$
\begin{aligned}
&= \frac{1}{1+\frac{\sum_{l\neq\ell\in\mathcal{L}} \mathbf{Pr}[u\in\ell@t\wedge C]}{\mathbf{Pr}[u\in l@t\wedge C]}} \\
&= \frac{1}{1+\frac{\sum_{l\neq\ell\in\mathcal{L}} c^t_\ell}{c^t_j}} = \frac{1}{1+\frac{n^t_u-c^t_j}{c^t_j}} = \frac{c^t_j}{n^t_u} \ .
\end{aligned}
$$

B.2 Convergence

For Laplace noise, the estimates $\mathbf{Pr}[x \mid y, \mathsf{A}]$ on attacker's posterior probability stay between $\mathbf{Pr}[x \mid \mathsf{A}]$, i.e. the prior, and $\mathbf{Pr}[x \mid c, \mathsf{A}]$, i.e. guessing from true counts.

Theorem B.1. *Let Laplace mechanism Lap$(1/\epsilon)$ be in use. We have*

- $\lim_{\epsilon\to 0} \mathbf{Pr}[x \mid y, \mathsf{A}] = \mathbf{Pr}[x \mid \mathsf{A}]$;

- $\lim_{\epsilon\to\infty} \mathbf{Pr}[x \mid y, \mathsf{A}] = \mathbf{Pr}[x \mid c, \mathsf{A}]$.

Proof. First of all, let us state the properties that we require from the noise distribution.

- $\lim_{\epsilon\to\infty} f_Y(y \mid c) = 1$;
- $\lim_{\epsilon\to 0} f_Y(y \mid z) = 0$ for $z \neq c$;
- $\lim_{\epsilon\to 0} f_Y(y \mid z) = \alpha(\epsilon)$ for some constant $\alpha(\epsilon)$ for all y, z.

For Laplace noise, we have $\lim_{\epsilon\to\infty} \frac{\epsilon}{2} \cdot e^{-\epsilon|y-c|} = 1$ iff $y = c$, and $\lim_{\epsilon\to 0} \frac{\epsilon}{2} \cdot e^{-\epsilon|y-c|} = \frac{\epsilon}{2}$. As we show next, these properties are sufficient to prove Theorem B.1, so our proof works for any other distribution satisfying these properties, e.g. any additive noise with bell-shaped distribution that scales with ϵ. Take the definition of $\mathbf{Pr}[x \mid y, \mathsf{A}]$ from Eq. 4.4. We have

$\lim_{\epsilon\to 0} \mathbf{Pr}[x \mid y, \mathsf{A}]$

$$
\begin{aligned}
&= \lim_{\epsilon\to 0} \frac{\sum_{z\in C} \mathbf{Pr}[x|\mathbf{C} = z, \mathsf{A}] \cdot f_Y(y|z)\mathbf{Pr}[z \mid \mathsf{A}]}{\sum_{z\in C} f_Y(y|z)\mathbf{Pr}[z \mid \mathsf{A}]} \\
&= \frac{\sum_{z\in C} \mathbf{Pr}[x|\mathbf{C} = z, \mathsf{A}] \cdot \alpha(\epsilon) \cdot \mathbf{Pr}[z \mid \mathsf{A}]}{\sum_{z\in C} \alpha(\epsilon) \cdot \mathbf{Pr}[z \mid \mathsf{A}]} \\
&= \mathbf{Pr}[x \mid \mathsf{A}] \ .
\end{aligned}
$$

$\lim_{\epsilon\to\infty} \mathbf{Pr}[x \mid y, \mathsf{A}]$

$$
\begin{aligned}
&= \lim_{\epsilon\to\infty} \frac{\sum_{z\in C} \mathbf{Pr}[x|\mathbf{C} = z, \mathsf{A}] \cdot f_Y(y|z)\mathbf{Pr}[z \mid \mathsf{A}]}{\sum_{z\in C} f_Y(y|z)\mathbf{Pr}[z \mid \mathsf{A}]} \\
&= \frac{\mathbf{Pr}[x|\mathbf{C} = c, \mathsf{A}] \cdot f_Y(y|c)\mathbf{Pr}[c \mid \mathsf{A}]}{f_Y(y|c)\mathbf{Pr}[c \mid \mathsf{A}]} \\
&= \mathbf{Pr}[x|\mathbf{C} = c, \mathsf{A}] \ .
\end{aligned}
$$

□

B.3 Composition

Let $c = (c_1, \ldots, c_n)$ be the true output. First of all, since we want to bound atatcker's inference probability by $\mathbf{Pr}[x|c]$, we need to take into account at least all elements of c on which $\mathbf{Pr}[x|c]$ depends, and we cannot estimate it only based on $\mathbf{Pr}[x|c_l]$ for some particular $l \in \mathcal{L}$. Hence, we need to constrain ourselves to the situtations $c = c_a, c_b$, where $\mathbf{Pr}[x|c] = \mathbf{Pr}[x|c_a]$. In our case study, we have $\mathbf{Pr}[x|z] = \frac{z}{m}$. Indeed, the value m does depend on the other counts as well, but we treat m as a public parameter that makes c_i correlated.

