Trail Re-identification and Unlinkability in Distributed Databases

Bradley Malin

CMU-ISRI-06-105

May 2006

Institute for Software Research, International
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Latanya A. Sweeney, Chair
Kathleen M. Carley
Christos Faloutsos
Christopher Clifton, Purdue University

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was supported in part by the National Science Foundation under Integrative Graduate Education and Research Training (IGERT) grant number 9972762 in Computational Analysis of Social and Organization Systems (CASOS) and in part by the Laboratory for International Data Privacy at Carnegie Mellon University.

The opinions expressed in this research are solely those of the author and do not necessarily reflect those of the United States government, nor its funding agencies.

Keywords: privacy, anonymity, record linkage, distributed systems, databases, data mining, secure multiparty computation, genetic variation, medical privacy, Internet privacy, public policy

Abstract

The term "re-identification" refers to the correct relation of seemingly anonymous data to explicit identifying information, such as the name or address of people who are the subjects of the data. Historically, re-identification methods were evaluated with respect to a single provider's disclosed databases. Imagine that a data holder discloses two databases: one consists of unidentified data, such as DNA sequences and the other contains identifiable data, such as personal names. The databases appear unrelated and, as a result, existing data protection policies certify sufficient protection from re-identification. However, when multiple locations make such releases available, an individual's data can be tracked across locations, resulting in a location-visit footprint, or a trail. Unique trails can be leveraged for re-identification. In this dissertation, traditional notions of privacy are extended to account for trail re-identification. The goals of this dissertation are: 1) to prove data protection policies can and do fail when they rely on ad-hoc strategies and 2) to demonstrate policies can be strengthened with formal computational models for privacy technology.

In this work, trails are studied in two principle parts. First, we concentrate on the trail re-identification problem and develop several learning algorithms for discovering re-identifications. The algorithms are evaluated on populations derived from real world databases, including hospital visits derived from medical databases and weblogs derived from Internet databases. It is demonstrated that susceptibility to trail re-identification is neither trivial nor the result of bizarre isolated occurrences. Experimental evidence with real world populations confirms that significant quantities of populations are at trail re-identification risk.

Second, we propose a protocol by which data holders can collaborate to provably prevent trail re-identification. To do so, we introduce a formal model of privacy called k-unlinkability, and several configurable algorithms to render protected trails. To satisfy real world policy constraints, we present a novel secure multiparty computation protocol that embeds the protection procedure. Using real world datasets, it is demonstrated that significant quantities of data can be disclosed with provable privacy guarantees.

Acknowledgements

Family and friends, I thank you for your patience, encouragement, and generosity. Foremost, without the support of Sara Zimmerman Malin this dissertation would remain uncompleted. During the course of my research and writing, she played the role of a sounding board, an encourager, a stabilizer, and a put-that-down-take-a-break-and-drink-a-pint realist. She provided infinite common sense insight for complex and insurmountable problems.

I am fortunate to have worked with an excellent dissertation committee in Kathleen Carley, Chris Clifton, Christos Faloutsos, and Latanya Sweeney. They donated time, expertise, and funding to ideas that were clear to me, but not always clear to them. I am deeply indebted to Latanya Sweeney for introducing me to the field of computational disclosure control. Years ago she showed me that privacy is more than ethics and law and that computer science is more than theory, engineering, and experimentation. For many years she has been my advisor, collaborator, and greatest champion. Her fostering has immeasurably improved my abilities as a scientist, researcher, and educator. For that and more, I am humbled.

The collegiality and interdisciplinary environment at Carnegie Mellon University made for a lively and unique setting from which this research greatly benefitted. I owe much to the Department of Biological Sciences and the Department of Engineering and Public Policy for an excellent undergraduate education. During that time, I was lucky to have met John Woolford, who provided me with the opportunity to immerse myself in research in my first year at Carnegie Mellon. He dedicated his time to teach me that research is more than just a series of experiments. I am especially thankful to have been advised by Indira Nair, who encouraged me to go beyond science and illustrate why philosophy, ethics, and technology are not independent.

The research in this dissertation was conducted across several schools at the university. I thank the Heinz School of Public Policy and Management for the opportunity to begin my doctorate and study technology policy issues in a supportive environment. It served as an excellent locale for the initiation and development of this research. I am further grateful to the School of Computer Science; first to the Center for Automated Learning and Discovery

(now the Machine Learning Department) where, as a master's student, I formalized my re-identification models, and second to the Institute for Software Research International where, as a doctoral student in Computation, Organizations and Society (COS) program, I was encouraged to solve problems and not just make them.

Many others have helped shape my views, research, and took the time to comment on sections of this dissertation during its evolution. Of notable merit, the members and close affiliates of the Data Privacy Laboratory at Carnegie Mellon University have been invaluable in their assistance. With recognition that I will undoubtly neglect to name some people, I thank Alessandro Acquisti, Edoardo Airlodi (for the coffee talk), Sylvia Barrett, Leonard Berman (for showing me its ok to think), Michael Benisch (for making me think deeper), Claus Boyens, the Bruisers, Josh Burnett, Brian and Kaisa Carini, the CASOS crew, Catherine Copetas, Lorrie Cranor, Dante, George Davis, Miguel DeLeon, Monika DeReno, Samuel Edoho-Eket, Serge Egelmen, Amy and Joe Elliott, David Farber, Stephen Fienberg, Gobi, William "Spike" Gronim, Ralph Gross, Thomas Harris, Susan Henry (for recommending I meet with a new faculty member at Carnegie Mellon named Latanya Sweeney), Ralf Hölzer, Jake the wonder-pup, Jeremy Johnston, Ponnorungam Kumaraguru, Yiheng Li, Tom Lin and Demiurge Studios, Sherice Livingston (for putting up with me), Kishore Madhava, the Malin family (especially to my parents), Christy McGuire, Robert Murphy, Elaine Newton, the Originals, Rema Padman, the sisters Panza, Michael Shamos, Mary Shaw, Vicenc Torra, Benjamin Vernot, Marshall Warfield, Victor Weedn, Audrey and Ryan (don't call me Philip) Wilson, William Winkler, and the Zimmerman family for their time, feedback, and assistance. I recognize the National Science Foundation for funding parts of this research.

Finally, I thank Kiva Han Coffeehouse and Coffee Tree Roasters for all of the yummy drinks, food, and white noise that helped me to sit down and write. So be it.

Contents

List of Figures

List of Tables

Chapter 0

Forward: Contributions and Organization

The goal of this dissertation is to investigate a specific data privacy problem that manifests in distributed systems. In the past, data holders protected the identity of subjects by disclosing collections of sensitive data that were stripped of explicit identifying information, such as the subject's name, phone number, or Social Security Number. However, in this research, I show that the location-visit patterns of an individual, or the trail, leads to the re-identification of seemingly anonymous data, such as the DNA sequence of a hospital patient. The trails model of distributed re-identification is a novel extension to the traditional viewpoint of re-identification of a single location's releases.

To achieve re-identification, I introduce several formal models and algorithms to link trails. Trails manifest in many data-centric environments and so I present evidence with real world healthcare and Internet data that trails lead to a large number of re-identifications in various populations. After developing a re-identification framework, I concentrate on how to share data while provably preventing trails from re-identification. To do so, I develop a novel formal privacy protection model called k-unlinkability to guarantee a trail can not be re-identified to less than k identities in a particular population. To transform trails to satisfy the k-unlinkability model I introduce several algorithms to work in both unsecure and secure scenarios. Finally, I illustrate with experimental evidence that shared data can satisfy k-unlinkability.

Contributions

This dissertation touches upon issues in many fields of research and application. As a result, there are a number of audiences that can benefit from the work herein, including:

Computer Scientists. The field of computer science is evolving to incorporate complex social, organizational, and political environments in which computers are integrated. One place in particular where the line blurs between computers and social systems is in the study of data privacy. The protection of personal privacy in computer systems is a growing challenge, and solutions based on policy regulations alone are no longer sufficient to protect data. In the face of increased erosion due to technological advancement, data privacy protection must incorporate technology and formal computational models.

As a result, the issue of privacy is increasingly studied within a number of computer science subdisciplines including databases, data mining, cryptography, ubiquitous and pervasive computing, and human-computer interaction. The research presented in this dissertation formalizes a specific data privacy challenge. It introduces formal computational models of data linkage in distributed data collection environments. Moreover, after characterizing and exploiting the data privacy vulnerability of trail re-identification, this work proposes a novel computational solution to prevent trail re-identification that integrates both cryptography and computational disclosure control theory.

Policy Designers. From a policy standpoint, this dissertation is situated to communicate several lessons. First, the re-identification models presented in this work are grounded in real world environments. The evaluation of the developed algorithms and protocols demonstrate the fallibility of current policy and legal oversight in complex social and technological settings that are amenable to automated learning methods. The fact that re-identification of significant portions of populations can be achieved provides a clear example regarding why policy can not be naïvely enacted. The technology policy design process must incorporate sound scientific analysis, models, and use foresight. Retroactive policies for technology are extremely costly, and in this research, do not address data privacy issues for databases already disclosed. Second, the data protection methods proposed in this dissertation demonstrate how policy can be strengthened by technological means. The protection methods are designed to complement, and not substitute, existing policy regulations.

Administrators. For administrators, this research demonstrates how their data collections can be evaluated for privacy vulnerabilities. More importantly, this dissertation provides a means by which independent organizations can legally collaborate to prevent unintended privacy compromises in data that is shared for research purposes. This work draws on examples that illustrate medical and Internet privacy issues.

In addition, the experimental validation of the data protection methods debunks the traditional privacy versus utility myth. In the past, it was assumed that privacy

and utility were polar opposites. Many administrators believed that, in order to ensure data privacy, they could not disclose their data collections. To preserve privacy they must limit the utility of the data. In contrast, many administrators believed that to ensure data utility, they had no choice but to disclose all of the data. Thus, they must give up on protecting the privacy of the individuals to whom their data corresponded. However, the models presented in this work permit, and the experiments prove, that the myth is just a myth - privacy and utility can be simultaneously achieved.

Overview

This research is presented in ten chapters, organized into two principle parts: 1) trail re-identification and 2) trail protection with provable guarantees.

Chapter 1 introduces the reader to data privacy in modern society and presents several real world examples of trails.

Chapter 2 defines a formal model, as well as an inefficient solution, for the trail linkage problem.

Chapter 3 presents how trails are composed from distributed databases.

Chapter 4 develops several efficient algorithms to achieve trail linkage.

Chapter 5 reports the results from re-identification experiments in real world and simulated datasets.

Chapter 6 introduces a novel formal model of data privacy called k-unlinkability.

Chapter 7 develops several algorithms to render trails k-unlinkable when locations can openly collaborate.

Chapter 8 extends the unlinking algorithms to operate in a secure multiparty environment to address existing policy regulations.

Chapter 9 presents experimental evaluation of the unlinking algorithms with several data utility functions.

Chapter 10 summarizes the contributions, limitations, and possible extensions to this research.

In addition, there are four appendices that supplement this research and help to frame this dissertation in a broader context.

Appendix A extends the trail linkage model to re-identify multiple pieces of data to the same individual.

Appendix B demonstrates how trails relate to other re-identification vulnerabilities, in the context of deployed protection systems for personal genomic data.

Appendix C provides details of the secure multiparty computation protocol.

Appendix D provides proofs regarding the mathematical bounds of the protocol.

This forward concludes with a disclaimer for the reader. This dissertation does not solve all of the world's privacy ills. This is not my intention. Rather, this research demonstrates how a specific aspect of personal privacy can be formally modelled in distributed environments. It serves as an example of how policy can be designed with an understanding of the environment in which it is meant to regulate. Finally, it demonstrates how social and computational problems can be studied in harmony.

Chapter 1

Introduction

As people hustle and bustle through daily life they leave behind fragments of personal information in various databases [137]. Individuals are not always endowed with the ability to exert control over whether or not their data is collected and, in some instances, they may not even be aware that they are shedding any information. For instance, images of an individual's automobile are recorded on different highway video cameras [140]; the IP address of a personal computer is logged at multiple websites [112, 118]; and, as is discussed in this dissertation, a patient's DNA can be sequenced and recorded in numerous hospital databases. In our data-driven society, there is an ever-increasing demand for the incorporation of new technologies to gather data on people for a variety of worthwhile, as well as nefarious, endeavors. In addition, database collections have become commodities that can be shared, licensed, or sold for profit across a range of communities. This is possible, in part, because both data holders and subjects consider these fragments of data innocuous. People often harbor the belief that their pieces of data are isolated and no one would systematically relate the fragments. These beliefs are fortified by current policy regulations, legal statutes, and certification procedures designed to uphold personal privacy in shared databases.

However, today's policies often lack proper representation, or characterization, of complex technologies and their interactions within social environments. Rarely are policies designed to account for heterogenous types of data strewn across today's distributed and decentralized environments. The research within this dissertation demonstrates that not only are current beliefs of data privacy protection false, but relatively simple learning algorithms can automate the process of linking, or re-identifying, identities back to seemingly anonymous data. Yet, remedies are possible.

The re-identification techniques presented in this research are a derivation of general learning methods that can be used for such activities as data mining and surveillance,

but simultaneously they reveal and amplify serious privacy risks inherent in current data sharing practices. The ability not only to track people, but to re-identify them in the process, poses risks to both the individuals and the data holders originally provided with the information.

For example, DNA sequences are increasingly becoming a part of electronic patient medical records [5, 124]. With the potential to support significant advances in healthcare, many groups that harbor collections of person-specific genomic information want to share information for various endeavors. Though the genome of an individual is understood to be as, or more, personal than a fingerprint [81], a database of mere DNA entries, with no additional explicit demographic information or identifiers included, appears sufficiently anonymous. However, patients leave information behind at multiple institutions and the collections are autonomously controlled. As a result, the location-visit pattern, or the trail, of an individual's DNA can be extracted from the shared databases. A trail provides a representation of which locations an individual visited by characterizing when the individual's data was observed or not observed each location. Alone, trails are not necessarily re-identifiable, but publicly available information, such as hospital discharge databases, are available and can reveal the trail of an identified individual. Common features in an individual's discharge and DNA trails can lead to re-identification and compromise of patient anonymity.

As a second example, consider the online consumer who leaves the IP address of his computer in access logs at each website he visits. At some websites, the consumer may also provide explicit identifying information; for example, his name and address are provided to complete a purchase. Each website can independently share the log containing the IP addresses of those who visited their site. As e-businesses, these websites can also share explicitly identified data such as customer lists, which typically include the name and address (e.g., residential, e-mail, etc.) of those who made purchases. By examining the trails of which IP addresses appeared at which locations and matching those visit patterns to which customers appeared in the identified customer lists, IP addresses can be related to names and addresses. These re-identifications can then be used to identify visits to websites where the consumer did not make purchases.

In both of the aforementioned examples, the goal is to share person-specific data while concealing the identities of the individuals from whom the data was derived. To achieve this goal, various solutions have been proposed in a number of communities. Many solutions have been developed with respect to healthcare, where data protection is a serious concern and federal policies have been enacted. Published solutions to protect privacy include the removal of explicit identifiers [17, 63, 79, 160], the encryption [30, 48] or hashing [30, 48] of identifiers , and the disclosure of data through a surrogate or third

party [59].[1] However, the simple removal of personally-identifying features is not sufficient to guarantee protection from re-identification. Rather, the use of ad-hoc approaches that neglect to account for the system in which data is disclosed or organizational practices only instill a false sense of privacy. In contrast, in this dissertation we develop a formal protection model for trails, so that data can continue to be shared with provable guarantees of re-identifiability. The protection model we present enable data sharing locations to collaborate in a manner that renders trails from being re-identified.

1.1 Re-identification and Data Privacy

Historically, the concept of re-identification, and the development of methods for such a task, was affiliated with the release of data from a single institution or collection [31, 46, 155]. In the past, it was generally believed that a data collection in which each piece of data related to a person could be shared somewhat freely, provided none of the features of the data included explicit identifiers, such as name, address, or Social Security number. However, an increasing number of data detective-like investigations revealed that collections of "de-identified" data, derived from ad-hoc protection models, can often be linked to other collections that do include explicit identifiers to uniquely, and correctly, re-identify disclosed information by personal name [11, 58, 90, 135, 152]. Fields appearing in both de-identified and identified tables can link the two, thereby relating names to the subjects of the de-identified data. For example, Sweeney's analysis of the combination of values for the fields $\{$*date of birth*, *gender*, *5-digit zip code*$\}$, which, until recently, commonly appeared in both de-identified databases and publicly available identified databases, such as voter registration lists, uniquely represented approximately 87% of the U.S. population [135].

One of the main contributors to the fallibility of ad-hoc privacy protection methods stems from the complexity that arises in today's decentralized systems. Policies designed to protect the anonymity of data released from a single institution can become riddled with holes as the number of institutions that release data grows. This is a phenomenon akin to Tragedy of the Commons: no particular institution violates existing laws or policy, but in combination all institutions contribute to the erosion of protections.

In this dissertation, we make an extension to traditional re-identification and demonstrate how re-identification can occur via the pattern of locations people visit, or trails. The main premise of the trail linkage model is based upon the observation that entities visit different sets of locations where they can, and do, leave behind similar pieces of de-identified information. The de-identified data can consist of only one or very few fields.

[1] An evaluation of existing protection solutions is detailed in Appendix B.

Each location visited collects and, subsequently, shares de-identified data on people who visited their location. In addition, locations also collect and share, in separate releases devoid of of de-identified data, explicitly identified data (i.e., name, residential address, etc.), thereby naming some people. Individually, a single location's releases appear unrelatable, and thus identity and sensitive information appear unlinkable. However, when multiple locations share their respective data, this allows for trails, a characterization of the locations that an individual visited, to be constructed. Similar patterns in the trails of de-identified and identified data can then be used for linkage purposes.

In the real world, trails manifest in a number of environments with various regulations and policies. Consider several scenarios that we will return to during the course of this dissertation.

1.1.1 Trail Scenario 1: Genomic Data Privacy

Modern medicine is currently in the midst of a genomics revolution that promises significant opportunities for healthcare advancement [6, 40]. At the same time, the increased incorporation of genomic data into medical records, and the subsequent sharing of such data, raises complex patient privacy issues. In current healthcare environments, patients visit and leave behind data at multiple data collecting locations, such as hospitals. When genomic data is shared from the place of collection, it may or may not be the case that a contractual agreement that manifests in healthcare in the form of a "data use agreement" (DUA), is required. This requirement is dependent on whether or not the data is provided under "research purposes" as specified by the Privacy Rule of the Health Insurance Portability and Accountability Act (HIPAA) [32].

For example, collections of hospital discharge data are not subject to HIPAA protections because the governing body over this type of information is not considered a "covered entity". Moreover, though an individual's DNA sample can be unique among a population [84], the HIPAA Privacy Rule does not explicitly classify DNA-based data (e.g., sequence data, expression microarrays) as an identifying attribute of a patient. The main reason for this is there exists no explicit relationship between genomic data and personal information, such as name. Without a primary key, associating autonomous DNA information to named persons seems impossible. After all, there does not exist a master registry against which the DNA information could be directly compared to reveal the associated person's identity. As such, one can make the argument that genomic data should be released under the Safe Harbor provision of the HIPAA Privacy Rule. Thus, with respect to healthcare environments, each hospital can sever genomic data from clinical data and, subsequently, release genomic data in order to enable such endeavors as basic research without being in violation of the HIPAA Privacy Rule [76, 153].

When considering genomic data, as with each type of disclosed collection of data, we need to clarify what exactly is the data sharing environment. For instance, when a dataset is made publicly available it is not subject to Institutional Review Board (IRB) review, nor are DUAs required. We have already seen the advent of a handful of public use DNA datasets, such as the National Center for Biotechnology's PopSet database. These types of collections circumvent the HIPAA-specified controls of "attendant protections" and "IRB oversight" because the data is already on publicly available websites. However, we recognize these modes of sharing might severely limit access or availability to genomic data for more complex types of research and analysis.

In contrast, if DNA data is to be 1) shared for research purposes and 2) subject to constraints of the HIPAA Privacy Rule, then a DUA is required. In addition, an IRB's approval is required if the research is federally funded. Yet, one of the exemptions to oversight an IRB will provide is if the data is believed to be anonymous. Thus, if DNA data is found to be potentially vulnerable to re-identification methods, such as those in this research, then the DUA and IRB protections may be forced into revision and strengthened.

Nonetheless, even when DUA and IRB approval are required, administrators may base their decision on false beliefs about the identifiability of the data. As a consequence, there is no guarantee that the data, which has been subject to a DUA and IRB review, are protected sufficiently from re-identification methods. While it is true that re-identification may be prohibited in the DUA, as a policy it is not sufficient to prevent someone from attempting the act. Rather than discuss the limitations of privacy protection policies, it is more fruitful to develop policy infused with technology. This dissertation argues that policies are stronger when complemented by technologies that ensure controllable and enforceable protection.

1.1.2 Trail Scenario 2: Online Data Privacy

A second environment we consider is the World Wide Web, where electronic commerce has facilitated the sharing and collection of personal information to an increasing number of independently functioning e-businesses [75, 125]. Within this environment, websites collect various types of data on individuals. Following the definitions of many online privacy policies, data is coarsely categorized as *identifiable* and *non-identifiable* information. The latter is defined as information that does not explicitly reveal the identity of the individual [29]. Many policies state the IP address of an individual's computer is considered non-identifiable information. As a result, an individual has minimal control over when their computer's IP address is collected and stored in a website's access log.[2] In contrast,

[2]We recognize that an individual can obfuscate their IP address by participating in a mix-net system, such as CROWDS [114] or Onion routing [56, 36]. However, the adoption of these systems are relatively

when visiting a website an individual does have a choice regarding whether or not to share identifiable information. This type of information explicitly reveals the identity of the individual, such as name, residential address, or credit card number.

Data collectors relate identifiable to non-identifiable information for a number of legitimate purposes in accordance with their privacy practices. These purposes may include direct marketing, website personalization, and intrusion and fraud detection. To facilitate an individual's choice of whether or not to provide identifiable information, websites post their privacy policies, that specify general aspects about how an individual's data will be used, managed, and shared. If an individual believes that a website's privacy practices are in accordance with their own, they may choose to provide their identifiable data. When an individual feels otherwise, they can choose not to reveal identifying information. In the latter case, individuals do not want such websites to know what name, or other identifiable information, corresponds to the visiting IP address.

Oftentimes, websites treat collections of person-specific information akin to commodities. The collected data can be legally shared, licensed, or sold with other parties for various purposes beyond in-house uses and in accordance with prespecified online policies. For example, continuing with our example of e-commerce, customer lists are routinely provided to affiliated third parties. In the online environment, and beyond, it has been recognized that certain types of collected information about an individual are more sensitive than others. As detailed in many organizations' privacy policies, non-identifiable data will not be shared in a manner that allows for it to be related to identifiable data. To ensure this policy, many locations separate identifiable from non-identifiable data and release the two as different datasets. Online privacy policies are considered tantamount to contractual agreements, and oversight in this realm is the Federal Trade Commission (FTC). Failure of a website to adhere to its online policies is considered to be in violation of the FTC's specification of deceptive practices.

This model of privacy protection is akin to that discussed in the health environment above, and again gives off the appearance that it protects the identity of the individual. Releasing a list of IP addresses provides no more information than any other website might collect. There is no information that a data user, or adversary, can employ to re-identify the individuals of the released dataset. So, from the perspective of each data releasing website, the partitioning of non-identifiable and identifiable data appears to protect their consumers' privacy.

However, data collectors rarely communicate information about their data collections to each other. Instead they release their data collections independently. As the number of websites that collect and share data increases, the visit patterns of an individual's identifiable and non-identifiable information tends toward uniqueness and sensitive information

small in comparison to the number of Internet users [45].

can re-identified.

1.2 Unlinkability and Data Privacy

Given the wealth of knowledge that can be mined from very large databases of person-specific data, it is essential to support the dissemination of collections for innovative research and worthwhile application-based endeavors, such as for healthcare initiatives. Yet, the dissemination of personal information must be performed in a manner that upholds the claimed level of privacy for the individuals to whom the data corresponds.

To integrate privacy into shared data, this dissertation introduces a novel formal privacy protection model that guarantees the unlinkabilty of trails, or the inability to achieve re-identification. Formal models of privacy have been developed in prior research [136], however, the issue addressed in this research extends earlier work into a distributed environment. Whereas prior models investigated how to anonymize data held by a single location, a trail is dependent on knowledge distributed across a set of locations. Thus, our anonymization procedure must account for data strewn across a set of data holders.

The solution presented in this dissertation is designed to prevent trail re-identification while adhering to defined policy constraints, such as the HIPAA Privacy Rule. Computational protocols are formalized by which a set of data holders can work together in an unsecure environment or with a third party in a secure setting, such that no recipient of disclosed de-identified data can achieve re-identification beyond a configurable parameter. The protocols make use of anonymization algorithms based on probabilistic intuition to maximize specified utility functions for the disclosed data. The execution of the protocol supports disclosure data with provable guarantees of trail re-identifiability.

1.3 Outline

This dissertation examines re-identification risks related to the trails of data people leave behind and introduces a provable privacy solution in the form of trail unlinkability. In this work, trails are examined from two perspectives: 1) as a data detective and 2) as a data protector. As a data detective, we develop the trail re-identification framework, present automated methods to complete the task, and experimentally validate the vulnerability in today's population-based systems. Once the problem is clearly understood and well-defined, we switch hats and approach the problem as a data protector and develop methods to prevent trail re-identification with provable guarantees of trail unlinkability.

The research in this dissertation is organized as follows.

Chapter 1 introduced the reader to data privacy in modern society. It informally presented why trails exist in given current policies and legal statutes in several real world environments, including healthcare and the Internet.

The remainder of this thesis is organized into two primary components as follows:

Part 1 focuses on the trail re-identification problem.

Chapter 2 provides a formal model of the trail linkage problem. Trails are defined and the linkage problem is formalized from a mathematical perspective. It is shown the problem has an intuitive relationship to graph theory. An optimal, but inefficient, graph theoretic solution is presented.

Chapter 3 describes how trails are constructed from distributed databases. The chapter provides a formal model of how to build trails, as well as the specific assumptions that are adopted for this dissertation.

Chapter 4 develops several efficient algorithms for trail linkage, collectively termed REIDIT (Re-identification in Trails).

Chapter 5 presents experimental re-identification results on several real world and simulated datasets. Intuition regarding how various locations influence linkage is presented.

Part 2 shifts the focus of this thesis to the trail unlinkability problem.

Chapter 6 introduces a novel formal model of privacy called k-unlinkability. It is shown how this model compares to prior models, such as k-map and k-anonymity.

Chapter 7 presents several algorithms to render trails k-unlinkable under the assumption that locations can openly collaborate.

Chapter 8 takes into account various legal, policy, and social constraints. Open communication between locations is not always possible, and this chapter extends the unlinking algorithms to operate in a secure and encrypted environment.

Chapter 9 experimentally evaluates the trail unlinking algorithms presented with several data utility functions.

Chapter 10 summarizes the thesis and presents several directions for future research.

In addition, there are three appendices that supplement this research and help to frame this dissertation in a broader context.

Appendix A extends the trail linkage model provided in the core of this dissertation to a more general setting. We present a novel algorithm to achieve re-identification when an individual leaves behind multiple pieces of data. We evaluate susceptibility on a real world dataset.

Appendix B demonstrates how trail re-identification is affiliated with a larger group of methods for evaluating privacy enhancing technologies. The presentation is placed in the context of of real world genomic data privacy protection systems.

Appendix C presents details of the secure multiparty computation framework that supports the unlinking of trail.

Appendix D provides theoretical bounds of the protocol's solutions.

1.4 My History With This Work

This dissertation ties together work from a number of publications in various outlets. My research on trails began several years ago as part of an investigation into the protection capabilities of HIPAA and current genomic data privacy systems with Dr. Latanya Sweeney, which initially resulted in the publication [90]. The first publication on trails surfaced in 2001 [91] in the biomedical informatics community. Following the initial publication, I continued to strengthen and make the algorithms for trail re-identification more sophisticated. They were extended [86], the re-identification framework was generalized [98], and the susceptibility of populations in a number of environments was experimentally proven to be significant [87, 92, 94]. Trail re-identification, in association with other re-identification methods I, as well as others, developed (see Appendix B), became a litmus test for system vulnerability. It was clear that trail re-identification demonstrated deficiencies exist in current data protection policies. However, pointing out vulnerabilities do not prevent them from existing and therefore I began research in formal protection models that explicitly thwart trail re-identification. The first formal protection model for unlinkability to prevent trail re-identification in healthcare was published in the biomedical informatics community in 2005 [96]. The unlinkability model was generalized and a short paper on protection models was presented to the computer science community in 2006 [97]. Some of the more rigorous proofs and experimental analysis of trail unlinkability that are presented in the latter chapters of this dissertation have yet to be published.

Part I

Re-identification

Chapter 2

Trail Linkage

Trail re-identification can be roughly split into two subproblems: 1) composition and 2) linkage. First, we need to understand where a trail comes from. Simply put, how do we compose trails from disparate databases? Second, once we have two sets of trails, we need to determine how trails can be linked to re-identify seemingly anonymous data.

For clarity in presentation, we will concentrate on the second problem first. This chapter addresses the linkage question at a generalized level. We assume two sets of trails have been generated and introduce algorithms that show how re-identifications are learned from the sets. In Chapter 3, we explain how the sets of trails are composed from separate data collections.

In this chapter, after defining the trail linkage problem, we prove an optimal solution can be constructed using existing graph theoretic techniques. The solution, while optimal, is not efficient and requires on the order of O($n!$), where n is the number of trails. In contrast, in the following chapters, we develop an approximation of the optimal solution (the REIDIT-Incomplete algorithm), which can discover re-identifications, without any false re-identifications, in O(n^2) time. Furthermore, we derive an optimal solution to a special case (the REIDIT-Complete algorithm), which can be solved in O($n \log n$) time.

2.1 A Formal Model of Trail Linkage

We commence our formalization of trail re-identification with a general model for trails and linkage. We call the basic data structure a *trail matrix* (Definition 1). This matrix summarizes information for data tracked over a set of locations. Each row of the trail matrix is termed a *trail* and represents a visit pattern over a set of locations (i.e., the columns). Each value of a trail is selected from a set of two unambiguous values, 0 and 1, and an ambiguous value, *, which symbolizes either 0 or 1. The value 1 indicates an

entity appeared at a location, whereas 0 indicates an entity failed to appear at the location, and * means we are unable to discern if an appearance was made.

Definition 1 (Trail Matrix / Trail). *A* trail matrix X *is an* $m \times p$ *matrix with cell values drawn from the range* $\{0, 1, *\}$. *A* trail *is a row* $x = [a_{x,1}, \ldots a_{x,p}]$ *in* X*, where* $a_{x,i} \in \{0, 1, *\}$.

Figure 2.1 provides an example of several trail matrices. Trail matrix X alerts us that data element x_1 was at locations 1 and 3 and was not at location 2. It is unknown if x_1 visited location 4.

In this research, the rows of a trail matrix correspond to persons or entities. The columns of a trail matrix correspond to locations, such as hospitals or URLs. We concentrate on the relationship between disparate trail matrices. We study a specific type of scenario, which we call the *association relationship* (Definition 2). Informally, this relationship implies one trail matrix is the duplicate of the other trail matrix with missing, or ambiguous, values. Both trail matrices are defined over a common and closed population of people and locations.

In the trail matrices, columns are aligned, such that each particular location is in the same column for both matrices. In Figure 2.1, for instance, Location 1 is in the first column of both trail matrices X and Y. Rows, on the other hand, are not aligned, so the same person's data can be represented by different rows of the two trail matrices. In Figure 2.1, an individual's information, which is unambiguously represented by w_1, is in the fourth row of trail matrix X as x_4 and is in the third row of trail matrix Y as y_3.

Definition 2 (Association Relationship). *We say there exists an* association relationship *between two trail matrices* X *and* Y*, if there exists a trail matrix* W *with range* $\{0, 1\}$*, such that:*

1. X *is a copy of* W*, such that cell values may be replaced with *,*
2. Y *is a copy of* W*, such that cell values may be replaced with *, and*
3. *row ordering in* X *and* Y *are random permutations of row ordering in* W*.*

Let $f : W \to X$ *and* $g : W \to Y$ *be bijective functions that map row positions in* W *to their permuted positions in* X *and* Y*, respectively.*

When there exists an association relationship, we can specify when two trails are derived from the same ancestor. When two trails are derived from the same ancestor, we say the two trails make up an *association*. In this sense, we do not necessarily imply the two trails are derived from the same trail[1], but from the same underlying concept. This

[1]For instance, there may exist multiple copies of a trail.

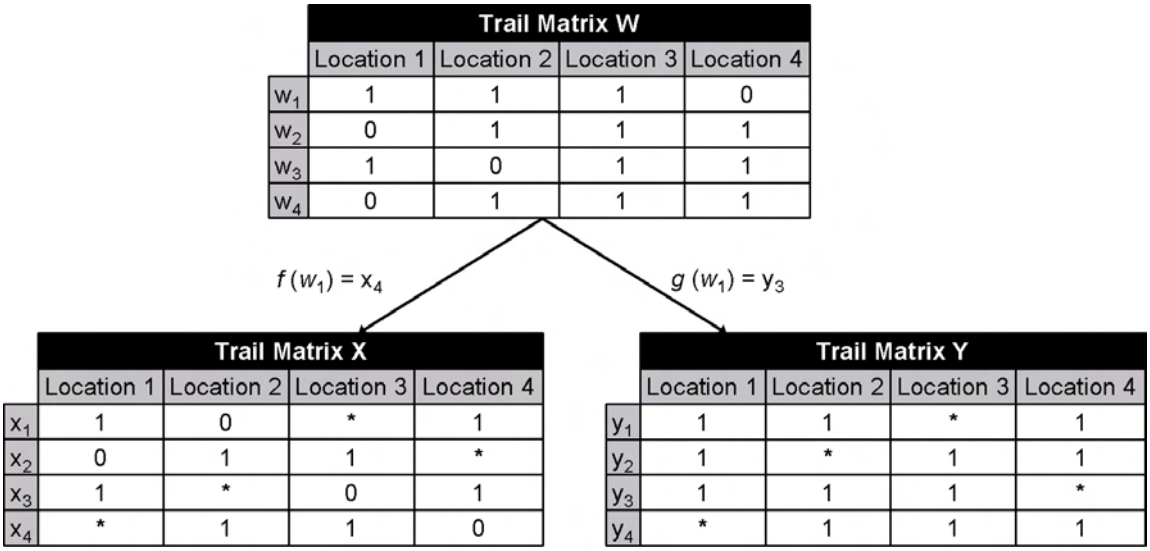


Figure 2.1: Trail Matrices X and Y have an association relationship via trail matrix W.

distinction, while subtle, is necessary to discriminate between trails and the entities that generate trails. The notion of an association is more precisely presented in Definition 3. Figure 2.1 provides an example of an association relationship for trail matrices X and Y. There is an association between x_4 and y_3.

Definition 3 (Association). *Given an association relationship exists between trail matrices X and Y, we say there exists an* association *between row $x \in X$ and row $y \in Y$ when $f^{-1}(x) = g^{-1}(y)$.*

Though we may know an association relationship exists, we are not provided with functions f and g. The goal of *trail linkage* is to match those trails in X and Y that have the same ancestor. In this dissertation, we approach the problem by discovering functions f and g. We concentrate on deterministic models of linkage, such that we group as many associations together as possible without grouping any trails that are not associated. We call this challenge the trail linkage problem (Definition 4).

Definition 4 (Trail Linkage Problem). *Given trail matrices X and Y with an association relationship, such that functions f and g are unknown, find the relation ${}_XR_Y$ with the largest membership, such that $(x, y) \in {}_XR_Y$ if, and only if, $f^{-1}(x) = g^{-1}(y)$.*

The objective of trail re-identification is to solve the trail linkage problem.

2.2 A Graph Theory Primer

How can we leverage trail matrices with an association relationship to discover associations? The answer to this question is best understood in the context of graph theory.

For completeness, we review relevant graph theory definitions. Let $G = (V_X \cup V_Y, E)$ be a bipartite graph, such that $V_X \cap V_Y = \emptyset$ and $E \subseteq V_X \times V_Y$. A *matching* in G is a subgraph $H = (V_X \cup V_Y, F \subseteq E)$, in which no two edges in F share an endpoint. A *maximum matching* is a matching with the largest number of edges.[2]

2.2.1 Stable Marriage vs. Maximum Matching

In the stable marriage problem [60], vertices in V_X and V_Y, are referred to as men and women. For conformity, we adopt these terms. Each man $x \in V_X$ ranks the women according to his preference, so edge e_{xy} is directed from vertex x to $y \in V_Y$ and weighted according to rank. Similarly, women rank the men. The goal is to find a matching, such that each edge connecting a man and woman, or marriage, is *stable*. A marriage is said to be stable if no man and woman would both prefer to be married to each other in comparison to their current state. More precisely, if x and y both rank each other higher than their current partners, then their marriage is said to be *unstable*. Otherwise, the matching is said to be stable.

There are many variants of the stable marriage problem. Of most relevance to the trail linkage problem, it is known that when the *Incomplete*, *Unweighted*, and *Undirected* constraints hold true:

> **Incomplete** Not every man/woman is an acceptable mate for a member of the opposite sex
>
> **Unweighted** Every man/woman is indifferent between its list of acceptable mates
>
> **Undirected** Every acceptable mate reciprocates (i.e., man likes woman and woman likes man)

solving the stable marriage problem reduces to finding a maximum matching in G. There can simultaneously exist more than one maximum matching in a graph and, consequentially, more than one stable marriage is possible for a vertex. For instance, in the graph in Figure 2.2(a), both of the maximum matchings (Figures 2.2(b) and 2.2(c)) depicted are stable marriages.

[2] In the literature, this is sometimes called the *maximum cardinality matching*.

