

DEFINING AND ENFORCING PRIVACY IN DATA SHARING

A Dissertation

Presented to the Faculty of the Graduate School
of Cornell University

in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

by

Ashwin Kumar VenkataNaga Machanavajhala

August 2008

© 2008 Ashwin Kumar VenkataNaga Machanavajjala

ALL RIGHTS RESERVED

DEFINING AND ENFORCING PRIVACY IN DATA SHARING

Ashwin Kumar VenkataNaga Machanavajhala, Ph.D.

Cornell University 2008

Recent advances in processing and storing information has led to an explosion of data collection. Many organizations like the Census, hospitals and even search engine companies collect, analyze and distribute personal information in return for useful services. However, the collected data track entire public and private lives of individuals, thus resulting in an immense privacy risk of unauthorized disclosure. This dissertation presents novel conceptual and practical tools to ensure privacy of individuals while enabling the dissemination of valuable data about humans to improve their lives. Our contributions include novel formal definitions of the privacy risk arising from unauthorized disclosure, and practical algorithms for enforcing these definitions of privacy.

We consider two distinct settings of data dissemination that require different notions of privacy. In the first part of this dissertation, we consider a setting where no sensitive information should be disclosed. We consider the problem of deciding whether answering a query on a relational database leads to any disclosure of sensitive information. This problem was shown to be intractable; we propose practical algorithms for a reasonably large set of query classes.

In the second part of the dissertation, we consider the problem of publishing “anonymous” aggregate information about populations of individuals while preserving the privacy of individual-specific information. We present a novel framework for reasoning about the privacy risk in this setting. We also propose the first formal privacy definition and practical algorithms for pub-

lishing “anonymous” data that provably guarantees privacy of the individuals contributing to the data while releasing useful aggregate information. We also present a case study of applying formal privacy definitions to a real Census data publishing application.

BIOGRAPHICAL SKETCH

Ashwin Machanavajjhala was born in Kakinada, A.P., India and was raised in Chennai, India. He received a Bachelor of Technology degree from the Indian Institute of Technology, Madras in 2002. He received an M.S. in Computer Science in 2007 and a Ph.D. in Computer Science in 2008, from Cornell University. Beginning September 2008, he will be a Research Scientist at Yahoo! Research in Sunnyvale, California.

To my parents Bhasker and Seshamani.

ACKNOWLEDGEMENTS

This work would not have been possible but for the able guidance of Johannes Gehrke. He is a brilliant researcher and an extremely gifted teacher who has instilled in me a great passion for research in databases and taught me the art of choosing the right research problems. He has been a patient and generous mentor, and I will be ever grateful for time he spent working closely with me to mould my research career. I hope I can maintain his high standards for research in my future endeavours.

I thank Jayavel Shanmugasundaram for his ongoing support and guidance in my research. I am grateful to Joseph Halpern and Eva Tardos for always having time for insightful conversations. I thank Alan Demers for participating on my committee. I am also grateful to C. Pandu Rangan and K. Srinathan for first sparking off my interest in computer science research.

My work has greatly benefited from many fruitful collaborations. I am especially grateful to Daniel Kifer and David Martin for a wonderful collaboration, and for our exciting and educational research discussions. I also thank Adina Crainiceanu, Muthu Venkitasubramaniam and Michaela Goetz for collaborating with me on some exciting research during the course of my Ph.D. I thank my friends and faculty at the Department of Computer Science, Cornell University, for providing an excellent research atmosphere. I also am grateful for my external research collaborations. I thank John Abowd and Lars Vilhuber from the School of Industrial and Labour Relations, Cornell University for allowing me to work with real Census data. I thank Rakesh Agrawal, Ramakrishnan Srikant, Alexandre Evfimievski and Phokion Kolaitis for enriching my research internship experience at IBM Almaden Research Center, and Raghu Ramakrishnan, Phil Bohannon, Srujana Merugu, Minos Garofalakis, Erik Vee and Pedro

DeRose for the wonderful internship experience at Yahoo! Research.

Much credit for my success goes to my friends and family. A special thanks to Karthik Raghupathy, Parvathinathan Venkitasubramaniam, Prakash Linga, Smita Shankar, Vivek Vishnumurthy and Anand Pappu for those long and enlightening debates on life and Ph.D. I am very grateful for the unending support of my parents Seshamani and Bhasker, and my sister Swetha. Finally, I am forever indebted to my wonderful wife Lavanya for making every day of my life special, for her unflinching support through my triumphs and failures, and for all her help and insightful comments on my dissertation.

TABLE OF CONTENTS

Biographical Sketch	iii
Dedication	iv
Acknowledgements	v
Table of Contents	vii
List of Tables	ix
List of Figures	x
1 Introduction	1
1.1 Problem Setting	2
1.2 What is Privacy?	5
1.3 Overview of Database Privacy Solutions	6
1.3.1 Access Control in Databases	7
1.3.2 Anonymous Data Publishing	11
1.4 Contributions and Organization	15
2 On the Efficiency of Enforcing Perfectly Private Access Control	19
2.1 Introduction	19
2.2 Overview	22
2.2.1 Preliminaries	22
2.2.2 Intuition	26
2.3 Critical Tuples	32
2.4 Perfect Privacy	35
2.4.1 Perfect Privacy with Independent Tuples	35
2.4.2 Perfect Privacy with Key Constraints	53
2.5 Finding Tractable Cases	55
2.5.1 Using Tractability of $Q_1 \subseteq Q_2$	56
2.5.2 Using Tractability of $Q _{(G, \hat{G})} \subseteq Q$	62
2.5.3 Queries from Different Classes	64
2.6 Summary	65
3 ℓ-Diversity: Privacy beyond k-Anonymity	66
3.1 Introduction	66
3.1.1 Attacks On k -Anonymity	69
3.1.2 Contributions and Chapter Outline	71
3.2 Model and Notation	72
3.3 Bayes-Optimal Privacy	76
3.3.1 Changes in Belief Due to Data Publishing	77
3.3.2 Privacy Principles	84
3.3.3 Limitations of the Bayes-Optimal Privacy	86
3.4 ℓ -Diversity: Privacy beyond k -Anonymity	87
3.4.1 The ℓ -Diversity Principle	87
3.4.2 ℓ -Diversity: Instantiations	93

3.4.3	Multiple Sensitive Attributes	99
3.4.4	Discussion	100
3.5	Implementing Anonymous Data Publishing	101
3.6	Experiments	107
3.7	A Formal Framework for Defining Privacy	123
3.8	Summary	134
4	From Theory to Practice on the Map	135
4.1	Introduction	135
4.1.1	Contributions and Chapter Outline	136
4.2	Starting Point: Privacy Definition	138
4.3	Starting Point: Anonymization Algorithm	142
4.4	Revising the Privacy Definition	146
4.4.1	Analysis of Privacy	146
4.4.2	(ϵ, δ) -Probabilistic Differential Privacy: Ignoring Unlikely Privacy Breaches	149
4.4.3	(ϵ, δ) -Probabilistic Differential Privacy for Synthetic Data	152
4.5	Revising the Algorithm	160
4.5.1	Accept/Reject	161
4.5.2	Shrinking the Domain	164
4.6	Experiments	169
4.7	Summary	174
5	Related Work	175
5.1	Trusted Data Collector	175
5.1.1	Access Control	176
5.1.2	Anonymous Data Publishing	180
5.1.3	Statistical Databases	187
5.1.4	Private Collaborative Computation	189
5.2	Untrusted Data Collector	191
6	Summary and Discussion	193
6.1	Summary of Contributions	193
6.2	Future Directions	195
	Bibliography	199
A	Proof of Theorem 4.5	210

LIST OF TABLES

1.1	Inpatient Medical Records relation.	3
1.2	View of the Inpatient Medical Records relation containing records of patients from Ithaca, NY.	4
2.1	Size of the instance (Q_S, Q_V)	32
3.1	Inpatient microdata.	67
3.2	4-Anonymous Inpatient microdata.	68
3.3	Notation used in the proof of Theorem 3.1.	79
3.4	3-Diverse Inpatient microdata.	92
3.5	Table T for proof of Theorem 3.2.	103
3.6	Table T^* for proof of Theorem 3.2.	103
3.7	Description of Adult database.	107
3.8	Description of Lands End database.	108
3.9	Effect of homogeneity attack on the Adult database.	109
3.10	Effect of homogeneity attack on the Lands End database.	110
3.11	Effect of 95% homogeneity attack on the Adult database.	111
3.12	Effect of 95% homogeneity attack on the Lands End database.	112
4.1	Noise required per block $(\alpha(d)_i)$ to ensure privacy of Algorithm 3.	155

LIST OF FIGURES

1.1	Data Publishing	12
1.2	Statistical Databases	13
2.1	Relational schema of Hospital database.	20
2.2	A database instance $I \in \mathcal{I}$ satisfying the relation schema of the Hospital database.	27
2.3	Query classes for which checking perfect privacy is tractable. . .	29
2.4	Queries Q_S and Q_V from Example 2.3	47
2.5	I_S, I_V , and Queries $Q_S _{(\mathcal{G}_S, \mathcal{G}^*)}, Q_V _{(\mathcal{G}_V, \mathcal{G}^*)}$ for Example 2.3	48
2.6	Query Q_S and its tableau representation.	59
2.7	A tableau Q ; a subgoal \hat{G} compatible with the first two rows \mathcal{G} ; and the query $Q _{(\mathcal{G}, \hat{G})}$	60
3.1	Performance of ℓ -diverse data publishing on Adult database. . .	113
3.2	Performance of ℓ -diverse data publishing on Lands End database. .	113
3.3	Comparing generalization height, minimum average group size, and discernibility of k -anonymous and ℓ -diverse versions of a sample of the Adult database. $Q = \{\text{age, gender, race}\}$	116
3.4	Comparing generalization height, minimum average group size, and discernibility of k -anonymous and ℓ -diverse versions of a sample of the Adult database. $Q = \{\text{age, gender, race, marital_status}\}$	117
3.5	Comparing generalization height, minimum average group size, and discernibility of k -anonymous and ℓ -diverse versions of a sample of the Adult database. $Q = \{\text{age, gender, race, marital_status, education}\}$	118
3.6	Comparing generalization height, minimum average group size, and discernibility of k -anonymous and ℓ -diverse versions of a sample of the Adult database. $Q = \{\text{age, gender, race, marital_status, education, work_class, native_country}\}$	119
3.7	Comparing KL-Divergence to k -anonymous and ℓ -diverse versions of a sample of the Adult database. From top to bottom, $Q = \{\text{Age, Gender, Race}\}, \{\text{Age, Gender, Marital Status, Race}\}$ and $\{\text{Age, Education, Gender, Marital Status, Race}\}$ respectively.	121
3.8	Comparing KL-Divergence to k -anonymous and ℓ -diverse versions of the Adult database. From top to bottom, $Q = \{\text{Age, Gender, Race}\}, \{\text{Age, Gender, Marital Status, Race}\}$ and $\{\text{Age, Education, Gender, Marital Status, Race}\}$ respectively.	122
4.1	Average commute distance in the synthetic data vs # primary job holders ($\epsilon' = 8.6, \delta = 10^{-5}, k = 233, 726$).	171
4.2	Average commute distance in the synthetic data vs # primary job holders ($\epsilon' = 8.6, \delta = 10^{-5}, k = 233, 726$) for short commutes. . . .	171

4.3	Average commute distance in the synthetic data vs # primary job holders ($\epsilon' = 8.6, \delta = 10^{-5}, k = 120, 690$).	172
4.4	Average commute distance in the synthetic data vs # primary job holders ($\epsilon' = 8.6, \delta = 10^{-5}, k = 120, 690$) for short commutes.	173

CHAPTER 1

INTRODUCTION

Knowledge will forever govern ignorance, and a people who mean to be their own governors, must arm themselves with the power knowledge gives. A popular government without popular information or the means of acquiring it, is but a prologue to a farce or a tragedy or perhaps both.

– James Madison

One of the greatest triumphs of computer science is the unprecedented availability of information. This has been made possible by breakthroughs in hardware, faster processing speeds, massive storage capabilities and reliable communication, coupled with rapid advances in information retrieval, data management and data integration. These capabilities, in turn, have revolutionized the way people work, communicate, gather information, and even shop. However, these advances have also led to an explosion in data collection. Almost every action performed by individuals is being electronically logged – the news they read, the clothes they buy, the friends they meet, the doctors they visit and even the food they eat. Individuals benefit immensely by allowing such data collection. Critical services like health-care have markedly improved with the digitization of medical records. Governments around the world base their economic policies on demographic information collected by Census Bureaus. Search engine and web service companies track user actions in return for personalized web services. However, as a side-effect, these electronic logs track the entire public and private lives of individuals, thus posing an immense *privacy risk* of unauthorized disclosure [1, 65]! For instance, in October 2004, Choicepoint inadvertently released private financial information of 145,000 individ-

uals to criminals operating a scam [1]. In August 2006, the reputed Internet company America OnLine (AOL) released anonymous logs of search queries collected from users to aid information retrieval research in academia. However, identities of individuals were easily uncovered¹, revealing their private lives to millions.

Privacy is globally recognized as a universal value. We as computer scientists have the power and the responsibility to create tools that ensure this integral part of human dignity while enabling the dissemination of valuable data about humans for improving the quality of their lives. This dissertation proposes techniques for collecting and sharing personal information that provably guarantee privacy, and is a step towards fulfilling this responsibility.

1.1 Problem Setting

We introduce our problem setting via the following example. A centralized *trusted* data collection agency, say Gotham City Hospital, collects information from a set of patients. The information collected for each patient includes identifying information like name, address and social security number, demographic information like age, race and gender, and personal information like symptoms and medical condition. There are a number of *users* of this data. For instance, Dr. Drake requires access to the medical histories of all his patients. Nurse Natalia needs access to the drugs prescribed to the patients in her care. Finally, researcher Riddler, who studies how diseases correlate with age and gender, may require age, gender and medical condition information from all patients.

¹www.washingtonpost.com/wp-dyn/content/article/2006/08/07/AR2006080700790.html

Table 1.1: Inpatient Medical Records relation.

Name	Zip Code	Age	Nationality	Condition	Ward #
Alex	13053	28	Russian	Heart Disease	2323
Bruce	13068	29	American	Heart Disease	2323
Umeko	13068	21	Japanese	Viral Infection	1212
Dick	13053	23	American	Viral Infection	1212
Eshwar	14853	50	Indian	Cancer	1212
Fox	14853	55	Russian	Heart Disease	1212
Gray	14850	47	American	Viral Infection	2323
Harry	14850	49	American	Viral Infection	2323
Igor	13053	31	American	Cancer	2323
Jay	13053	37	Indian	Cancer	2323
Ken	13068	36	Japanese	Cancer	1212
Lewis	13068	35	American	Cancer	1212

Over the last half a century many database systems have been built that can efficiently store, index and process the aforementioned queries on large volumes of data. Throughout this dissertation we will assume that the data is stored in a relational database system [103]. In a relational database, the collected data is stored in one or more *relations* or tables. Table 1.1 is an example of a relation. Each row in this relation, also called a *tuple*, contains the medical records of a unique patient in the hospital. Each column in the table represents different *attributes* of each patient. The value of attribute A for tuple t is denoted by $t[A]$ and is called a *cell*. Users request access to parts of the data by formulating *queries* over the data. We refer to the result of a query, which is also a relation, as a *view* of the database. For instance, Table 1.2 is the view of the database

Table 1.2: View of the Inpatient Medical Records relation containing records of patients from Ithaca, NY.

Name	Zip Code	Age	Nationality	Condition	Ward #
Eshwar	14853	50	Indian	Cancer	1212
Fox	14853	55	Russian	Heart Disease	1212
Gray	14850	47	American	Viral Infection	2323
Harry	14850	49	American	Viral Infection	2323

corresponding to the query “list all the information about patients from Ithaca NY (ZipCode = 14850 or 14853)”.

While we know how to efficiently answer queries using relational databases, with the increasing concerns about privacy, organizations like Gotham City Hospital face a conflict between a patient’s right to privacy and the hospital’s need to know and process relevant information. While patients may trust the hospital’s database systems, they may not trust the users who view the collected information. For instance, nurse Natalia, who only tends to the patients in ward #1212, need not be allowed to access the medical records of patient Bruce admitted to ward #2323. Moreover, though Bruce may want researchers to study the correlation between symptoms and diseases, he may not want them to be able to precisely infer what he is suffering from. Hence, we are interested in designing *privacy-aware database systems* that (i) allow legitimate users to efficiently query the data, while (ii) provably guaranteeing the privacy of the individuals who contribute to the data.

1.2 What is Privacy?

Few values so fundamental to society have been left so undefined in social theory or have been the subject of such vague and confused writing by social scientists.

– Alan Westin

In order to design a privacy-aware database system we first need to prescribe to some reasonable notion of privacy. The term “privacy” is often used in daily conversation, as well as in philosophical, political and legal discussions; yet there is no single definition or meaning for the term [44]. The first discussions of information privacy are accredited to the famous essay by Warren and Brandeis [126] who recognized privacy as the “right to be let alone”, and laid the foundation for a concept of privacy that has come to be known as *control over information about oneself*. In this dissertation, we subscribe to Alan Westin who describes privacy as “*the ability to determine for ourselves when, how, and to what extent information about us is communicated to others*” [127]. While this definition is a good starting point, we still need to adapt it to the context of databases, and formalize, for instance, “to what extent” information that is released to a user discloses sensitive information about an individual contributing to the data. Proposing formal definitions for privacy in the context of database systems will be a focus of this dissertation.

1.3 Overview of Database Privacy Solutions

A wide range of technological solutions are required to implement privacy-aware database systems. Privacy breaches occur in databases due to four main reasons [1]:

- *Lack of Physical Security*: Most privacy breaches occur due to the theft of physical storage devices like hard drives, laptops and backup tapes.
- *Intrusion by Hackers*: Poor system security and wrong network administration render the data vulnerable to malicious hackers. For instance, in 2007, U.S. Secret Service agents found credit card numbers of Ralph Polo Lauren customers in the hands of Eastern European cyber thieves who created high-quality counterfeit credit cards.
- *Dishonest Insiders*: Dishonest or disgruntled employees of a data collection agency may disclose personal information. For instance, a woman in Pakistan hired by the University of California San Francisco Medical Center for transcribing medical records threatened to post patients' confidential files on the Internet unless she was paid more money [2].
- *Incorrect Privacy Practices*: Supposedly privacy-aware databases can lead to egregious breaches of privacy if they are based on unsound principles. The AOL data release fiasco described earlier in this chapter is one glaring example.

The first three threats may be solved by employing technology like encryption, firewalls and intrusion detection systems, educating administrators, and improving legislation and government regulation.

In this dissertation, we focus on the fourth “threat”, namely that of information sharing by data collection agencies using unsound privacy practices. We consider two standard techniques for implementing privacy-aware databases, namely, *access control* and *anonymous data publishing*, and show that current implementations can cause unintentional breaches of sensitive information. We show that this is due to the ad-hoc notions of privacy used to implement these systems. Hence, this dissertation deals with *formally understanding and defining privacy for both access control and anonymous data publishing, developing efficient algorithms to enforce these privacy notions, and evaluating them on real-world applications*. Before we enumerate the contributions of this dissertation in more detail (Section 1.4), we describe the current state of the art in access control and anonymous data publishing.

1.3.1 Access Control in Databases

Access control is the traditional solution that allows *controlled access* to a set of resources. Access control typically includes three parts – *authentication*, *authorization* and *audit*. Authentication refers to the process of establishing the identity of a user. Authorization is the process of verifying that an authenticated user has the privileges to perform certain operations on the system. Finally, audit is the process of verifying *post facto* that all the operations on the system have been performed by authorized users. Though authentication and audit are very important problems, they are not the focus of this dissertation.

Any authorization system can be modeled as a set of *accessors* (e.g., users), a set of *resources* (e.g., files, relations or views), an *access control policy* or a set of

rules describing whether an accessor is authorized to access a resource (*positive authorization*) or not (*negative authorization*), and an *enforcement algorithm* that can efficiently decide whether a request to access a resource can be allowed based on the access control policy. Authorization systems also contain an administrative mechanism to *grant* and *revoke* authorizations.

The simplest kind of an access control system is usually seen in file systems. The files in the file system correspond to the resources, and the users of the file system are the accessors. The access control policy is described in the form of an *access control list* (ACL), where every file f is associated with the list of users $ACL(f)$ who are authorized to access it. The permissions to access a file are usually granted by a file system administrator or by owners of the individual files. Given a request from user u for file f , the enforcement algorithm allows u to access f if and only if $u \in ACL(f)$.

In the context of databases, however, such a simple policy does not suffice since database systems cannot afford to grant authorizations at the granularity of files or whole relations. In a hospital, for instance, a single *Patient* relation can contain the information about thousands of patients. For each patient, in turn, the *Patient* relation contains a variety of attributes. Hence, the hospital might want to specify more fine grained access control policies on this relation. For instance, the hospital may want nurse Natalia to only access the names and medical conditions of patients in ward #1212.

One simple solution to implement such *fine-grained access control* is to maintain access control lists for every cell in the relation. However, in such a naive solution, the number of authorizations in the access control policy is inordinately large – the number of cells in the relation times the number of users in the

system. This problem becomes even more daunting in the case of *discretionary access control* (DAC), where the user who creates a relation should specify an ACL for each cell.

This challenge is usually solved by specifying authorizations over groups of users and groups of cells. Rows (tuples) and columns (attributes) are a natural choice for grouping cells. More generally, arbitrary groups of cells can be described using views over the relation. Hence, in DAC based database systems, authorizations are granted by the owner of a relation to either the whole relation or to some derived view over the relation. For instance, Dr. Drake may create a *Drake-Patient* view relation containing the medical records of his patients. Dr. Drake may also create *Drake-Patient-Prescription* relation containing the names of his patients and the medication prescribed to them. Using DAC, Dr. Drake can authorize nurse Natalia to access both the above relations.

Users are typically grouped in two ways, leading to two different kinds of access control methods. In *mandatory access control* (MAC) [87], each resource (relation, view, row, column or an individual cell) is labeled by a security level, each accessor is associated with a security clearance, and the authorizations are granted to (level, clearance) pairs. In military settings, for instance, security levels and clearances both are one of $\{Unclassified, Classified, Secret \text{ and } Top\ Secret\}$. These labels are ordered as

$$Unclassified < Classified < Secret < Top\ Secret$$

Access is granted according to the Bell-LaPadula rules [22]: a user can read any data item that has a security level lower than her clearance, but write to only those data items that have a higher security level.

In *role based access control* (RBAC) [23], authorizations are based on the role

in which an accessor is accessing information, rather than her identity. Positive and negative authorizations are defined per role to resources that are either complete relations, views, or in some cases individual cells in the table. For instance, Dr. Drake may act in two roles both as a physician treating patients and as a researcher correlating symptoms to new diseases. Dr. Drake is allowed/disallowed access to very different kinds of information in these two roles. As a physician, Dr. Drake may be allowed to view the complete medical records of his patients, but as a researcher, he may not be able to access information that is not necessary for the purposes of research.

However, allowing authorization over cells and arbitrary views over base relations introduces the following problem. Unlike in file systems, where access to one file does not usually disclose information about other files, in databases, access to one view over a relation can disclose information about other negatively authorized views in the following two ways.

First, the answers to two queries might depend on each other if they access some common cell in the relation while computing the corresponding views. For instance, consider a user u who is not allowed to access the view “the names of Dr. Drake’s patients” (V_1), and requests access to “the names of patients in ward #1212” (V_2). Can we grant u access to the view V_2 ? Even though there may be no correlation between being in ward #1212 and being treated by Dr. Drake, user u can get information about V_1 from V_2 as follows. User u is allowed to access the view “the names and ward numbers of all the patients who are not treated by Dr. Drake” (V_3). Hence, user u can find out the names of patients who are treated by Dr. Drake and in ward #1212, if any, using $V_3 - V_2$. Therefore, user u should not be allowed to access V_2 .

Second, the problem becomes trickier in the presence of *inference channels* [25]. Inference channels occur in databases either due to integrity constraints like functional and multivalued dependencies [103], or due to known correlations between attributes. Consider the following example of a functional dependency in the *Patient* relation. A functional dependency $A \rightarrow B$ occurs in a relation R when for every two tuples t_1, t_2 in R , if $t_1[A] = t_2[A]$ then $t_1[B] = t_2[B]$. For instance, $Doctor \rightarrow Ward$ is a functional dependency if all the patients treated by a single doctor are housed in the same ward. Hence, if the patients of Dr. Drake are housed in ward #1212, access to V_2 by user u described above leaks all the information about V_1 .

In summary, database authorizations differ from traditional access control lists because access to one view in a database may disclose information about other views. Hence, database authorizations need to precisely define the semantics of what information can be accessed, given positive or negative authorizations; we call this a *privacy definition*. The first part of this dissertation (Chapter 2) deals with the study of formal privacy definitions for negative authorizations in access control.

1.3.2 Anonymous Data Publishing

Many organizations like hospitals, the Census and, more recently, search engine companies collect information from individuals and expose this information to the public with the goal of furthering medical, economic and information retrieval research, respectively. Hence, these organizations wish to provide researchers access to interesting aggregate statistical information about the per-

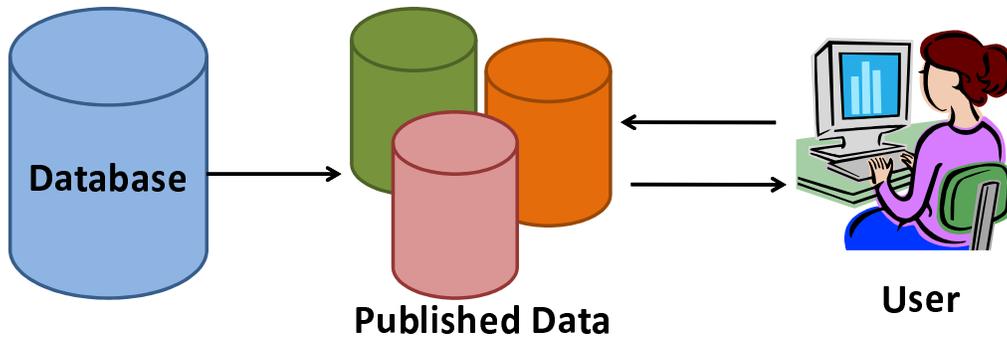


Figure 1.1: Data Publishing

sonal information they collect, such that researchers can not discover sensitive information about specific individuals from the data.

A number of techniques have been proposed for disseminating aggregate information about individuals (see Chapter 5), and most of them fall under one of two broad categories, *data publishing* and *statistical databases*. Figures 1.1 and 1.2 illustrate the main difference between the two classes of techniques. Statistical databases host a query answering service that evaluates queries posed by the researcher on the original data, and then decides to perform one of the following actions based on a privacy policy and the history of queries answered – (i) return the correct answer, (ii) deny an answer, or (iii) return a perturbed answer. In contrast, the data publishing class of techniques publish one or more views of the data that preserve statistical information about the original data. The researchers are allowed to execute arbitrary queries on these published views.

While both these classes of techniques have their advantages and disadvantages, many organizations prefer data publishing due to the following reasons:

- Data publishing requires only a one time effort in creating the published views. Thereafter, this data can be distributed to researchers, and they can

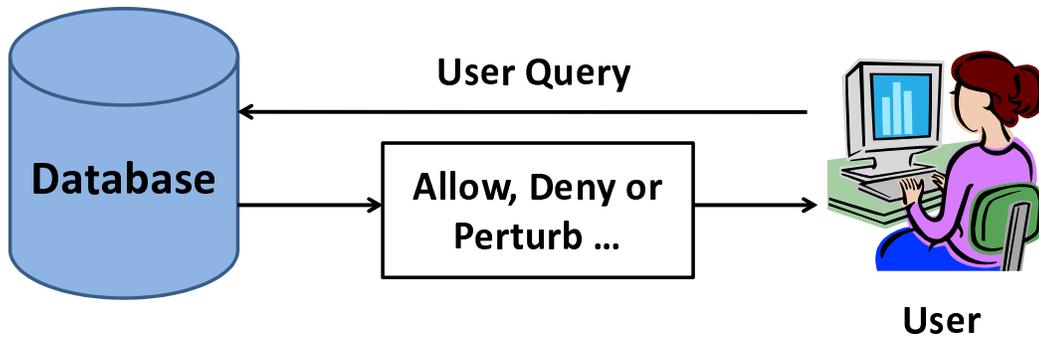


Figure 1.2: Statistical Databases

be allowed to run arbitrary queries on the published data. On the other hand, statistical databases require online server support to monitor user queries, and to evaluate how to answer those queries.

- Researchers usually need access to the data in order to determine how to analyze the data and may not be able to pose queries oblivious to the data.

Data collection agencies strive to publish data that disclose as much aggregate statistical information as possible, while preserving the privacy of the individuals contributing to the data. One popular technique, and a focus of this dissertation, is publishing *anonymous microdata*. Organizations like the Census distribute personal data in its raw, unaggregated form, also known as microdata, after removing attributes like name and social security number that are known to uniquely identify individuals. Anonymous microdata is particularly attractive to researchers since it retains all the statistical information about the original data, and because the data is in a form that can be readily input into existing statistical analysis algorithms.

However, individual-specific information can still be inferred from microdata even after removing identifying attributes as suggested in following excerpt [97],

Some deficiencies inevitably crop up even in the Census Bureau. In 1963, for example, it reportedly provided the American Medical Association with a statistical list of one hundred and eight-eight doctors residing in Illinois. The list was broken down into more than two dozen categories, and each category was further subdivided by medical specialty and area residence; as a result, identification of individual doctors was possible

In fact, a recent study conducted by Sweeney [121] estimated that 87% of the population of the United States could be uniquely identified using a combination of seemingly innocuous attributes such as gender, date of birth, and 5-digit zip code. In fact, those three attributes were used to link the voter registration records of Massachusetts (which included the name, gender, zip code, and date of birth) to supposedly anonymized medical data from GIC² (which included gender, zip code, date of birth and diagnosis). This “linking attack”, also called a *re-identification attack* managed to uniquely identify the medical records of the governor of Massachusetts in the medical data [122]. Sets of attributes (like gender, date of birth, and zip code in the example above) that can be linked with external data to uniquely identify individuals in the population are called *quasi-identifiers*.

Hence, there has been over a half a century of research primarily dedicated to developing algorithms and notions of privacy that disallow re-identification attacks [64, 128]. More recently, Samarati and Sweeney proposed a definition of privacy called *k*-anonymity [113, 122] to counter linking attacks using quasi-identifiers. A table satisfies *k*-anonymity if every record in the table is indistinguishable from at least $k - 1$ other records with respect to every set of quasi-

²Group Insurance Company (GIC) is responsible for purchasing health insurance for Massachusetts state employees.

identifier attributes. Such a table is called a k -anonymous table. Hence, for every combination of values of the quasi-identifiers in the k -anonymous table, there are at least k records that share those values. This ensures that individuals cannot be uniquely identified by linking attacks. k -anonymity has grown in popularity because of its conceptual simplicity and due to algorithmic advances in creating k -anonymous versions of a dataset [72, 18, 94, 6, 7, 81, 82, 83].

Though k -anonymity disallows re-identification attacks, we have no reason to believe that there are no other attacks that can disclose personal information about individuals. So, we asked the following question, “*Does k -anonymity really guarantee privacy?*” Interestingly, we show in this dissertation that the answer to this question is *no*. In Chapter 3, we illustrate using simple attacks that despite being k -anonymous a published dataset can still disclose sensitive information about individuals. Motivated by these attacks, in Chapters 3 and 4 we initiate a formal study of potential attacks, privacy definitions and new algorithms for publishing anonymous microdata that provably guarantee privacy.

1.4 Contributions and Organization

In this dissertation we propose novel formal definitions of privacy and develop efficient algorithms that provably guarantee privacy.

The first part of this dissertation (Chapter 2) deals with enforcing formal privacy definitions in access control. As we noted before, current access control systems do not associate formal semantics with negative authorizations over arbitrary views causing unwanted disclosure of sensitive information. We remedy this situation by adopting a formal privacy definition for negative autho-

rizations on views. Recently, Miklau et al. [96] proposed *perfect privacy*, a particularly attractive semantics for deciding whether the answer to a query Q_V discloses information about the answer to a secret query Q_S . Perfect privacy requires that query Q_V is truthfully answered by the database if and only if it discloses *no information* about the answer to the secret query Q_S . This ensures that the user’s belief about the answer to Q_S does not change even on observing the answer to the query Q_V . In this dissertation we propose adopting perfect privacy as *the* semantics for negative authorizations. Not only does it provide a theoretically sound semantics for negative authorizations, it also solves the problem of inference channels by enforcing access control at the tuple level – either a user can access a complete tuple or cannot access any attribute of the tuple.

However, Miklau et al. showed that deciding whether a query Q_V leaks any information about the sensitive view Q_S is intractable (π_2^P -complete) when Q_S and Q_V belong to a class of queries called conjunctive queries.

In Chapter 2, we show why the problem of enforcing perfect privacy is hard, and, in the process, identify large sub-classes of conjunctive queries for which perfect privacy can be efficiently enforced. This work originally appeared in the proceedings of the Principles of Database Systems, 2006 [88].

In the second part of this dissertation (Chapters 3–4), we consider the problem of formal privacy definitions for publishing anonymous microdata. In Chapter 3, we show that k -anonymity does not guarantee privacy. We give examples of two simple, yet subtle attacks on a k -anonymous dataset that allow an attacker to identify individual records. First, we show that an attacker can discover the values of sensitive attributes when there is little diversity in those

sensitive attributes. Second, attackers often have background knowledge, and we show that k -anonymity does not guarantee privacy against attackers using background knowledge. We give a detailed analysis of these two attacks and propose a novel and powerful privacy definition called ℓ -**diversity** that defends against such attacks. In addition to building a formal foundation for ℓ -diversity, we show in an experimental evaluation that ℓ -diversity is practical and can be implemented efficiently. This work originally appeared in the proceedings of the International Conference of Data Engineering, 2006 [89], and subsequently appeared in the Transactions of Knowledge Discovery from Data, 2007 [91].

One of the strengths of ℓ -diversity is its formal methodology to reason about privacy. We conclude Chapter 3 by describing a general framework for privacy definitions. Under this framework, we observe that ℓ -diversity is but a point in a whole space of privacy definitions. We then briefly describe some of the other points in this space, including perfect privacy from Chapter 2.

One of the main reasons for the popularity of microdata is the belief that the data can be anonymized with very little loss in statistical information; this holds true if removing identifying attributes alone guarantees sufficient privacy. However, when microdata is required to satisfy stronger privacy criteria there is a substantial loss of statistical information. For instance, it has been shown that for sparse high-dimensional data (with a large number of attributes), even a weak condition like k -anonymity dramatically reduces the utility of the data [5]. Hence, any study proposing new privacy definitions is incomplete without a concurrent study of the loss of utility due to enforcing privacy.

Quantifying the utility of an anonymous dataset is a complex task. It depends on the application scenario and the kinds of queries or analyses a re-

searcher would like to run on the published data. In Chapter 4, we study the utility of anonymous microdata in the context of a real world application. We consider a real world Census Bureau application, called OnTheMap, that plots the commuting patterns of workers in the United States. The source data for this application collected by the U.S. Census Bureau cannot be used directly by the mapping program due to privacy concerns. Hence, the application uses synthetic data that statistically mimics the original data. However, there is no proof that this synthetically generated data guarantees privacy.

Chapter 4 chronicles the challenges we faced in applying theoretical privacy definitions to the above application. We find that while some existing definitions of privacy are inapplicable to our target application, others are too conservative and render the synthetic data useless by guarding against privacy breaches that are very unlikely. Moreover, the data in our target application is sparse, and none of the existing solutions are tailored to anonymize sparse data. We propose efficient algorithms to address both the above issues. This work originally appeared in the proceedings of the International Conference of Data Engineering, 2008 [90].

We conclude this dissertation with a detailed discussion of related work in Chapter 5, and a summary of results in Chapter 6.

CHAPTER 2
ON THE EFFICIENCY OF ENFORCING PERFECTLY PRIVATE ACCESS
CONTROL

2.1 Introduction

This chapter provides a theoretical study of the problem of enforcing access control in databases. Specifically, we study the following problem of enforcing negative authorizations: given a user who is not allowed to access the views defined by *secret queries* Q_{S_1}, Q_{S_2}, \dots , efficiently decide whether or not to permit access to a view V defined by the query Q_V .

In order to answer the above question, one needs to first specify which views V can be *safely* accessed given a secret query Q_S ; we call such a specification a *privacy definition*. Intuitively, it is safe to release V if it does not disclose any information about the answer to Q_S ; but how does one check whether a view V discloses information about another query? In fact, a view can disclose information about a query to various extents, often in very subtle ways, as illustrated in the following example.

Example 2.1 Consider a hospital which maintains patient records in two relations with schemata shown in Figure 2.1. Relation P stores patient identifiers and their names. Relation D stores patient identifiers, diagnosis, prescribed medication, and their ward numbers. Suppose nurse Natalia is not allowed to access the answer to the query “list the names of patients who have cancer” (Q_S). First, consider the view V_1 defined by the query “list the names of patients who do not have cancer”. Since no patient who has cancer will feature in V_1 , it seems reasonable to assume that V_1 does not leak any

Relation P		Relation D			
PID	PName	PID	Diagnosis	Medication	Ward #

Figure 2.1: Relational schema of Hospital database.

information about Q_S , and hence, it is safe to allow Natalia to access V_1 . Next consider the view V_2 defined by the query “list the names of patients in ward #1212 who have cancer”. The names presented in V_2 will be a subset of the answer to Q_S , and clearly, it is not safe to allow access to V_2 .

Finally, consider the view V_3 defined by the query “list the names of patients in ward #1212”. It is not clear whether V_3 discloses information about Q_S , since there is no reason to believe there is a correlation between patients who are in ward #1212 and the patients who have cancer. However, it is, in fact, not safe to allow access to V_3 , since V_3 leaks information about Q_S in the following subtle way. Since Natalia can access V_1 , by excluding the names of patients who do not have cancer (V_1) from V_3 she can deduce the names of the patients in ward #1212 who have cancer (i.e., $V_2 = V_1 - V_3$). We argued that it is not safe to release V_2 ; therefore, it is should not be safe to release V_3 either.

Hence, we need a formal privacy definition that specifies which views can be safely accessed without disclosing information about a secret query. One such privacy definition called *perfect privacy* was proposed by Miklau et al. [96]. Perfect privacy guarantees that a view V defined by the query Q_V is deemed safe to be accessed if and only if it does not disclose any information about the secret query Q_S . This ensures that the user’s (or adversary’s) belief about the answer to Q_S does not change on observing the answer to the query Q_V .

Perfect privacy has two attractive properties. First, perfect privacy requires queries and views to be specified using a class of queries called *conjunctive*

queries. The usage of conjunctive queries enables the policy writer to express complex access control policies. In Example 2.1, the secret query can be expressed by the following conjunctive query:

$$Q_S(name) \text{ :- } P(pid, name), D(pid, cancer, med, ward).$$

The view V_2 can be expressed by the following conjunctive query:

$$Q_S(name) \text{ :- } P(pid, name), D(pid, cancer, med, 1212).$$

As a more complex example, suppose the hospital is required to release as much data as possible to a pharmaceutical company without disclosing the names of patients diagnosed with both hepatitis and cancer. The secret query in this scenario can be expressed as the following conjunctive query:

$$Q_S(name) \text{ :- } P(pid, name), \\ D(pid, hepatitis, med_1, ward_1), D(pid, cancer, med_2, ward_2).$$

Second, perfect privacy has the following *collusion resistance* property. Consider again a secret query Q_S , and assume that there are n adversaries $\{A_1, \dots, A_n\}$, where adversary A_i requests access to the view defined by query Q_{V_i} . Perfect privacy ensures that each Q_{V_i} will be answered only if each query Q_{V_i} does not disclose any information about Q_S . Moreover, perfect privacy also ensures that the adversaries cannot learn any sensitive information by colluding with each other [96]. This collusion-resistance is also crucial for efficient query answering when the database answers queries interactively. When enforcing perfect privacy, the database does not need to keep track of all queries ever answered in order to guarantee that future query answers cannot be combined with old query answers to disclose information [76].

However, these nice properties of perfect privacy do not come for free. The problem of checking whether a conjunctive query Q_V discloses any information

about another conjunctive query Q_S is Π_2^p -complete [96], even when the conjunctive queries have only equality comparisons.

In this chapter, we show that there are interesting sub-classes of queries for which enforcing perfect privacy is tractable (i.e., there is an algorithm which runs in time polynomial in the size of the two queries Q_S and Q_V). We first propose an alternate characterization of perfect privacy in terms of the well studied notion of *query containment* [27]. We exploit this connection between perfect privacy and query containment to identify many subclasses of conjunctive queries (described in Figure 2.3) where checking perfect privacy is tractable.

The rest of this chapter is organized as follows. In Section 2.2, we introduce notation and give the intuition behind the ideas presented in this chapter. Section 2.3 gives an alternative characterization of critical tuples, a core property of perfect privacy. We use this new characterization to connect perfect privacy with query containment in Section 2.4. In Section 2.5 we use this connection to prove tractability results for checking perfect privacy for different query classes. We conclude by summarizing our results in Section 2.6.

2.2 Overview

2.2.1 Preliminaries

We start with some basic notation.

Relations and Queries: A *relation* R consists of a *relation schema* $R(A_1, \dots, A_r)$ and an associated relation instance which is a finite two-dimensional table with

r columns; we call r the *arity* of relation R . The columns are called *attributes* and the rows are called *tuples*. Attribute A_i has domain $D(A_i)$. We define D_R as the domain of tuples in the relation R , i.e., $D_R = D(A_1) \times \dots \times D(A_r)$. We assume that all the domains are finite. Each tuple t in the relation is an element $(c_1, \dots, c_r) \in D_R$ and is denoted by $t = R(c_1, \dots, c_r)$; we call R the *relational symbol* of t . We denote by $t[A_i]$ or short $t[i]$ the value of attribute A_i in tuple t .

A *database schema* S is a finite set of relational schemata $\{R_1, \dots, R_m\}$. $Tup(S)$ denotes the set of all tuples over all relations in the schema that can be formed from the constants in the respective domains. A *database instance* I is a subset of $Tup(S)$ and let \mathcal{I} denote the set of all database instances. We henceforth use $Tup(\mathcal{I})$ to mean $Tup(S)$. We assume that the only constraints on the database are key constraints. A *key* of a relation R is a minimal set of attributes \mathcal{A} such that for every two tuples $t_1, t_2 \in D_R$, $t_1[\mathcal{A}] = t_2[\mathcal{A}] \Rightarrow t_1 = t_2$. Let $\mathcal{I}^{\mathcal{K}}$ denote the set of all possible database instances satisfying the key constraints in \mathcal{K} . Let \mathcal{K}_R denote the set of key constraints on relation R .

A *conjunctive query with inequalities* Q has the form

$$Q(a_1, a_2, \dots, a_m) \quad :- \quad G_1 \wedge G_2 \wedge \dots \wedge G_n \wedge \mathcal{C}_Q;$$

where \mathcal{C}_Q is a set of inequality constraints.

We call G_ℓ a *subgoal*, $Q(a_1, a_2, \dots, a_m)$ the *goal*. Each subgoal G_ℓ is of the form $R_\ell(x_1, x_2, \dots, x_{k_\ell})$, where R_ℓ is a relation. Similar to a tuple, we call R_ℓ the *relational symbol of subgoal* G_ℓ . Each $x_i = G_\ell[i]$ is either a constant in $D(R_\ell.A_i)$ or a variable ranging over $D(R_\ell.A_i)$. Note that while each field in a tuple is a constant, a field in a subgoal can be a variable. Each a_i in the goal is either a variable appearing in one of the subgoals, or a constant in the domain of one of the attributes of a relation appearing in the query. We denote the set of variables

appearing in the goal by A_Q and we also refer to it as the set of *distinguished variables*. The set of variables which appear in the subgoals but not in the goal is denoted by B_Q and called the set of *non-distinguished variables*. Let K_Q denote the set of all constants appearing in the query. We abuse the notation Q to also denote the set of subgoals in Q ; i.e., $Q = \{G_1, G_2, \dots, G_n\}$. The inequality constraints in the set \mathcal{C}_Q are of the form $x_i \theta x_j$, where $\theta \in \{\leq, <, \geq, >, \neq\}$, x_i is a variable in $A_Q \cup B_Q$ and x_j is either a variable in $A_Q \cup B_Q$ or a constant in the domain that x_i ranges over.

Semantically, a query is a function from database instances to relation instances. We define the output of a query on a database instance using the idea of a *valuation*. A *valuation* ρ on a conjunctive query Q is a function that (i) maps variables in the query ($A_Q \cup B_Q$) to constants in the respective domains and (ii) is the identity function for constants in K_Q . A *constraint preserving valuation* ρ on Q is a valuation such that $(x_i \theta x_j) \in \mathcal{C}_Q \Rightarrow \rho(x_i) \theta \rho(x_j)$. Unless otherwise specified, all valuations in the remainder of the paper are constraint preserving. A subgoal is said to be *compatible* with a tuple t if the subgoal and the tuple have the same relational symbol and there is a valuation ρ mapping the subgoal to t . A set of subgoals $\{G_1, \dots, G_n\}$ is *compatible* if there is a tuple t that $\forall i \exists \rho : \rho$ is a valuation mapping G_i to t .

Given a conjunctive query with inequalities Q and a database instance I , a tuple $t^o = Q(c_1, c_2, \dots, c_m)$ is said to be in the output of $Q(I)$ if there is a valuation ρ such that $\rho(a_i) = c_i$, i.e., each distinguished variable a_i is mapped to a constant c_i ; and $\rho(R(x_1, \dots, x_r)) \in I$, i.e., subgoal $R(x_1, \dots, x_r)$ is mapped to one of the tuples in relation instance R of the database instance I .

Query Containment: Query Q_1 is said to be *contained* in query Q_2 on the set of

databases in \mathcal{I} and is denoted by $Q_1 \subseteq Q_2$, if

$$\forall I \in \mathcal{I}, Q_1(I) \subseteq Q_2(I).$$

Throughout the paper, we use the term *conjunctive query* to denote a *conjunctive query without inequalities*, unless otherwise specified.

Perfect Privacy [96]: We give a quick recap of the basic notions of perfect privacy from Miklau et al. [96]. Let P be a probability distribution over all the possible tuples $P : \text{Tup}(\mathcal{I}) \rightarrow [0, 1]$, where $P(t) = p$ denotes the probability that tuple t will occur in a database instance. Miklau et al. call the pair $(\text{Tup}(\mathcal{I}), P)$ a *dictionary*. Let $P(Q_S = s)$ denote the probability that the query Q_S outputs s . Then query Q_S is perfectly private with respect to Q_V if for every probability distribution and for all answers s, v to the queries Q_S, Q_V respectively,

$$P(Q_S = s) = P(Q_S = s \mid Q_V = v). \quad (2.1)$$

In other words, the query Q_S is perfectly private with respect to Q_V if the adversary's belief about the answer to Q_S does not change even after seeing the answer to Q_V .

When the only constraints in the database are key constraints, the above condition can be rephrased in terms of critical tuples. A tuple t is *critical* to a query Q , denoted by $t \in \text{crit}(Q)$, if $\exists I \in \mathcal{I}, Q(I \cup \{t\}) \neq Q(I)$. Let us now state one of the main results from Miklau et al. [96].

Theorem 2.1 (Perfect Privacy [96]) *Let $(\text{Tup}(\mathcal{I}), P)$ be a dictionary. Two queries Q_S and Q_V are perfectly private with respect to each other if and only if*

$$\text{crit}(Q_S) \cap \text{crit}(Q_V) = \emptyset.$$

Let $\mathcal{K} = \{\mathcal{A}_1, \dots, \mathcal{A}_m\}$ be a set of key constraints over the relations involved in Q_V and Q_S . Then query Q_V and Q_S are perfectly private with respect to each other if

$$\forall t_S \in \text{crit}(Q_S), t_V \in \text{crit}(Q_V), \forall \mathcal{A}_i \in \mathcal{K}, t_S[\mathcal{A}_i] \neq t_V[\mathcal{A}_i].$$

Miklau et al. also proved the following complexity result.