Parallel composition for independent c_a, c_b is similar to the one of DP.

Theorem B.2. *Let $C = C_a \times C_b$. Let $\mathbf{Pr}[z_a, z_b \mid \mathsf{A}] = \mathbf{Pr}[z_a \mid \mathsf{A}] \cdot \mathbf{Pr}[z_b \mid \mathsf{A}]$ for all $z_a \in C_a$, $z_b \in C_b$. We have*

$$\mathbf{Pr}[x \mid y, \mathsf{A}] = \frac{\sum_{z_a \in C_a} \mathbf{Pr}[x|z_a, \mathsf{A}] \cdot f_Y(y|z_a)\mathbf{Pr}[z_a \mid \mathsf{A}]}{\sum_{z_a \in C_a} f_Y(y|z_a)\mathbf{Pr}[z_a \mid \mathsf{A}]} \ .$$

It is more complicated with sequential composition. In another extreme, c_i may be completely correlated, giving $\mathbf{Pr}[g(z_a)|\mathbf{C}_a = z_a] = 1$ for a deterministic function g. In a particular case where g is an identity function, we get something similar to sequential composition of DP. This allows to use the same z instead of z_a and z_b, so that summation over z_b can be avoided. We can use it if multiple queries are applied to the same data.

Theorem B.3. *Let $C = C_a \times C_b$, where $C_a = C_b$. Let $\mathbf{Pr}[\mathbf{C}_b = z_a|\mathbf{C}_a = z_a, \mathsf{A}] = 1$ for all $z_a \in C_a$. We have*

$$\mathbf{Pr}[x \mid y, \mathsf{A}] = \frac{\sum_{z_a \in C_a} \mathbf{Pr}[x|z_a, \mathsf{A}] f_Y(y_a, y_b|z_a)\mathbf{Pr}[z_a \mid \mathsf{A}]}{\sum_{z_a \in C_a} f_Y(y_a, y_b|z_a)\mathbf{Pr}[z_a \mid \mathsf{A}]} \ .$$

where $f_Y(y_a, y_b|z_a) = f_Y(y_a|z_a) \cdot f_Y(y_b|z_a)$.

Finally, it is possible that the outputs are strongly correlated, but g is not an identity function. In this case, the estimation is more difficult, as we want to get a general upper bound for all possible definitions of g. One idea is to apply the worst case bound to the Z_b part, e.g. Sec. 4.4.2 for a DP mechanism.

Theorem B.4. *Let* $C = C_a \times C_b$, $\mathbf{Pr}[g(z_a)|\mathbf{C}_a = z_a, \mathsf{A}] = 1$ *for a deterministic function* g. *We have*

$$\mathbf{Pr}[x \mid y, \mathsf{A}] \leq \max_{z_b, z_b' \in Z_b} \left(\frac{f_Y(y_b|z_b)}{f_Y(y_b|z_b')} \right) \cdot \cdot \frac{\sum_{z_a \in C_a} \mathbf{Pr}[x|z_a, \mathsf{A}] \cdot f_Y(y_a|z_a) \mathbf{Pr}[z_a \mid \mathsf{A}]}{\sum_{z_a \in C_a} f_Y(y_a|z_a) \mathbf{Pr}[z_a \mid \mathsf{A}]} \ .$$

We can instantiate Theorem B.4 on an ϵ-DP mechanism.

Corollary 1. *Let* $C = C_1 \times \cdots \times C_n$. *Let an* ϵ*-DP mechanism w.r.t. norm* $\|\cdot\|$ *be applied to each* C_j. *Let* $\max_{x,x' \in X} \|x - x'\| = m$. *We have*

$$\mathbf{Pr}[x \mid y, \mathsf{A}] \leq e^{\epsilon \cdot m(n-1)} \cdot \cdot \frac{\sum_{z \in C_1} \mathbf{Pr}[x|z, \mathsf{A}] \cdot f_Y(y_a|z) \mathbf{Pr}[z \mid \mathsf{A}]}{\sum_{z \in C_1} f_Y(y_a|z) \mathbf{Pr}[z \mid \mathsf{A}]} \ .$$

We will now state and prove some Lemmas which help us in proving Theorems B.2-B.4 and Corollary 1.