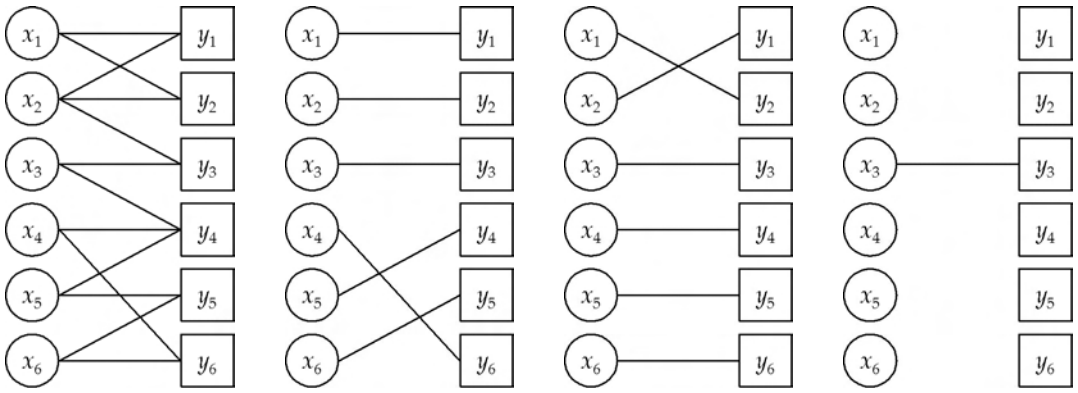


Figure 2.2: Graphs in 2.2(b) and 2.2(c) are maximum matchings for the graph in 2.2(a). Enumeration of the set of maximum matchings reveals only x_3 and y_3 make up a true love.

2.2.2 True Loves

We claim an association can be represented by a specific type of marriage that we call a *true love*. A true love is a marriage that exists in every maximum matching (Definition 5). Figure 2.2(d) depicts true loves for the sample graph in Figure 2.2(a).

Definition 5 (True Love). *Let* $\mathbf{T}_{XY} = \{T^1_{XY}, T^2_{XY}, \ldots, T^z_{XY}\}$ *represent the set of maximum matchings in bipartite graph* $G = (V_X \cup V_Y, E)$. *Vertices* $x \in V_X$ *and* $y \in V_Y$ *make up a* true love *when edge* $e_{xy} \in \bigcap_{i=1}^{z} T^i_{XY}$.

2.3 Trail Linkage and True Loves

Theorem 1 proves that associations equal true loves. We prove our claim regarding true loves and associations in a step-wise manner. First, we summarize the observed relationships between two trail matrices in what we term a *link matrix* (Definition 6). In essence, the link matrix communicates when two trails potentially correspond to the same unambiguous pattern of values.

Definition 6 (Link Matrix). *Let X and Y be trail matrices with an association relationship, respectively. A* link matrix L_{XY} *is an $m \times m$ matrix, such that:*

$$L_{XY}[i,j] = \begin{cases} 1, & \text{if } \forall z \in \{1,\ldots,p\}\text{: } a_{i,z} = a_{j,z} \vee a_{i,z} = * \vee a_{j,z} = * \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

When $L_{XY}[i,j] = 1$ we say there exists a link *between trails i and j, otherwise we say there exists a* non-link*.*

The link matrix is tantamount to the bipartite graph $G = (V_X \cup V_Y, E)$, where vertices V_X and V_Y correspond to the rows of trail matrices X and Y, respectively, and the edgeset $E = \{e_{ij} | L_{XY}[i,j] = 1\}$.

Next, we state an important lemma regarding the translation of associations into the link matrix (Lemma 1). Specifically, if two trails are an association, then they must be linked in the link matrix.

Lemma 1 (Associations are Links). *Given that trail matrices X and Y have an association relationship, if $x \in X$ and $y \in Y$ are an association, then $L_{XY}[x,y] = 1$.*

PROOF. By Definition 3, when trails x and y are an association, $f^{-1}(x) = g^{-1}(y)$. Therefore, trails x and y have a single unambiguous trail in common. As a result, trails x and y can be made equivalent by changing ambiguous to unambiguous values only, which satisfies the definition of a link. □

Now, we recognize that every trail participates in one association only (Lemma 2). Moreover, it follows as a corollary that every association is mutually exclusive, such that no two associations have an ancestor trail in common (Corollary 1).

Lemma 2 (One Association Per Trail). *Given trail matrix X and Y have an association relationship, every trail participates in one, and only one, association.*

PROOF. This follows directly from the existence of bijective functions f and g in Definition 2. □

Corollary 1 (Associations are Exclusive). *Given trail matrices X and Y have an association relationship, all associations are mutually exclusive.*

Finally, we have the necessary tools to prove true loves represent associations in trail matrices (Theorem 1).

Theorem 1 (True Loves Are Associations). *Let $G = (V_X \cup V_Y, E)$ be the bipartite graph of link matrix L_{XY}, such that there exists an association relationship between trail matrices X and Y. Every true love in G is an association for X and Y.*

PROOF. By Lemmas 1, 2, and Corollary 1 it directly follows that there exists one, and only one maximum matching that captures all associations between X and Y without capturing any non-associations. Therefore, if there exists a true love in the set of maximum matchings, it must correspond to an association. ■

Theorem 1 allows us to state every true love implies an association. Unfortunately, we can not claim the reverse implication always holds true. In other words, not every association corresponds to a true love. This begs the important question, "Can we construct a procedure to discover associations that are not true loves in G without including non-associations?" Interestingly, we can prove that this is impossible (Theorem 2). This has important ramifications. Of most interest with respect to the trail linkage problem it means that the set of true loves in G is the largest set of associations any deterministic algorithm can discover.

To prove the latter claim, we need several more tools. First of all, the removal of true loves and edges associated with them in the bipartite graph does not remove edges for associations that are not true loves (Lemma 3).

Lemma 3 (Removing True Loves Does not Affect Remaining Associations). *Let $G = (V_X \cup V_Y, E)$ be the bipartite graph of link matrix L_{XY}, such that there exists an association relationship between trail matrices X and Y. Let $H = (V_A \cup V_B, F)$, be the largest subgraph of G in which all vertices in a true love and edges that end at true vertices are removed. There exists a matching in H that captures every remaining association.*

PROOF. By Lemma 1, there exists an edge in G for every association between X and Y. By removing true loves and all edges ending at such vertices, we do not remove any edges that connect the remaining associations. □

Furthermore, a simple and useful corollary of the previous finding is that every remaining vertex has an association (Corollary 2).

Corollary 2 (All Remaining Vertices Have Associations). *With respect to Lemma 3, all remaining vertices in graph H have an edge corresponding to an association.*

Now, we prove every remaining edge in the bipartite graph can represent a feasible alternative solution to the trail linkage problem (Theorem 2). More specifically, for the remaining set of trails (i.e., not true loves), deciding if any edge corresponds to a non-association is at least as likely as deciding if the edge corresponds to an association.

Theorem 2 (True Loves Are Optimal Solution). *Let $G = (V_X \cup V_Y, E)$ be the bipartite graph of link matrix L_{XY}, such that there exists an association relationship between trail matrices X and Y. If a vertex does not participate in a true love, there exists*

no computational method that can decide when the vertex participates in an association.

PROOF. Let $H = (V_A \cup V_B, F)$, be the largest subgraph of G, such that no vertices participate in a true love and no edges connect to vertices in a true love. Let $\mathbf{T}_{AB}$ represent the set of maximum matchings in graph H. Let Z_{AB} be the union of maximum matchings $\bigcup_{i=1}^{z} T_{AB}^{i}$.

By Corollary 2, for each vertex $a \in V_A$ there exists a vertex $b \in V_B$, such that $e_{ab} \in Z_{AB}$ and a and b are an association. Similarly for V_B. Moreover, since no vertex in H participates in a true love, every vertex must participate in at least two edges in Z_{AB}. Again, since no vertex in H participates in a true love, it follows there must exist at least two maximum matchings in H that have zero edges in common. One of the maximum matchings captures all associations without any non-associations. However, it is not possible to discern which matching is the correct one. In other words, every edge in Z_{AB} is at least as likely to connect a non-association as an association. Therefore, if a vertex does not participate in a true love, no computational method can consistently correctly decide when it is in an association. ■

2.4 An Existing Graph Theory Solution

From Theorem 2, it follows that the largest set of associations we can discover without accepting non-associations is equivalent to the set of true loves. One way to extract all true loves is by enumerating all maximum matchings in G. After enumeration, we can find all re-identifications in a scan in O($|V_X| \cdot |V_Y|$) time, or O(n^2), where n is the number of trails in a trail matrix.

Based on prior and current research in graph theory, the enumeration of maximum matchings is a non-trivial feat. The fastest technique to find a single maximum matching is the alternating paths algorithm of Hopcroft and Karp, which has worst case complexity O($|E|\sqrt{|V_X \cup V_Y|}$) [66]. With respect to trail matrices, since $|V_X| = |V_Y| = n$, the complexity of maximum matching discovery is O($n^{5/2}$). Moreover, the fastest known algorithm to enumerate all maximum matchings has complexity O($|E|\sqrt{|V_X \cup V_Y|} + \mu_G |V_X \cup V_Y|$), where μ_G is the number of maximum matches in G [145]. The form of this complexity statement is similar to that of finding a single maximum matching. However, there are potentially on the order of $n!$ maximum matchings in the graph, so enumeration is O($n^{5/2} + n! \cdot n$). This latter term dominates for $n \geq 3$, so enumeration requires O($n! \cdot n$).

2.5 Conclusions

In this chapter we introduced a formal model of trails and the trail linkage problem. We demonstrated the problem has an intuitive relationship to graph theory. Given this relationship, we proved that an optimal solution to the trail linkage problem can be achieved using existing graph theoretic methods to enumerate the set of maximum matchings in a bipartite graph. However, this solution is grossly inefficient. In the following chapters we illustrate certain aspects of trail composition allow us to construct much more efficient algorithms than enumerating all maximum matchings to solve the trail linkage problem.

Chapter 3

Trail Composition

The previous chapter concentrated on trail linkage; however, trails must come from somewhere. In this chapter, we investigate and formalize how trails are composed from data stored at multiple locations. Table 3.1 provides a summary of the symbols used.

The basic terms and definitions are borrowed from relational database theory and [136]. The term *data* refers to information managed by a particular location. At each data collecting location, data is organized as a database which, for simplicity, we model as a table that consists of a set of rows and columns. Each column corresponds to an attribute, which is a semantic category of information that refers to people, machines, or other entities. Each row is made up of data specific to a person, machine, or other entity.

More formally, we represent a database as $\tau(A_1, A_2, \ldots, A_p)$, where the set of attributes is $A^\tau = \{A_1, A_2, \ldots, A_p\}$ and each attribute is associated with its own domain of specific values. Each row in the database is a p-tuple, which we represent in vector form $[a_1, a_2, \ldots, a_p]$, such that each value a_i is in the domain of attribute A_i. We define the size of the database as the number of tuples and use cardinality, or $|\tau|$, to represent this concept. For example, Figure 3.1 depicts the database of a location of the form τ(*Name, Birthdate, Gender, Zip, Treatment, DNA*). A specific tuple in this database is [*Alice, 1/5/1950, M, 10000, 900, actg*].

τ					
Name	Birthdate	Gender	Zip	Treatment	DNA
Alice	1/5/1950	M	10000	900	actg
Bob	2/6/1960	F	20000	800	ctga
Charlie	3/7/1970	M	30000	700	tgac

Figure 3.1: A database maintained by a location.

Symbol	Description
$C = \{c_1, \ldots, c_{\|C\|}\}$	Data collecting/releasing locations
$S = \{s_1, \ldots, s_{\|S\|}\}$	Population of entities
Database Particulars	
$\tau(A_1, \ldots, A_p)$	Database table with attribute set $A^\tau = \{A_1, \ldots, A_p\}$
$\|\tau\|$	Number of tuples in τ
$t[a_1, \ldots, a_p]$	Tuple t with sequence of values, $a_1 \in A_1$, ... ,$a_p \in A_p$
$T = \{\tau_1, \ldots \tau_{\|C\|}\}$	Private databases maintained by c_1, ... $c_{\|C\|}$
$\Gamma = \{\gamma_1, \ldots, \gamma_{\|C\|}\}$ $\Omega = \{\omega_1, \ldots, \omega_{\|C\|}\}$	Sets of partitioned databases from c_1, ... $c_{\|C\|}$, where $\tau \rightarrow \{\gamma_c, \omega_c\}$
Trail Particulars	
$M = \{m_1, \ldots, m_{\|M\|}\}$ $N = \{n_1, \ldots, n_{\|N\|}\}$	Union of tuples in Γ, Ω, respectively
$m_\Gamma = [b_{m,1}, \ldots, b_{m,\|C\|}]$	Trail of element m in Γ
M_Γ	Trail matrix for tuples in M as observed in Γ
$TRAILDATA_\Gamma$: $\{1, ...\|M\|\} \rightarrow M$	Pointer structure between rows in M_Γ and corresponding tuples in M
$m_\Gamma \preceq n_\Omega$	m_Γ is the subtrail of n_Ω
$m_\Gamma \succeq n_\Omega$	m_Γ is the supertrail of n_Ω

Table 3.1: Summary of symbols

Before continuing, we specify the fundamental assumptions adopted for this research.

3.1 Model Assumptions

In this section, four core assumptions that are inherent to the disclosure model are made evident. These assumptions are related to the integrity, uniqueness, traceability, and identifiability of the data. The assumptions are imposed at both the ancestor level and, as a consequence, on data releases.

Assumption 1 (Integrity). *Each location can certify the integrity of its internal and, subsequently, disclosed databases. As such, all data released from a location is truthful, such that each tuple correctly represents an entity that visited the location.*

The first assumption states that each location maintains control, or is confident, in the data collection and disclosure process. Thus, a tuple in a location's database corresponds to a

single underlying entity only. For example, it is not possible for a particular hospital to release the DNA sequences of a patient named "Alice" if "Alice" never visited the hospital.

Assumption 2 (Uniqueness). *In a location's database, the entity represented by a single tuple is unique, such that no tuple represents multiple entities.*

It follows from the uniqueness of tuples assumption that both de-identified and identified databases represent a distinct set of references to people, machines, or other entities that have visited a location, but not necessarily the frequency of visits. As a result, these references narrowly relate to a person, machine, household or other entity to be identified.

Assumption 3 (Traceable). *Data corresponding to entities and, subsequently, used to create trails are traceable. By traceable it is meant there exists the same type of information across a set of locations.*

For example, with respect to genomic data, we may assume the same DNA sequence tracked across multiple hospitals belongs to the same patient or set of patients.

Assumption 4 (Identifiable). *Some of the attributes disclosed for the data are explicitly identifiable.*

The identifiable assumption forms the basis for a compromise in privacy. If no data was identifiable, then data can be linked but not re-identified.

3.2 Database Partitions for De-identification

This dissertation is concerned with the re-identifiability of data. As such, Definition 7 indicates the difference between *identified* and the *de-identified* data types. This definition is based on the model introduced in prior work by Sweeney [139].

Definition 7 (Identified / De-Identified Database). *A database τ is said to be* identified *if A^{τ} includes explicit identifying attributes, such as name or residential address, or attributes that are known to be directly linkable to explicit identifiers. If τ is not identified, then it said to be* de-identified*.*

Identified data is information that can be applied to discriminantly contact an individual entity. For instance, the personal name, residential address, and phone number for a particular person can be considered identified data because it empowers the holder of such information with the ability to send a communication directly to a particular entity. In comparison, the notion of de-identified data is more subtle and requires a bit more clarification. In Definition 7, we state this is data devoid of identity, but this does not imply the

data is "anonymous". Rather, it means that viewed out of context, and without access to additional knowledge, there is no relationship between the observed data and the identity of the corresponding entity.

In this research, we study environments where data holding locations attempt to protect the anonymity of sensitive data by stripping explicit identifying attributes from sensitive data. In doing so, data holding locations partition identified and de-identified data and make separate database disclosures. As such, in our model, each data holder releases a two-table vertical partition of its privately held data. The notion of a partitioned database disclosure is more formally addressed in Definition 8. The latter part of Definition 8 states that the sets of attributes for the partitioned databases are independent. With respect to the model, consider a recipient of the partitioned databases. The recipient knows both γ_c and ω_c are derived from a master database τ_c stored at location c. However, without additional knowledge, there is no way for the recipient to directly relate the attributes of γ_c and ω_c. As a result, the recipient is incapable of relating the tuples in γ_c to tuples in ω_c with chance better than a random assignment.

Definition 8 (Partitioned Disclosure). *Let $\tau_c(A_1, A_2, \ldots, A_p)$ be a database maintained by location c. A vertical partitioning is achieved by splitting τ_c into two tables $\gamma_c(A_1, \ldots, A_i)$ and $\omega_c(A_{i+1}, \ldots, A_j)$, with attributes $A^{\gamma_c} \subset A^{\tau_c}$ and $A^{\omega_c} \subset A^{\tau_c}$. We assume there exists no known relation ${}_{A^{\omega_c}}R_{A^{\gamma_c}}$ that non-randomly relates attributes in A^{ω_c} to attributes A^{γ_c}.*[1]

From a broader perspective, Definition 8 implies this research is situated in an environment where no such inferential relationships between partitioned attributes exist. Stepping beyond the computational model, we note that we do encounter such environments in the real world and elaborate on this point in Chapter 5. Therefore, this aspect of our model is not a simplification, but a worst case scenario. If this assumption is found to be false, then, upon relaxation of the assumption, de-identified data can only become more re-identifiable because additional knowledge can be included in the linkage process.[2]

Taken in combination, Definitions 7 and 8 provide the basis for our model of partitioned disclosure for anonymity protection. Specifically, we investigate an environment

[1]For a more clean presentation, henceforth we assume all data holders use the same database schema in τ, γ, and ω. Thus, we do not use a location subscript when referring to a table's attribute set. For instance, the attribute sets of databases γ_i and γ_j, from locations i and j, are both referenced as A^{γ}, as opposed to A^{γ_i} and A^{γ_j}. This is not a necessary assumption and in practice we do not assume equivalence, but model the relations between databases of similar data types.

[2]We recognize there are cases when attribute-based inferential relations do exist. In Appendix B (Genomic Data Privacy Vulnerability Assessment) we provide an example of how they can be incorporated into the re-identification process for genomic data, as well as their relationship to trail re-identification.

where internal databases are disclosed in a partitioned state, such that one partition is an identified database and the other is a de-identified database.

Beyond attribute independence, we mention the order in which tuples appear in the disclosed partitioned databases is assumed to be randomized. Ordering permits additional knowledge, such as temporal constraints, to be included in the linkage process. For instance, linkage could be achieved between partitioned databases by performing a simple join on the tables' order. Figure 3.2 illustrates an example of the released partitions with random order for the private database in Figure 3.1. Name, demographics, and clinical codes are reported for a set of patients in the identified table ψ. Genomic sequences are reported in the de-identified table δ. Without a primary key, there appears to be no way to link sensitive records in δ to identified records in ψ.

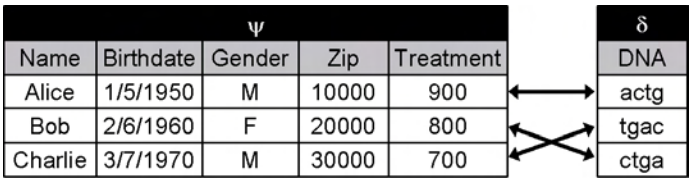


Figure 3.2: Vertical partition of private database into an identified table ψ and a de-identified table δ. Arrows represent the underlying truthful relationships of tuples known by the data holder.

Notice, if a tuple from τ is to be released in a partitioned database, Definition 8 specifies there is a constraint regarding which attributes the tuple may cover. However, Definition 8 does not specify how data in each of the tables relates to each other. For our research, we assume there do exist certain constraints regarding the relationships between the underlying data and the disclosed databases. The details of these constraints are laid out in the following section.

3.2.1 Data Multiplicity

With our fundamental assumptions specified, we continue with a discussion of the multiplicity of data. Multiplicity corresponds to the number of different data elements that can belong to the same entity. Our research addresses several types of multiplicity between data types. However, in the core chapters of this dissertation we design algorithms to address data that is *one-to-one*. A second kind of multiplicity we study is *one-to-many* and we refer the reader to Appendix A for a characterization of this latter environment.

Intuitively, *one-to-one* data means that for each distinct tuple of data of one type (e.g., defined over attribute set A^{γ}), there is only one corresponding tuple of data of the other

type (e.g., defined over A^{ω}), such that both tuples correspond to the same underlying entity. Formally, this is specified in Definition 9. For example, in a one-to-one environment, each genomic data sequence corresponds to one patient name only, and vice versa.

Definition 9 (One-to-One Data). *Let τ_c be partitioned into databases γ_c and ω_c as specified in Definition 8. Let M and N be the set of distinct tuples defined over A^{γ} and A^{ω}, respectively. The partitioned databases are said to be one-to-one if there exists a bijective function $\alpha : M \rightarrow N$.*

3.2.2 Data Completeness

The previous sections discussed various assumptions and data relationships. We now continue with a presentation of relationships between partitioned databases in the form of disclosure tactics. Briefly, in *unreserved* databases, a location discloses data from a tuple of an internal database in both partitions. Alternatively, in *reserved* databases, the location releases only a subset of one type of data.

The first tactic is referred to as an *unreserved* disclosure. Definition 10 presents a functional model for disclosed databases that adhere to an unreserved property. When this tactic is employed, tuples present in a de-identified database have corresponding tuples in an identified database, and vice versa.

Definition 10 (Unreserved Databases). *Let database τ_c be vertically partitioned by the functions F_{id} and F_{de} such that F_{id}: $\tau_c \rightarrow \gamma_c$ and F_{de}: $\tau_c \rightarrow \omega_c$. Databases γ_c and ω_c are* unreserved *if and only if:*

1. $\forall t_{id} \in \gamma_c, \exists t_{de} \in \omega_c : {F_{id}}^{-1}(t_{id}) = {F_{de}}^{-1}(t_{de})$, *and*
2. $\forall t_{de} \in \omega_c, \exists t_{id} \in \gamma_c : {F_{de}}^{-1}(t_{de}) = {F_{id}}^{-1}(t_{id})$.

In releases that adhere to the unreserved property, every tuple from the data-collecting location that is present in the de-identified table is also present in the identified table, and vice versa. Figure 3.2 depicts an unreserved release of the data shown in Figure 3.1. For example, *Alice* is in ψ and her corresponding genomic data *actg* is disclosed in δ.

Nonetheless, entities shedding data, as well as data releasers, are autonomous. Either of these groups may choose to withhold information. For example, a patient may refuse to contribute a blood or DNA sample to a hospital's databank. Similarly, a location may choose not to share all collected information. As a result, unreserved releases are neither always practical nor possible. In this case, at least one of the disclosed databases is missing information with respect to the other disclosed table. For this research, we consider a special case, in which only one type of data is missing information. Informally, one data

type can be disclosed only when the other type of data is disclosed. When this occurs, we say the database with missing information is *reserved* to the other table. This is formalized in Definition 11.

In Figure 3.3, the de-identified database δ is reserved to the identified database ψ. With reference to the true mappings of genomic data to identity in Figure 3.2, *Bob* is in ψ, but *ctga* is not disclosed in δ. It can be visually verified there is no genomic sequence released without its corresponding identity.

Definition 11 (Reserved Databases). *Let database τ_c be vertically partitioned by functions G_{id} and G_{de} such that G_{id}: $\tau_c \rightarrow \gamma_c$ and G_{de}: $\tau_c \rightarrow \omega_c$. The tables γ_c and ω_c are in a* reserved *state if either:*

1. $\forall t_{id} \in \gamma_c, \exists t_{de} \in \omega_c : {G_{id}}^{-1}(t_{id}) = {G_{de}}^{-1}(t_{de})$*; or,*
2. $\forall t_{de} \in \omega_c, \exists t_{id} \in \gamma_c : {G_{de}}^{-1}(t_{de}) = {G_{id}}^{-1}(t_{id})$.

When the first condition holds, we say γ_c is reserved to ω_c; and vice versa when the second condition holds.

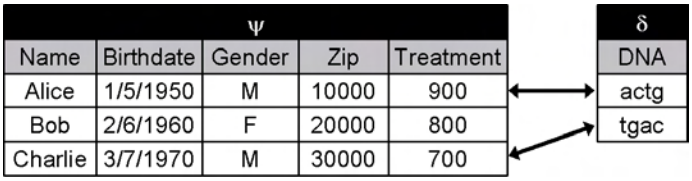


Figure 3.3: Vertical partitioning that satisfies the reserved property: δ is reserved to ψ.

Clearly, reserved databases are a generalization of unreserved databases. In terms of Definition 11, two databases are said to be unreserved when both conditions 1 and 2 hold.

3.3 Multilocation Environment and Trails

Trail re-identification manifests because multiple locations collect and disclose information on a common population of entities. So far, we have considered relationships between a single location's databases. In this section, we describe how a multi-location system contributes to the tracking of an individual entity.

For modelling purposes, we specify several additional variables. Let $C = \{c_1, \ldots, c_{|C|}\}$ be the set of locations disclosing data. Let $\Gamma = \{\gamma_1, \ldots, \gamma_{|C|}\}$ and $\Omega = \{\omega_1, \ldots, \omega_{|C|}\}$ be the set of partitioned databases disclosed over the set of locations. In addition, let M and N be the distinct union of tuples from Γ and Ω, respectively.

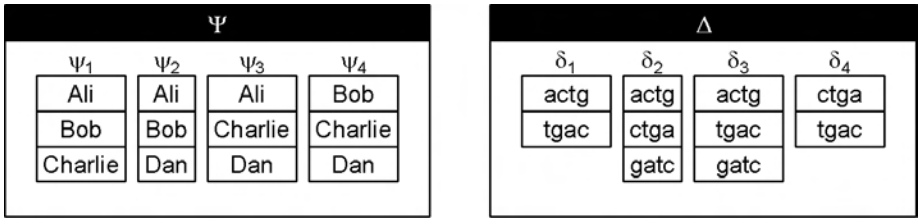


Figure 3.4: Identified and de-identified databases disclosed by four hospitals.

An entity can be tracked by observing which locations report the presence and/or absence of particular data elements. Recall, in Definition 1 we defined a trail as the representation of a particular data element across locations. We now clarify this definition to provide semantics for the values 1, 0, and * (Definition 12).

Definition 12 (Trail, Revisited). *Let Z be the set of unique data elements derived from a set of disclosed databases $\Lambda = \{\lambda_1, \ldots, \lambda_{|C|}\}$. The* trail *of data element $z \in Z$ is a vector of values $z_\Lambda = [b_{z,1}, \ldots, b_{z,|C|}]$, such that:*

$$b_{z,c} = \begin{cases} 1, & \textit{if } z \textit{ unambiguously reveals the presence of the corresponding entity at location } c. \\ 0, & \textit{if } z \textit{ unambiguously reveals the absence of the corresponding entity at location } c. \\ *, & \textit{otherwise (i.e., the ambiguous value).} \end{cases}$$

Also recall from Definition 1, trails for a particular type of data are succinctly organized into a matrix. Informally, a trail matrix is constructed such that each row contains information about a particular data element's visit status.[3] Definition 13 provides a more formal model of trail matrices with respect to disclosed databases.

Definition 13 (Trail Matrix, Revisited). Let Λ be a set of disclosed databases and let C be a set of locations. A trail matrix Z_Λ is a matrix with dimensions $|Z| \times |C|$ matrix, such that each row corresponds to the trail of a different element in Z. ■

A trail corresponds to the observed location-visit pattern for a particular data element. From a procedural standpoint, however, there is no clear way to reference a tuple from its corresponding row, and vice versa. Thus, in Definition 14 a simple function, which we call a *trail-data pointer*, is provided to map between row numbers in a trail matrix and the affiliated data elements.

[3]When trail matrices correspond to identified and de-identified data, the matrices are termed de-identified and identified trail matrices.

Definition 14 (Trail-Data Pointer). *A* trail-data pointer $TRAILDATA_\Lambda : \{1, \ldots, |Z|\} \rightarrow Z$ *is a bijective function that maps each row number of a trail matrix* Z_Λ *to its corresponding data element in* Z.

For example, in Figure 3.5, *TRAILDATA*$_\Psi$(3) = *Charlie*.

With a mathematical definition of data, trails, and matrices, we now extend the definitions of unreserved and reserved. We extend our definition from a single location to all locations in the form of the constructed trail matrices. These extensions are detailed in Definitions 15 and 16.

Definition 15 (Unreserved Trail Matrices). *Trail matrices* M_Γ *and* N_Ω *are unreserved if* $\forall c \in C : \gamma_c$ *and* ω_c *are unreserved.*

Definition 16 (Reserved Trail Matrices). *Trail matrix* N_Ω *is reserved to* M_Γ *if* $\forall c \in C :$ ω_c *is reserved to* γ_c.

In Figure 3.4 we depict an example of a data sharing scenario with four locations in which every database δ_i is reserved to databases ψ_i. From Figure 3.4, we can construct the trail matrices shown in Figure 3.5. Note, all values in X_Ψ are unambiguous 0's and 1's because of the manner by which databases are reserved. In contrast, there are ambiguous values in Y_Δ.

X_Ψ	H_1	H_2	H_3	H_4
Ali	1	1	1	0
Bob	1	1	0	1
Charlie	1	0	1	1
Dan	0	1	1	1

Y_Δ	H_1	H_2	H_3	H_4
actg	1	1	1	*
ctga	*	1	*	1
tgac	1	*	1	1
gatc	*	1	1	*

Figure 3.5: Trail matrices constructed from databases in Figure 3.4. Trail matrix Y_Δ is reserved to X_Ψ.

Trails provide a common feature set for discovering linkages between databases of different data types. One particular relationship between trails that we will leverage is called the subtrail relationship (Definition 17). We say trail s is the *subtrail* of trail t if s can be converted into t by changing ambiguous values in s only. Conversely, trail t is said to be the supertrail of s.

Definition 17 (Subtrails / Supertrails). *Let* M_Γ *and* N_Ω *be the trail matrices that are constructed from location set* C. *Trail* $n_\Omega = [b_{n,1}, \ldots, b_{n,|C|}]$ *is the* subtrail *of* $m_\Gamma =$

$[b_{m,1}, \ldots, b_{m,|C|}]$ *if, and only if,* $\forall c \in C$*:* $b_{n,c} \in \{*, b_{m,c}\}$*. For brevity, we use the notation* $n_\Delta \preceq m_\Gamma$ *to represent that* n_Ω *is the subtrail of* m_Γ*. Similarly,* m_Γ *is said to be the* supertrail*, or* $m_\Gamma \succeq n_\Delta$*.*

For example, in Figure 3.5, the identified trails *Bob*[1,1,0,1] and *Dan*[0,1,1,1] are both supertrails of the de-identified trail *ctga*[*,1,*,1]. Similarly, the de-identified trails for *gatc*[*,1,1,*] and *actg*[1,1,1,*], are subtrails of the identified trail for *Ali*[1,1,1,0].

This section precisely described how people, machines, and other entities leave information behind at visited locations, how that information can be shared and used to construct trails, and how those trails can pose a re-identification problem. In the next section, we provide a method to automate the construction of a pair of trail matrices.

3.4 Automated Construction of Trails

In the following Chapter 4 we present algorithms that exploit unique occurrences in trails for re-identification. The re-identification algorithms accept trail matrices as input. In this chapter, though we presented various disclosure tactics, we glossed over how these matrices come to exist. Specifically, there is no simple join or query that can be called in a standard database to construct a trail matrix from a set of tables. Therefore, for a characterization of the base level of computational complexity for the re-identification algorithms, we need to describe how databases are converted into a matrix representation.

3.4.1 FillTrails

We accomplish the composition of trail matrices through the *FillTrails* procedure shown in Algorithm 1. Before we walk through the computational steps of *FillTrails*, let us explore how the procedure relates to the real world. For illustration, consider the examples used throughout this chapter in which de-identified DNA databases are reserved to identified clinical databases. We know that an individual's name must be disclosed from location c when the individual's corresponding DNA is disclosed from location c. As a consequence, the identified trail matrix can not contain any ambiguous values. If it did contain ambiguous values, then the reserved property would not be satisfied. As a result, we can initialize the identified trail matrix to be all 0's. Then, we can populate the identified trail matrix with 1's as we read information from the disclosed identified databases.

In contrast, an individual's DNA is not always disclosed when the corresponding name is disclosed. If a piece of DNA is not disclosed location c, we can not confirm that the corresponding name failed to visit location c. This means that the DNA trail matrix can contain ambiguous values and thus we initialize the DNA trail matrix to be all *'s. As we

read information from the disclosed DNA databases, we populate the DNA trail matrix with 1's. Finally, when a location's DNA database and identified database are the same size, then we confirm that for each piece of DNA that was not disclosed from location c, the corresponding name was not disclosed from location c as well. Therefore, every remaining * in the DNA trail matrix for location c's column must correspond to a 0.

A more technical walkthrough of the *FillTrails* procedure follows. *FillTrails* accepts as input the partitioned databases collected from locations $c_1, \ldots, c_{|C|}$, and a vector Q, each cell of which contains the name of a particular location. The algorithm basically scans all databases two times.[4] First, the algorithm scans all submitted databases to construct the union of tuples, called M and N. Second, the algorithm initializes trail matrices M_Γ and N_Ω, to hold the trails for the tuples in M and N, respectively. These matrices have size $|M| \times |C|$ and $|N| \times |C|$, respectively. The algorithm assumes databases in Ω are reserved to databases in Γ. As such all cells in M_Γ are initialized to 0 and cells in N_Γ are initialized to *. For completeness, it should be noted that the *TRAILDATA* pointer structure is accomplished during this step. Third, the algorithm scans all tuples again. It assigns a value of 1 to the corresponding trail matrix when a tuple is found in a particular location's database. In contrast, when the tuple is not found in a particular location's database, if the database is from Ω then M_Γ is assigned the value of 0 when the partitioned databases from the location in consideration are of the same size.

3.4.2 Base Complexity

The computational complexity of *FillTrails* can be analyzed as follows. We need only to consider the time necessary for each scan, since the initialization of the matrix can be performed in a single memory allocation. In a worst case scenario, each database contains every tuple. In this case, we require $|C|$ scans of a database with size potentially as large as $|M|$. Thus, the base level of complexity for the REIDIT algorithms presented below is O($|C| \cdot |M|$).

From an application perspective, big-O complexity based on maximum size of a database is a poor characterization for the complexity of trail matrix construction. This is because the distribution of people to locations is such that the density of the trail matrix (i.e., ratio of number of cells with value 1's to total number of cells) tends to be far below 1. As a result, we claim complexity in real world situations is on the order of $\Theta(|C| \log |M|)$. This will be empirically validated in Chapter 5.

[4] A more efficient version that requires only one scan can be designed. However, such a doubling does not affect our complexity analysis. The one scan version requires a bit more explanation, so for clarity we present this slightly less efficient version.

Algorithm 1 FillTrails(Γ, Ω, Q)

Input: $\Gamma = \{\gamma_1, \ldots, \gamma_{|C|}\}$, $\Omega = \{\omega_1, \ldots, \omega_{|C|}\}$, sets of partitioned databases for locations $c_1, \ldots, c_{|C|}$ such that ω_i reserved to γ_i; Q, a vector containing the members of C.

Output: M, N, the set of distinct tuples in Γ, Ω, respectively; M_Γ, N_Ω, trail matrix for Γ, Ω, respectively.

let $M \leftarrow \bigcup_{c \in C} \gamma_c$
let $N \leftarrow \bigcup_{c \in C} \omega_c$ *//the set of distinct tuples in* Γ, Ω
let M_Γ be a $|M| \times |C|$ matrix with each cell $M_\Gamma[i, j]$ initialized to 0 value
let N_Ω be a $|N| \times |C|$ matrix with each cell $N_\Omega[i, j]$ initialized to * value
//Concurrently, build the trail-data pointer structures
for each $c \in C$ **do**
  $q \leftarrow$ the index of c in C
  for each $m \in \gamma_c$ **do**
    $M_\Gamma[TRAILDATA_\Gamma{}^{-1}(m), q] \leftarrow 1$
  end for
  for each $n \in N$ **do**
    if $n \in \omega_c$ **then**
      $N_\Omega[TRAILDATA_\Omega{}^{-1}(n), q] \leftarrow 1$
    else if $|\gamma_c| \geq |\omega_c|$ **then**
      $N_\Omega[TRAILDATA_\Omega{}^{-1}(n), q] \leftarrow 0$
    end if
  end for
end for
return M, N, M_Γ, N_Ω

3.5 Conclusions

This chapter described where trails come from and how we can construct them in an automated manner. We walked through how people, machines, and other entities leave data behind at visited locations and how data can be shared. Data relationships and disclosure tactics were described in a formal manner with precise definitions of the disclosure environment. In addition, we presented a procedure to construct a pair of trail matrices from partitioned databases. In Chapter 4, we describe several algorithms for discovering associations and solving the trail linkage problem.

Chapter 4

Trail Re-identification Algorithms

To learn re-identifications, we developed algorithms that exploit unique features in data trails. Collectively, the algorithms are termed the RE-Identification of Data in Trails, or REIDIT, algorithms. We have constructed three algorithms, called REIDIT-Complete, REIDIT-Incomplete, and REIDIT-Multiple. For short, they are referred to as REIDIT-C, -I, and -M. Both the REIDIT-C and REIDIT-I algorithms assume data multiplicity is one-to-one, whereas REIDIT-M assumes the data is not one-to-one. For clarity in presentation, we introduce REIDIT-C and REIDIT-I in this chapter, and introduce REIDIT-M in Appendix A. With respect to the trail re-identification problem in Definition 4, each of the REIDIT algorithms are designed to produce correct results, such that no false re-identifications are made.

As in the previous chapter, let $\Gamma = \{\gamma_1, \ldots, \gamma_{|C|}\}$ and $\Omega = \{\omega_1, \ldots, \omega_{|C|}\}$ be two sets of partitioned databases as specified in Definition 8. Similarly, let M and N be the set of distinct elements from databases in Γ and Ω, respectively.

4.1 REIDIT-Complete

The first trail re-identification algorithm is named REIDIT-Complete, or REIDIT-C. The algorithm performs exact match on the trails in trail matrices. The pseudocode of REIDIT-C is provided in Algorithm 2. Details regarding how the match is performed follows. For every tuple $m \in M$, REIDIT-C determines if there exists one, and only one, element in N such that the corresponding trails are bitwise equivalent and unambiguous (i.e., $b_{m,c}$ = $b_{n,c}$ for all $c \in C$). When there is an exact and unique match, then m is re-identified to n. However, if there exists another tuple in N with a trail equivalent to m's trail, then there is an ambiguity and REIDIT-C does not predict a re-identification for m.

REIDIT-C makes re-identification predictions based on equivalence and uniqueness of

Algorithm 2 REIDIT-C(M_Γ, N_Ω, M, N)

Input: M_Γ and N_Ω, trail matrices for sets of partitioned databases; M and N, sets of corresponding tuples for M_Γ and N_Ω, respectively.