Theorem 2.2 (Intractability of Perfect Privacy [96]) *The problem of checking if a query Q_V is perfectly private with respect to query Q_S is Π_2^P -complete.*

2.2.2 Intuition

In this section we provide some intuition behind the rest of the chapter. We are interested in identifying subclasses of conjunctive queries which allow specification of interesting privacy policies while permitting tractable enforcement of perfect privacy. We first take a very simple class of queries – queries without self-joins – and illustrate that enforcing perfect privacy is indeed tractable. In order to determine the complexity of other query classes, we could have laboriously enumerated every interesting query class and investigated the hardness of checking perfect privacy for queries in that class. Instead, we show a connection between the problem of checking perfect privacy and the problem of query containment. We use this connection and the vast literature in query containment to identify interesting query classes.

Relation P		Relation D			
PID	PName	PID	Diagnosis	Medication	Ward #
123	John	123	Hepatitis	RX1	1212
135	Daisy	135	Cancer	RX2	2312
146	Chris	135	Hepatitis	RX1	1212

Figure 2.2: A database instance $I \in \mathcal{I}$ satisfying the relation schema of the Hospital database.

A Simple Tractable Case

Example 2.2 Consider the hospital database from Figure 2.1. Let the sensitive information be specified by the following query Q_S :

$$Q_S(\text{name}) \text{ :- } P(\text{pid}, \text{name}), D(\text{pid}, \text{cancer}, \text{med}_1, \text{ward}_1)$$

Query Q_S has no self joins, i.e., no two subgoals have the same relational symbol. Which tuples in the domain of relation D are critical to this query? Clearly, any tuple t having the diagnosis attribute valued cancer, for example $t = D(135, \text{cancer}, \text{RX2}, 2312)$, is critical to Q_S . In particular, for $t = D(135, \text{cancer}, \text{RX2}, 1212)$, the instance I in Figure 2.2 is such that $Q_S(I \cup \{t\}) \neq Q_S(I)$. Also, it is easy to see that any tuple not having cancer as the value for the diagnosis attribute, such as $t' = D(123, \text{hepatitis}, \text{RX1})$, is not critical to Q_S . For any instance $I' \in \mathcal{I}$, adding the tuple t' will not lead to a new output tuple being created since t' is not compatible with any subgoal. Hence, a tuple in the domain of relation D is critical to Q_S if and only if the value for the diagnosis attribute is cancer.

As illustrated in this example, if Q_S has no self-joins, a tuple t is critical to Q_S if and only if t is compatible with some subgoal in Q_S . Using this simple check for critical tuples, the condition for perfect privacy can be simplified as follows. Queries Q_S and Q_V , both without self-joins, share a critical tuple if and only if

there are subgoals $G_S \in Q_S$ and $G_V \in Q_V$ both of which are compatible with some common tuple $t \in D_R$. Recall that subgoals are compatible if they have the same relation symbol and if the subgoals do not have differing constants for any attribute. Checking this condition is linear in the size of the queries and hence the problem is tractable.

Connecting Privacy and Query Containment

Let us now give an intuition behind our alternate characterization of perfect privacy. Suppose a tuple t is critical to Q . By definition, there is some database instance I , such that $Q(I \cup \{t\}) \neq Q(I)$. Since Q is a conjunctive query, it is monotone, i.e., $Q(I) \subseteq Q(I \cup \{t\})$. Therefore, there is an output tuple t° , such that $t^\circ \in Q(I \cup \{t\})$ and $t^\circ \notin Q(I)$. Consider a valuation ρ on Q which creates this output tuple $t^\circ \in Q(I \cup \{t\})$. Recall that ρ should map Q 's goal to t° and the subgoals in Q to tuples in $(I \cup \{t\})$. Since, $t^\circ \notin Q(I)$, ρ cannot map all the subgoals in Q to tuples in I . Hence, ρ should map some set of subgoals from Q , say \mathcal{G} , to the critical tuple t .

Consider a query Q' which is constructed from Q by replacing all of the variables x_i appearing in Q by $\rho(x_i)$. Since a valuation is an identity function on constants, ρ is still a valuation from Q' to $(I \cup \{t\})$ which outputs the tuple t° . So we still have $t^\circ \in Q'(I \cup \{t\})$ and $t^\circ \notin Q(I)$.

Remove from Q' the subgoals in \mathcal{G} . Call this new query $Q|_{(\mathcal{G},t)}$ (see Section 2.3 for a formal definition). The valuation ρ now maps all the subgoals in $Q|_{(\mathcal{G},t)}$ to I and the goal of $Q|_{(\mathcal{G},t)}$ to t° . Thus we have, $t^\circ \in Q|_{(\mathcal{G},t)}(I)$ and $t^\circ \notin Q(I)$; i.e.,

Query Class	Efficiency	Query Class Description
Queries with no self-joins	$O(nr)$	Queries where each subgoal is associated with a unique relation symbol e.g., $Q(a_1, a_2) :- R(a_1, b_1, b_2), S(a_2, b_2), T(a_1, b_1, b_2)$ Complexity of $Q \subseteq Q' : O(nr)$, where Q has no self joins
Maximal Row Tableaux	$O(n^2r)$	Query Q represented as tableau. Each variable appears only in one column. No two rows r_1, r_2 sharing the same relation symbol are s.t. $\forall i, r_1[i] = c \Rightarrow r_2[i] = c$, for all $c \in K_Q$ e.g., $Q(a_1, a_2) :- R(a_1, b_1, 0), R(a_2, b_2, 1), T(2, b_2, b_3)$ Complexity of $Q \subseteq Q' : \text{NP-complete}$
Queries with $k \leq 1$	$O(n^3r)$	Queries with at most one self-join per relation e.g., $Q(a_1, a_2) :- R(a_1, b_1, b_2), R(a_2, b_1, b_3), S(a_1, b_2, b_3), S(a_2, b_1, b_3)$ Complexity of $Q \subseteq Q' : O(n^2r)$, where Q has at most 1 self join [115]
Simple Tableaux	$O(m2^{2k} \cdot n^3r^2)$	Query Q represented as tableau. Each variable appears only in one column. Each column containing a repeated non-distinguished variable b_i contains no other repeated symbol. e.g., $Q(a_1, a_2) :- R(a_1, b_2, b_3), R(a_2, b_2, b_4), R(a_1, b'_2, b_3), R(a_2, b''_2, b_5)$ Complexity of $Q \subseteq Q' : O(n^3r^2)$, where Q_1, Q_2 are simple tableaux queries [15]
Acyclic Queries	$O(m2^{2k} \cdot rn^2 \log n)$	The hypergraph constructed using the variables in the query Q as nodes and subgoals in the query as (hyper-)edges is acyclic. e.g., $Q(a_1, a_2) :- R(a_1, b_2, b_3), R(a_2, b_3, b_4), R(a_1, b_5, b_6)$ Complexity of $Q \subseteq Q' : O(rn^2 \log n)$, where Q is acyclic [30].
Queries with Bounded Query-Width	$O(m2^{2k} \cdot rc^2n^{c+1} \log n)$	The hypergraph constructed using the variables in the query Q as nodes and subgoals in the query as (hyper-)edges has a query-width bounded by c . e.g., ($c = 2$): $Q(a_1, a_2) :- R(a_1, b_3, b_1), R(a_2, b_4, b_1), R(a_1, b_5, b_2), R(a_2, b_6, b_2)$ Complexity of $Q \subseteq Q' : O(crn^{c+1} \log n)$, where Q has bounded tree-width [30]

Figure 2.3: Query classes for which checking perfect privacy is tractable.

$Q|_{(\mathcal{G},t)} \not\subseteq Q$. This gives the following condition:

$$t \in \text{crit}(Q) \Rightarrow \exists \mathcal{G} \text{ compatible with } t, Q|_{(\mathcal{G},t)} \not\subseteq Q \quad (2.2)$$

Interestingly, whenever \mathcal{G} is a set of subgoals compatible with t , the query $Q|_{(\mathcal{G},t)}$ is such that $Q|_{(\mathcal{G},t)}(I) \subseteq Q(I \cup \{t\})$ (see Section 2.3). This is because any valuation ρ from $Q|_{(\mathcal{G},t)}$ to I can be extended to a valuation ρ' from Q to $(I \cup \{t\})$, where the subgoals in \mathcal{G} are mapped to t and the other subgoals are mapped to tuples in I as in ρ .

When t is not critical to Q , for every instance I , $Q(I \cup \{t\}) = Q(I)$. Hence, for every instance I , $Q|_{(\mathcal{G},t)}(I) \subseteq Q(I)$. This gives us the condition

$$t \notin \text{crit}(Q) \Rightarrow \forall \mathcal{G} \text{ compatible with } t, Q|_{(\mathcal{G},t)} \subseteq Q \quad (2.3)$$

Using Statements 2.2 and 2.3, we can now characterize the perfect privacy condition using query containment:

$$\begin{aligned} Q_V \text{ is perfectly private wrt } Q_S & \\ \Leftrightarrow \text{crit}(Q_V) \cap \text{crit}(Q_S) &= \emptyset \\ \Leftrightarrow \forall t \in \text{Dup}(\mathcal{I}), t \notin \text{crit}(Q_V) \vee t \notin \text{crit}(Q_S) & \\ \Leftrightarrow \forall t \in \text{Dup}(\mathcal{I}), \forall \mathcal{G}_S, \mathcal{G}_V \text{ compatible with } t, & \\ Q_S|_{(\mathcal{G}_S,t)} \subseteq Q_S \vee Q_V|_{(\mathcal{G}_V,t)} \subseteq Q_V & \end{aligned}$$

Our final goal is to identify cases when the above check is tractable. This will, however, not be easy as long as the condition has a universal quantifier over all the tuples in the domain. We show that when attributes have sufficiently large domains, the universal quantifier over the domain of tuples can be eliminated. Given a set of subgoals \mathcal{G} , let G be a subgoal such that every tuple compatible

with G is also compatible with \mathcal{G} . Let $Q|_{(\mathcal{G},G)}$ be constructed analogous to $Q|_{(\mathcal{G},t)}$. We show, in Section 2.4, that if $Q|_{(\mathcal{G},G)} \not\subseteq Q$, then some tuple compatible with G is critical to Q . Using this property, we provide an alternate characterization for perfect privacy as follows. A query Q_V does not disclose any information about Q_S if and only if for every set of compatible subgoals \mathcal{G}_S in Q_S and \mathcal{G}_V in Q_V :

$$Q_S|_{(\mathcal{G}_S, G^*)} \subseteq Q_S \vee Q_V|_{(\mathcal{G}_V, G^*)} \subseteq Q_V,$$

where G^* is the most general unifier of the subgoals in $\mathcal{G}_S \cup \mathcal{G}_V$ (see Section 2.4).

Using the Connection

We can draw three key insights from this alternate definition. First, it is now intuitively clear why the problem of enforcing perfect privacy is in Π_2^P ; for every set of compatible subgoals (universal quantification over a finite number of choices) we perform a query containment check (which is in NP for conjunctive queries [27] and is in Π_2^P when queries have inequalities [124]). Next, we observe that if the maximum number of subgoals in a query which share the same relational symbol is bounded by a constant (alternately, the number of self-joins per relation is bounded), then the problem of enforcing perfect privacy is only as hard as performing a constant number of containment checks. Finally, the query containments that we check are not between two arbitrarily complex queries, and we can use their structure to propose efficient algorithms for enforcing perfect privacy even for query subclasses where query containment is intractable.

Table 2.3 enumerates our complexity results. The first column enumerates the subclasses of conjunctive queries where perfect privacy can be enforced

Table 2.1: Size of the instance (Q_S, Q_V) .

n	Total number of subgoals: $n = Q_S + Q_V $
m	Number of relations appearing in Q_S and Q_V
r	Max arity of relation appearing in Q_S and Q_V
k	Max number of self-joins per relation in either query

in time polynomial in the size of the queries. The second column gives the bound on the time required to enforce perfect privacy when Q_S and Q_V are from the specified query class; Table 2.1 explains our notation. Notice that the case of queries without self-joins is actually a special case of our framework. For maximal row tableaux, a subclass of tableaux queries, query containment is intractable. However, since the query containments we check are of a special form, we can show that perfect privacy can be enforced for these queries efficiently. Tractability of the rest of the query classes leverage tractability results for query containment [15, 30, 115]. When Q_S and Q_V are from different classes, the tractability result for the less efficient query class holds.

2.3 Critical Tuples

In this section we propose an alternate characterization of critical tuples for conjunctive queries in terms of query containment. In order to state the alternate characterization, we first describe how to construct $Q|_{(\mathcal{G}, t)}$ from a conjunctive query Q , a set of compatible subgoals \mathcal{G} in Q , and a tuple t which is compatible with all the subgoals in \mathcal{G} . The construct $Q|_{(\mathcal{G}, t)}$ has interesting properties (Lemmas 2.1 and 2.2) which we will exploit in the rest of the paper. Finally, we

re-characterize the critical tuple condition for conjunctive queries in terms of query containment (Theorem 2.3).

Denote by $Q - \mathcal{G}$ the query constructed by removing all the subgoals in \mathcal{G} from the query Q . Denote by $Q|_{(\mathcal{G},t)}$ the query $\rho_t(Q - \mathcal{G})$, where ρ_t is a *partial valuation* which maps the variables appearing in subgoals in \mathcal{G} to corresponding constants in tuple t , and is the identity function for all other variables in the query $Q - \mathcal{G}$. Formally ρ_t is defined as follows: Let V_Q denote the variables appearing in Q , and $V_{\mathcal{G}}$ denote the set of variable appearing in \mathcal{G} . Let K denote the set of all constants. Then, ρ_t is a function $\rho_t : K \cup V_Q \rightarrow K \cup (V_Q - V_{\mathcal{G}})$ such that,

$$\rho_t(x_i) = \begin{cases} t[j], & \text{if } G[j] = x_i, \text{ for some } G \in \mathcal{G} \\ x_i, & \text{otherwise} \end{cases}$$

$Q|_{(\mathcal{G},t)}$ has the following two properties.

Lemma 2.1 *Let Q be a conjunctive query and t be a tuple. For every set of subgoals \mathcal{G} in Q which are compatible with t ,*

$$\forall I, Q|_{(\mathcal{G},t)}(I) \subseteq Q(I \cup \{t\}).$$

Proof. Let ρ_t be the partial valuation that maps the subgoals \mathcal{G} to the tuple t , and is the identity function for all other variables and constants. $Q|_{(\mathcal{G},t)}$ is equivalent to $\rho_t(Q - \mathcal{G})$. Consider any valuation ρ that maps the subgoals of $Q|_{(\mathcal{G},t)}$ to tuples in the instance I and the goal to output tuple t^o . Since ρ_t maps subgoals in \mathcal{G} to t , $\rho \circ \rho_t$ is a valuation which maps the subgoals in Q to the tuples in $I \cup \{t\}$ outputting the same tuple t^o . Hence, every tuple in the output of $Q|_{(\mathcal{G},t)}(I)$ is also in the output of $Q(I \cup \{t\})$. ■

Lemma 2.2 *Let Q be a conjunctive query, t a tuple and I a database instance. Let ρ be a valuation mapping the subgoals in Q to tuples in $(I \cup \{t\})$. Let \mathcal{G} be the set of subgoals in Q that are mapped to t under ρ . If ρ maps the goal of Q to t° , then $t^\circ \in Q|_{(\mathcal{G},t)}(I)$.*

Proof. ρ is a valuation mapping the subgoals in Q to tuples in $I \cup \{t\}$. Since ρ maps the subgoals in \mathcal{G} to t , it is clear that ρ is a valuation from subgoals in $Q - \mathcal{G}$ to tuples in I . Hence, $t^\circ \in (Q - \mathcal{G})(I)$. Consider a valuation ρ' which maps variables appearing $Q|_{(\mathcal{G},t)}$ to constants as follows: $\rho'(x_i) = \rho(x_i)$. Also ρ_t coincides with ρ on the variables appearing in \mathcal{G} . Hence, ρ' maps the subgoals in $Q|_{(\mathcal{G},t)}$ to tuples in I , and the goal of $Q|_{(\mathcal{G},t)}$ to the same output tuple t° . Therefore, $t^\circ \in Q|_{(\mathcal{G},t)}(I)$. ■

We can now state our alternate characterization of critical tuples in terms of query containment.

Theorem 2.3 *A tuple t is critical to a conjunctive query Q if and only if there is some set of subgoals \mathcal{G} in Q which are compatible with t , such that $Q|_{(\mathcal{G},t)} \not\subseteq Q$.*

Proof. $t \notin \text{crit}(Q)$ if and only if for every instance I , $Q(I \cup \{t\}) = Q(I)$. From Lemma 2.1, for any set of subgoals $\mathcal{G} \subseteq Q$ which are compatible with t , for every instance I , $Q|_{(\mathcal{G},t)}(I) \subseteq Q(I \cup \{t\})$. Therefore,

$$t \notin \text{crit}(Q) \Rightarrow \forall \mathcal{G} \text{ compatible with } t, Q|_{(\mathcal{G},t)} \subseteq Q$$

To prove the other direction, let I be an instance such that $Q(I \cup \{t\}) \neq Q(I)$. Since Q is monotone, there is a tuple t° , such that $t^\circ \in Q(I \cup \{t\})$ and $t^\circ \notin Q(I)$. Hence, there is a valuation ρ which maps the goal of Q to t° and the subgoals of Q to tuples in $I \cup \{t\}$. Since $t^\circ \notin Q(I)$, ρ should map some set of subgoals \mathcal{G} in

Q to the tuple t (otherwise $t^o \in Q(I)$). We can now use Lemma 2.2 to show that $t^o \in Q|_{(\mathcal{G},t)}(I)$. Therefore,

$$t \in \text{crit}(Q) \Rightarrow \exists \mathcal{G} \text{ compatible with } t, Q|_{(\mathcal{G},t)} \not\subseteq Q$$

■

2.4 Perfect Privacy

In this section, we give an alternative characterization of perfect privacy in terms of query containment. In particular, we use our new characterization of critical tuples to give a novel characterization of perfect privacy in terms of query containment. Our new characterization, however, involves a universal quantifier over all the tuples in the domain, making this characterization very expensive to check. The major technical contribution of this section is to eliminate this universal quantification over all tuples (Theorem 2.5).

We first discuss the case where in every database instance in \mathcal{I} the tuples are independent of each other (Section 2.4.1). We also discuss the scenario with key constraints (Section 2.4.2).

2.4.1 Perfect Privacy with Independent Tuples

In the case when the tuples are independent of each other, a query Q_V does not disclose any information about the query Q_S if and only if no tuple is critical to both Q_S and Q_V [96]. Hence, given our novel characterization for critical tuples (Theorem 2.3), perfect privacy can be rephrased as

Input: Conjunctive Queries Q_S and Q_V .
Output: Whether Q_S and Q_V share a critical tuple.

- 1: //for every tuple in the domain
- 2: **for all** tuples $t \in Tup(\mathcal{I})$ **do**
- 3: //for every set of subgoals in the query
- 4: **for all** $\mathcal{G}_S \subseteq Q_S, \mathcal{G}_V \subseteq Q_V$ having the same relational symbol as t **do**
- 5: **if** $\mathcal{G}_S \cup \mathcal{G}_V$ is compatible with the tuple t **then**
- 6: //consider the partial valuation ρ_t mapping the
//subgoals in $\mathcal{G}_S \cup \mathcal{G}_V$ to the tuple t
- 7: $\rho_t : A_Q \cup B_Q \rightarrow A_Q \cup B_Q \cup K_Q$ such that

$$\rho_t(x_i) = \begin{cases} t[j], & \text{if } G[j] = x_i, \text{ for some } G \in \mathcal{G}_S \cup \mathcal{G}_V \\ x_i, & \text{otherwise} \end{cases}$$
- 8: $Q_S|_{(\mathcal{G}_S,t)} = \rho_t(Q_S - \mathcal{G}_S), Q_V|_{(\mathcal{G}_V,t)} = \rho_t(Q_V - \mathcal{G}_V)$.
- 9: **if** $(Q_S|_{(\mathcal{G}_S,t)} \not\subseteq Q_S \text{ AND } Q_V|_{(\mathcal{G}_V,t)} \not\subseteq Q_V)$ **then**
- 10: **return** Q_S and Q_V SHARE a critical tuple.
- 11: **end if**
- 12: **end if**
- 13: **end for**
- 14: **end for**
- 15: **return** Q_S and Q_V do NOT SHARE a critical tuple.

Algorithm 1: Perfect Privacy (naive).

Theorem 2.4 *A conjunctive query Q_V does not disclose any information about another conjunctive query Q_S if and only if for every tuple $t \in Tup(\mathcal{I})$, and for every set of subgoals \mathcal{G}_S in Q_S and \mathcal{G}_V in Q_V such that $\mathcal{G}_S \cup \mathcal{G}_V$ is compatible with t ,*

$$Q_S|_{(\mathcal{G}_S,t)} \subseteq Q_S \vee Q_V|_{(\mathcal{G}_V,t)} \subseteq Q_V.$$

Consider an algorithm which naively implements the above condition for perfect privacy (see Algorithm 1). There are three steps which would make such an algorithm intractable.

- H1: **for every** tuple $t \in Tup(\mathcal{I})$ (Line 2).
- H2: **for every** set of subgoals \mathcal{G}_S in Q_S and \mathcal{G}_V in Q_V which are compatible with t (Line 4).
- H3: Checking whether $Q_S|_{(\mathcal{G}_S,t)} \subseteq Q_S$ OR $Q_V|_{(\mathcal{G}_V,t)} \subseteq Q_V$ (Line 9).

To identify query classes for which Algorithm 1 is tractable, each of the above three steps should be tractable.

Step H3: First, given sets of subgoals \mathcal{G}_S and \mathcal{G}_V which are compatible with a tuple t , constructing the queries $Q_S|_{(\mathcal{G}_S,t)}$ and $Q_V|_{(\mathcal{G}_V,t)}$ is at most linear in the size of the two queries. Performing the two containment checks may not be polynomial in the size of the queries [27]. However, there are subclasses of conjunctive queries for which Step H3 is tractable [30, 115]. These query classes include queries with no self joins, queries with at most one self join, simple tableaux queries, acyclic queries and queries with bounded query-width. We discuss these cases in Section 2.5.

Step H2: Since we are only considering conjunctive queries (without inequalities), checking whether a set of subgoals \mathcal{G} is compatible with a tuple t is linear in the number of subgoals. It is sufficient to check for every $G \in \mathcal{G}$, if $G[i]$ has a constant, then $t[i]$ has the same constant. Hence, Steps H2 and H3 can be performed in time $O(m2^{2k}(nr + T(\text{H3})))$, where $T(\text{H3})$ is the time taken to perform Step H3, k is the maximum number of subgoals in either query which share the same relational symbol, n is the total number of subgoals which appear in both the queries, r is the maximum arity of a relation which appears in both the queries, and m is the number of relation symbols that appear in both queries. If k is a constant, both the Steps H2 and H3 can be performed in time polynomial in the size of the inputs if the two queries Q_S and Q_V belong to one of the query classes listed above.

Step H1: Step H1 involves a universal quantification over the set of all tuples in the domain. Clearly, assuming that the domain of tuples is very small (bounded by some polynomial in the size of the input) ensures that the algo-

rithm is tractable. However, such an assumption would greatly limit the utility of our result. In most cases, the size of the domain of tuples is very large compared to size of the queries. Hence, in the rest of the section we propose an alternate characterization for perfect privacy where we eliminate the universal quantification over the domain of tuples. Interestingly, we are able to remove the quantification over the domain of tuples when we assume that the domain of tuples is sufficiently large.

Definition 2.1 (Sufficiently Large Domains)

Let $Q(a_1, a_2, \dots, a_m) : \neg G_1 \wedge G_2 \wedge \dots \wedge G_n \wedge C_Q$ be a conjunctive query. We say that the domains of the attributes appearing in Q are sufficiently large with respect to Q , if in every distinct variable x_i appearing in the query can be assigned to a unique constant c_i that does not appear in Q and is in the domain of the associated attribute A_i .

The plan for the rest of the section is as follows. First, we generalize the construction $Q|_{(\mathcal{G},t)}$ to $Q|_{(\mathcal{G},G)}$, where G is a subgoal such that every tuple compatible with G is also compatible with all the subgoals in \mathcal{G} . We then show that this construction can also be used to reason about critical tuples. More specifically, we show that $Q|_{(\mathcal{G},G)} \subseteq Q$ if and only if every tuple compatible with G is not critical to Q . We then use this construction to state an alternate characterization for perfect privacy (Theorem 2.5). This new characterization does not involve a universal quantifier over all the tuples in the domain. Finally, we prove Theorem 2.5 and bound the running time of an algorithm which implements Theorem 2.5.

Let us start with some definitions. Let \mathcal{G} be a set of compatible subgoals in conjunctive query Q . We say that a subgoal G is *compatible* with \mathcal{G} if every tuple compatible with G is also compatible with \mathcal{G} . We call a subgoal G^* the *most general unifier* of \mathcal{G} (denoted by $mgu(\mathcal{G})$) if a tuple is compatible with G^*

if and only if the tuple is compatible with \mathcal{G} . Since we are only dealing with conjunctive queries without inequalities, it is easy to see that the most general unifier of a set of subgoals is unique and well defined. Let V_G denote the set of variables and constants appearing in G .

Given a query Q , a set of compatible subgoals \mathcal{G} and a subgoal G which is compatible with \mathcal{G} , denote by $Q|_{(\mathcal{G},G)}$ the query $\rho_G(Q - \mathcal{G})$. Here, ρ_G is a partial valuation defined as follows:

$$\rho_G : A_Q \cup B_Q \rightarrow A_Q \cup B_Q \cup K_Q \cup V_G \text{ such that,}$$

$$\rho_G(x_i) = \begin{cases} G[j], & \text{if } G_\ell[j] = x_i, \text{ for some } G_\ell \in \mathcal{G} \\ x_i, & \text{otherwise} \end{cases}$$

The following two lemmas show interesting properties of $Q|_{(\mathcal{G},G)}$. Lemma 2.3 shows that the query $Q|_{(\mathcal{G},t)}$ is contained in the query $Q|_{(\mathcal{G},G)}$. Lemma 2.4 connects $Q|_{(\mathcal{G},G)}$ and critical tuples.

Lemma 2.3 *Let Q be a conjunctive query, \mathcal{G} be a set of compatible subgoals, G be a subgoal (not necessarily in Q) which is compatible with \mathcal{G} , and t be a tuple compatible with G . Then*

$$Q|_{(\mathcal{G},t)} \subseteq Q|_{(\mathcal{G},G)}$$

Proof. Compare $Q|_{(\mathcal{G},t)}$ and $Q|_{(\mathcal{G},G)}$. If ρ_t is a mapping which maps variables in G to corresponding constants in the tuple t and is the identity function for every other symbol, then it is easy to see that ρ_t is a homomorphism from $Q|_{(\mathcal{G},G)}$ to $Q|_{(\mathcal{G},t)}$. Hence, from [27], $Q|_{(\mathcal{G},t)} \subseteq Q|_{(\mathcal{G},G)}$. ■

Lemma 2.4 *Let Q be a conjunctive query, \mathcal{G} be a set of compatible subgoals in Q , and G be a subgoal (not necessarily in Q) which is compatible with \mathcal{G} . There is a tuple*

compatible with G that is critical to Q if and only if

$$Q|_{(\mathcal{G},G)} \not\subseteq Q$$

Proof. Suppose some tuple t , that is compatible with G , is critical to Q . From Theorem 2.3, $Q|_{(\mathcal{G},t)} \not\subseteq Q$. From Lemma 2.3, $Q|_{(\mathcal{G},t)} \subseteq Q|_{(\mathcal{G},G)}$. Hence, $Q|_{(\mathcal{G},G)} \not\subseteq Q$.

To show the converse, suppose $Q|_{(\mathcal{G},G)} \not\subseteq Q$. Then there is an instance I and an output tuple t^o such that $t^o \in Q|_{(\mathcal{G},G)}(I)$ while $t^o \notin Q(I)$. Let ρ be the valuation mapping $Q|_{(\mathcal{G},G)}$ to tuples in I . Let t be a tuple such that $t[i] = \rho(G[i])$. Clearly, t is compatible with G . We show that t is critical to Q , completing the proof. Like in the previous case, we can define a homomorphism ρ_t from $Q|_{(\mathcal{G},G)}$ to $Q|_{(\mathcal{G},t)}$. Since the substitutions made by ρ_t conform to the substitutions made by ρ , ρ is also a valuation from $Q|_{(\mathcal{G},t)}$ to I outputting t^o . This shows that $Q|_{(\mathcal{G},t)} \not\subseteq Q$ and hence, t is critical to Q . ■

We are now ready to state our alternate characterization for perfect privacy.

Theorem 2.5 *Let Q_S and Q_V be conjunctive queries, and let the domains of attributes be sufficiently large with respect to each of the queries. Q_V does not disclose any information about Q_S if and only if for every set of compatible subgoals \mathcal{G}_S in Q_S and \mathcal{G}_V in Q_V ,*

$$Q_S|_{(\mathcal{G}_S, G^*)} \subseteq Q_S \text{ OR } Q_V|_{(\mathcal{G}_V, G^*)} \subseteq Q_V$$

where $G^* = \text{mgu}(\mathcal{G}_S \cup \mathcal{G}_V)$.

In the remainder of this section, we prove the above theorem. Proving the “if” direction of the above condition is quite easy. Suppose tuple t is critical to both Q_S and Q_V . Then from Theorem 2.3, there are sets of subgoals \mathcal{G}_S in Q_S

and \mathcal{G}_V in Q_V such that

$$Q_S|_{(\mathcal{G}_S, t)} \not\subseteq Q_S \text{ AND } Q_V|_{(\mathcal{G}_V, t)} \not\subseteq Q_V$$

Since t is compatible with \mathcal{G}_S and \mathcal{G}_V , t is also compatible with $G^* = \text{mgu}(\mathcal{G}_S \cup \mathcal{G}_V)$. Hence, using Lemma 2.3, we get

$$Q_S|_{(\mathcal{G}_S, G^*)} \not\subseteq Q_S \text{ AND } Q_V|_{(\mathcal{G}_V, G^*)} \not\subseteq Q_V$$

Proving the converse, however, is trickier. We need to show that if $Q_S|_{(\mathcal{G}_S, G^*)} \not\subseteq Q_S$ and $Q_V|_{(\mathcal{G}_V, G^*)} \not\subseteq Q_V$, then Q_S and Q_V share a critical tuple. From Lemma 2.4, all we can say is that there are tuples t_S and t_V such that t_S and t_V are compatible with G^* (and hence compatible with \mathcal{G}_S and \mathcal{G}_V), such that t_S is critical to Q_S and t_V is critical to Q_V . In fact, unless we assume the domains are sufficiently large, this is all the above condition guarantees.

The sufficiently large domain assumption ensures that given a query Q there is a valuation which maps every symbol in Q to a distinct constant in the respective domain. Using this property, we can prove Lemma 2.5, a stronger version of Lemma 2.4, which in turn will help us to prove the theorem.

Definition 2.2 (Fine $Q|_{(\mathcal{G}, G^*)}$ -Critical Tuple)

Let Q be a query, \mathcal{G} a set of subgoals in Q and G^* be a subgoal compatible with \mathcal{G} . A tuple t_f which is critical to a query Q is called a fine $Q|_{(\mathcal{G}, G^*)}$ -critical tuple¹ if there is a database instance I_f and a valuation ρ_f such that

- ρ_f is a valuation mapping $Q|_{(\mathcal{G}, G^*)}$ to tuples in I_f such that every variable appearing in $Q|_{(\mathcal{G}, G^*)}$ is mapped to a distinct constant not appearing in the query.

¹A similar definition is used in the proof of hardness of checking perfect privacy in [96]

- $\rho_f(G^*) = t_f$.
- ρ_f maps the goal of $Q|_{(\mathcal{G}, G^*)}$ to a tuple t_f^o such that $t_f^o \notin Q(I)$.

Lemma 2.5 *Let Q be a conjunctive query, \mathcal{G} be a set of compatible subgoals in Q , and G^* be a subgoal (not necessarily in Q) which is compatible with \mathcal{G} . Under the sufficiently large domain assumption, $Q|_{(\mathcal{G}, G^*)} \not\subseteq Q$ if and only if one of the tuples compatible with G^* is a fine $Q|_{(\mathcal{G}, G^*)}$ -critical tuple.*

Proof. Suppose there exists a tuple t which is a fine $Q|_{(\mathcal{G}, G^*)}$ -critical tuple. Then since t is critical to Q and is compatible with G^* , $Q|_{(\mathcal{G}, G^*)} \not\subseteq Q$ from Lemma 2.4.

To prove the other direction, suppose $Q|_{(\mathcal{G}, G^*)} \not\subseteq Q$. Then there is an instance I and a valuation ρ such that ρ maps the goal of $Q|_{(\mathcal{G}, G^*)}$ to a tuple t^o , such that $t^o \notin Q(I)$. From I and ρ we construct the fine critical tuple t_f and the instance-valuation pair (I_f, ρ_f) which satisfy the conditions in Definition 2.2. Simultaneously, we also build a function h whose domain is the set of all constants K and whose range is the set of constants in I . We will use h to explain why the t_f is indeed a fine critical tuple.

- *Initialization:* Fix an arbitrary total order on the variables in $Q|_{(\mathcal{G}, G^*)}$ and number the variables $\{x_1, x_2, \dots\}$. Initialize h as follows. For constants $c \in K$ that appear in I , $h(c) = c$. For all other constants $c' \in K$ that do not appear in I , $h(c') = d$ for some constant d appearing in I . Initialize $\rho_f = \rho$.
- *Substitution:* Consider the variables in increasing order of their numbering. Suppose $\rho(x_i) = c$, and c is either a constant appearing in $Q|_{(\mathcal{G}, G^*)}$ or $\rho_f(x_j) = c$ for some $j < i$. Then $\rho_f(x_i) = c_i$, for some c_i not appearing in $Q|_{(\mathcal{G}, G^*)}$ and not equal to $\rho_f(x_j)$ for any $j < i$. Set $h(c_i) = c$.

- *Fine critical tuple*: I_f is the set $\{\rho_f(G) \mid G \in Q|_{(\mathcal{G}, G^*)}\}$ and $t_f = \rho_f(G^*)$.

Note that the above construction is possible because of the sufficiently large domain assumption. To show that t_f is indeed a fine critical tuple, we need to show that ρ_f is a valuation mapping the goal of $Q|_{(\mathcal{G}, G^*)}$ to a tuple t_f^o such that $t_f^o \notin Q(I_f)$. From the construction it is clear that ρ_f is a valuation from $Q|_{(\mathcal{G}, G^*)}$ to I_f . Suppose for a contradiction, there is some valuation ρ' such that $t_f^o \in Q(I_f)$. Then $h \circ \rho'$ is a valuation which maps subgoals of Q to I and outputs the tuple t^o leading to the required contradiction. ■

Lemma 2.5 guarantees that if $Q_S|_{(\mathcal{G}_S, G^*)} \not\subseteq Q_S$ and $Q_V|_{(\mathcal{G}_V, G^*)} \not\subseteq Q_V$, then there exist t_S and t_V compatible with G^* such that t_S is a fine $Q_S|_{(\mathcal{G}_S, G^*)}$ -critical tuple and t_V is a fine $Q_V|_{(\mathcal{G}_V, G^*)}$ -critical tuple. We use this property to show that t_S is critical to Q_V , thus completing the proof.

Lemma 2.6 *Let Q_S and Q_V be conjunctive queries, \mathcal{G}_S be a set of subgoals in Q_S , \mathcal{G}_V be a set of subgoals in Q_V and G^* be a subgoal compatible with $\mathcal{G}_S \cup \mathcal{G}_V$. Let t_S and t_V be tuples compatible with G^* such that t_S is a fine $Q_S|_{(\mathcal{G}_S, G^*)}$ -critical tuple, and t_V is a fine $Q_V|_{(\mathcal{G}_V, G^*)}$ -critical tuple. Under the sufficiently large domain assumption, there exists a mapping $\gamma : K \rightarrow K$ that satisfies the following properties*

1. $\gamma(t_V) = t_S$.
2. γ is a bijection.
3. for every constant c appearing in Q_V , $\gamma(c) = c$.

Proof. We construct a candidate γ using the following steps. Let K_S and K_V denote the sets of constants appearing in t_S and t_V , respectively.

- *Initialization:* Let $\gamma : K \rightarrow K$ be the identity function. Set $C_V = C_S = \emptyset$.
- *Set $\gamma(t_V) = t_S$:* For every i , if $t_V[i] = c_i$ and $t_S[i] = d_i$, then set $\gamma(c_i) = d_i$, $C_V = C_V \cup \{c_i\}$ and $C_S = C_S \cup \{d_i\}$.
- *Make γ a bijection:* For every constant $d \in K_S - C_V$, set $\gamma(d)$ to a unique constant in $K_V - C_S$.

That $\gamma(t_V) = t_S$ is obvious. Property 3, namely that γ is the identity function for every constant appearing in Q_V , is preserved in each of the three steps. Clearly, this property holds after the initialization step. In the second step, suppose $t_V[i] = c_i$ appears in Q_V ; then since t_V is a fine critical tuple, surely c_i should also appear in G^* . Both t_V and t_S are compatible with G^* . This forces $t_S[i] = c_i$ and $\gamma(c_i) = c_i$. As a side effect, every c_i appearing in Q_V appears both in C_V and C_S . Hence, $\gamma(c_i)$ remains unchanged after the third step.

To prove that γ is a bijection, first we show γ is indeed a function. Suppose on the contrary, γ maps c_i to two different values d_i and d_j ; this can happen only in the second step where $t_V[i]$ is mapped to $t_S[i]$ and $t_S[j]$. This is possible only if $t_V[i] = t_V[j]$. Since G^* is mapped to t_V under the fine valuation, $G^*[i] = G^*[j]$ forcing $t_S[i] = t_S[j]$. Similarly, in the second step, no two distinct constants c_i and c_j are mapped to the same constant d .

To complete the proof that γ is indeed a bijection, note that the number of distinct constants in t_S and t_V are the same; since t_S and t_V are fine critical tuples, each distinct variable in G^* is assigned a distinct constant. Therefore, after the second and third steps, γ is a bijection. ■

With Lemma 2.6, the rest of the proof of Theorem 2.5 is straightforward. Let $I'_V = \gamma(I_V)$. Then from Lemma 2.6, $I'_V \cup \{t_S\} = \gamma(I_V \cup \{t_V\})$. Lemma 2.6 also

Input: Conjunctive Queries Q_S and Q_V

Output: Whether Q_S and Q_V share a critical tuple.

- 1: //for every set of subgoals in the two queries
- 2: **for all** $\mathcal{G}_S \subseteq Q_S, \mathcal{G}_V \subseteq Q_V$ **do**
- 3: //if all the subgoals in $\mathcal{G}_S \cup \mathcal{G}_V$ are compatible.
- 4: **if** $\mathcal{G}_S \cup \mathcal{G}_V$ are compatible subgoals of relation R of arity r **then**
- 5: //consider the partial valuation ρ' mapping the
//subgoals in $\mathcal{G}_S \cup \mathcal{G}_V$ to $G^* = \text{mgu}(\mathcal{G}_S \cup \mathcal{G}_V)$
- 6: Let G^* be a subgoal such that
$$1 \leq i \leq r, \exists G \in \mathcal{G}_S \cup \mathcal{G}_V : G[i] = c_i \Rightarrow G^*[i] = c_i$$
$$1 \leq i, j \leq r, \exists G, G' \in \mathcal{G}_S \cup \mathcal{G}_V, G[i] = G'[j] \Rightarrow G^*[i] = G^*[j]$$

Let $V_{G^*} = \{g_1, g_2, \dots\}$ be the set of distinct variables appearing in G^* .
- 7: $\rho_{G^*} : A_Q \cup B_Q \rightarrow A_Q \cup B_Q \cup K_Q \cup V_{G^*}$ such that
$$\rho_{G^*}(x_i) = \begin{cases} G_j^*, & \text{if } G_\ell[j] = x_i, \text{ for some } G_\ell \in \mathcal{G}_S \cup \mathcal{G}_V \\ x_i, & \text{otherwise} \end{cases}$$
- 8: $Q_S|_{(\mathcal{G}_S, G^*)} = \rho_{G^*}(Q_S - \mathcal{G}_S),$
 $Q_V|_{(\mathcal{G}_V, G^*)} = \rho_{G^*}(Q_V - \mathcal{G}_V).$
- 9: //check if $Q_S|_{(\mathcal{G}_S, G^*)} \subseteq Q_S$ and $Q_V|_{(\mathcal{G}_V, G^*)} \subseteq Q_V$
- 10: **if** $(Q_S|_{(\mathcal{G}_S, G^*)} \not\subseteq Q_S \text{ AND } Q_V|_{(\mathcal{G}_V, G^*)} \not\subseteq Q_V)$ **then**
- 11: **return** Q_S and Q_V SHARE a critical tuple.
- 12: **end if**
- 13: **end if**
- 14: **end for**
- 15: **return** Q_S and Q_V do NOT SHARE a critical tuple.

Algorithm 2: Perfect Privacy (Sufficiently Large Domains)

shows that γ changes the value of a constant only if the constant does not appear in Q_V . Hence, ρ is a valuation mapping Q_V to some I if and only if $\gamma \circ \rho$ is a valuation mapping Q_V to $\gamma(I)$. Therefore,

$$\begin{aligned} Q_V(I_V \cup \{t_V\}) &\not\subseteq Q_V(I_V) \\ &\Rightarrow Q_V(\gamma(I_V \cup \{t_V\})) \not\subseteq Q_V(\gamma(I_V)) \\ &\Rightarrow Q_V(I'_V \cup \{t_S\}) \not\subseteq Q_V(I'_V) \end{aligned}$$

Hence, $t_S \in \text{crit}(Q_S) \cap \text{crit}(Q_V)$, completing the proof. ■

Algorithm 2 sketches a simple algorithm which implements Theorem 2.5. Since we have eliminated the universal quantifier over tuples, we are left with only two steps which contribute to the complexity of Algorithm 2.

- H2': **for every** set of subgoals \mathcal{G}_S in Q_S and \mathcal{G}_V in Q_V which are compatible (Line 2).
- H3': Checking whether $Q_S|_{(\mathcal{G}_S, G^*)} \subseteq Q_S$ OR $Q_V|_{(\mathcal{G}_V, G^*)} \subseteq Q_V$ (Line 10).

We bound the running time of Algorithm 2 in terms of the notation in Table 2.1 and use it in Section 2.5 to motivate our search for tractable query classes. Given a set of compatible subgoals \mathcal{G}_S in Q_S , \mathcal{G}_V in Q_V (say, of relation R), the most general unifier G^* can be constructed in time $O(r_R(|\mathcal{G}_S| + |\mathcal{G}_V|))$, where r_R is the arity of relation R . The queries $Q_S|_{(\mathcal{G}_S, G^*)}$ and $Q_V|_{(\mathcal{G}_V, G^*)}$ can be constructed in time $O(r_R(|Q_S| - |\mathcal{G}_S| + |Q_V| - |\mathcal{G}_V|))$. Hence in total this takes $O(rn)$. Let $T(\text{H3}')$ be the time taken to perform the two containment checks in (H3'). The number of sets of subgoals $\mathcal{G}_S \cup \mathcal{G}_V$ which share the same relation symbol is at most $m2^{2k}$. Hence, the running time can be bounded by

$$O(m2^{2k}(nr + T(\text{H3}'))) \tag{2.4}$$

Before we extend out alternate characterization to the case with key constraints (in Section 2.4.2), we shed more intuition on the sufficiently large domain assumption.

The Sufficiently Large Domain Assumption

In the proof of Theorem 2.5 we showed that the sufficiently large domain assumption is “sufficient” to eliminate the universal quantification over all tuples

in the domain in the naive alternate characterization of perfect privacy (i.e., Theorem 2.4). We now show that this assumption is in some sense “necessary” to help us in our quest for tractable subclasses of queries; i.e., even if Step H2 (considering all sets of compatible subgoals) and Step H3 (query containment) can be performed in time polynomial in the size of the input, there are small domains and queries Q_S and Q_V such that checking perfect privacy is co-NP-complete.

We first present an example which illustrates how Algorithm 2 gives incorrect results when the domains are small. We then extend the intuition gained from the example to prove the “necessity” result.

$$\begin{aligned}
 Q_S(a_1, a_2, a_3) & :- R(a_1, a_2, a_3, b_4, 0), R(1, a_2, a_3, 0, 0) \\
 & \quad R(a_1, 0, 0, 0, 0), R(a_1, 0, 1, 0, 0) \\
 \\
 Q_V(a_1, a_2, a_3) & :- R(a_1, a_2, a_3, 1, b_5), R(1, a_2, a_3, 1, 1) \\
 & \quad R(a_1, 1, 0, 1, 1), R(a_1, 1, 1, 1, 1)
 \end{aligned}$$

Figure 2.4: Queries Q_S and Q_V from Example 2.3

Example 2.3 Consider two queries Q_S and Q_V shown in Figure 2.4 on a relation R having five binary attributes – each attribute takes values 0 or 1. Note that the domains are not sufficiently large since each query has six symbols (two constants and four variables) while the attribute domains have only two constants.

A tuple critical to both the queries should be compatible with at least one subgoal in Q_S and one subgoal in Q_V (i.e., $|\mathcal{G}_V| \geq 1$ and $|\mathcal{G}_S| \geq 1$). One can easily check that the only possible candidate for $\mathcal{G}_S \cup \mathcal{G}_V$ is

$$\mathcal{G}_S = \{R(a_1, a_2, a_3, b_4, 0)\}; \mathcal{G}_V = \{R(a_1, a_2, a_3, 1, b_5)\}$$

$$\begin{array}{l}
I_S = \begin{array}{|c|c|c|c|c|}
\hline
A_1 & A_2 & A_3 & A_4 & A_5 \\
\hline
1 & 1 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
\hline
\end{array} \\
I_V = \begin{array}{|c|c|c|c|c|}
\hline
A_1 & A_2 & A_3 & A_4 & A_5 \\
\hline
1 & 1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 \\
\hline
\end{array}
\end{array}$$

$$Q_S|_{(\mathcal{G}_S, G^*)}(a_1, a_2, a_3) :- R(1, a_2, a_3, 0, 0), R(a_1, 0, 0, 0, 0), R(a_1, 0, 1, 0, 0)$$

$$Q_V|_{(\mathcal{G}_V, G^*)}(a_1, a_2, a_3) :- R(1, a_2, a_3, 1, 1), R(a_1, 1, 0, 1, 1), R(a_1, 1, 1, 1, 1)$$

Figure 2.5: I_S , I_V , and Queries $Q_S|_{(\mathcal{G}_S, G^*)}$, $Q_V|_{(\mathcal{G}_V, G^*)}$ for Example 2.3

The most general unifier of $\mathcal{G}_S \cup \mathcal{G}_V$, is $G^* = R(a_1, a_2, a_3, 1, 0)$. The queries $Q_S|_{(\mathcal{G}_S, G^*)}$ and $Q_V|_{(\mathcal{G}_V, G^*)}$ are shown in Figure 2.5. Also shown in the same figure are instances I_S and I_V . It is easy to check that $Q_S|_{(\mathcal{G}_S, G^*)} \not\subseteq Q_S$ and $Q_V|_{(\mathcal{G}_V, G^*)} \not\subseteq Q_V$.

$$\{(0, 1, 1) \in Q_S|_{(\mathcal{G}_S, G^*)}(I_S) \text{ and } Q_S(I_S) = \emptyset\} \Rightarrow Q_S|_{(\mathcal{G}_S, G^*)} \not\subseteq Q_S$$

$$\{(0, 1, 1) \in Q_V|_{(\mathcal{G}_V, G^*)}(I_V) \text{ and } Q_V(I_V) = \emptyset\} \Rightarrow Q_V|_{(\mathcal{G}_V, G^*)} \not\subseteq Q_V$$

Hence, Algorithm 2 returns that Q_S and Q_V share a critical tuple.

In the rest of the example we will show using Lemma 2.4 that Q_S and Q_V , in fact, do not share any critical tuples. Recall that if Q_S and Q_V share any critical tuples, then the tuple should be compatible to G^* ; i.e, the tuple should be of the form $R(_, _, _, 1, 0)$, where ‘ $_$ ’ can be replaced by either 1 or 0. It is easy to check the following properties:

1. A tuple t is compatible with G^* if and only if it is compatible with at least one of the following subgoals:

- $G_1^* = R(1, a_2, a_3, 1, 0)$.
- $G_2^* = R(a_1, 0, 0, 1, 0)$.