Lemma B.1. *Let* $\mathcal{M}$ *be any* ϵ*-differentially private mechanism. Let* $z, z' \in q(X)$. *Let* $M = \max_{x \in q^{-1}(z), x' \in q^{-1}(z')} \|x - x'\|$. *For all* $y, y' \in \mathcal{M}(X)$ *we have*

$$f_Y(y|z) \leq f_Y(y|z') \cdot e^{\epsilon M} \ .$$

Proof. We have $z = q(x)$ and $z' = q(x')$ for some $x, x \in X$. Since $q(x)$ can be computed from x, we have $f_Y(y|q(x), x) = f_Y(y|x)$. Let $q^{-1}(z) := \{x \mid q(x) = z\}$. Denote $X_z := q^{-1}(z)$ and $X'_z := q^{-1}(z')$. We have

$$\begin{aligned}
\frac{f_Y(y|z)}{f_Y(y|z')} &= \frac{\sum_{x \in X_z} f_Y(y|x,z)\mathbf{Pr}[x \mid z]}{\sum_{x' \in X'_z} f_Y(y|x',z')\mathbf{Pr}[x' \mid z']} \\
&= \frac{\sum_{x \in X_z} f_Y(y|x)\mathbf{Pr}[x \mid z]}{\sum_{x' \in X'_z} f_Y(y|x')\mathbf{Pr}[x' \mid z']} \\
&= \sum_{x \in X_z} \frac{f_Y(y|x)\ \mathbf{Pr}[x \mid z]}{\sum_{x' \in X'_z} f_Y(y|x')\mathbf{Pr}[x' \mid z']} \\
&\leq \sum_{x \in X_z} \frac{e^{\epsilon\|x-x'\|}}{\sum_{x' \in X'_z} \mathbf{Pr}[x' \mid z']}\mathbf{Pr}[x \mid z] \\
&\leq e^{\epsilon M} \frac{\sum_{x \in X_z} \mathbf{Pr}[x \mid z]}{\sum_{x' \in X'_z} \mathbf{Pr}[x' \mid z']} \\
&= e^{\epsilon M}\ .
\end{aligned}$$

□

For shortness of notation, let us remove the additional knowledge A from all conditional probabilities and make it implicit. This will not affect the proofs, as A is found in the premises of all theorems. We have

$$\begin{aligned}
\mathbf{Pr}[x\ |y] = \frac{\sum_{z \in C} \mathbf{Pr}[x|z] f_Y(y|z)\mathbf{Pr}[z]}{\sum_{z \in C} f_Y(y|z)\mathbf{Pr}[z]} \\
= \frac{\sum_{z_a \in C_a} \sum_{z_b \in C_b} \mathbf{Pr}[x|z_a] f_Y(y_a, y_b|z_a, z_b)\mathbf{Pr}[z_a, z_b]}{\sum_{z_a \in C_a} \sum_{z_b \in C_b} f_Y(y_a, y_b|z_a, z_b)\mathbf{Pr}[z_a, z_b]}
\end{aligned}$$

Assuming that the noise distribution depends only on a single output (as in the case of Laplace noise), we get $f_Y(y_a, y_b|z_a, z_b) = f_{Y_a}(y_a|z_a) \cdot f_{Y_b}(y_b|z_b)$.

B.3.1 Proof of Theorem B.2

For independent variables, $\mathbf{Pr}[z_a, z_b] = \mathbf{Pr}[z_a]\mathbf{Pr}[z_b]$. This allows to split both the numerator and the denominator into a product of two independent sums.

$$\mathbf{Pr}[x\ |y] = \frac{\sum_{z_a \in C_a} \mathbf{Pr}[x|z_a] f_{Y_a}(y_a|z_a)\mathbf{Pr}[z_a]}{\sum_{z_a \in C_a} f_{Y_a}(y_a|z_a)\mathbf{Pr}[z_a]} \cdot \frac{\sum_{z_b \in C_b} f_{Y_b}(y_b|z_b)\ \mathbf{Pr}[z_b]}{\sum_{z_b \in C_b} f_{Y_b}(y_b|z_b)\ \mathbf{Pr}[z_b]}$$
$$= \frac{\sum_{z_a \in C_a} \mathbf{Pr}[x|z_a] f_Y(y|z_a)\mathbf{Pr}[z_a]}{\sum_{z_a \in C_a} f_Y(y|z_a)\mathbf{Pr}[z_a]} .$$