Output: R, the set of re-identifications in pair form $\langle m \in M, n \in N \rangle$.

```
1: R ← {}
2: for each m ∈ M do
3:     if there exists one and only one n ∈ N, such that m_Γ ≡ n_Ω then
4:         R ← R ∪ {⟨m, n⟩}
5:     end if
6: end for
7: return R
```

trails. When the disclosed databases are one-to-one and unreserved, these predictions can fall into one of four categories as shown in the continency table of Table 4.1. In the table, light-shaded cells correspond to predictions that REIDIT-C is capable of achieving and dark-shaded cells are impossible for REIDIT-C. In order to prove this claim, we first prove several properties regarding the trail matrices.

		REIDIT-C PREDICTION	
		Re-identification	**No Re-identification**
OBSERVED TRAILS	$m_\Gamma = n_\Omega$	Correct link	False non-link
	$m_\Gamma \neq n_\Omega$	False link	Correct non-link

Table 4.1: Classification of re-identifications made by REIDIT-Complete when databases are unreserved and one-to-one. The first and second rows of the contingency table correspond to outcomes for when the considered trails are equivalent or not, respectively. Light-shaded cells are possible outcomes and the darkened cell is an impossible outcome.

First, we prove Lemma 4, which states that when two databases are one-to-one and unreserved, then they have the same size.

Lemma 4 (One-to-One and Unreserved Size Equivalence). *If partitioned databases γ_c and ω_c are one-to-one and unreserved, there exists a bijective function $f : \gamma_c \rightarrow \omega_c$.*

PROOF. This follows directly from Definitions 9 and 10. Since the databases are unreserved, for every tuple in γ_c, there exists a corresponding tuple in ω_c, such that both tuples are derived from a single tuple in τ_c. Taken in combination with the one-to-one property, there must exist a bijective function $f : \gamma_c \rightarrow \omega_c$. □

Building upon Lemma 4, we derive Lemma 5, which states that when two trail matrices are one-to-one and unreserved, there must exist a bijection between the sets of corresponding tuples.

Lemma 5 (One-to-One and Unreserved Tuple Bijection). *If M_Γ and N_Ω are one-to-one and unreserved, then there exists a bijective function $f : M \to N$.*

PROOF. Since Lemma 4 holds for all locations $c \in C$ and we assume data is truthful, traceable, and unique (i.e., Assumptions 1 - 3 in Chapter 3), then it directly follows that every element $m \in M$ has a unique counterpart $n \in N$, and vice versa. □

With the help of Lemmas 4 and 5, we can prove the correctness of REIDIT-C in Theorem 3. By correct, we mean that REIDIT-C produces the results in the contingency table shown in Table 4.1.

Theorem 3 (REIDIT-C Correctness). *When trail matrices M_Γ and N_Ω are one-to-one and unreserved, REIDIT-C outputs no false re-identifications.*

PROOF. From Lemma 5, we know there is a bijection between M_Γ and N_Ω. As a result, for every trail $m_\Gamma \in M_\Gamma$, there exists a corresponding trail $n_\Omega \in N_\Omega$. Taken in combination with Lemma 4, it follows that for all $c \in C$, $b_{m,c} = b_{n,c}$, which implies $m_\Gamma = n_\Omega$. Therefore, for m_Γ, there must exist at least one equivalent trail in N_Ω. If there exists more than one equivalent trail, then multiple trails will be recognized and the singleton requirement will not be satisfied. When only one equivalent trail is found for m_Γ, REIDIT-C makes a correct link. Similarly, when an inequivalent trail is found, REIDIT-C makes a correct non-link for m_Γ. When more than one equivalent trail is found, REIDIT-C makes a false non-link for m_Γ. However, simultaneously with the previous decision, REIDIT-C will not make any false links for m_Γ. Therefore, we conclude REIDIT-C produces the contingency table in Table 4.1. ■

4.1.1 A Graphical Interpretation

Theorem 3 has an intuitive interpretation in the space of bipartite graphs and true loves. Since the trail matrices are unreserved, the bipartite graph must consist of a set of disjoint components, such that each component is a complete bipartite subgraph (i.e., there exists an edge between every pair of nodes) with an equal number of vertices from each class of nodes. Thus, REIDIT-C detects re-identifications from connected components with two nodes only. By definition, these nodes make up a true love, since they must exist in every maximum matching.

4.1.2 Efficiency

A quick and dirty implementation of the REIDIT-C algorithm will have O($|M| \cdot |N|$) runtime. However, more efficient versions of the algorithm can be written. In the pseudocode shown in Algorithm 3, we provide one such version called REIDIT-C-Sort. In this version, we make use of Corollary 3, which states the trail matrices are equivalent.

Corollary 3 (One-to-One and Unreserved Matrix Equivalence). *If trail matrices M_Γ and N_Ω are one-to-one and unreserved, M_Γ and N_Ω are equivalent.*

PROOF. This follows directly from Theorem 3. Every tuple $m \in M$ has a corresponding tuple $n \in N$, such that $m_\Gamma = n_\Omega$, and vice versa, so the trail matrices must be equivalent. □

This corollary enables us to write pseudocode for REIDIT-C-Sort, such that we only use row indices from one trail matrix. Specifically, since M_Γ and N_Ω are equivalent, an ascending sort of the matrices results in equivalent matrices as well. Thus, when searching for unique re-identifications, we need only to sort through one trail matrix. For instance, when we find row r is unique within the sorted trails matrix (i.e., M_Γ), then we know that row r is unique, as well as equivalent, in the other sorted trails matrix (i.e., N_Ω).

4.1.3 Complexity Analysis

We analyze computational complexity by following the REIDIT-C-Sort algorithm. First, we construct the base 10 representation of each trail. This is basically the equivalent of constructing the trails matrix, and thus requires the same number of steps as the FillTrails procedure, or O($|C| \log |M|$) steps. Second, we sort both trails matrices which, using an efficient strategy such as quicksort [65], can be done in O($|M| \log |M|$) steps. Third, we perform a linear scan of the sorted trail matrix to discover re-identifications, or O($|M|$) steps. This provides an overall complexity of O($\max(|C|, |M|) \log |M|$).

4.2 REIDIT-Incomplete

The second trail re-identification algorithm is named REIDIT-Incomplete, or REIDIT-I. Pseudocode for a basic implementation of the following description of REIDIT-I is presented in Algorithm 4. Unlike REIDIT-C's equality criteria, the REIDIT-I algorithm performs subtrail/supertrail matching on the trail matrices. When a tuple's trail is the subtrail of one, and only one, supertrail in the other trail matrix, a re-identification of the corresponding tuples occurs. Subsequently, the re-identified tuples and their trails are removed from further consideration. The removal of the re-identified trails is a crucial step. Since

Algorithm 3 REIDIT-C-Sort(M_Γ, N_Ω, M, N)

Input: See REIDIT-C.

Output: See REIDIT-C.

1: $R \leftarrow \{\}$
2: **let** V_Γ, V_Ω be vectors of length $|M|$, such that $V_\Gamma[i]$ equals the base 10 representation of the binary string concatenation of the i^{th} row in M_Γ (Similarly defined for $V_\Omega[i]$)
3: Sort the rows of M_Γ, N_Ω in ascending order according to the values in V_Γ and V_Ω, respectively
4: **let** $COUNT \leftarrow 1$
5: **for** $i \leftarrow 2$ to $|M|$ **do**
6: **if** $V_\Gamma[i] \neq V_\Gamma[i-1]$ AND $COUNT \equiv 1$ **then**
7: $R \leftarrow R \cup \{\langle TRAILDATA_\Gamma(i-1), TRAILDATA_\Omega(i-1)\rangle\}$
8: $COUNT \leftarrow 0$
9: **end if**
10: $COUNT \leftarrow COUNT + 1$
11: **end for**
12: **if** $COUNT \equiv 1$ **then** //*Perform the re-identification check for the final tuple*
13: $R \leftarrow R \cup \{\langle TRAILDATA_\Gamma(|M|), TRAILDATA_\Omega(|M|)\rangle\}$
14: **end if**
15: **return** R

trails can be mapped to multiple supertrails, failure to remove trails that are guaranteed not to be a match can prevent additional re-identifications from being discovered. Processing of the trail matrices continues until no more re-identifications can be made because one of two conditions is satisfied:

1. M_Γ has no more trails to process; or,

2. there are no re-identifications made in the current iteration.

Algorithm 4 REIDIT-I(M_Γ, N_Ω, M, N)

Input: See REIDIT-C.
Output: See REIDIT-C.

1: $R \leftarrow \emptyset$
2: **repeat**
3: $NUMFOUND \leftarrow |R|$
4: **for** $m \leftarrow 1$ to $|M|$ **do**
5: **if** there exists one and only one $n \in N$, such that $m_\Gamma \preceq n_\Omega$ **then**
6: $R \leftarrow R \cup \{\langle m, n\rangle\}$ *//remove m and n from further consideration*
7: $M \leftarrow M - \{m\}$; $N \leftarrow N - \{n\}$
8: **end if**
9: **end for**
10: **until** $NUMFOUND \equiv |R|$ *// no new matches found*
11: **return** R

Since the REIDIT-I algorithm predicts re-identifications based on subtrail/supertrail relationships, REIDIT-I is applicable when one trail matrix is reserved to the other and data is one-to-one. As a result, REIDIT-I's predictions can fall into one of the results in the light-shaded cells shown in the contingency table of Table 4.2. To prove this claim, we state several useful properties regarding the trail matrices in question.

First, in Lemma 6, we prove a simple relationship regarding the size of partitioned databases.

Lemma 6 (One-to-One and Reserved Containment). *If partitioned databases γ_c and ω_c are one-to-one and γ_c is reserved to ω_c, then $|\gamma_c| \leq |\omega_c|$.*

PROOF. This follows directly from Definitions 9 and 11. □

From Lemma 6, we derive Lemma 7 that states there exists an injection between the two trail matrices.

Lemma 7 (One-to-One and Reserved are Injective). *If trail matrices M and N are one-to-one and M_Γ is reserved to N_Ω, then there exists a one-to-one function $f : M \to N$ that is an injection.*

PROOF. Since Lemma 6 holds for all $c \in C$, it follows that every element in M has a counterpart in N, but not vice versa. □

		REIDIT-I PREDICTION	
		Re-identification	**No Re-identification**
OBSERVED TRAILS	$m_\Gamma \preceq n_\Omega$	Correct link	False non-link
	$m_\Gamma \not\preceq n_\Omega$	False link	Correct non-link

Table 4.2: Classification of re-identifications made by REIDIT-I when databases are reserved and one-to-one. The first and second rows of the contingency table correspond to outcomes for when the considered trails are equivalent or not, respectively. Light-shaded cells are possible outcomes and the darkened cell is an impossible outcome.

Theorem 4 (REIDIT-I Correctness). *If trail matrices M_Γ and N_Ω are one-to-one and M_Γ is reserved to N_Ω, then REIDIT-I outputs no false re-identifications.*

PROOF. Based on Lemma 7, it must be true that every trail in $m_\Gamma \in M_\Gamma$ has a corresponding trail $n_\Omega \in N_\Omega$, such that m and n correspond to same underlying entity. Since Lemma 6 holds true for all $c \in C$, it must be true that the corresponding trail is a supertrail of m_Γ. REIDIT-I searches for re-identifications for m in the set of supertrails of m in N_Ω. Thus, for any two trails m_Γ and n_Ω, where m_Γ is not a subtrail of n_Ω, only correct non-links will be recorded. In the event that there are multiple supertrails, no re-identification will be made in the current iteration. Yet, in the both current and subsequent iteration, the set size may be reduced. When the set size equals 1, then a correct link is made. If the set size cannot be reduced to 1, then a false non-link will be made. Moreover, trails that are subtrails of m, but the elements do not correspond to the same underlying entity will be labelled as a correct non-link. This accounts for all scenarios, and therefore REIDIT-I will never produce a false link. As a result, the REIDIT-I produces the contingency table in Figure 4.2. ■

4.2.1 Efficiency

A back-of-the-envelope calculation will reveal that REIDIT-I, as depicted in Algorithm 4, appears to be O($(|M| \cdot |N|)^2$). However, the complexity can be reduced with a more

efficient data structure. In Algorithm 5 we present an alternative structure, which we call REIDIT-I-Precomputation, or REIDIT-I-Precomp. For complexity analysis, we study this version.

The re-identification process can be made more efficient by precomputation of a matrix D, where $D[TRAILDATA_\Gamma{}^{-1}(m), TRAILDATA_\Omega{}^{-1}(n)]$ = 1 if $m_\Gamma \preceq n_\Omega$, and a vector W of length $|M|$, where each cell $W[i]$ is the rowsum of the i^{th} row of D. The precomputation step is completed in O($|M| \cdot |N|$).

The re-identification process proceeds as follows. The vector W is sequentially scanned. When $W[i] = 1$, the i^{th} row of the D matrix is scanned until a column j is found, such that $D[i, j] = 1$. These coordinates reveal a re-identification, so the corresponding tuple of the row (i.e., $TRAILDATA_\Gamma(i)$) is re-identified to the corresponding tuple of the column (i.e., $TRAILDATA_\Omega(j)$). Next, each cell $W[z]$ is subtracted by $D[z, j]$ and all cells in the j^{th} column of D are set equal to 0. This scanning process is continued until no more cells in W equal 1.

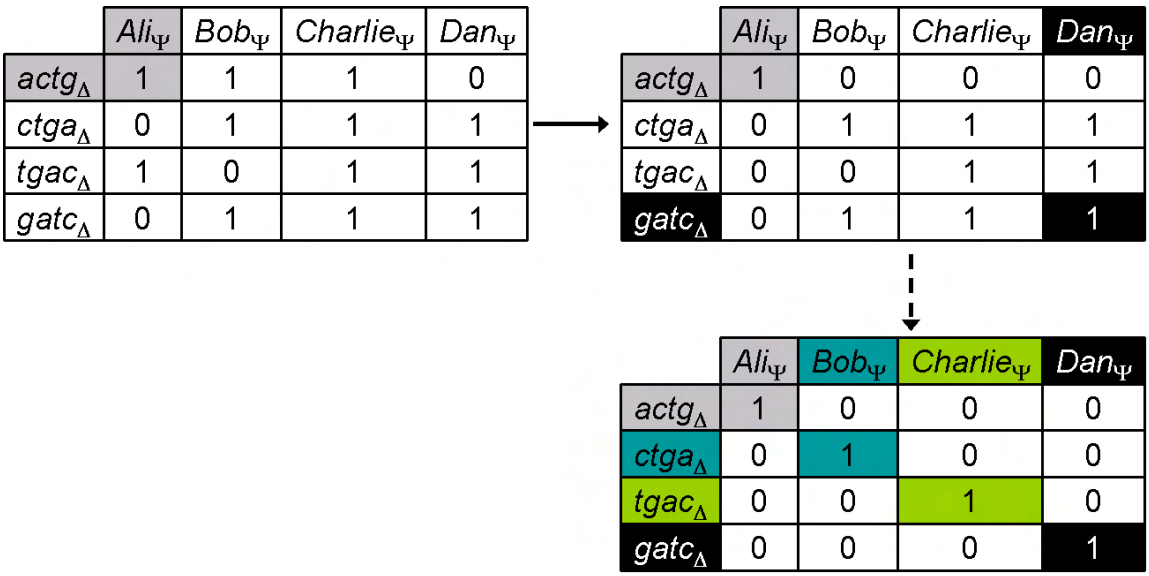


Figure 4.1: The first two matrices depict the first two iterations of the REIDIT-I algorithm, as performed by REIDIT-I-Precomp. Re-identifications are shown in cell with the same shading. The last matrix corresponds to the final predictions made upon the completion of the algorithm.

Figure 4.1 walks through the first two iterations of REIDIT-I on the example matrices depicted in Figure 3.5. In the final matrix, $D[TRAILDATA_\Delta{}^{-1}(m), TRAILDATA_\Psi{}^{-1}(n)]$ = 1 means $m_\Delta \preceq n_\Psi$ and the pair $\langle m, n \rangle$ could not be ruled out as a re-identification. If

$D[TRAILDATA_{\Delta}{}^{-1}(m), TRAILDATA_{\Psi}{}^{-1}(n)]$ = 0, then REIDIT-I could definitely rule out the pair $\langle m, n\rangle$ as a re-identification. In the first iteration, *actg* is re-identified to *Ali*. Then, *Ali* and *actg* are removed from consideration, so $D[TRAILDATA_{\Delta}{}^{-1}(gatc), TRAILDATA_{\Psi}{}^{-1}(Ali)]$ is flipped from 1 to 0. In the second iteration, *gatc* is now free to be uniquely re-identified to *Dan*.

Algorithm 5 REIDIT-I-Precomp(M_{Γ}, N_{Ω}, M, N)

Input: See REIDIT-C.
Output: See REIDIT-C.

let D be a $|M| \times |N|$ matrix, such that $D[TRAILDATA_{\Gamma}{}^{-1}(m), TRAILDATA_{\Omega}{}^{-1}(n)]$ = 1 if $m_{\Gamma} \preceq n_{\Omega}$ and 0 otherwise
let W be a $|M| \times 1$ column vector, such that $W[i] = \sum_{k=1}^{|N|} D[i,k]$
let $R \leftarrow \{\}$
repeat
  $NUMFOUND \leftarrow |R|$
  for $i \leftarrow 1$ to $|M|$ **do**
    if $W[i] \equiv 1$ **then**
      for $j \leftarrow 1$ to $|N|$ **do**
        if $D[i,j] \equiv 1$ **then**
          $R \leftarrow R \cup \langle TRAILDATA_{\Gamma}(i), TRAILDATA_{\Omega}(j)\rangle$
          for $k \leftarrow 1$ to $|M|$ **do**
            if $D[k,j] \equiv 1$ **then**
              $D[k,j] \leftarrow 0$
              $W[k] \leftarrow W[k] - 1$
            end if
          end for
        end if
      end for
    end if
  end for
until $NUMFOUND \equiv |R|$
return R

4.2.2 Complexity Analysis

In a worst-case scenario, each scan of W yields one re-identification, thus taking $|M|$ iterations. During each iteration, a sequential scan of the W vector takes place in $O(|M|)$

time. Once a cell $W[x] = 1$ is found, a scan of one row of the D matrix occurs in O($|N|$) steps. When cell $D[x, y]$ with value 1 is found, the found column in D and the W vector are updated with a scan taking O($|M|$) steps. In worst case there is only one re-identification per scan, so this process only occurs once per iteration. Thus, the total number of steps is approximately $|M|\cdot(2\cdot|M|+|N|)$, which is approximately O($|M|^2+|M|\cdot|N|$). Therefore, the order of complexity will be O(*matrix setup*) + O(*matrix scanning*). By Lemma 7, we know $|N| \geq |M|$, so $|M| \cdot |N| \geq |M|^2$, and complexity is bounded by O($|M| \cdot |N|$).

4.2.3 Special Case: $|M|$ = $|N|$

While the versions of REIDIT-I presented in Algorithms 4 and 5 are correct in the re-identifications they discover, it should be noted that a greater number of re-identifications can be discovered in the special case where $|M| = |N|$. When M is reserved to N, it is guaranteed that for each tuple $m \in M$, there exists a tuple $n \in N$ that re-identifies m. Yet, it is not true that every tuple in N can be re-identified by a tuple in M. So, if a tuple in N is found to have only one subtrail in M, this may not be a correct re-identification. This is why the trails matrix that is reserved to the other is put on the outer loop in REIDIT-I (and corresponds to the rowsum vector W in REIDIT-I-Precomp).

However, when $|M| = |N|$, there exists a bijective function and every trail in N_Ω must have a corresponding trail in M_Γ. Thus, after the re-identifications from M to N are discovered and every remaining unidentified $W[i] > 1$, there can exist columnsums of size 1 that will reveal additional correct re-identifications from N to M. In this specific case, Algorithm 6 should be inserted between lines 9 and 10 of Algorithm 4. Basically, we add a second for loop with the trail matrices exchanged. With respect to the more efficient implementation of REIDIT-I-Precomp, this corresponds to performing both columsum and rowsum scans and reductions over the matrix D.

Algorithm 6 REIDIT-I-Flip(M_Γ, N_Ω, M, N)

//Insert the following between lines 9 and 10 of Algorithm 4

```
1: if |M| ≡ |N| then
2:    for i ← 1 to |N| do
3:       if there exists one and only one m ∈ M, such that n_Ω ⪰ m_Γ then
4:          R ← R ∪ {⟨m, n⟩} //remove m and n from further consideration
5:          M ← M − {m};                N ← N − {n}
6:       end if
7:    end for
8: end if
```

Incorporation of the second loop has little influence on the overall complexity of the REIDIT-I-Precomp algorithm. Complexity will remain O($|M| \cdot |N|$), but can now be simplified to O($|M|^2$), since $|M| = |N|$.

4.2.4 A Graphical Interpretation

Akin to REIDIT-C, Theorem 4 has an intuitive interpretation in the space of bipartite graphs and true loves. First, note the precomputed matrix D is representative of a bipartite graph. Let us call this graph $H = (V_M \cup V_N, F)$. Each iteration of the REIDIT-I algorithm uncovers the only possible marriage for a pair of vertices. By definition, these marriages are true loves in a bipartite graph.

Note $|V_N| \leq |V_M|$, so trail matrices M_Γ and N_Ω do not necessarily have the same number of rows. As such, the observed trail matrices do not necessarily translate into our definition of an association relationship. However, M_Γ is a trail matrix of unambiguous values. Also, we know that N and M are one-to-one and N_Ω is reserved to M_Γ, so we can extend N_Ω to satisfy an association relationship. Basically, let P_Ψ be a trail matrix with the same rows as N_Ω plus $|M| - |N|$ rows that consist of *'s only. There exists an association relationship between M_Γ and P_Ψ.

Now, let L_{MP} be the link matrix for M_Γ and P_Ψ and let $G = (V_M \cup V_P, E)$ be the bipartite graph representative of L_{MP}. P_Ψ is an extension of N_Ω, so it must be true that $V_N \subseteq V_P$ and $F \subseteq E$. Since vertices V_M exist in both graphs H and G, and $F \subseteq E$, any true love that exists in H must also exist in G. As a result, every re-identification made by REIDIT-I is an association.

4.2.5 Extending REIDIT-I: Stable Set Reduction

The REIDIT-I algorithm, as presented is not optimal. It does not discover all true loves. For example, one way in which the algorithm could be improved is through the discovery of set covers. More specifically, let M and N be one-to-one and M_Γ be reserved to N_Ω. Now, let U be a subset of trails in M_Γ and let V be the set of trails for which U are subtrails. When $|U| = |V| \geq 1$, while we may not be able to uniquely re-identify trails in U to trails V, we know that every trail in U is correctly linked to its counterpart in V. Thus, we can remove any links between V and trails that are not in U. To illustrate this concept, we refer the reader to Figure 2.2 at the beginning of Chapter 2. This is precisely what happens when $U = \{x_4, x_5, x_6\}$. Alone, each of trail in U links to two trails in Y and thus it is not definitive which particular trail in Y is the incorrect link. Similarly, every subset of two trails in U links to three trails in Y. However, all three nodes in U link to only three nodes in Y, and we can clearly see that the edge between x_3 and y_4 must not be indicative

of a true love. As such, all edges entering into the set of six nodes $\{x_4, x_5, x_6, y_4, y_5, y_6\}$ can be dropped from the graph. Similarly, we can drop all edges that enter the set set $\{x_1, x_2, y_1, y_2\}$, which reveals that x_3 and y_3 should be re-identified to each other.

Though we can definitively determine if a particular set of trails have this type of stable set, the challenge is finding a set U in an efficient manner without enumerating all possible subsets of M_Γ. We believe that discoveries are possible, but for now we leave this open for future investigations.

4.3 REIDIT Theoretical Upper Bounds

For both REIDIT-C and REIDIT-I, the maximum number of re-identifications is dependent on the number of permutations of a binary string. Given a trail matrix M_Γ, which contains the trails of tuples over a set of data disclosing locations C, when $|M| \leq |C|$ the maximum number of trail re-identifications is bounded by $|M|$, the number of tuples. This implies that all tuples may be re-identifiable. However, when $|M| > |C|$, the maximum number of trail re-identifications is bounded by the number of locations in an exponential of the form $2^{|C|} - 1$.[1] When $|M| > 2^{|C|}$, it is impossible to re-identify all tuples.

4.4 Related Research: Linkage and Re-identification

Trail re-identification is not the first type of re-identification method to be studied. In fact, linkage and re-identification are special cases of a more general phenomena called entity resolution: the process of determining if two or more references correspond to the same entity. Previous research on entity resolution and linkage has been conducted within a variety of communities, and a more formal and detailed characterization of the problem, as well as where trail linkage fits in, can be found in [95]. Here we consider three of the areas most related to trail linkage and re-identification: record linkage, data linkage, and dimensionality reducing data mining approaches. With the quantity of research and rich theory that has already been developed in these areas, it is possible that techniques from any of these fields may be adapted for trail re-identification. Here, we provide a brief overview of related re-identification research.

Record linkage [12, 43, 105, 47, 154, 156] attempts to automate the updating of two lists, A and B, or the deduplicating of a single list. It assumes that there are two files with common variables and that there is typographical error, or alternate representation, of information (e.g., *John Smith* vs. *Jon Smith*) in the files. Initially, the process was

[1] We subtract one due to the fact that we can not re-identify a trail of all *'s.

not designed for compromising privacy, but instead to relate records of an individual for which minor corruption in one or both of the records has occurred. While the technique does relate the records of a particular subject, for the most part, record linkage has not been associated with linking de-identified data to identified data in a distributed environment. More formally, the process of record linkage corresponds to building a statistical model to classify pairs from the product space $A \times B \rightarrow \{M, U, C\}$, where M is the set of definite matches, U is the set of definite non-matches, and C is the set of pairs that need clerical review. The goal is to minimize the error in the sets M and U, while simultaneously minimizing the size of C. After the construction of the trail matrices, trail re-identification is similar to a deterministic version of record linkage that partitions the space into two classes: matches and non-matches. However, they differ in two aspects. First, the REIDIT algorithms do not return pairs for clerical review, but push clerical review pairs into the set of non-matches. Second, the REIDIT algorithms assume underlying relationship of the data, such as one-to-one, which is not the case in record linkage applications.

Data linkage differs from record linkage in several fundamental aspects. The most notable difference is that data linkage techniques are specifically designed for re-identification purposes. In addition, the attributes of the two files are not required to be the same. Data linkage is concerned with the exploitation of inferential relations between attributes of two files. A combination of the values in the attributes is utilized to estimate the uniqueness of an entity's identity in a known population [11, 58, 84, 90, 135]. Fields appearing in both de-identified and identified tables link the two, thereby relating names to de-identified data. For example, {*date of birth*, *gender*, *5-digit zip code*}, which, until recently, commonly appeared in both de-identified and identified databases, uniquely identifies 87% of the U.S. population. When a de-identified record cannot be uniquely re-identified, the process ceases for the considered record. Trail re-identification is most related to data linkage, where it extends the linkage of two tables to the simultaneous linkage of multiple tables. Like data linkage, trail re-identification first links as many tables as possible using common attributes (e.g., all tables that have a DNA sequence attribute), but also appends a set of new attributes representing the locations from which the tables were disclosed. Then, the new location attributes are used for a two table data linkage, where each table corresponds to a trail matrix.

A third method of re-identification relies on ordered weighted aggregation (OWA) operations [142, 143, 144]. This approach attempts to re-identify when there are no common attributes between releases. The procedure takes a table of records and performs dimensionality reduction by converting the data vector V of a record [v_1, v_2, ... , v_n] into a new vector W of several weighted scalars [w_1, w_2, ... , w_m], where $m < n$ and w_i is a weighted scalar for the i^{th} parameterization of the OWA operator. The goal is to create an ordering of the data using combinations of attributes. Re-identification is then achieved

by matching records from disparate tables that have similar weighted W vectors. The technique has been demonstrated to work well for the re-identification of attributes, where the data vectors are the values of an attribute for all records. The claim has been made that this technique can re-identify individual records in a table, but there is not yet any proof to substantiate this claim. In contrast, the trail re-identification problem is extremely sensitive to minute differences between trails. It is not clear that dimensionality reduction techniques can preserve the correctness of the re-identifications found by the REIDIT algorithms. Furthermore, it appears that OWAs can return false re-identifications.

4.5 Summary and Conclusions

This chapter presented several algorithms, collectively termed the RE-Identification of Data in Trails (REIDIT), that perform trail re-identification. Each of the algorithms is tailored to maximize re-identifications given particular assumptions of data multiplicity and disclosure tactics. We proved when and how the algorithms discover correct re-identifications (i.e., no false re-identifications made) and which algorithm is appropriate under which set of assumptions.

The development of the REIDIT algorithms is important to society simply because people seek safety without unnecessarily relinquishing their privacy. Clearly, the REIDIT algorithms exacerbate privacy concerns. The fact that trail re-identification can be done efficiently, as evidenced by the existence of this work, informs society and data privacy researchers of a real challenge to protecting privacy. Nonetheless, the REIDIT algorithms merely provide an architecture for re-identification. In the next chapter we investigate the degree to which real world populations are susceptible to trail re-identification.

Chapter 5

Trail Re-identification in the Real World

In this chapter we investigate the degree to which trail re-identification manifests in the real world using several cases studies. We present two case studies based on different environments of data capture and disclosure. The first case study uses health and genomic data and the second study uses online capture data. The results demonstrate that significant portions of real world populations are susceptible to trail re-identifaction. Following a presentation of these cases, we abstract the location visit distributions and simulate populations to formulate foundations for trail re-identification in the real world. Our analysis suggests that features such as location popularity and location access distributions influence the degree to which a population is re-identified.

5.1 Real World Cases and Dataset Descriptions

The case studies are based on the scenarios set forth in the first chapter of this dissertation. First, we describe databases that are representative of the aforementioned scenarios, the derived datasets for evaluation with the re-identification algorithms, as well as justification for our cases. Briefly, the first case corresponds to a healthcare environment in which people physically visit various locations for health-related functions. The second case corresponds to a virtual environment in the form of online browsing and purchasing behavior over the Internet.

5.1.1 Dataset 1: A Case Study in Hospital Visits and Genomic Databases

Consider the environment in which a set of hospitals collect genomic and identified clinical data from a set of patients. Subsequently, each hospital severs genomic data from identified

data and releases records of each type in different databases.

Though many common diseases have weak genotype-phenotype relations, this does not diminish the fact that both DNA and identified data form trails of data left behind. The trail problem only requires that DNA and identity can be tracked over multiple locations. The REIDIT algorithms are independent of specific genotype-phenotype relationships. Nonetheless, in this case study we investigate several different patient populations, each characterized by the diagnosis of a disorder that is caused by a DNA mutation. Though these disorders are more rare than common diseases, our goal is to study trail re-identification on multiple well-defined populations.

The application of the derived populations for re-identification analysis does not diminish the findings of the REIDIT algorithms. The populations we derive are neither made up nor exaggerated. It is true that if trail re-identification is evaluated with genomic data collected on individuals with more complex genetically-influenced diseases, then we could not use such well defined populations. Yet, one of the main goals of biomedical research is to learn and formally characterize complex genotype-phenotype relationships. In the future, trail re-identification risk for more common diseases will become similar to our findings with rare diseases.

Motivation for Study

There exist many factors that influence where an individual leaves behind personal health-related data. For example, many hospitals and physicians have referral programs, such that there is non-trivial correlation between the visits of several hospitals or healthcare providers. Previous research has demonstrated referral and location correlation is partially attributed to a patient's clinical features, as well as demographics [50], such as age, gender, and monetary or Medicaid status [108]. It has also been shown that patients tend to visit hospitals within close proximity to their residence [53, 54] and within densely populated areas [67]. These studies confirm expected social phenomena. For instance, certain hospitals offer specialized care or treatment for particular diseases and thus their patient population should correlate more with a patient's clinical status as opposed to geographic distance from a provider. In contrast, for a more diverse patient population, a hospital that is situated in the middle of a city will be visited by more patients than a hospital in a rural setting. Given these, and additional idiosyncrasies of the real world, we investigate the susceptibility of populations to trail re-identification with real healthcare-derived data.

Dataset Construction

The National Association of Health Data Organizations (NAHDO) reported that as of the year 2000, 44 of 50 states had legislative mandates to gather hospital-level data on each

patient visit [104]. For our experiments, we selected publicly available hospital discharge databases from the State of Illinois for the years 1990-1997. In these databases, there are approximately 1.3 million discharges per year, with compliance for greater than 99% of all hospital discharges in the state [132]. Typically, discharge databases, including those from Illinois, are made up of both demographic and clinical attributes. Patient demographics, hospital identity, diagnosis codes, and procedure codes are among the attributes stored with each hospital visit. In the Illinois databases, the demographic attributes include date of birth (DOB), gender (SEX), five-digit zip code of residence (ZIP), as well as an identifier for the hospital visited (HID). In addition, clinical information per patient visit includes a set of one to nine International Classification of Disease - 9th Revision (ICD-9) codes [159][1].

For our experiments, we assume demographic attributes are linkable to explicitly identifying information, and thus are considered identifiable attributes. This is not an unreasonable assumption to make, especially considering that patients can be uniquely re-identified via direct linkage on demographics to publicly available identified data [131, 135, 139]. For instance, in prior research performed by Latanya Sweeney, it was shown that 80-100% of discharge database entries can be accurately re-identified by linkage with publicly available voter registration lists [135], which includes the personal name (first and last) of registrations, over the combination of values in demographic attributes $\{$*DOB*, *SEX*, *ZIP*$\}$ that discharge and population registration databases have in common. In fact, the set of patients that we extracted for trail re-identification analysis are 98-100% identifiable.

We constructed the set of hospitals visited by each specific patient (i.e., the "1's" of the trail) from the Illinois databases via the procedure outlined in Figure 5.1. First, we queried the databases for tuples that contained an ICD-9 diagnosis code of interest (described in the following section), say $code_i$. From this result set, we extracted the set of distinct values over the demographic attributes $\{$*DOB*, *SEX*, *ZIP*$\}$. Next, we requeried the databases for all tuples that matched on the set of demographic attributes - regardless of whether or not $code_i$ was a value in the tuple. From this result set we grouped together tuples, based on census demographics for $\{$*Age*, *SEX*, *ZIP*$\}$, such that the tuples in each group were highly likely to relate to the same person [131]. Recall, the REIDIT algorithms assume that each patient has a unique combination of values [*dob*, *gender*, *zip*]. A ZIP chart is available that reports the identifiability of each zip code, so that likelihood measures could be assigned. We used such information with respect to the real-world discharge databases and found that the identifiability of values from the attributes $\{$*DOB*, *SEX*, *ZIP*$\}$ for our extracted populations was 98-100% unique.[2]

[1]The complete list of ICD-9 codes is available for download from the National Center for Health Statistics at `http://www.cdc.gov/nchs/about/otheract/icd9/abticd9.htm`.

[2]The identifiability of this combination was evaluated by determining the uniqueness of the quasi-

Though the demographics for almost every individual is identifiable, it does not imply that trails are re-identifiable. Rather, it demonstrates that the extracted populations' demographics can be used to construct trails from the discharge databases with confidence that each trail corresponds to a different person. The re-identifiability of the trails is evaluated by the REIDIT algorithms.

What is $code_i$ and Why Use It to Separate Populations?

Many diseases are known to have genetic influence, and of such diseases, a growing number are found to depend on interactions of multiple gene products in addition to environmental influence, such as certain cancers. Mutations in specific genes may not necessarily cause a certain disease, but instead can raise (or lower) the risk of developing the disease. For instance, mutations in the BRCT domain of the BRCA1 gene increase the risk of developing breast cancer [28], and allelic variants of the APOE4 gene increase the risk of acquiring late onset Alzheimer's disease [126]. Moreover, there exists a non-trivial sized group of diseases that are caused by the mutation of a single gene, or Mendelian trait diseases [52]. This set of diseases span a wide range of biological processes involving cancer, the immune system, metabolism, the nervous system, cellular signaling, and molecular transportation [49].

As a result of the growing number of characterized disease genes, we can extract relationships between single gene detectable diseases and diagnosis codes that are provided in publicly available hospital discharge databases. Through the use of publicly available resources, we searched for diseases with a deterministic genetic basis. Broadly speaking, we searched for Mendelian trait diseases in the ICD-9 diagnosis codes of the Illinois discharge databases. Details of this process are provided in Section B.2.2 (*Genotype-Phenotype Inference*) of Appendix B. From our searches, we discovered over 30 diseases, with distinct annotations, in the ICD-9 diagnosis codes that can be brought about by a mutation in a single gene. A partial listing of the diagnoses that can be extracted from the hospital discharge databases is presented in Table B.2.2 of Appendix B.

Mutation in the genes listed in Table B.2.2 are directly responsible for the manifestation of the clinical phenotype associated with the corresponding ICD-9 codes. However, it is not necessarily a bidirectional relationship. In other words, some of the diseases that are listed can be brought about by multiple genetic factors, some of which are unknown.

identifier value combinations in the population. For example, consider the quasi-identifying value $\{4/6/75, M, 61010\}$. If there exists only one living person with this combination during the period the data was collected, then this individual could be uniquely identified. However, if there existed more than one, then attempts to determine the identity of the individual, using solely the quasi-identifier, would yield ambiguous results.