- $G_3^* = R(a_1, 0, 1, 1, 0)$.
 - $G_4^* = R(a_1, 1, 0, 1, 0)$.
 - $G_5^* = R(a_1, 1, 1, 1, 0)$.
2. $Q_S|_{(\mathcal{G}_S, G_1^*)}(1, a_2, a_3) : -R(1, a_2, a_3, 0, 0), R(1, 0, 0, 0, 0), R(1, 0, 1, 0, 0)$.
 $Q_S|_{(\mathcal{G}_S, G_1^*)} \subseteq Q_S$: The homomorphism which maps a_1 to 1 and b_4 to 0 and is an identity function for the rest of the symbols is a homomorphism from Q_S to $Q_S|_{\mathcal{G}_S, G_1^*}$. Hence, every tuple compatible with G_1^* is not critical to Q_S .
 3. $Q_S|_{(\mathcal{G}_S, G_2^*)}(a_1, 0, 0) : -R(1, 0, 0, 0, 0), R(a_1, 0, 0, 0, 0), R(a_1, 0, 1, 0, 0)$
 $Q_S|_{(\mathcal{G}_S, G_2^*)} \subseteq Q_S$: The homomorphism mapping a_2, a_3 and b_4 to 0 is the required homomorphism. Hence, every tuple compatible with G_2^* is not critical to Q_S .
 4. $Q_S|_{(\mathcal{G}_S, G_3^*)}(a_1, 0, 1) : -R(1, 0, 1, 0, 0), R(a_1, 0, 0, 0, 0), R(a_1, 0, 1, 0, 0)$
 $Q_S|_{(\mathcal{G}_S, G_3^*)} \subseteq Q_S$: The homomorphism mapping a_2 and b_4 to 0, and a_3 to 1 is the required homomorphism. Hence, every tuple compatible with G_3^* is not critical to Q_S .
 5. $Q_V|_{(\mathcal{G}_V, G_4^*)}(a_1, 1, 0) : -R(1, 1, 0, 1, 1), R(a_1, 1, 0, 1, 1), R(a_1, 1, 1, 1, 1)$
 $Q_V|_{(\mathcal{G}_V, G_4^*)} \subseteq Q_V$: The homomorphism mapping a_2 and b_5 to 1, and a_3 to 0 is the required homomorphism. Hence, every tuple compatible with G_4^* is not critical to Q_V .
 6. $Q_V|_{(\mathcal{G}_V, G_5^*)}(a_1, 1, 1) : -R(1, 1, 1, 1, 1), R(a_1, 1, 0, 1, 1), R(a_1, 1, 1, 1, 1)$
 $Q_V|_{(\mathcal{G}_V, G_5^*)} \subseteq Q_V$: The homomorphism mapping a_2, a_3 and b_5 to 1 is the required homomorphism. Hence, every tuple compatible with G_5^* is not critical to Q_V .

Hence, from Lemma 2.4, every tuple compatible with $\mathcal{G}_S \cup \mathcal{G}_V$ is either not critical to Q_S or not critical to Q_V . Recall that Lemma 2.4 is true irrespective of the domain size. Hence, Q_S and Q_V do not share any critical tuple. ■

The construction in the above example can be extended to formally prove the “necessity” of the sufficiently large domain assumption.

Theorem 2.6 *There exist queries Q_S and Q_V with corresponding small domains such that*

1. *There is one subgoal $G_S \in Q_S$ and one subgoal $G_V \in Q_V$ such that G_S and G_V are compatible and G^* is their most general unifier.*
2. *For every other set of subgoals $\mathcal{G}_S \in Q_S$ and $\mathcal{G}_V \in Q_V$, $\mathcal{G}_S \cup \mathcal{G}_V$ are not compatible.*
3. *$Q_S|_{(\{G_S\}, G^*)} \not\subseteq Q_S$ and $Q_V|_{(\{G_V\}, G^*)} \not\subseteq Q_V$.*

Checking whether Q_S and Q_V indeed share a critical tuple is NP-complete.

Proof. Clearly the problem is in NP, since it is sufficient to guess the shared critical tuple from a finite domain. To show hardness, we employ a reduction from the following NP-complete problem (see Lemma 2.8):

Joint Satisfiability Problem: Given two satisfiable CNF formulae ϕ and ψ , checking whether $\phi \wedge \psi$ is satisfiable is NP-complete.

Let ϕ_S and ϕ_V be the inputs to the joint satisfiability problem. Let x_1, x_2, \dots, x_n be the variables mentioned in $\phi_S \wedge \phi_V$. Without loss of generality, ϕ_S and ϕ_V are in 3-CNF. Hence, every clause $c_j = (\ell_{j_1} \vee \ell_{j_2} \vee \ell_{j_3})$, where each ℓ_{j_k} is either x_{j_k} or \bar{x}_{j_k} .

Let R be a relation with $(n + 2)$ binary attributes. Construct Q_S as follows:

- The goals of Q_S is the tuple (a_1, a_2, \dots, a_n) .

- $R(a_1, a_2, \dots, a_n, b_{n+1}, 0)$ is the first subgoal in the query Q_S .
- For every clause $c_j = (\ell_{j_1} \vee \ell_{j_2} \vee \ell_{j_3})$, $G_j = R(r_1, r_2, \dots, r_n, 0, 0)$ is a subgoal of Q_S such that
 - If $i = j_k$, $k \leq 3$ and $\ell_{j_k} = x_{j_k}$, then $r_i = 0$.
 - If $i = j_k$, $k \leq 3$ and $\ell_{j_k} = \bar{x}_{j_k}$, then $r_i = 1$.
 - Else, $r_i = a_i$.

Construct Q_V in the same manner with the following changes:

- The first subgoal of Q_V is $R(a_1, a_2, \dots, a_n, 1, b_{n+2})$.
- Every other subgoal has $r_{n+1} = r_{n+2} = 1$ instead of 0 unlike in Q_S .

Queries Q_S and Q_V from Example 2.3 have been constructed as described above using the following 3-CNF formulae:

$$\phi_S = (\bar{x}_2)(x_1 + x_3)(x_1 + \bar{x}_3)$$

$$\phi_V = (\bar{x}_2)(\bar{x}_1 + x_3)(\bar{x}_1 + \bar{x}_3)$$

The constructed queries satisfy the three requirements in the statement of the theorem. The first subgoal of Q_S is compatible with the first subgoal of Q_V . It is easy to see that no other subgoals are compatible.

In order to show that Q_S and Q_V satisfy the third requirement, we prove the following key result in Lemma 2.7: a tuple t satisfies ϕ_S if and only if $Q_S|_{(\{G_S\}, t)} \not\subseteq Q_S$.

The third requirement and the NP-hardness follow directly from Lemma 2.7. Since ϕ_S and ϕ_V are satisfiable, there are tuples t_S and t_V which satisfy them.

Clearly in this case, $Q_S|_{(\{G_S\}, t_S)} \not\subseteq Q_S$ and $Q_S|_{(\{G_V\}, t_V)} \not\subseteq Q_V$. Requirement 3 follows from Lemma 2.3.

To show NP-hardness, it is enough to show that t satisfies $\phi_S \wedge \phi_V$ if and only if t is critical to both Q_S and Q_V . This follows directly from Lemma 2.7 and Theorem 2.3. ■

Lemma 2.7 *Let Q_S be as constructed in Theorem 2.6. Let $t \in \{0, 1\}^n$ be a binary tuple, and let $t' = \{0, 1\}^{n+2}$ such that $t'[i] = t[i]$ for $i = 1, 2, \dots, n$ and $t'[n+1] = t'[n+2] = 0$. A tuple t satisfies ϕ_S if and only if $Q_S|_{(\{G_S\}, t')} \not\subseteq Q_S$, where .*

Proof. Suppose $Q_S|_{(\{G_S\}, t')} \subseteq Q_S$, there is a homomorphism from Q_S to $Q_S|_{(\{G_S\}, t')}$. Note that this homomorphism maps each a_i to $t[i]$. The first subgoal of Q_S , namely $R(a_1, a_2, \dots, b_{n+1}, 0)$, should be mapped to some $G_j = R(r_1, r_2, \dots, r_n, 0, 0)$. This is possible only if t does not satisfy ϕ_S .

If t does not satisfy ϕ_S , there is some clause, say c_j , which is not satisfied. Then $a_i = t[i]$ is a homomorphism from Q_S to $Q_S|_{(\{G_S\}, t)}$ (which maps the first subgoal in Q_S to G_j in $Q_S|_{(\{G_S\}, t)}$ and the other subgoals G_ℓ in Q_S to the corresponding subgoals in $Q_S|_{(\{G_S\}, t)}$). ■

Lemma 2.8 *Given two satisfiable CNF formulae ϕ and ψ , checking whether $\phi \wedge \psi$ is satisfiable is NP-complete.*

Proof. Clearly, the problem is in NP. To show hardness, suppose there is a polynomial time algorithm to solve the joint satisfiability problem. We can construct a polynomial time algorithm for SAT as follows. Every conjunct in a CNF is trivially satisfiable. Start by checking whether the first two conjuncts are jointly

satisfiable (using the supposedly efficient algorithm for joint satisfiability). At any point of time, if the first k conjuncts are unsatisfiable, stop since the whole formula is unsatisfiable. If the first k conjuncts, say ϕ_k , is satisfiable, since the $(k + 1)^{st}$ conjunct c_{k+1} is satisfiable, check if $\phi_k \wedge c_{k+1}$ is satisfiable (again using the efficient algorithm). If all the conjuncts are exhausted then the CNF formula is satisfiable. ■

2.4.2 Perfect Privacy with Key Constraints

In this section we consider the case when the space of database instances $\mathcal{I}^{\mathcal{K}}$ should satisfy key constraints specified in $\mathcal{K} = \{\mathcal{A}_1, \dots, \mathcal{A}_k\}$; i.e., for every instance $I \in \mathcal{I}^{\mathcal{K}}$

$$\forall \mathcal{A}_i \in \mathcal{K}, (t_1, t_2 \in I \wedge t_1[\mathcal{A}_i] = t_2[\mathcal{A}_i] \Rightarrow t_1 = t_2)$$

In this scenario, a query Q_V discloses some information about the answer to Q_S if and only if there exist tuples t_S, t_V in the domain of relation R , such that $t_S \in \text{crit}(Q_S)$ and $t_V \in \text{crit}(Q_V)$, and for some set of attributes \mathcal{A}_i which is a key for relation R , $t_S[\mathcal{A}_i] = t_V[\mathcal{A}_i]$. We define sets of subgoals \mathcal{G}_S and \mathcal{G}_V to be *compatible on \mathcal{A}_i* if there are tuples $t_S, t_V \in D(R)$ such that $t_S[\mathcal{A}_i] = t_V[\mathcal{A}_i]$ and t_S and t_V are compatible with \mathcal{G}_S and \mathcal{G}_V , respectively. The following theorem provides an alternate characterization of perfect privacy in the presence of key constraints in terms of query containment.

Theorem 2.7 *A conjunctive query Q_V does not disclose any information about con-*

conjunctive query Q_S if and only if for every relation R appearing in both Q_S and Q_V ,

$$\begin{aligned} & \forall t_S, t_V \in D(R), \forall \mathcal{A}_i \in \mathcal{K}_R, \\ & \forall \mathcal{G}_S \subseteq Q_S \text{ compatible with } t_S, \forall \mathcal{G}_V \subseteq Q_V \text{ compatible with } t_V \\ & (t_S[\mathcal{A}_i] = t_V[\mathcal{A}_i] \Rightarrow (Q_S|_{(\mathcal{G}_S, t_S)} \subseteq Q_S \text{ OR } Q_S|_{(\mathcal{G}_V, t_V)} \subseteq Q_V)) \end{aligned}$$

Note that this definition too has a universal quantifier on the tuples in the domain; the remainder of this section is focused on removing this quantifier under the sufficiently large domain assumption. Note that in Theorem 2.7, one needs to check the query containment only for \mathcal{G}_S and \mathcal{G}_V which are compatible on some $\mathcal{A}_i \in \mathcal{K}_R$. To simulate Theorem 2.7 for the case when the domains are sufficiently large, we construct for every key $\mathcal{A}_i \in \mathcal{K}_R$ two subgoals – \hat{G}_{S_i} compatible with \mathcal{G}_S and \hat{G}_{V_i} compatible with \mathcal{G}_V – that are compatible on \mathcal{A}_i as follows:

- $G_S^* = mgu(\mathcal{G}_S)$ and $G_V^* = mgu(\mathcal{G}_V)$.
- Now start with $\hat{G}_{S_i} = G_S^*$ and $\hat{G}_{V_i} = G_V^*$ and make changes such that any pair of tuples t_S, t_V , compatible with \hat{G}_{S_i} and \hat{G}_{V_i} respectively, are compatible with both \mathcal{G}_S and \mathcal{G}_V on the attributes in \mathcal{A}_i . This we do by chasing the following two rules on \hat{G}_{S_i} and \hat{G}_{V_i} :
 - If for some $A_\ell \in \mathcal{A}_i$, $\hat{G}_{S_i}[\ell] = c$ a constant, then set $\hat{G}_{V_i}[\ell] = c$ and vice versa.
 - If for some $A_\ell, A_m \in \mathcal{A}_i$, $\hat{G}_{S_i}[\ell] = \hat{G}_{S_i}[m]$, then set $\hat{G}_{V_i}[\ell] = \hat{G}_{V_i}[m]$ and vice versa.

It is easy to see that the above chase procedure terminates in time proportional to $|\mathcal{A}_i|$.

Theorem 2.8 *When the domains of attributes of the relations appearing in the conjunctive queries Q_S and Q_V are sufficiently large with respect to the queries, Q_V does not disclose any information about conjunctive query Q_S if and only if for every relation R appearing in both Q_S and Q_V ,*

$$\forall \mathcal{A}_i \in \mathcal{K}_R, \forall \mathcal{G}_S \subseteq Q_S, \mathcal{G}_V \subseteq Q_V, \text{ compatible on } \mathcal{A}_i \text{ on } R$$

$$\left(Q_S|_{(\mathcal{G}_S, \hat{\mathcal{G}}_{S_i})} \subseteq Q_S \text{ OR } Q_S|_{(\mathcal{G}_V, \hat{\mathcal{G}}_{V_i})} \subseteq Q_V \right)$$

Proof. The proof of the above theorem is very similar to the proof of Theorem 2.5. When there are t_S and t_V critical to Q_S and Q_V such that for some \mathcal{A}_i , $t_S[\mathcal{A}_i] = t_V[\mathcal{A}_i]$, the condition in the theorem follows from Lemma 2.3. For the other direction, when $Q_S|_{(\mathcal{G}_S, \hat{\mathcal{G}}_{S_i})} \not\subseteq Q_S$ and $Q_S|_{(\mathcal{G}_V, \hat{\mathcal{G}}_{V_i})} \not\subseteq Q_V$, from Lemma 2.5 that there are t_S and t_V compatible with $\hat{\mathcal{G}}_{S_i}$ and $\hat{\mathcal{G}}_{V_i}$, respectively, which are (fine) critical to Q_S and Q_V . Since $\hat{\mathcal{G}}_{S_i}$ and $\hat{\mathcal{G}}_{V_i}$ are compatible on \mathcal{A}_i , we can construct a γ mapping $t_V[\mathcal{A}_i]$ to $t_S[\mathcal{A}_i]$ like in Lemma 2.6 to show that there is a t'_S which is critical to Q_V having $t'_S[\mathcal{A}_i] = t_S[\mathcal{A}_i]$. ■

2.5 Finding Tractable Cases

In this section we show how to utilize the alternate characterizations for perfect privacy to identify query classes where checking perfect privacy is tractable. We restrict our discussion to the scenario with independent tuples, since the reason for the intractability of checking perfect privacy is the same in the other case. We will be using the notation from Figure 2.1 in this section.

We utilize two observations to identify tractable query classes. Our analysis of the running time of Algorithm 2 (Equation 2.4) shows that apart from looping

over all sets of compatible subgoals \mathcal{G}_S in Q_S and \mathcal{G}_V in Q_V (H2') and checking two containments (H3'), all the other steps involved can be implemented in time polynomial in the size of the queries. So for queries with a bounded number of self-joins per relation (k), any query class for which query containment is tractable also permits efficient enforcement of perfect privacy (Section 2.5.1). More interestingly, in our definitions of privacy, the query containment checks we are interested in are of a special form. We illustrate this for a subclass of tableau queries; even though query containment is not tractable for this class of queries, the query containments we are interested in are so simple that we can check perfect privacy in time $O(n^2r)$ (Section 2.5.2). In fact, for these queries, perfect privacy is tractable even if the number of self-joins per relation is not bounded by a constant. We conclude this section with a brief discussion of the complexity of checking perfect privacy when the two queries are from two different query classes (Section 2.5.3).

2.5.1 Using Tractability of $Q_1 \subseteq Q_2$

Since the seminal paper by Chandra and Merlin [27], which showed the *NP*-hardness of query containment for conjunctive queries, there has been a lot of work in finding subclasses of conjunctive queries for which the query containment problem is tractable. We utilize this work to identify four major classes of queries where checking perfect privacy is tractable – queries with at most one self join per relation (using results in [115]), simple tableau queries with bounded k (using results in [15]), acyclic queries with bounded k , and queries with bounded query-width and bounded k (using [30]).

Queries with at most one self join per relation

We start with a very simple case. Saraiya [115] shows that checking $Q \subseteq Q'$ is tractable whenever Q and Q' are such that for every subgoal G' in Q' there are at most two subgoals in Q which are compatible with G' .

Theorem 2.9 (Tractability of Containment [115]) *Let Q_1 and Q_2 be two conjunctive queries such that every subgoal in Q_2 is compatible with at most two subgoals in Q_1 . Then $Q_1 \subseteq Q_2$ can be checked in time $O(n^2r)$*

For the class of queries with at most one self-join, there are at most two subgoals of a relation in any query. Hence, any $Q|_{(\mathcal{G}, \hat{\mathcal{G}})}$ will also be a query with at most one self join. Therefore, $Q|_{(\mathcal{G}, \hat{\mathcal{G}})} \subseteq Q$ can be checked correctly using the algorithm in [115]. Moreover, since $k(Q) \leq 2$ for any query, we only need to loop over $O(n)$ sets of the form $\mathcal{G}_S \cup \mathcal{G}_V$.

Theorem 2.10 (Tractability of Perfect Privacy) *Let Q_S and Q_V be two queries having at most one self-join. Then perfect privacy can be checked in time $O(n^3r)$.*

Acyclic & Bounded Query-width Queries

There has been extensive work on relating the structure of conjunctive queries and the hardness of query containment. In particular, Chekuri and Rajaram [30] relate the hardness of query containment to the cyclicity of the query graph. The *query graph* is a hyper graph whose nodes are the symbols appearing in the query and hyper-edges are the subgoals in the query. A query is *acyclic* if the query graph is acyclic.

The “degree of cyclicity” in a query is usually measured using *query-width*. Acyclic queries have a query-width of 1. The basic idea behind query-width is the following. An acyclic query can be decomposed into a tree, with each vertex in the tree corresponding to one subgoal in the query.² This tree preserves properties like which subgoals share variables. For a cyclic query one may be able to construct a tree with each vertex in the tree corresponding to more than one subgoal. The query-width is the largest number of subgoals which are mapped to a single vertex in the “best” tree decomposition. We refer the reader to [30] for formal definitions.

Theorem 2.11 (Tractability of Containment [30]) *Let Q_2 be a query with a query-width bounded by c . Then $Q_1 \subseteq Q_2$ can be checked in time $O(c^2rn^{c+1} \log n)$.*

This gives us the following result on the complexity of checking perfect privacy.

Theorem 2.12 (Tractability of Perfect Privacy) *Let Q_S and Q_V be queries having query-width bounded by c and at most k self-joins per relation. Then perfect privacy can be checked in time $O(m2^{2k}c^2rn^{c+1} \log n)$.*

Simple Tableau Queries

In many database schemas, different attributes in the schema are incomparable. The hospital database schema in Figure 2.1 is one such example. Hence, in any conjunctive query on an instance of the hospital schema, a variable cannot be bound to two different attributes. Such queries are called *tableau queries*, or short *tableaux*.

²This can be done by using the *GYO*-reduction on the query-graph ([134]).

$$Q_S(\text{name}) \text{ :- } P(\text{pid}, \text{name}), \\ D(\text{pid}, \text{Hepatitis}, b_1), D(\text{pid}, \text{Cancer}, b_2)$$

	PID	PName	Disease	Medication
Q_S		name		
P	pid	name		
R	pid		Hepatitis	b_1
R	pid		Cancer	b_2

Figure 2.6: Query Q_S and its tableau representation.

Definition 2.3 (Tableau Query [15])

A tableau query Q is a conjunctive query where every variable (distinguished or non-distinguished) is associated with only one attribute in the relational schema.

As the name suggests, a tableau query is usually represented in a tabular form. The columns of this table are the attributes that appear in the query. The rows of the table are the subgoals of the query, and all definitions from Section 2.2.1 that involve subgoals thus extend to rows of a tableau query. The goal of the query is represented by the first row in the table called the summary row. Each row is labeled with the relation associated with the corresponding subgoal. Figure 2.6 illustrates a tableau representation of a tableau query. Note that no variable appears in more than one column. Even for this restricted class of queries, Aho et al. [15] show that the query containment problem is *NP*-hard. They identify a further subclass of queries, called *simple tableau queries*, wherein query containment is tractable. Figure 2.6 illustrates a simple tableau query.

Definition 2.4 (Simple Tableau Query [15])

A tableau query is simple if in any column with a repeated non-distinguished variable there is no other symbol that appears in more than one row.

A	B	C
b_1	b_2	c_3
c_1	b'_2	b_3
d_1	b''_2	b'_3
b'_1	d_2	e_3
e_1	b'_2	b''_3
b_1	e_2	b''_3

c_1	x_2	c_3
-------	-------	-------

A	B	C
d_1	b''_2	b'_3
b'_1	d_2	e_3
e_1	x_2	b''_3
c_1	e_2	b''_3

Figure 2.7: A tableau Q ; a subgoal \hat{G} compatible with the first two rows \mathcal{G} ; and the query $Q|_{(\mathcal{G}, \hat{G})}$.

Theorem 2.13 (Tractability of Containment [15]) *Let Q_1 and Q_2 be two simple tableaux. Then $Q_1 \subseteq Q_2$ can be checked in time $O(n^3r^2)$.*

For us to be able to use the above tractability result, we must show that given a simple tableau query Q , a set of compatible subgoals \mathcal{G} and a subgoal \hat{G} compatible with \mathcal{G} , the query $Q|_{(\mathcal{G}, \hat{G})}$ is also a simple tableau query.

Lemma 2.9 *Let Q be a simple tableau query, \mathcal{G} a set of compatible subgoals in Q and \hat{G} a row (not necessarily in Q) compatible with \mathcal{G} . Then $Q|_{(\mathcal{G}, \hat{G})}$ is a simple tableau query.*

Proof. We give an illustrative example followed by the formal proof.

ILLUSTRATIVE EXAMPLE: Consider the simple tableau in Figure 2.7 on a single relation R with schema $R(A, B, C)$. Let \mathcal{G} be the first two rows and \hat{G} be the most general unifier of \mathcal{G} . In order to construct $Q|_{(\mathcal{G}, \hat{G})}$, the variable b_1 is mapped to constant c_1 and b_3 to c_3 . The variables b_2 and b'_2 are both mapped to variable x_2 . Notwithstanding these substitutions, the resultant $Q|_{(\mathcal{G}, \hat{G})}$ is still a simple tableau query. The reasons are clear from the example. b_1 appears in \mathcal{G} and in $Q - \mathcal{G}$. Hence, no other symbol can repeat. So when b_1 in $Q - \mathcal{G}$ was replaced

by a c_1 , the only symbol which can repeat is c_1 . Also, b_3 does not repeat and hence the mapping b_3 to c_3 does not even appear in the column. So there is only one repeating symbol in column C in both Q and $Q|_{(\mathcal{G}, \hat{\mathcal{G}})}$. Finally, both b_1 and b'_1 cannot repeat in column B. Hence, mapping the two variables to a single variable does not cause a violation of the simple tableau property.

FORMAL PROOF Recall that $Q|_{(\mathcal{G}, \hat{\mathcal{G}})} = \rho_{\hat{\mathcal{G}}}(Q - \mathcal{G})$. Since Q is a simple tableau, $Q - \mathcal{G}$ is also a simple tableau. Suppose on the contrary, the mapping $\rho_{\hat{\mathcal{G}}}$ violates the simple tableau condition. Then $\rho_{\hat{\mathcal{G}}}$ should map two distinct symbols s_1 and s_2 in some column in $Q - \mathcal{G}$ to the same variable or constant. Both s_1 and s_2 cannot be non-distinguished variable, since in this case s_1 and s_2 appear in both $Q - \mathcal{G}$ and in \mathcal{G} , violating the simple tableau property of $Q - \mathcal{G}$. Both s_1 and s_2 cannot be constants, since in this case they cannot be mapped to the same symbol under $\rho_{\hat{\mathcal{G}}}$. Since there is only one distinguished variable in a column, s_1 and s_2 both cannot be variables; this would violate the simple tableau property of $Q - \mathcal{G}$ as one of s_1 and s_2 must be non-distinguished variable. Hence, exactly one of s_1, s_2 , is a constant. Say $s_1 = c$ and $s_2 = x$.

For x to be mapped to c under $\rho_{\hat{\mathcal{G}}}$, x should appear in one of the subgoals in \mathcal{G} . Hence, x appears in both Q and $Q - \mathcal{G}$. Since mapping x to c causes $Q|_{(\mathcal{G}, \hat{\mathcal{G}})}$ to violate the simple tableau property, there is some other non-distinguished variable b which repeats in $Q - \mathcal{G}$. Thus both non-distinguished variables x and b repeat in a column in Q . This is the required contradiction. ■

Using Lemma 2.9 and Theorem 2.13 we obtain the following theorem.

Theorem 2.14 (Tractability of Perfect Privacy) *Let Q_S and Q_V be two simple*

tableaux with at most k rows tagged with the same relation. Perfect privacy can be checked in time $O(m2^{2k}n^3r^2)$.

2.5.2 Using Tractability of $Q|_{(\mathcal{G}, \hat{\mathcal{G}})} \subseteq Q$

Our strategy to identify tractable query classes until now was limited to identifying query classes where query containment is tractable, without taking into account that the containments we check have a specific form. In this section we exploit this to identify another query class where checking perfect privacy is tractable.

We continue our discussion on tableau queries. Define a partial order \prec on the rows of the tableau as follows: if two rows r_1 and r_2 are tagged with the same relation, $r_1 \prec r_2$ if

$$\forall i \forall c : r_1[i] = c \Rightarrow r_2[i] = c$$

The above condition says that $r_1 \prec r_2$ if in every column that r_1 has a constant, r_2 has the same constant. We can now define a *maximal row*.

Definition 2.5 (Maximal Row) We call a row r in a tableau a maximal row if there is no other row r' tagged with the same relation as r such that $r \prec r'$.

The query class we are considering in this section are those tableau queries which only contain maximal rows. We call such tableaux *maximal row tableaux*. The query Q_S from Figure 2.6 is an example of a maximal row tableaux.

Definition 2.6 (Maximal Row Tableau) A tableau is a maximal row tableau if for every two rows r_1 and r_2 tagged with the same the relation, $r_1 \not\prec r_2$ and $r_2 \not\prec r_1$.

Unlike in the previous cases, where we considered query classes which permitted efficient query containment, we show that query containment is *NP*-hard (similar to [15]) even when the two queries are maximal row tableaux.

Theorem 2.15 (Intractability of Containment)

Let Q_1 and Q_2 be two maximal row tableaux. Checking $Q_1 \subseteq Q_2$ is *NP*-complete.

Proof. (sketch) The problem is clearly in *NP* since Q_1 and Q_2 are conjunctive queries. To show *NP*-hardness, given a 3-SAT instance, construct two tableaux like in Example 11 in [15] and replacing distinct a_i 's by distinct constants c_i . The two queries constructed are maximal row tableau and the proof in [15] holds even when the distinguished variables are replaced by constants. ■

Nevertheless, we are only interested in specific containments. We illustrate this using the critical tuple check. To show that a tuple t is critical to Q , it is enough to show a set of subgoals \mathcal{G} in Q which are compatible with t such that $Q|_{(G,t)} \not\subseteq Q$. If t is compatible with \mathcal{G} , then t is compatible with every $G \in \mathcal{G}$. Consider $Q|_{(G,t)}$, when Q is a maximal row tableau.

$Q|_{(G,t)} \subseteq Q$ iff there is a homomorphism h from the symbols in Q to the symbols in $Q|_{(G,t)}$ such that h is an identity function on constants and for any subgoal G' in Q , $h(G')$ is a subgoal in $Q|_{(G,t)}$ [27]. However, the following argument shows that there is no h such that $h(G)$ is a subgoal in $Q|_{(G,t)}$. First, G does not appear in $Q|_{(G,t)}$. Second, since Q is a maximal row tableau, for any other subgoal G' in Q having the same relation as G , there is some column i where $G[i] = c$, a constant, and $G'[i]$ is either a variable or a different constant. G' appears as $\rho_t(G')$ in $Q|_{(G,t)}$. Since no variable repeats across columns, $\rho_t(G')[i] = G'[i]$. Hence, no homomorphism will map G in Q to $\rho_t(G')$ in $Q|_{(G,t)}$,

for all $G' \neq G$, implying that $Q|_{(G,t)} \not\subseteq Q$.

Lemma 2.10 *Let Q be a maximal row tableau, t a tuple and G a subgoal in Q which is compatible with t . Then*

$$Q|_{(G,t)} \not\subseteq Q$$

As a consequence of Lemma 2.10, a tuple is critical to a maximal row tableau if and only if it is compatible with some row in the maximal row tableau. Thus we can now prove the following theorem:

Theorem 2.16 (Tractability of Perfect Privacy) *Let Q_S and Q_V be two maximal row tableaux. Then Q_S and Q_V share a critical tuple if and only if there are subgoals G_S in Q_S and G_V in Q_V such that G_S is compatible with G_V . Hence, perfect privacy can be checked in time $O(n^2r)$.*

2.5.3 Queries from Different Classes

Recall that the running time of Algorithm 2 is bounded by the product of $(nr + T(H3'))$ and the number of query containments to be checked. When Q_S and Q_V are from the query classes specified in Section 2.5.1, the number of query containments to be checked is bounded by $O(m2^k)$. $T(H3')$ is bounded by the running time of query containment for the class where checking query containment is less efficient.

As we showed in Theorem 2.16, for maximal row tableaux, we need not loop over all subsets of compatible subgoals. Hence, if Q_S is a maximal row tableau, and Q_V is from one of the subclasses listed in Section 2.5.1 (or vice versa), the

number of containments to be checked is bounded by $O(n2^k)$. Hence, we get the following theorem,

Theorem 2.17 *Let Q_S be a maximal row tableau. The complexity of checking whether a conjunctive query Q_V is perfectly private with respect to Q_S is*

- $O(n \cdot r)$, if Q_V has no self-joins.
- $O(n \cdot n^2r)$, if Q_V has at most one self-join.
- $O(n2^k \cdot c^2rn^{c+1} \log n)$, if Q_V has query-width bounded by c and at most k self-joins per relation.
- $O(n2^k \cdot n^3r^2)$, if Q_V is a simple tableau query with at most k self-joins per relation.

2.6 Summary

We adopted perfect privacy as the semantics for enforcing access control in databases. The problem of enforcing perfect privacy was known to be highly intractable even for conjunctive queries without inequalities. In this chapter we identified many subclasses of conjunctive queries for which enforcing perfect privacy is tractable. This helps policy designers to decide what policies can be efficiently enforced.

3.1 Introduction

In this chapter we initiate a formal study of privacy definitions for anonymous data publishing. Many organizations are publishing microdata, or tables that contain unaggregated information about individuals. These tables can include medical, voter registration, census, and customer data. Microdata is a valuable source of information for the allocation of public funds, medical research, and trend analysis. However, publishing such microdata poses a great privacy risk; if individuals can be uniquely identified in the microdata, then their private information (such as their medical condition) would be disclosed.

To avoid the identification of records in microdata, uniquely identifying information like names and social security numbers are removed from the table. However, this first sanitization still does not ensure the privacy of individuals in the data; combinations of seemingly innocuous attributes can uniquely identify individuals. For instance, it is estimated that 87% of the population of the United States can be identified by linking the combination of their gender, date of birth, and 5-digit zip code [121] with public records. Sets of attributes (like gender, date of birth, and zip code) that can be linked with external data to uniquely identify individuals in the population are called *quasi-identifiers*. To counter “linking attacks” using quasi-identifiers, Samarati and Sweeney proposed a definition of privacy called *k-anonymity* [113, 122]. A table satisfies *k-anonymity* if every record in the table is indistinguishable from at least $k - 1$ other records with respect to every set of quasi-identifier attributes; such a ta-

Table 3.1: Inpatient microdata.

	Non-Sensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	13053	28	Russian	Heart Disease
2	13068	29	American	Heart Disease
3	13068	21	Japanese	Viral Infection
4	13053	23	American	Viral Infection
5	14853	50	Indian	Cancer
6	14853	55	Russian	Heart Disease
7	14850	47	American	Viral Infection
8	14850	49	American	Viral Infection
9	13053	31	American	Cancer
10	13053	37	Indian	Cancer
11	13068	36	Japanese	Cancer
12	13068	35	American	Cancer

ble is called a k -anonymous table. Hence, for every combination of values of the quasi-identifiers in the k -anonymous table, there are at least k records that share those values. This ensures that individuals cannot be uniquely identified by linking attacks.

Example 3.1 Table 3.1 shows medical records from Gotham City hospital. Note that the table contains no uniquely identifying attributes like name, social security number, etc. In this example, we divide the attributes into two groups: the sensitive attributes (consisting only of medical condition) and the non-sensitive attributes (zip code, age, and nationality). An attribute is marked sensitive if an adversary must not be allowed to discover the value of that attribute for any individual in the dataset. Attributes

Table 3.2: 4-Anonymous Inpatient microdata.

	Non-Sensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	130**	< 30	*	Heart Disease
2	130**	< 30	*	Heart Disease
3	130**	< 30	*	Viral Infection
4	130**	< 30	*	Viral Infection
5	1485*	≥ 40	*	Cancer
6	1485*	≥ 40	*	Heart Disease
7	1485*	≥ 40	*	Viral Infection
8	1485*	≥ 40	*	Viral Infection
9	130**	3*	*	Cancer
10	130**	3*	*	Cancer
11	130**	3*	*	Cancer
12	130**	3*	*	Cancer

not marked sensitive are non-sensitive. Furthermore, let the collection of attributes $\{\text{zip code, age, nationality}\}$ be the quasi-identifier for this dataset. Table 3.2 shows a 4-anonymous table derived from the table in Table 3.1 (here “*” denotes a suppressed value; so, for example, “zip code = 1485*” means that the zip code is in the range [14850 – 14859] and “age=3*” means the age is in the range [30 – 39]). Note that in the 4-anonymous table, each tuple has the same values for the quasi-identifier as at least three other tuples in the table.

Because of its conceptual simplicity, k -anonymity has been widely discussed as a viable definition of privacy in data publishing, and due to algorithmic ad-

vances in creating k -anonymous versions of a dataset [7, 18, 81, 94, 113, 122, 136], k -anonymity has grown in popularity. However, does k -anonymity really guarantee privacy? In the next section, we will show that the answer to this question is interestingly *no*. We give examples of two simple, yet subtle attacks on a k -anonymous dataset that allow an attacker to learn sensitive information about individuals in the table. Defending against these attacks requires a stronger notion of privacy that we call ℓ -diversity, the focus of this chapter. Let us first show the two attacks to give the intuition behind the problems with k -anonymity.

3.1.1 Attacks On k -Anonymity

In this section we present two attacks, the *homogeneity attack* and the *background knowledge attack*, and we show how they can be used to compromise a k -anonymous dataset.

Homogeneity Attack: Alice and Bruce are antagonistic neighbors. One day Bruce falls ill and is taken by ambulance to the hospital. Having seen the ambulance, Alice sets out to discover what disease Bruce is suffering from. Alice discovers the 4-anonymous table of current inpatient records published by the hospital (Table 3.2), and so she knows that one of the records in this table contains Bruce's data. Since Alice is Bruce's neighbor, she knows that Bruce is a 31-year-old American male who lives in the zip code 13053 (the quiet town of Dryden). Therefore, Alice knows that Bruce's record number is 9, 10, 11, or 12. Now, all of those patients have the same medical condition (cancer), and so Alice concludes that Bruce has cancer.

Observation 1 *k -Anonymity can create groups that leak information due to lack of*

diversity in the sensitive attribute.

Such a situation is not uncommon. As a back-of-the-envelope calculation, suppose we have a dataset containing 60,000 distinct tuples where the sensitive attribute can take three distinct values and is not correlated with the non-sensitive attributes. A 5-anonymization of this table will have around 12,000 groups¹ and, on average, 1 out of every 81 groups will have no diversity (the values for the sensitive attribute will all be the same). Thus we should expect about 148 groups with no diversity. Therefore, information about 740 people would be compromised by a homogeneity attack. This suggests that in addition to k -anonymity, the sanitized table should also ensure “diversity” – all tuples that share the same values of their quasi-identifiers should have diverse values for their sensitive attributes.

The possibility of a homogeneity attack has been previously discussed in the literature (e.g., [100]). One solution to the homogeneity problem, as presented by Ohrn et al. [100], turns out to be a specific instance of our general principle of ℓ -diversity (see Section 3.4). For reasons that will become clear in Section 3.4, we refer to that method as *entropy ℓ -diversity*.

The next observation is that an adversary could use “background” knowledge to discover sensitive information.

Background Knowledge Attack: Alice has a pen-friend named Umeko who is admitted to the same hospital as Bruce, and whose patient records also appear in the table shown in Figure 3.2. Alice knows that Umeko is a 21 year-old Japanese female who currently lives in zip code 13068. Based on this informa-

¹Our experiments on real data sets show that data is often very skewed and a 5-anonymous table might not have so many groups

tion, Alice learns that Umeko's information is contained in record number 1,2,3, or 4. Without additional information, Alice is not sure whether Umeko caught a virus or has heart disease. However, it is well-known that Japanese have an extremely low incidence of heart disease. Therefore Alice concludes with near certainty that Umeko has a viral infection.

Observation 2 *k-Anonymity does not protect against attacks based on background knowledge.*

We have demonstrated (using the homogeneity and background knowledge attacks) that a k -anonymous table may disclose sensitive information. Since both of these attacks are plausible in real life, we need a stronger definition of privacy that takes into account diversity and background knowledge. This chapter addresses this very issue.

3.1.2 Contributions and Chapter Outline

We start by introducing an ideal notion of privacy called *Bayes-optimal* for the case that both data publisher and the adversary have knowledge of the complete joint distribution of the sensitive and nonsensitive attributes (Section 3.3). Unfortunately in practice, the data publisher is unlikely to possess all this information, and in addition, the adversary may have more specific background knowledge than the data publisher. Hence, while Bayes-optimal privacy sounds great in theory, it is unlikely that it can be guaranteed in practice. To address this problem, we show that the notion of Bayes-optimal privacy naturally leads to a novel *practical* criterion that we call ℓ -diversity. ℓ -Diversity provides privacy

even when the data publisher does not know what kind of knowledge is possessed by the adversary. The main idea behind ℓ -diversity is the requirement that the values of the sensitive attributes are well-represented in each group (Section 3.4). We show that existing algorithms for k -anonymity can be adapted to compute ℓ -diverse tables (Section 3.5), and we experimentally show that ℓ -diversity is practical and can be implemented efficiently (Section 3.6).

One of the main contributions of ℓ -diversity is the formal methodology used to reason about privacy. In Section 3.7 we present a more general framework for privacy definitions. This framework extends our formal reasoning presented in this chapter to help (a) design privacy definitions for new applications, and (b) compare different privacy metrics.

Before presenting our contributions, we introduce the notation needed to formally discuss data privacy in the next section.

3.2 Model and Notation

In this section we will introduce some basic notation that will be used in the remainder of the paper. We will also discuss how a table can be anonymized and what kind of background knowledge an adversary may possess.

Basic Notation. We assume that the microdata is in a single relational table $T\{t_1, t_2, \dots, t_n\}$ with attributes A_1, \dots, A_m , and that T is a subset of some larger population Ω . We also assume that each tuple $t_i \in T$ represents a unique individual from the population. For example, if T houses the in-patient records of Gotham City hospital, then Ω could be the population of the Gotham County.

Let \mathcal{A} denote the set of all attributes $\{A_1, A_2, \dots, A_m\}$ and $t[A_i]$ denote the value of attribute A_i for tuple t . If $\mathcal{C} = \{C_1, C_2, \dots, C_p\} \subseteq \mathcal{A}$ then we use the notation $t[\mathcal{C}]$ to denote the tuple $(t[C_1], \dots, t[C_p])$, which is the projection of t onto the attributes in \mathcal{C} .

In privacy-preserving data publishing, there exist several important subsets of \mathcal{A} . A *sensitive attribute* is an attribute whose value for any particular individual must be kept secret from people (e.g., researchers) who have no direct access to the original data. Let \mathcal{S} denote the set of all sensitive attributes. An example of a sensitive attribute is medical condition from Table 3.1. The association between individuals and medical condition should be kept secret; thus we should not disclose which particular patients have cancer, but it is permissible to disclose the information that there exist cancer patients in the hospital. We assume that the data publisher knows which attributes are sensitive. To simplify the discussion, for much of this paper we will also assume that there is only one sensitive attribute; the extension of our results to multiple sensitive attributes is not difficult and is handled in Section 3.4.3. All attributes that are not sensitive are called *nonsensitive* attributes. Let \mathcal{N} denote the set of nonsensitive attributes. We are now ready to formally define the notion of a quasi-identifier.

Definition 3.1 (Quasi-identifier) *A set of nonsensitive attributes $\{Q_1, \dots, Q_w\}$ of a table is called a quasi-identifier if these attributes can be linked with external data to uniquely identify at least one individual in the general population Ω .*

One example of a quasi-identifier is a primary key like social security number. Another example is the set $\{\text{Gender, Age, Zip Code}\}$ in the GIC dataset that was used to identify the governor of Massachusetts as described in the in-

troduction. Let us denote the set of all quasi-identifiers by \mathcal{QI} . We are now ready to formally define k -anonymity.

Definition 3.2 (k -Anonymity) *A table T satisfies k -anonymity if for every tuple $t \in T$ there exist $k - 1$ other tuples $t_{i_1}, t_{i_2}, \dots, t_{i_{k-1}} \in T$ such that $t[\mathcal{C}] = t_{i_1}[\mathcal{C}] = t_{i_2}[\mathcal{C}] = \dots = t_{i_{k-1}}[\mathcal{C}]$ for all $\mathcal{C} \in \mathcal{QI}$.*

The Anonymized Table T^* . Since the quasi-identifiers might uniquely identify tuples in T , the table T is not published; it is subjected to an *anonymization procedure* and the resulting table T^* is published instead.

There has been a lot of research on techniques for anonymization (see Chapter 5 for a discussion of related work). These techniques can be broadly classified into *generalization* techniques [7, 81], *generalization with tuple suppression* techniques [18, 114], and *data swapping and synthetic data generation* techniques [3, 4, 56, 106]. In this chapter we limit our discussion only to generalization techniques; synthetic data generation techniques will be the focus of the next chapter.

Definition 3.3 (Domain Generalization) *A domain $D^* = \{P_1, P_2, \dots\}$ is a generalization (partition) of a domain D if $\bigcup P_i = D$ and $P_i \cap P_j = \emptyset$ whenever $i \neq j$. For $x \in D$ we let $\phi_{D^*}(x)$ denote the element $P \in D^*$ that contains x .*

Note that we can create a partial order \prec_G on domains by requiring $D \prec_G D^*$ if and only if D^* is a generalization of D . Given a table $T = \{t_1, \dots, t_n\}$ with the set of nonsensitive attributes \mathcal{N} and a generalization D_N^* of domain(\mathcal{N}), we can construct a table $T^* = \{t_1^*, \dots, t_n^*\}$ by replacing the value of $t_i[\mathcal{N}]$ with the

generalized value $\phi_{D_N^*}(t_i[N])$ to get a new tuple t_i^* . The tuple t_i^* is called a *generalization* of the tuple t_i and we use the notation $t_i \xrightarrow{*} t_i^*$ to mean “ t_i^* generalizes t_i ”. Extending the notation to tables, $T \xrightarrow{*} T^*$ means “ T^* is a generalization of T ”. Typically, ordered attributes are partitioned into intervals, and categorical attributes are partitioned according to a user-defined hierarchy (for example, cities are generalized to counties, counties to states, and states to regions).

Example 3.1 (Continued) *Table 3.2 is a generalization of Table 3.1. We generalized on the Zip Code attribute by partitioning it into two sets: “1485*” (representing all zip codes whose first four digits are 1485) and “130**” (representing all zip codes whose first three digits are 130). Then we partitioned Age into three groups: “< 30”, “3*” (representing all ages between 30 and 39), and “≥ 40”. Finally, we partitioned Nationality into just one set “*” representing all nationalities.*

The Adversary’s Background Knowledge. Since the background knowledge attack was due to the adversary’s additional knowledge about the table, let us briefly discuss the type of background knowledge that we are modeling.

First, the adversary has access to the published table T^* and she knows that T^* is a generalization of some base table T . The adversary also knows the domain of each attribute of T .

Second, the adversary may know that some individuals are in the table. This knowledge is often easy to acquire. For example, GIC published medical data about all Massachusetts state employees. If the adversary Alice knows that her neighbor Bruce is a Massachusetts state employee then Alice is almost certain that Bruce’s information is contained in that table. In this case, we assume that Alice knows all of Bruce’s nonsensitive attributes. In addition, the adversary

could have knowledge about the sensitive attributes of specific individuals in the population and/or the table. For example, the adversary Alice might know that neighbor Bruce does not have pneumonia since Bruce does not show any of the symptoms of pneumonia. We call such knowledge “instance-level background knowledge,” since it is associated with specific instances in the table. In addition, Alice may know complete information about some people in the table other than Bruce (for example, Alice’s data may be in the table).

Third, the adversary could have partial knowledge about the distribution of sensitive and nonsensitive attributes in the population. We call this “demographic background knowledge.” For example, the adversary may know $P(t[\text{Condition}] = \text{“cancer”} \mid t[\text{Age}] \geq 40)$ and may use it to make additional inferences about records in the table.

Now armed with the right notation, let us start looking into principles and definitions of privacy that leak little information.

3.3 Bayes-Optimal Privacy

In this section we analyze an ideal notion of privacy. We call it *Bayes-Optimal Privacy* since it involves modeling background knowledge as a probability distribution over the attributes and uses Bayesian inference techniques to reason about privacy. We introduce tools for reasoning about privacy (Section 3.3.1), use them to discuss theoretical principles of privacy (Section 3.3.2), and then point out the difficulties that need to be overcome to arrive at a practical definition of privacy (Section 3.3.3).

3.3.1 Changes in Belief Due to Data Publishing

For simplicity of discussion, we combine all the nonsensitive attributes into a single, multi-dimensional quasi-identifier attribute Q whose values are generalized to create the anonymized table T^* from the base table T . Since Bayes-optimal privacy is only used to motivate a practical definition, we make the following two simplifying assumptions: first, we assume that T is a simple random sample from some larger population Ω (a sample of size n drawn without replacement is called a *simple random sample* if every sample of size n is equally likely); second, we assume that there is a single sensitive attribute. We would like to emphasize that both these assumptions will be dropped in Section 3.4 when we introduce a practical definition of privacy.

Recall that in our attack model, the adversary Alice has partial knowledge of the distribution of the sensitive and non-sensitive attributes. Let us assume a worst case scenario where Alice knows the complete joint distribution f of Q and S (i.e., she knows their frequency in the population Ω). Consider any individual Bruce that Alice knows is in the table. She knows that Bruce corresponds to a record $t \in T$ that has been generalized to a record $t^{*\star}$ in the published table T^* . She also knows the value of Bruce’s non-sensitive attributes (i.e., she knows that $t[Q] = q$). Alice’s goal is to use her background knowledge to discover Bruce’s sensitive information — the value of $t[S]$. We gauge her success using two quantities: Alice’s *prior belief*, and her *posterior belief*.