B.3.2 Proof of Theorem B.3

As $\mathbf{Pr}[z_a|\mathbf{C}_a = z_a] = 1$ for all $z_a \in C_a$, we can write $\mathbf{Pr}[z_a, z_b] = \mathbf{Pr}[z_a]$ and only sum over C_a. We are left with

$$\begin{aligned}\mathbf{Pr}[x\ |y] &= \frac{\sum_{z_a \in C_a} \mathbf{Pr}[x|z_a] f_Y(y_a, y_b|z_a, z_a)\mathbf{Pr}[z_a]}{\sum_{z_a \in C_a} f_Y(y_a, y_b|z_a, z_a)\mathbf{Pr}[z_a]} \\ &= \frac{\sum_{z_a \in C_a} \mathbf{Pr}[x|z_a, \mathsf{A}] \cdot f_Y(y_a, y_b|z_a)\mathbf{Pr}[z_a]}{\sum_{z_a \in C_a} f_Y(y_a, y_b|z_a)\mathbf{Pr}[z_a]} .\end{aligned}$$

B.3.3 Proof of Theorem B.4

Let g be a deterministic function such that $\mathbf{Pr}[g(z_a)|\mathbf{C}_a = z_a, \mathsf{A}] = 1$. We have

$$\mathbf{Pr}[x\ |y] = \frac{\sum_{z_a \in C_a} \mathbf{Pr}[x|z_a] \cdot f_{Y_a}(y_a|z_a) f_{Y_b}(y_b|g(z_a))\mathbf{Pr}[z_a]}{\sum_{z_a \in C_a} f_{Y_a}(y_a|z_a) f_{Y_b}(y_b|g(z_a))\mathbf{Pr}[z_a]} .$$

Differently from the previous case, we cannot compute the quantity more precisely unless we know g. We rewrite the expression as

$$\mathbf{Pr}[x\ |y] = \sum_{z_a \in C_a} \mathbf{Pr}[x|z_a] \cdot f_{Y_a}(y_a|z_a) \cdot \mathbf{Pr}[z_a] \cdot \\ \cdot \frac{1}{\sum_{z'_a \in C_a} f_{Y_a}(y_a|z'_a) \frac{f_{Y_b}(y_b|g(z'_a))}{f_{Y_b}(y_b|g(z_a))} \mathbf{Pr}[z'_a]} \ .$$

A trivial upper bound on $\frac{f_{Y_b}(y_b|g(z_a))}{f_{Y_b}(y_b|g(z'_a))}$ is $\max_{z_b, z'_b \in Z_b} \frac{f_{Y_b}(y_b|z_b)}{f_{Y_b}(y_b|z'_b)}$. This upper bound is independent of the summation variables and can be taken out of the sum.

B.3.4 Proof of Corollary 1

By assumption of Corollary 1, we are dealing with an ϵ-DP mechanism. by Lemma B.1, we have $\frac{f_{Y_i}(y_i|z_i)}{f_{Y_i}(y_i|z'_i)} \leq e^{\epsilon \cdot \max_{x,x'} \|x-x'\|}$. In our case $\|x - x'\| \leq m$ for a singe occupancy count. We instantiate Theorem B.4 with $Z_a = C_1$ and $Z_b = (C_2, \ldots, C_n)$. We have $\frac{f_{Y_b}(y_n|z_b)}{f_{Y_b}(y_b|z'_b)} = \frac{\prod_{j=2}^{n} f_{Y_j}(y_j|z_j)}{\prod_{j=2}^{n} f_{Y_j}(y_j|z'_j)} \leq \prod_{j=2}^{n} e^{\epsilon \cdot \max_{x,x'} \|x-x'\|} = e^{\epsilon \cdot m(n-1)}$.

B.4 Evaluating Particular Instances of Noisy Counts

While Figures 4.7-4.9 show average guessing advantage over 30 instances of noisy data, Figure B.1 depicts results for some single run experiments. Each row corresponds to a particular instance of noisy counts. In this example, we are comparing Laplace and PeGaSus for 75% day utility, for the student attacker (two left columns) and the administrator attacker (two right columns). We see that success of PeGaSus comes from the fact that localizations

for some timepoints get completely missed. In overall, using Laplace noise seems to give more stable results that do not vary much over instances. Using PeGaSus, some people can be predicted better and some worse. For the student attacker, the advantage of PeGaSus is not as clear as it is in Figure 4.8.

In general, choosing a suitable privacy mechanism depends on how we define the privacy goal. Do we want to minimize the average probability of localization over all people (an individual localized with 100% confidence is fine if there is only one such person), or do we want to minimize the maximum probability over all people (we guarantee privacy for all, but less for each).

Figure B.1: Comparison of PeGaSus and Laplace mechanisms for single runs on particular noisy data instances