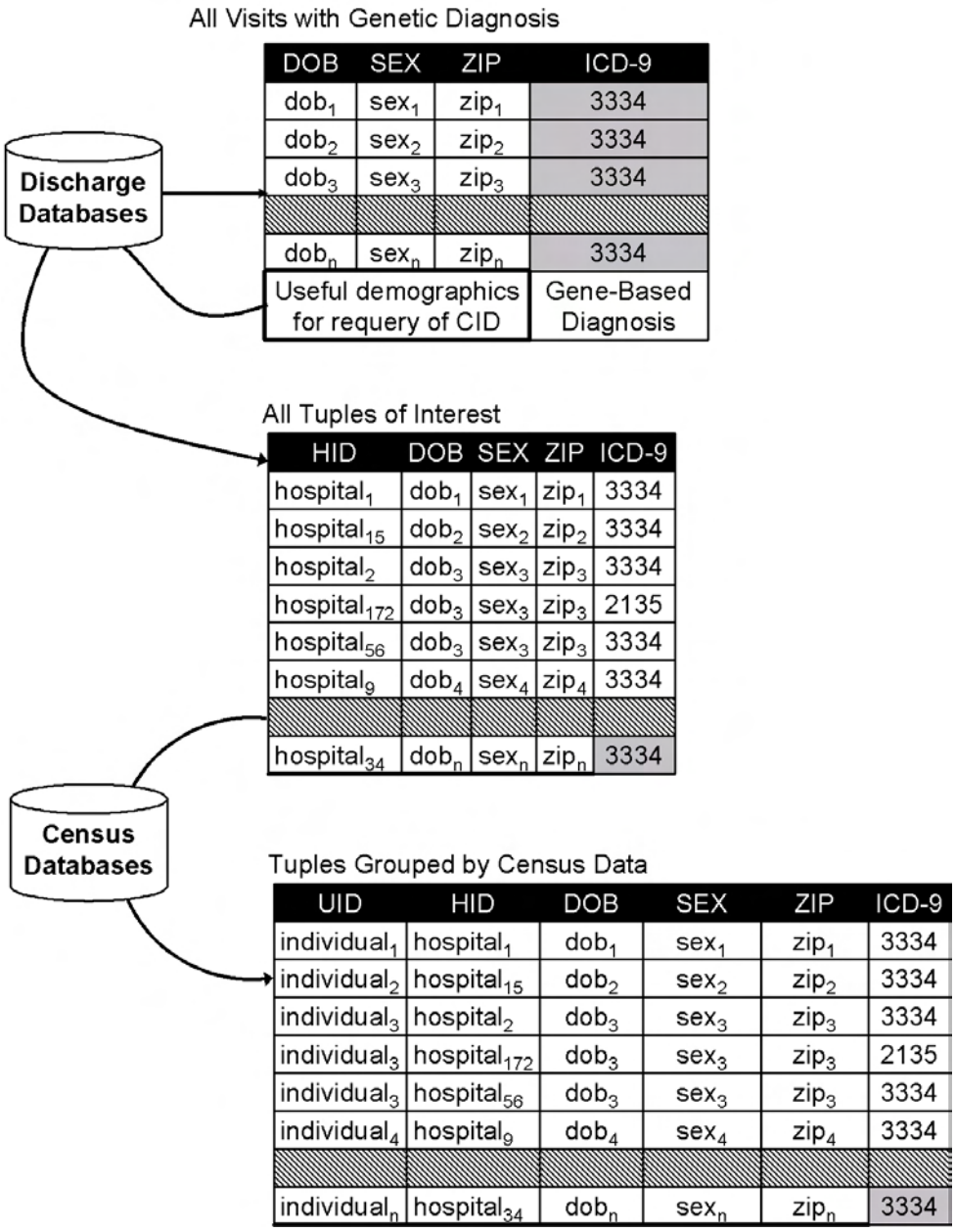


Figure 5.1: Extraction and construction of trails from hospital discharge databases.

Other diseases may be brought about by either a mutated gene or by environmental influence. Therefore, for traceability purposes, we selected a subset of the diseases to continue

with an analysis of trail re-identification. Specifically, we chose disease that are known to have a bidirectional relationship between genotype and clinical phenotype. We selected eight gene-based diseases for further analysis. They correspond to the following disorders: cystic fibrosis (CF), Friedrich's Ataxia (FA), hereditary hemorrhagic teleangiectasia (HT), Huntington's disease (HD), phenylketonuria (PK), Refsum's disease (RD), sickle cell anemia (SC), and tuberous sclerosis (TS). Furthermore, we derived an additional sixteen datasets by splitting each of the eight populations by gender. These datasets provide additional examples of how trail re-identification is influenced when there exists a relationship between the identified and de-identified database attributes. The choice of gender for dataset blocking corresponds to the fact that there exists a correlation between certain DNA sequences and an individual's gender. For illustration, if DNA sequences are derived from X or Y chromosomes, then the amelogenin (Aml) and sex-determining region Y (Sry) genes may be present, which can be used to categorize DNA specimens as male or female [18, 111, 117].

Some summary statistics of the twenty-four datasets are provided in Table 5.1.

5.1.2 Dataset 2: A Case Study in Internet Browsing and Purchasing Behavior

In an online environment, the data sharing situation we study corresponds to a set of website that collect the IP addresses and identified information of webusers. After collection, each website partitions IP addresses from identified data and releases each data type in separate databases.

Motivation for Study

In comparison to the healthcare environment, the Internet does not impose as many geographical constraints due, in part, to the lack of physical interaction between providers and users. However, websites on the Internet are not devoid of semantics. For example, sets of websites often compete against each other for users' attention and/or money (e.g., Amazon.com versus BarnesAndNoble.com). Recently, marketing research for e-commerce suggests there exist trends in the way that Internet users access websites. More specifically, it has been shown that there exist patterns in the pages webusers access before making purchases from online websites [16, 70, 101, 102, 103]. With respect to a single website, the pattern of pages accessed, also known as the intrasite clickstream, helps managers determine if the user will make a purchase, which page(s) to serve to the user in order to increase the probability of a purchase, and, given the items the user has already purchased, which additional offers should be served to entice additional purchases. For

Disease	Gender	# of Patients	# of Hospitals	# Hospitals Visited Per Patient			
				Min	Max	Average (St. Dev.)	Median
CF		1149	174	1	8	1.80 (1.16)	1
	Female	557	142	1	8	1.86 (1.20)	1
	Male	592	150	1	7	1.75 (1.11)	1
FA		129	105	1	5	1.70 (1.07)	2
	Female	60	68	1	3	1.67 (0.83)	2
	Male	69	72	1	8	1.72 (1.24)	5
HD		419	172	1	17	1.80 (1.41)	1
	Female	236	149	1	8	1.73 (1.20)	2
	Male	183	127	1	17	1.87 (1.63)	1
HT		429	159	1	8	1.79 (1.11)	1
	Female	244	140	1	8	1.75 (1.12)	1
	Male	185	114	1	6	1.83 (1.09)	1
PK		77	57	1	5	1.59 (0.99)	2
	Female	52	48	1	5	1.71 (1.13)	1
	Male	25	25	1	3	1.36 (0.57)	2
RD		4	8	2	2	2 (0)	2
	Female	2	4	2	2	2 (0)	2
	Male	2	4	2	2	2 (0)	2
SC		7730	207	1	34	2.38 (1.82)	4
	Female	4175	189	1	34	2.52 (1.87)	4
	Male	3555	191	1	24	2.20 (1.74)	6
TS		220	119	1	10	2.07 (1.53)	2
	Female	97	88	1	8	2.36 (1.66)	2
	Male	123	87	1	10	1.84 (1.38)	2

Table 5.1: Summary statistics of the derived populations from the discharge databases.

instance, in research conducted by Montgomery et al. [103], it was empirically demonstrated that users of an online bookstore could be correctly categorized as a purchaser with approximately forty percent accuracy (a significant level in marketing standards).

More recently, researchers have investigated intersite clickstreams, where they have discovered associations between the website domains visited by users [109]. For instance, in an online population studied by Park et. al. [109] several non-random associations that were observed include a) approximately one-quarter of users visited two major online

booksellers and b) almost twenty percent visited two major music distribution sites.

To analyze some of the intricacies of real world data with respect to the REIDIT algorithms, we study a real world dataset consisting of household Internet usage behavior.

Dataset Construction

The dataset was compiled by the Homenet project [77] at Carnegie Mellon University, who provide families in the Pittsburgh area with Internet service in exchange for the monitoring and recording of the families' online services and transactions. For our research, we use URL access data collected over a two-month period that included 86 households and 144 individuals. The Homenet dataset consists of 56, 9, 14, 5, 1, and 1 household(s) of 1, 2, 3, 4, 5, and 6 people respectively. Each household, as well as each individual within the household, was issued a unique login and password for fine-grained monitoring. Overall, this population accessed approximately 66,000 distinct webpages across 5000 distinct top-level domains. We call this the "general" population. With respect to domains some summary statistics follow, and are shown in Table 5.2, the mean number of domains visited per individual user was around 50, with a standard deviation of 100 domains. The number of domains per household was almost 25% greater at approximately 80 with a standard deviation of 80 domains.

From the Homenet dataset, we extracted purchase data and weblogs for websites accessed by this population to construct a more specific dataset. The URL data was manually labeled as "purchase made" or "purchase not made" as inferred from the accessed page. For example, a purchase confirmation URL at Greyhound.com was labeled as a purchase, while the frontpage of the website was labeled as not being a purchase. It was determined that purchases were made at 24 distinct domains, including Amazon.com, Ticketmaster.com, and Hotwire.com. We call this the "purchasing" population. Summary data is shown in Table 5.2. The purchases were made by 30 individuals distributed across 26 households. There were 23 households with a single purchasing individual, 2 households with 2 individuals, and 1 household with 3 individuals (i.e., a total of 30 individuals). In general, websites where purchases were made tended to be more popular (i.e., receive more users) than the average website in the complete dataset. Out of the 24 domains, there was a mean of around 6 domains visited per user, with a standard deviation of about 3. In comparison, the number of websites each user made purchases at was smaller, with a mean of approximately 2 locations visited per user and standard deviation of approximately 1.

Preliminary Validation: Relating Homenet to General Populations

In previous studies, it was observed that the popularity of webpages within a particular website varies widely with high skew [14]. The popularity adheres to a power law, namely

Population	# Users	Domains	# Domains Visited Per User			
			Min	Max	Average (St. Dev.)	Median
General Homenet Population						
Individuals	144	4945	1	613	51.92 (114.83)	25
Households	86	4945	1	714	80.93 (83.64)	46.5
Purchasing Homenet Population						
Individuals	30	24	1	2	1.20 (0.41)	1
Households	26	24	1	3	1.20 (0.55)	1

Table 5.2: Summary statistics of the derived populations from the Homenet databases.

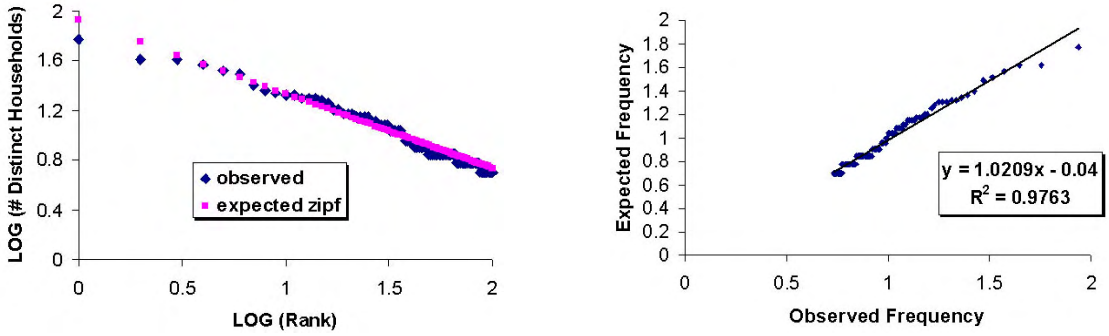
the Zipf distribution. In this type of distribution, given a set of pages and a set of users that visit the pages, a two dimensional log-log plot of page rank (in terms of number of visitors) vs. the actual number of visitors will follow an inverse linear trend. This finding is validated in a number of environments, including traffic over websites and, subsequently, has been employed for the design of more efficient search engines [15].

In a Zipf distribution, the probability of occurrence of an event, f_i, is inversely proportional to the event's rank (as determined by its frequency) r_i:

$$X * f_i = {r_i}^{-\alpha}, \tag{5.1}$$

where α is a constant and X is the number of observations. With respect to the re-identification studies of this research, consider an environment where C is a set of locations and X is a population of subjects visiting those locations. The probability that any particular entity visits location $c \in C$ is equal to ${r_c}^{-\alpha}$, where r_c is the rank of c's popularity. Subsequently, the number of entities that visit c is $X * {r_c}^{-\alpha}$. Moreover, the α constant is a term that controls the magnitude of skew, such that when $\alpha = 1$ the distribution is a true Zipf with maximum skew and as α tents toward 0 the Zipf distribution is in a generalized form with decreasing skew. As a result, the log-log plot of "number of visitors" to "location rank" is linear, while the coefficient functions as a dampening factor on the slope of the plotted curve.

For our studies, we consider f_i to be the probability that an individual visits website i and X as the set of households in the Homenet dataset. To determine if the Homenet dataset exhibits this property of a real world environment, we analyzed the traffic at each domain with respect to the number of distinct visitors. The log-log plot of the observed distribution is shown in Figure 5.2(a). Similarly, in Figure 5.2(b), we show the expected frequencies for a Zipf distribution with an α of 0.6. A linear fit of observed frequencies to



(a) Observed

(b) Correlation

Figure 5.2: *a)* Log-log (generalized Zipf) distribution of Homenet households accessing the 100 most popular websites. *b)* Correlation plot of observed and expected log(frequency) for Zipf distribution with $\alpha = 0.6$.

expected frequencies yields a correlation coefficient of approximately 0.98, and confirms that the high skew trend does hold in the Homenet dataset.

5.2 REIDIT-C Re-identifiability

For the following experiments, we assume data is one-to-one and unreserved. As proved in the previous chapter, under these assumptions, all re-identifications returned by REIDIT-C are a correct match.

5.2.1 Batch DNA Re-identification

With respect to hospital visit data, we make the following assumption. If a patient's discharge profile specifies that the patient made a visit to a particular hospital, then both clinical (including demographic) and de-identified DNA data about the patient are partitioned and released by the hospital.

REIDIT-C was evaluated on the 24 population-based datasets (i.e., 8 populations devoid of gender and 16 populations that include gender). The results are summarized in Table 5.3.

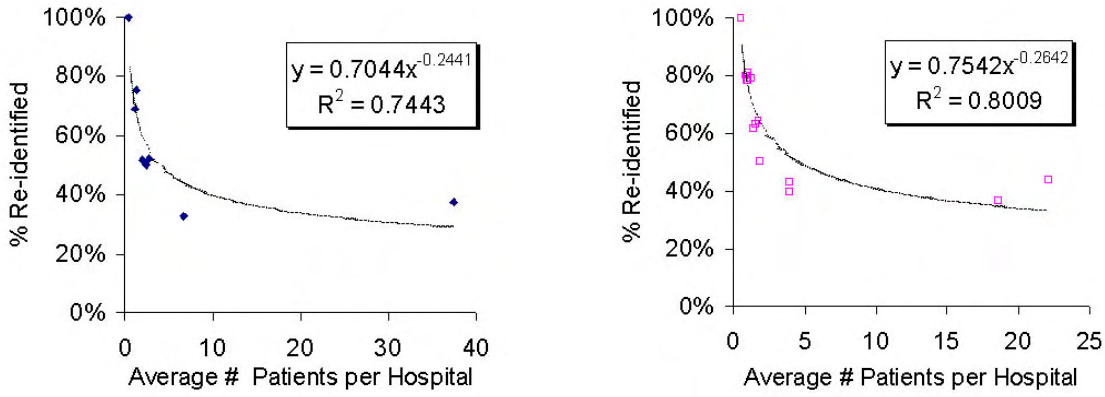
Since the number of patients for each population is less than two to the number of total hospitals visited, the maximum number of re-identifications in theory is the number of patients. However, the observed number of re-identifications only achieves this maximum

Disease	Gender	# Patients / # Hospitals	% Re-identified via REIDIT-C
CF		6.60	32.90%
	Female	3.92	43.09%
	Male	3.95	39.36%
FA		1.23	68.99%
	Female	0.88	80.00%
	Male	0.96	78.26%
HD		2.48	50.00%
	Female	1.26	79.14%
	Male	1.87	50.63%
HT		2.70	52.21%
	Female	1.74	64.34%
	Male	1.62	63.24%
PK		1.35	75.32%
	Female	1.08	80.77%
	Male	1.00	80.00%
RD		0.50	100.00%
	Female	0.50	100.00%
	Male	0.50	100.00%
SC		37.34	37.34%
	Female	22.09	43.76%
	Male	18.64	36.51%
TS		2.10	51.60%
	Female	1.10	78.35%
	Male	1.41	61.78%

Table 5.3: Summary of the percentage of actual re-identifications made by REIDIT-C for genetic disease patient populations.

for the RD population, where there is only one patient with the disease at each of the hospitals considered. For the remaining populations, it appears that healthcare factors have a profound effect on the uniqueness of trails. A quick inspection reveals that the re-identifiability of these populations is related to the average number of patients visiting a hospital. This effect is graphically depicted in Figure 5.3. From the trendline depicted, it is apparent that as the number of people per hospital increases, the more difficult it is for re-identifications to occur. This phenomenon is due, in part, to the fact that an increase

in population size, over a fixed set of locations, increases the probability that multiple patients will have the same trail. Thus, the average number of patients per hospital is a gross measure of re-identification. There are additional features about the environment that affect the re-identifiability of a population, some of which we explore below.



(a) Genderless datasets

(b) Gender-blocked datasets

Figure 5.3: REIDIT-C re-identification of genetic disease populations as a function of the number of patients per hospital.

5.2.2 Batch Internet Re-identification

In this experiment, we explore re-identification in the purchasing Homenet population. Two different scenarios were explored: re-identification to a) individual users and b) households. For re-identifications to individual users, the attributes released with the de-identified databases were {*website domain, purchaser IP address*}. The attributes released with the identified databases were {*website domain, name, residential address*} for each targeted website location. For re-identifications to households, the attributes released with the de-identified databases were {*website domain, household IP address*} and the attributes released with the identified databases were {*website domain, street address*} for each targeted website location.

In the first scenario, REIDIT-C achieved re-identification of 16 IP addresses to household, approximately 62% of households. In the second scenario, 22 IP addresses were re-identified to individual users, approximately 66% of individual users.

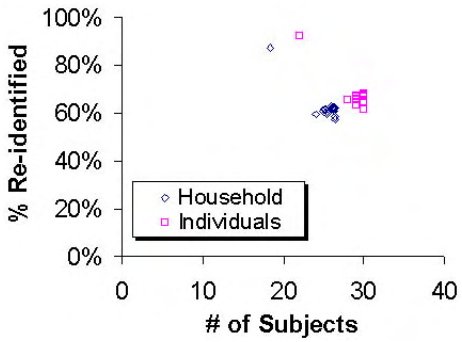


Figure 5.4: Sensitivity of REIDIT-C re-identification to the removal of one domain's dataset removed in the Homenet "purchase" dataset. The values are slightly jittered for visual inspection.

Given the small size of the "purchaser" dataset, we conducted further investigations on the sensitivity of REIDIT-C to certain domains. Specifically REIDIT-C re-identification was repeated 24 times, each time dropping information from a different domain. The results are shown in Figure 5.4. For the most part, it appears re-identifications using REIDIT-C were minimally affected. In this graph "percent re-identified" corresponds to the percent of the population remaining after a domain was removed. The observed outlier corresponds to the domain "Ticketmaster.com", which was accessed by many purchasers but played a minimal role in revealing re-identification. Removal of this domain led to an improvement in re-identifiability by approximately 25%. The main reason is that removal of "Ticketmaster.com" dropped the individuals/households that made purchases only at this domain.

5.2.3 Sensitivity Analysis

The belief that each location will collect and release data is not always feasible. In the previous experiments, we touched upon the sensitivity of a population to re-identification as a function of the locations that were releasing data. For example, in Figure 5.3, the number of patients per location affects re-identifiability. Moreover, in Figure 5.4 we demonstrated that specific locations can influence re-identification. Yet, these experiments do not provide the insight necessary to indicate the properties of locations that have an effect on trail re-identification.

In the following experiments, we consider a more fine-grained perspective by analyzing how particular locations and sets of locations can influence re-identifiability. We investigate this concept by performing a sensitivity analysis of a location's popularity on

re-identification susceptibility. For illustrative purposes, we continue with two of the genetic datasets, specifically cystic fibrosis (CF) and phenylketonuria (PK). By doing so, we compare a population in which the number of subjects per location is relatively large (CF - approximately 6.60), to a population in which the average is closer to a single subject per location (PK - approximately 1.35).

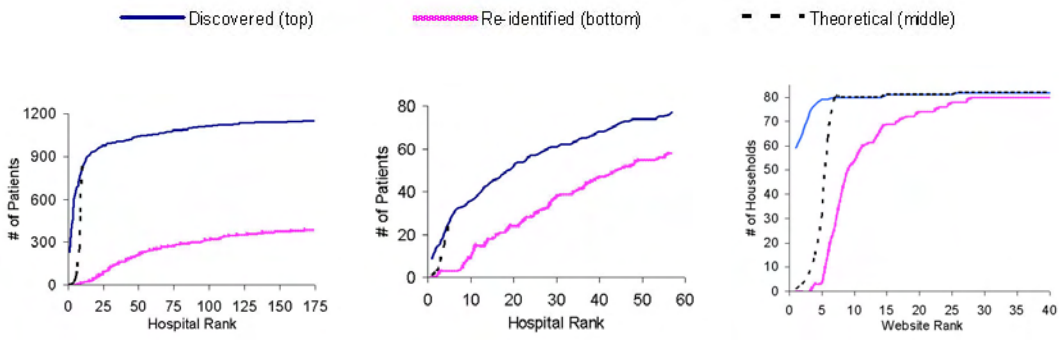
Furthermore, we include the Homenet dataset in order to study a population in which the number of subjects per location is extremely small. It would be ideal to continue with the labeled purchasing dataset, but this dataset is limited in its size. So, to perform a more in-depth analysis of web browsing behavior, we continue with the general population of 86 households in the dataset and make the following simplifying assumption: when an individual visits a website, both their IP address and their identifying information is left behind. In this population, the number of individuals/households per location averages much lower than 1.

First, we ask the question "To what extent do heavily visited/accessed locations contribute to re-identification?" And if these locations contribute, how many locations need to supply information for significant quantities of re-identification to be achieved? To answer this question, we model re-identification as a function of increasingly less popular websites. In the datasets, we ranked the popularity of each location by the number of distinct subjects visiting the location. A total rank ordering of the locations was achieved by randomly ordering locations with the same number of subjects. Given a set of locations from highest rank to a low rank location, called $rank$, we measured the re-identifiability of the trails that were discovered over the set of locations ranked 1 to $rank$.[3] For both CF and PK, the rate of trail discovery is logarithmic as can be seen in Figures 5.5(a) and 5.5(b). The r^2 correlation coefficients for fit curves are 0.92 and 0.97, respectively. However, while the rate of trail re-identification for CF is logarithmic, the rate for PK is linear. It appears that this is an artifact of the slope in the logarithmic discovery rate. The slope of trail discovery for CF is much greater than for PK. This implies that most individuals visited the more popular locations for CF, while for PK patients are more dispersed in hospitals.

In Figure 5.5(c), we present the results for the Homenet dataset. In this plot, the top line represents the number of households with discovered trails made up of information from the considered websites. We find that the actual number of households re-identified is slow to approach the theoretical number of possible re-identifications. Though this suggests users have similar visit patterns over highly travelled websites; as the number of websites that contribute to a trail increases, the number of re-identifications also increases. By around 20-25 websites, almost all re-identifications are discovered.

Popular locations capture information on most of the population, but we need to consider the other end of the spectrum. Re-identification as a function of decreasing rank

[3] A trail is considered to be discovered when at contains least one "1" value.

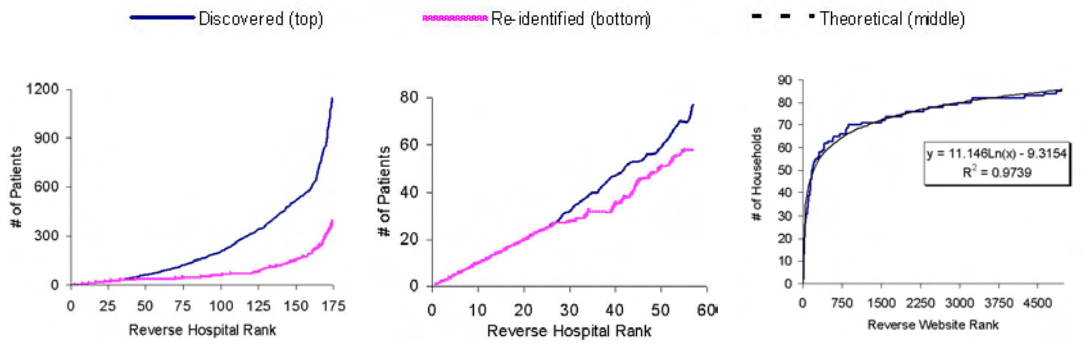


(a) CF (b) PK (c) Homenet

Figure 5.5: REIDIT-C re-identification of trails with number of locations increasing from most-visited to least-visited. The "theoretical" line corresponds to the theoretical maximum number of re-identifications possible.

obscures the effect that less popular locations have on re-identification. One would expect that the incorporation of less popular locations would make re-identification more likely and that more popular locations would make re-identification more difficult. To evaluate this hypothesis, we added locations in reverse rank, and measured the re-identifiability of the discovered trails constructed from the contributing locations. The results are depicted in Figure 5.6. In these plots, the theoretical rate of re-identification follows the line corresponding to the number of discovered samples, and thus it is obscured. We find that for the first quarter of reverse rank hospitals, almost all patients in the population are re-identified. This is due to the fact that for most of these hospitals, the number of patient trails observed and the number of re-identifications increase linearly with slope approximately to 1. This suggests that at these locations, usually only one (or very few) patient(s) existed at the hospital with the disorder. Thus, the first part of our hypothesis holds true. After the first quarter of locations, the re-identification rate for PK remains linear, with a slightly lower rate than the rate of trail discovery. However, the trail discovery rate for PK tends toward an exponential, and subsequently, after a delay, so too does the CF trail re-identification rate. This is mainly due to the fact that as the number of people per location increases, the ability to distinguish a larger number of trails increases as well.

In terms of the re-identifiability in the Homenet dataset, the Zipf distribution suggests that websites accessed less often should be more useful than others for re-identification using the REIDIT algorithms. Additionally, since the number of websites is much greater than the number of hospitals, the ability for a less popular location to contribute to re-identification is greater. We confirm this belief, and depict this effect, in Figure 5.6(c).



(a) CF (b) PK (c) Homenet

Figure 5.6: REIDIT-C re-identification of trails with number of locations increasing from least-visited to most-visited.

In general, we observe an interesting trend in Figure 5.6. With respect to the average number of subjects to location, we consider the continuum from 0 to infinity. As the ratio tends towards infinity, such as in the CF dataset shown in Figure 5.6(a), we observe an increasingly exponential rate of re-identification. However, as the ratio tends toward 0, as observed in the Homenet dataset in Figure 5.6(c), we approach a logarithmic rate of re-identification. Furthermore, the PK dataset helps to support this trend. Note, here the ratio is approximately 1, and we observe an approximately linear rate of re-identification. From this observation, we conjecture that in a skewed distribution, if there is an equal probability for a new location to enter the set of locations, then there is a greater probability the location will be a less popular location. Thus, as the number of locations increases, the tail of the distribution will grow and re-identification will occur at a faster rate. We investigate this conjecture below.

5.3 REIDIT-I Re-identifiability

For analysis of the REIDIT-I algorithm, we continue with the CF and Homenet datasets. For the CF dataset, we assume DNA databases are reserved to identified clinical databases. From the other perspective, for the Homenet dataset, we assume identified online purchasing databases are reserved to IP address databases. As shown in the previous chapter, all re-identifications returned by REIDIT-I are correct matches.

The CF dataset does not have a natural reserved scenario; however, the Homenet dataset does provide such an opportunity for study. Thus, we first subjected the Homenet

dataset to REIDIT-I, such that the trail matrix constructed from the purchase databases, which contains identified information, was re-identified to the trail matrix constructed from the general databases.

5.3.1 Batch Internet Analysis

Using the Homenet datasets described above, we assume websites reported IP addresses for all visitors to their site, regardless of whether they made a purchase. Again, we explored two scenarios, trail re-identifications to a) individual users and b) households. For re-identifications to individual users, the attributes released with the de-identified databases were {*website*, *individual IP address*} and the attributes released with the identified databases remained {*website*, *name*, *address*} for each targeted location. The de-identified trail matrix for this scenario (individual user IP addresses) contained 53 rows and the identified track had 30 rows (i.e purchaser names). The number of locations remained at 24.

In the second scenario, or re-identification of IP addresses to households, the attributes released with the de-identified databases were {*website*, *household IP address*} and the attributes released with the identified databases were {*website*, *street address*} for each targeted website. The de-identified trail matrix for this scenario (i.e., household IP addresses) had 39 rows and the identified trail matrix had 26 rows (i.e., household addresses). Again, the number of locations remained 24.

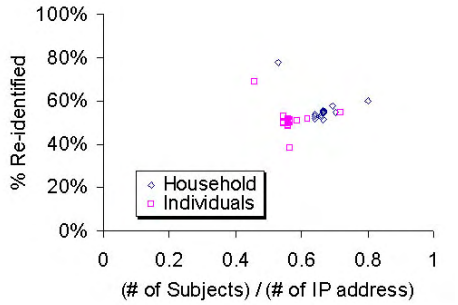


Figure 5.7: Sensitivity of REIDIT-I re-identification to the removal of one domain's dataset removed. The values are slightly jittered for visual inspection.

REIDIT-I re-identified 9 IP addresses to households (approximately 40%) and 15 IP addresses to individuals (approximately 50%). Sensitivity of REIDIT-I to single locations was analyzed in the same leave one out manner as performed for REIDIT-C. The results are depicted in Figure 5.7. One location, Amazon.com, had a significant effect on the ability to re-identify individuals, in that removal of this location decreased the size of the considered population and increased the ability to re-identify IP addresses by about 25%.

5.3.2 Sensitivity Analysis

The unreserved trail matrices used in the evaluation of REIDIT-C were used to generate reserved trail matrices. We utilize a simple model of how reserved databases come to exist. Basically, we investigate a scenario where each location's database is missing information with respect to the reserved to database using the same probability $miss$. We varied the probability of information missing and attempted re-identification with REIDIT-I. Plots of the results for $miss$ equal to 0.1, 0.5, and 0.9 are shown in Figure 5.8. Each point of a graph depicts the average result for 100 experiments of missing information.

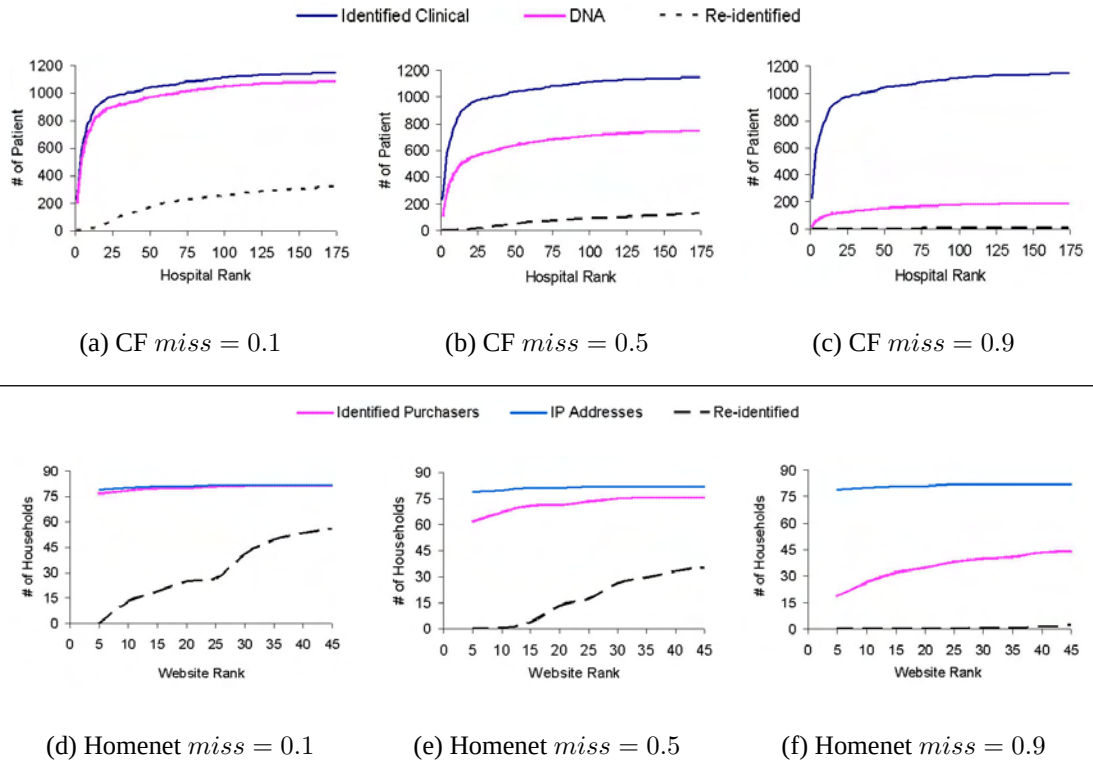
As the probability of missing information increases, the probability that an individual will not show up at all (i.e., no unambiguous values in the trail) in the population of trails increases. Thus, in the graphs we show three lines. For the CF dataset, the topmost line represents the number of discovered identified clinical data trails for a given set of hospitals. The middle line represents the average number of discovered genomic data trails. And the lowest line represents the average number of genomic data trails that were re-identified. In contrast, for the Homenet dataset, the topmost line represents the number of discovered IP address trails for a given set of Websites. The middle line represents the average number of discovered identifying information trails. And the bottom line represents the number of IP addresses that were re-identified.

As expected, we find that as the amount of information withheld increases, the number of releasing locations necessary to perform re-identification increases as well. This is due to the fact that as additional information is withheld, trails become less informative. Nonetheless, even though trails become less informative, there remains a significant disposition toward re-identification. This is observable even after 50% of a trail is rendered missing. We find that there is an inverse relationship between the slope of re-identification (as a function of location rank) and the amount of information withheld.

5.4 REIDIT Scalability

In the previous chapter, we addressed the theoretical scalability of the REIDIT algorithms. Here we examine how the REIDIT algorithms scale to very large populations using experimental evidence. To conduct these experiments we generated synthetic datasets with distributions based on those found in the Homenet database. Trails were simulated based on the probability that an individual visited a location. Let $\Psi = \{\psi_1, \ldots, \psi_{|C|}\}$ and $\Delta = \{\delta_1, \ldots, \delta_{|C|}\}$ be the set of identified purchasing lists and de-identified IP address databases disclosed by locations $C = \{c_1, \ldots, c_{|C|}\}$. Also, let X and Y be the distinct union of tuples in Ψ and Δ, respectively.

For simulated IP address trails the probability the i^{th} value in the trail equals 1 is set to a



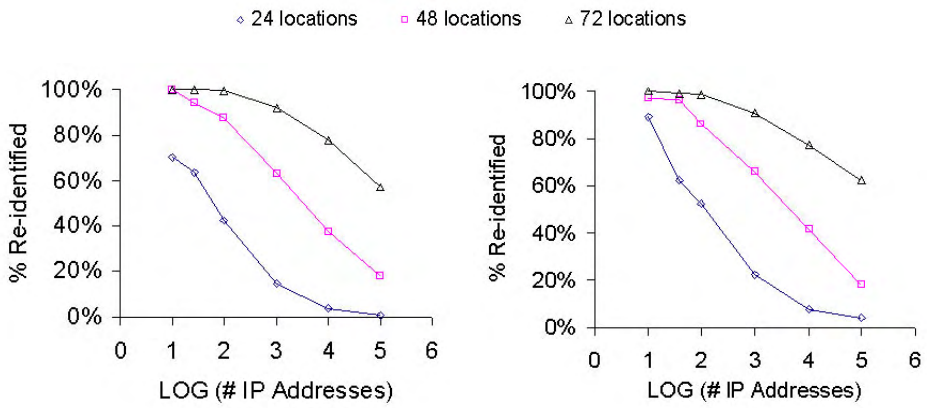
(a) CF $miss = 0.1$ (b) CF $miss = 0.5$ (c) CF $miss = 0.9$

(d) Homenet $miss = 0.1$ (e) Homenet $miss = 0.5$ (f) Homenet $miss = 0.9$

Figure 5.8: REIDIT-I re-identification of trails as an increasing amount of identifying information is withheld from the release. Information is removed from the release with a left) 0.1, center) 0.5, and right) 0.9 probability.

simple Bernoulli probability $\frac{|\psi_i|}{|X|}$ (i.e., ratio of visitors at website i to total number subjects). If a value is not set to 1, then it is set to 0. Next, we simulated identified purchasing trails from the set of simulated IP address trails. We initialized identified purchasing trails to be equal to the set of IP address trails. Then, missing data was simulated by flipping trail values of 1 and 0 to the ambiguous value * with probability equal to $1 - \frac{|\delta_i|}{|\psi_i|}$ (i.e., 1 - ratio of purchasers to visitors at website i). We considered increases in the number of websites sharing data as a multiple w of the estimated probabilities, such that we concatenated w trails to consider a larger trail. Results for REIDIT-I are provided in Figure 5.9(b) for increasing size datasets.

Holding visit and purchase probabilities constant, the algorithms scale to accommodate very large populations and numbers of locations. The number of trail re-identifications in a dataset decreases linearly in a log scale of the size of the population. The slope decreases as the number of locations increases. For Figure 5.9(b), the slopes are approximately -



(a) REIDIT-C (b) REIDIT-I

Figure 5.9: Scaling of REIDIT-C and -I algorithms to changes in increased populations (in base 10 log scale) and number of locations.

0.27, -0.20, and -0.08, for 24, 48, and 72 purchasing locations, respectively, and continue to decrease with increasing numbers of locations. Similar scaling characteristics were found for REIDIT-C, as shown in Figure 5.9(a).

5.5 Discussion: Probabilistic Basis for REIDIT

In Chapters 2 through 4 we presented a formal model of trail re-identification and a family of algorithms for achieving re-identification. From a theoretical standpoint, we proved the limits of REIDIT scale exponentially with the number of data sharing locations. This is due to the fact that a trail can be represented as a binary string. However, when we step beyond theory and into the real world, there is no guarantee that exponential scaling of re-identifiability via REIDIT is guaranteed. Research in diverse areas, including demography, epidemiology, e-commerce, and web personalization suggests that there are trends in the manner that individuals choose which locations to visit. One of the main reasons is rooted in social, physical, and economic constraints placed on entities and data collectors in the real, as well as the virtual, world. Entities are not random agents who generate binary strings with uniform probabilities, but autonomous individuals who make conscious decisions according to various features associated with both the available locations and personal preferences.

The above analyses demonstrate the popularity of a location influences the re-identifiability

of trails in a population. The REIDIT algorithms provide a computational means for linking data, however, can we use trails to estimate the re-identifiability of a particular trail? This is an interesting concept, and we begin to address this with respect to REIDIT-C using a basic probabilistic model.