Alice’s *prior belief*, $\alpha_{(q,s)}$, that Bruce’s sensitive attribute is s given that his nonsensitive attribute is q , is just her background knowledge:

$$\alpha_{(q,s)} = P_f(t[S] = s \mid t[Q] = q)$$

After Alice observes the table T^* , her belief about Bruce's sensitive attribute changes. This new belief, $\beta_{(q,s,T^*)}$, is her *posterior belief*:

$$\beta_{(q,s,T^*)} = P_f \left(t[S] = s \mid t[Q] = q \wedge \exists t^* \in T^*, t \xrightarrow{*} t^* \right)$$

Given f and T^* , we can derive a formula for $\beta_{(q,s,T^*)}$ which will help us formulate our new privacy definition in Section 3.4. The main idea behind the derivation is to find a set of equally likely disjoint random worlds (like in [16]) such that a conditional probability $P(A|B)$ is the number of worlds satisfying condition $A \wedge B$ divided by the number of worlds satisfying condition B .

Theorem 3.1 *Let T^* be a published table which is obtained by performing generalizations on a table T ; let X be an individual with $X[Q] = q$ who appears in the table T (and also T^*); let q^* be the generalized value of q in T^* ; let s be a possible value of the sensitive attribute; let $n_{(q^*,s')}$ be the number of tuples $t^* \in T^*$ where $t^*[Q] = q^*$ and $t^*[S] = s'$; and let $f(s' \mid q^*)$ be the conditional probability of the sensitive attribute being s' conditioned on the fact that the nonsensitive attribute Q is some q' which can be generalized to q^* . Then the observed belief that $X[S] = s$ is given by:*

$$\beta_{(q,s,T^*)} = \frac{n_{(q^*,s)} \frac{f(s|q)}{f(s|q^*)}}{\sum_{s' \in S} n_{(q^*,s')} \frac{f(s'|q)}{f(s'|q^*)}} \quad (3.1)$$

Proof. For ease of reference, we review the notation used in this proof in Table 3.3. To help us model the adversary's uncertainty about the value of Bruce's sensitive attribute after seeing the anonymized table T^* , we will construct a set of *random worlds* such that T^* could have come from any one of these random worlds with equal probability. In all of these worlds, Bruce (or X , as we will call him in this proof) appears in T^* . In any two different random worlds, either some individual in the population has a different value for the sensitive

Table 3.3: Notation used in the proof of Theorem 3.1.

Notation	Description
T	Un-anonymized table
T^*	The anonymized table
Q	Domain of the quasi-identifier attribute
Q^*	Generalized domain of the quasi-identifier attribute
S	Domain of the sensitive attribute
Ω	Population of individuals
X	Bruce, the individual in the population Ω with $X[Q] = q$ and who is known to be in T
N_q	Number of individuals w in the population Ω such that $w[Q] = q$
$N_{(q,s)}$	Number of individuals w in the population Ω such that $w[Q] = q$ and $w[S] = s$
$N_{(q^*,s)}$	Number of individuals w in the population Ω such that $w[S] = s$ and $w[Q^*] = q^*$
n	Number of tuples in the anonymized table T^*
$n_{(q^*,s)}$	Number of tuples t^* in the anonymized table T^* such that $t^*[S] = s$ and $t^*[Q^*] = q^*$

attribute, or a different set of individuals appear in T^* . Since the random worlds are equally likely and mutually exclusive, the required conditional probability is the fraction of the total number of worlds in which $X[S] = s$ (as in [16]).

Constructing the set of random worlds:

Formally, a random world is a pair (ψ, Z_n) . $\psi : \Omega \rightarrow S$ is an assignment of sensitive values for each individual $\omega \in \Omega$. Z_n is a simple random sample of n

individuals from Ω . We are interested in only those assignments ψ which are consistent with the adversary's background knowledge. In particular, the adversary knows the size of Ω and the distribution of sensitive and nonsensitive attributes; i.e., for every (q, s) , the adversary knows $N_{(q,s)}$ – the number of individuals with nonsensitive attribute q who have sensitive value s . Therefore for every (q, s) , ψ should assign the value s to exactly $N_{(q,s)}$ out of the N_q individuals who have the nonsensitive value q . Note that in any two distinct assignments ψ_1, ψ_2 there is some individual ω such that $\psi_1(\omega) \neq \psi_2(\omega)$; i.e., ω is assigned to different values of S . Given only knowledge of the distribution of sensitive and nonsensitive attributes, the adversary has no preference for any of the ψ and, invoking the principle of indifference, considers each ψ to be equally likely.

The second component of a random world is Z_n . Z_n is a size n simple random sample from the population Ω . By the definition of a simple random sample, each Z_n is equally likely. Since the sample Z_n is picked independent of the assignment ψ , each random world (ψ, Z_n) is equally likely.

Each (ψ, Z_n) describes a table $T_{(\psi, Z_n)}$ containing n tuples with Q and S as attributes. We are interested in only those random worlds where X appears in $T_{(\psi, Z_n)}$ and where $T_{(\psi, Z_n)} \xrightarrow{*} T^*$. We can rephrase this condition as follows. We say that a random world (ψ, Z_n) is *compatible* with the published table T^* containing X , written as $(\psi, Z_n) \vdash (T^*, X)$, if the following two conditions hold:

1. $X \in Z_n$, where X is the individual with $X[Q] = q$ who is known to be in the table; and
2. for every (q^*, s) pair there are $n_{(q^*, s)}$ individuals ω in Z_n such that $\omega[Q]$ is generalized to q^* and such that $\psi(\omega) = s$.

The set of compatible random worlds completely characterizes the set of worlds which give rise to the anonymized table T^* containing X . It is clear that these worlds are equally likely. Also any two compatible random worlds are mutually exclusive because either some individual in the population is assigned a different value for S or the sample of individuals Z_n is different.

Calculating the conditional probability $\beta_{(q,s,T^*)}$:

To calculate the conditional probability $\beta_{(q,s,T^*)}$, we need to find the fraction of the total number of compatible random worlds in which X is assigned the sensitive value s . Let $\mathcal{T}_X^* = \{(\psi, Z_n) \vdash (T^*, X)\}$ be the set of random worlds which are compatible with T^* containing X . Let $\mathcal{T}_{(X,s)}^* = \{(\psi, Z_n) \vdash (T^*, X) \mid \psi(X) = s\}$ be the set of random worlds compatible with T^* where X is assigned the sensitive value s . Then,

$$\beta_{(q,s,T^*)} = \frac{|\mathcal{T}_{(X,s)}^*|}{|\mathcal{T}_X^*|}$$

Note that $\mathcal{T}_{(X,s_1)}^*$ and $\mathcal{T}_{(X,s_2)}^*$ are disjoint sets of random worlds – in all the worlds in $\mathcal{T}_{(X,s_1)}^*$, X is assigned the sensitive value s_1 and in all the world in $\mathcal{T}_{(X,s_2)}^*$, X is assigned the sensitive value s_2 . Thus

$$|\mathcal{T}_X^*| = \sum_{s' \in S} |\mathcal{T}_{(X,s')}^*|$$

We now proceed to calculate the cardinality of $\mathcal{T}_{(X,s)}^*$ for each s . First we will compute the number of assignments ψ such that $\psi(X) = s$ and then for each ψ we will compute the number of samples Z_n such that $(\psi, Z_n) \vdash (T^*, X)$. The number of assignments ψ compatible with the background knowledge such that $\psi(X) = s$ can be calculated as follows. X is assigned the sensitive value s . Since $X[Q] = q$, out of the remaining $N_q - 1$ individuals having the nonsensitive value q , $N_{(q,s)} - 1$ of them are assigned s . For every other sensitive value s' , $N_{(q,s')}$ out of the $N_q - 1$ individuals are assigned s' . For every $q' \neq q$ and every s' , some

$N_{(q',s')}$ out of the N'_q individuals having the nonsensitive value q' are assigned s' .

The number of these assignments is

$$\begin{aligned} & \frac{(N_q - 1)!}{(N_{(q,s)} - 1)! \prod_{s' \neq s} N_{(q,s')}!} \times \prod_{q' \neq q} \frac{N_{q'}!}{\prod_{s' \in S} N_{(q',s')}!} \\ &= \frac{N_{(q,s)}}{N_q} \prod_{q' \in Q} \frac{N_{q'}!}{\prod_{s' \in S} N_{(q',s')}!} \end{aligned} \quad (3.2)$$

For each mapping ψ such that $\psi(X) = s$, we count the number of Z_n 's such that $(\psi, Z_n) \vdash (T^*, X)$ as follows. Let q^* be the generalized value of $q = X[Q]$. X 's record will appear as $t_X^* = (q^*, s)$ in the table T^* . Apart from t_X^* , T^* contains $n_{(q^*,s)} - 1$ other tuples of the form (q^*, s) . Hence, apart from X , Z_n should contain $n_{(q^*,s)} - 1$ other individuals ω with $\psi(\omega) = s$ and $\omega[Q] = q'$ where q' generalizes to q^* . For all other (q^*, s') such that $q^* \neq q^*$ or $s' \neq s$, Z_n should contain $n_{(q^*,s')}$ individuals ω' where $\psi(\omega') = s'$ and q^* is the generalized value of $\omega[Q]$. The number of Z_n 's is given by

$$\begin{aligned} & \binom{N_{(q^*,s)} - 1}{n_{(q^*,s)} - 1} \times \prod_{(q^*,s') \in (Q^* \times S) - \{(q^*,s)\}} \binom{N_{(q^*,s')}}{n_{(q^*,s')}} \\ &= \frac{n_{q^*,s}}{N_{(q^*,s)}} \prod_{(q^*,s') \in Q^* \times S} \binom{N_{(q^*,s')}}{n_{(q^*,s')}} \end{aligned} \quad (3.3)$$

The cardinality of $\mathcal{T}_{(X,s)}^*$ is therefore the product of Equations 3.2 and 3.3 and can be expressed as

$$\begin{aligned} |\mathcal{T}_{(X,s)}^*| &= \frac{N_{(q,s)}}{N_q} \prod_{q' \in Q} \frac{N_{q'}!}{\prod_{s' \in S} N_{(q',s')}!} \times \frac{n_{q^*,s}}{N_{(q^*,s)}} \prod_{(q^*,s') \in Q^* \times S} \binom{N_{(q^*,s')}}{n_{(q^*,s')}} \\ &= n_{(q^*,s)} \frac{N_{(q,s)}}{N_{(q^*,s)}} \times \frac{1}{N_q} \prod_{q' \in Q} \frac{N_{q'}!}{\prod_{s' \in S} N_{(q',s')}!} \times \prod_{(q^*,s') \in Q^* \times S} \binom{N_{(q^*,s')}}{n_{(q^*,s')}} \\ &= n_{(q^*,s)} \frac{N_{(q,s)}}{N_{(q^*,s)}} \times \mathcal{E} \end{aligned}$$

The expression \mathcal{E} is the same for all $s' \in S$. Hence, the expression for the observed belief is

$$\begin{aligned}\beta_{(q,s,T^*)} &= \frac{|\mathcal{T}_{(X,s)}^*|}{\sum_{s' \in S} |\mathcal{T}_{(X,s')}^*|} \\ &= \frac{n_{(q^*,s)} \frac{N_{(q,s)}}{N_{(q^*,s)}}}{\sum_{s' \in S} n_{(q^*,s')} \frac{N_{(q,s')}}{N_{(q^*,s')}}}\end{aligned}$$

Using the substitutions $f(q, s) = N_{(q,s)}/N$ and $f(q^*, s) = N_{(q^*,s)}/N$, we get the required expression.

$$\begin{aligned}\beta_{(q,s,T^*)} &= \frac{n_{(q^*,s)} \frac{f(q,s)}{f(q^*,s)}}{\sum_{s' \in S} n_{(q^*,s')} \frac{f(q,s')}{f(q^*,s')}} \\ &= \frac{n_{(q^*,s)} \frac{f(s|q)}{f(s|q^*)}}{\sum_{s' \in S} n_{(q^*,s')} \frac{f(s'|q)}{f(s'|q^*)}}\end{aligned}$$

Note that in the special case when S and Q are independent, The expression for the observed belief simplifies to

$$\begin{aligned}\beta_{(q,s,T^*)} &= \frac{n_{(q^*,s)} \frac{f(s|q)}{f(s|q^*)}}{\sum_{s' \in S} n_{(q^*,s')} \frac{f(s'|q)}{f(s'|q^*)}} \\ &= \frac{n_{(q^*,s)} \frac{f(s)}{f(s)}}{\sum_{s' \in S} n_{(q^*,s')} \frac{f(s')}{f(s')}} \\ &= \frac{n_{(q^*,s)}}{\sum_{s' \in S} n_{(q^*,s')}}\end{aligned}$$

■

Armed with a way of calculating Alice's belief about Bruce's private data after she has seen T^* , let us now examine some principles for building definitions of privacy.

3.3.2 Privacy Principles

Given the adversary's background knowledge, a published table T^* might leak private information in two important ways: *positive disclosure* and *negative disclosure*.

Definition 3.4 (Positive disclosure) *Publishing the table T^* that was derived from T results in a positive disclosure if the adversary can correctly identify the value of a sensitive attribute with high probability; i.e., given a $\delta > 0$, there is a positive disclosure if $\beta_{(q,s,T^*)} > 1 - \delta$ and there exists $t \in T$ such that $t[Q] = q$ and $t[S] = s$.*

Definition 3.5 (Negative disclosure) *Publishing the table T^* that was derived from T results in a negative disclosure if the adversary can correctly eliminate some possible values of the sensitive attribute (with high probability); i.e., given an $\epsilon > 0$, there is a negative disclosure if $\beta_{(q,s,T^*)} < \epsilon$ and there exists a $t \in T$ such that $t[Q] = q$ but $t[S] \neq s$.*

The homogeneity attack in Section 3.1.1 where Alice determined that Bruce has cancer is an example of a positive disclosure. Similarly, in the example from Section 3.1.1, even without background knowledge Alice can deduce that Umeko does not have cancer. This is an example of a negative disclosure.

Note that not all positive disclosures are disastrous; if the prior belief was that $\alpha_{(q,s)} > 1 - \delta$, the adversary would not have learned anything new. Similarly, negative disclosures are not always bad: discovering that Bruce does not have Ebola might not be very serious because the prior belief of this event was small. Hence, the ideal definition of privacy can be based on the following principle:

Principle 3.1 (Uninformative Principle) *The published table should provide the adversary with little additional information beyond the background knowledge. In other words, there should not be a large difference between the prior and posterior beliefs.*

The uninformative principle can be instantiated in several ways, for example with the (ρ_1, ρ_2) -privacy breach definition [62].

Definition 3.6 ((ρ_1, ρ_2) -privacy) *Given a table T^* and two constants ρ_1 and ρ_2 , we say that a (ρ_1, ρ_2) -privacy breach has occurred when either $\alpha_{(q,s)} < \rho_1 \wedge \beta_{(q,s,T^*)} > \rho_2$ or when $\alpha_{(q,s)} > 1 - \rho_1 \wedge \beta_{(q,s,T^*)} < 1 - \rho_2$. If a (ρ_1, ρ_2) -privacy breach has not occurred, then table T^* satisfies (ρ_1, ρ_2) -privacy.*

An alternative privacy definition based on the uninformative principle would bound the maximum difference between $\alpha_{(q,s)}$ and $\beta_{(q,s,T^*)}$ using any of the functions commonly used to measure the difference between probability distributions. Any privacy definition that is based on the uninformative principle, and instantiated either by a (ρ_1, ρ_2) -privacy breach definition or by bounding the difference between $\alpha_{(q,s)}$ and $\beta_{(q,s,T^*)}$ is a Bayes-optimal privacy definition. The specific choice of definition depends on the application.

Note that any Bayes-optimal privacy definition captures diversity in addition to background knowledge. To see how it captures diversity, suppose that all the tuples whose nonsensitive attribute Q have been generalized to q^* have the same value s for their sensitive attribute. Then $n_{(q^*,s')} = 0$ for all $s' \neq s$ and hence the value of the observed belief $\beta_{(q,s,T^*)}$ becomes 1 in Equation 3.1. This will be flagged as a breach whenever the prior belief is not close to 1.

3.3.3 Limitations of the Bayes-Optimal Privacy

For the purposes of our discussion, we are more interested in the properties of Bayes-optimal privacy rather than its exact instantiation. In particular, Bayes-optimal privacy has several drawbacks that make it hard to use in practice.

Insufficient Knowledge. The data publisher is unlikely to know the full distribution f of sensitive and nonsensitive attributes over the general population Ω from which T is a sample.

The Adversary's Knowledge is Unknown. It is also unlikely that the adversary has knowledge of the complete joint distribution between the non-sensitive and sensitive attributes. However, the data publisher does not know how much the adversary knows. For example, in the background knowledge attack in Section 3.1.1, Alice knew that Japanese have a low incidence of heart disease, but the data publisher did not know that Alice knew this piece of information.

Instance-Level Knowledge. The theoretical definition does not protect against knowledge that cannot be modeled probabilistically. For example, suppose Bruce's son tells Alice that Bruce does not have diabetes. The theoretical definition of privacy will not be able to protect against such adversaries.

Multiple Adversaries. There will likely be multiple adversaries with different levels of knowledge, each of which is consistent with the full joint distribution. Suppose Bruce has a disease that is (a) very likely among people in the age group [30-50], but (b) is very rare for people of that age group who are doctors. An adversary who only knows the interaction of age and illness will think that it is very likely for Bruce to have that disease. However, an adversary who also knows that Bruce is a doctor is more likely to think that Bruce does not

have that disease. Thus, although additional knowledge can yield better inferences on average, there are specific instances where it does not. Thus the data publisher must take into account all possible levels of background knowledge.

3.4 ℓ -Diversity: Privacy beyond k -Anonymity

In this section we discuss how to overcome the difficulties outlined at the end of the previous section. We derive the ℓ -diversity principle (Section 3.4.1), show how to instantiate it with specific definitions of privacy (Section 3.4.2), outline how to handle multiple sensitive attributes (Section 3.4.3), and discuss how ℓ -diversity addresses the issues raised in the previous section (Section 3.4.4).

3.4.1 The ℓ -Diversity Principle

In this subsection we will derive the principle of ℓ -diversity in two ways. First, we will derive it in an ideal theoretical setting where it can be shown that the adversary's background knowledge will not lead to a privacy breach. Then we will re-derive the ℓ -diversity principle from a more practical starting point and show that even under less-than-ideal circumstances, ℓ -diversity can still defend against background knowledge that is unknown to the data publisher. Although the arguments in this subsection can be made precise, we will keep our discussion at an intuitive level for the sake of clarity.

Let us re-examine the expression for computing the adversary's observed

belief (Theorem 3.1):

$$\beta_{(q,s,T^*)} = \frac{n_{(q^*,s)} \frac{f(s|q)}{f(s|q^*)}}{\sum_{s' \in S} n_{(q^*,s')} \frac{f(s'|q)}{f(s'|q^*)}} \quad (3.4)$$

For the moment, let us consider an ideal setting where if two objects have “similar” nonsensitive attributes then their sensitive attributes have similar probabilistic behavior. More formally, given a similarity measure $d(\cdot, \cdot)$ then $\forall \epsilon > 0, \exists \delta$ such that if $d(q_1, q_2) < \delta$ then $\max_s |f(s|q_1) - f(s|q_2)| < \epsilon$. This similarity assumption is implicit in many data analysis algorithms, for instance in all k -Nearest Neighbor classifiers.

Now let us define a q^* -block to be the set of tuples in T^* whose nonsensitive attribute values generalize to q^* . If all tuples in a q^* -block are “similar” based on their nonsensitive attributes, then $f(s|q) \approx f(s|q^*)$ for those q that appear in the q^* -block, and because of (approximate) cancellations, Equation 3.4 could be approximated arbitrarily well by Equation 3.5:

$$L(q, s, T^*) = \frac{n_{(q^*,s)}}{\sum_{s' \in S} n_{(q^*,s')}} \quad (3.5)$$

Thus given enough data and a good partitioning, background knowledge cancels out and has no effect on the inferences that can be made from the table! The only inferences that can be made are those that depend solely on the $n_{(q^*,s')}$ – the frequencies of each $s' \in S$ for each q^* -block. Therefore to prevent privacy breaches, we need to ensure for every q^* -block that the ℓ most frequent values of S have roughly the same frequencies. This guarantees that $P(s|q^*) \leq 1/(\ell + \epsilon)$ for some small $\epsilon > 0$ and for all $s \in S$ and ensures that Alice will be uncertain about Bruce’s true medical condition. This is the essence of ℓ -diversity.

All of the above arguments relied on the following three assumptions: tuples

with similar non-sensitive attributes values have similar sensitive attributes values, there is a good partitioning of the data, and there is a large amount of data so that many “similar” tuples fall into each partition. Let us re-examine privacy breaches when these assumptions do not hold.

Recall that Theorem 3.1 allows us to calculate the observed belief of the adversary. Consider the case of positive disclosures; i.e., Alice wants to determine that Bruce has $t[S] = s$ with very high probability. From Theorem 3.1, this can happen only when:

$$\exists s, \forall s' \neq s, \quad n_{(q^*, s')} \frac{f(s'|q)}{f(s'|q^*)} \ll n_{(q^*, s)} \frac{f(s|q)}{f(s|q^*)} \quad (3.6)$$

The condition in Equation (3.6) could occur due to a combination of two factors: (i) a lack of diversity in the sensitive attributes in the q^* -block, and/or (ii) strong background knowledge. Let us discuss these in turn.

Lack of Diversity. Lack of diversity in the sensitive attribute manifests itself as follows:

$$\forall s' \neq s, \quad n_{(q^*, s')} \ll n_{(q^*, s)} \quad (3.7)$$

In this case, almost all tuples have the same value s for the sensitive attribute S , and thus $\beta_{(q, s, T^*)} \approx 1$. Note that this condition can be easily checked since it only involves counting the values of S in the published table T^* . We can ensure diversity by requiring that *all* the possible values $s' \in \text{domain}(S)$ occur in the q^* -block with roughly equal proportions. This, however, is likely to cause significant loss of information: if $\text{domain}(S)$ is large then the q^* -blocks will necessarily be large and so the data will be partitioned into a small number of q^* -blocks. Another way to ensure diversity and to guard against Equation 3.7 is to require that a q^* -block has at least $\ell \geq 2$ different sensitive values such that the ℓ most

frequent values (in the q^* -block) have roughly the same frequency. We say that such a q^* -block is *well-represented by ℓ sensitive values*.

Strong Background Knowledge. The other factor that could lead to a positive disclosure (Equation 3.6) is strong background knowledge. Even though a q^* -block may have ℓ “well-represented” sensitive values, Alice may still be able to use her background knowledge to eliminate sensitive values when the following is true:

$$\exists s', \quad \frac{f(s'|q)}{f(s'|q^*)} \approx 0 \quad (3.8)$$

This equation states that Bruce with quasi-identifier $t[Q] = q$ is much less likely to have sensitive value s' than any other individual in the q^* -block. For example, Alice may know that Bruce never travels, and thus he is extremely unlikely to have Ebola. It is not possible for a data publisher to reveal some information about the data while still guarding against attacks employing arbitrary amounts of background knowledge (since the revealed information may be precisely what the adversary needs to recreate the entire table). However, the data publisher can still guard against many attacks even without having access to Alice’s background knowledge. In our model, Alice might know the distribution $f(q, s)$ over the sensitive and non-sensitive attributes, in addition to the conditional distribution $f(s|q)$. The most damaging type of such information has the form $f(s|q) \approx 0$, e.g., “men do not have breast cancer”, or the form of Equation 3.8, e.g., “Japanese have a very low incidence of heart disease”. Note that *a priori* information of the form $f(s|q) = 1$ is not as harmful since this positive disclosure is independent of the published table T^* . Alice can also eliminate sensitive values with instance-level knowledge such as “Bruce does not have diabetes”.

In spite of such background knowledge, if there are ℓ “well represented” sensitive values in a q^* -block, then Alice needs $\ell - 1$ damaging pieces of background knowledge to eliminate $\ell - 1$ possible sensitive values and infer a positive disclosure! Thus, by setting the parameter ℓ , the data publisher can determine how much protection is provided against background knowledge — even if this background knowledge is unknown to the publisher.

Note that Alice may know ℓ pieces of instance-level background knowledge of the form “individual X_i does not have disease Y ” (for $i = 1 \dots \ell$), where each X_i is a different individual. However, we have been talking only about eliminating sensitive values for a single individual. It has been shown [92] that for a specific individual Bruce, the worst case disclosure occurs when $X_i = \text{Bruce}$ in all the ℓ pieces of information Alice possesses.

Moreover, when inferring information about Bruce, knowing the exact sensitive values of some other individuals in the table is less damaging than statements of the form “Bruce does not have cancer”. This is because knowing the sensitive value for some other individual only eliminates from consideration one tuple that may have corresponded to Bruce while the latter statement eliminates *at least* one tuple.

Putting these arguments together, we arrive at the following principle for protecting against adversaries with unknown background knowledge.

Principle 3.2 (ℓ -Diversity Principle) *A q^* -block is ℓ -diverse if contains at least ℓ “well-represented” values for the sensitive attribute S . A table is ℓ -diverse if every q^* -block is ℓ -diverse.*

Returning to our example, consider the inpatient records shown in Table 3.1.

Table 3.4: 3-Diverse Inpatient microdata.

	Non-Sensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	1305*	≤ 40	*	Heart Disease
4	1305*	≤ 40	*	Viral Infection
9	1305*	≤ 40	*	Cancer
10	1305*	≤ 40	*	Cancer
5	1485*	> 40	*	Cancer
6	1485*	> 40	*	Heart Disease
7	1485*	> 40	*	Viral Infection
8	1485*	> 40	*	Viral Infection
2	1306*	≤ 40	*	Heart Disease
3	1306*	≤ 40	*	Viral Infection
11	1306*	≤ 40	*	Cancer
12	1306*	≤ 40	*	Cancer

Table 3.4 is a 3-diverse version of the table. Comparing it with the 4-anonymous Table 3.2 we see that the attacks against the 4-anonymous table are prevented by the 3-diverse table. For example, Alice cannot infer from the 3-diverse table that Bruce (a 31 year old American from zip code 13053) has cancer. Even though Umeko (a 21 year old Japanese from zip code 13068) is extremely unlikely to have heart disease, Alice is still unsure whether Umeko has a viral infection or cancer.

The ℓ -diversity principle advocates ensuring ℓ “well represented” values for the sensitive attribute in every q^* -block, but does not clearly state what “well

represented” means. Note that we called it a “principle” instead of a definition — we will use it to give two concrete instantiations of the ℓ -diversity principle and discuss their relative trade-offs.

3.4.2 ℓ -Diversity: Instantiations

In this section we will give two instantiations of the ℓ -diversity principle: entropy ℓ -diversity and recursive ℓ -diversity. After presenting the basic definitions, we’ll extend them to cases where some positive disclosure is allowed.

The first instantiation of the ℓ -diversity principle, and the simplest one to describe, uses the information-theoretic notion of entropy:

Definition 3.7 (Entropy ℓ -Diversity [100]) *A table is Entropy ℓ -Diverse if for every q^* -block*

$$-\sum_{s \in S} p_{(q^*, s)} \log(p_{(q^*, s)}) \geq \log(\ell)$$

where $p_{(q^*, s)} = \frac{n_{(q^*, s)}}{\sum_{s' \in S} n_{(q^*, s')}}$ is the fraction of tuples in the q^* -block with sensitive attribute value equal to s .

As a consequence of the above condition, every q^* -block has at least ℓ distinct values for the sensitive attribute. Using this definition, Figure 3.4 is actually 2.8-diverse.

Entropy ℓ -diversity was first proposed by Ohrn et al. [100] as a way of defending against the homogeneity problem (without considering the role of background knowledge). Note that entropy ℓ -diversity captures the notion of well-represented groups due to the fact that entropy increases as frequencies become

more uniform. We can also capture the role of background knowledge more explicitly with an alternate definition.

Let s_1, \dots, s_m be the possible values of the sensitive attribute S in a q^* -block. Assume that we sort the counts $n_{(q^*, s_1)}, \dots, n_{(q^*, s_m)}$ in descending order and name the elements of the resulting sequence r_1, \dots, r_m . One way to think about ℓ -diversity is the following: the adversary needs to eliminate at least $\ell - 1$ possible values of S in order to infer a positive disclosure. This means that, for example, in a 2-diverse table, none of the sensitive values should appear too frequently. We say that a q^* -block is $(c, 2)$ -diverse if $r_1 < c(r_2 + \dots + r_m)$ for some user-specified constant c . For $\ell > 2$, we say that a q^* -block satisfies *recursive (c, ℓ) -diversity* if we can eliminate one possible sensitive value in the q^* -block and still have a $(c, \ell-1)$ -diverse block. This recursive definition can be succinctly stated as follows:

Definition 3.8 (Recursive (c, ℓ) -Diversity) *In a given q^* -block, let r_i denote the number of times the i^{th} most frequent sensitive value appears in that q^* -block. Given a constant c , the q^* -block satisfies recursive (c, ℓ) -diversity if $r_1 < c(r_\ell + r_{\ell+1} + \dots + r_m)$. A table T^* satisfies recursive (c, ℓ) -diversity if every q^* -block satisfies recursive ℓ -diversity. We say that 1-diversity is always satisfied.*

Now, both entropy and recursive ℓ -diversity may be too restrictive. To see why, let us first look at entropy ℓ -diversity. Since $-x \log(x)$ is a concave function, it can be shown that if we split a q^* -block into two sub-blocks q_a^* and q_b^* then $\text{entropy}(q^*) \geq \min(\text{entropy}(q_a^*), \text{entropy}(q_b^*))$. This implies that in order for entropy ℓ -diversity to be possible, the entropy of the entire table must be at least $\log(\ell)$. This might not be the case, especially if one value of the sensitive at-

tribute is very common – for example, if 90% of the patients have cancer as the value for the medical condition attribute.

This is also a problem with recursive ℓ -diversity. It is easy to see that if 90% of the patients have cancer as the value for the medical condition attribute then there will be at least one q^* -block where cancer will have frequency of at least 90%. Therefore if we choose $c < 9$ in Definition 3.8, no generalization of the base table will satisfy recursive (c, ℓ) -diversity.

On the other hand, some positive disclosures may be acceptable. For example, a clinic might be allowed to disclose that a patient has a cancer because it is well known that most patients who visit the clinic have heart problems. It may also be allowed to disclose that *medical condition = healthy* if this is not considered an invasion of privacy.

At this point one may be tempted to remove tuples with nonsensitive medical condition values, publish them unaltered, and then create an ℓ -diverse version of the remaining dataset. In some cases this is acceptable. However, there are three important issues why the above suggestion may not be acceptable: the anonymity of the unaltered tuples, the privacy of the remaining tuples, and the utility of the resulting published data.

First, publishing unaltered tuples gives an adversary the ability to link them to external data and identify the corresponding individuals. This may be considered a privacy breach [29], since it is reasonable for individuals to object to being identified as respondents in a survey. To avoid this, one could publish a k -anonymous version of tuples with nonsensitive medical condition values and a ℓ -diverse version of the rest of the table.

Second, separating individuals with nonsensitive medical conditions from the rest can impact the individuals with *sensitive* medical conditions. As an extreme case, suppose medical condition can only take two values: *healthy* and *sick*. There is no way to achieve 2-diversity on the table of patients that are sick; if Alice knows Bruce is in the table and Bruce is not listed as a healthy patient, he must then be sick. More generally, separating records with sensitive values from records with nonsensitive values reduces the possible choices for the security parameter ℓ .

A third issue with partitioning the data into two tables is related to the utility of the data for a researcher. Since each of the tables is smaller than the whole dataset, to satisfy k -anonymity and ℓ -diversity the tables might have to be generalized more than if a single table had been anonymized. For instance, consider a table reporting the gender and medical condition of 2,000 individuals, where the attribute medical condition can take three values: healthy, cancer, and hepatitis. Suppose in this table there are 1,000 males and 1,000 females; 700 of the 1,000 males are healthy and the other 300 have hepatitis, and 700 of the 1,000 females are healthy while the other 300 have cancer. If the disclosure of *medical condition = healthy* is not considered an invasion of privacy, then this table satisfies 2-diversity (and thus requires no further generalizations). In contrast, if we were to publish the healthy patients separately, we would need to suppress the gender information of the unhealthy individuals in order to achieve 2-diversity on the table containing the unhealthy patients. Additionally, if the data is separated then the two resulting tables are likely to have different schemas. For example, one table may be generalized so that age appears as an interval of length 5 (i.e. [30-34]) and only the first 4 digits of zip code are given, while the second table may give the full zip code but may generalize age to intervals of

length 10. Learning from such data is not as straightforward as learning from a single table, and in some cases can be intractable [86].

Thus an alternate approach is needed to handle the case when some of the values in the domain of the sensitive attribute need not be kept private. To capture this notion that some positive disclosure is acceptable, let Y be the set of those sensitive values for which positive disclosure is allowed. We call Y a *don't-care* set; we are not worried about those values being too frequent. Let s_y be the most frequent sensitive value in the q^* -block that is *not* in Y and let r_y be the associated frequency. Then the q^* -block satisfies ℓ -diversity if we can eliminate the $\ell - 2$ most frequent values of S *not including* r_y without making s_y too frequent in the resulting set. Thus, if we remove the sensitive values with counts r_1, \dots, r_{y-1} , then the result is $(\ell - y + 1)$ -diverse. This brings us to the following definition.

Definition 3.9 (Positive Disclosure-Recursive (c, ℓ) -Diversity). *Let $Y \subset S$ be a don't-care set. In a given q^* -block, let the most frequent sensitive value not in Y be the y^{th} most frequent sensitive value. Let r_i denote the frequency of the i^{th} most frequent sensitive value in the q^* -block. Such a q^* -block satisfies pd-recursive (c, ℓ) -diversity when one of the following holds:*

- *when $y \leq \ell - 1$ and $r_y < c \sum_{j=\ell}^m r_j$*
- *when $y > \ell - 1$ and $r_y < c \sum_{j=\ell-1}^{y-1} r_j + c \sum_{j=y+1}^m r_j$*

We denote the summations on the right hand side of the both conditions by $\text{tail}_{q^}(s_y)$.*

Now, note that if $r_y = 0$ then the q^* -block only has sensitive values that can be disclosed and so both conditions in Definition 3.9 are trivially satisfied.

Second, note that if $c > 1$ then the second condition clearly reduces to just the condition $y > \ell - 1$ because $r_y \leq r_{\ell-1}$. The second condition states that even though the $\ell - 1$ most frequent values can be disclosed, we still do not want r_y to be too frequent if $\ell - 2$ of them have been eliminated (i.e., we want the result to be 2-diverse).

To see this definition in action, suppose there are two values for medical condition, *healthy* and *sick*. If *healthy* is a don't-care value, then $(c, 2)$ -diversity states that the number of sick patients in a q^* -block is less than c times the number of healthy patients or, equivalently, at most $\frac{c}{c+1}$ patients in a q^* -block are sick. Thus if $c = 0.03$ then at most 3% of the patients in any q^* -block are not healthy, and if $c = 1$ then at most half the patients in any q^* -block are not healthy.

Entropy ℓ -diversity can also be extended to handle don't-care sets. The entropy ℓ -diversity with don't-care sets can be analogously defined; its description is a bit more involved, and the interested reader may find the details in Machanavajhala et al. [91].

Until now we have treated negative disclosure as relatively unimportant compared to positive disclosure. However, negative disclosure may also be important. If W is the set of values for the sensitive attribute for which negative disclosure is not allowed then, given a user-specified constant $c_2 < 100$, we require that each $s \in W$ appear in at least c_2 -percent of the tuples in every q^* -block, resulting in the following definition. This is incorporated into ℓ -diversity definitions in a straightforward way:

Definition 3.10 (NPD-Recursive (c_1, c_2, ℓ) -Diversity). *Let W be the set of sensitive values for which negative disclosure is not allowed. A table satisfies negative/positive*

disclosure-recursive (c_1, c_2, ℓ) -diversity (npd-recursive (c_1, c_2, ℓ) -diversity) if it satisfies pd -recursive (c_1, ℓ) -diversity and if every $s \in W$ occurs in at least c_2 percent of the tuples in every q^* -block.

3.4.3 Multiple Sensitive Attributes

Multiple sensitive attributes present some additional challenges. Suppose S and V are two sensitive attributes, and consider the q^* -block with the following tuples: $\{(q^*, s_1, v_1), (q^*, s_1, v_2), (q^*, s_2, v_3), (q^*, s_3, v_3)\}$. This q^* -block is 3-diverse (actually recursive (2,3)-diverse) with respect to S (ignoring V) and 3-diverse with respect to V (ignoring S). However, if we know that Bruce is in this block and his value for S is not s_1 then his value for attribute V cannot be v_1 or v_2 , and therefore must be v_3 . One piece of information destroyed his privacy. Thus we observe that a q^* -block that is ℓ -diverse in each sensitive attribute separately may still violate the principle of ℓ -diversity.

Intuitively, the problem occurred because within the q^* -block, V was not well-represented for each value of S . Had we treated S as part of the quasi-identifier when checking for diversity in V (and vice versa), we would have ensured that the ℓ -diversity principle held for the entire table. Formally,

Definition 3.11 (Multi-Attribute ℓ -Diversity) *Let T be a table with nonsensitive attributes Q_1, \dots, Q_{m_1} and sensitive attributes S_1, \dots, S_{m_2} . We say that T is ℓ -diverse if for all $i = 1 \dots m_2$, the table T is ℓ -diverse when S_i is treated as the sole sensitive attribute and $\{Q_1, \dots, Q_{m_1}, S_1, \dots, S_{i-1}, S_{i+1}, \dots, S_{m_2}\}$ is treated as the quasi-identifier.*

As the number of sensitive attributes grows, it is not hard to see that we will necessarily need larger and larger q^* -blocks to ensure diversity. This problem may be ameliorated through tuple suppression, generalization on the sensitive attributes, and publishing marginals (rather than the full table) containing different sensitive attributes. This is a subject for future work.

3.4.4 Discussion

Recall that we started our journey into Section 3.4 motivated by the weaknesses of Bayes-optimal privacy. Let us now revisit these issues one by one.

- ℓ -Diversity no longer requires knowledge of the full distribution of the sensitive and nonsensitive attributes.
- ℓ -Diversity does not even require the data publisher to have as much information as the adversary. The parameter ℓ protects against more knowledgeable adversaries; the larger the value of ℓ , the more information is needed to rule out possible values of the sensitive attribute.
- Instance-level knowledge (Bruce's son tells Alice that Bruce does not have diabetes) is automatically covered. It is treated as just another way of ruling out possible values of the sensitive attribute.
- Different adversaries can have different background knowledge leading to different inferences. ℓ -Diversity simultaneously protects against all of them without the need for checking which inferences can be made with which levels of background knowledge.

Overall, we believe that ℓ -diversity is practical, easy to understand, and addresses the shortcomings of k -anonymity with respect to the background knowledge and homogeneity attacks. Let us now see whether we can give efficient algorithms to implement ℓ -diversity. We will see that, unlike Bayes-optimal privacy, ℓ -diversity possesses a property called *monotonicity*. We define this concept in Section 3.5, and we show how this property can be used to efficiently generate ℓ -diverse tables.

3.5 Implementing Anonymous Data Publishing

In this section we discuss how to build algorithms for ℓ -diverse anonymous data publishing using domain generalization. Let us first review the search space for anonymous data publishing using domain generalization [18, 81, 82]. For ease of explanation, we will combine all the nonsensitive attributes into a single multi-dimensional attribute Q . For attribute Q , there is a user-defined generalization lattice. Formally, we define a generalization lattice to be a set of domains partially ordered by a generalization relation \prec_G (as described in Section 2.2.1). The bottom element of this lattice is $\text{domain}(Q)$ and the top element is the domain where each dimension of Q is generalized to a single value. Given a table T , each domain D_Q^* in the lattice defines an anonymized table T^* which is constructed by replacing each tuple $t \in T$ by the tuple t^* , such that the value $t^*[Q] \in D_Q^*$ is the generalization of the value $t[Q] \in \text{domain}(Q)$. An algorithm for data publishing should find a point on the lattice such that the corresponding generalized table T^* preserves privacy and retains as much utility as possible. In the literature, the utility of a generalized table is usually defined as a distance metric on the lattice – the closer the lattice point is to the

bottom, the larger the utility of the corresponding table T^* . Hence, finding a suitable anonymized table T^* is essentially a lattice search problem. There has been work on search strategies for k -anonymous tables that explore the lattice top-down [18] or bottom-up [81, 82].

In general, searching the entire lattice is computationally intractable. However, lattice searches can be made efficient if there is a stopping condition of the form: if T^* preserves privacy then every generalization of T^* also preserves privacy [81, 114]. This is called the *monotonicity property*, and it has been used extensively in frequent itemset mining algorithms [11]. k -Anonymity satisfies the monotonicity property, and it is this property which guarantees the correctness of all efficient algorithms [18, 81, 82]. Thus, if we show that ℓ -diversity also possesses the monotonicity property, then we can re-use these efficient lattice search algorithms to find the ℓ -diverse table with optimal utility. The same cannot be said of Bayes-optimal privacy; the following theorem gives a computational reason why Bayes-optimal privacy does not lend itself to efficient algorithmic implementations.

Theorem 3.2 *Bayes-optimal privacy does not satisfy the monotonicity property.*

Proof. We shall prove this theorem for the (ρ_1, ρ_2) version of the Bayes-optimal privacy definition (see Definition 3.6 and [62]); the proof can easily be extended to other instantiations. We set $\rho_1 = 0.31$ and $\rho_2 = 0.58$ and we will create an example where the prior belief $a_{(q,s)} < \rho_1$ but the observed belief is $\beta_{(q,s,T^*)} > \rho_2$.

First consider Table 3.5 which shows a base table T with two values for Q and two values for S . Based on this information, we can compute the prior and observed beliefs for table T :

Table 3.5: Table T for proof of Theorem 3.2.

	q_1	q_2
s_1	$f(q_1, s_1) = .15$ $n_{(q_1, s_1)} = 1$	$f(q_2, s_1) = .25$ $n_{(q_2, s_1)} = 35$
s_2	$f(q_1, s_2) = .35$ $n_{(q_1, s_2)} = 1$	$f(q_2, s_2) = .25$ $n_{(q_2, s_2)} = 15$

Table 3.6: Table T^* for proof of Theorem 3.2.

	q^*
s_1	$f(q^*, s_1) = .4$ $n_{(q^*, s_1)} = 36$
s_2	$f(q^*, s_2) = .6$ $n_{(q^*, s_2)} = 16$

- $\alpha_{(q_1, s_1)} = .3, \beta_{(q_1, s_1, T)} = .5$
- $\alpha_{(q_1, s_2)} = .7, \beta_{(q_1, s_2, T)} = .5$
- $\alpha_{(q_2, s_1)} = .5, \beta_{(q_2, s_1, T)} = .7$
- $\alpha_{(q_2, s_2)} = .5, \beta_{(q_2, s_2, T)} = .3$

Clearly, publishing T does not breach privacy. However, suppose we generalized T by generalizing both q_1 and q_2 to q^* , as in Table 3.6:

If Bob has nonsensitive value q_1 , then as before, $\alpha_{(q_1, s_1)} = .3 < \rho_1$. However,

$$\beta_{(q_1, s_1, T^*)} = \frac{36 \cdot \frac{.15}{.4}}{36 \cdot \frac{.15}{.4} + 16 \cdot \frac{.35}{.6}} > \frac{13.5}{13.5 + 9.34} > .59 > \rho_2$$

Thus while publishing T would not cause a privacy breach, publishing T^* would. This counterexample proves that Bayes-optimal privacy is not monotonic. ■

This seemingly counterintuitive result has a simple explanation. Note that there are many more tuples t with $t[Q] = q_2$ than there are with $t[Q] = q_1$. This causes the probabilistic behavior of the q^* -block in T^* to be heavily influenced by the tuples with $t[Q] = s_2$ and so it “pulls” the value of $\beta_{(q_1, s_1, T^*)} = \beta_{(q_2, s_1, T^*)}$ closer to $\beta_{(q_2, s_1, T)}$ (this can be verified with Equation 3.1 for observed belief). Since the prior belief $\alpha_{(q_1, s_1)}$ doesn’t change, and $\alpha_{(q_1, s_1)}$ and $\alpha_{(q_2, s_1)}$ are very different, we get a privacy breach from publishing T^* but not from publishing T .

Theorem 3.3 (Monotonicity of entropy ℓ -diversity) *Entropy ℓ -diversity satisfies the monotonicity property: if a table T^* satisfies entropy ℓ -diversity, then any generalization T^{**} of T^* also satisfies entropy ℓ -diversity.*

Theorem 3.3 follows from the fact that entropy is a concave function. Thus if the q^* -blocks q_1^*, \dots, q_d^* from table T^* are merged to form the q^* -block q^{**} of table T^{**} , then the entropy(q^{**}) $\geq \min_i(\text{entropy}(q_i^*))$.

Theorem 3.4 (Monotonicity of npd recursive ℓ -diversity) *The npd recursive (c_1, c_2, ℓ) -diversity criterion satisfies the monotonicity property: if a table T^* satisfies npd recursive (c_1, c_2, ℓ) -diversity, then any generalization T^{**} of T^* also satisfies npd recursive (c_1, c_2, ℓ) -diversity.*

Proof. We shall prove this for the case where T^{**} is derived from T^* by merging two q^* -blocks; the general case follows by induction. Let q_a^* and q_b^* be the q^* -blocks of T^* that are merged to form the q^* -block q^{**} of table T^{**} . The frequencies

of the sensitive values in q^{**} is the sum of the corresponding frequencies in q_a^* and q_b^* .

First, let us consider negative disclosures. If every sensitive value $s \in W$ occurs in at least c_2 percent of the tuples in q_a^* and q_b^* , then surely s should also occur in at least a c_2 percent of the tuples in the q^{**} .

Next let us consider positive disclosures. Let Y be the set of sensitive values for which positive disclosure is allowed. Let s_y be the most frequent sensitive value in q^{**} that does not appear in Y . Let s_{y_a} and s_{y_b} be the most frequent sensitive values in q_a^* and q_b^* , respectively, which are not in Y . Clearly if r_y , r_{y_a} and r_{y_b} are the respective counts, then

$$r_y \leq r_{y_a} + r_{y_b}$$

We also know that the q_a^* -blocks and q_b^* -block are (c_1, ℓ) -diverse (by hypothesis). Hence,

$$r_{y_a} \leq c_1 \text{tail}_{q_a^*}(s_{y_a})$$

$$r_{y_b} \leq c_1 \text{tail}_{q_b^*}(s_{y_b})$$

We are done if we prove that $r_y \leq c_1 \text{tail}_{q^{**}}(s_y)$. Since s_{y_a} is at least as frequent as s_y in q_a^* (and similarly for s_{y_b}) then by the definition of tail_{q^*} , we have

$$\text{tail}_{q_a^*}(s_y) \geq \text{tail}_{q_a^*}(s_{y_a})$$

$$\text{tail}_{q_b^*}(s_y) \geq \text{tail}_{q_b^*}(s_{y_b})$$

$$\text{tail}_{q^{**}}(s_y) = \text{tail}_{q_a^*}(s_y) + \text{tail}_{q_b^*}(s_y)$$

Hence

$$\begin{aligned}
r_y &\leq r_{y_a} + r_{y_b} \\
&\leq c_1(\text{tail}_{q_a^*}(s_{y_a}) + \text{tail}_{q_b^*}(s_{y_b})) \\
&\leq c_1(\text{tail}_{q_a^*}(s_y) + \text{tail}_{q_b^*}(s_y)) \\
&= c_1 \text{tail}_{q^{**}}(s_y)
\end{aligned}$$

and so the q^* -block q^{**} is $\text{npd}(c_1, c_2, \ell)$ -diverse. ■

We can also show that entropy ℓ -diversity with don't-care sets satisfies the monotonicity property using analogous concavity arguments and is therefore amenable to efficient algorithms. We refer the interested reader to Machanavajjhala et al. [91] for the details.

Thus to create an algorithm for ℓ -diversity, we can take an algorithm for k -anonymity that performs a lattice search and we make the following change: every time a table T^* is tested for k -anonymity, we check for ℓ -diversity instead. Since ℓ -diversity is a property that is local to each q^* -block and since all ℓ -diversity tests are solely based on the counts of the sensitive values, this test can be performed very efficiently.