As we have stated multiple times, the primary reason why the theoretical maximum of re-identification is not achieved is that people do not visit data collecting locations randomly. At a gross level, we can imagine each location is associated with a distinct probability of an entity visiting it. Using the representation for datasets from section 5.4, let the data be one-to-one and unreserved. If we consider locations independently of each other, then based on the Bernoulli probabilities in the scaling experiments, we can represent probability of an arbitrary trail, with unambiguous values, $x_\Psi = [b_{x,1}, b_{x,2}, \ldots, b_{x,|C|}]$ being observed is as a multinomial:

$$Pr(x_\Psi|\Psi) = \sum_{i=1}^{|C|} \left[b_{x,i} \frac{|\psi_i|}{|S|} + (1 - b_{x,i}) \left(1 - \frac{|\psi_i|}{|S|} \right) \right].$$

where S is the set of entities in the population.

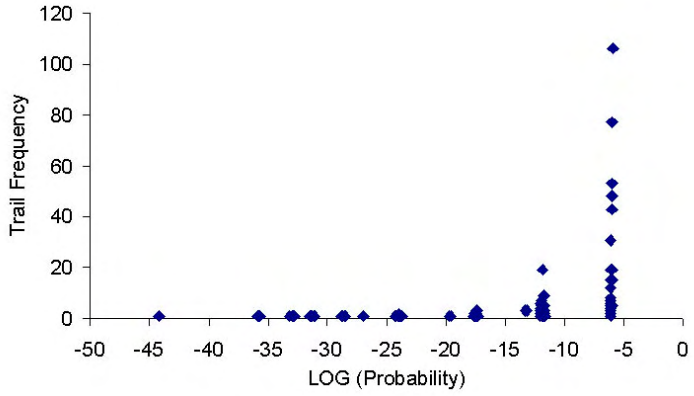


Figure 5.10: Probability of trail uniqueness for the CF dataset as a function of trail frequency (i.e., the number of times a particular trail is actually observed).

The resulting probabilities for the CF dataset trails are shown in Figure 5.10. To orient the reader, since probabilities are measured in the log scale, the more negative a probability, the more improbable it is to observe the trail. Intuitively, we find that as the frequency of a trail decreases, there is an increase in the probability of the trail being unique.

5.6 Discussion: An Information Theoretic Perspective

Though the REIDIT algorithms expose unique patterns for trail re-identification, the above analyses provide only rudimentary intuition regarding how the underlying distribution of individuals' data to locations influences the re-identifiability of the system. In this section, we begin to address this relationship in a more in-depth manner by studying several types of controlled distributions of individuals to locations, including uniform and power law distributions.

There are many aspects of location-based information that influences the re-identifiability of a system. From the analyses above, our findings suggest the contributing components include the number of subjects, the number of locations, the distribution of subjects to locations, as well as the parameters controlling said distributions. In this section, we concentrate on the number of locations and the distributions guiding subject access to these locations. Thus, for the analyses herein, the number of subjects is fixed as 1000. For our populations, we generate two types of systems, the first is based on uniform access behavior and the second is based on a Zipf access behavior.

A subject's trail in a uniform distribution is controlled by a single parameter p. Basically, the probability that any arbitrary value $b_{x,c}$ equals 1 is p (and $b_{x,c}$ equals 0 with probability $1 - p$). For our experiments we sample p from the range $[0, 1]$ at equidistant intervals of (i.e., p over $\{0.1, 0.2, \ldots, 1\}$).[4] Similarly, populations that are guided by the Zipf distribution are generated using Equation 5.1 with X set to 100. The Zipf is studied by varying the parameter α over the interval $[0, 1]$, and sampling points as done for the uniform distribution. Recall, the α parameter functions as a dampening factor on the slope of the plotted curve. In Figure 5.11 we demonstrate this effect by plotting the expected number of visitors for a population of 100 entities.

For each distribution type and parametrization, the populations are allocated to a set of locations C over the range of 3 to 40 locations. For each tested data point, such as $\langle |C| = 10, p = 0.3 \rangle$, we generate 100 populations. Each population is subjected to either the REIDIT-C or REIDIT-I algorithm.

The resulting 10-point plots for REIDIT-C are depicted in Figures 5.12 and 5.13, and the plots for REIDIT-I are depicted in Figures 5.14 and 5.15. In these plots the mean percentage re-identified and +/- one standard deviation for the 100 simulated populations are represented by the lower of the two plotted curves. The x-axis corresponds to the parameter of the distribution in question, while the left y-axis corresponds to values of the mean percentage re-identified. For completeness, and to dispel confusion, the upper curve corresponds to entropy (which will be formally defined in a moment) on the right y-axis.

[4] In theory, any number of points on the [0, 1] range will suffice. We choose 10 equidistant points for equal coverage of the distributions in consideration for this research.

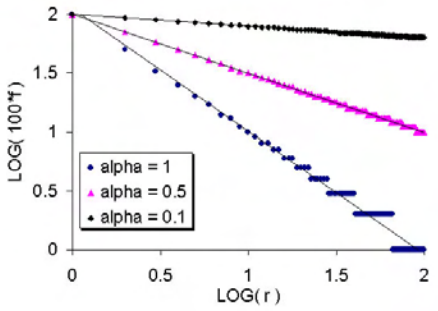
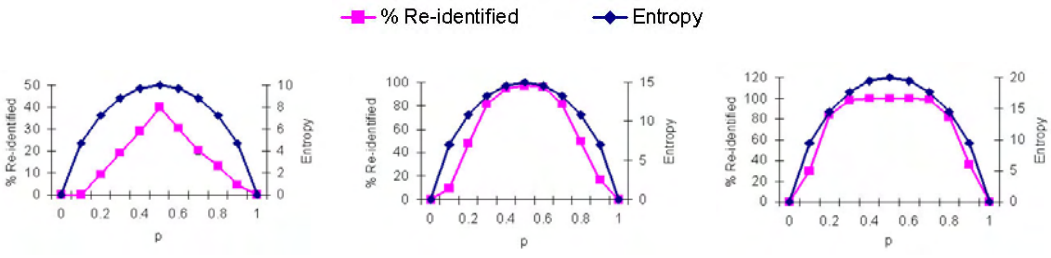


Figure 5.11: The dampening effect of α on Zipf skew in a 100 entity population.

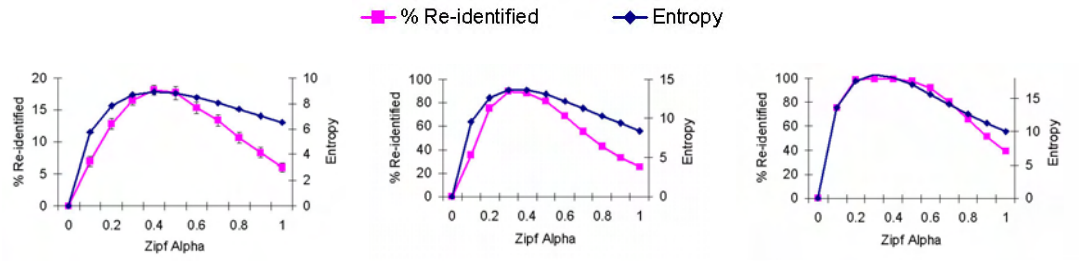


(a) 10 Locations (b) 15 Locations (c) 20 Locations

Figure 5.12: REIDIT-C mean re-identifiability of simulated subjects distributed according to uniform access distributions. Error bars correspond to one standard deviation of the simulated populations. The top line corresponds to entropy (represented by the right y-axis) and the lower line corresponds to percent of 1000 individuals re-identified.

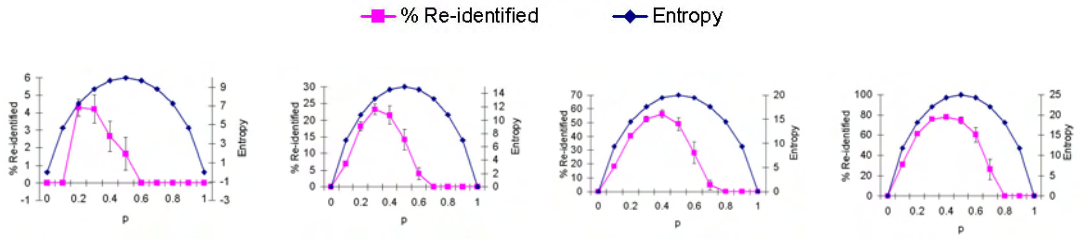
Though there is no direct way to compare the parameterizations of the uniform and Zipf distribution, there are several interesting observations that can be made from the re-identification plots. First with respect to both the REIDIT-C and REIDIT-I re-identification algorithms, it is apparent that the uniform distribution consistently yields a larger number of re-identifications than the Zipf distribution. This is observable, even by visual inspection, by considering the maximum re-identifiability of the distribution type. For example, when considering 10 locations, REIDIT-C re-identifies a maximum of approximately 40% of the subjects distributed uniformly (which occurs when $p = 0.5$), as opposed to around 16% of the subjects that are distributed in Zipf high skew (which occurs when $\alpha = 0.4$). This finding is consistent across all systems as the number of the locations in consideration is increased.

Second, we consider a less readily observable feature that directly relates to the gen-



(a) 10 Locations (b) 15 Locations (c) 20 Locations

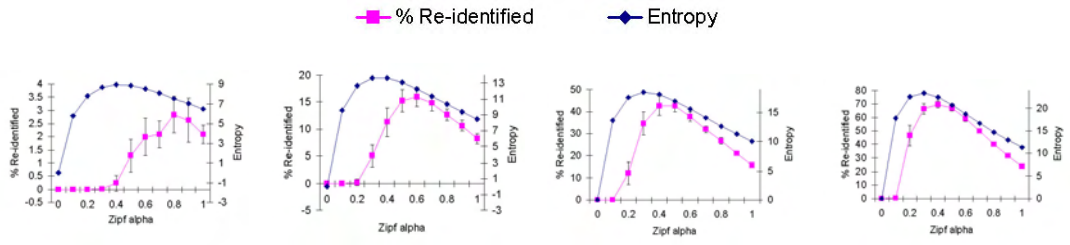
Figure 5.13: REIDIT-C mean re-identifiability of simulated subjects distributed according to generalized Zipf distributions. Error bars, the representation of the two curves, and the two y-axes are the same as in Figure 5.12.



(a) 10 Locations (b) 15 Locations (c) 20 Locations (d) 25 Locations

Figure 5.14: REIDIT-I mean re-identifiability of simulated entities distributed according to uniform distribution with probability p. Error bars, the representation of the two curves, and the two y-axes are the same as in Figure 5.12.

eral re-identification capability of a distribution type. To compare distributional types (i.e., uniform vs. Zipf), we consider the area under the re-identification curve. This is calculated as the total area under the 10-point mean re-identification curve (i.e., the average number of re-identifications in 100 simulated populations). The results of this calculation with respect to distributions and algorithm results are presented in Figure 5.16. Though the uniform distribution always yields the larger maximum number of re-identifications, the Zipf distribution is almost always the more linkable when considering all parameterizations. This is obviously so in the case of REIDIT-I re-identification, where Figure 5.16(b) shows that the Zipf always dominates. Similarly, under REIDIT-C, Zipf is both the initial and inevitable dominant. However, this analysis reveals an unanticipated and intriguing finding. In certain ranges, the uniform distribution dominates the Zipf! In Figure 5.16(a),



(a) 10 Locations (b) 15 Locations (c) 20 Locations (d) 25 Locations

Figure 5.15: REIDIT-I mean re-identifiability of simulated entities distributed according to Zipf distribution with probability α. Error bars, the representation of the two curves, and the two y-axes are the same as in Figure 5.12.

this finding is observed between approximately 8 and 18 locations.

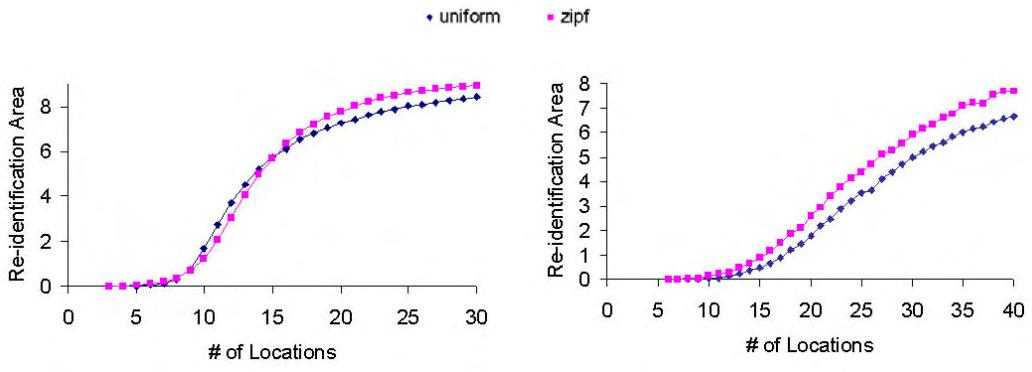
The flip in distribution re-identifiability dominance occurs for two reasons. First, Zipf dominates when there are not many locations in consideration because it is more difficult to realize trails of all 1's. Second, Zipf dominates as the number of locations increase because it is easier for lesser accessed locations, which corresponds to the newly considered locations, to convert an unlikely trail into an extremely unlikely trail.

5.6.1 Interpretation of System Re-identifiability

The trails that were generated for the experiments are initially Boolean vectors of 0's and 1's. As such, it seems feasible that each trail can be likened to a measure of information available on a subject. Continuing along this line of thought, it is plausible that the trail re-identifiability of a system is related to the Shannon entropy, as defined in information theory [128]. From a general standpoint, the entropy provides a characterization of the total amount of randomness in the distribution of 1's and 0's for a variable. Thus, the entropy of the trail matrices is a general predictor of re-identifiability of a system.

For our purposes, let X_Ψ be the trail matrix that maps a population of subjects S to a set of locations C. Also, let f_c be the fraction of subjects in S that unambiguously visit location c (i.e., the number of trails with a value of 1 for location c). Given this information, the entropy for a single location c, $H(c)$, can be calculated as $H(c) = -f_c \log f_c - (1 - f_c) \log (1 - f_c)$. For synthetic populations generated during the experiments, each location is allocated subjects independently. Thus, the entropy measure for the entire system X_Ψ is computed as $H(X_\Psi) = \sum_{c=1}^{|C|} H(c)$.

Both the entropy of the system and the re-identification of populations over different



(a) REIDIT-C (b) REIDIT-I

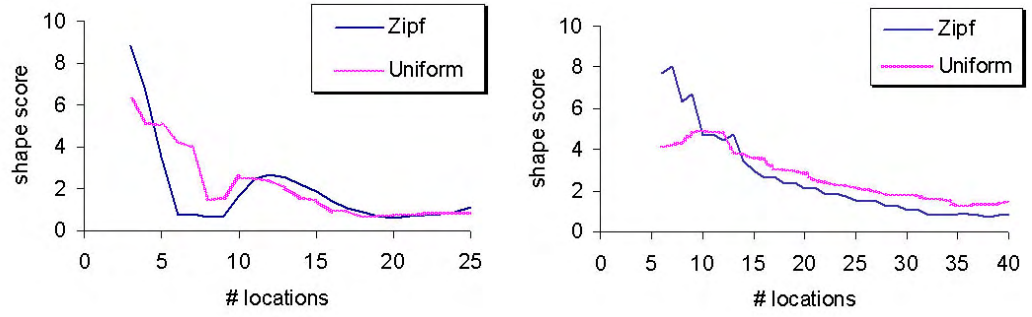
Figure 5.16: Area under the mean re-identifiability curves for simulated populations.

distributions produce response curves in terms of how re-identification capability is influenced. Visually, the results can be observed in Figures 5.12 and 5.14. As stated above, the entropy is the upper line, while actual re-identifications is the lower line. The scale for the entropy is provided on the right y-axis.

It is apparent from these plots that there exists a relationship between the actual re-identification curve (R) and the expected re-identification curve as predicted by entropy (E). At the most general level, it is visually verifiable that, as the number of locations increases, the actual re-identification curve tends towards the entropy prediction. From a mathematical standpoint, we consider these curves as functions, such that $R(x) = y$.

To determine how E and R relate to each other, we define several basic metrics for comparison. Though it is desirable to use known techniques for comparison, the curves generated for re-identification analysis do not relate to standard probability, or cumulative, distribution functions. Thus, there is no statistical or numerical test to compare the resulting curves to one another. To address this issue, we define several metrics based on observable features relating the curves. The first measure is called the *shift* σ of the curves. Shift measures the distance along the x-axis between the maximum y-value peaks of the two curves. The second measure is called the *shape* κ of the curves. Shape relates the general shape of the two curves to one another. Shape is calculated as the scaled difference between the 10-point plot of E and R. More formally, both metrics are computed as:

$$\sigma(E, R) = |max_x R(x) - max_x E(x)| \tag{5.2}$$



(a) REIDIT-C (b) REIDIT-I

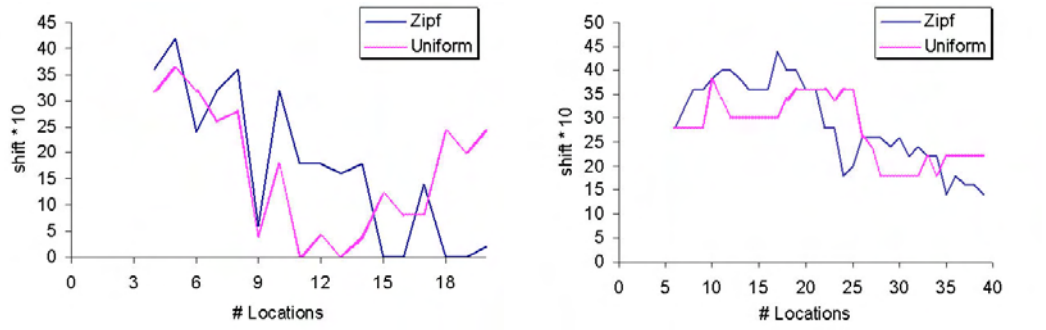
Figure 5.17: Shape metric for similarity in actual and entropy re-identification curves. The valley characterizes when there the actual number of re-identifications curve begins to plateau.

and

$$\kappa(E,R) = \sum_{i=1}^{10} |E(i)\frac{max(R)}{max(E)} - R(i)| \tag{5.3}$$

The resulting information from these metrics is summarized in Figure 5.17 and 5.18. Both of these metrics are a characterization of features that measure the distance between the distributions. As values for the metrics tends toward 0, the curves converge. As expected, the curves tend toward convergence as the number of locations increase. Yet after convergence comes into the line of sight, a counter-intuitive phenomenon occurs. Specifically, after a certain number of locations are considered for a particular distribution and trail re-identification algorithm, the E and R curves begin to diverge from each other. This is an artifact of the limits of re-identification. Notice that in Figures 5.12 through 5.15, when a lesser number of locations are considered the re-identification curve has a well defined peak. This peak corresponds to the parameter at which the distribution is most amenable to re-identification. However, this peak is only discernible when less than all of the trails are linked. Thus, when the system is fully linked at multiple parameterizations of the distribution, the re-identification curve plateaus at 100%, while the entropy continues to be well defined. This limit to re-identification causes the observed re-identification curve to be improperly matched to the entropy of the system. Therefore, there is no real divergence, but rather a limit to independent use of the entropy metric.

The shape metric allows for the discovery of another notable feature that captures how the distribution type influence different trail re-identification algorithms. Note that via



(a) REIDIT-C (b) REIDIT-I

Figure 5.18: Shift metric for distance between max peak of re-identifiability curves.

REIDIT-C, the uniform distribution converges earlier than the Zipf distribution. In contrast, when subjected to the REIDIT-I algorithm, the uniform distribution converges after the Zipf distribution. Ah, a paradox! At first consideration, one would expect that one distribution type, either uniform or Zipf, would converge earlier in both algorithms. However, this paradox results from both how trails are generated under the two distributions as well as how the trail re-identification algorithms leverage information. First, consider the re-identification algorithms. REIDIT-C looks for a unique bit pattern. In this sense, both 1's and 0's are contributing evenly to the trail re-identification process. This is why the re-identification curve for the uniform distribution is balanced, or has no shift around the midpoint of p . In other words, the % linked is approximately equivalent for +/-x around the parametrization of p = 0.5. With respect to REIDIT-I though, a * value in a trail functions as a fuzzy bit, since it can be used as either a 0 or a 1. Thus, as p tends toward 1, trails with fewer unambiguous values than $p|C|$ become extremely difficult to link, and the re-identification curve shifts away from high values of p, which allow for trails with large amounts of unambiguous values. This is only one part of the problem though. In effect, the Zipf distribution should be hindered by this feature as well. However, due to the fact that the Zipf distribution allows for locations to have different entropy values (because the system is a single uniform distribution), the Zipf system ends up revealing more re-identifications. Thus, the total amount of re-identification in the Zipf tends to be greater. To validate this claim, it is simple to observe that the average re-identification, but not the maximum, for the Zipf is greater than the uniform.

5.6.2 Limitations and Extensions

Though this analysis provides a theoretical foundation for why and how particular distributions are susceptible to different types of trail re-identification, there are certain caveats of the simulation design that limit the extension of these results. First, this research is biased in that it does not completely represent real world populations. This is because in the real world most individuals do not consider locations independently. Rather, there tend to be patterns in location-based access. These patterns are different than the unique features we exploit in the REIDIT algorithms. As a result of such location access behavior, the re-identifiability of the synthetic populations used in this research are probably inflated. Thus, one possible extension to this work is to study re-identification under different types of collation patterns. Generating datasets that adhere to complex patterns is still very much an open question in the statistics and data mining communities; however, there is some research that may be of use, including multivariate distributional theory, genetic algorithms for binary string evolution, and market basket data synthesis. Expanding on the latter, several groups have introduced approaches for generating synthetic market baskets [82] and, in some respects, the Boolean trail re-identification problem can be positioned as a market basket problem. Each location can be considered a different product that an individual decides whether or not to make a purchase. Thus, if one was to define a set of purchasing patterns, possibly as association rules, then it is possible that more realistic trail matrices could be constructed using synthetic market baskets. The foreseeable limitations lie in the fact that market basket generation is useful for studying interesting patterns, but may not facilitate the consideration of outliers, which the re-identification algorithms are designed to discover. Therefore, before synthetic market basket data can be used, we must assess how realistically outliers can be represented.

Second, the distributions used in this study consist of homogenous populations. Either all of the population adheres to skew or uniform distributions. However, what is the effect of mixture models of populations on re-identification? Is it possible that re-identifiability is facilitated when half the population is uniformly distributed while the other half is skewed? It is a complex problem that offers another feasible direction for research into the fundamentals of trail re-identification.

5.7 Conclusions

In this chapter, we demonstrated that real world data is susceptible to trail re-identification. We illustrated this concept with two data sharing environments: 1) healthcare and genomic data and 2) online data capture. Our findings highlight certain aspects of a distributed system influence susceptibility in the real world, such as location popularity and data to

location distribution. With respect to the latter, we extended our analysis into controlled simulated populations to provide intuition into the theoretical aspects of different location access distributions on trail re-identification. Specifically, our analysis focused on the relationship between uniform and high skew distributions. From a theoretical perspective, we discovered that flat distributions (e.g., uniform) are always more susceptible to re-identification with trails in unreserved matrices (i.e., there are no ambiguous * values). Yet, when there is incomplete information in the trails, then the Zipf distribution yields greater susceptibility to re-identification.

This chapter concludes our presentation on trail re-identification. In the next chapter we begin the second part of this work, which addresses the issue of trail anonymization.

Part II

Unlinkability

Chapter 6

k-Unlinkability

Part I of this dissertation followed a logical progression in the study of trail re-identification. In Chapters 1 through 3, the reader was introduced to trails, at both informal and formal levels, and was provided with several examples of how trails manifest in today's society. In Chapter 4, the reader was presented with tools for re-identifying seemingly anonymous data trails in an efficient manner. And in Chapter 5, we investigated the degree to which real world populations produce trails that are susceptible to re-identification, as well as why certain populations are more susceptible than others. Throughout our presentation we strived to answer the question, "Do trails pose a privacy problem?", and, based on our findings, we showed trails can be constructed and that in the real world they are sufficiently unique to support significant amounts of re-identification. This leads us to a definitive answer - yes.

While the previous investigations point out that re-identification vulnerabilities exist, we are now presented with the question "How do you prevent trail re-identification while allowing data sharing?" To answer this question, we shift our attention from re-identification to anonymization. In this chapter, we begin to address the issue of anonymity, with respect to trails, from a general perspective.

The organization of this chapter is as follows. We begin by reviewing related prior research on models of privacy protection. Next, we introduce a novel formal protection model that we call k-unlinkability. Finally, we compare this model to pre-existing models. In Chapter 7 we introduce several methods for transforming data to satisfy k-anonymity.

6.1 Background and Related Research

The number of scholarly scientific publications on topics such as privacy and confidentiality is vast. Not only is the number of papers on these topics large, but the areas of ap-

plication for these concepts is equally substantial. For instance, there exists research and applications presented from the perspective of contingency table statistics, location-based services (e.g., GPS, mobile communications, etc.), policy specification, data mining, secure multiparty computation, e-commerce, surveillance, electronic communications, and the list goes on - and is ever-increasing. Additionally, each community employs their own definition for terms such as "privacy" and "anonymity". To perform a comprehensive presentation of all privacy-related research fields would drown the reader in an ocean of concepts, references, and architectures to understand the inherent relationships between various ideas - much of which bear little influence on this dissertation's research. This chapter therefore reviews several key ideas in privacy research most-related to this dissertation.

Previous research has produced a number of models for protecting the identities of a disclosed database. In this chapter, we review models related to our model of protection. Specifically, we study *k-map* [136], *k-anonymity* [119], and *k-ambiguity* [107].

To review these models, we add to notation presented in earlier chapters. For reference, let S be the set of entities. Let $\tau(A_1, \ldots, A_p)$ be a privately held table with $A^\tau = \{A_1, \ldots, A_p\}$. Let $t_i = [a_{i,1}, \ldots, a_{i,p}]$ be a tuple in τ. Let $QI^\tau \subseteq A^\tau$ be the set of attributes called a "quasi-identifier" that can be used in the linkage process between τ and a collection of information available on S. Let $f(\tau) = \tau'$ be a function that outputs a table with quasi-identifying values protected for disclosure purposes. Finally, let $\alpha : \tau \rightarrow S$ be a function that maps tuples back to their underlying entities.

6.1.1 k-Map

The *k-map* protection model [136] provides guidelines regarding the relationships between observed features in the protected table and the population from which it was derived. It is a formal protection model in that exact guarantees of protection against re-identification can be derived. Informally, k-map protection means that each record in a release refers to at least k entities in the population.

More specifically, let ${}_\tau G_S$ be a relation between tuples in τ and entities in S. Then, according to Sweeney [136], *k-map* protection is satisfied for τ if there exists a ${}_\tau G_S$, such that $\forall t \in \tau$:

1. $\langle t, \alpha(t) \rangle \in {}_\tau G_S$, and
2. $|\{s | \langle t, s \rangle \in {}_\tau G_S\}| \geq k$.

The k-map protection model does not describe an algorithm for augmenting the quasi-identifier for protection. This work introduces k-unlinkability as an enforcement of k-map applied to trails.

6.1.2 k-Anonymity

The k-map protection model specifies what must be true regarding the disclosed data and the population from which it was derived. However, it is not always possible to ascertain the information regarding the population that is external to the shared dataset. As a result, a second formal protection model known as *k-anonymity* was introduced [136, 120, 139]. The k-anonymity protection model is a more restrictive case of k-map and is based on the notion of *indistinguishability* with respect to the disclosed dataset.

Formally, let ${}_{\tau}D_{\tau}$ be a relation, such that $\langle i, j\rangle \in {}_{\tau}D_{\tau}$ if tuples i and j are indistinguishable according to a decidable criterion. Table τ is said to be protected if:

$$\forall t \in \tau, |\{i|\langle t, i\rangle \in {}_{\tau}D_{\tau}\}| \geq k. \tag{6.1}$$

k-anonymity defines indistinguishability as *strict equality*. Tuples t_i and t_j are said to be strictly equal when

$$\forall A_x \in QI^{\tau}, a_{i,x} = a_{j,x}. \tag{6.2}$$

Like the k-map protection model, k-anonymity does not define a method for augmenting the quasi-identifier for protection. Rather, it defines when a database satisfies protection.

6.1.3 k-Ambiguity

There are a number of methods that have been proposed for privacy protection that are not formal protection models. For this research, the most related is a method for privacy protection known as *k-ambiguity* [107, 150]. Like k-anonymity, the k-ambiguity method specifies protection via indistinguishability. However, k-anonymity and k-ambiguity differ in their definition of the relation ${}_{\tau}D_{\tau}$. While k-anonymity calls for strict equivalence, k-ambiguity defines indistinguishability in the form of *indiscernability* [130]. Two tuples $t_i[a_{i,1}, \ldots, a_{i,p}]$ and $t_j[a_{j,1}, \ldots, a_{j,p}]$ are said to be indiscernible when

$$\forall A_x \in QI^{\tau}, a_{i,x} = a_{j,x} \vee a_{i,x} = * \vee a_{j,x} = *. \tag{6.3}$$

Notice that in Equation 6.3, the k-ambiguity model defines how data must look in order to be protected. For data to satisfy the k-ambiguity model, it must be made more ambiguous, such that a tuple's values are made less specific in the form of replacement by the ambiguous * value. As a result, two tuples are indiscernible when all unambiguous values are the same for the quasi-identifier.

Equation 6.3 is a fundamental deviation from the k-map and k-anonymity protection models. The latter specify an architecture for protection, but not how the data must be

protected. Yet, if ambiguation is the only option for data protection, then Equation 6.2 is a special case of Equation 6.3 and every table that satisfies k-anonymity must satisfy k-ambiguity. We mention this relationship because the manner by which k-unlinkability is achieved in this dissertation uses ambiguation.

The reader should recognize that k-ambiguity is not a generalized form of k-anonymity. Ambiguation is merely one form of data protection and there are many alternative schemas by which k-anonymous data can be generated. In fact, an extremely important aspect of k-ambiguity to note is that it cannot formally provide privacy protection in an environment where aggregate information on a population is published. When aggregate information is available, as is more often the case than not in the real world, k-ambiguity leaks inferences to the suppressed values. Thus, k-ambiguity is not a formal protection model, whereas k-map and k-anonymity are formal protection models.

6.1.4 Implementation Issues and Complexity

The first instantiation of k-anonymity and k-ambiguity in practice was with respect to suppression and generalization [107, 121]. Suppression is the replacement of a value with a wildcard value (similar to the * value for trails discussed in the first part of this dissertation). Generalization is the replacement of a value with a less specific value, such as when zip code 15213 is generalized to 1521*, which can be generalized into 152**, and so forth, until the final generalization is full suppression of the value.

In theory, however, discovering the function $f(\tau) \rightarrow \tau'$ that minimizes the number of suppressions to satisfy k-anonymity, or k-ambiguity appears to require exhaustive search strategies [9, 80, 119, 138]. This comment is made in light of the problem's computational hardness. First, Meyerson and Williams [100] proved achieving k-anonymity by minimizing the number of attributes suppressed is an NP-hard problem. Second, Aggarwal et al [1] proved minimizing the number of cells to suppress (regardless of attribute) to satisfy k-anonymity is an NP-hard problem as well. Third, Vinterbo [149] proved minimizing the number of cell suppressions to satisfy k-ambiguity is an NP-hard problem.

Yet, in real-world populations, quasi-identifying values are non-uniformly distributed[1]. Experimental evidence suggests certain heuristics, such as those in [68, 157], which use genetic and simulated annealing algorithms, work well in practice [80, 150].

It should also be kept in mind that k-anonymity does not require generalization or suppression. For instance, work presented in [38] and [106] propose averaging techniques to achieve k-anonymity. It remains an open question as to whether or not k-anonymity in general is NP-hard or if there are certain types of solutions that can be achieved efficiently.

[1]For several examples, see the case studies in Chapter 5 for trail distributions, see [135] for demographic distributions, or see [84] for genomic data distributions.

Age	Gender	Zip
30	Male	15213
33	Male	15217
33	Female	15213
30	Male	15213

(a) Quasi-identifiers in a private table.

Age	Gender	Zip
30	Male	15213
33	*	1521*
33	*	1521*
30	Male	15213

(b) Quasi-identifiers in a 2-anonymous table.

Age	Gender	Zip
30	Male	15213
33	Male	1521*
33	*	15213
30	Male	15213

(c) Quasi-identifiers in a 2-ambiguous table.

Figure 6.1: Tables protected by generalization and suppression.

Figure 6.1 provides an example of the difference between a formal protection model, such as k-anonymity, and a non-formal model, such as k-ambiguity, that can arise from generalization and suppression. The original table τ is shown to the left in Figure 6.1(a). The middle and right tables depict protection that satisfy 2-ambiguity and 2-anonymity, respectively. In this example, the table produced by k-ambiguity in Figure 6.1(c) leaks inferences.

Specifically, it can be inferred that the second and third tuples, $[33, Male, *]$ and $[33, *, 15213]$, must have different values than each other in the suppressed cells for the same attributes, otherwise suppression would have been unnecessary. Thus, both tuples can not correspond to $[33, Male, 15213]$. When an adversary is aware of the domains for the attributes, then the table can be partially, and at times completely, reconstructed. If an adversary knows that the domain for the gender and zip attributes were $\{Male, Female\}$ and $\{15213, 15217\}$, then a recipient could directly infer, by process of elimination, that the second and third tuples corresponds to $[33, Male, \text{NOT}15213]$ and $[33, \text{NOT}Male, 1]$, or $[33, Male, 15217]$ and $[33, Female, 15213]$, respectively. Yet, such exact inferences can not be made from the 2-anonymous table in Figure 6.1(b). This is true even if an adversary was produced with the original values in Figure 6.1(a).

6.2 k-Unlinkability: A Formal Protection Model

When considering generalization and suppression to satisfy models, such as k-anonymity and k-ambiguity, the tuples in τ' are protected with respect to the original private table τ. However, in the trail re-identification problem, trail matrices communicate all quasi-identifying data (i.e., the trails) that is available. As a result, we can define a formal privacy model that accounts for the relationships between the trail matrices and, by transitivity, for the underlying population.

We base our model of data privacy on the notion of "unlinkability". Informally, we define unlinkability as the degree to which data corresponding to the same entity can not be discriminantly related. First, we define an *undisclosed* trail as a trail that is devoid of 1's (Definition 18). These are trails that correspond to data that is never disclosed by any location.

Definition 18 (Null Trail). *A trail is said to be a* null trail *if it consists only of *'s.*

Next, we define k-unlinkability (Definition 19) from the perspective of maximum matchings in bipartite graphs.

Definition 19 (k-Unlinkability). *Let X and Y be trail matrices with an association relationship, and let L_{XY} be the corresponding link matrix. Let $G = (V_X \cup V_Y, E)$ be the bipartite graph for link matrix L_{XY} and let $\mathbf{T}_{XY}$ be the union of maximum matchings in G. We say G satisfies* k-unlinkability *when each vertex v in G either:*

1. *participates in at least k different edges in $\mathbf{T}_{XY}$, or*
2. *all edges in $\mathbf{T}_{XY}$ that end at v are connected to a null trail.*

By Definition 19, there are two ways an element can satisfy k-unlinkabilty. The first way in which an element can satisfy k-unlinkability is when the element is matched to at least k different elements in the set of maximum matches. For example, consider the trail matrices X and Y in Figure 6.2. These matrices have an association relationship, as demonstrated with trail matrix W. Based on the trail matrices in Figure 6.2, we derive a link matrix with the corresponding bipartite graph shown in Figure 6.3(a). The maximum matchings of this graph are shown in Figures 6.3(b), 6.3(c), and 6.3(d). Each of these graphs are feasible unique trail linkage solutions. The union of these solutions corresponds to the original graph, and since every node in this graph has a degree of 2 or greater, the corresponding trail matrices satisfy 2-unlinkability.

Elements are not always disclosed, so there is another way in which an element can satisfy k-unlinkability. The second occurs when the element participates in edges in the maximum matchings, such that every edge is incident with a null trail. In this scenario, a vertex can be matched to less than k elements, but the re-identification is never completed because every elements corresponds to a null trail. In other words, we know that a re-identification could be made if the elements were disclosed, but we can not complete the re-identification due to the fact that such elements are never shared by any location.

For example, consider the trail matrices in Figure 6.4. These matrices have an association relationship and notice that the trail for y_1 in trail matrix Y is a null trail. We know that an element exists for y_1, but we do not know the real world values for this element. In the bipartite graph for the corresponding link matrix, shown in Figure 6.5(a), we represent

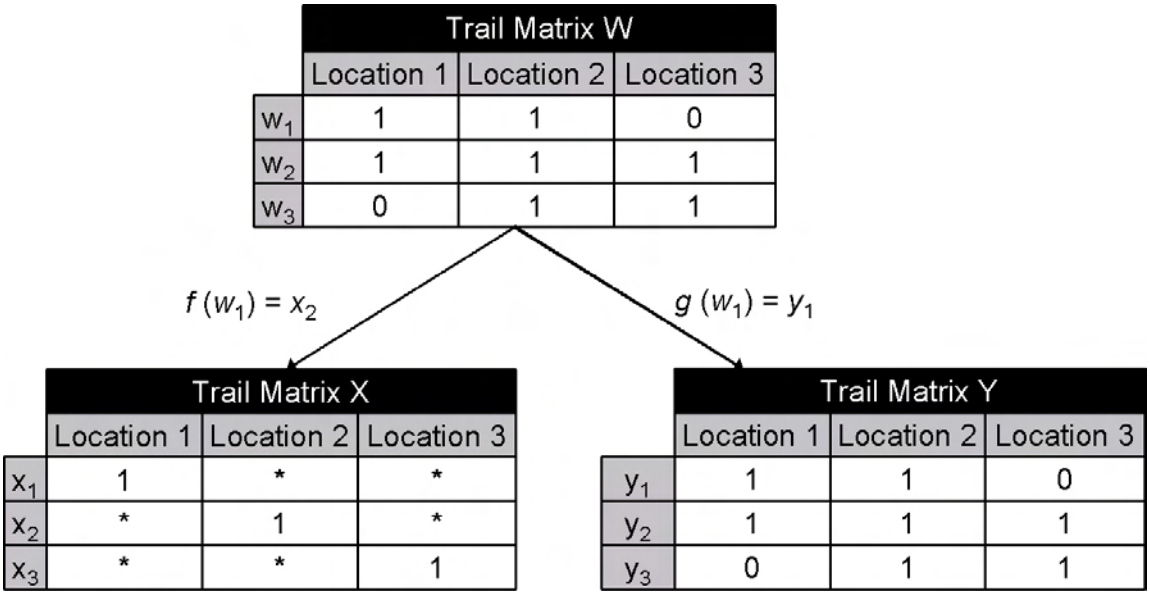
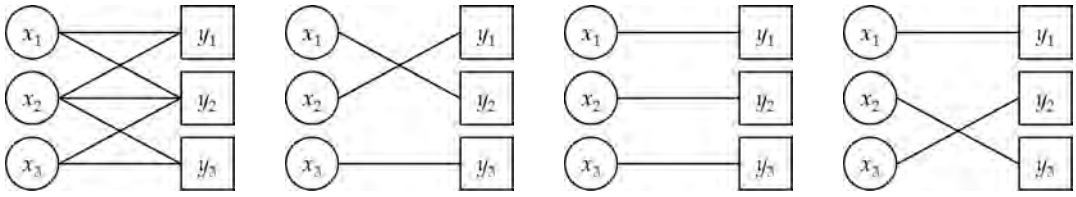


Figure 6.2: Trail Matrices X and Y have an association relationship via trail matrix W.



(a) Original Graph (b) Maximum Matching (c) Maximum Matching (d) Maximum Matching

Figure 6.3: Graphs in (b), (c), and (d) cover the set of maximum matchings for the original bipartite graph in 6.3(a). The union of the maximum matching graph is the same as the original graph, which is 2-unlinkable.

y_1 with a "????". The maximum matchings of this graph is shown in Figure 6.5(b). Notice, x_1, x_2, y_2, and y_3 make up a complete subgraph, such that each node has degree 2. Thus, each of these elements satisfy the first criteria for k-unlinkability. In contrast, x_3 and y_1 both participate in one edge only. However, this edge is incident with a null trail, so the elements satisfy k-unlinkability by the second criteria.

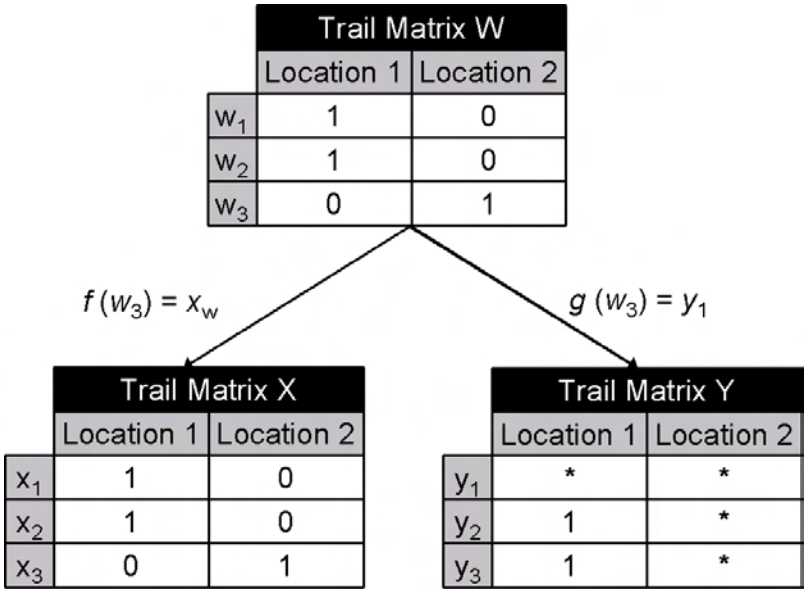
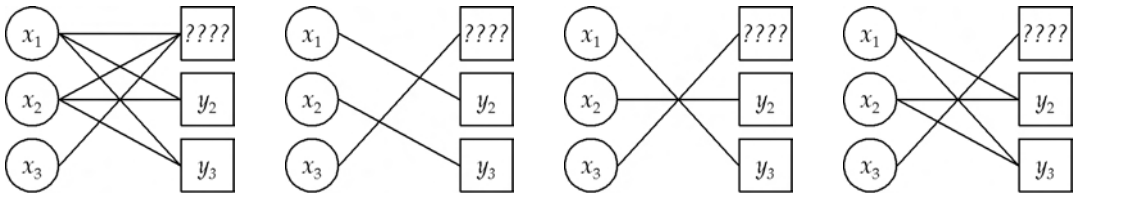


Figure 6.4: Trail Matrices X and Y have an association relationship via trail matrix W.



(a) Original Graph (b) Maximum Matching (c) Maximum Matching (d) Union of Maximum Matching

Figure 6.5: Graphs in (b) and (c) cover the set of maximum matchings for the original bipartite graph in 6.5(a). The union of the maximum matching graph is shown in (d), which is 2-unlinkable.

6.3 Comparison To Alternative Models

In this section, we address several aspects of the k-unlinkability model in more depth. Specifically we address its relationship to 1) k-anonymity and k-ambiguity as achieved through generalization and suppression and 2) alternative models for unlinkability quantification.

6.3.1 k-Unlinkability versus k-Anonymity

If we use the strict equality criterion with respect to X (or Y), then it is clear L_{XY} satisfies k-unlinkability. In other words, if a trail matrix is k-anonymous, as achieved through cell suppression, then any link matrix constructed from the trail matrix is k-unlinkable. For example, consider Figure 6.6. This link matrix satisfies 2-anonymity, which in turn satisfies 2-unlinkability. Notice that there are four maximum matchings that can be derived from the link matrix.

X_Ψ

	H_1	H_2	H_3	H_4
Ali	1	1	1	0
Bob	1	1	0	1
Charlie	1	0	1	1
Dan	0	1	1	1

Y_Λ

	H_1	H_2	H_3	H_4
actg	1	1	*	*
ctga	1	1	*	*
tgac	*	*	1	1
gatc	*	*	1	1

L_{XY}

	Ali	Bob	Charlie	Dan
actg	1	1	0	0
ctga	1	1	0	0
tgac	0	0	1	1
gatc	0	0	1	1

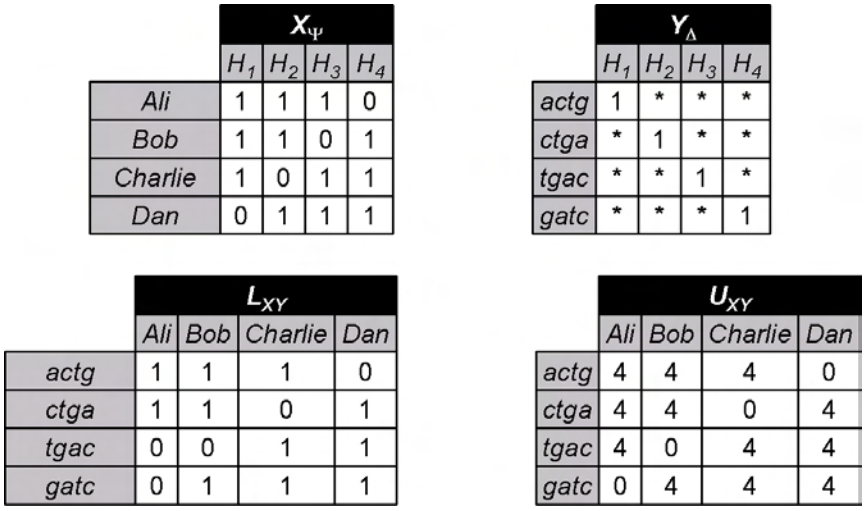
U_{XY}

	Ali	Bob	Charlie	Dan
actg	2	2	0	0
ctga	2	2	0	0
tgac	0	0	2	2
gatc	0	0	2	2

Figure 6.6: k-anonymity guarantees k-unlinkability.

However, k-anonymity is unnecessary to satisfy k-unlinkability. As an example, consider the trail matrices in Figure 6.7. There are 12 distinct maximum matchings that can be derived from the link matrix. Given the maximum matchings, it can be validated that the link matrix is 3-unlinkable while neither trail matrix is 3-anonymous.

The primary difference between k-anonymity, from a generalization and suppression perspective, and k-unlinkability is that the latter does not require equivalence of trails to satisfy protection. With respect to the bipartite graph representation, k-anonymity protection implies that every connected component of the graph is a complete bipartite graph, such that every node of one data type is connected to every node of the other data type in the component. In contrast, k-unlinkability protection does not imply how many components a protected graph can be partitioned into. In terms of the corresponding trails, k-unlinkability can satisfy the k-map formal protection model and preserve more variation in trails in comparison to k-anonymity.



X_Ψ	H_1	H_2	H_3	H_4
Ali	1	1	1	0
Bob	1	1	0	1
Charlie	1	0	1	1
Dan	0	1	1	1

Y_Δ	H_1	H_2	H_3	H_4
actg	1	*	*	*
ctga	*	1	*	*
tgac	*	*	1	*
gatc	*	*	*	1

L_{XY}	Ali	Bob	Charlie	Dan
actg	1	1	1	0
ctga	1	1	0	1
tgac	0	0	1	1
gatc	0	1	1	1

U_{XY}	Ali	Bob	Charlie	Dan
actg	4	4	4	0
ctga	4	4	0	4
tgac	4	0	4	4
gatc	0	4	4	4

Figure 6.7: k-unlinkability does not guarantee k-anonymity.

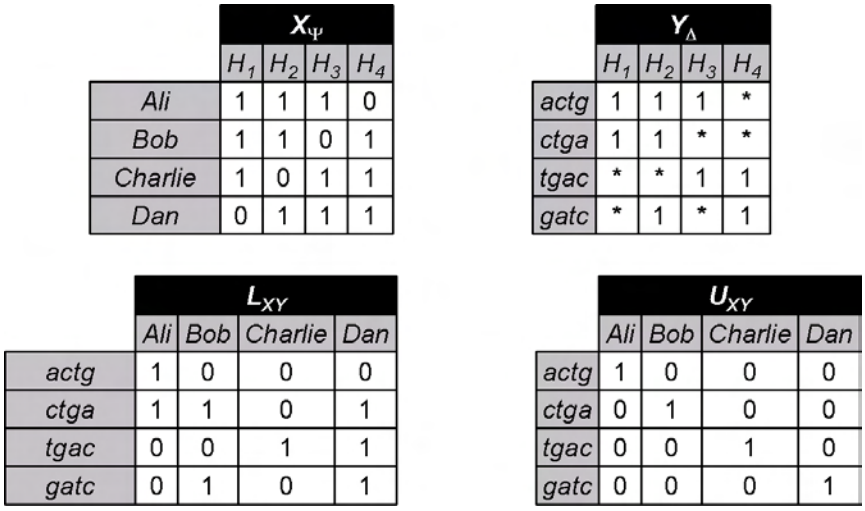
6.3.2 k-Unlinkability versus k-Ambiguity

In the previous subsection, it was shown there exists an explicit relationship between k-anonymity and k-unlinkability. The relationship between k-ambiguity and k-unlinkability is more subtle.

We begin by presenting an important non-implication regarding the relationship between k-ambiguity and k-unlinkability. Specifically, if a trail matrix is k-ambiguous, it is not necessarily k-unlinkable. In Theorem 5 we characterize this concept for a pair of databases. Though one database may contain ambiguous values, it can be linked to a database that does not satisfy k-ambiguity. This theorem formalizes one of the limitations of the k-ambiguity model.

Theorem 5 (k-Ambiguity does not imply k-unlinkability). *Given trail matrices X and Y have an association relationship, Y can satisfy k-ambiguity while failing to satisfy k-unlinkability with respect to X.*

PROOF. We can prove this theorem by contradiction. In Figure 6.8 we show two trail matrices, in which Y_Δ is reserved to X_Ψ. Note, Y_Δ is 4-ambiguous, but given the set of maximum matchings we observe every trail in Y_Δ can be uniquely re-identified to a trail Ψ, or is 1-unlinkable to X_Ψ. ■



X_Ψ	H_1	H_2	H_3	H_4
Ali	1	1	1	0
Bob	1	1	0	1
Charlie	1	0	1	1
Dan	0	1	1	1

Y_Δ	H_1	H_2	H_3	H_4
actg	1	1	1	*
ctga	1	1	*	*
tgac	*	*	1	1
gatc	*	1	*	1

L_{XY}	Ali	Bob	Charlie	Dan
actg	1	0	0	0
ctga	1	1	0	1
tgac	0	0	1	1
gatc	0	1	0	1

U_{XY}	Ali	Bob	Charlie	Dan
actg	1	0	0	0
ctga	0	1	0	0
tgac	0	0	1	0
gatc	0	0	0	1

Figure 6.8: Trails in Y_Δ are 4-ambiguous, but every trail in Y_Δ is 1-unlinkable to X_Ψ.

6.3.3 k-Unlinkability versus *ent*-Unlinkability

In previous research, Steinbrecher and Köpsell integrated various definitions from communications anonymity [34, 35, 127] and introduced a formal model of unlinkability [133] based upon information theory. To paraphrase Steinbrecker and Köpsell, let X and Y be two sets of data and let ${}_XR_Y$ be a relation that specifies which pair of elements $\langle x \in X, y \in Y \rangle$ belong to the same underlying concept. Their model uses a probabilistic framework, such that for a recipient of the data, $P((x,y) \in {}_XR_Y)$ represents the recipient's posterior probability of discerning if x and y are correctly related. Similarly, the recipient's posterior probability of correctly claiming two arbitrary elements are unrelated is $P((x,y) \notin {}_XR_Y)$. A fundamental assumption of this model is:

$$P((x,y) \in {}_XR_Y) + P((x,y) \notin {}_XR_Y) = 1. \tag{6.4}$$

Based on this assumption, the degree of unlinkability between two elements x and y is quantified as the entropy

$$\begin{aligned} H(x,y) = &-P((x,y) \in {}_XR_Y) \log P((x,y) \in {}_XR_Y) \\ &-(1 - P((x,y) \in {}_XR_Y)) \log (1 - P((x,y) \in {}_XR_Y)). \end{aligned} \tag{6.5}$$

For reference, we call this the *ent-unlinkability* of the data. According to Steinbrecker and Köpsell, an *ent*-unlinkability of 0 means the recipient has complete confidence in whether or not x and y correspond to the "same concept", whereas an *ent*-unlinkability of 1 means

the attacker is equally confident in stating x and y correspond to the "same concept" and "not the same concept".

The k-unlinkability model can be translated into an *ent*-unlinkability setting. Consider the scenario in Figure 6.7. Let E_y be the set of elements in Y for which $(x, y) \in {}_X R_Y$ holds true. Recall, *ent*-unlinkability defines $P((x, y) \in {}_X R_Y)$ as the recipient's posterior probability of discerning if x and y are correctly related. Since X and Y are one-to-one, the recipient knows that for an arbitrary trail y_Δ there is only one trail x_Ψ for which it is correctly related. Given ${}_X R_Y$ is output from the sum of maximum matching matrices, the recipient's posterior probability of correctly and discriminantly relating y_Δ and x_Ψ is:

$$P((x, y) \in {}_X R_Y) = \begin{cases} \dfrac{1}{|E_y|}, & \text{if } x \in E_y \\ 0, & \text{otherwise} \end{cases} \tag{6.6}$$

Similarly, the recipient can derive the posterior probability of correctly claiming two arbitrary trails y_Δ and x_Ψ are unrelated is:

$$P((x, y) \notin {}_X R_Y) = \begin{cases} \dfrac{|E_y| - 1}{|E_y|}, & \text{if } x \in E_y \\ 1, & \text{otherwise} \end{cases} \tag{6.7}$$

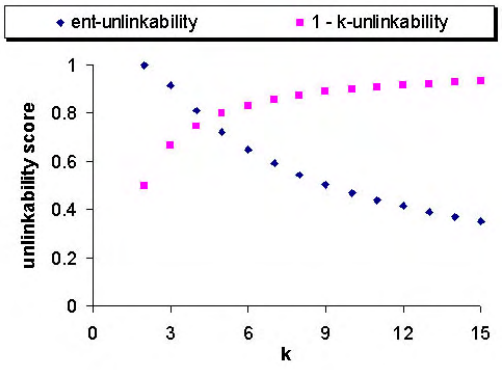
Given these probabilities, k-unlinkability satisfies the assumption of *ent*-unlinkability in Equation 6.4, such that

$$\begin{aligned} &P((x, y) \in {}_X R_Y) + P((x, y) \notin_X R_Y) \\ &\quad = \frac{1 + (|E_y - 1|)}{E_y} \\ &\quad = 1. \end{aligned}$$

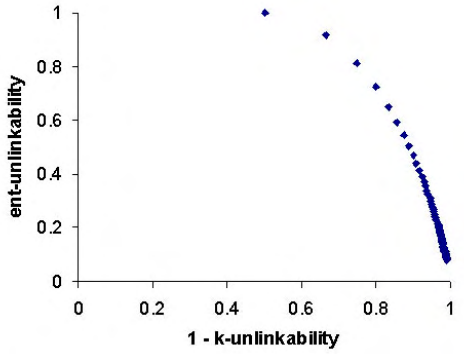
As a consequence, the *ent*-unlinkability of two arbitrary trails is calculated as:

$$H(x, y) = \begin{cases} -\dfrac{1}{|E_y|} \log \dfrac{1}{|E_y|} - \dfrac{|E_y| - 1}{|E_y|} \log \dfrac{|E_y| - 1}{|E_y|}, & \text{if } x \in E_y \\ 0, & \text{otherwise} \end{cases} \tag{6.8}$$

Though k-unlinkability can be rewritten in *ent*-unlinkability form, there is a disparity between the models. Since X and Y are one-to-one, it can be derived that the maximum *ent*-unlinkability score of 1 is only achieved when $P((x, y) \in {}_X R_Y) = \frac{1}{2}$. However, as k



(a) *Ent*-unlinkability score for trails x and y, where $x \in E_y$.



(b) *Ent*-unlinkability versus k-unlinkability.

Figure 6.9: Comparison of *ent*-unlinkability versus k-unlinkability in terms of protection scores.

gets larger, the unlinkability of two elements should become more pronounced. We depict this paradox in Figure 6.9(a).

The reason behind this paradox is that *ent*-unlinkability places equal dependence on $P((x,y) \in {}_X R_Y)$ and $P((x,y) \notin {}_X R_Y)$. When X and Y are one-to-one, then the probability $P((x,y) \in {}_X R_Y)$ never dominates, such that $P((x,y) \notin {}_X R_Y) \geq P((x,y) \in {}_X R_Y)$. However, for k-unlinkability it does not matter if a recipient can unequivocally discern that two elements are unrelated. We only want to ensure that the recipient can not discriminantly determine when two elements are related. So, we use $1 - \frac{1}{k}$ to represent the degree of unlinkability between two elements. Therefore, k-unlinkability is a variant of *ent*-unlinkability, such that protection is equal to $\max(0, 1 - \frac{1}{k})$. The protection score for k-unlinkability in comparison to *ent*-unlinkability is shown in Figure 6.9(a). Notice that the k-unlinkability and ent-unlinkability scores are monotonically inversely proportional as shown in Figure 6.9(b).

6.4 Conclusions

In this chapter we introduced a novel approach to privacy protection called k-unlinkability that satisfies the k-map formal protection model. It is similar to prior privacy models in that it is based on the notion of indistinguishability. Unlike prior models, however, k-unlinkability accounts for linkage in environments where there exists closure in the

population between known tables. As a formal model, k-unlinkability provides guarantees that are more exact than models of linkability based on probabilistic formulations and information theory. We further proved that k-ambiguity, an existing privacy protection method, leaks linkage inferences when tables are closed over a population. We also proved that generalization and suppression for k-anonymity for trails is a special case of k-unlinkability. In the Chapter 7 we address how we develop algorithms to transform trail matrices to satisfy k-unlinkability.

Chapter 7

Trail Unlinkability Algorithms

The k-unlinkability protection model presented in Chapter 6 describes what must be true regarding a privacy protected link matrix. However, this definition does not state how quasi-identifiers can be augmented to satisfy these conditions. Before we define algorithms to generate k-unlinkable trails we specify a fundamental constraint on permissable data protection methods: we cannot falsify information in the system. To prevent the injection of false information into the system from an operational perspective, we grant only the operation of suppression to methods that are used to unlink trails. In other words, unambiguous values can become ambiguous, but unambiguous values can not be swapped for other unambiguous values (i.e., 0 can not become 1 and vice versa), nor can ambiguous values be made unambiguous (i.e, * can not become 0 or 1).

7.1 General Trail Unlinkability

Given the suppression constraint, the problem definition for k-unlinkable trail construction can be stated in the following general form.

Definition 20 (k-Unlinkable Trails via Suppression). *Let Γ and Ω be sets of partitioned databases from a set of locations C. Derive Γ' and Ω', such that*

1. $\forall c \in C, \gamma_c{}' \subseteq \gamma_c$,
2. $\forall c \in C, \omega_c{}' \subseteq \omega_c$, *and*
3. L_{MN} *is k-unlinkable, where* $M = \bigcup_{c \in C} \gamma_c{}'$ *and* $N = \bigcup_{c \in C} \omega_c{}'$.

7.2 A Specific Trail Unlinkability Problem

Though the general trail unlinkability problem can be formally defined, it is not clear when this problem can be solved in the real world. One of the main reasons for this dilemma is that we do not always have the ability to suppress information from both trail matrices. For example, consider the genomic data disclosure scenario described in Chapter 2. In this environment there exists a set of partitioned databases for a set of hospitals; one database contains identified clinical data and the other contains de-identified genomic data. Legally, however, identified databases are always fully disclosed from the place of collection and, as a result, no data in this set can be altered. In contrast, data in the genomic database set is augmentable. In fact, genomic data can be held private until the collector, possibly through the oversight of an institutional review board, is confident the data is sufficiently de-identified.

Therefore, in this research we address a subproblem of the general case based on our observations of the real world. We assume one set of databases is always publicly disclosed. Thus, we refine our goal to produce trail matrices via suppression with the added constraint $\Gamma' = \Gamma$.

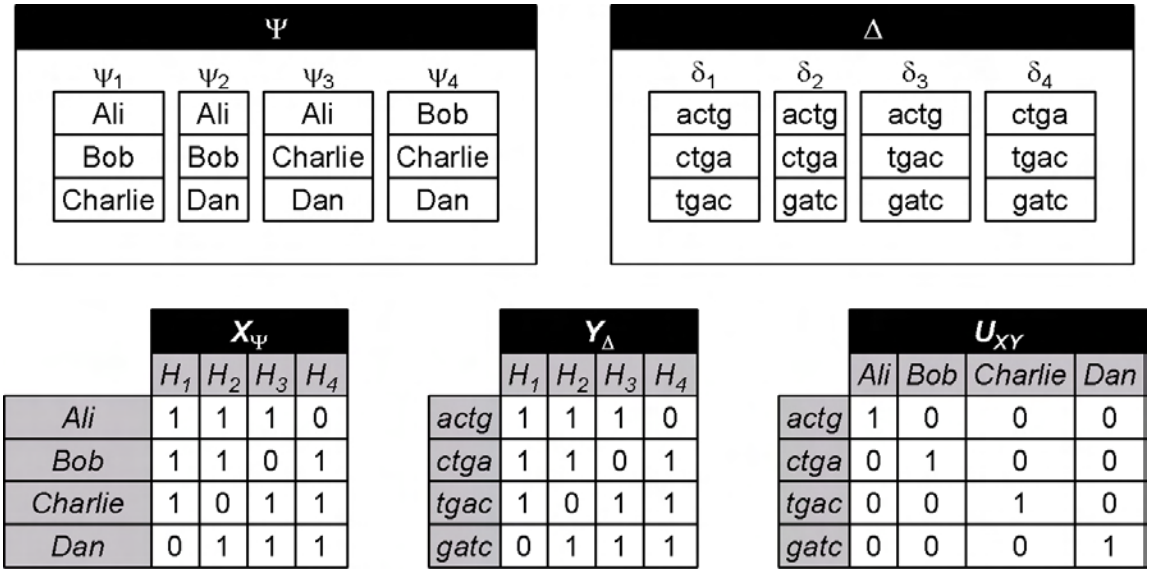
Definition 21 (k-Unlinkable Trails via Suppression in One Matrix). *Let Γ and Ω be sets of partitioned databases from a set of locations C, such that Ω is reserved to Γ. Derive Ω', such that*

1. *$\forall c \in C, \omega_c{}' \subseteq \omega_c$, and*
2. *L_{MN} is k-unlinkable, where $M = \bigcup_{c \in C} \gamma_c$ and $N = \bigcup_{c \in C} \omega_c{}'$.*

An example of suppression for k-unlinkability in trails is depicted in Figure 7.1. The original databases shown in Figure 7.1(a) are unreserved and every DNA record is uniquely re-identifiable. This finding is summarized in the U_{YX} matrix, which corresponds to the union of edges in maximum matchings, i.e., a 1 corresponds to the existence of an edge. Suppression of DNA records from the original databases, shown in Figure 7.1(b), makes every DNA record 2-unlinkable.

7.2.1 Data Utility Metrics

A trivial solution to the trail unlinkability problem is to set $\omega_1{}' = \omega_2{}' = \cdots = \omega_{|C|}{}' = \emptyset$. Clearly, such a solution is undesired, since no protected data is disclosed. Thus, we need a model of data utility. There are a number of measures of utility that are available. For instance, in previous k-anonymity and k-ambiguity research, a popular utility model is minimization of the total number of suppressions.



Ψ			
ψ_1	ψ_2	ψ_3	ψ_4
Ali	Ali	Ali	Bob
Bob	Bob	Charlie	Charlie
Charlie	Dan	Dan	Dan

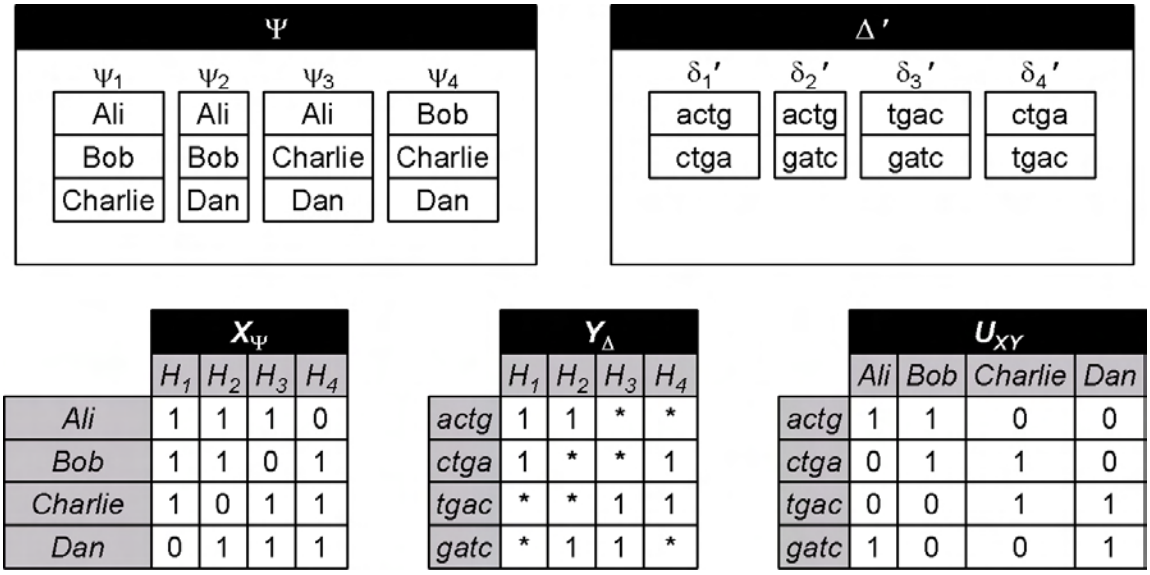
Δ			
δ_1	δ_2	δ_3	δ_4
actg	actg	actg	ctga
ctga	ctga	tgac	tgac
tgac	gatc	gatc	gatc

	X_Ψ			
	H_1	H_2	H_3	H_4
Ali	1	1	1	0
Bob	1	1	0	1
Charlie	1	0	1	1
Dan	0	1	1	1

	Y_Δ			
	H_1	H_2	H_3	H_4
actg	1	1	1	0
ctga	1	1	0	1
tgac	1	0	1	1
gatc	0	1	1	1

	U_{XY}			
	Ali	*Bob*	*Charlie*	*Dan*
actg	1	0	0	0
ctga	0	1	0	0
tgac	0	0	1	0
gatc	0	0	0	1

(a) Original Databases: 1-unlinkable



Ψ			
ψ_1	ψ_2	ψ_3	ψ_4
Ali	Ali	Ali	Bob
Bob	Bob	Charlie	Charlie
Charlie	Dan	Dan	Dan

Δ′			
δ_1'	δ_2'	δ_3'	δ_4'
actg	actg	tgac	ctga
ctga	gatc	gatc	tgac

	X_Ψ			
	H_1	H_2	H_3	H_4
Ali	1	1	1	0
Bob	1	1	0	1
Charlie	1	0	1	1
Dan	0	1	1	1

	Y_Δ			
	H_1	H_2	H_3	H_4
actg	1	1	*	*
ctga	1	*	*	1
tgac	*	*	1	1
gatc	*	1	1	*

	U_{XY}			
	Ali	*Bob*	*Charlie*	*Dan*
actg	1	1	0	0
ctga	0	1	1	0
tgac	0	0	1	1
gatc	1	0	0	1

(b) Suppressed Databases: 2-unlinkable

Figure 7.1: Suppression in DNA databases to transform trails from (a) 1-unlinkable to (b) 2-unlinkable.

For this research, we investigate two measures of utility: 1) deduplication and 2) location participation:

Deduplication The first measure is based on the deduplication problem in record linkage [123, 154]. Informally, we define the deduplication goal as a utility metric, such that the goal is to maximize the number of distinct records that are released. Formally, our goal is to minimize the following the equation:

$$\left|\bigcup_{c\in C}\omega_c\right| - \left|\bigcup_{c\in C}\omega_c{}'\right|. \tag{7.1}$$

A real world scenario this constraint addresses is the construction of a de-identified data research repository. For such a repository, we assume only one copy of a data element is needed, but we must supply the recipient with the place of collection.

Location Participation The second measure of utility is an extension of deduplication in which data holders wish to maximize the number of locations that release data. Thus, in addition to the deduplication constraint in Equation 7.1, we wish to maximize the number of locations that get to share data. Formally, we want to minimize the following equation:

$$|\{c|\omega_c = \emptyset\}|. \tag{7.2}$$

The scenario this constraint addresses is the basis for a revenue-driven model, in which compensation is provided to each location for the disclosure of his information.

7.3 Unlinking Algorithms

In this section we present several heuristic-based algorithms to transform distributed databases via suppression, such that the resulting trail matrices are k-unlinkable. Specifically, each algorithm accepts as input the set of partitioned databases Γ and Ω, as well as the protection parameter k. Given the inputs, each algorithm returns a set of databases Ω' to be disclosed in place of Ω, such that the trail matrices constructed from Γ and Ω' are k-unlinkable.

In essence, we protect elements in Ω' by guaranteeing unlinkability to elements in Γ. Therefore, in the following descriptions, we refer to elements in Γ as *protectors* and elements in Ω' as *protectees*.

7.3.1 Greedy-Dedup

The first algorithm is named *Greedy-Dedup*, with pseudocode shown in Algorithm 7. Here, we provide a brief walkthrough. Step 1 of the algorithm is for initialization purposes. This step instantiates and assigns all databases in Ω' to empty sets. In doing so, we default to telling the locations no data can be disclosed. Similarly, in Step 2 we make a copy of the protectors, which will be used for allocation purposes. Next, Step 3 suppresses all data from locations with less than k protectors via the *Explicit-Clean* procedure (Algorithm 8). The intuition behind this step is that every protectee released by a location with less than k protectors is guaranteed to be linked to less than k elements in the link matrix. Since disclosed protectees are deduplicated, it can be determined that the aforementioned protectees are guaranteed to violate k-unlinkability without having to enumerate the set of maximum matchings.

Algorithm 7 Greedy-Dedup(Γ, Ω, k)

Input: $\Gamma = \{\gamma_1, \ldots, \gamma_{|C|}\}$ and $\Omega = \{\omega_1, \ldots, \omega_{|C|}\}$, partitioned databases from a set of locations C; k, an integer specifying the protection parameter.

Output: $\Omega' = \{\omega_1{}', \ldots, \omega_{|C|}{}'\}$, a set of databases.

1: $\omega_1{}' \leftarrow \{\}, \ldots, \omega_{|C|}{}' \leftarrow \{\}$ //*Initialize Ω' to null sets*
2: $\Gamma' \leftarrow \Gamma$
3: $\{\Gamma', \Omega\} \leftarrow$ *Explicit-Clean*(Γ', Ω, k)
4: **while** $\forall c \in C, |\gamma_c{}'| > 0$ **do** //*Select location with smallest non-null database*
5: $p \leftarrow \arg\min_{c \in C} |\gamma_c{}'| \geq 0$
6: **let** $\omega_p{}' \leftarrow \{\min(|\omega_p|, |\gamma_p{}'|)$ elements in ω_p in the least number of databases in $\Omega\}$
7: **let** ${\gamma_p}^k \leftarrow \{\max(|\omega_p{}'|, k)$ elements in $\gamma_p{}'$ in the least number of databases in $\Gamma'\}$
8: $\{\Gamma', \Omega\} \leftarrow$ *Reduce*$({\gamma_p}^k, \omega_p{}', \Gamma', \Omega)$
9: $\{\Gamma', \Omega\} \leftarrow$ *Explicit-Clean*(Γ', Ω, k) //*Correct for new constraints*
10: **end while**
11: **return** Ω'

During steps 4-10, data from Ω is iteratively transferred to Ω'. In each iteration, the location with the smallest remaining database of at least k protectors is allocated the rights to its protectees. Once protectees are allocated, the location is assigned at least as many, and at minimum k, protectors. Elements are allocated to a location using a maximum likelihood estimation. From a probabilistic perspective, the elements selected have the lowest probability of being observed in any randomly chosen submission at the current iteration. This probability is unweighted and calculated as the inverse of the number of databases a sample appears within. Then, before the next iteration, the allocated protectors

Procedure 8 Explicit-Clean(Γ, Ω, k)

Input: See Greedy-Dedup.
Output: Γ, Ω, a set of databases.

//Remove infeasible datasets (either less than k protecting elements or no elements need to be protected at this location)

1: **for** each location c in C **do**
2: **if** $|\gamma_c| < k$ OR $|\omega_c| \equiv 0$ **then**
3: $\gamma_c \leftarrow \{\}$
4: $\omega_c \leftarrow \{\}$
5: **end if**
6: **end for**
7: **return** $\{\Gamma, \Omega\}$

Procedure 9 Reduce($GammaSuppress, OmegaSuppress, \Gamma, \Omega$)

Input: $GammaSuppress, OmegaSuppress$, data to suppress in Γ and Ω, respectively; Γ, Ω: See Greedy-Dedup.
Output: Γ, Ω, a set of partitioned databases.

//Remove data that has been assigned

1: **for** $j \leftarrow 1$ to $|C|$ **do**
2: $\omega_j \leftarrow \omega_j \setminus OmegaSuppress$ *//"$\setminus$" represents set remainder*
3: $\gamma_j \leftarrow \gamma_j \setminus GammaSuppress$
4: **end for**
5: **return** $\{\Gamma, \Omega\}$

and protectees are removed from consideration for any other location using the *Reduce* procedure (Procedure 9).

For illustrative purposes, in Figure 7.2(a), we depict two trail matrices X_Γ and Y_Ω. In Figure 7.2(c) we show the trail matrices X_Γ and $Y_{\Omega'}$ that result from *Greedy-Dedup*. These matrices will be used as examples throughout the following sections.

Correctness

The output from *Greedy-Dedup* satisfies k-unlinkability (Theorem 6). To prove this claim, we begin by proving that the output from the algorithm is deduplicated (Lemma 8) for all allocated protectees (i.e., elements in Ω').

Lemma 8 (Allocated Protectees are Deduplicated). *If Ω' is the output of Greedy-Dedup, then $\forall c_i, c_j (i \neq j) : {\omega_i}' \cap {\omega_j}' = \emptyset$.*

PROOF. Elements are allocated to ${\omega_i}'$ during one iteration of *Greedy-Dedup* only (Line 5). After allocation, ${\omega_i}'$ is suppressed (Line 8) from every dataset in Ω'. □

To prove *Greedy-Dedup* satisfies k-unlinkability, it suffices to prove L_{XY} is k-unlinkable (See Definition 21). To construct this proof, let ${\gamma_i}^k$ be the set of protectors allocated to c_i by *Greedy-Dedup* in Line 7. First, as a corollary to Lemma 8, it can be validated the set of allocated protectors are deduplicated as well.

Corollary 4 (Allocated Protectors are Deduplicated). *$\forall c_i, c_j, i \neq j \in C$ Greedy-Dedup allocates ${\gamma_i}^k$ and ${\gamma_j}^k$, such that ${\gamma_i}^k \cap {\gamma_j}^k = \emptyset$.*

PROOF. See Lines 7 and 8 of *Greedy-Dedup*. □

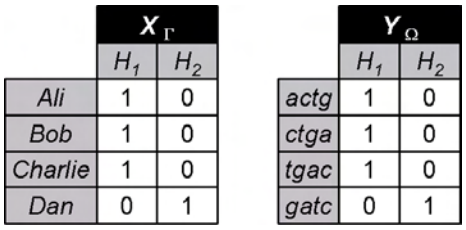
Next, we state several additional lemmas that will assist in our proof of k-unlinkability. Specifically, in the output of *Greedy-Dedup*: no location is allocated fewer protectors than protectees (Lemma 9) and no location that has been allocated protectees is allocated less than k protectors (Lemma 10).