We emphasize that this is only one way of generating ℓ -diverse tables and it is motivated by the structural similarities between k -anonymity and ℓ -diversity. Alternatively, one can post-process a k -anonymous table and suppress groups that are not ℓ -diverse or suppress tuples in groups until all groups are ℓ -diverse; one can directly modify a k -anonymity algorithm that uses suppression into an ℓ -diversity algorithm; or one can devise a completely new algorithm.

Table 3.7: Description of Adult database.

	Attribute	Domain size	Generalizations type	Ht.
1	Age	74	ranges-5,10,20	4
2	Gender	2	Suppression	1
3	Race	5	Suppression	1
4	Marital Status	7	Taxonomy tree	2
5	Education	16	Taxonomy tree	3
6	Native Country	41	Taxonomy tree	2
7	Work Class	7	Taxonomy tree	2
8	Salary class	2	<i>Sensitive att.</i>	
9	Occupation	14	<i>Sensitive att.</i>	

3.6 Experiments

In our experiments, we used an implementation of Incognito, as described in [81], for generating k -anonymous tables. We modified this implementation so that it produces ℓ -diverse tables as well. Incognito is implemented in Java and uses the database manager IBM DB2 v8.1 to store its data. All experiments were run under Linux (Fedora Core 3) on a machine with a 3 GHz Intel Pentium 4 processor and 1 GB RAM.

We ran our experiments on the Adult Database from the UCI Machine Learning Repository [108] and the Lands End Database. The Adult Database contains 45,222 tuples from US Census data and the Lands End Database contains 4,591,581 tuples of point-of-sale information. We removed tuples with missing

Table 3.8: Description of Lands End database.

	Attribute	Domain size	Generalizations type	Ht.
1	Zipcode	31953	Round each digit	5
2	Order date	320	Taxonomy tree	3
3	Gender	2	Suppression	1
4	Style	1509	Suppression	1
5	Price	346	Round each digit	4
6	Quantity	1	Suppression	1
7	Shipment	2	Suppression	1
8	Cost	147	<i>Sensitive att.</i>	

values and adopted the same domain generalizations as [81]. Tables 3.7 and 3.8 provide a brief description of the data including the attributes we used, the number of distinct values for each attribute, the type of generalization that was used (for non-sensitive attributes), and the height of the generalization hierarchy for each attribute.

Homogeneity Attack. In Tables 3.9, 3.10, 3.11, and 3.12, we illustrate the *homogeneity* attacks on k -anonymized datasets using the Lands End and Adult databases. For the Lands End Database, we treated $\{Zipcode, Order Date, Gender, Style, Price\}$ as the quasi-identifier. We partitioned the *Cost* attribute into 147 buckets by rounding to the nearest 100 and used this as the sensitive attribute. For the Adult Database, we used $\{Age, Gender, Race, Marital Status, Education\}$ as the quasi-identifier and *Salary Class* as the sensitive attribute. For values of

Table 3.9: Effect of homogeneity attack on the Adult database.

k	Affected /Total tables	Avg. Gps. Affected	Avg. Tuples Affected
2	8/8	7.38	558.00
5	11/12	3.58	381.58
10	10/12	1.75	300.42
15	7/8	2.12	317.25
20	8/10	1.20	228.20
30	7/10	0.90	215.40
50	5/5	1.00	202.80

$k = 2, 5, 10, 15, 20, 30, 50$, we then generated all k -anonymous tables that were minimal with respect to the generalization lattice (i.e. no table at a lower level of generalization was k -anonymous).

Tables 3.9 and 3.10 show an analysis of groups in k -anonymous tables that are completely homogeneous in the Adult and Lands End databases, respectively, while Tables 3.11 and 3.12 show a corresponding analysis of groups in k -anonymous tables that are “nearly” homogeneous (i.e., the most frequent sensitive value s in a group appears in at least 95% of the tuples in the group). Both cases should be avoided since an adversary would believe, with near certainty, that an individual in a homogeneous or nearly homogeneous group has the sensitive value s that appears most frequently. Note that the minority (i.e., $\leq 5\%$) of the individuals in nearly homogeneous groups whose sensitive values are not s are also affected even though the best inference about them (that they have s) is wrong. As a concrete example, consider the case when $s = AIDS$. An in-

Table 3.10: Effect of homogeneity attack on the Lands End database.

k	Affected /Total tables	Avg. Gps. Affected	Avg. Tuples Affected
2	2/3	12.3	2537.6
5	2/3	12.3	2537.6
10	2/2	18.5	3806.5
15	2/2	18.5	3806.5
20	1/2	2.5	1750
30	1/2	2.5	1750
50	1/3	0.6	1156

dividual that values privacy would not want to be associated with s with near certainty regardless of whether the true value is s . In the four tables shown in Tables 3.9, 3.10, 3.11, and 3.12, the first column indicates the value of k . The second column shows the number of minimal k -anonymous tables that have groups that are completely homogeneous (Tables 3.9 and 3.10) or 95% homogeneous (Tables 3.11 and 3.12). The third column shows the average number of such groups per minimal k -anonymous table. The fourth column shows the average number of tuples per minimal k -anonymous table that were affected by the two homogeneity attacks. As we can see from Tables 3.9, 3.10, 3.11 and 3.12, the homogeneity attack is a real concern, affecting a very large fraction of both datasets. Even for relatively large values of k (such as 30 and 50), many tables still had nearly homogeneous groups.

Note that the average number of affected groups, average number of affected tuples, etc., are not strictly decreasing functions of k . In particular, tables with

Table 3.11: Effect of 95% homogeneity attack on the Adult database.

k	Affected /Total tables	Avg. Gps. Affected	Avg. Tuples Affected
2	8/8	20.50	13574.5
5	12/12	12.67	13328.3
10	12/12	7.83	10796.5
15	8/8	8.88	12009.4
20	10/10	7.10	11041.0
30	10/10	5.50	11177.0
50	5/5	5.80	8002.0

small values of affected tuples are sometimes close to each other in the lattice of k -anonymous tables and may be generalized to the same table when k increases (thus reducing the total number of “safe” tables).

Performance. In our next set of experiments, we compare the running times of entropy ℓ -diversity and k -anonymity. The results are shown in Figures 3.1 and 3.2. For the Adult Database, we used *Occupation* as the sensitive attribute, and for Lands End we used *Cost*. We varied the quasi-identifier size from 3 attributes up to 8 attributes; a quasi-identifier of size j consisted of the first j attributes of its dataset as listed in Tables 3.7 and 3.8. We measured the time taken to return all 6-anonymous tables and compared it to the time taken to return all 6-diverse tables. In both datasets, the running times for k -anonymity and ℓ -diversity were similar. Sometimes the running time for ℓ -diversity was faster, which happened when the algorithm pruned parts of the generalization

Table 3.12: Effect of 95% homogeneity attack on the Lands End database.

k	Affected /Total tables	Avg. Gps. Affected	Avg. Tuples Affected
2	2/3	13.0	2825.33
5	2/3	13.0	2825.33
10	2/2	19.5	4238.00
15	2/2	19.5	4238.00
20	1/2	3.0	2119.00
30	1/2	3.0	2119.00
50	1/3	1.0	1412.66

lattice earlier than it did for k -anonymity.

Utility. The next set of experiments compare the utility of anonymized tables which are k -anonymous, entropy ℓ -diverse, or recursive $(3, \ell)$ -diverse. We use the Adult Database in all the experiments with sensitive attribute *Occupation*. For the purposes of comparison, we set $k = \ell$ and experimented with the following values of ℓ (and hence k): 2, 4, 6, 8, 10. The sensitive attribute *Occupation* takes only 14 values. Hence, there is no table which can be more than 14-diverse for any reasonable definition of diversity. Since some of the values appeared very infrequently, we found that there is no generalization of the Adult Database that is recursive $(3, \ell)$ -diverse for $\ell = 12$. We also found that the marginal distribution of the sensitive attribute is entropy 10.57-diverse. This means that no generalization of the Adult Database can be more than entropy 10.57-diverse unless the entire data set is suppressed.

The utility of a dataset is difficult to quantify. As a result, we used four

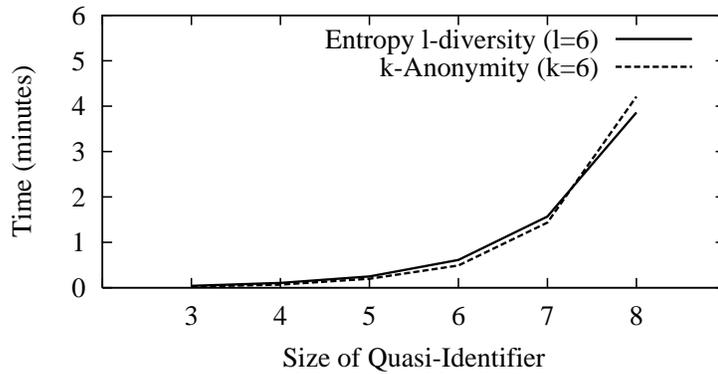


Figure 3.1: Performance of ℓ -diverse data publishing on Adult database.

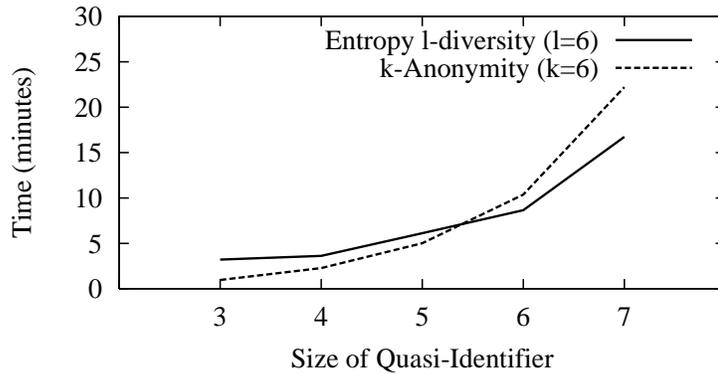


Figure 3.2: Performance of ℓ -diverse data publishing on Lands End database.

different metrics to gauge the utility of the generalized tables – generalization height, average group size, discernibility, and KL-divergence. The first metric, generalization height [81, 113], is the height of an anonymized table in the generalization lattice; intuitively, it is the number of generalization steps that were performed. The second metric is the average size of the q^* -blocks generated by the anonymization algorithm. The third metric, *discernibility* [18], measures the number of tuples that are indistinguishable from each other. Each tuple in a q^* block B_i incurs a cost $|B_i|$ and each tuple that is completely suppressed incurs

a cost $|D|$ (where D is the original dataset). Since we did not perform any tuple suppression, the discernibility metric is equivalent to the sum of the squares of the sizes of the q^* -blocks.

Neither generalization height, nor average group size, nor discernibility take the data distribution into account. For this reason we also use the KL-divergence metric described next. In many data mining tasks, we would like to use the published table to estimate the joint distribution of the attributes. Now, given a table T with categorical attributes A_1, \dots, A_m , we can view the data as an i.i.d. sample from an m -dimensional distribution F . We can estimate this F with the empirical distribution \hat{F} , where $\hat{F}(x_1, \dots, x_m)$ is the fraction of tuples t in the table such that $t.A_1 = x_1, \dots, t.A_m = x_m$. When a generalized version of the table is published, the estimate changes to \hat{F}^* by taking into account the generalizations used to construct the anonymized table T^* (and making the uniformity assumption for all generalized tuples sharing the same attribute values). If the tuple $t = (x_1, \dots, x_m)$ is generalized to $t^* = (x_1^*, \dots, x_m^*)$, then $\hat{F}^*(x_1, \dots, x_m)$ is given by

$$\hat{F}^*(x_1, \dots, x_m) = \frac{|\{t^* \in T^*\}|}{|T^*| \times \text{area}(t^*)}$$

where, $\text{area}(x_1^*, \dots, x_m^*) = \prod_{i=1}^m |\{x_i \in A_i \mid x_i \text{ is generalized to } x_i^*\}|$

To quantify the difference between the two distributions \hat{F} and \hat{F}^* , we use the Kullback-Leibler divergence (KL-divergence) which is defined as

$$\sum_{\mathbf{x} \in A_1 \times \dots \times A_m} \hat{F}(\mathbf{x}) \log \frac{\hat{F}(\mathbf{x})}{\hat{F}^*(\mathbf{x})}$$

where $0 \log 0$ is defined to be 0. The KL-divergence is non-negative and is 0 only when the two estimates are identical.

In Figures 3.3, 3.4, 3.5, and 3.6, we show the minimum generalization height, average group size, and discernibility of k -anonymous, entropy ℓ -diverse, and recursive $(3, \ell)$ -diverse tables for $\ell = k = 2, 4, 6, 8, 10$, while Figures 3.7 and 3.8 show our results for KL-divergence. For each of the graphs in Figures 3.3, 3.4, 3.5, 3.6, and 3.7, we performed the anonymizations on a 5% subsample of the original data, while Figure 3.8 shows results for anonymization of the entire data set.

Before explaining why it was necessary to subsample the data, we should first note that in general, the graphs show that ensuring diversity in the sensitive attribute does not require many more generalization steps than for k -anonymity (note that an ℓ -diverse table is automatically ℓ -anonymous); the minimum generalization heights for identical values of k and ℓ were usually identical. Nevertheless, we found that generalization height was not an ideal utility metric because tables with small generalization heights can still have very large group sizes. For example, using full-domain generalization on the Adult Database with the quasi-identifier $\{Age, Gender, Race, Marital Status, Education\}$, we found minimal (with respect to the generalization lattice) 4-anonymous tables that had average group sizes larger than 1,000 tuples. The large groups were caused by data skew. For example, there were only 114 tuples with age between 81 and 90, while there were 12,291 tuples with age between 31 and 40. So if age groups of length 5 (i.e. [1-5], [6-10], [11-15], etc) were generalized to age groups of length 10 (i.e. [1-10], [11-20], etc), we would end up with very large q^* -blocks.²

Thus, to better understand the loss of utility due to domain generalization, we chose to study a subsample of the Adult Database with a lesser data skew

²Generalization hierarchies that are aware of data skew may yield higher quality anonymizations. This is a promising avenue for future work because some recent algorithms [18] can handle certain dynamic generalization hierarchies.

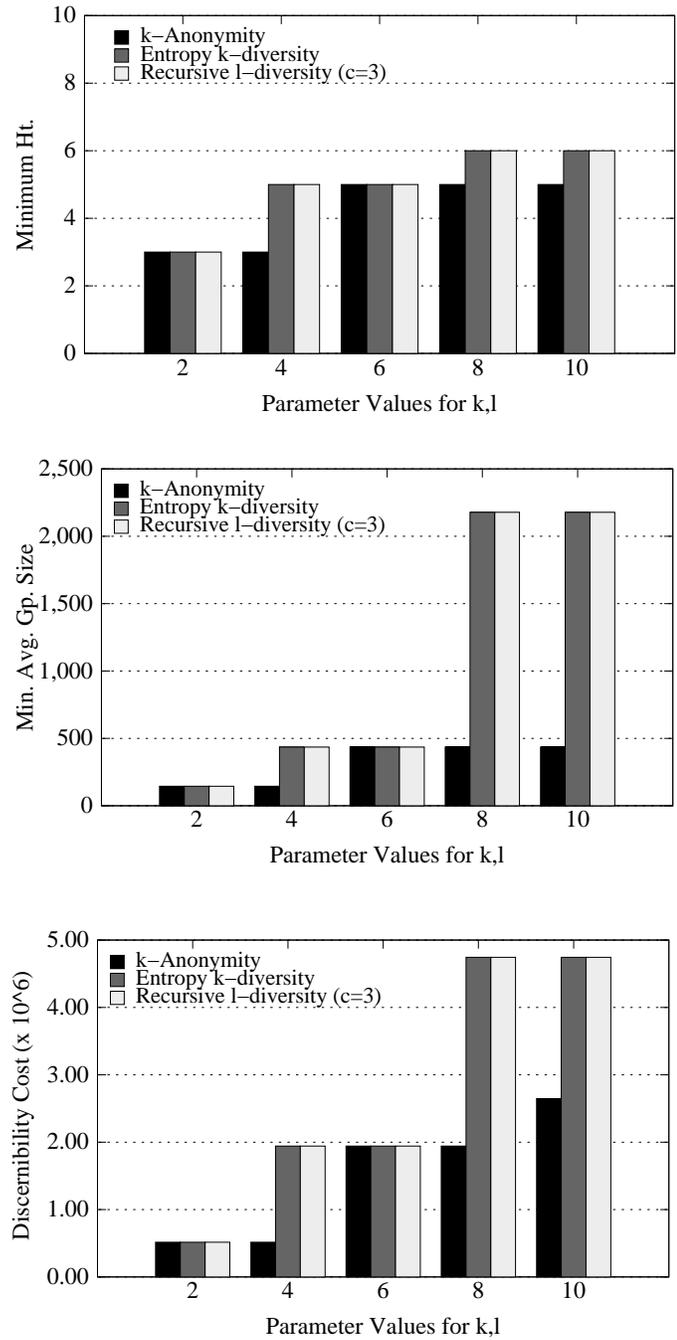


Figure 3.3: Comparing generalization height, minimum average group size, and discernibility of k -anonymous and l -diverse versions of a sample of the Adult database. $Q = \{\text{age, gender, race}\}$.

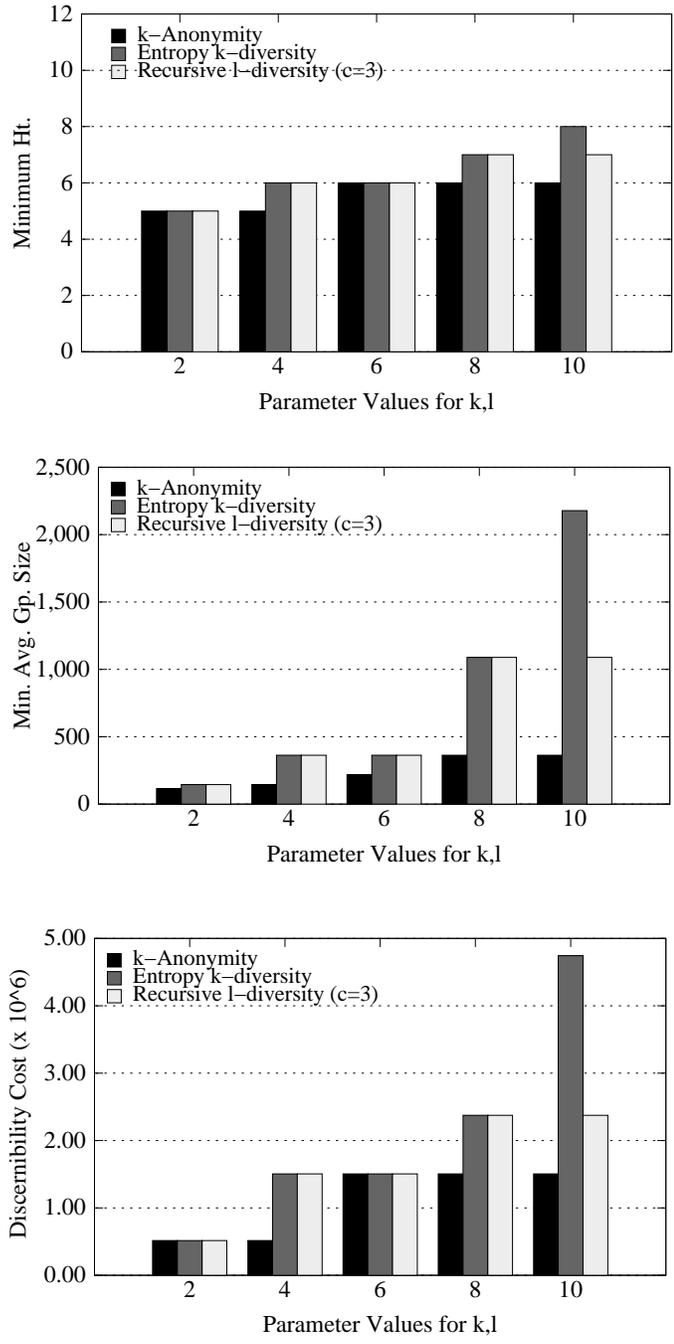


Figure 3.4: Comparing generalization height, minimum average group size, and discernibility of k -anonymous and ℓ -diverse versions of a sample of the Adult database. $Q = \{\text{age, gender, race, marital_status}\}$.

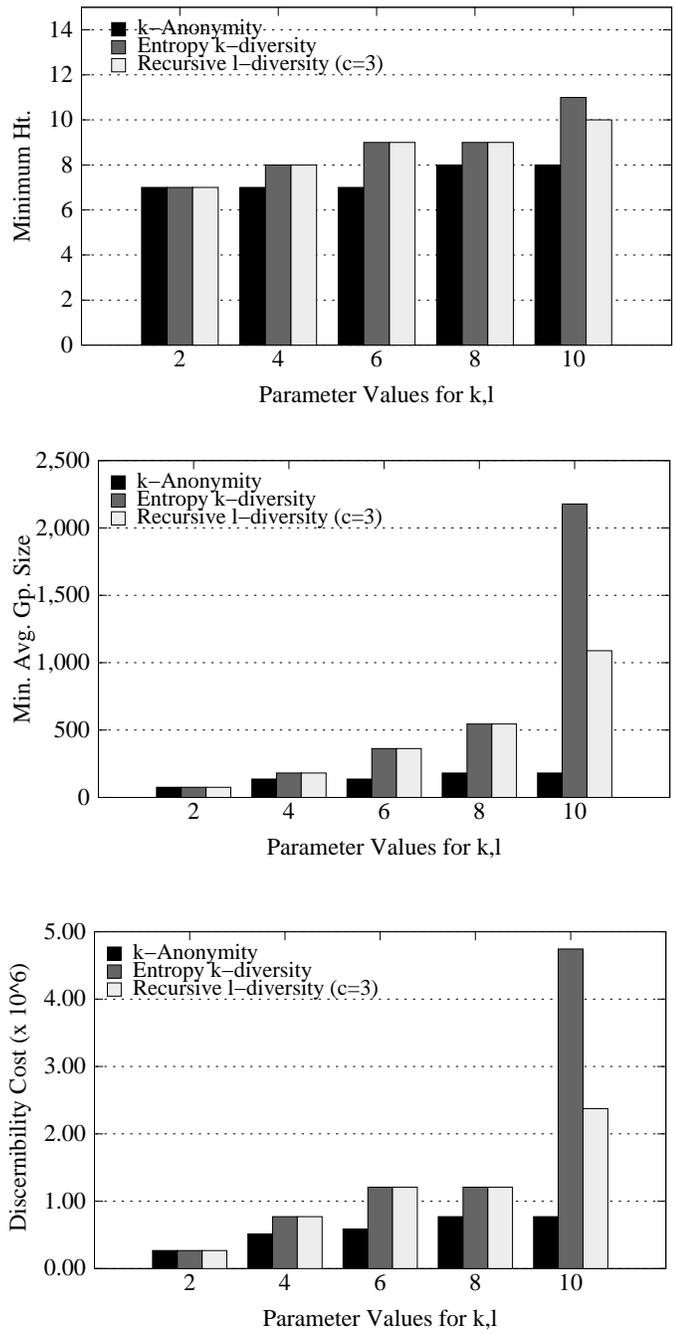


Figure 3.5: Comparing generalization height, minimum average group size, and discernibility of k -anonymous and l -diverse versions of a sample of the Adult database. $Q = \{\text{age, gender, race, marital_status, education}\}$.

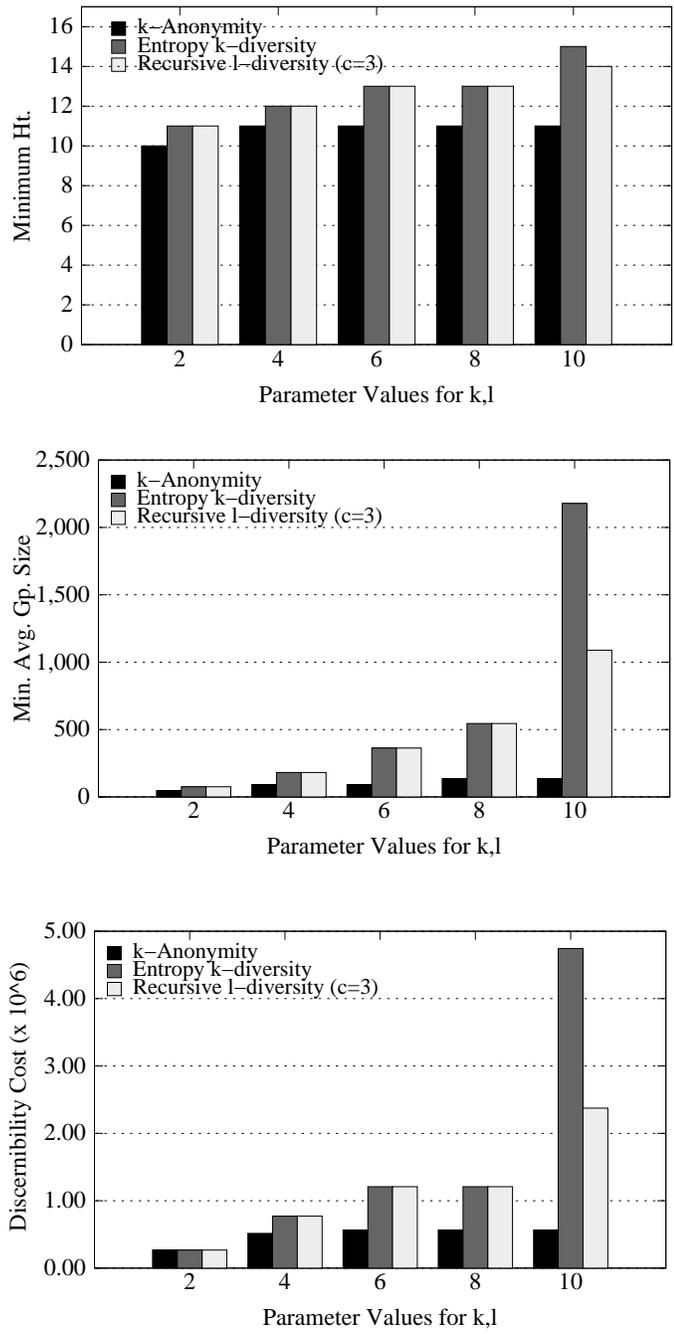


Figure 3.6: Comparing generalization height, minimum average group size, and discernibility of k -anonymous and l -diverse versions of a sample of the Adult database. $Q = \{\text{age, gender, race, marital_status, education, work_class, native_country}\}$.

in the Age attribute. It turned out that a 5% Bernoulli subsample of the Adult Database suited our requirements – most of the Age values appeared in around 20 tuples each, while only a few values appeared in less than 10 tuples each. The second and third graphs in each of Figures 3.3, 3.4, 3.5, and 3.6 show the minimum average group size and the discernibility metric cost, respectively, of k -anonymous and ℓ -diverse tables for $k, \ell = 2, 4, 6, 8, 10$. Smaller values for utility metrics represent higher utility. We found that the best t -anonymous and t -diverse tables often (but not always) had comparable utility. It is interesting to note that recursive $(3, \ell)$ -diversity permits tables which have better utility than entropy ℓ -diversity. Recursive (c, ℓ) -diversity is generally less restrictive than entropy ℓ -diversity, because the extra parameter, c , allows us to control how much skew is acceptable in a q^* -block. Since there is still some residual skew even in our 5% subsample, the entropy definition performs worse than the recursive definition.

In Figures 3.7 and 3.8 we compare k -anonymous and ℓ -diverse tables using the KL-divergence utility metric. Figure 3.7 shows our results for a 5% subsample of the table and Figure 3.8 shows our results on the whole Adult Database. In each of the graphs, we wish to publish a table from which the joint distribution $Q \times S$ can be estimated. In all the cases $S = Occupation$. Q is the multi-dimensional attribute $\{Age, Gender, Race\}$, $\{Age, Gender, Marital_Status, Race\}$ and $\{Age, Gender, Marital_Status, Education, Race\}$, respectively.

Each of the graphs shows a base-line (the bar named “Base”) that corresponds to the KL-divergence for the table where all the attributes in Q were completely suppressed (thus the resulting table had only one attribute – the

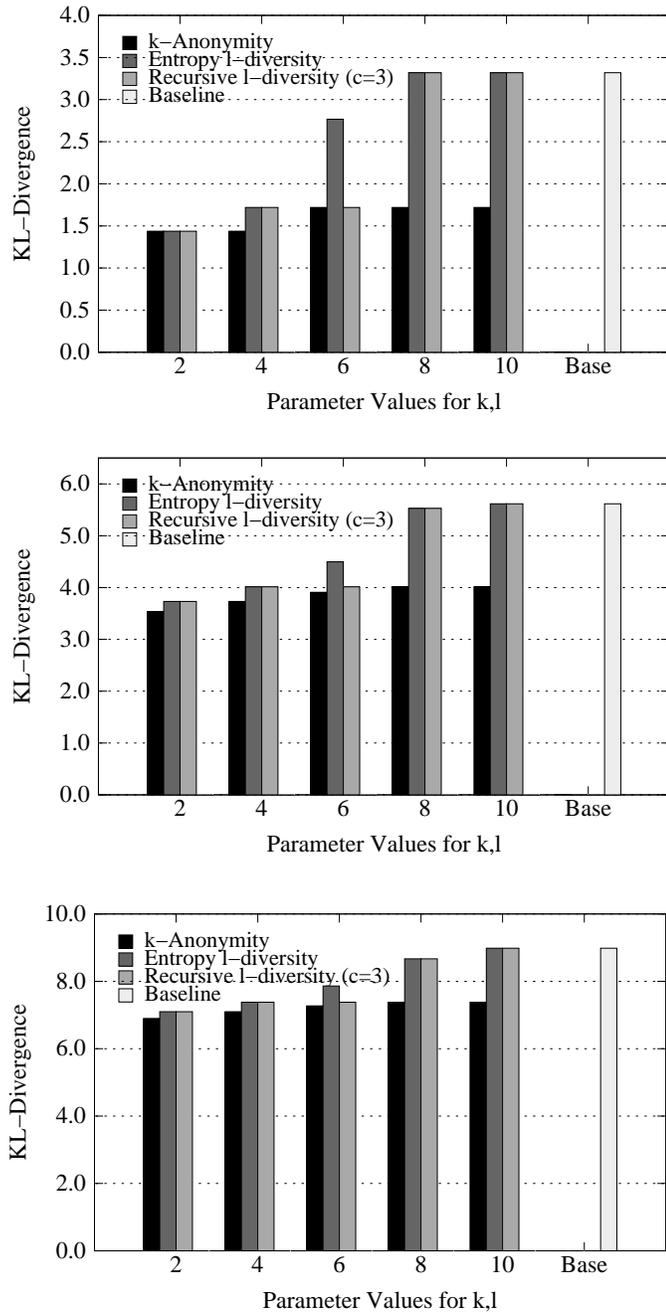


Figure 3.7: Comparing KL-Divergence to k -anonymous and ℓ -diverse versions of a sample of the Adult database. From top to bottom, $Q = \{Age, Gender, Race\}$, $\{Age, Gender, Marital Status, Race\}$ and $\{Age, Education, Gender, Marital Status, Race\}$ respectively.

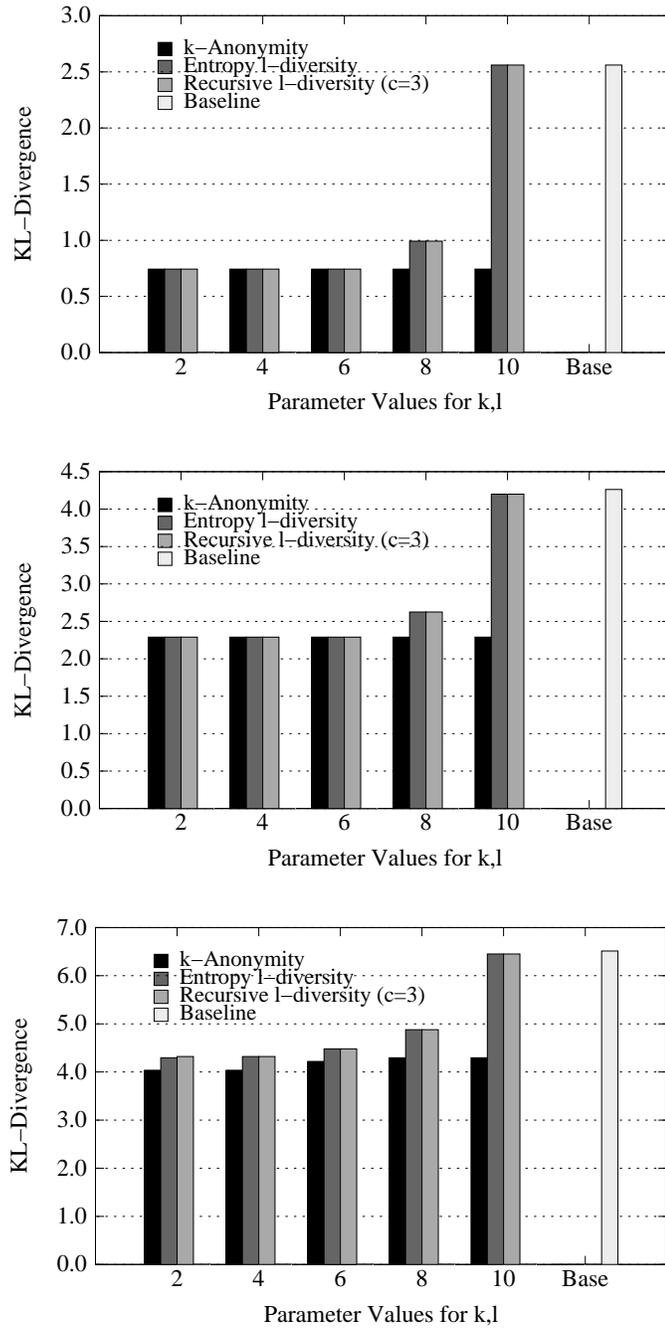


Figure 3.8: Comparing KL-Divergence to k -anonymous and ℓ -diverse versions of the Adult database. From top to bottom, $Q = \{Age, Gender, Race\}$, $\{Age, Gender, Marital Status, Race\}$ and $\{Age, Education, Gender, Marital Status, Race\}$ respectively.

sensitive attribute). This table represents the least useful anonymized table that can be published. The rest of the bars correspond to the KL-divergence to the best k -anonymous, entropy ℓ -diverse, and recursive $(3, \ell)$ -diverse tables, respectively for $k = \ell = 2, 4, 6, 8, 10$.

In the experiments run on the full Adult Dataset, we see that the KL-divergence to the best ℓ -diverse table (entropy or recursive) is very close to the KL-divergence to the best k -anonymous table, for $k = \ell = 2, 4, 6$. As expected, for larger values of ℓ , the utility of ℓ -diverse tables is lower. The best tables for the entropy and recursive variants of the definition often have similar utility. When a sample of Adult Database table was used, some of the sensitive values with small counts were eliminated. Hence, for $\ell = 8, 10$, the best tables were very close to the baseline. For $\ell = 6$, the recursive definition performs better than the entropy definition since recursive $(3, \ell)$ -diversity allows for more skew in the sensitive attribute.

3.7 A Formal Framework for Defining Privacy

In the last two chapters, we have described two seemingly distinct techniques for ensuring privacy in databases, namely, access control and anonymous data publishing. These techniques are based on two very different notions of privacy, namely, perfect privacy and ℓ -diversity, respectively. In recent years, many other privacy metrics have been proposed in the literature for a variety of data privacy settings including anonymous data publishing – k -anonymity [122], ℓ -diversity, perfect privacy, (c, k) -safety [92], privacy skyline [32], γ -amplification [62], differential privacy [58], etc. Unfortunately, the current state of the art poses two

problems – (i) there is no mandate on how to define privacy for new applications, and (ii) there is no means by which the plethora of privacy metrics in literature can be compared.

In this section, we propose a solution to the above problem by classifying privacy definitions as *syntactic* and *semantic*. A *syntactic privacy definition* is an algorithmic condition that can be efficiently enforced. Most of the privacy metrics prevalent in literature, e.g., k -anonymity, recursive (c, ℓ) -diversity, and critical tuple perfect privacy (from Theorem 2.5) are syntactic privacy definitions. However, one cannot easily infer the semantics of the privacy guaranteed by a syntactic definition. For instance, we showed previously in this chapter that recursive (c, ℓ) -diversity guarantees privacy against adversaries with at most $\ell - 2$ negation statements. However, this can not be immediately deduced from the syntactic definition. A *semantic privacy definition*, on the other hand, defines privacy based on a model of disclosure, and hence, precisely states the assumptions under which privacy is guaranteed. Bayes-optimal privacy (Section 3.3.2), and perfect privacy (Equation 2.1) are examples of a semantic privacy definitions. As we will describe later in this section, unlike syntactic privacy definitions, the privacy guaranteed by different semantic privacy definitions can be readily compared, and these definitions can be easily extended to suit the requirements of new applications. We will also see that, every syntactic privacy definition should ideally be associated with an equivalent semantic privacy definition.

We now propose a framework for semantic privacy definitions, thus laying a common ground across privacy definitions that aids the design of new privacy definitions and helps compare different privacy metrics. We next answer a progression of questions and in the process describe our framework for formu-

lating semantic privacy definitions. Simultaneously, we formulate a semantic privacy definition equivalent to recursive (c, ℓ) -diversity.

Q 1: *Is there a single “good” definition of privacy?*

Though it would be ideal to have a single definition of privacy, clearly as illustrated in this dissertation, one privacy definition does not meet the needs of all kinds of applications. For instance, in hospital or military applications privacy means “do not disclose any sensitive information”; if a nurse is not authorized to view Bruce’s medical records she should not learn any information about Bruce’s disease. Since k -anonymity and ℓ -diversity allow partial disclosure, they are not applicable to this scenario. On the other hand, in Census applications privacy means *bounded partial disclosure* or “disclose very little information”. Perfect privacy is not applicable here as a researcher can not learn the correlation between age, race and diseases if she is not allowed to learn any information about any individual’s disease. So this leads us to the next question,

Q 2: *What are the properties of a “good” privacy definition?*

We argued in the beginning of Chapter 3 that k -anonymity does not suffice for ensuring privacy of anonymous data publishing. While k -anonymity was attempting to protect sensitive information about individuals, we showed that it does not protect against simple adversarial attacks which are reasonable for the data publishing application. However, perfect privacy and ℓ -diversity also do not guard against every possible adversary. In fact, there are applications where the assumptions made by both perfect privacy and ℓ -diversity do not hold. For

instance, ℓ -diversity assumes that all attributes are categorical; clearly, this assumption does not hold if individuals could have a set of diseases instead of a single disease, or if the sensitive attribute is numeric. Perfect privacy assumes that every tuple in the relation is independent of the rest of the tuples; the same is not true when the data is inherently linked, like in the case of a social network.

Despite their shortcomings, we claim that perfect privacy and ℓ -diversity are reasonable definitions of privacy due to the following reasons. k -anonymity and earlier syntactic privacy metrics [128] defined privacy based on known adversarial attacks such as privacy breaches due to re-identification of individuals in the data. Such a methodology for defining privacy is doomed to fail because a data publisher can not enumerate every possible scenario in which an adversary can breach private information. This has also been noted in the field of cryptography; early ciphers were broken since they satisfied only ad-hoc definitions of security.

In contrast to k -anonymity, ℓ -diversity and perfect privacy are based on a model for disclosure; they explicit state their assumptions about the data and the sensitive information, the adversary's background knowledge and how they measure disclosure. The following excerpt from a seminal paper by Lambert [79] highlights the importance of models of disclosure,

One could argue that models of disclosure are hopeless because the issues are too complex and the intruder too mysterious. Instead, this paper argues that models of disclosure are indispensable. At the least, they force definitions and assumptions to be stated explicitly. And, when the assumptions are realistic, models of disclosure can lead to practical measures of disclosure risk and harm.

Since ℓ -diversity and perfect privacy explicitly state their assumptions, given an application, one can gauge whether or not these privacy metrics will guard against all the attacks that are typical for that application. This leads us to the next question,

Q 3: *How are the semantics of a privacy definition formulated? That is, what assumptions made by the privacy metric should be precisely defined?*

We answer this question by presenting our framework for defining semantic privacy definitions. The framework is inspired by the notion of semantic security for public key encryption [68], and generalizes the formal reasoning in ℓ -diversity. As we describe the framework, we simultaneously instantiate the semantic privacy definition for ℓ -diversity.

Let us first recall some basics. There is a population of individuals Ω , and the information about a subset of these individuals (Ω_D) appears in a secret database D . The secret database D (and the subset of individuals Ω_D) is known to the data publisher.

In our framework, a semantic privacy definition should necessarily define three important concepts – *sensitive information*, *adversarial background knowledge*, and *measure of disclosure*.

Sensitive information. The first step towards understanding the semantics of privacy is a precise definition of the information being protected, i.e., the sensitive information. Defining the sensitive information, though seemingly obvious, can be quite tricky. Informal specifications can be misleading; for instance, just stating that medical condition is the sensitive attribute is insufficient. It could mean, like in perfect privacy, that all properties of the sensitive attribute

including “total number of diseases appearing in the table” should be protected. Or, like in ℓ -diversity, it could mean that only the link between an individual and her disease should be protected. Such ambiguity should be avoided since a privacy definition only protects the information that is defined as sensitive.

We recommend that the sensitive information be specified as a set of boolean predicates (Φ). Our semantic definition defines disclosure based on these predicates. For instance, recursive (c, ℓ) -diversity aims to protect the following set of sensitive predicates,

$$\Phi_{\ell\text{-div}} = \{eq(X[S], s) \mid \forall X \in \Omega_D \wedge \forall s \in S\} \quad (3.9)$$

where S is the domain of the sensitive attribute, Ω_D is the set of individuals who appear in the secret database D , and $eq(X[S], s)$ is the predicate that is true when $X[S] = s$, and false otherwise.

Adversarial Background Knowledge. As previously described in this chapter, adversaries may know public information about individuals in the data from other external sources. We model such information using a probability distribution over the set of all possible databases. That is, if \mathcal{D} is the set of all possible databases, an adversary’s background knowledge is the probability function $w : \mathcal{D} \rightarrow [0, 1]$, $\sum_{D' \in \mathcal{D}} w(D') = 1$.

However, we must reason about privacy from a data publisher’s standpoint, and as we described in the case of ℓ -diversity, the data publisher may not know all the information the adversary knows. We model this scenario using a set of adversaries, $\{\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_a\}$, one for each distinct possible background knowledge the data publisher may think the adversary possesses. Each adversary \mathcal{A}_i in this set is modeled using a probability distribution w_i .

For instance, in ℓ -diversity, the set of adversarial distributions can be described as follows. Let \mathcal{D} denote the set of tables constructed by first choosing a set of n individuals from the population, and then assigning to each of these chosen individuals a value from the domain of the sensitive attribute. Each adversary \mathcal{A}_i is characterized by an individual X and a set of $\ell - 2$ negation statements $\{X[S] \neq s_1, X[S] \neq s_2, \dots, X[S] \neq s_{\ell-2}\}$ (as described in Equation 3.10).

$$\{\mathcal{A}_i = (X, s_1, \dots, s_{\ell-2}) \mid \forall X \in \Omega, \forall \{s_1, \dots, s_{\ell-2}\} \subset S\} \quad (3.10)$$

For a specific adversary \mathcal{A}_i , let $\mathcal{D}_i^{bad} \subseteq \mathcal{D}$ be the set of databases such that either X does not appear in the database or $X[S] = s_i, i \in [1, \ell - 2]$. These databases are not possible under \mathcal{A}_i 's knowledge. w_i assigns equal probability to the rest of the possible databases in $\mathcal{D} - \mathcal{D}_i^{bad}$ as,

$$w_i[D'] = \begin{cases} 0, & D' \in \mathcal{D}_i^{bad}; \\ 1/(|\mathcal{D} - \mathcal{D}_i^{bad}|), & D' \notin \mathcal{D}_i^{bad}. \end{cases} \quad (3.11)$$

Measure for Disclosure. We say that privacy is breached if the value of at least one of the sensitive predicates is disclosed under one of the adversarial settings after releasing a view of the database, say V . In the case of anonymous data publishing, V is the anonymized table T^* . As we saw in Section 3.3.2 there are many ways to model disclosure. A c -positive disclosure occurs if

$$\exists w_i \in \{w_1, \dots, w_a\}, \exists \phi \in \Phi \quad \text{s.t.} \quad P_{w_i}(\phi \mid V) > c$$

Recursive (c, ℓ) -diversity considers c -positive disclosures as privacy breaches. A c -negative disclosure occurs if

$$\exists w_i \in \{w_1, \dots, w_a\}, \exists \phi \in \Phi \quad \text{s.t.} \quad P_{w_i}(\phi \mid V) < c$$

Recall that disclosure can also be modeled using the uninformative principle.

$$\exists w_i \in \{w_1, \dots, w_a\}, \exists \phi \in \Phi \quad \text{s.t.} \quad \text{dist}(P_{w_i}(\phi \mid V), P_{w_i}(\phi)) > \epsilon$$

where, $dist(\cdot, \cdot)$ is a distance function, e.g., difference or ratio.

Putting all of our above discussion together, we have developed a framework for defining semantic privacy definitions that (i) defines the sensitive information as a set of boolean predicates, (ii) models the adversaries as a set of probability distributions over the set of all possible databases \mathcal{D} , and (iii) defines that privacy is breached if under at least one adversarial distribution, the truth value of some sensitive predicate is disclosed by a positive disclosure, negative disclosure and/or by the uninformative principle. Along the way, we have instantiated one such definition which we will call *semantic (c, ℓ) -diversity*.

Definition 3.12 (Semantic (c, ℓ) -Diversity) *An anonymized table T^* satisfies semantic ℓ -diversity if*

$$\forall w_i \in \{w_1, \dots, w_a\}, \forall \phi \in \Phi_{\ell\text{-div}}, P_{w_i}(\phi | T^*) < c$$

where, $\Phi_{\ell\text{-div}}$ is as defined in Equation 3.9, and the set of adversarial distributions is as defined in Equation 3.10 and 3.11.

It has been shown that Definition 3.12 is equivalent to recursive (c, ℓ) -diversity (Definition 3.8) [92]; i.e., a generalized table T^* satisfies semantic (c, ℓ) -diversity if and only if it satisfies recursive (c, ℓ) -diversity.

We can analogously formulate semantic privacy definitions equivalent to pd-recursive (c, ℓ) -diversity (Definition 3.9) and npd-recursive (c_1, c_2, ℓ) -diversity (Definition 3.10). A semantic definition for pd-recursive (c, ℓ) -diversity would differ from semantic (c, ℓ) -diversity only in the specification of sensitive information; the predicates $eq(X[S], y)$, where $y \in Y$ is a don't care sensitive value, are not considered sensitive.

$$\Phi_{(\ell\text{-div}, Y)} = \{eq(X[S], s) \mid \forall X \in \Omega_D \wedge \forall s \in S - Y\}$$

A semantic definition for npd-recursive (c_1, c_2, ℓ) -diversity would differ from semantic (c, ℓ) -diversity in the measure for disclosure – both c_1 -positive and c_2 -negative disclosures are considered breaches of privacy.

Q 4: *Are there other examples of semantic privacy definitions?*

ℓ -Diversity is not the first or the only privacy definition with an equivalent semantic privacy definition; perfect privacy [96], γ -amplification [62], and differential privacy [58] have equivalent semantic definitions that describe the sensitive information, the adversarial knowledge and the measure of disclosure. However, we would like to point out that this is the first attempt toward a general framework for formulating semantic privacy definitions. To illustrate its generality, we cast the semantic privacy definition for perfect privacy (Equation 2.1) in terms of our framework.

Theorem 2.1 [96] shows that the perfect privacy definition based on critical tuples (Chapter 2) is equivalent to an information theoretic semantic privacy definition (Equation 2.1), which we will call *semantic perfect privacy*. The sensitive information is modeled as the following set of sensitive predicates,

$$\Phi_{perfect}^{Q_S} = \{answer(Q_S, S) | \forall S \subseteq Tup(Q_S)\}$$

where $Tup(Q_S)$ is the set of all tuples that could appear in the answer to secret query Q_S , and $answer(Q_S, S)$ is true if and only if S is the answer to query Q_S on the secret database D .