Lemma 9 (No More Protectees Than Protectors). *$\forall c_i \in C$, Greedy-Dedup outputs $|{\omega_i}'| \leq |{\gamma_i}^k|$.*

PROOF. From Line 7 of *Greedy-Dedup*, we know $|{\omega_i}'| \leq |{\gamma_i}^k|$. Since elements are allocated to ${\omega_i}'$ during one iteration only, this inequality holds true after completion of the loop. □

Lemma 10 (At Least k Protectors). *$\forall c_i \in C$, Greedy-Dedup outputs $|{\gamma_i}^k| \geq k$ when $|{\omega_i}'| > 0$.*

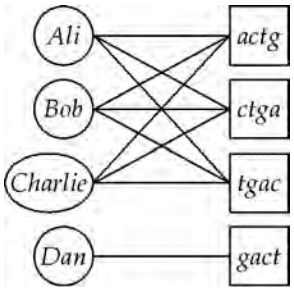
PROOF. Elements are allocated to ${\gamma_i}^k$ only when ${\omega_i}'$ is non-null. When ${\gamma_i}^k$ is allocated elements, we know $|{\omega_i}'| > 0$ (Line 6) and $|{\gamma_i}^k| \geq k$. □



	X_Γ	
	H_1	H_2
Ali	1	0
Bob	1	0
Charlie	1	0
Dan	0	1

	Y_Ω	
	H_1	H_2
actg	1	0
ctga	1	0
tgac	1	0
gatc	0	1

(a) Original Matrices

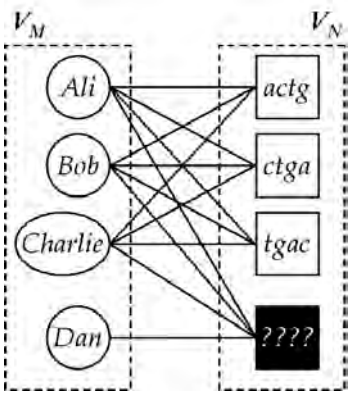


(b) Original Graph

	X_Γ	
	H_1	H_2
Ali	1	0
Bob	1	0
Charlie	1	0
Dan	0	1

	$Y_{\Omega'}$	
	H_1	H_2
actg	1	*
ctga	1	*
tgac	1	*
gatc	*	*

(c) Matrices after *Greedy-Dedup*



(d) Graph after *Greedy-Dedup*

Figure 7.2: Trail matrices, and their corresponding bipartite graphs, as they are transformed by *Greedy-Dedup* to satisfy 3-unlinkability.

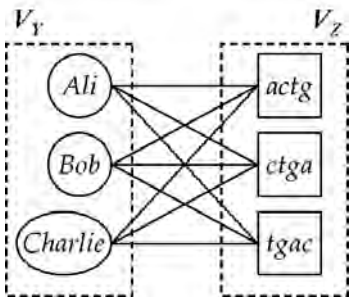


Figure 7.3: Graph of data allocated to hospital H_1.

By leveraging Lemmas 9 and 10, we find each location's allocated data satisfies k-unlinkability when considered independently of the data allocated to the other locations. We derive this finding for Lemma 11.

Lemma 11 (Allocated Data is k-Unlinkable). *Let L_i be the link matrix for the data allocated to c_i by Greedy-Dedup. If $|{\gamma_i}^k| \geq |\omega_i{}'|$, then L_i is k-unlinkable.*

PROOF. Let M_i and N_i be $|{\gamma_i}^k| \times |C|$ trail matrices. For N_i, every row is equivalent, with a 1 in column i, and a * in every other column. Similarly, for M_i, there are $|\omega_i{}'|$ rows with a 1 in column i, and a * in every other column. Moreover, for M_i, the remaining $|{\gamma_i}^k| - |\omega_i{}'|$ rows have a * in every column.

It follows that every cell in matrix L_i has a value of 1. Furthermore, since $|{\gamma_i}^k| \geq k$ (Lemma 10), every element in $\omega_i{}'$ and ${\gamma_i}^k$ is matched to at least k different elements in the set of maximum matches. Thus, L_i satisfies k-unlinkability. □

A useful corollary of Lemma 11 is that the corresponding bipartite graph is k-unlinkable.

Corollary 5 (Allocated Graph is k-Unlinkable). *Let $H_i = ({\gamma_i}^k \cup \omega_i{}', F_i)$ be a complete bipartite graph, such that $F_i = {\gamma_i}^k \times \omega_i{}'$. H_i satisfies k-unlinkability.*

As an example of Lemma 11 and Corollary 5, consider Figure 7.3. In this figure, we depict the samples allocated to hospital H_1 from the trail matrices in Figure 7.2(a) after step 10 of *Greedy-Dedup*. Note the complete bipartite graph of size 3.

Continuing with the graph representation, it can be proved the union of the bipartite graphs for allocated data is k-unlinkable. This is stated in Lemma 12.

Lemma 12 (Allocated Data Union is k-Unlinkable). *Let $H = (V_Y \cup V_Z, F) = \bigcup_{c_i \in C} H_i$. Graph H is k-unlinkable.*

PROOF. Recall, allocated data is deduplicated (Lemma 8 and Corollary 4).

As such, every pair of graphs H_i, H_j is vertex-disjoint, such that they have no vertices in common, and is a different connected component in H. Since each connected component satisfies k-unlinkability (Corollary 5), the graph H must satisfy k-unlinkability. □

Unfortunately, Lemma 12 is only sufficient to claim the output from *Greedy-Dedup* satisfies k-unlinkability in a very specific case. The reason why the Lemma is not generally applicable is that the set of disclosed databases corresponds to Ω' and Γ, the latter of which does not equal $\{{\gamma_1}^k, \ldots, {\gamma_{|C|}}^k\}$. So, to prove k-unlinkability, we first show that any graph that extends H by adding additional edges (Lemma 13) does not violate k-unlinkability.

Lemma 13 (Intersecting Allocated Data Union is k-Unlinkable). *Let H be defined as in Lemma 12. Let $I = (V_Y \cup V_Z, D)$ be a bipartite graph, such that $F \subset D$. Graph I is k-unlinkable.*

PROOF. Let T_I and T_H be the set of maximum matchings in graphs I and H, respectively. The addition of edges to a graph does not change the set of pre-existing maximum matchings, so $T_H \subseteq T_I$. Thus, every vertex remains k-unlinkable. □

Lemma 13 brings us closer to general applicability. It accounts for links that exist between an allocated protectee and a protector that is disclosed by a location that was not allocated the protector. However, the lemma does not account for any of the additional nodes in $\Gamma \backslash \{{\gamma_1}^k, \ldots, {\gamma_{|C|}}^k\}$. To do so, we first note that the resulting graph from *Greedy-Dedup* is an extension of the graph for allocated data from Lemma 13.

Lemma 14 (*Greedy-Dedup* Graph Extends Allocated Graph). *Let graph $I = (V_Y \cup V_Z, D)$ be defined as in Lemma 13. Let Ω' be the output of Greedy-Dedup(Γ, Ω, k). Let $M = \bigcup_{c \in C} \gamma_c$ and $N = \bigcup_{c \in C} {\omega_c}'$. Let M_Γ and N_Ω be $|M| \times |C|$ trail matrices with link matrix L_{MN}. Let $G = (V_M \cup V_N, E)$ be the corresponding bipartite graph, where $E = \{e_{mn} | L_{MN}[m, n] = 1\}$. $I \subseteq G$.*

PROOF. Since $\forall c \in C : {\gamma_c}^k \subseteq \gamma_c$, it is true that $V_Y \subseteq V_M$. Similarly, since Ω' is the set of disclosed protectees, $V_Z \subseteq V_N$. Moreover, because Ω' is deduplicated (Lemma 8), every edge in D is in E as well. □

Figure 7.4 provides an example of how the *Greedy-Dedup* solution extends the allocated graph in Figure 7.3 and maintains k-unlinkability, which in this case k = 3.

Now, since I satisfies k-unlinkability and G is an extension of I, then G satisfies k-unlinkability when none of the additional vertices and edges violate k-unlinkability. Therefore, we prove the output of *Greedy-Dedup* is k-unlinkable in Theorem 6 by extending Lemma 14 to include all elements from the disclosed databases Γ.

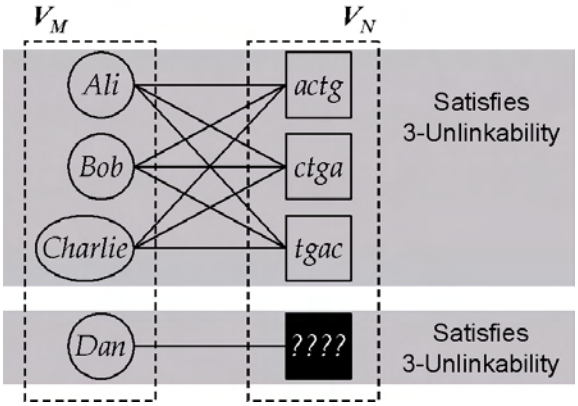


Figure 7.4: Union of maxim matching graphs after *Greedy-Dedup* satisfies k-unlinkability.

Theorem 6 (*Greedy-Dedup* Output is k-Unlinkable). *Let Ω' be the output of Greedy-Dedup(Γ, Ω, k). The link matrix for Γ and Ω' is k-unlinkable.*

PROOF. Let M, N, M_Γ, $N_{\Omega'}$, $G = (V_M \cup V_N, E)$, and $I = (V_Y \cup V_Z, D)$ be as defined in Lemmas 13 and 14.

First, we note every vertex in the set $V_N \backslash V_Z$ represents a null trail (i.e., only *'s). This is true because the corresponding elements are never disclosed. As such there exists an edge in G between every vertex in $V_N \backslash V_Y$ and V_M.

Second, we note every vertex in the set $V_M \backslash V_Y$ represents a protector that was not allocated during *Greedy-Dedup*. Call this set of vertices V_X. If a vertex in V_X corresponds to a sample that is disclosed by a location allocated protectees, then it extends the graph in Corollary 5 by making it a larger complete bipartite graph. As such, it does not affect the linkability. However, if a vertex in V_X is disclosed by a location that is not allocated protectees, then it can only be linked to vertices in the set $V_N \backslash V_Z$, the corresponding elements of which are never disclosed. Therefore, G, and its corresponding link matrix, satisfies k-unlinkability. □

Complexity

The computational complexity of *Greedy-Dedup* can be calculated as follows. Steps 1, 2, and 3 are initializations that each require O($|C|$) steps. Next, the loop in Steps 4-10 proceeds for a maximum of $|C|$ steps, and at each iteration a location is allocated a set of elements. We assume that at each iteration, the elements are ordered by frequency and thus we can simply pop off the allocated elements in constant time. During the iteration, the allocated dataset is removed from all input datasets, which are checked for explicit violations. Though *Reduce* and *Explicit-Clean* are written as separate procedures, this is

for clarity. They can be combined into one method, which internally iterates a maximum of $|C|$ steps. Thus, upper bound of *Greedy-Dedup* is O($|C|^2$) complexity.

7.3.2 Force-Dedup

The second algorithm we present is called *Force-Dedup*. Akin to *Greedy-Dedup*, it uses a greedy heuristic, with the same probabilistic basis, to allocate elements to locations. However, unlike the previous algorithm, *Force-Dedup* splits the allocation process into two phases, which we call the a) *force* and b) *boost* phases. First, in the force phase, *Force-Dedup* enforces a limit on the number of protectees allocated to a maximum of k. Specifically, each location is allocated k protectors until either every location has been allocated k or no more locations can be allocated k samples. The intuition behind the adoption of a force phase is to increase the number of locations that are allowed to disclose data.

When no more locations can be assigned k protectees using the greedy process, the algorithm enters the boost phase. In a manner similar to the greedy allocation process of *Greedy-Dedup*, the algorithm enlarges the set of protectees allocated to each location. Recall, these locations were allocated k protectors in the force phase. Now, each location is allocated as many protectors as protectees, again using the maximum likelihood estimation.

Complexity

The complexity analysis of *Force-Dedup* is similar to *Greedy-Dedup*. The primary difference is the additional loop for the allocation of elements in the force phase. Assuming k is a relatively small constant (i.e., $k << |S|$), this requires O($|C|^2$) steps. Thus, the complexity of *Force-Dedup* remains O($|C|^2$).

Correctness

The main difference between *Greedy-Dedup* and *Force-Dedup*, is that the latter incorporates a second heuristic to increase the number of locations that are allocated data. Here, we sketch the k-unlinkability correcteness of *Force-Dedup*. The initial allocation of data to locations in the *force* phase does not change the fact that allocated data is deduplicated. After the end of the *force* phase, every location was allocated either k samples or 0 samples. Also note that at this point, the allocated data is deduplicated. So if only data allocated during *force* phase was disclosed, it would satisfy k-unlinkability.

Algorithm 10 Force-Dedup(Γ, Ω, k)

Input: See Greedy-Dedup.
Output: See Greedy-Dedup.

1: ${\omega_1}' \leftarrow \{\}, \ldots, {\omega_{|C|}}' \leftarrow \{\}$ //*Initialize Ω' to null sets*
2: $\Gamma' \leftarrow \Gamma$
3: $\{\Gamma', \Omega\} \leftarrow$ *Explicit-Clean*(Γ', Ω, k)
4: $AVAIL \leftarrow C$
//*FORCE PHASE: Allocate up to k elements per location*
5: **while** $\forall c \in AVAIL, |\gamma_c| > 0$ **do**
6: $p \leftarrow \arg\min_{c \in AVAIL} |\gamma_c| \geq k$
7: $AVAIL \leftarrow AVAIL \setminus \{p\}$
8: **let** ${\omega_p}' \leftarrow \{\min(|\omega_p|, k|)$ elements in ω_p in the least number of databases in $\Omega\}$
9: **let** ${\gamma_p}^k \leftarrow \{k$ elements in γ_p in the least number of databases in $\Gamma\}$
10: $\{\Gamma', \Omega\} \leftarrow Reduce({\gamma_p}^k, {\omega_p}', \Gamma', \Omega)$
11: **end while**
12: $AVAIL \leftarrow C \setminus AVAIL$ //*Use locations previously allocated data*
//*BOOST PHASE: Allocate remaining elements*
13: **while** $\forall c \in AVAIL, \min(|{\gamma_c}'|, |\omega_c|) > 0$ **do**
14: $p \leftarrow \arg\min_{c \in AVAIL} |{\gamma_c}'| \geq 0$
15: **let** ${\omega_p}' \leftarrow {\omega_p}' \cup \{\min(|\omega_p|, |{\gamma_p}'|)$ elements in ω_p in the least number of databases in $\Omega\}$
16: **let** ${\gamma_p}^k \leftarrow \{|{\gamma_p}'| - |{\omega_p}'|$ elements in ${\gamma_p}'$ in the least number of databases in $\Gamma\}$
17: $\{\Gamma, \Omega\} \leftarrow Reduce({\gamma_p}^k, {\omega_p}', \Gamma', \Omega)$
18: **end while**
19: $\Omega' \leftarrow$ *Final-Clean*(Ω', Γ, k) //*Correct for null trail constraints*
20: **return** Ω'

Next, in the *boost* phase, we only allocate data to the locations that have already satisfied k-unlinkability. The allocation of more samples to these locations, again in a deduplicated manner, can not make any of these locations violate k-unlinkability.

In theory, any output that is produced by *Force-Dedup* could have been output by *Greedy-Dedup*. The difference is that the two algorithms use different heuristics.

7.4 Discussion: Greedy, Not Optimal

The greedy heuristics employed by the presented algorithms do not guarantee optimal solutions for the utility metrics. We suspect there are more robust heuristics. For instance,

in the presented set of unlinking algorithms, a protector is allocated to one location only. However, this is unnecessarily restrictive. In actuality, a protector can be allocated to protect k protectees - regardless of which location these protectees are located.

Nonetheless, given k-unlinkability's relationship to protection methods k-anonymity and k-ambiguity, we suspect that optimal solutions for k-anonymity may be part of the NP-hard class of problems. In the future, we intend to investigate the complexity of this problem.

7.5 Conclusions

In this chapter, we defined a general solution space for k-unlinkability with respect to trail matrices. In essence, we stated that k-unlinkability for trail matrices can only be achieved by making unambiguous values ambiguous. Then, we investigated a more specific problem based upon real world observations, in which only one trail matrix is permitted to have information made ambiguous. Given this constraint, we presented two heuristic-based deduplication algorithms to achieve k-unlinkability for trails without enumerating the set of maximum matchings in a graph by using deduplication.

This chapter considered the scenario when all locations are able to openly collaborate. In the following chapter, we investigate how to generate k-unlinkable trail matrices when locations are not permitted to view the contents of each others databases before disclosure.

Chapter 8

Secure Trail Anonymization

In Chapter 7 we assumed data holders could collaborate to disclose unlinkable databases. Unfortunately, this assumption is not always plausible. In some cases, there exists a lack of trust and a data holder may not allow other data holders to view the contents of its private database. In other instances, data holders are legally restricted from discussing the contents of their private databases. For example, with respect to health information, the HIPAA Privacy Rule does not necessarily allow covered providers to divulge sensitive personal health information until the data is sufficiently de-identified. As a consequence, a fundamental challenge to the deployment of methods to prevent trail re-identification stems from a lack of support for communication between data holders. Specifically, open communication is hindered because it can comprise the anonymity of the records the data holders must protect.

In this chapter, we present a solution to prevent trail re-identification in the midst of policy constraints. We ensure k-unlinkability is satisfied while adhering to defined policy regulations. This is achieved via a multiparty computation protocol in which data holders collaborate with a trusted third party through encrypted versions of sensitive data. After completion of the protocol, no recipient of de-identified data, including the data holders and the third party, can achieve re-identification to less than k identities via observed trails.

In Chapter 9 we evaluate the effectiveness of the protocol on real world data.

8.1 Security Framework

Data holding locations need to compare their databases without revealing the contents to render trails unlinkable. To achieve such comparisons, we employ the assistance of a semi-trusted third party (*sTTP*). The *sTTP* is trusted to perform computations on encrypted databases and correctly respond to each location, but the *sTTP* is not trusted to view plain-

text data during the execution of the protocol. In other words, the data holders must submit their data to the *sTTP* such that it is comparable to other data holders' submissions, while guaranteeing the *sTTP* can not determine the plaintext contents.

In Appendix C, however, we present a secure multiparty computation framework for the centralized data analysis of lists [89]. The framework endows the *sTTP* with the power to analyze encrypted databases and provide differential responses. In this chapter, we define a specific implementation of the framework for our anonymization procedure. We call the protocol the Secure TRail ANONymizer, or STRANON. Though details of the multiparty framework are beyond the scope of this chapter, a basic understanding of the framework is beneficial to characterize certain constraints.

8.1.1 Basics

The framework for secure multiparty computation that we employ is based on keyed commutative hashing. Each location c_i hashes data x using hash key ϵ_i and a function h that satisfies

$$h(h(x, \epsilon_i), \epsilon_j) = h(h(x, \epsilon_j), \epsilon_i),$$

for any subset and ordering of the keys $\epsilon_1, \ldots, \epsilon_{|C|}$. By choosing keys appropriately [10], it can be shown that $h(h(a, \epsilon_i), \epsilon_j) = h(h(b, \epsilon_j), \epsilon_i)$, only if $a = b$. As a result, the *sTTP* can not make any false equivalences (i.e., $h(h(a, \epsilon_i), \epsilon_j) = h(h(b, \epsilon_j), \epsilon_i)$ and $a \neq b$), nor can the *sTTP* make any false inequalities (i.e., $h(h(a, \epsilon_i), \epsilon_j) \neq h(h(b, \epsilon_j), \epsilon_i)$ and $a = b$).

These properties are satisfied when keys are chosen according to a variant of modular exponentiation as is used in RSA cryptosystems, so $h(x, y) = x^y \text{mod}(n)$.[1] Since RSA is feasible to satisfy a commutative property, the schema can be converted into an asymmetric keyed cryptosystem when keys are chosen appropriately (see Appendix C). To do so, each encryption key ϵ_i is paired with a decryption key κ_i, such that

$$h(h(h(h(x, \epsilon_i), \epsilon_j), \kappa_i), \kappa_j) = x.$$

Note, unlike public-key cryptosystems, such as RSA, where ϵ_i is made public and κ_i is kept private, we specify a private-key cryptosystem. Neither the encryption nor the decryption key is known beyond location c_i.

[1] Alternatives to an RSA basis exist, such as the Pohlig-Hellman algorithm [113]. Though we recognize this possibility, for simplicity we present the RSA basis.

8.1.2 Communication Protocol

We now walk through the STRANON protocol, which is detailed in Algorithm 11. Recall, each location $c_i \in C$ maintains a private database τ_i. Before the multiparty aspect of the protocol begins, each c_i partitions τ_i into databases γ_i and ω_i and publishes γ_i. The latter step of publication is based on real world health systems, where identifiable hospital databases, such as discharge information, are readily available [134].

In the first collaborative part of the protocol, each location encrypts every location's data items with its encryption key. Next, the fully encrypted databases are sent to the *sTTP*. Upon reception of all databases, the *sTTP* runs a modified version of a deduplication algorithm, generally referred to as *DEDUP*, from Chapter 7. Then, the *sTTP* responds to each location with a dataset of encrypted values to disclose. Finally, each location decrypts every dataset that was returned by the *sTTP*, and the values are disclosed.

As presented, STRANON is susceptible to various security breaches. In Appendix C we show it can be made resilient to various attacks without amendment to the embedded *DEDUP* method. For instance, it can be shown that during execution of the protocol no set of locations can collude to learn the plaintext contents of a non-colluding location's database.

The computational complexity of the STRANON framework in terms of the number of hashes and bandwidth is detailed in Appendix C. To summarize, the number of encryptions and decryptions that must be performed is quadratic, or O($|C|^2$). However, since each location performs these operations in parallel, the runtime is linear, or O($|C|$). The total amount of bandwidth necessary to execute STRANON is O($\max_{c \in C} |\omega_i| \cdot |C|$).

8.2 Unlinkability Constraints

In this section we delve into details of the *DEDUP* method. First, we consider the operational constraints. Second, we focus on constraints regarding feasible solutions that can be generated via the operational constraint. Due to the fact that there exist recipients with private knowledge, we need to satisfy a more restrictive form of k-unlinkability. In this respect, we must move beyond traditional secure multiparty computation proofs to account for knowledge retained by the *sTTP* and the participating data holders. Specifically, the *sTTP* gains private knowledge during execution of the *DEDUP* procedure. Similarly, each data holder has prior knowledge in the form of its private database.

In Chapter 7 the set of locations collaborated to execute the protection methods over the raw databases. Now the *sTTP* must execute the *DEDUP* algorithm in an encrypted space. With respect to the cryptosystem, we restate the operational constraints in the encrypted space where the *sTTP* must execute *DEDUP*. In Figure 8.1 we present an example of par-

Algorithm 11 STRANON (C, $sTTP$)

for each $c_i \in C$ **do** //*Each location partitions and discloses identifiable data*
  c_i generates $\{\gamma_i, \omega_i\} \leftarrow \tau_i$
  c_i discloses γ_i
end for
for each $c_i \in C$ **do** //*Every location encrypts each unpublished database*
  $M_i \leftarrow h(\omega_i, \epsilon_i)$
  for each $c_j \in C$, $i \neq j$ **do**
    c_i sends M_i to c_j
    c_j sends $M_i \leftarrow h(M_i, \epsilon_j)$ to c_i
  end for
end for
Each $c_i \in C$ sends M_i to *sTTP*
//*Third party runs deduplication method on encrypted databases*
sTTP executes $DEDUP(\Gamma, \{M_1, \ldots, M_{|C|}\}, k)$ to generate encrypted datasets $\{N_1, \ldots, N_{|C|}\}$ //*The DEDUP method is substituted with either Algorithm 12: Greedy-Dedup-Secure or Algorithm 14: Force-Dedup-Secure*
sTTP sends N_1 to c_1, N_2 to c_2, . . . , and $N_{|C|}$ to $c_{|C|}$
for each $c_i \in C$ **do** //*Everyone decrypts all responses from the third party and discloses k-unlinkable data*
  for each $c_j \in C$, $i \neq j$ **do**
    c_i sends N_i to c_j
    c_j sends $N_i \leftarrow h(N_i, \kappa_j)$ to c_i
  end for
  c_i discloses $\omega_i{}' \leftarrow h(N_i, \kappa_i)$
end for

titioned identified and de-identified databases. We will return to this example throughout this chapter.

8.2.1 Operational Constraints

Let $h(\Omega) = \{h(\omega_1), \ldots, h(\omega_{|C|})\}$ be the set of encrypted datasets sent to the third party. According to the STRANON protocol, the *sTTP* runs *DEDUP*(Γ, $h(\Omega)$, k) to generate $h(\Omega') = \{h(\omega_1{}'), \ldots, h(\omega_{|C|}{}')\}$, which are personalized encrypted responses to send back to the appropriate locations. As in Chapter 7, we do not permit the injection of false information into the system. As a result, the operational constraint on $DEDUP$ as executed by the third party is such that for all $c_i \in C$, $\omega_i{}' \subseteq \omega_i$. This constraint states that the only

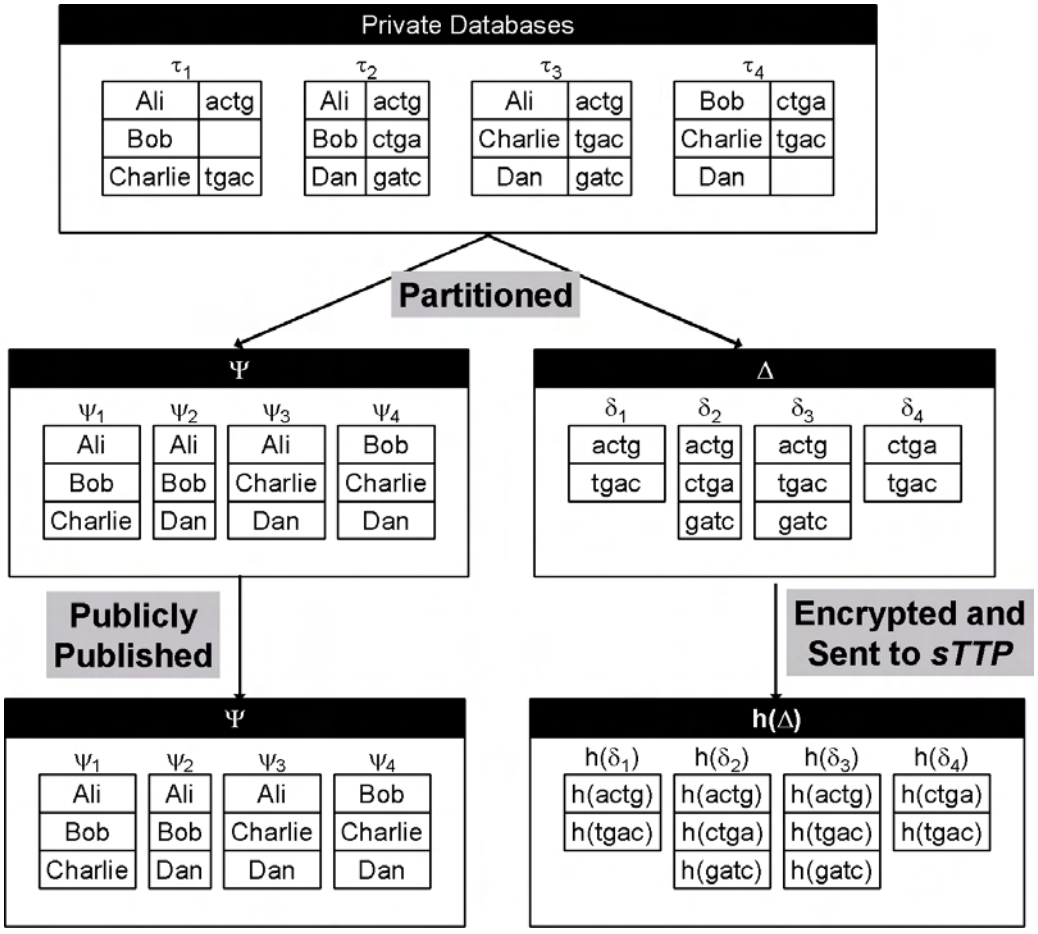


Figure 8.1: Identified and de-identified databases disclosed by four hospitals.

X_Ψ

	H_1	H_2	H_3	H_4
Ali	1	1	1	0
Bob	1	1	0	1
Charlie	1	0	1	1
Dan	0	1	1	1

$Y_{h(\Delta)}$

	H_1	H_2	H_3	H_4
h(*actg*)	1	1	1	*
h(*ctga*)	*	1	0	1
h(*tgac*)	1	0	1	1
h(*gatc*)	*	1	1	*

Figure 8.2: Trail matrices constructed by the *sTTP*.

operation the *sTTP* is permitted to perform is suppression. Notice that when ω_i is reserved to γ_i, it follows that $\omega_i{}'$ must be reserved to γ_i. Thus, we retain the reserved property from

Definition 11 in Chapter 3.

8.2.2 Privacy Constraints

The STRANON protocol uses multiparty cryptography, so it seems appropriate to prove privacy preservation based on secure multiparty computation (SMC). Unfortunately, this type of analysis does not properly characterize formal privacy models. Traditional proofs in SMC analyze each participant's input into a protocol and what each participant views at each point during protocol execution. In the end, a protocol satisfies SMC protection when it does not reveal anything in the final result that could not be learned from the inputs or the result. However, as others [73] have observed, the final result itself can violate privacy constraints.

Therefore, in this chapter we model privacy with respect to a set of constraints. Jiang and Clifton [69] provide an intuitive definition of privacy protection proofs from a general standpoint. Informally, they state that the inferences extracted from the inputs into the protocol, the execution of a protocol over the inputs, as well as the protocol's results, must not violate privacy constraints that could be inferred from the inputs. In this framework, Jiang and Clifton [69] developed a privacy preserving protocol to construct a k-anonymous table from distributed data.

In the following sections, we define several types of privacy constraints in the form of k-unlinkability with respect to various disclosed database recipients. We show that satisfying these constraints ensures k-unlinkability, as observed by the participants.

External Constraints

In Chapter 7 the goal was to prevent a recipient from achieving trail re-identification. A recipient was not a member of the set of locations that collaborate to satisfy k-unlinkability. For reference, we call such recipients *external entities*. We say that *external k-unlinkability* is satisfied when the set of disclosed databases are protected from trail re-identification by an external recipient. This notion of external k-unlinkability is formally described in Definition 22.

Informally, with respect to STRANON, an external recipient has access to the set of identified databases in Γ and the set of de-identified databases in Ω' that were protected by the *sTTP*. External k-unlinkability is satisfied when the trails derived from these databases fail to yield re-identifications.

Definition 22 (External k-Unlinkability). *Let M_Γ and $N_{\Omega'}$ be the trail matrices derived from Γ and Ω', respectively. Let $G = \{V_M \cup V_N, E\}$ be the bipartite graph that rep-*

resents the set of links between trail matrices M_Γ and $N_{\Omega'}$. External k-unlinkability *is achieved when G satisfies k-unlinkability.*

As an example of external k-unlinkability, imagine the *sTTP* is supplied with the partitioned databases Ψ and $h(\Delta)$ from Figure 8.1. The *sTTP* suppresses information in $h(\Delta)$ to produce the disclosed trail matrices X_Ψ and $Y_{\Delta'}$ in Figure 8.3. An external recipient observes the link matrix for X_Ψ and $Y_{\Delta'}$ in Figure 8.4(a), depicted as L_{YX}. When the recipient extracts the union of maximum matchings from the link matrix, shown in matrix U_{YX} in Figure 8.4(b), he finds the trail matrices satisfy external 2-unlinkability.

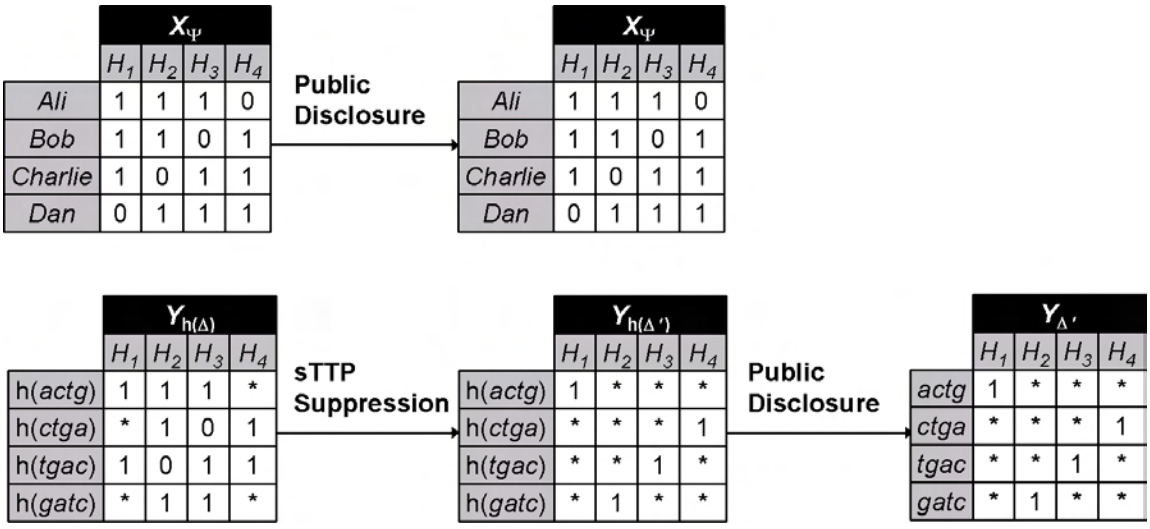


Figure 8.3: Suppressions performed by the *sTTP*.

Centralized Constraints

External k-unlinkability does not account for knowledge retained by the *sTTP*. When the *sTTP* can not use private knowledge for trail re-identification, we say *central k-unlinkability* is satisfied. This protection model is formally developed in Definition 24.

Informally, to account for the *sTTP* we need to model the information that the *sTTP* gathers during the execution of the *DEDUP* procedure in Line 13 of Algorithm 11. During execution the *sTTP* can construct a relation called *Omega-Map*, which is formally presented in Definition 23. *Omega-Map* specifies which encrypted data trails, held by the *sTTP*, can be related to which disclosed data trails.

	L_{YX}			
	Ali	Bob	Charlie	Dan
actg	1	1	1	0
ctga	0	1	1	1
tgac	1	0	1	1
gatc	0	1	1	1

(a) Link Matrix for X_Ψ and $Y_{\Delta'}$

	U_{YX}			
	Ali	Bob	Charlie	Dan
actg	1	1	1	0
ctga	0	1	1	1
tgac	1	0	1	1
gatc	0	1	1	1

(b) Union of maximum matchings between X_Ψ and $Y_{\Delta'}$

Figure 8.4: The link matrix constructed from the disclosed trail matrices in Figure 8.3 satisfies external 2-unlinkability.

Definition 23 (Omega-Map Relation). *Let* Omega-Map$(h(\Omega), h(\Omega'))$ *be a program that outputs matrix* P *with dimensions* $|N| \times |N|$*, where* $P[x, y] = 1$ *means trail* $x_{h(\Omega')}$ *may have been derived from trail* $y_{h(\Omega)}$*.*

Since *Omega-Map* is a relation, multiple trails from $N_{h(\Omega)}$ can map to the same trail in $N_{h(\Omega')}$. *Omega-Map* models the relationships between trails, but not their corresponding elements. As such, *Omega-Map*$(h(\Omega), h(\Omega'))$ is equivalent to *Omega-Map*$(h(\Omega), \Omega')$. By inspecting Γ, Ω', and $h(\Omega)$, the *sTTP* can remove links in the bipartite graph that fail to satisfy the *Omega-Map* relation. In Definition 24, we show how central k-unlinkability is satisfied from a computational perspective.

Definition 24 (Central k-Unlinkability). *Let* M *and* N *be the set of distinct elements in databases in* Γ *and in* Ω*, respectively. Let* Q *be an* $|N| \times |N|$ *matrix representing the set of links between trail matrices* M_Γ *and* $N_{h(\Omega)}$*. Let* P *be a* $|N| \times |N|$ *matrix representing the set of feasible links between* $N_{\Omega'}$ *and* $N_{h(\Omega)}$ *as derived from Omega-Map.* Central k-unlinkability *is achieved when the graph with the corresponding link matrix*

$$B = QP^t \tag{8.1}$$

satisfies k*-unlinkability.*

By Equation 8.1, it can be shown that the *sTTP* is an external recipient with additional knowledge. As a result, when trail matrices satisfy central k-unlinkability, they are guaranteed to satisfy external k-unlinkability.

Returning to the example in Figures 8.3 and 8.4, we find that the *sTTP* can construct a mapping relation called *Delta-Map*, which we depict *Delta-Map* as matrix P in Figure 8.5. Through *Delta-Map* the *sTTP* can remove links that an external recipient could not necessarily rule out. For instance, by inspecting P the *sTTP* will discover that the link between *actg* and *Bob* must not be a correct re-identification. This is because when the *sTTP* inspects matrix P he finds that the corresponding cell for the trails of *actg* and *Bob* are not really linked because $P[[1,*,*,*],[*,1,0,1]] = 0$. Thus, this link can be dropped, which means the value of the link matrix $L_{YX}[actg, Bob]$ can be changed from 1 to 0.

	P held by sTTP			
	[1,1,1,*]	[*,1,0,1]	[1,0,1,1]	[*,1,1,*]
[1,*,*,*]	1	0	0	0
[*,*,*,1]	0	1	0	0
[*,*,1,*]	0	0	1	0
[*,1,*,*]	0	0	0	1

Figure 8.5: The *Delta-Map* relation derived from Figure 8.3.

Continuing with this example, we return to Figure 8.4. Given the suppressions in Figure 8.3, the *sTTP* can construct mapping matrix P, which is an identity matrix. In addition, the *sTTP* can construct a link matrix Q for the submitted databases. With respect to the matrix multiplication model, the general structure of these matrices is shown in Figure 8.6.

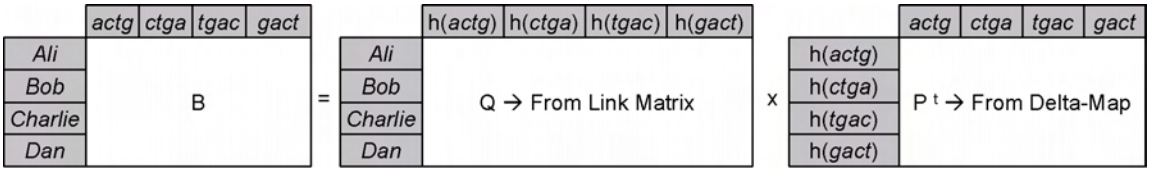


Figure 8.6: *sTTP* matrix multiplication architecture.

In combination, the *sTTP* can construct matrix B using the following matrix multiplication:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

The set of maximum matchings as observed by the *sTTP* leads to the following matrix:

$$\text{union of maximum matchings} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This result reveals all trails can be uniquely re-identified by the *sTTP*. This is despite the fact that the trail matrices satisfied external 2-unlinkability.

Contributor Constraints

Each data holder has private knowledge that it does not reveal to the *sTTP*. As a consequence, central k-unlinkability does not model knowledge retained by members of C. When data holders in the set C can not use private knowledge to achieve trail re-identification, we say that *contributor k-unlinkability* is satisfied. This notion of privacy is formally developed in Definition 25.

Informally, location $c_i \in C$ does not have access to $h(\Omega)$. Yet, it does have access to the disclosed sets of databases Γ and Ω', as well as its private database τ_i. Prior to relying upon trails for re-identification, c_i can remove elements for which it already knows the unique mappings. For c_i these mappings are provided in τ_i.