The adversarial knowledge is modeled thus. Let R be a publicly known relational schema. Let $Tup(R)$ denote the set of all tuples satisfying R . Let \mathcal{D} denote the set of all possible distinct relational instances using tuples in $Tup(R)$. Let

p be a function mapping each tuple $t \in Tup(R)$ to a probability in $[0, 1]$. Then, the set of adversaries is characterized by the set of distinct tuple probability functions p ; adversary \mathcal{A}_p knows that the tuple probabilities are given by the function p . The adversarial distribution w_p assigns each database D' the probability,

$$w_p(D') = \prod_{t \in D'} p(t) \prod_{t \notin D'} (1 - p(t))$$

Answering a query Q_V satisfies semantic perfect privacy with respect to secret query Q_S , if for all answers V to the query Q_V ,

$$\forall w_p \forall \phi \in \Phi_{perfect}^{Q_S}, P_{w_p}(\phi) = P_{w_p}(\phi \mid Q_V(D) = V)$$

γ -amplification, a privacy metric for independent tuple randomization (see Section 5.2), was shown to be equivalent to ρ_1, ρ_2 -privacy [62]. Differential privacy, a privacy metric we will use in Chapter 4, was shown to be equivalent to ϵ -semantic security [59]; both ρ_1, ρ_2 -privacy and ϵ -semantic security can be formulated in our framework.

While our framework is capable of instantiating a very large number of semantic privacy definitions, not all definitions are useful unless they are proven to be equivalent to an efficiently enforceable syntactic privacy metric.

Q 5: *Why do we need a framework for semantic privacy definitions?*

The general theory of semantic privacy definitions has two main advantages. First, since it precisely states the assumptions under which privacy is guaranteed, a semantic privacy definition can be used to identify limitations of an existing privacy definition, and to extend existing definitions to new settings. For

instance, ℓ -diversity does not guarantee privacy of numeric sensitive attributes. Consider a 3-diverse table where salary is the sensitive attribute, and every group has at least 3 distinct values for salary. While a group that has salaries \$100,000, \$100,001, and \$99,999 in roughly equal proportions is 3-diverse, it constitutes a privacy breach as an adversary has a very good estimate of an individual's salary in that group. The privacy breach occurs because $\Phi_{\ell\text{-div}}$, the set of sensitive predicates for ℓ -diversity, does not capture the fact that inferring an individual's salary is within a small range is also considered a privacy breach. We can remedy this situation by adding the following predicates to $\Phi_{\ell\text{-div}}$.

$$\{in(X[S], x - lb, x + ub), \forall x \in S\}$$

where $in(X[S], x - lb, x + ub)$ is true iff $(X[S] \in [x - lb, x + ub])$ for parameters lb and ub . (c, k) -Safety [92] and privacy skyline [31] are semantic privacy definitions that extend ℓ -diversity by considering adversaries with more powerful kinds of background knowledge.

A more crucial advantage of a general theory of semantic privacy definitions is that it lays down a common framework for various privacy metrics. Traditionally, privacy has been defined in terms of a specific application or a specific privacy preserving technique. However, note that the semantic privacy definitions are not tied to any application or privacy preserving algorithm. Hence, if the assumptions stated by a semantic privacy definition are reasonable for two applications, then the equivalent syntactic privacy metric that is defined in the context of one application can be used in the other application as well. For instance, differential privacy [58] was traditionally proposed in the context of *output perturbation techniques*, where a statistical database answers aggregate queries by adding random noise. In fact, in the next chapter, we will use differential privacy as the privacy definition for anonymous data publishing.

The common framework can also be used to compare different privacy metrics; this will be especially useful for analyzing the privacy of a new application. In the next chapter, we will present a case study of anonymizing data from real-world mapping application. We will use semantic privacy definitions to guide our choice of a syntactic privacy definition for that application.

In summary, we have proposed a framework for semantic privacy definitions that will aid understanding existing privacy metrics, extend them to new applications, and help data publishers to choose the right privacy definition.

3.8 Summary

In this chapter we have shown theoretically and experimentally that a k -anonymized dataset permits strong attacks due to lack of diversity in the sensitive attributes. We have introduced ℓ -diversity, a privacy definition that gives stronger privacy guarantees. We have also demonstrated that ℓ -diversity and k -anonymity have enough similarity in their structure that k -anonymity algorithms can be modified to work with ℓ -diversity. Finally, we presented a general theory of the semantics of privacy definitions that will help us build upon existing formal privacy definitions for new applications.

CHAPTER 4

FROM THEORY TO PRACTICE ON THE MAP

4.1 Introduction

In this chapter, we consider the problem of applying formal privacy definitions to a Census Bureau application, called OnTheMap¹, that plots commuting patterns of workers in the U.S. This application uses a novel privacy preserving technology known as *synthetic data generation*. We present the first formal privacy analysis (to the best of our knowledge) for this technology. This chapter chronicles the challenges we faced in this endeavour.

The target application, OnTheMap, is based on data developed by the U.S. Census Bureau’s Longitudinal Employer-Household Dynamics Program (LEHD). By combining various Census datasets it is possible to construct a table *Commute_Patterns* with schema $(id, origin_block, destination_block)$ where each row represents a worker. The attribute *id* is a random number serving as a key for the table, *origin_block* is the census block in which the worker lives, and *destination_block* is where the worker works. An origin block *o* corresponds to a destination block *d* if there is a tuple with *origin_block* *o* and *destination_block* *d*. The goal is to plot points on a map that represent commuting patterns for the U.S. population. For each destination block, we plot points on the map representing the corresponding origin blocks. There are roughly 8 million Census blocks so that the domain is very large and the data is very sparse.

Information about destination blocks has already been publicly released,

¹<http://lehdmap2.dsd.census.gov/>

and is available to any adversary; thus *origin_block* is treated as the sensitive attribute. Due to privacy constraints and legal issues, unanonymized origin block data cannot be used as an input to such a mapping application. An anonymized version must be used instead.

The algorithm used to anonymize the data for the above mapping application is known as synthetic data generation [111], which is becoming popular in the statistical disclosure limitation community. The main idea behind synthetic data generation is to build a statistical model from the data and then to sample points from the model. These sampled points form the synthetic data, which is then released instead of the original data. While much research has focused on the utility of these synthetic datasets (deriving the variance and confidence intervals for various estimators from synthetic data) [102, 106], there has been little research on deriving formal guarantees of privacy for such an approach (an exception is [107]).

4.1.1 Contributions and Chapter Outline

We present the derivation of our techniques as a case study of anonymizing data for the novel mapping application. In this spirit, we discuss the initial selection of an off-the-shelf privacy definition in Section 4.2 and an off-the-shelf anonymization algorithm in Section 4.3 based on the requirements of the mapping application.

As outlined in Section 3.7, we first describe the privacy requirements of the mapping application in terms of the sensitive information that needs to be protected, the background knowledge an adversary may possess, and the measure

of disclosure. We found that apart from the differential privacy criterion [58], none of the other existing privacy conditions applied to our scenario.

We found that picking an off-the-shelf synthetic data generation algorithm and tuning it to satisfy the differential privacy criterion was unsatisfactory for the following reasons. First, in order to satisfy the differential privacy criterion, the generated synthetic data contained little or no information about the original data. We show that this is because differential privacy guards against breaches of privacy that are very unlikely.

Next, no deterministic algorithm can satisfy differential privacy. Randomized algorithms can (albeit with a very small probability) return anonymized datasets that are totally unrepresentative of the input. This is a problem, especially, when we want to publish a single, or only a few, anonymous versions of the whole data. We remedy these two issues by showing that a revised probabilistic version of differential privacy yields a practical privacy guarantee for synthetic data (Section 4.4).

Finally, the data in our application is very sparse – there are roughly 8 million blocks on the U.S. map, and only about a few tens or hundreds of workers commuting to each destination. Most previous work deals with data where the number of individuals is typically larger than the size of the sensitive attribute domain. We identify this important open research problem and propose our first solutions for solving the sparsity issue by modifying the synthetic data generation algorithm (Section 4.5). In Section 4.6 we present experiments and discuss the utility of the resulting synthetic data.

To summarize, this chapter’s contributions are as follows: we provide the

first formal privacy analysis for a synthetic data generation method (to the best of our knowledge); we present a case-study of applying state-of-the-art research in privacy to real applications; we identify new challenges for the privacy research community (such as handling large domains), and we propose initial solutions for these challenges.

4.2 Starting Point: Privacy Definition

To help select a privacy definition, we use the framework from Section 3.7 to describe the requirements of this application.

Sensitive Information. The original *Commute_Pattern* table contains the origin block of every worker. Origin blocks are represented as *census blocks*; census blocks could be as small as a few streets or a few houses. Hence, an adversary who has access to the application should not be able to learn either the exact or approximate origin block of any worker. Let B denotes the set of all blocks on the map of the United States, and Ω the set of all workers. To disallow the disclosure of exact origin block information, it is sufficient to disallow the disclosure of the following predicates.

$$\Phi_{exact} = \{X[origin_block] = b \mid \forall X \in \Omega \wedge \forall b \in B\}$$

To encode approximate origin information, we consider a neighbourhood function N that maps each block b to a region $N(b) \subseteq B$, where $N(b)$ is the largest region around b such that knowing that a worker comes from the region $N(b)$ is a breach of privacy. For instance, for a worker living in $b_1 = \text{Devon Street, Chicago}$, $N(b_1)$ could be the city of Chicago, whereas for a worker living in $b_2 = \text{Stewart Ave, Ithaca}$, $N(b_2)$ could be the union of all blocks in Ithaca, and

the neighbouring town of Dryden. To disallow disclosure of approximate origin block information, it is sufficient to disallow the disclosure of predicates in Φ_{approx} .

$$\Phi_{approx} = \{X[origin_block] \in N(b) \mid \forall X \in \Omega \wedge \forall b \in B\}$$

Adversary Knowledge. We assume that an adversary knows every worker’s destination block information. As for the origin block, consider the following scenario. Alice and Bruce work for the same company, say in Syracuse, NY. Alice knows that Bruce takes the longest time to commute to work (about one hour) amongst all the other employees in the company. Alice may also know that Bruce drives south to go home. Given all this information, Alice can deduce that Bruce comes from one of the towns in {Ithaca, Freeville, Dryden, Groton, Trumansburg}.

In the worst case, an adversary may know that a worker commutes from one of two origin blocks on the map, and has to deduce which of these block the worker actually comes from looking at the published data.

Measure of Disclosure. Mapping applications come with a lot of background knowledge; workers usually commute to cities, and most origins are close to destinations. Hence, it would not be advisable to consider only positive or negative disclosures. Rather one should use the uninformative principle (Section 3.3.2), where privacy is breached if the adversary’s prior belief in an individual’s sensitive predicate is very different from the adversary’s posterior belief about that predicate.

Rather than developing a new privacy criterion that precisely matches the above privacy requirements, we decided to shop for a privacy definition that

captures requirements of our application. We next describe three potential candidates ℓ -diversity (Chapter 3), (d, γ) -privacy [105] and differential privacy [58], and discard two out of the three.

In its most basic form, ℓ -diversity requires that for each destination block, the ℓ origin blocks with the most number of workers with jobs in this destination block, have roughly equal number of workers residing in them. Although ℓ -diversity can protect against adversaries with background knowledge, it does not always guarantee privacy when there is a semantic relationship between the sensitive values. Here the semantic relationship is physical proximity which is encoded by our requirement of protecting (neighbourhood) regions of blocks. That is, ℓ -diversity does not offer theoretical guarantees against an adversary who has information such as “Bruce probably lives near Newark and works near New York City.”

(d, γ) -Privacy is a semantic privacy definition with the following properties. The sensitive information is whether or not a tuple t appears in the database. That is, in the context of our mapping application, predicates of the form “*Does Bruce travel from Syracuse to Ithaca*” are considered sensitive. The adversary believes in some prior probability $P(t)$ of a tuple t appearing in the database. After seeing the anonymized data D , the adversary forms a posterior belief $P(t|D)$. (d, γ) -Privacy is only designed to protect against adversaries that are *d-independent*: an adversary is *d-independent* if for all tuples t are considered a priori independent and, the prior belief $P(t)$ satisfies the conditions $P(t) = 1$ or $P(t) \leq d$. For all such adversaries, the privacy definition requires that $P(t|D) \leq \gamma$ and $P(t|D)/P(t) \geq d/\gamma$. This privacy definition does not apply to our scenario for a couple of reasons. First, this privacy definition does not apply

for adversaries that believe that $P(t) = d + \epsilon$ (no matter how small $\epsilon > 0$ is) for some t even though in those cases we would also like to have some guarantee about privacy. For instance, an adversary may know apriori that Bruce comes from either Ithaca with probability 0.2 or from Syracuse with probability 0.8; for $d < 0.8$, this privacy definition cannot be used. Second, tuple-independence is a very strong assumption that is not compatible with our application. In order to be d -independent, an adversary has to consider the facts “Worker #1234 commutes to Block 12 from Block 34” and “Worker #1234 commutes to Block 12 from Block 35” independent. This is not true in our application since, the two events described above are mutually exclusive.

Differential privacy is privacy definition with the following intuition. If one individual is considering lying about her data to a data collector (such as the U.S. Census Bureau), the result of the anonymization algorithm will not be very different whether or not the individual lies. The differential privacy criterion can be defined as follows.

Definition 4.1 (ϵ -Differential Privacy) *Let \mathcal{A} be a randomized algorithm, \mathcal{S} be the set of possible outputs of the algorithm, and $\epsilon > 1$. The algorithm \mathcal{A} satisfies ϵ -differential privacy if for all pairs of data sets (D_1, D_2) that differ in exactly one row,*

$$\forall S \in \mathcal{S}, \quad \left| \ln \frac{P(\mathcal{A}(D_1)=S)}{P(\mathcal{A}(D_2)=S)} \right| \leq \epsilon \quad (4.1)$$

Differential privacy fits our application since the semantics of its privacy guarantee match the privacy requirements of our application. Differential privacy is equivalent to a semantic privacy definition with the following properties. Each tuple in the original table describes the information about a unique individual. Every property of this tuple is considered sensitive. In the context of our appli-

cation, every predicate “ Does Bruce commute to one of the blocks in B' ”, where B' is an arbitrary subset of B is considered sensitive. The adversary is assumed to have complete information about all individuals but one (say X) and believes in an arbitrary prior probability P for the tuple corresponding to X ; in the context of our mapping application, the adversary may know that the X commutes to exactly one of two origin blocks on the map². Semantically, differential privacy ensures that [59] after seeing the anonymized data D ,

$$\forall X \in \Omega, \forall B' \subseteq B, \left| \ln \frac{P(X[\text{origin_block}] \in B' | D)}{P(X[\text{origin_block}] \in B')} \right| < \epsilon$$

4.3 Starting Point: Anonymization Algorithm

The original data can be viewed as a histogram where each combination of *origin_block* and *destination_block* is a histogram bucket. Histograms can be anonymized by modifying the counts in each bucket (for example, by adding random noise). Both [59] and [105] provide randomized anonymization algorithms for histograms that satisfy ϵ -differential privacy. One method is to add an independent Laplace random variable to each bucket of the histogram [59]. Another is to extract a Bernoulli subsample from the data and then to add independent Binomial random variables to each histogram bucket [105]. Intuitively, both methods mix the original data with a dataset that is generated from independently and identically distributed noise.

Instead of using one of these approaches, we use synthetic data generation [111] for protecting privacy. Prior to this work, synthetic data methods did not have formal privacy guarantees, despite there being significant work on

²This is the reason differential privacy compares two tables that differ in exactly one entry.

performing statistical analyses of and drawing inferences from synthetic data [102, 106] based on Rubin’s multiple imputation framework [112]. Thus, our goal is to complement the research on utility for synthetic data by providing results about its privacy.

The main idea behind synthetic data generation is to build a statistical model from the data and then to sample points from the model. These sampled points form the synthetic data, which is then released instead of the original data. The motivation behind such statistical modeling is that inferences made on the synthetic data should be similar to inferences that would have been made on the real data.

Privacy comes from the fact that noise is added to the data from two sources: the bias that comes from the creation of the model, and the noise due to random sampling from the model. Note that the process of learning in the context of a model for synthetic data differs significantly from the normal application of machine learning. In machine learning, it is imperative not to overfit the data. For synthetic data, we want to overfit as much as possible (subject to privacy constraints), so that the synthetic data contains many of the characteristics of the original data.

For this application, we will use a multinomial model with a dirichlet prior [66] as the initial mechanism for generating synthetic data. We describe the model below, starting with some necessary definitions:

Definition 4.2 (Multinomial distribution) *Let $\vec{p} = (p_1, \dots, p_k)$ be a vector of non-negative values such that $p_1 + \dots + p_k = 1$. A multinomial distribution of size m with parameters (p_1, \dots, p_k) , $\mathcal{M}(\vec{p}, m)$, is a probability distribution over k -*

dimensional vectors with non-negative integer coordinates that sum up to m , with

$$P(m_1, \dots, m_k) = \frac{m!}{\prod_{i=1}^k m_i!} \prod_{i=1}^k p_i^{m_i}$$

Definition 4.3 (Dirichlet distribution) Let $\vec{\alpha} = (\alpha_1, \dots, \alpha_k)$ be a vector of positive values and let $|\alpha| = \alpha_1 + \dots + \alpha_k$. The dirichlet distribution with parameters $(\alpha_1, \dots, \alpha_k)$, $\mathcal{D}(\vec{\alpha})$, is a probability distribution over all k -dimensional vectors with non-negative coordinates that sum up to 1, with density³

$$f(p_1, \dots, p_k | \vec{\alpha}) = \frac{\Gamma(|\alpha|)}{\prod_{i=1}^k \Gamma(\alpha_i)} \prod_{i=1}^k p_i^{\alpha_i - 1}$$

The vector $\vec{\alpha} = (\alpha_1, \dots, \alpha_k)$ is known as the prior sample, $|\alpha|$ is the prior sample size, and the vector $(\alpha_1/|\alpha|, \dots, \alpha_k/|\alpha|)$ is the shape of the prior.

Multinomial sampling with a dirichlet prior $\mathcal{D}(\vec{\alpha})$ proceeds as follows:

1. Draw a vector $\vec{p} = (p_1, \dots, p_k)$ from the $\mathcal{D}(\vec{\alpha})$ distribution.
2. Interpret \vec{p} as a vector of multinomial probabilities and draw a sample of size n from the multinomial distribution $\mathcal{M}(\vec{p}, n)$.

It is well known that if (n_1, \dots, n_k) was drawn using multinomial sampling with a dirichlet prior, then the posterior distribution $P(\vec{p} | (n_1, \dots, n_k))$ is the dirichlet distribution $\mathcal{D}((\alpha_1 + n_1, \alpha_2 + n_2, \dots, \alpha_k + n_k))$. This can be interpreted informally as first having no information, observing the sample data $(\alpha_1, \dots, \alpha_k)$ (note the α_i do not have to be integers), and updating the prior to $\mathcal{D}((\alpha_1, \dots, \alpha_k))$; then observing the new data (n_1, \dots, n_k) , and updating the prior

³where $\Gamma(t)$ is the gamma function defined as $\int_0^\infty x^{t-1} e^{-x} dx$

Algorithm 3: Synthesize

for all destination blocks d **do**

Let $\overrightarrow{n(d)}$ be the histogram of origin blocks

Choose prior sample $\overrightarrow{\alpha(d)}$ with $|\alpha(d)| = O(n(d))$

Choose output sample size $m = O(n(d))$

Sample $m(d)$ points from a multinomial distribution with prior $\mathcal{D}((n(d)_1 + \alpha_1, \dots, n(d)_k + \alpha_k))$

end for

once again to $\mathcal{D}((\alpha_1 + n_1, \dots, \alpha_k + n_k))$. Thus if we have a $\mathcal{D}((\alpha_1, \dots, \alpha_k))$ prior, we are acting as if $(\alpha_1, \dots, \alpha_k)$ was a piece of data we had already seen. For this reason $(\alpha_1, \dots, \alpha_k)$ is called the prior sample.

Now we can describe the initial algorithm for generating synthetic data. Let k be the total number of Census blocks on the U.S. map. We number the blocks from 1 to k and for each destination block d we form a k -dimensional vector $\overrightarrow{n(d)} = (n(d)_1, \dots, n(d)_k)$ where $n(d)_i$ is the number of people whose origin block is the Census block i and who commute to destination block d . For each destination block d we choose a prior sample $\overrightarrow{\alpha(d)}$ with $|\alpha(d)| = O(n(d))$ (the choice of $\overrightarrow{\alpha(d)}$ is discussed in Sections 4.4 and 4.5), condition on the vector $\overrightarrow{n(d)}$ to get a dirichlet $\mathcal{D}((n(d)_1 + \alpha(d)_1, \dots, n(d)_k + \alpha(d)_k))$ prior, and then use multinomial sampling with this prior to create a vector $\overrightarrow{m(d)} = (m(d)_1, \dots, m(d)_k)$ of $|m(d)| = O(n(d))$ synthetic individuals such that $m(d)_i$ synthetic people commute to destination block d from origin block i . This procedure is described in Algorithm 3.

Note that when the destination block d is clear from context, we will drop

the notational dependency on d and abbreviate $\overrightarrow{n(d)}$, $\overrightarrow{m(d)}$ and $\overrightarrow{\alpha(d)}$ as $\overrightarrow{n} = (n_1, \dots, n_k)$, $\overrightarrow{m} = (m_1, \dots, m_k)$ and $\overrightarrow{\alpha} = (\alpha_1, \dots, \alpha_k)$, respectively.

4.4 Revising the Privacy Definition

In this Section we evaluate our initial choice of privacy definition on the initial choice of anonymization algorithm. Contrary to intuition, it will turn out that these initial choices do not give good results for privacy. By analyzing the problem, we will show where intuition fails, and then we will revise the privacy definition to account for the discrepancy.

4.4.1 Analysis of Privacy

We start with an explanation behind the intuition that synthetic data should preserve privacy. Since the anonymized data is synthesized, it consists of synthetic people. Thus, linking real-world individuals to synthetic individuals (as in k -anonymity [122]) does not make sense. The prior sample $\overrightarrow{\alpha(d)}$ controls the amount of noise we add to the data generator. The larger the components of $\overrightarrow{\alpha(d)}$, the more noise is added. Now, we could have achieved ϵ -differential privacy by adding i.i.d. Laplace random variables with density $\frac{\epsilon}{4} \exp(-\epsilon x/2)$ and variance $8/\epsilon^2$ to the counts in each origin block [59]. For common values of ϵ (i.e., $\epsilon > 1$) this is a relatively small amount of noise per origin block. So intuitively, we also shouldn't need to add too much noise (i.e., a large prior sample) using our synthetic data generator.

This turns out not to be true. Let $\overrightarrow{\alpha(d)}$ be the prior sample for a destination

block d (note that the prior sample does not depend on the data). Differential privacy requires us to consider adversaries who have complete information about all but one of the individuals in the data; let us call this individual Bruce. Since a histogram of destination blocks has already been published, our adversary can use this information to determine Bruce's destination block d ; we need to determine how well Bruce's origin block is hidden from the adversary. Since the synthetic data is generated independently for every destination block, we only consider the synthetic data generated for destination block d . There are n individuals commuting to destination block d , and we generate m synthetic individuals for that destination block. To determine if the synthetic data satisfies ϵ -differential privacy (Definition 4.1) we have to find the maximum of

$$\frac{P((m_1, \dots, m_k) \mid (n_1, \dots, n_k), \vec{\alpha})}{P((m_1, \dots, m_k) \mid (n'_1, \dots, n'_k), \vec{\alpha})} \quad (4.2)$$

over all non-negative integer vectors $(m_1, \dots, m_k), (n_1, \dots, n_k), (n'_1, \dots, n'_k)$ with $\sum m_i = m, \sum n_i = \sum n'_i = n, n_i = n'_i + 1$ for some $i, n_j + 1 = n'_j$ for some $j \neq i$, and $n_h = n'_h$ for all $h \notin \{i, j\}$ (thus the difference between \vec{n} and \vec{n}' is Bruce's origin block). If this maximum is less than $\exp(\epsilon)$, the the synthetic data generator satisfies ϵ -differential privacy.

Theorem 4.1 *The maximum of Equation 4.2 is $\frac{m + \min_i(\alpha_i)}{\min_i(\alpha_i)}$ and this is at most $\exp(\epsilon)$ if and only if each $\alpha_i \geq \frac{m}{\exp(\epsilon) - 1}$.*

Proof. First note that

$$\begin{aligned} P(\vec{m} \mid \vec{n}, \vec{\alpha}) &= \int P(\vec{m} \mid \vec{p}) P(\vec{p} \mid \vec{n}, \vec{\alpha}) d\vec{p} \\ &= \frac{m!}{\prod_{i=1}^k m_i!} \frac{\Gamma(n + |\alpha|)}{\prod_i \Gamma(n_i + \alpha_i)} \int \prod_{i=1}^k p_i^{n_i + \alpha_i + m_i} d\vec{p} \\ &= \frac{m!}{\prod_i m_i!} \frac{\Gamma(n + |\alpha|)}{\prod_i \Gamma(n_i + \alpha_i)} \frac{\prod_i \Gamma(m_i + n_i + \alpha_i)}{\Gamma(m + n + |\alpha|)} \end{aligned}$$

so that

$$\begin{aligned}
\frac{P(\vec{m} \mid \vec{n}, \vec{\alpha})}{P(\vec{m} \mid \vec{n}', \vec{\alpha})} &= \frac{m_i + n_i + \alpha_i - 1}{n_i + \alpha_i - 1} \frac{n_j + \alpha_j}{m_j + n_j + \alpha_j} \\
&\leq \frac{m_i + n_i + \alpha_i - 1}{n_i + \alpha_i - 1} \\
&\leq \frac{m_i + \alpha_i}{\alpha_i} \\
&\leq \frac{m + \alpha_i}{\alpha_i}
\end{aligned}$$

where the first inequality is an equality when $m_j = 0$, the second inequality is an equality when $n_i = 1$ (note that $n_i = n'_i + 1$ and so its minimum value is 1), and the third inequality is an equality when $m_i = m$. Since $(m + \alpha_i)/\alpha_i$ is a decreasing function of α_i , the maximum of equation 4.2 is $\frac{m + \min_i \alpha_i}{\min_i \alpha_i}$ and this is at most $\exp(\epsilon)$ if and only if each $\alpha_i \geq \frac{m}{\exp(\epsilon) - 1}$. ■

The effect of Theorem 4.1 can be illustrated with the following example. Suppose there are one million people in destination block d and we choose to synthesize one million data points (i.e. we set $m = 1,000,000$). By setting $\epsilon = 7$ we are effectively requiring that the ratio in Equation 4.2 is at most $\exp(\epsilon) \approx 1096$. To achieve this privacy definition, we need to set $\alpha_i \geq 914$ for each i . In other words, for destination d , our prior sample $\vec{\alpha}$ has to have at least 914 people in each origin block, which in many cases will be more than n_i (the actual number of people in that origin block that commute to destination block d).

We can analyze where the intuition went wrong by examining the worst case in Theorem 4.1. The adversary has complete information about $n - 1$ individuals and is trying to determine whether or not Bruce (the remaining individual) commutes from origin block i . Suppose that (a) α_i achieves its minimum value in origin block i , (b) the adversary does not know of any individual in origin

block i , and (c) all m synthetic data points occur in origin block i . In this case, Equation 4.2 is maximized, and can be interpreted as the likelihood that Bruce commutes from origin block i divided by the likelihood that Bruce does not commute from origin block i . Note that this scenario leaks the most amount of information about Bruce; in the example described above, if the noise added to block i (α_i) is smaller than 914, the adversary can deduce that it is 1096 times more likely that Bruce commutes from origin block i than any other block.

However, it is also a scenario where the synthetic data is completely unrepresentative of the original data – in the original data at most one person could have come from origin block i , but all of the synthetic individuals came from origin block i . The probability of such an unrepresentative sample is at most $\frac{\Gamma(n+|\alpha|)\Gamma(m+\alpha_i+1)}{\Gamma(1+\alpha_i)\Gamma(m+n+|\alpha|)}$. As an example, let $m = n = 20$ and $\epsilon = 7$, and let all the α_j have their minimum values of $m/(\exp(7) - 1) \approx 0.018$. If there are even just $k = 2$ origin blocks then the probability of that event is approximately $e^{-24.9}$. The worst case is, therefore, an extremely unlikely event.

4.4.2 (ϵ, δ) -Probabilistic Differential Privacy:

Ignoring Unlikely Privacy Breaches

In Section 4.4.1 we saw that the worst-case privacy breaches are in outputs of the synthetic data generator that are extremely unlikely to occur. We say that a function $f(x)$ is *negligible* if $f(x)/x^{-k} \rightarrow 0$ as $x \rightarrow \infty$ for all $k > 0$. An output $\overrightarrow{m(d)}$ of the synthetic data generator is negligible if $P(\overrightarrow{m(d)} \mid \overrightarrow{n(d)}, \overrightarrow{\alpha(d)})$ is negligible in $|n(d)|$ (i.e. $P(\overrightarrow{m(d)} \mid \overrightarrow{n(d)}, \overrightarrow{\alpha(d)})/|n(d)|^{-k} \rightarrow 0$ for all $k > 0$).

Because these outputs are so unlikely, we would like to exclude them from the analysis of privacy. This leads us to the following privacy definition [99] which is a relaxation of differential privacy.

Definition 4.4 (Indistinguishability) *Let $\epsilon > 0$. Let \mathcal{A} be a randomized algorithm and let \mathcal{S} be the space of outputs of \mathcal{A} . Algorithm \mathcal{A} satisfies (ϵ, δ) -indistinguishability if for all $T \subset \mathcal{S}$ and all tables D_1 and D_2 differing in one row,*

$$P(\mathcal{A}(X_1) \in T) \leq e^\epsilon P(\mathcal{A}(X_2) \in T) + \delta(|X_2|)$$

where δ is a negligible function.

Now, for a given destination block d , by construction we have $|m(d)| = O(|n(d)|)$. The number of synthetic data sets for destination block d is the number of ways to assign $|m(d)|$ people to k origin blocks and is equal to $\binom{|m(d)| + k - 1}{k - 1}$, which is a polynomial in $|n(d)|$; thus, not all outputs occur with a negligible probability.

However, (ϵ, δ) -indistinguishability is an asymptotic privacy guarantee and so requires each destination block to have a large number of people commuting to it. Since our application contains many destination blocks with a small amount of commuters, this asymptotic privacy definition would not provide usable guarantees. For this reason we developed a different relaxation of differential privacy. First, we need to identify which outputs are “bad” in the sense that they leak too much information.

Definition 4.5 (Disclosure Set) *Let D be a table and \mathcal{D} be the set of tables that differ from D in at most one row. Let \mathcal{A} be a randomized algorithm and \mathcal{S} be the space of*

outputs of \mathcal{A} . The disclosure set of D , denoted by $\text{Disc}(D, \epsilon)$ is $\{S \in \mathcal{S} \mid \exists D_1, D_2 \in \mathcal{D}, |D_1 - D_2| = 1 \wedge |\ln \frac{P(\mathcal{A}(D_1)=S)}{P(\mathcal{A}(D_2)=S)}| > \epsilon\}$

Intuitively, the disclosure set for D is constructed as follows. For each tuple $t \in D$, let D^{-t} be the table D with tuple t removed. Treat D^{-t} as the adversary's background knowledge, so that the adversary is trying to guess the origin block for t . Now, if our data generator creates the synthetic data \vec{m} , then there are two likelihoods we are interested: the maximum likelihood of \vec{m} over all possible origin blocks for t , and the minimum likelihood of \vec{m} over all possible origin blocks for t . If the ratio of the two likelihoods is greater than e^ϵ then \vec{m} is an output that leaks too much information for some adversary. Consequently \vec{m} is in the disclosure set for D , and all \vec{m} in the disclosure set for D arise in this way.

Thus, to preserve privacy with high probability, we want the disclosure set to have a low probability, and so we arrive at the following privacy definition.

Definition 4.6 (Probabilistic Differential Privacy (pdp)) *Let \mathcal{A} be a randomized algorithm and let \mathcal{S} be the set of all outputs of \mathcal{A} . Let $\epsilon > 1$ and $0 < \delta < 1$ be constants. We say that \mathcal{A} satisfies (ϵ, δ) -probabilistic differential privacy (or, (ϵ, δ) -pdp) if for all tables D , $P(\mathcal{A}(D) \in \text{Disc}(D, \epsilon)) \leq \delta$.*

Note that in our application, a histogram of destination blocks has already been made publicly available. Thus the assumption that the adversary contains full information about all but one individuals in the data implies that the adversary knows the destination block of the remaining individual. Only the identity of the origin block is unknown to the adversary. For this reason we can partition the origin/destination dataset by destination block and treat each partition as a

disjoint dataset. Hence, a dataset satisfies (ϵ, δ) -probabilistic differential privacy if each partition of the dataset satisfies (ϵ, δ) -probabilistic differential privacy. Moreover, in such datasets, for an adversary with complete information about all but one individuals in the dataset, the probability that the adversary gains significant information about the remaining individual is at most δ .

4.4.3 (ϵ, δ) -Probabilistic Differential Privacy for Synthetic Data

We now describe the technical conditions under which the synthetic data generated by Algorithm 3 satisfies (ϵ, δ) -probabilistic differential privacy. We first state Theorem 4.2 which provides an efficient criterion for checking whether Algorithm 3 satisfies probabilistic differential privacy given the prior sample $\{\overrightarrow{\alpha(d)}\}_d$, and privacy parameters ϵ and δ . In a practical scenario, however, an application developer would prefer an efficient algorithm that given only ϵ and δ , returns a prior sample $\{\overrightarrow{\alpha(d)}\}_d$ such that Algorithm 3 is private. Hence, we next present Algorithm 4 that computes a prior sample to ensures (ϵ, δ) -probabilistic differential privacy. We conclude the section with a proof of Theorem 4.2. We will assume throughout that $\epsilon \geq \ln 3$, and that the number of synthetic individuals equals the number of original individuals for each destination (i.e., $m(d) = n(d)$).

To formally state the privacy guarantees for Algorithm 3 in terms of probabilistic differential privacy, we will need the following technical definition:

Definition 4.7 *Given constants $n, m, \alpha_1, \alpha_2, c$, define the function $f(x) = c \cdot (\alpha_1 + [x - 1]^+)$ (where, $[y]^+ = \max(y, 0)$). Then the reference 0-sample, denoted by*

$\rho(n, m, \alpha_1, \alpha_2, c)$ is the quantity:

$$\max_{x \in [0, n] \text{ s.t., } f(x) < m} \frac{\frac{\Gamma(m+1)}{\Gamma(f(x)+1)\Gamma(m-f(x)+1)} \frac{\Gamma(n+\alpha_1+\alpha_2)}{\Gamma(x+\alpha_1)\Gamma(n-x+\alpha_2)}}{\frac{\Gamma(m+n+\alpha_1+\alpha_2)}{\Gamma(x+f(x)+\alpha_1)\Gamma(n-x+m-f(x)+\alpha_2)}}$$

where the max is taken over all integers x in the range $[0, n]$.

Theorem 4.2 Let D be a data set, let $\epsilon > \ln 3$ and let $m(d) = n(d)$ for each destination block $d \in \{1, 2, \dots, k\}$. Algorithm 3 with prior sample $\{\overrightarrow{\alpha(d)}\}_d$ satisfies (ϵ, δ) -probabilistic differential privacy if for each destination block d , the reference 0-sample $\rho\left(n(d), m(d), \min_i(\alpha(d)_i), |\alpha(d)| - \min_i(\alpha(d)_i), e^\epsilon - 1\right)$ is at most $\delta(e^\epsilon - 2)/(2ke^\epsilon)$.

We next present Algorithm 4 that given ϵ and δ returns a prior sample such that Algorithm 3 satisfies privacy. The following observations guide our choice of the prior sample. First, while any $\{\overrightarrow{\alpha(d)}\}_d$ that satisfies Theorem 4.2 is sufficient for privacy, we would like to pick the one that ensures the maximum utility. Second, since the privacy criterion depends only on $\min_i \alpha(d)_i$ (for a given prior sample size) we always choose a *uniform* prior sample; i.e., $\forall d, \forall i, \alpha(d)_i = |\alpha(d)|/k$. Third, we observe that when $\overrightarrow{\alpha(d)}$ is uniform, increasing $|\alpha(d)|$ (which is the same as increasing each $\alpha(d)_i$) decreases the probability of a privacy breach. Intuitively, increasing the prior sample size introduces more noise into each block, and thus ensures more privacy. Therefore, the simple binary search algorithm presented in Algorithm 4 returns a uniform prior sample $\{\overrightarrow{\alpha(d)}\}_d$ with the smallest prior sample size that guarantees privacy.

In general, we found that the prior sample size required for ϵ -differential privacy was orders of magnitude larger than the prior sample size required for (ϵ, δ) -probabilistic differential privacy. Table 4.1 illustrates this difference; we show the $\alpha(d)_i$ per block required for ϵ -differential privacy (from Theorem 4.1)

Algorithm 4: Choose_Prior_Sample

Input: $\epsilon, \delta, k, \forall d, |n(d)| (|m(d)| = |n(d)|)$

Set $err = 0.0001$.

for all destination blocks d **do**

// Upper bound definitely guarantees privacy.

 Set $\alpha(d)_{ub} = \frac{m(d)}{e^\epsilon - 1}$, and $\delta_{ub} = 0$

// Lower bound is an infinitesimally small number; does not guarantee privacy.

 Set $\alpha(d)_{lb} = \eta$.

 Set $\alpha(d)_{mid} = (\alpha(d)_{lb} + \alpha(d)_{ub})/2$.

 Set $\delta_{mid} = \rho(|n(d)|, |m(d)|, \alpha(d)_{mid}, (k - 1)\alpha(d)_{mid}, e^\epsilon - 1) \times \frac{2ke^\epsilon}{e^\epsilon - 2}$.

while $\delta_{mid} > \delta$ OR $\delta - \delta_{mid} > err$ **do**

if $\delta_{mid} > \delta$ **then**

// Probability of breach too high. Increase α_{mid} .

$\alpha(d)_{lb} = \alpha(d)_{mid}$.

else

// Probability of breach too low. Decrease α_{mid} .

$\alpha(d)_{ub} = \alpha(d)_{mid}$.

end if

 Set $\alpha(d)_{mid} = (\alpha(d)_{lb} + \alpha(d)_{ub})/2$.

 Set $\delta_{mid} = \rho(|n(d)|, |m(d)|, \alpha(d)_{mid}, (k - 1)\alpha(d)_{mid}, e^\epsilon - 1) \times \frac{2ke^\epsilon}{e^\epsilon - 2}$.

end while

 Set for all $i \in [1, k], \alpha(d)_i = \alpha(d)_{mid}$.

end for

Return: $\{\overrightarrow{\alpha(d)}\}_d$

Table 4.1: Noise required per block ($\alpha(d)_i$) to ensure privacy of Algorithm 3.

Privacy Definition	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 20$	$\epsilon = 50$
ϵ -differential privacy	25×10^4	11×10^4	5.2×10^4	2×10^4
(ϵ, δ) -probabilistic differential privacy ($\delta = 10^{-6}$)	17.5	5.5	2.16	0.74

and (ϵ, δ) -probabilistic differential privacy (returned by Algorithm 4), respectively, for $n = m = 10^6$ individuals and a domain size of 10^4 blocks. For (ϵ, δ) -probabilistic differential privacy the amount of noise (the prior sample) is very small compared to the data and its influence on the output reduces as the amount of real data grows. Thus with probabilistic differential privacy we satisfy the intuition that synthetic data generation (Algorithm 3) can achieve privacy with little noise (as discussed at the beginning of Section 4.4.1).

Before we describe the utility of our synthetic data generator (Section 4.5), we prove Theorem 4.2.

Proof. of Theorem 4.2

Let $\vec{n}(d)$ be the histogram of origins for destination d in dataset D . Let \mathcal{A} denote Algorithm 3 with $\vec{\alpha}(d)$ as the prior sample for destination d . Let S be a synthetically generated output with histogram of origins $\vec{m}(d)$ for destination d .

The first step in the privacy analysis is identifying the disclosure set, $\text{Disc}(D, \epsilon)$. As illustrated in the beginning of this section, the worst disclosure for ϵ -differential privacy occurs when the output synthetic data, $\vec{m}(d)$, was un-

representative of the input data, $D = \overrightarrow{n(d)}$. We characterize the disclosure set in terms of the following representativeness metric:

Definition 4.8 (γ -Rep) Let S be the synthetic data generated by Algorithm 3 on input D . Let $\overrightarrow{m(d)}$, $\overrightarrow{n(d)}$ and $\overrightarrow{\alpha(d)}$ be the histogram for destination d in S , D and the prior sample, respectively. We define $S \in \gamma\text{-Rep}(D)$ ($\gamma > 1$) if

$$\forall i, \forall D, m(d)_i \leq \gamma(\alpha(d)_i + [n(d)_i - 1]^+)$$

where, $[y]^+ = \max(y, 0)$.

We can show that if $\epsilon = \ln(\gamma + 1)$, the disclosure set consists only of outputs that are not representative; i.e.,

$$\text{Disc}(D, \epsilon) \cap \gamma\text{-Rep}(D) = \emptyset \quad (4.3)$$

Let D and D' differ in the origin of one tuple t with destination d . Let t commute from origin block i in D and from origin block j in D' . If $\overrightarrow{n(d)'}^t$ denote the histograms of tuples with destination d in D' , then $n(d)_i = n(d)'_i - 1$ and $n(d)_j = n(d)'_j + 1$. Recall from the proof of Theorem 4.1,

$$\begin{aligned} \frac{P(\mathcal{A}(D) = S)}{P(\mathcal{A}(D') = S)} &= \frac{m(d)_i + n(d)_i + \alpha(d)_i - 1}{n(d)_i + \alpha(d)_i - 1} \frac{n(d)_j + \alpha(d)_j}{m(d)_j + n(d)_j + \alpha(d)_j} \\ &\leq \frac{m(d)_i}{n(d)_i + \alpha(d)_i - 1} + 1 \\ &\leq \gamma + 1 = e^\epsilon \quad \text{if } S \in \gamma\text{-Rep}(D) \text{ and } \epsilon = \ln(\gamma + 1) \end{aligned}$$

This means that bounding the probability of unrepresentative outputs by δ implicitly bounds the probability of the disclosure set. That is, we can guarantee (ϵ, δ) -probabilistic differential privacy if for every input D , the probability of the outputs that are not in $\gamma\text{-Rep}(D)$ is bounded by δ .

$$\forall D, P(S \notin \gamma\text{-Rep}(D)) \leq \delta$$

We first consider D with $k = 2$ and a single destination; i.e., $D = (n_1, n_2)$, prior sample is (α_1, α_2) , and the synthetic data is (m_1, m_2) . The set of all outputs is $\{(0, m), (1, m - 1), \dots, (m, 0)\}$. An output is unrepresentative if

$$m_1 \geq \gamma(n_1 + \alpha_1) \text{ or } m_2 \geq \gamma(n_2 + \alpha_2)$$

Let S_D^γ be the output such that $m_1 = \gamma(n_1 + \alpha_1)$ if $n_1 < n_2$, or $m_2 = \gamma(n_2 + \alpha_2)$, otherwise. Let $\mathcal{S}_D^{\gamma+}$ denote the set of all outputs with $m_1 \geq \gamma(n_1 + \alpha_1)$ if $n_1 < n_2$ ($m_2 \geq \gamma(n_2 + \alpha_2)$, if $n_1 \leq n_2$). Intuitively, S_D^γ is the output that is *just* unrepresentative, and $\mathcal{S}_D^{\gamma+}$ is the set of outputs that are unrepresentative⁴. As an output S becomes more and more unrepresentative, the probability that S is generated decreases. Hence, we can bound the probability of the unrepresentative outputs in terms of the probability of S_D^γ (Lemma 4.1).

$$P(\mathcal{A}(D) \in \mathcal{S}_D^{\gamma+}) = \frac{2(\gamma + 1)}{\gamma - 1} P(\mathcal{A}(D) = S_D^\gamma) \quad (4.4)$$

Next, note that by definition, the *reference 0-sample* is the maximum probability of S_D^γ , over all $D = (x, n - x)$, $x \in [0, n]$. Hence, we can bound the probability of S_D^γ by the *reference 0-sample*. Let $\alpha_{min} = \min(\alpha_1, \alpha_2)$ and $|\alpha| = \alpha_1 + \alpha_2$.

$$\max_D P(\mathcal{A}(D) = S_D^\gamma) \leq \rho(n, m, \alpha_{min}, |\alpha| - \alpha_{min}, e^\epsilon - 1) \quad (4.5)$$

Combining Equations 4.4 and 4.5, we derive the following condition for (ϵ, δ) -probabilistic differential privacy when $k = 2$.

\mathcal{A} satisfies (ϵ, δ) -probabilistic differential privacy, if

$$\frac{2e^\epsilon}{e^\epsilon - 2} \rho(n, m, \alpha_{min}, |\alpha| - \alpha_{min}, e^\epsilon - 1) \leq \delta \quad (4.6)$$

In order to extend this analysis to larger k (i.e., $k > 2$), note that there is no longer only a single output that is *just* unrepresentative. For instance, for $k = 4$

⁴This follows from our assumption that $e^\epsilon > 3$.

and $n = m = 10$, if $D = (3, 3, 2, 2)$ and $\vec{\alpha} = (1, 1, 1, 1)$, then all of the following outputs are *just* unrepresentative for $\gamma = 2$.

$$\{8, 2, 0, 0\}, \{8, 0, 2, 0\}, \{8, 0, 0, 2\}$$

$$\{8, 1, 1, 0\}, \{8, 1, 0, 1\}, \{8, 0, 1, 1\}$$

$$\{2, 8, 0, 0\}, \{0, 8, 2, 0\}, \{0, 8, 0, 2\}$$

$$\{1, 8, 1, 0\}, \{1, 8, 0, 1\}, \{0, 8, 1, 1\}$$

and so on ...

Nevertheless, we can still use Equations 4.4 and 4.5 to bound the probability of the unrepresentative outputs as shown below. Let \mathcal{S}^{bad} be the set of unrepresentative outputs; i.e.,

$$\mathcal{S}^{bad} = \{S \mid \exists i, m_i \geq \gamma(n_i + \alpha_i)\}$$

If \mathcal{S}_j^{bad} denotes the set of outputs such that $m_j \geq \gamma(n_j + \alpha_j)$, we have

$$\mathcal{S}^{bad} = \bigcup_j \mathcal{S}_j^{bad} \text{ and } P(\mathcal{A}(D) \in \mathcal{S}^{bad}) \leq \sum_{j=1}^k P(\mathcal{A}(D) \in \mathcal{S}_j^{bad}) \quad (4.7)$$

\mathcal{S}_j^{bad} can be written as follows:

$$\mathcal{S}_j^{bad} = \bigcup_{\ell=\gamma(n_j+\alpha_j)}^m \mathcal{S}_{j,\ell}^{bad} \text{ where } \mathcal{S}_{j,\ell}^{bad} = \{S \mid m_j = \ell\} \quad (4.8)$$

Now $\mathcal{S}_{j,\ell}^{bad}$ is the set of outputs where ℓ synthetic individuals are drawn from the j^{th} block and $m - \ell$ individuals are drawn from the rest of the blocks. The probability that Algorithm 3 outputs some $S \in \mathcal{S}_{j,\ell}^{bad}$ is, in fact, the same as the probability that Algorithm 3 (\mathcal{A}') with prior sample $(\alpha_j, |\alpha| - \alpha_j)$ outputs the synthetic data $S_\ell^{(2)} = (\ell, m - \ell)$ on input $D^{(2)} = (n_j, n - n_j)$. This property follows from the definition of multinomial sampling with a Dirichlet prior. Therefore,

$$P(\mathcal{A}(D) \in \mathcal{S}_{j,\ell}^{bad}) = P(\mathcal{A}'(D^{(2)}) = S_\ell^{(2)}) \quad (4.9)$$

Note that the $\{S_\ell^{(2)} \mid \ell = \gamma(n_j + \alpha_j), \dots, m\}$ is the same as $\mathcal{S}_{D^{(2)}}^{\gamma+}$, the set of outputs that are unrepresentative for $D^{(2)}$ for \mathcal{A}' . Therefore, if $\alpha_{min} = \min_{i=1}^k \alpha_i$,

$$\begin{aligned}
P(\mathcal{A}(D) \in \mathcal{S}_j^{bad}) &\leq \sum_{\ell=\gamma(n_j+\alpha_j)}^m P(\mathcal{A}(D) \in \mathcal{S}_{j,\ell}^{bad}) && \text{from Eq. 4.8} \\
&= \sum_{\ell=\gamma(n_j+\alpha_j)}^m P(\mathcal{A}'(D^{(2)}) = S_\ell^{(2)}) && \text{from Eq. 4.9} \\
&= P(\mathcal{A}'(D^{(2)}) \in \mathcal{S}_{D^{(2)}}^{\gamma+}) \\
&\leq \frac{2(\gamma+1)}{\gamma-1} P(\mathcal{A}'(D^{(2)}) = S_{D^{(2)}}^\gamma) && \text{from Eq. 4.4} \\
&\leq \frac{2(\gamma+1)}{\gamma-1} \rho(n, m, \alpha_{min}, |\alpha| - \alpha_{min}, e^\epsilon - 1) && (4.10)
\end{aligned}$$

Combining Equations 4.7 and 4.10 and , we get

$$\begin{aligned}
P(\mathcal{A}(D) \in \mathcal{S}^{bad}) &\leq \sum_{j=1}^k P(\mathcal{A}(D) \in \mathcal{S}_j^{bad}) \\
&\leq \sum_{j=1}^k \frac{2(\gamma+1)}{\gamma-1} \rho(n, m, \alpha_{min}, e^\epsilon - 1) \\
&= \frac{2ke^\epsilon}{e^\epsilon - 2} \rho(n, m, \alpha_{min}, |\alpha| - \alpha_{min}, e^\epsilon - 1)
\end{aligned}$$

Thus we have derived the condition for (ϵ, δ) -probabilistic differential privacy for arbitrary k .

\mathcal{A} satisfies (ϵ, δ) -probabilistic differential privacy, if

$$\frac{2ke^\epsilon}{e^\epsilon - 2} \rho(n, m, \alpha_{min}, |\alpha| - \alpha_{min}, e^\epsilon - 1) \leq \delta \tag{4.11}$$

■

Lemma 4.1 *Let D be a dataset with a single destination and $k = 2$. Let (n_1, n_2) ($n_1 < n_2$) represent the histogram of D . Let \mathcal{A} be Algorithm 3 with prior sample (α_1, α_2) . Let S_D^γ be an output such that $m_1 = \gamma(n_1 + \alpha_1)$ and $m_2 = m - m_1$, and $m = n$. Let $\mathcal{S}_D^{\gamma+}$ be the set of unrepresentative outputs with $m_1 \geq \gamma(n_1 + \alpha_1)$. Then*

$$P(\mathcal{A}(D) \in \mathcal{S}_D^{\gamma+}) = \frac{2(\gamma+1)}{\gamma-1} P(\mathcal{A}(D) = S_D^\gamma)$$

Proof. Consider two outputs $S_\ell = (\ell, m - \ell)$ and $S_{\ell+1} = (\ell + 1, m - \ell - 1)$. Then,

$$\begin{aligned} \frac{P(\mathcal{A}(D) = S_{\ell+1})}{P(\mathcal{A}(D) = S_\ell)} &= \frac{\ell + n_1 + \alpha_1}{\ell + 1} \frac{m - \ell}{m - \ell + n_2 + \alpha_2} \\ &= \left(1 - \frac{n_2 + \alpha_2}{m - \ell + n_2 + \alpha_2}\right) \bigg/ \left(1 - \frac{n_1 + \alpha_1 - 1}{\ell + n_1 + \alpha_1}\right) \end{aligned} \quad (4.12)$$

For every $S_\ell \in \mathcal{S}_D^{\gamma+}$, $\ell \geq \gamma(n_1 + \alpha_1)$ and $m - \ell \leq (n_2 + \alpha_2)$. Hence, from Equation 4.12,

$$\frac{P(\mathcal{A}(D) = S_{\ell+1})}{P(\mathcal{A}(D) = S_\ell)} \leq \left(1 - \frac{1}{2}\right) \bigg/ \left(1 - \frac{1}{\gamma + 1}\right) = \varphi \quad (4.13)$$

We can complete the proof using Equation 4.13.

$$\begin{aligned} &P(\mathcal{A}(D) \in \mathcal{S}_D^{\gamma+}) \\ &= \sum_{\ell=\gamma(n_1+\alpha_1)}^m P(\mathcal{A}(D)S_\ell) \\ &= P(\mathcal{A}(D)S_D^\gamma) \left(1 + \sum_{\ell=\gamma(n_1+\alpha_1)+1}^m \frac{P(\mathcal{A}(D) = S_\ell)}{P(\mathcal{A}(D) = S_D^\gamma)}\right) \\ &\leq P(\mathcal{A}(D) = S_D^\gamma) (1 + \varphi + \varphi^2 + \dots + \varphi^{m-\gamma(n_1+\alpha_1)}) \\ &\leq P(\mathcal{A}(D) = S_D^\gamma) (1 + \varphi + \varphi^2 + \dots) \\ &= P(\mathcal{A}(D) = S_D^\gamma) \left(\frac{1}{1-\varphi}\right) \leq P(\mathcal{A}(D) = S_D^\gamma) \left(\frac{1}{\frac{1}{2} - \frac{1}{\gamma+1}}\right) \\ &= \frac{2(\gamma+1)}{\gamma-1} P(\mathcal{A}(D) = S_D^\gamma) \end{aligned} \quad (4.14)$$

■

4.5 Revising the Algorithm

In this section we discuss several problems with the utility of Algorithm 3 and refine Algorithm 3 to make it produce more useful synthetic data. The first problem is that the resulting data may be very unrepresentative of the original

data (and therefore useless). For example, it is possible (albeit with very small probability) that in the synthetic data, all the workers commuting to New York city come from Boston (or worse, from San Francisco) even though this is not the case in the original data. The second problem is that with a large domain, the total amount of noise required to guarantee privacy (from Algorithm 4) may swamp most of the signal. These issues are not unique to the Algorithm 3 and our mapping application, but also to existing techniques (like [58] and [105]). We show how to ameliorate these effects by employing the probabilistic differential privacy as the privacy criterion. In Section 4.5.1 we will discuss when to throw away unrepresentative synthetic data, and in Section 4.5.2 we will discuss how to effectively shrink the size of the domain.

4.5.1 Accept/Reject

Randomized synthetic data generation algorithms produce unrepresentative outputs with small probability. Although rare, these events cause problems for data analysis. A simple technique to avoid unrepresentative outputs, which we call the *accept/reject* method, is to choose a “representativeness” metric and re-run the algorithm until we get an output which is representative of the input. If the algorithm generates an unrepresentative output with a probability at most p , then the expected number of steps before the accept/reject algorithm halts is at most $\frac{1}{1-p}$. However, special care must be taken to avoid privacy breaches.

Lemma 4.2 *Let $\text{Good}(D)$ be the set of outputs that are representative of D . If there exists D_1 and D_2 that differ in only one entry and $\text{Good}(D_1) \neq \text{Good}(D_2)$, then for every $\epsilon > 1$, Algorithm 3 combined with the accept/reject method does not satisfy ϵ -*

differential privacy.

Proof. Suppose D_1 and D_2 are two tables that differ in one entry such that $\text{Good}(D_1) \neq \text{Good}(D_2)$. Then, there exists an output S such that $S \in \text{Good}(D_1)$, but $S \notin \text{Good}(D_2)$. That is, $P(\mathcal{A}(D_2) = S) = 0$. The required result follows:

$$\frac{P(\mathcal{A}(D_1) = S)}{P(\mathcal{A}(D_2) = S)} = \infty \geq \epsilon, \forall \text{ finite } \epsilon$$

■

We illustrate the above lemma with an example. Intuitively, the number of synthetically generated individuals in origin block i , $m(d)_i$, should be proportional to $n(d)_i + \alpha(d)_i$. Hence, consider the following γ -representativeness metric for a table D :

$$\gamma\text{-Rep}(D) = \{S | \forall i, \forall d, m(d)_i \leq \gamma(\alpha(d)_i + [n(d)_i - 1]^+)\}$$

where $[y]^+$ is the non-negative part of y (i.e. $[y]^+ = \max(y, 0)$). In fact, we showed in the proof of Theorem 4.2 that any output S which is in $\gamma\text{-Rep}(D)$ is not in $\text{Disc}(D, \ln(\gamma + 1))$. Hence, one may expect that the synthetic data generator coupled with the accept/reject method that accepts only outputs in $\gamma\text{-Rep}(D)$ (denoted by $\mathcal{A}^{a/r}$) guarantees ϵ -differential privacy, for $\epsilon = \ln(\gamma + 1)$. On the contrary, however, this algorithm violates differential privacy because the probability $P(\mathcal{A}^{a/r}(D) = S)$ is no longer the same as $P(\mathcal{A}(D) = S)$. Consider, a synthetic dataset S with $m(d)_i = \gamma(\alpha(d)_i + [n(d)_i - 1]^+)$, $\gamma > 1$. Let D' be a table that differs from D in one entry such that $n'(d)_i = n(d)_i - 1$. Clearly, $S \notin \gamma\text{-Rep}(D')$. Hence,

$$\frac{P(\mathcal{A}^{a/r}(D) = S)}{P(\mathcal{A}^{a/r}(D') = S)} = \infty$$

Despite this result, the accept/reject method is compatible with the (ϵ, δ) -probabilistic differential privacy approach. Again, let p denote the probability

that the synthetic data generator outputs a synthetic dataset S that is not representative of the data ($S \notin \gamma\text{-Rep}(D)$). We show, in Lemma 4.3, that the accept/reject algorithm guarantees privacy with probability at least $(1 - p)$ when it only rejects synthetic datasets that *does not* belong to the set $\gamma\text{-Good}$, defined as follows:

$$\gamma\text{-Good}(D) = \bigcup_{D' : \begin{array}{l} |D-D'| \leq 1, \\ |D'-D| \leq 1 \end{array}} \gamma\text{-Rep}(D')$$

Before we prove Lemma 4.3, we remark on the implications for utility when we accept data from $\gamma\text{-Good}(D)$. We can easily show that the counts in a good S satisfy the following condition

$$S \in \gamma\text{-Good}(D) \Rightarrow \forall i, \forall d, m(d)_i \leq \gamma(n(d)_i + 1 + \alpha(d)_i)$$

Lemma 4.3 *Let $\mathcal{A}^{a/r}$ denote the synthetic data generator coupled with an accept/reject method which discards $S \notin \gamma\text{-Good}(D)$. If p is the maximum probability over all D that the synthetic data is not in $\gamma\text{-Rep}(D)$, then $\mathcal{A}^{a/r}$ guarantees (ϵ, p) -probabilistic differential privacy, where $\epsilon = \ln \frac{(\gamma+1)}{1-p}$.*

Proof. The proof follows from the observation that for every table D ,

$$\gamma\text{-Rep}(D) \cap \text{Disc}(D, \ln(\gamma + 1)) = \emptyset$$

Let $\delta(D) = P(S \notin \gamma\text{-Rep}(D))$; by definition of p , $\delta(D) \leq p$. Then, for any $S \in \gamma\text{-Rep}(D)$,

$$\left| \ln \frac{P(\mathcal{A}^{a/r}(D) = S)}{P(\mathcal{A}^{a/r}(D') = S)} \right| = \left| \ln \frac{P(\mathcal{A}(D) = S)}{P(\mathcal{A}(D') = S)} \right| + \left| \ln \frac{1 - \delta(D)}{1 - \delta(D')} \right| \leq \ln \frac{(\gamma + 1)}{1 - p}$$

Recall from the discussion of Lemma 4.2, that there is a breach of privacy for synthetic data S when $S \in \gamma\text{-Good}(D)$ and $S \notin \gamma\text{-Good}(D')$, which happens with a probability at most p . ■

Theorem 4.3 *Let \mathcal{A} be a synthetic data generator that satisfies (ϵ, δ) -probabilistic differential privacy. Let $\mathcal{A}^{a/r}$ denote the synthetic data generator coupled with an accept/reject method which discards $S \notin \gamma\text{-Good}(D)$, such that $\epsilon = \ln(\gamma + 1)$. Then, $\mathcal{A}^{a/r}$ guarantees (ϵ', δ) -probabilistic differential privacy, where $\epsilon' = \epsilon - \ln(1 - \delta)$.*

Proof. Since \mathcal{A} satisfies (ϵ, δ) -pdp,

$$P(\mathcal{A}(D) \in \text{Disc}(D, \epsilon)) \leq \delta$$

Since $\epsilon = \ln(\gamma + 1)$, we already know that

$$\gamma\text{-Rep}(D) \cap \text{Disc}(D, \epsilon) = \emptyset$$

Hence, we get

$$P(S \in \gamma\text{-Rep}(D)) \leq \delta$$

This coupled with Lemma 4.3 proves the required result. ■

4.5.2 Shrinking the Domain

For a randomized algorithm to satisfy differential privacy, or probabilistic differential privacy, it must add noise to every origin block so that for every origin block/destination block pair, there is a chance that the synthetic data will contain a synthetic individual that commutes from that origin block to that destination block. On the contrary, if noise is *not* added to some origin block, privacy can be breached as follows. Consider all the workers with jobs in a destination d (say, in New York). Suppose the adversary knows the origins of all the workers except one, and suppose the adversary knows that the last worker commutes from either o_1 (in Albany) or o_2 (in Schenectady), and no other worker commutes from o_1 or o_2 . Now, if noise is added to o_1 , but not to o_2 , and the output

synthetic data contains at least one worker commuting from o_2 , then it is clear that the last worker comes from o_2 and not from o_1 , thus breaching his privacy.

For Algorithm 3 to maintain privacy guarantees, for each destination block d , it needs to set a value $c(d)$ so that $\alpha(d)_i \geq c(d)$ for all origin blocks i . Since the data is sparse (i.e. the origin blocks for a destination block d are usually the surrounding census blocks and also major metropolitan areas, rather than the whole United States), most of this noise is concentrated in areas where there are no commuters to destination block d . In fact, the amount of noise is the sum of the $\alpha(d)_i$ for all blocks i that have no commuters to d . Thus the data generator may generate many strange fictitious commuting patterns.

To illustrate this point, let d be a block in Washington, D.C., and let $n = m = 10^6$. Let $\epsilon = \ln(50) = 3.912$. From Theorem 4.1, in order to satisfy ϵ -differential privacy, $c(d)_{diff} = 10^6 / (50 - 1) = 2.05 \times 10^5$. Remember that the total number of Census blocks on the U.S. map is 8 million. Hence, in order to preserve the privacy of a dataset with a million individuals, we are required to add 16 *billion* fake individuals (i.e., the prior sample size $|\alpha(d)| = 16 \times 10^9$)! Such noise will completely drown out all the signal.

Let $\delta = 10^{-6}$, and let us consider the situation with (ϵ, δ) -probabilistic differential privacy. In order to guarantee privacy, Algorithm 4 returns $c(d)_{pdp} = 0.74$ fake individuals per block. Though $c(d)_{pdp}$ is much smaller than $c(d)_{diff}$, it still causes spurious commute patterns. Since there are 8 million blocks in total, the total number of fake individuals ($|\alpha(d)|$) added to the dataset is $8 \times 10^6 \times 0.74 \approx 6 \times 10^6$. These 6 millions fake individuals are uniformly distributed across all the blocks; hence, about a million of them come from blocks on the West Coast. That is, out of a total of 7 million points $(\overrightarrow{n(d)} + \overrightarrow{\alpha(d)})$, $1/7^{th}$ of the population

commute from the West Coast to Washington, D.C., for work! This spurious commute pattern will be represented in the synthetic data, since on average the synthetic data, $\overrightarrow{m(d)}$, closely mirrors $\overrightarrow{n(d)} + \overrightarrow{\alpha(d)}$.

One way to tackle this problem is to reduce the size of the origin block domain. This reduces the number of blocks with no commuters to d , thus reducing the sum of $\alpha(d)_i$ for such blocks. We propose two ways of handling this: *coarsening the domain* and *probabilistically ignoring blocks*.

To coarsen the domain, much like generalization, we partition the origin blocks and merge the blocks within each partition. A partition must be chosen with care because it can leak information about the data. One way to do this is to cluster the data using a privacy-preserving clustering algorithm that is compatible with differential privacy, such as the k-means algorithm in the SULQ framework [24]. However, it is hard to evaluate the quality of such a clustering. Despite metrics that measure the quality of a clustering, numbers do not tell the whole story and an expert’s subjective judgment is often necessary. The effect of the expert (the data publisher generating synthetic data) choosing from multiple clustering on privacy is difficult to quantify and so it lies outside the differential privacy framework.

For this reason, we suggest that the partition be selected from publicly available data. For our application this is possible because of a previous release of similar data. Thus for a given destination block d we can coarsen the domain of its origin blocks using this partition. To plot (on a map) the origin of a synthetic individual, we first select the partition the individual is commuting from (as in Algorithm 3), and then we choose a specific point inside this partition using a density function derived from publicly available external data.

Theorem 4.4 *Let \mathcal{A} be a randomized algorithm that satisfies ϵ -differential privacy, or (ϵ, δ) -probabilistic differential privacy for our mapping application. If for each destination block d the domain of origin blocks is coarsened, then \mathcal{A} satisfies the same privacy criterion with the same parameters.*

Proof. The first step where \mathcal{A} generated synthetic data S_{coarse} in terms of the coarsened blocks clearly satisfies the original privacy guarantee. Since the second step generated synthetic data S from S_{coarse} and the external dataset, S does not leak any more information than S_{coarse} . ■

Even after coarsening, the domain may still be large. We trade off privacy versus the amount of noise added to regions with no commuters by probabilistically ignoring blocks. We say that a block i is *ignored* by the synthetic data generation algorithm if α_i is set to 0 in Algorithm 3; i.e., none of the points in the synthetic data having destination block = d will have origin block = i . Algorithm 5 illustrates our approach. For a given destination block d , let f_d be a function that assigns to every origin block a number in the interval $(0, 1]$ (note: this function must not depend on the data). For each origin block i that does not appear in the data (i.e., $n(d)_i = 0$), we keep it in the domain with probability $f_d(i)$ and ignore it with probability $1 - f_d(i)$. Effectively, we are reducing the size of the domain by throwing out origin blocks when creating synthetic data for destination block d . Note that it is important to choose the vector $\overrightarrow{\alpha(d)}$ (in particular, to determine the minimum value of any $\alpha(d)_i$) before shrinking the domain (and for those i that do not belong to the domain, α_i is set to 0). Algorithm 3 is then run using this new domain and $\overrightarrow{\alpha(d)}$. We can quantify the loss in privacy due to applying Algorithm 5 for each destination block d followed by Algorithm 3 with the following theorem:

Algorithm 5: Sample_Domain

Input: $\vec{n(d)}$, function $f_d : \{1, \dots, k\} \rightarrow (0, 1]$
Select $\vec{\alpha(d)}$ so that Theorem 4.2 is satisfied
New_Domain = \emptyset
for $i = 1..k$ **do**
 if $n(d)_i > 0$ **then**
 New_Domain = New_Domain $\cup \{i\}$
 else
 Let X be a binomial($f_d(i)$) random variable
 if $X == 1$ **then**
 New_Domain = New_Domain $\cup \{i\}$
 else
 $\alpha(d)_i = 0$
 end if
 end if
end for
Return: New_Domain

Theorem 4.5 *Let \mathcal{A} be a randomized algorithm that satisfied (ϵ, δ) -probabilistic differential privacy. Let \mathcal{C} be the randomized algorithm that chooses the domain of origin blocks for each destination block d using Algorithm 5 and then applies \mathcal{A} on this new domain. Then \mathcal{C} satisfies (ϵ', δ) -probabilistic differential privacy, where $\epsilon' = \epsilon + \max_i \ln(1/f_d(i)) + \max_i \lceil \alpha(d)_i \rceil \ln 2$.*

The interested reader is referred to Appendix A for the proof.

4.6 Experiments

The goal of combining probabilistic differential privacy with synthetic data generation was to develop a system that can be used for practical distribution of products from statistical agencies. In order to assess our progress towards that goal, we applied Algorithms 3, 4 and 5 to public-use data from the Census Bureau’s OnTheMap (OTM) microdata files (<http://lehdmap2.did.census.gov/themap/>). The versions of those files that are available for public use are themselves synthetic data. However, in our experimental evaluation we treat the Census Bureau’s released file as if it were the ground truth. Thus we measure the privacy protection and the utility of our experimental synthetic data against the “gold standard” of the OTM data.

Although much privacy research has focused on answering range queries [13, 105, 129, 135], in contrast to this work, we decided to evaluate the quality of the data using a measure for which we did *not* explicitly tune our anonymization algorithm. We computed the average commute distance for each destination block and compared it to the ground truth from OnTheMap. Note that the domain covers a two-dimensional surface since the average commute distance is not a linear statistic.

For our data evaluation, we selected a subset of OTM Version 2 data such that all destination workplaces are in Minnesota, yielding 1,495,415 distinct origin/destination block pairs (O/D pairs) which contain a commuter traveling from the origin block to the destination block. Many of these O/D pairs are singletons (i.e., they had only one commuter).

The actual partition of the United States into Census blocks comes from the

Census 2000 Summary File⁵ 1, which also contains information about the block characteristics and population counts. 8,206,261 blocks cover the 50 United States and the District of Columbia. We also used data from the previously published Census Transportation Planning Package (CTPP)⁶, which contains data similar to the O/D pairs from OnTheMap, based on Census 2000. We used CTPP to reduce the size of the domain for the origin blocks.

A preliminary data analysis using the CTPP data revealed that in Minnesota, 50% of all O/D work commutes were less than 6.23 miles, 75% less than 13.1 miles, and 90 percent less than 21.54 miles. Commutes longer than 200 miles were beyond the 99th percentile; so we ignored all origin blocks that were over 200 miles from their destination (and we also suppressed such O/D pairs in the data). As a result of this pruning, the maximum size of the domain of origin blocks in the experiments is 233,726.

We selected the minimum $\alpha(d)_i$ values using Theorem 4.2 with $\epsilon = 4.6$ and $\delta = 0.00001$. Recall that ϵ represents the maximum allowed disclosure due to the generation of synthetic data, and δ represented the maximum probability of a breach of ϵ -privacy. We also used Algorithm 5 to shrink the domain and the probability function f_d that is used in Algorithm 5 was created based on the CTPP data. The probability function f_d was roughly proportional to the histogram of commute distances as determined by the CTPP data. The maximum of the $\lceil \alpha(d)_i \rceil$ was 1 and the minimum value of $f_d(i)$ was 0.0378. Thus the additional disclosure due to shrinking the domain was $\ln(1/0.0378) + \ln 2 \leq 4$. The overall ϵ' for the procedure was 8.6 and disclosure probability δ was = 0.00001.

The results presented in this section are those of 120 blocks that are selected

⁵<http://www.census.gov/Press-Release/www/2001/sumfile1.html>

⁶<http://www.fhwa.dot.gov/ctpp/>

uniformly at random, averaging 15 primary job holders per destination block. For each block we computed the length of the average commute to that block, and compared the corresponding average commute distances in the OTM data to the synthetic data that we generated.

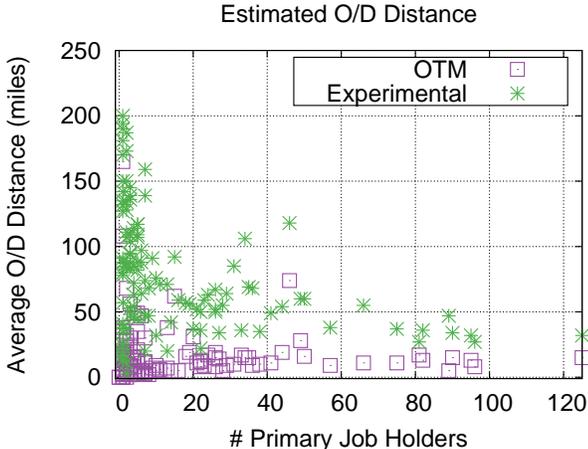


Figure 4.1: Average commute distance in the synthetic data vs # primary job holders ($\epsilon' = 8.6, \delta = 10^{-5}, k = 233, 726$).

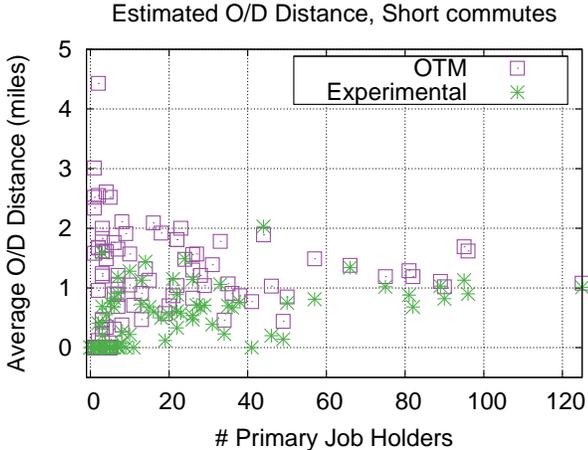


Figure 4.2: Average commute distance in the synthetic data vs # primary job holders ($\epsilon' = 8.6, \delta = 10^{-5}, k = 233, 726$) for short commutes.

In experiment 1, our privacy algorithm added an additional 419 primary job holders to each destination with a $\min \alpha(d)_i$ of 0.01245. Figure 4.1 shows the

relation between average commute distance as measured in OTM and in our experimental data. Each point in the figure corresponds to a particular destination block. The x -axis is the number of people whose primary job is in the destination block and the y -axis is the average commute distance to that destination block. It is clear that the experimental shape of f_d and values of $\min f_d$ and $\min \alpha_i$ admitted too many distant synthetic workers. The synthetic data overestimated the number of commuters with long commutes. This effect is strongest when the destination block has few workers and diminishes as the number of workers increases.

Figure 4.2 shows the same plot restricted to short commutes (i.e., those commutes that are shorter than the 6.23 miles which is the median commute distance in CTPP). Here the synthetic data better matches the ground truth. Note that the synthetic data slightly underestimates the commute distances (as a result of the fact that long commutes were overestimated, while the total number of commuters matched the ground truth). Again, the estimation error diminishes as the number of workers in a destination block increases.

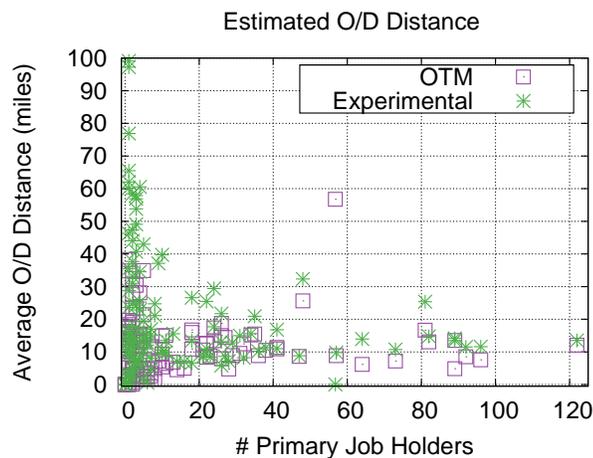


Figure 4.3: Average commute distance in the synthetic data vs # primary job holders ($\epsilon' = 8.6$, $\delta = 10^{-5}$, $k = 120, 690$).

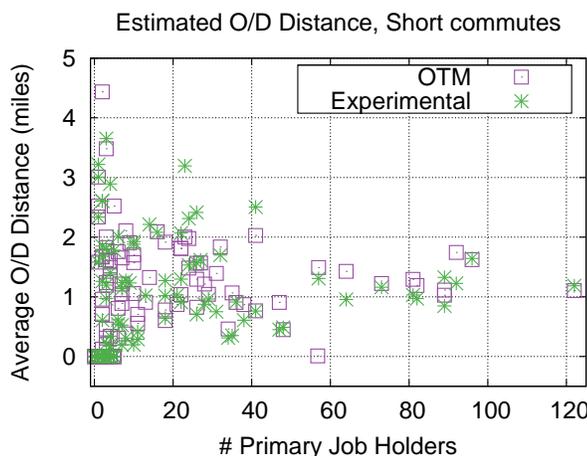


Figure 4.4: Average commute distance in the synthetic data vs # primary job holders ($\epsilon' = 8.6$, $\delta = 10^{-5}$, $k = 120,690$) for short commutes.

To further limit the effect of domain size on the estimation of commute distances, in our second experiment we restricted the domain of the origin blocks to be within 100 miles of the destination, keeping ϵ and δ values unchanged. Commutes of more than 100 miles were still beyond the 99th percentile for CTPP. This reduced the domain size to 120,690. The overall ϵ and δ values remained unchanged. In this case, $\min \alpha(d)_i = 0.039764$. Figures 4.3 and 4.4 show the results for all distances and for short commutes, respectively. The tighter restriction on the domain significantly enhanced data quality in the moderate commute distances, did not diminish quality in the short distances, and reduced the bias at all distances. We can see that the extremely long commutes are again over-estimated, but the destination blocks with long average commutes have few workers, so accurate estimates are not expected. This suggests that long commutes should be modeled separately when creating synthetic data. We leave this as an interesting avenue for future work.

4.7 Summary

In this chapter we showed that, with a little work, theoretical privacy definitions can be applied state-of-the-art ideas from statistical inference to create a practical yet provably private application. One remaining challenge is handling datasets with very sparse domains because noise must be spread throughout the domain. We found that this is necessary to hide outliers; if insufficient noise is added, an outlier appearing in the synthetic data is more likely due to a similar outlier in the real data than due to random noise. In our application we were able to use exogenous data and other techniques to help reduce the domain size. However, such data are not always available. Furthermore, even after reducing the domain size, the data was still sparse; as a result, the addition of noise to all parts of the reduced domain created many outliers. The distribution of commute distances was reasonable only for the study of commutes that were not extremely long.

We believe that judicious suppression and separate modeling of outliers may be the key since we would not have to add noise to parts of the domain where outliers are expected. For future work, we must consider methods for incorporating outlier identification, suppression, and modeling to the privacy and utility guarantees for the mapping application.

CHAPTER 5

RELATED WORK

Data privacy has been a focus of much research in the fields of computer science and statistics. In this chapter, we provide an overview of representative work. Most of the work can be broadly classified based on whether or not the data collector is trusted. In Section 5.1, we first discuss the trusted data collector scenario, the setting for all of the work presented in this dissertation. We then discuss the untrusted data collector scenario in Section 5.2.

5.1 Trusted Data Collector

In many scenarios, individuals contributing to the data trust the data collector to not breach their privacy. Examples of such data collectors are Census Bureaus and hospitals. However, these data collectors wish to share data with potentially untrusted third parties to engender research, and this poses a privacy risk. Trusted data collectors use a variety of techniques to limit this privacy risk, and these techniques can be broadly classified into four classes (each of these is discussed in detail in the following sections):

- *Access Control*: Only authorized users are allowed to view portions of the data based on an access control policy.
- *Anonymous Data Publishing*: An anonymous version of the data is published for public use.
- *Online Statistical Databases*: An online database that answers aggregate queries is exposed to the public.

- *Private Collaborative Computation*: Two or more data collectors can cryptographically share data for collaborative research in such a way that apart from the result, no other information about their respective data is disclosed to the other data collector.

5.1.1 Access Control

The three main approaches to access control are *discretionary access control* (DAC), *mandatory access control* (MAC), and *role-based access control* (RBAC). Discretionary access control allows owners of relations and views to grant and revoke authorizations to users [69]. Mandatory access control labels users and parts of the relations (views, rows, columns or even cells) with security clearances and security levels, respectively. Positive and negative authorizations are then specified on (clearance, level) pairs [26, 87]. Role-based access control groups users based on their roles [23]; a single user may have access to different parts of the data based in two different roles.

Many techniques have been used to enforce access control policies. A common method, discussed in Chapter 2, is to allow or deny queries based on whether a user is allowed to access information [23, 25, 109, 110]. Most of the existing techniques only allow positive authorizations on views. Under positive authorizations, a user u 's query Q_u is allowed if and only if the query can be answered using all the information that u is positive authorized to access. Rizvi et al.'s [109] techniques for positive authorizations on views are based on query equivalence [70]. Bertino et al. [23] proposed a solution when there are positive authorizations on arbitrary views and negative authorizations on base relations.

While Brodsky et al. [25] considered negative authorizations on arbitrary views, they used a privacy metric that allows multiple user queries to leak information about the negatively authorized views. Hence, their algorithms have the additional overhead of keeping track of the history of queries posed by a user.

Motro [98] presented a different technique for enforcing access control with positive authorization on views. Queries that can be answered using the views are answered correctly. However, if the query accesses unauthorized information, the database returns a partial answer that is computed only from the authorized views along with a description of the partial answer (for instance, “only the prescriptions of patients in ward #1212 have been returned”).

Access control is also enforced using multilevel relational databases [87]. These databases allow users to read and write data, and use mandatory access control at the granularity of individual cells; i.e., each cell is associated with a security level. Rather than denying queries that do not satisfy the access control policy, multilevel relations use a technique called *polyinstantiation* [73]. For instance, suppose Dr. Drake’s salary can only be accessed by users with security level *high*. The relation maintains two salary values with different security levels – the correct value with security level *high* and a random number (or null) with security level *low*. This ensures that only users with *high* security clearance can access the correct salary of Dr. Drake.

While access control methods that grant negative authorizations strive to disallow unauthorized disclosure, often sensitive information (defined by the negative authorizations) can be indirectly inferred using covert inference channels [63]. Inference channels occur either due to database design (constraints like functional and multivalued dependencies) [43, 120], or by using external in-

formation or mining patterns (like correlations between attributes). These inference channels are very much like the adversarial background knowledge in anonymous data publishing. However, to the best of our knowledge, there has been work only on modeling inference channels that are known to the system.

Ozsoyoglu and Su [120] proposed techniques to combat inference channels caused by functional and multivalued dependencies in multilevel relational databases. The techniques involve changing the security levels (to higher levels) of the least number of attributes. Dawson et al. [43] imposed constraints on the function that assigns security levels to data values based on known constraints in the database. For instance, if salary is determined by education qualifications, and salary has a security level *high*, then education qualifications should also be labeled *high*. Given such constraints and a lattice on security levels, Dawson et al. proposed efficient algorithms to find the security labeling of the data that satisfies the constraints while releasing the most information.

Perfect privacy [96] is the first step toward modeling unknown inference channels (in the form of correlations between attributes). However, enforcing perfect privacy for conjunctive queries is shown to be intractable. This model has been extended to allow for arbitrary dependencies in the database and views which have already been published [47]. Checking perfect privacy in this setting is even harder. Allowing only queries which disclose no information about the Q_S might be too restrictive. Dalvi et al. [42] presented a less restrictive privacy guarantee based on asymptotic conditional probabilities. Stoffel et al. [119] proposed certain answer privacy, yet another privacy guarantee, that disallows only those Q_V which disclose that some tuple is for sure in the output of Q_S .

Hippocratic databases [10] are a recently proposed design principle for building database systems that regulate the sharing of private data with third parties. Such a solution requires both the individuals who provide data and the databases that collect it to specify privacy policies describing the *purposes* for which the data can be used and the *recipients*, or third parties, who can access parts of the data. The policies are specified using a policy specification language like APPEL [60], which satisfies the P3P standard [61]. The recent focus on specifying privacy policies based on *purposes* rather than identities or roles requires new techniques for authentication beyond standard password based techniques, coupled with legal enforcement. A Hippocratic database also needs other functionality, like query rewriting for disclosure limitation [80], support for data retention, and techniques for maintaining audit trails [9]. Snodgrass et al. [118] proposed schemes for auditing the operations of a database such that any tampering with the audit logs can be detected. Such a solution can guard against the database's manipulation of the audit logs, thus giving assurance of eventual post-breach detection.

Finally, there has been work on publishing XML documents and ensuring access control on these documents [95, 132]. Miklau et al. [95] used cryptographic techniques to ensure that only authorized users can access the published document. Yang et al. [132] proposed publishing partial documents that hide sensitive data in the presence of known constraints in the XML schema.

5.1.2 Anonymous Data Publishing

Statistical agencies, especially those related to the Census, have studied the problem of data privacy for the past half a century. The Census Bureau collects information about individuals, and provides researchers with access to public-use microdata (PUMS). The Census uses a variety of sanitization techniques to ensure privacy and utility of the published data. Hence, there is a huge amount of research on anonymization algorithms, privacy metrics, and techniques to use these anonymized datasets. While many of the proposed techniques in this literature guarantee utility of the anonymous data, very little is known about their privacy properties.

Historically, the main focus of the Census literature has been to limit the ability of an attacker to re-identify individuals from the anonymized data. Hence, techniques in this literature attempt to identify and protect the privacy of sensitive entries in *contingency tables* ([37, 38, 39, 52, 53, 64, 117]). A contingency table T_C^A of a relation T is a table of counts that represents the complete cross-classification of T over all its attributes $\mathcal{A} = A_1, \dots, A_k$. That is, the contingency table T_C has one cell for every value $i \in \text{dom}(A_1) \times \dots \times \text{dom}(A_k)$, and records the number of times i appears in the table in cell $T_C[i]$ (equivalently, `SELECT COUNT(*) GROUP BY A_1, \dots, A_m`). The contingency table $T_C^{A'}$ is called a *marginal* of T if it cross-classifies only the attributes in $\mathcal{A}' = A_{i_1}, \dots, A_{i_j}$ (equivalently, `SELECT COUNT(*) GROUP BY A_{i_1}, \dots, A_{i_j}`). A nonzero cell value in the contingency table is considered sensitive if it is smaller than a fixed threshold k , which is usually chosen in an ad-hoc manner. An alternate approach is to determine a *safety range* or a *protection interval* for each cell [51], and to publish only those marginals which ensure that the feasibility intervals (i.e. upper and

lower bounds on the values a cell may take) contain the protection intervals for all cell entries.

Cell suppression [36, 37] is one of the earliest techniques proposed to protect the privacy of sensitive cells, where cells with low counts are simply deleted. Due to data dependencies caused by marginal totals that may have been previously published, additional related cell counts may also need to be suppressed. On the other hand, *generalization* techniques, described in Chapter 3, group cells with small counts in the contingency table such that the new contingency table has cells with counts either 0 or at least k . This is the k -anonymity technique. All information about the distribution of the cells within each group is lost. Many techniques have been proposed for efficiently creating k -anonymous tables. Samarati et al. [114] proposed a binary search technique, for computing a k -anonymous table using full-domain generalization techniques. Bayardo et al. [18] modeled k -anonymization as an optimization problem between privacy and utility, and proposed an algorithm similar to a frequent itemset mining algorithm [11]. LeFevre et al. [81, 82] extended the approach of full-domain generalization and proposed an algorithm for returning all valid k -anonymous tables, again using techniques similar to frequent itemset mining. The problem of k -anonymization has been shown to be NP -hard [94], and approximation algorithms for producing k -anonymous tables have been proposed [7].

Clustering techniques have been proposed to protect sensitive cells. *Microaggregation* techniques group low count cells, and publish the centroid of each group [93]. Aggarwal et al. [6] proposed the *condensation* technique that clusters data in T using a distance metric on the domain $(\text{dom}(A_1) \times \dots \times \text{dom}(A_k))$ and then publishes averages and variances for each cluster.

Recent research in the statistics community has focused on generating *synthetic microdata*. *Data swapping*, the simplest of the synthetic data generation techniques, involves moving data entries from one cell in the contingency table to another so that the table remains consistent with a set of published marginals [41, 48, 56]. Newer techniques for partial and fully synthetic populations [3, 106, 111] preserve statistical properties of the original data using resampling and multiple-imputation techniques [112]. While much research has focused on deriving the variance and confidence intervals for various estimators from synthetic data [102, 106], with the exception of Reiter [107], there has been little research on deriving formal guarantees of privacy for such an approach. In Chapter 4 we studied the privacy properties of one such technique and adapted it to meet formal privacy guarantees.

Chawla et al. [29] proposed a formal definition of privacy for published data, based on the notion of *blending in a crowd*, to protect against re-identification attacks. Here privacy of an individual is said to be protected if an adversary cannot isolate a record having attributes similar (according to a suitably chosen distance metric) to those of a given individual without being sufficiently similar to several other individuals; these other individuals are the crowd. The authors proposed perturbation and histogram-based techniques for data sanitization prior to publication. This formal notion of privacy presents a theoretical framework for studying the privacy-utility trade-offs of the proposed data sanitization techniques. However, due to the heavy reliance on an inter-tuple distance measure of privacy, the proposed definition of privacy fails to capture scenarios where identification of even a single sensitive attribute may constitute a privacy breach.

While re-identification is an important attack on anonymous data, adversaries can learn sensitive information about individuals even without re-identifying them in the data. Such disclosure has been called *attribute disclosure* in the Census literature [57]. ℓ -Diversity [89, 91], presented in Chapter 3, is a formal privacy definition that not only protects against attribute disclosure, but also guarantees privacy when adversaries have unknown background knowledge. However, as pointed out in Section 3.7, ℓ -diversity does not guarantee privacy in all adversarial settings; subsequent work has extended ℓ -diversity to handle set valued [130] and numeric [84] sensitive attributes, more powerful adversaries [32, 92], and to incorporate prior knowledge about the distribution of the sensitive attribute [85]. We briefly describe these below.

ℓ -Diversity assumes that in the input table every individual is represented by a unique tuple, and that each individual is associated with a single categorical attribute. Hence, ℓ -diversity would not apply if there were semantic relations between different values of the sensitive attribute. For instance, consider a group in an anonymized table where the diseases dyspepsia (s_1), gastroenteritis (s_2) and ulcer (s_3) appear in equal proportions. While this satisfies 3-diversity, it discloses the information that an individual in the that group has a stomach related disease. Semantically, we can extend semantic (c, ℓ) -diversity (Definition 3.12) to prevent such disclosures by also considering predicates like $eq(X[S], s_1 \vee s_2 \vee s_3)$ as sensitive. Xiao et al. [130] proposed a syntactic definition and efficient algorithms that limit such disclosures. Li et al. [84] extended ℓ -diversity to handle numeric attributes by considering ranges as sensitive values. While both the above papers correctly recognize that semantic information about the sensitive attribute makes more kinds of information sensitive, they fail to recognize that adversaries can possess more powerful kinds of knowl-

edge. For instance, in the former case, an adversary may also know negation statements of the form *“Bruce does not have a stomach related disease”*. In the latter case, negation statements of the form *“Bruce’s salary is not \$100,000”* is not very informative; it is more reasonable to consider inequality statements like *“Bruce’s salary is greater than \$100,000”*, or *“Bruce’s salary is between \$100,000 and \$150,000”*.

Talking about more expressive forms of background knowledge, Martin et al. [92] proposed a formal study of adversarial background knowledge. They consider adversaries whose background knowledge can be represented as propositional formulas over literals of the form $X[S] = s$, where X is an individual in the population and $s \in S$ is a value in the domain of the sensitive attribute. Such background knowledge can not only express negation statements (like those in ℓ -diversity) but also implications of the form *“If Bruce has the flu, then his spouse Ivy also has the flu”*. In fact, Martin et al. showed that any propositional formula is equivalent to a conjunction of implication statements; hence, they measured the power of an adversary in terms of the number of implication statements the adversary knows. They showed that checking whether a generalized table guarantees privacy given an arbitrary set of k implications is intractable. Nevertheless, the worst case k implications had a special structure, and this allows for efficiently checking for privacy under unknown adversarial knowledge. Chen et al. [32] extended this work by considering adversaries who know the correct sensitive information of k individuals, ℓ negation statements and m implication statements. Both these pieces of work do not change either the sensitive information specification or the measure of disclosure used in ℓ -diversity; they considered categorical attributes and positive disclosures.

Li et. al [85] proposed t -closeness, a syntactic privacy metric that attempts to extend ℓ -diversity by measuring privacy breaches as a distance between the adversary's posterior distribution and his prior distribution. t -Closeness requires that in a generalized table, the distribution of the sensitive attribute in every group of tuples that share the same quasi-identifier value should be close to the distribution of the sensitive attribute in the whole table.

Apart from proving privacy properties of generalization techniques, recent research has also focussed on generating synthetic data with provable privacy guarantees [17, 90, 105]. Barak et al. [17] proposed a solution to publish marginals of a contingency table. Publishing a set of noise infused marginals is not satisfactory; such marginals may not be consistent, i.e., there may not exist a contingency table that satisfies all these marginal contingency tables. Barak et al. solved this problem by adding noise to a small number of Fourier coefficients; any set of Fourier coefficients correspond to a (fractional and possibly negative) contingency table. They showed that only a "small" number of Fourier coefficients are required to generate the required marginals, and hence only a small amount of noise (proportional to the size of the marginal domain) is required. In order to create non-negative and integral marginals, the authors employed a linear program solution (in time polynomial in the size of multidimensional domain) to generate the final non-negative integral set of noise infused marginals. The synthetic contingency tables thus generated were shown to satisfy differential privacy and guarantee that the L-1 distance between the counts of the original and the synthetic contingency table is bounded with high probability. However, the quality bound and the running time of the linear program are proportional to the size of the domain, rather than the size of the data; hence this technique may not be suited for sparse data.

Rastogi et al. [105] proposed the $\alpha\beta$ algorithm for synthetically generating databases. Given an database instance I that is a subset of the domain of all tuples D , the $\alpha\beta$ algorithm creates a randomized database V by retaining tuples in I with probability $\alpha + \beta$ and adding tuples in $D - I$ with probability β . This algorithm satisfies (d, γ) -privacy (described in Section 4.2). The authors also show that for aggregate queries $Q : 2^D \rightarrow R$, $Q(I)$ can be estimated as:

$$\hat{Q}(I) = (Q(V) - \beta Q(D)) / \alpha \quad (5.1)$$

where $Q(V)$ and $Q(D)$ are the answers to the query Q on the randomized database V and the full domain D , respectively. $\hat{Q}(I)$ is shown to provably approximate $Q(I)$ with high probability.

While much research has focussed on publishing a single anonymized microdata table, as described in Chapter 4, these tables may not preserve enough useful information about the original data. Aggarwal [5] showed that for sparse high-dimensional data (with a large number of attributes), even a weak condition like k -anonymity dramatically reduces the utility of the data. Hence, multiple anonymous version of the data may need to be published. However, it has been shown that checking whether a set of generalizations is private can be $\#P$ -hard; even learning from multiple generalizations can be intractable [86]. Kifer et al. [77] proposed a technique to publish multiple generalizations of the original data in such a way that there are efficient algorithms for both checking for privacy (k -anonymity or ℓ -diversity) and utilizing these multiple tables. Yao et al. [133] analyzed the privacy of releasing multiple SQL views.

Finally, all of the above work assumes that a static snapshot of the data is published; there has been very little work that considers updates on the original data. There is a privacy risk in sequential releases of the same data, since

adversaries can link information about individuals across the releases. Xiao et al. [131] proposed efficient techniques to sequentially release tables after a batch of insert and delete operations.

5.1.3 Statistical Databases

The third scenario in the trusted data collector model is hosting a *query answering service*. This is addressed by the statistical database literature. In this model, the database answers only aggregate queries (COUNT, SUM, AVG, MIN, MAX) over a specified subset of the tuples in the database. The goal of a statistical database is to answer the queries in such a way that there are no positive or negative disclosures. Techniques for statistical database query answering can be broadly classified into three categories – query restriction, query auditing, data and output perturbation. Though the literature proposes a large number of techniques for ensuring privacy, only a few of the techniques provably guarantee privacy. Adam et al. [4] provide an excellent literature survey of early techniques. In this section we describe three approaches to statistical databases – *query restriction*, *query auditing*, and *output perturbation*.

Query restriction techniques ensure that privacy is not breached by specifying that a set of queries should not be answered. Legal queries are answered truthfully. These techniques focus on the case where a query specifies an aggregate function and a set of tuples C that are aggregated. The *query set size control* technique [64, 116] answers only those queries that access at least $|C| \geq k$ and at most $|C| \leq L - k$ tuples. Here k is a parameter and L is the size of the database. However, it was shown that snooping tools called *trackers* [46] can be used to

learn values of sensitive attributes. The *query set overlap control technique* [50] disallows queries which have a large intersection with previous queries.

Query auditing in statistical databases has been studied in detail. The *query monitoring* approach [50, 33] is an online version of the problem where the $(t + 1)^{th}$ query is answered or denied depending on the first t queries that have been asked. The decision is only based on the queries and not on the answers to those queries. Pure SUM queries and pure MAX queries can be audited efficiently but the mixed SUM/MAX problem is *NP*-hard. In the *offline auditing* problem [33, 34], the queries are presented all at once and the problem is to choose the maximum number of queries that can be answered. Kleinberg et al. [78] considered auditing SUM queries over boolean attributes and shows that it is co-*NP* hard to decide whether a set of queries uniquely determines one of the data elements. Kenthapadi et al. [76] studied the problem of *simulatable auditing*. This is a variant of the query monitoring approach where the decision to allow a query can depend on the answers to the previous queries as well. The main challenge in this model is that if a query answer is denied, information could be disclosed. Hence, the solutions proposed are such that any decision (to allow or deny a query) that is made by the database can also be simulated by the adversary.

Output perturbation techniques evaluate the query on the original data but return a perturbed version of the answer. Techniques here include returning answers to aggregate queries over a sample of the database [45], rounding off the answers to a multiple of a prespecified base b [40], and adding random noise to the outputs [21]. More recently, Dinur et al. [49] proved that in order to protect against an adversary who is allowed to ask arbitrarily many queries to

a database, the random noise added to the answers should be at least $\Omega(\sqrt{n})$, n being the number of tuples in the database. On the positive side, they also showed a technique that provably protects against a bounded adversary who is allowed to ask only $\mathcal{T}(n) \geq \text{polylog}(n)$ queries by using additive perturbation of the magnitude $\tilde{O}(\sqrt{\mathcal{T}(n)})$. Building on this result, Blum et al. [24] propose the SULQ framework that answers upto a sub-linear number of aggregate queries by adding Laplacian noise while guaranteeing differential privacy [58]. The amount of noise added to each query S depends on the *sensitivity* of Q . For an aggregate query that outputs a single real number, the sensitivity of Q , is the smallest number $S(Q)$, such that for any two databases D_1 and D_2 that differ in one record, $|Q(D_1) - Q(D_2)| \leq S(Q)$. In order to guarantee ϵ -differential privacy, it is sufficient add noise drawn from a Laplacian distribution with parameter $S(Q)/\epsilon$; i.e., more the sensitivity, more the noise. This technique is useful when the amount of noise added is small; i.e., when the Q has low sensitivity. Examples of queries with low sensitivity are histograms, linear aggregation queries and nearest-neighbour queries.

5.1.4 Private Collaborative Computation

In *private collaborative computation* there are n parties, each holding a secret input x_i , who want to collaboratively compute a function $f(x_1, \dots, x_n)$ such that (i) all the parties get the correct answer, and (ii) no information about the secret inputs is disclosed beyond whatever can be inferred from the output of the computation. For instance, the $n = 2$ parties could be the Department of Homeland Security of the United States (DHS) and an airline company (FlyUSA), their secret inputs are the list of purported terrorist names (x_{DHS}) and the list of pas-

sengers on an aircraft (x_{FlyUSA}), and the function is the intersection of these two lists ($f(x_{DHS}, x_{FlyUSA}) = x_{DHS} \cap x_{FlyUSA}$).

Private collaborative computation can be solved using secure multiparty computation techniques that are popular in the cryptography literature [28, 67, 101]. Most of the early work focused on building solutions for general functions by representing any function as a boolean circuit. In such general solutions, cryptographic protocols were used to simulate the computation of every gate in the boolean circuit description of the function. Hence, general solutions are perceived to be communication inefficient (of the order of the square of the number of parties involved for each gate in the boolean circuit being evaluated).

More recently, there has been much research proposing more efficient solutions to secure multiparty computations for specific functions. Du [54] proposed efficient protocols for various specific (secure) two-party computations problems. The commodity server model [19, 20] has been used for privately computing the scalar product of two vectors [55]. In the commodity server model, the two (or more) parties involved in the multiparty computation protocol employ the services of an untrusted third party to provide some randomness [19] or to help with some computation [55]. It is assumed that this untrusted third party does not collude with the players involved in the multiparty computation.

Agrawal et al. [8] employed commutative encryption techniques for information sharing across private database. Their techniques can be used to calculate the intersection and equijoin of two databases while disclosing only the sizes of the two databases. Clifton et al. [35] described methods to implement basic operations like secure sum, secure set union, secure set intersection, and secure scalar product using both encryption and additive randomization. These

primitives are used in various application scenarios to build multiparty protocols for private EM clustering, private association rule mining in horizontally partitioned data [74], and private association rule mining in vertically partitioned data [123].

One drawback which permeates the above literature is that there is no clear characterization of how much information is disclosed by the output of the protocol about the sensitive inputs.

5.2 Untrusted Data Collector

Randomization methods have been historically used to elicit accurate answers to surveys of sensitive yes/no questions. Respondents may be reluctant to answer such questions truthfully when the data collector (surveyor) is untrusted. Warner’s classical paper on *randomized response* [125] proposed a simple technique, where each individual X independently randomized the answer as follows: X answers truthfully with probability p_X , and lies with probability $(1 - p_X)$. Randomized response intuitively ensures privacy since no individual reports the true value. However, Warner did not formalize this intuition.

More recently, Agrawal et al. [12] proposed adding independent random noise to numeric data to help individuals mask their sensitive information while enabling data collectors to build good decision trees on the perturbed data. However, Kargupta et al. [75] showed that adding independent zero mean random noise does not necessarily preserve privacy. The techniques provided in the paper exploit spectral properties of random matrices to remove the noise, recover the original data, and thus the data collector could breach privacy. Huang

et al. [71] showed that the correlation between attributes is the key factor behind the attacks proposed in [75]. The paper proposes two techniques based on Principle Component Analysis (PCA) and the Bayes Estimate (BE) to reconstruct the original data from the randomized data.

Subsequent work [14, 62] generalized the randomized response technique to other domains. Evfimievski et al. [62] studied the problem where individuals share itemsets (e.g., a set of movies rented) with an untrusted server (e.g., an online movie rental company) in return for services (e.g., movie recommendations), and were the first to propose a formal privacy analysis of randomized response techniques using a semantic privacy definition called (ρ_1, ρ_2) -privacy. They invented a provably private randomization technique where users submit independently randomized itemsets to the server. They also proposed data reconstruction algorithms to help the server mine association rules from these randomized itemsets, and experimentally illustrated the accuracy of the reconstruction techniques.

CHAPTER 6

SUMMARY AND DISCUSSION

This dissertation addresses conceptual and practical issues in combating the privacy risk arising from the dissemination of personal information. Privacy, though a much debated concept, does not have a unique definition. In addition, the privacy risk from information disclosure must be modeled differently in different settings. One also needs practical algorithms to enforce these varied metrics of privacy. This dissertation presents formal privacy definitions and practical algorithms that can help data collectors to disseminate personal information while mitigating the privacy risk in various settings.

6.1 Summary of Contributions

In Chapter 2, we considered the setting where any disclosure of sensitive information is considered a breach of privacy. We adopted an existing privacy definition, called perfect privacy [96], to help decide whether answering a conjunctive query on a relational database leads to any disclosure of sensitive information (defined by another conjunctive query). This problem is known to be intractable; we proposed practical and efficient algorithms for a large set of subclasses of conjunctive queries.

In Chapter 3, we considered the problem of publishing “anonymous” aggregate information about populations of individuals while preserving the privacy of individual-specific information. Extant techniques publish information by ensuring that each individual is k -anonymous, or hidden in a crowd of k similar

individuals. Using simple attacks we illustrated that guaranteeing anonymity alone does not sufficiently eliminate the privacy risk from data publication; first, adversaries can discover sensitive information if all the similar individuals have the same sensitive information, and second, adversaries usually have background knowledge, and k -anonymity does not guarantee privacy against such adversaries.

This dissertation initiated a formal study of privacy definitions for data publishing. We presented a novel formal definition, ℓ -diversity, that ensures diversity of sensitive information, guards against adversaries with limited background knowledge, and allows practical algorithms for data publishing that release useful information.

It is well known that different settings need different definitions of the privacy risk arising from information disclosure. Since we proposed ℓ -diversity [89], many newer privacy metrics have been proposed. However, there is neither a mandate on the how to define privacy for new applications, nor a means to compare the plethora of privacy metrics in literature. We presented a novel framework for privacy definitions that, for the first time, distinguishes between semantic and syntactic privacy definitions. While syntactic privacy metrics can be efficiently enforced, semantic definitions explicitly state their assumptions about the data and the sensitive information, the adversary's background knowledge and the measure of disclosure, and thus, lay a common ground across different definitions. We believe that our framework will aid develop formal privacy definitions for newer applications. We urge that, in the future, privacy metrics must be proven equivalent to a semantic definition to help evaluate their strengths and weaknesses.

In Chapter 4, we analyzed the privacy of a real Census data publishing application. We presented the derivation of our techniques as a case study of anonymizing data with provable privacy guarantees. We identified that the data in our target application was sparse, and that none of the existing data publishing solutions are tailored to anonymize sparse data. We proposed privacy definitions and efficient algorithms to for this application.

6.2 Future Directions

Personal data is being increasingly collected on the Web. According to a very recent study [104], an estimated 8-10 GB of professional and user generated content is published on the web every day. There is still an estimated 3 TB of untapped private content generated per day in the form of emails, instant messages, browsing histories, user-click data, search logs, social networks, etc. If exploited, these new types of data can result in fascinating new applications and improved user experience on the Web. Harnessing this wealth of private data would require novel privacy-aware web-scale data management techniques, highlighting the increasing importance of privacy research.

There are a number of challenges arising from deploying privacy solutions to the Web. These include overcoming some of the challenges presented in this work, as well as, new techniques to incentivize users to specify their privacy preferences and to handle distributed data sharing. Below we discuss a few avenues for future work.

Distributed Data Anonymization

Web companies collect immense amounts of data everyday. This data is usually stored in geographically distinct locations. Hence, in order to extend the practical solutions presented in this work to the Web, one needs to study the locality properties of these algorithms so that they can be deployed on geographically distributed datasets.

A Suite of Provably Private Privacy-Aware Techniques

A variety of data analyses are performed on data collected on the Web. Rather than designing different privacy-aware algorithms for different uses of private data, it might be worthwhile to build a suite of primitive privacy-aware algorithms that can be composed together to satisfy the needs of various applications. Whether privacy definitions and algorithms are composable while retaining their privacy properties is an open question.

Moreover, new types of data need new privacy definitions. For instance, in traditional scenarios, adversaries are passive and restricted to existing incriminating information about the individual being attacked. However, newer applications allow adversaries to actively gather information by creating new friends in the social network, or rating new movies, etc. Characterizing such adversaries and their knowledge is a very interesting challenge.

Learning from Anonymized Views

Much current research has focused on publishing a single anonymized view of the data in a format similar to the original unanonymized data, so that existing analysis techniques can be directly applied to the anonymized data. Recent work has shown that the utility of the anonymized data can be improved by publishing multiple anonymous views of the data. New algorithms must be designed to manage and use these multiple views. Inference algorithms exist if the views are either SQL views, or marginal distributions of the original data. However, for the case of synthetic data generation, where the anonymized data is a random sample from a distribution, and for the case of output perturbation, where noise is added to the outputs of a set of prespecified queries, no general solutions exist.

Solving the above problem requires a formal study of learning from anonymized views. We would need to (a) propose a language for expressing these anonymized views, (b) explore and characterize the properties of the views that allow efficient learning, and (c) develop a suite of general purpose algorithms that can efficiently answer typical database queries and perform statistical analyses on the original data using the anonymized views.

Eliciting Privacy Preferences from Users

Though a system for sharing private data may guarantee utility and satisfy mathematical privacy definitions, it may not be adopted by users unless the privacy guarantees can be easily translated into user preferences. For instance, consider a system like del.icio.us that lets users tag web pages, and allows them to

share their tags with friends in an underlying social network. Suppose George tags <http://www.abc.xyz> as 'junk'. George can either share this tag with Tony, in which case Tony knows for sure that George tagged the page as 'junk', or not share this tag, in which case Tony has no information whether George tagged the above page. However, this sharing is binary, and George has no way to allow his tags to be anonymously shared with Tony.

In tomorrow's web applications, simple binary notions of sharing will not suffice. We already have technology that supports sharing with provable guarantees of partial disclosure. Unfortunately, we do not know how to elicit from users their preferences for the allowable limits on disclosure. Removing this barrier is very important to facilitate the adoption of these new applications.

BIBLIOGRAPHY

- [1] Privacy rights clearinghouse: A chronology of privacy breaches, <http://www.privacyrights.org/ar/chrondatabreaches.htm>.
- [2] A tough lesson on medical privacy: Pakistani transcriber threatens ucsf over back pay. *San Francisco Chronicle*, October 2003.
- [3] J. M. Abowd and S. D. Woodcock. Disclosure limitation in longitudinal linked data. *Confidentiality, Disclosure, and Data Access: Theory and Practical Applications for Statistical Agencies*, pages 215–277, 2001.
- [4] N. R. Adam and J. C. Wortmann. Security-control methods for statistical databases: A comparative study. *ACM Comput. Surv.*, 21(4):515–556, 1989.
- [5] C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB*, pages 901–909, 2005.
- [6] C. C. Aggarwal and P. S. Yu. A condensation approach to privacy preserving data mining. In *EDBT*, pages 183–199, 2004.
- [7] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Approximation algorithms for k-anonymity. *Journal of Privacy Technology (JOPT)*, 2005.
- [8] R. Agrawal, A. V. Evfimievski, and R. Srikant. Information sharing across private databases. In *SIGMOD Conference*, pages 86–97, 2003.
- [9] R. Agrawal, R. J. Bayardo Jr., C. Faloutsos, J. Kiernan, R. Rantzau, and R. Srikant. Auditing compliance with a hippocratic database. In *VLDB*, pages 516–527, 2004.
- [10] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocratic databases. In *VLDB*, pages 143–154, 2002.
- [11] R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases. In *VLDB*, 1994.
- [12] R. Agrawal and R. Srikant. Privacy preserving data mining. In *SIGMOD*, May 2000.

- [13] R. Agrawal, R. Srikant, and D. Thomas. Privacy preserving olap. In *Proceedings of the 23th ACM SIGMOD Conference on Management of Data*, June 2004.
- [14] S. Agrawal and J. R. Haritsa. A framework for high-accuracy privacy-preserving mining. In *ICDE*, 2005.
- [15] A. V. Aho, Y. Sagiv, and J. D. Ullman. Equivalences among relational expressions. *Siam Journal of Computing*, 1979.
- [16] F. Bacchus, A. J. Grove, J. Y. Halpern, and D. Koller. From statistical knowledge bases to degrees of belief. *Artificial Intelligence*, 87(1-2):75–143, 1996.
- [17] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy and consistency too: A holistic solution to contingency table release. In *PODS*, 2007.
- [18] R. J. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *ICDE*, 2005.
- [19] D. Beaver. Commodity-based cryptography (extended abstract). In *STOC '97: Proceedings of the 29th ACM Symposium on Theory of Computing*, pages 446–455, 1997.
- [20] D. Beaver. Server-assisted cryptography. In *NSPW '98: Proceedings of the 1998 Workshop on New security paradigms*, pages 92–106, 1998.
- [21] L. Beck. A security mechanism for statistical database. *ACM Transactions on Database Systems*, 5(3):316–338, 1980.
- [22] D. Bell and L. LaPadula. Secure computer systems: Unified expositions and multics interpretation. Technical Report ESD-TR-75-306, MITRE Corp., Bedford, MA, 1976.
- [23] E. Bertino, S. Jajodia, and P. Samarati. A flexible authorization mechanism for relational data management systems. *ACM Trans. Inf. Syst.*, 1999.
- [24] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: The sulq framework. In *PODS*, 2005.

- [25] A. Brodsky, C. Farkas, and S. Jajodia. Secure databases: Constraints, inference channels, and monitoring disclosures. *IEEE Transactions on Knowledge and Data Engineering*, 12(6):900–919, 2000.
- [26] S. Castano, M. Fugini, G. Martella, and P. Samarati. *Database Security*. Addison Wesley, 1995.
- [27] A. K. Chandra and P. M. Merlin. Optimal implementation of conjunctive queries in relational databases. In *STOC*, 1977.
- [28] D. Chaum, C. Crépeau, and I. Damgard. Multiparty unconditionally secure protocols. In *STOC*, 1988.
- [29] S. Chawla, C. Dwork, F. McSherry, A. Smith, and H. Wee. Toward privacy in public databases. In *Theory of Cryptography Conference*, 2005.
- [30] C. Chekuri and A. Rajaraman. Conjunctive query containment revisited. In *ICDT*, 1997.
- [31] B. Chen, L. Chen, R. Ramakrishnan, and D. R. Musicant. Learning from aggregate views. In *ICDE*, 2006.
- [32] B. Chen, K. Lefevre, and R. Ramakrishnan. Privacy skyline: Privacy with multidimensional adversarial knowledge. In *VLDB*, 2007.
- [33] F. Chin. Security problems on inference control for sum, max, and min queries. *J. ACM*, 33(3):451–464, 1986.
- [34] F. Chin and G. Ozsoyoglu. Auditing for secure statistical databases. In *ACM 81: Proceedings of the ACM '81 conference*, pages 53–59, 1981.
- [35] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu. Tools for privacy preserving data mining. *SIGKDD Explorations*, 4(2):28–34, 2002.
- [36] L. Cox. Network models for complementary cell suppression. *Journal of the American Statistical Association*, 90:1453–1462, 1995.
- [37] L. H. Cox. Suppression, methodology and statistical disclosure control. *Journal of the American Statistical Association*, 75, 1980.
- [38] L. H. Cox. Solving confidentiality protection problems in tabulations using network optimization: A network model for cell suppression in

- the u.s. economic censuses. In *Proceedings of the International Seminar on Statistical Confidentiality*, pages 229–245, International Statistical Institute, Dublin, 1982.
- [39] L. H. Cox. New results in disclosure avoidance for tabulations. In *International Statistical Institute Proceedings of the 46th Session*, pages 83–84, Tokyo, 1987.
- [40] T. Dalenius. A simple procedure for controlled rounding. *Statistik Tidsskrift*, 1981.
- [41] T. Dalenius and S. Reiss. Data swapping: A technique for disclosure control. *Journal of Statistical Planning and Inference*, 6:73–85, 1982.
- [42] N. N. Dalvi, G. Miklau, and D. Suci. Asymptotic conditional probabilities for conjunctive queries. In *ICDT*, pages 289–305, 2005.
- [43] S. Dawson, S. de Capitani di Vimercati, P. Lincoln, and P. Samarati. Maximizing sharing of protected information. *Journal of Computer and System Sciences*, 64(3), 2002.
- [44] J. Decew. Privacy. *Stanford Encyclopedia of Philosophy*, 2006.
- [45] D. Denning. Secure statistical databases with random sample queries. *ACM Transactions on Database Systems*, 5(3):291–315, 1980.
- [46] D. E. Denning, P. J. Denning, and M. D. Schwartz. The tracker: A threat to statistical database security. *ACM Transactions on Database Systems (TODS)*, 4(1):76–96, 1979.
- [47] A. Deutsch and Y. Papakonstantinou. Privacy in database publishing. In *ICDT*, 2005.
- [48] P. Diaconis and B. Sturmfels. Algebraic algorithms for sampling from conditional distributions. *Annals of Statistics*, 1:363–397, 1998.
- [49] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *PODS*, pages 202–210, 2003.
- [50] D. P. Dobkin, A. K. Jones, and R. J. Lipton. Secure databases: Protection against user influence. *ACM: Transactions on Database Systems (TODS)*, 4(1):76–96, March 1979.

- [51] A. Dobra. *Statistical Tools for Disclosure Limitation in Multiway Contingency Tables*. PhD thesis, Carnegie Mellon University, 2002.
- [52] A. Dobra and S. E. Feinberg. *Assessing the risk of disclosure of confidential categorical data*. Bayesian Statistics 7, Oxford University Press, 2000.
- [53] A. Dobra and S. E. Feinberg. Bounding entries in multi-way contingency tables given a set of marginal totals. In *Foundations of Statistical Inference: Proceedings of the Shores Conference 2000*. Springer Verlag, 2003.
- [54] W. Du. *A Study of Several Specific Secure Two-party Computation Problems*. PhD thesis, Purdue University, 2001.
- [55] W. Du and Z. Zhan. A practical approach to solve secure multi-party computation problems. In *New Security Paradigms Workshop 2002*, 2002.
- [56] G. T. Duncan and S. E. Feinberg. Obtaining information while preserving privacy: A markov perturbation method for tabular data. In *Joint Statistical Meetings*, Anaheim, CA, 1997.
- [57] G. T. Duncan and D. Lambert. Disclosure-limited data dissemination. *Journal of the American Statistical Association*, 81(393), 1986.
- [58] C. Dwork. Differential privacy. In *ICALP*, 2006.
- [59] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.
- [60] M. Langheinrich (editor). A p3p preference exchange language 1.0 (appel1.0). W3C Working Draft, February 2001.
- [61] M. Marchiori (editor). The platform for privacy preferences 1.0 (p3p1.0) specification. W3C Proposed Recommendation, January 2002.
- [62] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *PODS*, 2003.
- [63] C. Farkas and S. Jajodia. The inference problem: A survey. *SIGKDD Explorations*, 4(2):6–11, 2002.
- [64] I. P. Fellegi. On the question of statistical confidentiality. *Journal of the American Statistical Association*, 67:337:7–18, 1972.

- [65] S. Garfinkel. *Database nation: the death of privacy in the 21st century*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 2001.
- [66] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC, 2nd edition, 2003.
- [67] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC '87: Proceedings of the nineteenth annual ACM conference on Theory of computing*, pages 218–229, 1987.
- [68] S. Goldwasser and S. Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *STOC*, 1982.
- [69] P. P. Griffiths and B. W. Wade. An authorization mechanism for a relational database system. *ACM Trans. Database Syst.*, 1(3), 1976.
- [70] A. Halevy. Answering queries using views. *VLDB Journal*, 10(4):270–294, 2001.
- [71] Z. Huang, W. Du, and B. Chen. Deriving private information from randomized data. In *Proceedings of the 23th ACM SIGMOD Conference on Management of Data*, June 2004.
- [72] V. S. Iyengar. Transforming data to satisfy privacy constraints. In *KDD*, pages 279–288, 2002.
- [73] S. Jajodia, R. Sandhu, and B. Blaustein. Solutions to the polyinstantiation problem. *Information Security: An Integrated Collection of Essays*, 1, 1994.
- [74] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. In *DMKD*, 2002.
- [75] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *ICDM*, pages 99–106, 2003.
- [76] K. Kenthapadi, N. Mishra, and K. Nissim. Simulatable auditing. In *PODS*, 2005.
- [77] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In *SIGMOD*, 2006.

- [78] J. Kleinberg, C. Papadimitriou, and P. Raghavan. Auditing boolean attributes. *Journal of Computing Systems Science*, 66(1):244–253, 2003.
- [79] D. Lambert. Measures of disclosure risk. *Journal of Official Statistics*, 1993.
- [80] K. LeFevre, R. Agrawal, V. Ercegovac, R. Ramakrishnan, Y. Xu, and D. J. DeWitt. Limiting disclosure in hippocratic databases. In *VLDB*, pages 108–119, 2004.
- [81] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain k -anonymity. In *SIGMOD*, 2005.
- [82] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Mondrian multidimensional k -anonymity. In *ICDE*, 2006.
- [83] K. LeFevre, D. J. DeWitt, and R. Ramakrishnan. Workload-aware anonymization. In *KDD*, 2006.
- [84] J. Li, Y. Tao, and X. Xiao. Preservation of proximity privacy in publishing numeric sensitive data. In *SIGMOD*, 2008.
- [85] N. Li, T. Li, and S. Venkatasubramanian. t -closeness: Privacy beyond k -anonymity and ℓ -diversity. In *ICDE*, 2007.
- [86] J. De Loera and S. Onn. The complexity of three-way statistical tables. *SIAM J. Comput.*, 33(4):819–836, 2004.
- [87] L. Lunt, D. Denning, R. Schell, M. Heckman, and W. Shockley. The seaview security model. *IEEE Transactions on Software Engineering*, 16(6):593–607, 1990.
- [88] A. Machanavajjhala and J. Gehrke. On the efficiency of checking perfect privacy. In *PODS*, 2006.
- [89] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkatasubramanian. ℓ -diversity: Privacy beyond k -anonymity. In *ICDE*, 2006.
- [90] A. Machanavajjhala, D. Kifer, J. Abowd, J. Gehrke, and L. Vihuber. Privacy: From theory to practice on the map. In *ICDE*, 2008.

- [91] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkatasubramanian. ℓ -diversity: Privacy beyond k -anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1), 2007.
- [92] D. Martin, D. Kifer, A. Machanavajjhala, J. Gehrke, and J. Halpern. Worst case background knowledge for privacy preserving data publishing. In *ICDE*, 2007.
- [93] J. M. Mateo-Sanz and J. Domingo-Ferrer. A method for data-oriented multivariate microaggregation. In *SDP*, 1998.
- [94] A. Meyerson and R. Williams. On the complexity of optimal k -anonymity. In *PODS*, 2004.
- [95] G. Miklau and D. Suciu. Controlling access to published data using cryptography. In *VLDB*, pages 898–909, 2003.
- [96] G. Miklau and D. Suciu. A formal analysis of information disclosure in data exchange. In *SIGMOD*, 2004.
- [97] A. R. Miller. The assault on privacy – computers, data banks and dossiers. *University of Michigan Press, Ann Arbor, Mich.*, 1996.
- [98] A. Motro. An access authorization model for relational databases based on algebraic manipulation of view definitions. In *ICDE*, 1989.
- [99] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *39th ACM Symposium on Theory of Computing (STOC)*, 2007.
- [100] A. Ohrn and L. Ohno-Machado. Using boolean reasoning to anonymize databases. *Artificial Intelligence in Medicine*, 15(3):235–254, 1999.
- [101] M. Ben Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *STOC*, 1998.
- [102] T.E. Raghunathan, J.P. Reiter, and D.B. Rubin. Multiple imputation for statistical disclosure limitation. *Journal of Official Statistics*, 19:1–16, 2003.
- [103] R. Ramakrishnan and J. Gehrke. *Database Management Systems*. McGraw-Hill Science/Engineering/Math, 3rd edition, April 2002.

- [104] R. Ramakrishnan and A. Tomkins. Towards a people web (abstract). *IEEE Computer*, 40(8), 2007.
- [105] V. Rastogi, D. Suciu, and S. Hong. The boundary between privacy and utility in data publishing. Technical report, University of Washington, 2007.
- [106] J.P. Reiter. Inference for partially synthetic, public use microdata sets. *Survey Methodology*, 29(2):181–188, 2003.
- [107] J.P. Reiter. Estimating risks of identification disclosure for microdata. *Journal of the American Statistical Association*, 100:1103 – 1113, 2005.
- [108] U.C.Irvine Machine Learning Repository.
<http://www.ics.uci.edu/mllearn/mlrepository.html>.
- [109] S. Rizvi, A. Mendelzon, S. Sudarshan, and P. Roy. Extending query rewriting techniques for fine-grained access control. In *ACM SIGMOD international conference on Management of data*, 2004.
- [110] A. Rosenthal and E. Sciore. View security as the basis for data warehouse security. In *DMDW*, 2000.
- [111] D. B. Rubin. Discussion statistical disclosure limitation. *Journal of Official Statistics*, 9(2), 1993.
- [112] D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. Wiley-Interscience, 2004.
- [113] P. Samarati. Protecting respondents’ identities in microdata release. In *TKDE*, pages 1010 – 1027, 2001.
- [114] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, CMU, SRI, 1998.
- [115] Y. P. Saraiya. Polynomial-time program transformations in deductive databases. In *PODS*, 1990.
- [116] J. Schlorer. Identification and retrieval of personal records from a statistical bank. In *Methods Info. Med.*, 1975.

- [117] A. Slavkovic and S. E. Feinberg. Bounds for cell entries in two-way tables given conditional relative frequencies. In *Privacy in Statistical Databases*, 2004.
- [118] R. T. Snodgrass, S. Yao, and C. S. Collberg. Tamper detection in audit logs. In *VLDB*, pages 504–515, 2004.
- [119] K. Stoffel and T. Studer. Provable data privacy. In *DEXA*, pages 324–332, 2005.
- [120] T. Su and G. Ozsoyoglu. Controlling fd and mvd inferences in multilevel relational database systems. *IEEE Transactions of Knowledge and Data Engineering*, 3(4), 1991.
- [121] L. Sweeney. Uniqueness of simple demographics in the u.s. population. Technical report, Carnegie Mellon University, 2000.
- [122] L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.
- [123] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *KDD*, pages 639–644, 2002.
- [124] R. van der Meyden. The complexity of querying indefinite data about linearly ordered domains. *JCSS*, 54(1), 1997.
- [125] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 1965.
- [126] S. Warren and L. Brandeis. The right to privacy. *Harvard Law Review*, 1890.
- [127] A. Westin. *Privacy and Freedom*. Atheneum, 1967.
- [128] L. Willenborg and T. de Waal. *Statistical Disclosure Control in Practice*. Springer-Verlag, 1996.
- [129] X. Xiao and Y. Tao. Anatomy: Simple and effective privacy preservation. In *VLDB*, 2006.
- [130] X. Xiao and Y. Tao. Personalized privacy preservation. In *SIGMOD*, 2006.

- [131] X. Xiao and Y. Tao. *m*-invariance: Towards privacy preserving re-publication of dynamic datasets. In *SIGMOD*, 2007.
- [132] X. Yang and C. Li. Secure xml publishing without information leakage in the presence of data inference. In *VLDB*, pages 96–107, 2004.
- [133] C. Yao, X. S. Wang, and S. Jajodia. Checking for k-anonymity violation by views. In *VLDB*, pages 910–921, 2005.
- [134] C. T. Yu and M. Z. Ozsoyoglu. An algorithm for tree-query membership of a distributed query. In *IEEE COMPSAC*, 1979.
- [135] Q. Zhang, N. Koudas, D. Srivastava, and T. Yu. Aggregate query answering on anonymized tables. In *ICDE*, 2007.
- [136] S. Zhong, Z. Yang, and R. N. Wright. Privacy-enhancing k-anonymization of customer data. In *PODS*, 2005.

APPENDIX A
PROOF OF THEOREM 4.5

Let B denote the set of all Census blocks on the map. Below we recall the algorithm for probabilistically dropping blocks given a particular destination d . Without loss of generality, we ignore the dependence on d in the rest of the proof¹.

1. Set $m = n$ (i.e., number of synthetic workers commuting to destination d is equal to the number of workers in the input data commuting to destination d).
2. Choose minimum α_i using Algorithm 4 based on the original domain B .
3. Let $f(i)$ be a probability distribution over all the blocks in B .
4. Initialize B_{new} to be the set of origin blocks that appear in the input dataset.
5. For every origin block i that does not appear in the input data, add it to B_{new} with probability $f(i)$ to get the new domain.

Let n^1 and n^2 be two different input datasets that differ in one tuple, say corresponding to Bruce. In either input dataset, Bruce commutes to the same destination block d but from different origin blocks. Let the origin in the first case be a and the origin in the second case be b . There are three cases: (i) a is an origin for d for some tuple in n^2 and b is an origin for d for some tuple in n^1 ($a \in n^2 \wedge b \in n^1$), (ii) $a \in n^2 \wedge b \notin n^1$ (the case $a \notin n^2 \wedge b \in n^1$ is the same), and (iii) $a \notin n^2 \wedge b \notin n^1$.

¹In collaboration with Daniel Kifer.

Case 1: $a \in n^2 \wedge b \in n^1$

We'll use the random variable D to denote the set blocks that were probabilistically added. In this case, $P(D|n^1) = P(D|n^2)$ so

$$\frac{P(m|n^1, \alpha)}{P(m|n^2, \alpha)} = \frac{\sum_D P(m|n^1, D, \alpha)P(D|n^1)}{\sum_D P(m|n^2, d, \alpha)P(D|n^2)}$$

Now, this is a weighted sum (with weights summing to one) of the terms

$$\begin{aligned} \frac{P(m|n^1, D, \alpha)P(D|n^1)}{P(m|n^2, D, \alpha)P(D|n^2)} &= \frac{P(m|n^1, D, \alpha)}{P(m|n^2, D, \alpha)} \\ &= \frac{n_b^1}{n_a^1 - 1} \frac{m_a + n_a^1 - 1 + \alpha_a}{m_b + n_b^1 + \alpha_b} \\ &= \frac{P(m|n^1, B, \alpha)}{P(m|n^2, B, \alpha)} < e^\epsilon \end{aligned}$$

We let the disclosure set for our data be the events $\bigcup_D \bigcup_m \{D \wedge m_i > (e^\epsilon - 1)(n_i + \alpha_i)\}$; i.e., each event corresponds to a chosen domain D and an unrepresentative output with $m_i > (e^\epsilon - 1)(n_i + \alpha_i)$. The reference 0-sample is independent of the domain size. Hence, if it is less than $\delta(e^\epsilon - 2)/(2|B|e^\epsilon)$ then for all fixed domain sizes $\leq |B|$, the disclosure set has probability $< \delta$. Thus the disclosure probability is

$$\begin{aligned} &P\left(\bigcup_D \bigcup_m \{D \wedge m_i > (e^\epsilon - 1)(n_i + \alpha_i)\}\right) \\ &= \sum_D P\left(\bigcup_m \{D \wedge m_i > (e^\epsilon - 1)(n_i + \alpha_i)\}\right) \\ &= \sum_D P\left(\bigcup_m \{D \wedge m_i > (e^\epsilon - 1)(n_i + \alpha_i)\} | D\right) P(D) \\ &\leq \delta P(D) \leq \delta \end{aligned}$$

After removing the “bad” samples m , the maximum value for the ratio of the probabilities $P(m|n^1, \alpha)$ and $P(m|n^2, \alpha)$ is e^ϵ . Taking logs and absolute values, we find that we have with probability at least $1 - \delta$ the maximum information leakage is ϵ .

Case 2: $a \in n^2 \wedge b \notin n^1$

So $a \in n^1, n^2$ and $b \in n^2$. There are two subcases depending on whether or not $b \in m$ (i.e., b appears in the output).

Subcase 2a, $b \in m$:

In this case, b is a block that is certainly added to the domain in the case of n^1 .

$$\frac{P(m|n^1, \alpha)}{P(m|n^2, \alpha)} = \frac{\sum_D P(m|n^1, D \cup b, \alpha) P(D \cup b|n^1)}{\sum_D P(m|n^2, D, \alpha) P(D|n^2)}$$

where the summations are over all D not containing b . (Note that for such D , $P(D \cup b|n^1) = f(b)P(D|n^2)$). Again, we have a weighted average of terms (with weights summing to 1)

$$\begin{aligned} \frac{P(m|n^1, D \cup b, \alpha) P(D \cup b|n^1)}{P(m|n^2, D, \alpha) P(D|n^2)} &= \frac{P(m|n^1, D \cup b, \alpha) f(b)}{P(m|n^2, D, \alpha)} \\ &= \frac{P(m|n^1, B, \alpha) f(b)}{P(m|n^2, B, \alpha)} \end{aligned}$$

(because $n^1 \cup D \cup b$ has the same domain as $n^2 \cup D$). As in case 1, the bad outputs m have probability less than δ , and the ratio here is $\leq e^\epsilon f(b)$. To compute the inverse of the ratio, we have $\frac{P(m|n^2, \alpha)}{P(m|n^1, \alpha)}$ is a weighted sum of terms $\frac{P(m|n^2, B, \alpha)}{P(m|n^1, B, \alpha) f(b)}$. Hence, the ratio is less than $e^\epsilon / f(b)$. Taking logs we see that we need to add $\ln(1/f(b))$ to our ϵ .

Subcase 2b, $b \notin m$

In this case $n_b^2 = 1, n_b^1 = 0$.

$$\frac{P(m|n^1, \alpha)}{P(m|n^2, \alpha)} = \frac{\sum_D [P(m|n^1, D, \alpha) P(D|n^1) + P(m|n^1, D \cup b, \alpha) P(D \cup b|n^1)]}{\sum_D P(m|n^2, D, \alpha) P(D|n^2)}$$

where the summation is over D not containing b . This is a weighted average of

terms (note $P(D \cup b|n^1) = f(b)P(D|n^2)$ and $P(D|n^1) = (1 - f(b))P(D|n^2)$)

$$\begin{aligned}
& \frac{P(m|n^1, D, \alpha)P(D|n^1) + P(m|n^1, D \cup b, \alpha)P(D \cup b|n^1)}{P(m|n^2, D, \alpha)P(D|n^2)} \\
&= \frac{P(m|n^1, D, \alpha)(1 - f(b)) + P(m|n^1, D \cup b, \alpha)f(b)}{P(m|n^2, D, \alpha)} \\
&= \frac{P(m|n^1, D, \alpha)(1 - f(b))}{P(m|n^2, D, \alpha)} + \frac{P(m|n^1, B, \alpha)f(b)}{P(m|n^2, B, \alpha)} \\
&= \frac{P(m|n^1, D \cup b, \alpha) \frac{P(m|n^1, D, \alpha)}{P(m|n^1, D \cup b, \alpha)}(1 - f(b))}{P(m|n^2, D, \alpha)} + \frac{P(m|n^1, B, \alpha)f(b)}{P(m|n^2, B, \alpha)} \\
&= \frac{P(m|n^1, B, \alpha)}{P(m|n^2, B, \alpha)} \left(f(b) + (1 - f(b)) \frac{P(m|n^1, D, \alpha)}{P(m|n^1, D \cup b, \alpha)} \right) \\
&= \frac{\frac{m!}{\prod_i m_i!} \frac{\Gamma(n^1 + \alpha)}{\Gamma(n_a^1 + \alpha_a) \prod_{i \neq a, b} \Gamma(n_i^1 + \alpha_i)} \frac{\Gamma(n_a^1 + m_a + \alpha_a) \prod_{i \neq a, b} \Gamma(n_i^1 + m_i + \alpha_i)}{\Gamma(n^1 + m + \alpha)}}{\frac{m!}{\prod_i m_i!} \frac{\Gamma(n^1 + \alpha + \alpha_b)}{\Gamma(n_a^1 + \alpha_a) \Gamma(\alpha_b) \prod_{i \neq a, b} \Gamma(n_i^1 + \alpha_i)} \frac{\Gamma(n_a^1 + m_a + \alpha_a) \Gamma(\alpha_b) \prod_{i \neq a, b} \Gamma(n_i^1 + m_i + \alpha_i)}{\Gamma(n^1 + m + \alpha + \alpha_b)}}} C(1 - f(b)) \\
&+ Cf(b) \\
&\text{where } C = \frac{P(m|n^1, B, \alpha)}{P(m|n^2, B, \alpha)}
\end{aligned}$$

$$\begin{aligned}
&= C \left(f(b) + (1 - f(b)) \frac{\Gamma(n^1 + \alpha)}{\Gamma(n^1 + \alpha + \alpha_b)} \cdot \frac{\Gamma(n^1 + m + \alpha + \alpha_b)}{n^1 + m + \alpha} \right) \\
&\leq C \left(f(b) + (1 - f(b)) \frac{\Gamma(n^1 + \alpha)}{\Gamma(n^1 + \alpha + \lceil \alpha_b \rceil)} \cdot \frac{\Gamma(n^1 + m + \alpha + \lceil \alpha_b \rceil)}{n^1 + m + \alpha} \right) \\
&= C \left(f(b) + (1 - f(b)) \frac{(n + m + \alpha) \times \dots \times (n + m + \alpha + \lceil \alpha_b \rceil - 1)}{(n + \alpha) \times \dots \times (n + \alpha + \lceil \alpha_b \rceil - 1)} \right) \\
&\leq \frac{P(m|n^1, B, \alpha)}{P(m|n^2, B, \alpha)} \left(f(b) + (1 - f(b)) \left(1 + \frac{m}{n + \alpha} \right)^{\lceil \alpha_b \rceil} \right) \\
&\leq \frac{P(m|n^1, B, \alpha)}{P(m|n^2, B, \alpha)} \left(2^{\lceil \alpha_b \rceil} \right)
\end{aligned}$$

Taking the logarithm, we see that we have to add $\ln(2^{\lceil \max \alpha_b \rceil})$ to our ϵ .

The high confidence result holds as usual. The inverse of the ration, namely,

$\frac{P(m|n^2, \alpha)}{P(m|n^1, \alpha)}$, is a weighted sum of terms

$$\frac{C^{-1}}{f(b) + (1 - f(b)) \frac{\Gamma(n^1 + \alpha)}{\Gamma(n^1 + \alpha + \alpha_b)} \cdot \frac{\Gamma(n^1 + m + \alpha + \alpha_b)}{n^1 + m + \alpha}}$$

The above fraction is less than e^ϵ . Therefore, taking into account Cases 1 and 2, we have to add $\max_b [\ln(1/f(b)), \ln(2^{\lceil \max \alpha_b \rceil})]$ to our ϵ .

Case 3: $a \notin n^2 \wedge b \notin n^1$

Here we have 3 subcases: (a) $a, b \in m$, (b) $a \in m, b \notin m$ ($a \notin m, b \in m$ is the same), and (c) $a \notin m, b \notin m$.

Subcase 3a: $a, b \in m$

$$\frac{P(m|n^1, \alpha)}{P(m|n^2, \alpha)} = \frac{\sum_D P(m|n^1, D \cup b, \alpha)P(D \cup b|n^1)}{\sum_D P(m|n^2, D \cup a, \alpha)P(D \cup a|n^2)}$$

where the summations are over all D not containing a and b . (Note that for such D , $f(a)P(D \cup b|n^1) = f(b)P(D \cup a|n^2)$). Again, we have a weighted average of terms (with weights summing to 1)

$$\begin{aligned} \frac{P(m|n^1, D \cup b, \alpha)P(D \cup b|n^1)}{P(m|n^2, D \cup a, \alpha)P(D \cup a|n^2)} &= \frac{P(m|n^1, D \cup b, \alpha)f(b)}{P(m|n^2, D \cup a, \alpha)f(a)} \\ &= \frac{P(m|n^1, B, \alpha)f(b)}{P(m|n^2, B, \alpha)f(a)} \end{aligned}$$

(because $n^1 \cup D \cup b$ has the same domain as $n^2 \cup D \cup a$). So for this we have to add to our ϵ the value $\max_{a,b} \ln(f(b)/f(a))$. The bad m_i still have $< \delta$ chance of occurring, so taking into account Cases 1 and 2, adding $\max_b [\ln(1/f(b)), \ln(2^{\lceil \max \alpha_b \rceil})]$ to our ϵ gives a worst case bound on disclosure.

Subcase 3b: $a \in m, b \notin m$

In this case we have

$$\frac{P(m|n^1, \alpha)}{P(m|n^2, \alpha)} = \frac{\sum_D [P(m|n^1, D, \alpha)P(D|n^1) + P(m|n^1, D \cup b, \alpha)P(D \cup b|n^1)]}{\sum_D P(m|n^2, D \cup a, \alpha)P(D \cup a|n^2)}$$

where the summations are over the D that don't contain a and b . Noting that $f(a)P(D|n^1) = (1 - f(b))P(D \cup a|n^2)$ and $f(a)P(D \cup b|n^1) = f(b)P(D \cup a|n^2)$, we see that we have a weighted average (with weights summing to 1) of the terms:

$$\begin{aligned}
& \frac{P(m|n^1, D, \alpha)P(D|n^1) + P(m|n^1, D \cup b, \alpha)P(D \cup b|n^1)}{P(m|n^2, D \cup a, \alpha)P(D \cup a|n^2)} \\
= & \frac{P(m|n^1, D, \alpha)(1 - f(b)) + P(m|n^1, D \cup b, \alpha)f(b)}{P(m|n^2, D \cup a, \alpha)f(a)} \\
= & \frac{P(m|n^1, D, \alpha)(1 - f(b))}{P(m|n^2, D \cup a, \alpha)f(a)} + \frac{P(m|n^1, B, \alpha)f(b)}{P(m|n^2, B, \alpha)f(a)} \\
= & \frac{P(m|n^1, D \cup b) \frac{P(m|n^1, D, \alpha)}{P(m|n^1, D \cup b)}(1 - f(b))}{P(m|n^2, D \cup a, \alpha)f(a)} + \frac{P(m|n^1, B, \alpha)f(b)}{P(m|n^2, B, \alpha)} \\
= & \frac{P(m|n^1, B, \alpha)}{P(m|n^2, B, \alpha)f(a)} \left(f(b) + (1 - f(b)) \frac{P(m|n^1, D, \alpha)}{P(m|n^1, D \cup b)} \right) \\
= & \frac{\frac{m!}{\prod_i m_i!} \frac{\Gamma(n^1 + \alpha)}{\Gamma(n_a^1 + \alpha_a) \prod_{i \neq a, b} \Gamma(n_i^1 + \alpha_i)} \frac{\Gamma(n_a^1 + m_a + \alpha_a) \prod_{i \neq a, b} \Gamma(n_i^1 + m_i + \alpha_i)}{\Gamma(n^1 + m + \alpha)}}{\frac{m!}{\prod_i m_i!} \frac{\Gamma(n^1 + \alpha + \alpha_b)}{\Gamma(n_a^1 + \alpha_a) \Gamma(\alpha_b) \prod_{i \neq a, b} \Gamma(n_i^1 + \alpha_i)} \frac{\Gamma(n_a^1 + m_a + \alpha_a) \Gamma(\alpha_b) \prod_{i \neq a, b} \Gamma(n_i^1 + m_i + \alpha_i)}{\Gamma(n^1 + m + \alpha + \alpha_b)}}} C \frac{1 - f(b)}{f(a)} \\
& + C \frac{f(b)}{f(a)}
\end{aligned}$$

$$\text{where } C = \frac{P(m|n^1, B, \alpha)}{P(m|n^2, B, \alpha)}$$

$$\begin{aligned}
& = \frac{C}{f(a)} \left(f(b) + (1 - f(b)) \frac{\Gamma(n^1 + \alpha)}{\Gamma(n^1 + \alpha + \alpha_b)} \cdot \frac{\Gamma(n^1 + m + \alpha + \alpha_b)}{n^1 + m + \alpha} \right) \\
& \leq \frac{C}{f(a)} \left(f(b) + (1 - f(b)) \frac{\Gamma(n^1 + \alpha)}{\Gamma(n^1 + \alpha + \lceil \alpha_b \rceil)} \cdot \frac{\Gamma(n^1 + m + \alpha + \lceil \alpha_b \rceil)}{n^1 + m + \alpha} \right) \\
& = \frac{C}{f(a)} \left(f(b) + (1 - f(b)) \frac{(n + m + \alpha) \times \cdots \times (n + m + \alpha + \lceil \alpha_b \rceil - 1)}{(n + \alpha) \times \cdots \times (n + \alpha + \lceil \alpha_b \rceil - 1)} \right) \\
& \leq \frac{P(m|n^1, B, \alpha)}{P(m|n^2, B, \alpha)f(a)} \left(f(b) + (1 - f(b)) \left(1 + \frac{m}{n + \alpha} \right)^{\lceil \alpha_b \rceil} \right) \\
& \leq \frac{P(m|n^1, B, \alpha)}{P(m|n^2, B, \alpha)} (2^{\lceil \alpha_b \rceil}) / f(a)
\end{aligned}$$

So we now have to add $\max_b \ln(1/f(b)) + \ln(2^{\lceil \max \alpha_b \rceil})$ to our ϵ . To compute the inverse fraction, as in Case 2b, we get the upper bound of $f(a)/C$ which is less than e^ϵ . So after all cases examined so far, adding $\max_b \ln(1/f(b)) + \ln(2^{\lceil \max \alpha_b \rceil})$

to our ϵ is sufficient to bound the worst case disclosure.

Subcase 3c: $a \notin m, b \notin m$

In this case we have

$$\frac{P(m|n^1, \alpha)}{P(m|n^2, \alpha)} = \frac{\sum_D [P(m|n^1, D, \alpha)P(D|n^1) + P(m|n^1, D \cup b, \alpha)P(D \cup b|n^1)]}{\sum_D [P(m|n^2, D \cup a, \alpha)P(D \cup a|n^2) + P(m|n^2, D, \alpha)P(D|n^2)]}$$

where the summations are over the D that don't contain a and b . If we remove the second term in the denominator, we increase the fraction and we get the same computations as in Case 2b. If we remove the first term on the numerator, we get the same computations as in the part of Case 2b where the inverse is considered.

Thus in all cases, the bad sets have probability less than δ if we add $\max_b \ln(1/f(b)) + \ln(2^{\lceil \max \alpha_b \rceil})$ to our ϵ .