	L_{YX}			
	Ali	*Bob*	*Charlie*	*Dan*
actg	1	1	1	0
ctga	0	1	1	1
tgac	1	0	1	1
gatc	0	1	1	1

(a) Cells in black correspond to information in hospital H_1's private database

	W_1	
	Bob	*Dan*
ctga	1	1
gatc	1	1

(b) Removal of links by H_1 reveals W_1

Figure 8.7: H_1 can remove links from the link matrix. The resulting link matrix W_1 is 2-unlinkable.

Definition 25 (Contributor k-Unlinkability). *Let G be the bipartite graph that represents the set of links between trail matrices M_Γ and $N_{\Omega'}$. When $\langle m, n\rangle \in \tau_i$, then c_i drops nodes and edges in G that correspond to m and n. We call this reduced graph W_i. We say* contributor k-unlinkability *is satisfied when for all $c_i \in C$, W_i is k-unlinkable.*

In Figure 8.7 we provide an example with respect to hospital H_1. In Figure 8.7(a), the cells highlighted in black correspond to the links that the private database τ_1 reveals are true loves (i.e., are correct linkages). With this knowledge, H_1 can remove all links for *actg*, *tgac*, *Ali*, and *Charlie*. Upon doing so, however, H_1 still can not determine which are the unique re-identifications for the remaining elements. The system is 2-unlinkable with respect to H_1.

Contributor k-unlinkability takes into consideration the fact that the trail linkage of two elements is considered a re-identification only if the entity learning such knowledge was unaware of the relationship beforehand. If a data element $n \in N$ is not within private database τ_i, then c_i should not be able to independently violate k-unlinkability for n to its corresponding identity m via trails. Drawing upon our example, if hospital H_4 learns *actg* corresponds to *Ali* then a trail re-identification has occurred. This is because $\langle Ali, actg\rangle \notin \tau_4$. However, if hospital H_1 learns $\langle Ali, actg\rangle \in \tau_3$, this is not a trail re-identification because H_1 did not learn anything new. This is due to the fact that H_1 already knew $\langle Ali, actg\rangle \in \tau_1$.[2]

8.3 Secure Unlinking Algorithms

It is the *sTTP*'s burden to certify that the disclosed de-identified databases satisfy both central and contributor k-unlinkability constraints. Unfortunately, the deduplication algorithms presented in Chapter 7 do not satisfy these constraints. In the following sections, we augment the deduplication algorithms to satisfy central and contributor k-unlinkability.

8.3.1 Greedy-Dedup-Secure

First, we augment the *Greedy-Dedup* algorithm from Chapter 7. Pseudocode for the revised version, which we call *Greedy-Dedup-Secure*, is presented in Algorithm 12.

As in Chapter 7, we protect elements in Ω' by guaranteeing unlinkability to elements in Γ. Therefore, in the following description, we refer to elements in Γ as *protectors* and elements in Ω' as *protectees*.

[2] As a result, we do not model the space of all privacy concerns, only anonymity.

Algorithm 12 Greedy-Dedup-Secure($\Gamma, h(\Omega), k$)

Input: $\Gamma = \{\gamma_1, \ldots, \gamma_{|C|}\}$, the set of published databases; $h(\Omega) = \{h(\omega_1), \ldots, h(\omega_{|C|})\}$, the set of encrypted databases sent to the third party by the participating locations; k, an integer specifying the protection parameter.

Output: $h(\Omega') = \{h(\omega_1'), \ldots, h(\omega_{|C|}')\}$, the set of encrypted databases to return to c_1, ... , $c_{|C|}$, respectively.

1: $h(\omega_1') \leftarrow \{\}, \ldots, h(\omega_{|C|}') \leftarrow \{\}$ //*Initialize* $h(\Omega')$ *to null sets*
2: $\Gamma' \leftarrow \Gamma$
3: **let** $\Theta \leftarrow$ *Contributor-Clean*(Γ', $h(\Omega)$, k)
4: **let** $\{\Gamma', \Theta\} \leftarrow$ *Explicit-Clean*(Γ', Θ, k) //*See Algorithm 8 in Chapter 7*
5: **while** $\exists \theta_i \in \Theta : |\theta_i| > 0$ **do**
6: $p \leftarrow \arg\min_{c \in C} |\theta_c| \geq k$
7: **let** $h(\omega_p') \leftarrow \theta_p$
8: **let** ${\gamma_p}^k \leftarrow \{|h(\omega_p')|$ elements in γ_p in the least number of databases in $\Gamma'\}$
9: $\{\Gamma', \Theta\} \leftarrow$ *Reduce*(${\gamma_p}^k, h(\omega_p'), \Gamma', \Theta$) //*See Algorithm 9 in Chapter 7*
10: $\{\Gamma', \Theta\} \leftarrow$ *Explicit-Clean*(Γ', Θ, k) //*Correct for new constraints*
11: **end while**
12: **return** $h(\Omega')$

Algorithm 13 Contributor-Clean(Γ, $h(\Omega)$, k)

Input: $\Gamma' = \{\gamma_1', \ldots, \gamma_{|C|}'\}$, the set of data elements in published databases that are available to protect elements in $h(\Omega)$ from trail re-identification; $h(\Omega)$, k, see Greedy-Dedup-Secure.

Output: $\Theta = \{\theta_1, \ldots, \theta_{|C|}\}$, encrypted databases with cleaned pairwise non-intersections.

1: **let** $\Theta \leftarrow h(\Omega)$
//Make sure no disclosed elements violate *contributor k-unlinkability*
2: **for** all $c_i, c_j \in C$ **do**
3: **if** $\max(|\gamma_i \backslash \gamma_j|, |\gamma_i| - |h(\omega_j)|, |h(\omega_i) \backslash h(\omega_j)|) < k$ **then** //"$\backslash$" *represents set remainder*
4: $\theta_i \leftarrow \theta_i \cap \theta_j$
5: **end if**
6: **end for**
7: **return** Θ

The first line is for initialization of the databases to be allocated protectees. We default to telling the locations that no protectee can be disclosed. Next, we run a procedure

called *Contributor-Clean*, pseudocode for which is presented in Algorithm 13. The goal of *Contributor-Clean* is to guarantee that the disclosed trail matrices M_Γ and $N_{\Omega'}$ do not violate any of the contributor k-unlinkability constraints. Briefly, the *Contributor-Clean* method suppresses encrypted samples that violate contributor k-unlinkability. As such, it instantiates the temporary databases from which protectees will be drawn and allocated for disclosure. In step 4, we suppress all data from locations with less than k protectors via the *Explicit-Clean* procedure (Algorithm 8 in Chapter 7) as done for the non-secure version of the algoritm.

In lines 5 through 11, we use a greedy heuristic to allocate protectees to locations for decryption and subsequent disclosure. In line 6 we choose the smallest remaining dataset of unallocated protectees with size greater than k. Then, we choose k protectors and remove allocated samples from all locations' datasets. The algorithm terminates when no more data can be allocated.

Complexity

The complexity analysis of the *Greedy-Dedup-Secure* algorithm is akin to that of the *Greedy-Dedup* algorithm in the previous chapter. There is one primary difference. Specifically, in line 3, databases are reduced based on their intersections. This can be computed in O($|C|^2$) comparisons. Thus, since complexity for *Greedy-Dedup* is O($|C|^2$), the complexity for *Greedy-Dedup-Secure* remains O($|C|^2$).

We now prove that execution of the *Greedy-Dedup-Secure* algorithm satisfies central and contributor k-unlinkability.

Central k-Unlinkability

First, we prove that the output from *Greedy-Dedup-Secure* satisfies central k-unlinkability (Theorem 7). To do so, we state several basic aspects of the output from *Greedy-Dedup-Secure*. First, in Lemma 15 we show that the data output from *Greedy-Dedup-Secure* is deduplicated.

Lemma 15 (GDS Deduplicates). *If Ω' is output from Greedy-Dedup-Secure, then it is deduplicated.*

PROOF. The primary difference between *Greedy-Dedup-Secure* and *Greedy-Dedup* is the incorporation of *Contributor-Clean*. As a result, any output of Ω' from *Greedy-Dedup-Secure* could also be output of *Greedy-Dedup*. It follows from Lemma 8 in Chapter 7 that $\forall c_i, c_j \in C : \omega_i{}' \cap \omega_j{}' = \emptyset$. □

Second, each non-null database in Ω' that is output from *Greedy-Dedup* must consist of at least k elements (Lemma 16).

Lemma 16 (GDS Allocates at least k or 0). *If Ω' is output from* Greedy-Dedup-Secure, *then $\forall \omega_i' \in \Omega'$: $|\omega_i'| \geq k$ or $|\omega_i'| = 0$.*

PROOF. All databases in Ω' are initialized to size 0. Each database in Ω' is allocated elements at only one time during execution of *Greedy-Dedup-Secure*. This occurs at Line 7, at which point a database is allocated at least k elements. Once data is allocated it is never deallocated. □

From Lemmas 15 and 16 it follows for Lemma 17 that every element in an outputted database must have an equivalent row vector in *Omega-Map*.

Lemma 17 (GDS Allocated Protectees Have Equivalent Trails). *Let P be the matrix representation of Omega-Map. If Ω' is output from Greedy-Dedup-Secure, then in $Omega - Map(h(\Omega'), h(\Omega)) \rightarrow P$, the rows for all elements in $\omega_i' \in \Omega'$ are equivalent with at least $|\omega_i'|$ 1's per row.*

PROOF. From Lemma 15 we know that data in Ω' is deduplicated. Thus, every element in database ω_i' must be linked to the same set of elements in $h(\Omega)$. Moreover, by Lemma 10, we know that there are at least as many protectors as the size of database $|\omega_i|$. As a result, all corresponding rows in P for elements in ω_i' must be equivalent and linked to at least $|\omega_i'|$ protectors. In terms of P, this means there are at least $|\omega_i'|$ 1's per row. □

Finally, we can prove *Greedy-Dedup-Secure* satisfies central k-unlinkability (Theorem 7).

Theorem 7 (GDS is Central k-Unlinkable). *Output from Greedy-Dedup-Secure satisfies central k-unlinkability.*

PROOF. Let M and $h(N)$ be the set of elements in Γ and Ω', respectively. Matrix P represents the edge set for a bipartite graph between M and $h(N)$ as observed by the *sTTP*. Now, let n be the number of non-null databases in Ω'. There exist $n+1$ connected components in P. The first n components correspond to the non-empty database in Ω'. Then, by Lemma 17, each of these components corresponds to a complete bipartite graph with at least k vertices from each vertex set (i.e., M and $h(N)$). In this respect, each of these components satisfies k-unlinkability.

The additional component corresponds to the set of elements that are completely suppressed by the *sTTP*. Each of these elements has a null trail. If the suppressed elements corresponding to null trails are disclosed by locations with non-null databases of allocated protectees, then these trails must increase the size of cliques.

As a result, P^t satisfies k-unlinkability for all trails in $N_{\Omega'}$. Recall, central k-unlinkability is defined with respect to $B = QP^t$. Since the suppression operations

performed by the *sTTP* can only make trails more ambiguous, and Q is a link matrix that uses the least ambiguous trails, B must satisfy k-unlinkability as well. ■

Contributor k-Unlinkability

Prior to suppressing whole datasets that violate external k-unlinkability via the *Explicit-Clean* procedure, the *Contributor-Clean* method suppresses data that violates contributor k-unlinkability. The conditional statement at line 3 of *Contributor-Clean* evaluates the degree to which location c_j can leverage private knowledge to re-identify data disclosed by c_i. First we prove in Lemma 18 that when location c_i discloses data that c_j submitted to the *sTTP*, the *sTTP* can verify c_j can not re-identify c_i's data.

Lemma 18 (*sTTP* Can Verify Clean Intersections). *The* sTTP *can verify $\omega_i{}'$ satisfies contributor k-unlinkability when $\forall c_j \in C$: $h(\omega_i{}') \subseteq h(\omega_i) \cap h(\omega_j)$.*

PROOF. By the definition of a re-identification, elements in the intersection $h(\omega_i) \cap h(\omega_j)$ provide neither c_i nor c_j with knowledge lacking in τ_i or τ_j. So, if $h(\omega_j{}') \subseteq h(\omega_i) \cap h(\omega_j)$, then the *sTTP* knows c_i can discover no more re-identifications than without access to $\omega_j{}'$. □

A corollary of Lemma 18 is that the third party can verify that the intersections of all location pairs satisfy contributor k-unlinkability. We state this finding for Corollary 6.

Corollary 6 (*sTTP* Can Verify All Clean Intersections). *Given output from* Greedy-Dedup-Secure, *the* sTTP *can verify $N_{\Omega'}$ is contributor k-unlinkable for all locations if $\forall c_i, c_j \in C$: $h(\omega_i{}') \subseteq h(\omega_i) \cap h(\omega_j)$.*

Lemma 18 accounts for data in the intersection of submitted databases, but data outside of the intersection is also available for disclosure. There must exist at least k identified elements in τ_i for whom c_j does not know the corresponding de-identified data in τ_j, otherwise we fail to satisfy contributor k-unlinkability. This is because c_j would know that each of the remaining elements could be re-identified using c_i's data. For Theorem 8, we prove the *sTTP* can determine when such a phenomenon is guaranteed to occur via the *Contributor-Clean* method.

Theorem 8 (*sTTP* Certifies Contributor k-Unlinkability). *Given Ω' is the output from* Greedy-Dedup-Secure, *the* sTTP *can verify $N_{\Omega'}$ satisfies contributor k-unlinkability with respect to c_j.*

PROOF. Following from Lemma 18, only elements in the set $h(\omega_i{}') \backslash h(\omega_j{}')$ are potentially re-identifiable by c_j. Since Ω' is deduplicated (Lemma 15), the *sTTP* can simulate location c_j's abilities by relating databases γ_i, γ_j, $h(\omega_i)$, and $h(\omega_j)$ as shown in Figures 8.8 and 8.9. The regions that influence the *sTTP*'s certification ability are as follow.

❶ This region specifies identified elements in γ_i that are not in τ_j, or $\gamma_i \backslash \gamma_j$. This region contributes $|\gamma_i \backslash \gamma_j|$ elements for protection of elements in $h(\omega_i{}')$.[3]

❷ This region represents identified elements that both locations have in common but c_j does not know the corresponding de-identified elements in τ_j. Identified elements in this region protect $h(\omega_i{}')$, but the *sTTP* can only bound the number of elements. The smallest size of this region, which we call $❷_{min}$, is

$$\max(0, |h(\omega_i) - ❸| - |\gamma_i \backslash \gamma_j|). \tag{8.2}$$

❸ This region represents de-identified elements that both locations have in common, or $h(\omega_i) \cap h(\omega_j)$. As stated in Theorem 7, no elements from γ_i in this region protect elements in $h(\omega_i{}')$.

❹ This region represents identified elements that 1) both locations have in common, 2) c_j knows the corresponding de-identified elements in τ_j, and 3) c_i does not know the corresponding elements in τ_i. The maximum size of this region, which we call $❹_{max}$, is

$$\min(|h(\omega_j)|, |\gamma_i \cap \gamma_j| - |❷_{min}|) - |❸|. \tag{8.3}$$

When regions ❷, ❸, and ❹ fail to cover the region $\gamma_i \cap \gamma_j$, then region ❺ exists and contributes to protection. Figure 8.9 provides an example of this region. Based on these regions, we find the minimum number of elements available to protect $h(\omega_i{}')$ is:

$$\begin{aligned}
&|\gamma_i| - |❸| - |❹_{min}| \\
&= |\gamma_i| - \min(|h(\omega_j)|, |\gamma_i \cap \gamma_j| - |❷_{min}|) \\
&= |\gamma_i| - \max(|\gamma_i \cap \gamma_j| - |h(\omega_j)|, |❷_{min}|) \\
&= |\gamma_i| - \max(|\gamma_i \cap \gamma_j| - |h(\omega_j)|, \max(0, |h(\omega_i) \backslash h(\omega_j)| - |\gamma_i \backslash \gamma_j|)) \\
&= |\gamma_i| - \max(0, |\gamma_i \cap \gamma_j| - |h(\omega_j)|, |h(\omega_i) \backslash h(\omega_j)| - |\gamma_i \backslash \gamma_j|). \\
&= |\gamma_i| - \max(|\gamma_i \backslash \gamma_j|, |\gamma_i| - |h(\omega_j)|, |h(\omega_i) \backslash h(\omega_j)|).
\end{aligned}$$

This is precisely the criteria in line 3 of *Contributor-Clean*. When the value is $\geq k$, then the *sTTP* can certify the set of maximum matchings will reveal that each element in $h(\omega_i{}')$ is linked to at least k identified elements. In combination with deduplication, the *Contributor-Clean* protocol guarantees contributor k-unlinkability is

[3] "$A \backslash B$" corresponds to set remainder, i.e., elements from A that are not in B.

satisfied for c_i's data with respect to c_j. Thus, when this holds for each location's data in C with respect to c_j, then contributor k-unlinkability is satisfied for c_j as an adversary. ■

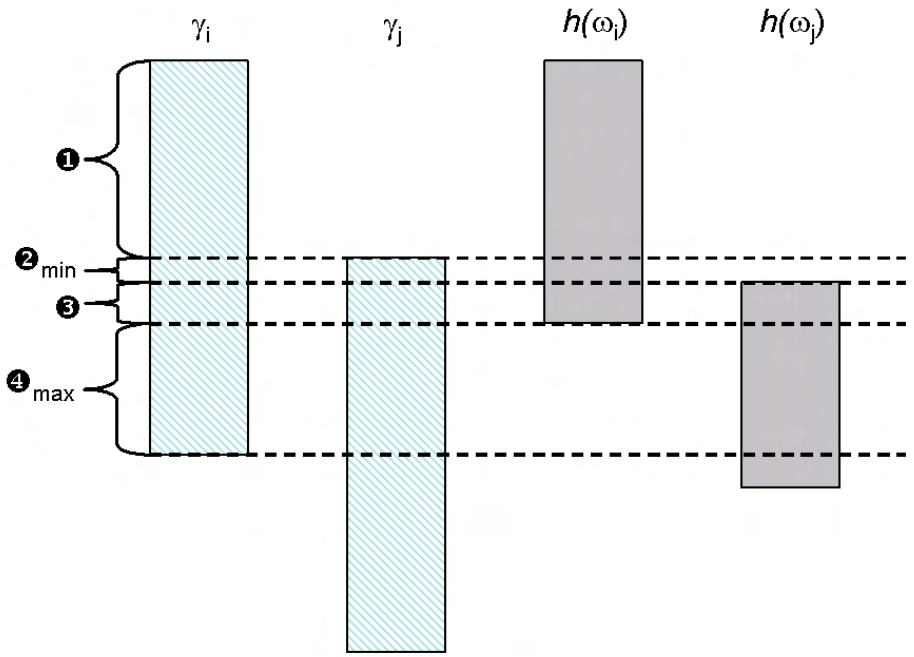


Figure 8.8: Databases for locations c_i and c_j as observed by the *sTTP* in a reserved environment.

When Theorem 8 holds true for all locations, all disclosed datasets output by *Greedy-Dedup-Secure* are certifiably contributor k-unlinkable.

8.3.2 Contributor k-Unlinkability Bounds

While the secure deduplication algorithms guarantee all forms of k-unlinkability are satisfied, the *Contributor-Clean* method always uses the minimum number of elements in γ_i that can protect elements disclosed in $\omega_i{}'$. In Appendix D, we show that the underestimation of the true number of elements has a bound dependent on regions ❷ and ❹.

Furthermore, we find that in an unreserved environment, which Figure 8.10 shows from a visual perspective, the only regions that exist are ❶, ❸, and ❻. As a result, *Contributor-Clean* does not underestimate the number of elements available for protection.

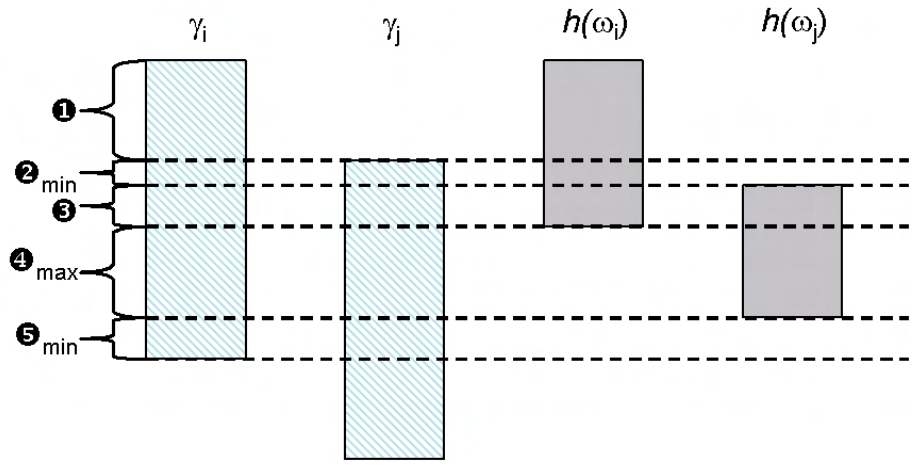


Figure 8.9: Worst case protection scenario for ω_i as observed by the *sTTP* in a reserved sharing environment.

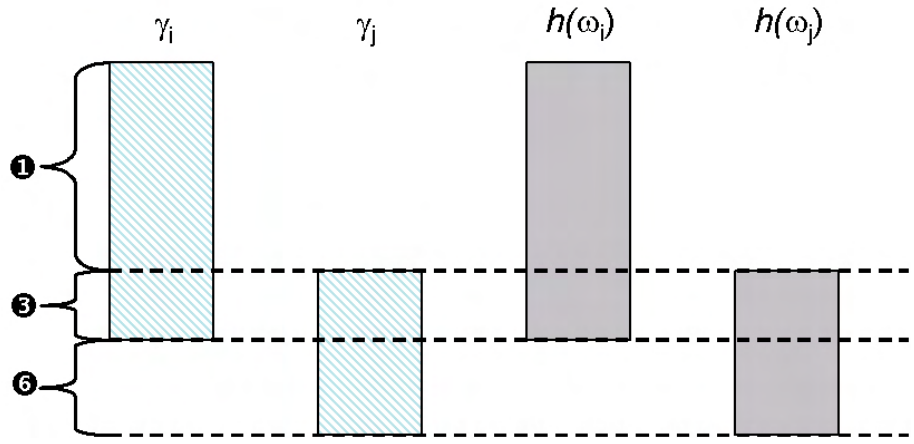


Figure 8.10: Databases for c_i and c_j in an unreserved environment as observed by the *sTTP*.

8.3.3 Force-Dedup-Secure

Second, we augment the *Force-Dedup* algorithm from the previous chapter. Pseudocode for the revised version which we call *Force-Dedup-Secure* is presented in Algorithm 14. The *Force-Dedup-Secure* uses the two phase process of data allocation used in *Force-Dedup* (Algorithm 10 in Chapter 7) in combination with the new contributor and central k-unlinkability constraints introduced above. Again, All solutions derived from *Force-*

Dedup-Secure are in the same solution space as covered by the k-unlinkability proof for *Greedy-Dedup-Secure*.

Algorithm 14 Force-Dedup-Secure($\Gamma, h(\Omega), k$)

Input: $\Gamma = \{\gamma_1, \ldots, \gamma_{|C|}\}$, the set of published databases, $h(\Omega) = \{h(\omega_1), \ldots, h(\omega_{|C|})\}$, the set of encrypted databases sent to the third party by the participating locations; k, an integer specifying the protection parameter.

Output: $h(\Omega') = \{h(\omega_1'), \ldots, h(\omega_{|C|}')\}$, the set of encrypted databases to return to c_1, ... , $c_{|C|}$, respectively.

1: $h(\omega_1') \leftarrow \{\}, \ldots, h(\omega_{|C|}') \leftarrow \{\}$ //*Initialize* $h(\Omega')$ *to null sets*
2: $\Gamma' \leftarrow \Gamma$
3: **let** $\Theta \leftarrow$ *Contributor-Clean*(Γ', $h(\Omega)$, k)
4: **let** $\{\Gamma', \Theta\} \leftarrow$ *Explicit-Clean*(Γ', Θ, k)
5: $AVAIL \leftarrow C$
//*FORCE PHASE: Allocate up to* k *elements per location*
6: **while** $\forall c \in AVAIL, |\theta_c| > 0$ **do**
7: $p \leftarrow \arg\min_{c \in AVAIL} |\theta_c| \geq k$
8: $AVAIL \leftarrow AVAIL \setminus \{p\}$
9: **let** $h(\omega_p') \leftarrow \{k$ elements in θ_p in the least number of databases in $\Theta\}$
10: **let** ${\gamma_p}^k \leftarrow \{k$ elements in $\gamma_p{}'$ in the least number of databases in $\Gamma'\}$
11: $\{\Gamma', \Theta\} \leftarrow$ *Reduce*(${\gamma_p}^k, h(\omega_p'), \Theta, \Gamma'$))
12: $\{\Gamma', \Theta\} \leftarrow$ *Explicit-Clean*(Θ, Γ', k) //*Correct for new constraints*
13: **end while**
14: $AVAIL \leftarrow C \setminus$ AVAIL //Use locations previously allocated data
//*BOOST PHASE: Allocate remaining elements*
15: **while** $\forall c \in AVAIL, \min(|\gamma_c'|, |\theta_c|) > 0$ **do**
16: $p \leftarrow \arg\min_{c \in AVAIL} |\theta_c| \geq 0$
17: **let** $h(\omega_p') \leftarrow h(\omega_p') \cup \{\min(|\theta_p|, \gamma_p'|)$ elements in $\gamma_p{}'$ in the least number of databases in $\Theta\}$
18: **let** ${\gamma_p}^k \leftarrow \{|\gamma_p'| - |h(\omega_p')|$ elements in $\gamma_p{}'$ in the least number of databases in $\Gamma'\}$
19: $\{\Gamma', \Theta\} \leftarrow$ *Reduce*(${\gamma_p}^k, h(\omega_p'), \Theta, \Gamma'$))
20: **end while**
21: **return** $h(\Omega')$

8.4 Conclusions

In this chapter, we introduced a multiparty computation protocol, *STRANON* for centralized analysis of trails and execution of deduplication algorithms to satisfy the general k-unlinkability definition (i.e., suppression in both trail matrices). We extended basic k-unlinkability definitions, where it was assumed all locations could openly communicate the contents of their databases, to account for more strict privacy requirements for k-unlinkability in comparison to the previous chapter. Specifically, we provided models of *contributor* (i.e., a member of the participating locations) and *central k-unlinkability* (i.e., a semi-trusted third party that executes the unlinking algorithms over encrypted data). We then introduced several extensions to the unlinking algorithms in the previous chapter to transform trails to satisfy the more strict k-unlinkability requirements. Moreover, the protocol is applicable within current data privacy policies, such as recent federal health data privacy regulations. In the next chapter, we investigate the efficacy of the proposed algorithms on various real world datasets.

Chapter 9

Trail Unlinkability in the Real World

In the previous chapters, deduplication algorithms were introduced that guarantee k-unlinkability over disclosed trail matrices. In this chapter we investigate the utility preservation capabilities of the deduplication algorithms in real world scenarios.

Reporting the number of samples that can be disclosed after deduplication offers little indication of pre-existing re-identifiability. Thus, prior to presenting our results, and in order to fully evaluate the deduplication algorithms, we introduce a method that measures the re-identifiability of the data beyond unique re-identification. We call this type of privacy compromise a k-re-identification.

9.1 REIDIT-I-K: Beyond Unique Re-identification

During the presentation of trail re-identification in Part I of this dissertation, we introduced several re-identification algorithms. The aforementioned algorithms are correct in the re-identifications they discover, but they only report unique re-identifications. For example, the REIDIT-I algorithm merely discovers trails that satisfy 1-unlinkability. We need to specify levels of privacy at arbitrary levels of k-unlinkability; thus REIDIT-I is not ideal for measuring the trail re-identifiability of data if non-unique linkage is considered a privacy compromise.

In order to estimate the number of k-re-identifications that exist for an arbitrary level k, we present the REIDIT-I-K algorithm, pseudocode for which is provided in Algorithm 15, which is an extension of REIDIT-I. Given two sets of databases Γ, Ω such that ω_c is reserved to γ_c for each location c, the algorithm determines when trails of one trail matrix are k-re-identified (i.e., linked to less than k trails) of another trail matrix. To elaborate, let M and N be the set of distinct elements in Γ and Ω, respectively. REIDIT-I-K performs unique re-identification, or 1-re-identified, in the same manner as REIDIT-I. Next, if a trail

n_Ω is the subtrail of less than k trails in M_Γ, then n and the elements of its supertrails are added to the set of re-identifications. Unlike REIDIT-I, when n is mapped to more than one element, then the elements are not removed from M.

Execution of REIDIT-I-K guarantees for every trail n_Ω in the set of returned k-re-identifications: 1) there exists no more than $k-1$ elements in M to which n is paired and 2) one of the pairs is the correct linkage (by the subtrail/supertrail criteria).[1]

Algorithm 15 REIDIT-I-K(M_Γ, N_Ω, M, N, k)

Input: M_Γ and N_Ω, trail matrices for sets of partitioned databases; M and N, sets of corresponding tuples for M_Γ and N_Ω, respectively; k, integer specifying the protection parameter.

Output: R, the set of k-re-identifications in pair form $\langle m \in M, n \in N \rangle$.

$R \leftarrow \emptyset$
//Same as REIDIT-I, iteratively discover and remove all 1-unlinkable
repeat
  $NUMFOUND \leftarrow |R|$
  for $n \leftarrow 1$ to $|N|$ **do**
    if there exists one and only one $m \in M$, such that $m_\Gamma \preceq n_\Omega$ **then**
      $R \leftarrow R \cup \{\langle m, n \rangle\}$ *//remove m and n from further consideration*
      $M \leftarrow M - \{m\}$; $N \leftarrow N - \{n\}$
    end if
  end for
until $NUMFOUND \equiv |R|$
//Extension to REIDIT-I, find x-unlinkable, $x < k$
for $n \leftarrow 1$ to $|N|$ **do**
  $Q \leftarrow \{m : n_\Omega \preceq m_\Gamma\}$
  if $|Q| < k$ **then**
    $R \leftarrow R \cup \{\langle m, n \rangle : m \in Q\}$
  end if
end for
return R

Like REIDIT-I, the REIDIT-I-K algorithm is not optimal. It does not construct the set of all maximum matchings and thus is not guaranteed to discover all k-re-identifications. Nonetheless, every k-re-identification is correct and therefore output from REIDIT-I-K represents a minimum of the number of k-re-identifications that exist in the system.

[1] As written, REIDIT-I-K neglects the situation when $|M| = |N|$. We can extend REIDIT-I-K to account for this scenario in a similar fashion to REIDIT-I-Flip. For presentation purposes we neglect this case.

9.2 Experiments

We evaluated both the non-secure algorithms, which satisfy external k-unlink-ability only, and secure deduplication algorithms, which satisfy both central and contributor k-unlink-ability. First, we report results for the non-secured versions with the hospital discharge databases described in Chapter 5. More specifically, we use the genderless datasets. Next, we investigated the algorithms with respect to a simulated population. Then, we performed a series of experiments to compare the non-secure to the secure deduplication algorithms to determine how much data utility is preserved in the face of more strict privacy constraints.

9.2.1 Results with Genetic Populations

Recall, the populations distilled from the hospital discharge databases are quite susceptible to unique trail re-identification. A summary of the number of elements, hospitals, and a snapshot of k-re-identification susceptibility via REIDIT-I-K is shown in Table 9.1. In these experiments, we investigated disclosure with respect to an unreserved environment. A more detailed analysis is shown for the CF dataset for a range of k from 2 to 50 is shown in Figure 9.1. As anticipated, our findings confirm that the number of samples that are k-re-identifiable increases as a function of k. We present similar plots for the other genetic datasets in Figure 9.12.[2]

Disease	**# of Samples**	**# of Hospitals**	**Percent Re-identified By REIDIT-I-K**	
			$k = 2$	$k = 5$
CF	1149	174	32.55%	51.96%
FA	129	105	82.94%	92.24%
HD	419	172	65.16%	89.73%
HT	429	159	65.97%	84.15%
PK	77	57	81.82%	90.91%
RD	4	8	100.00%	100.00%
SC	7730	207	35.95%	37.72%
TS	220	119	70.00%	79.55%

Table 9.1: The trail re-identifiability of the genetic disease populations via REIDIT-I-K for $k = 2$ and $k = 5$.

[2]We do not present a plot for the RD dataset; this dataset is always 100% re-identifiable.

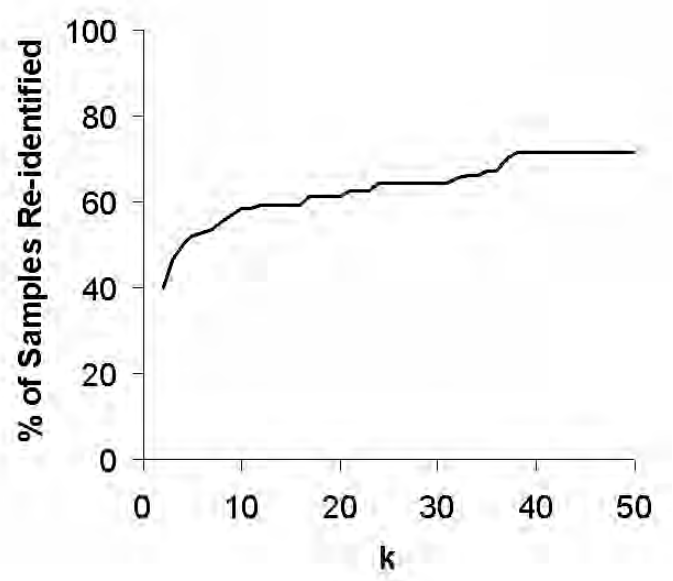


Figure 9.1: Percent of 1149 samples in the CF databases re-identified by REIDIT-I-K if every hospital discloses all of the samples in its private database.

A snapshot of the results for disclosure via the deduplication algorithms is shown in Table 9.2. Recall, the methods use greedy heuristics to arrive at their solutions. Since there can exist ties between dataset sizes as well as the probability estimates regarding which elements to choose from a particular database, we report results from multiple executions of the algorithms. Table 9.2 presents the mean, and one standard deviation, for 25 runs of the deduplication algorithms. With the exception of the HT dataset, note the standard deviation for the number of samples disclosed is within around 1% of the total number of samples.

Continuing with the CF dataset, a more detailed analysis is shown in Figure 9.2(a). Despite the use of a greedy heuristic, it is evident that the *Greedy-Dedup* permits disclosure of significant quantities of data in the face of trail re-identification. For instance, at $k = 5$, 84% of the elements in the HT dataset are re-identifiable prior to deduplication, but after execution of *Greedy-Dedup*, we are able to disclose 78% of the samples with zero re-identifications. Similar findings are observed for the other databases and are depicted in Figure 9.13.[3] Interestingly, *Greedy-Dedup* and *Force-Dedup* exhibit similar results and trends for most datasets.

A more detailed plot is shown for the CF dataset is shown in Figure 9.2(a). From Figure

[3]We do not present a plot for the RD dataset; this dataset always yields 0% disclosure.

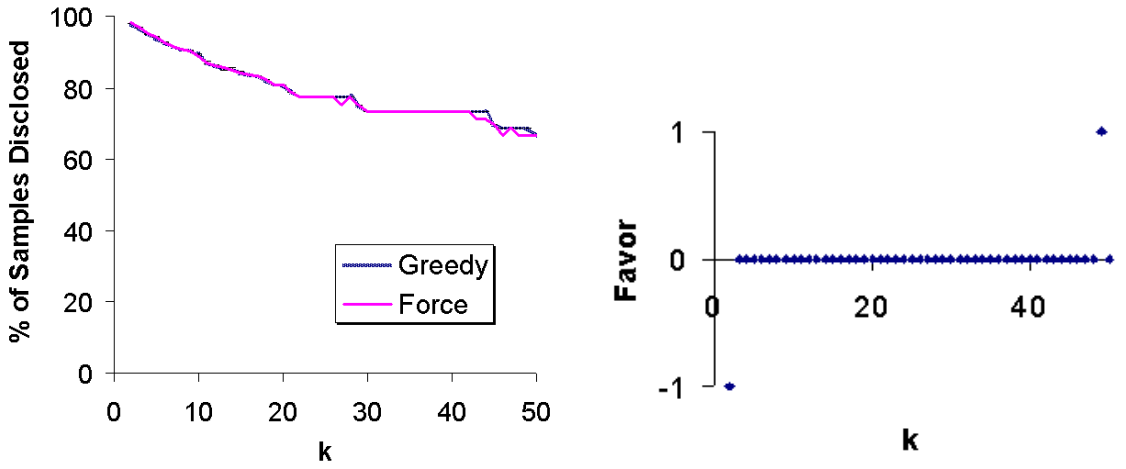
Disease	Total # Samples	# of Samples Disclosed (Standard Deviation)			
		k=2		k=5	
		Greedy-Dedup	*Force-Dedup*	*Greedy-Dedup*	*Force-Dedup*
CF	1149	1140 (0.0)	1141 (0.1)	1093 (1.1)	1094 (1.9)
FA	129	99 (1.2)	72 (2.5)	49 (0.0)	49 (0.4)
HD	419	398 (0.5)	397 (0.8)	304 (3.8)	306 (5.0)
HT	429	408 (0.0)	408 (0.1)	340 (2.7)	337 (4.2)
PK	77	56 (1.2)	57 (1.1)	5 (3.0)	26 (3.0)
RD	4	0 (0.0)	0 (0.0)	0 (0.0)	0 (0.0)
SC	7730	7723 (0.2)	7723 (0.0)	7700 (0.5)	7696 (0.1)
TS	220	203 (0.9)	204 (0.7)	174 (2.3)	173 (2.3)

Table 9.2: The mean number, from 25 runs, of DNA samples the deduplication algorithms disclose to satisfy 2-unlinkability.

9.2(a) we note that deduplication permits substantial disclosures of data that is guaranteed to be protected from re-identification at level k. Similarly, in Figure 9.2(a) we observe that the number of samples that can be disclosed by the unlinking algorithms decreases as a function of k. The standard deviation was within approximately 1% for each data point k, so error bars are not shown.

To compare the *Greedy-Dedup* and *Force-Dedup* algorithms beyond visual inspection, we checked for statistical significance in the form of t-tests to determine if either of the deduplication algorithms outperformed the other in the task of allocating samples for disclosure. We performed a t-test for each data point k, where the mean and variances for each algorithm were calculated from the 25 experimental runs mentioned earlier. For the CF dataset we present the results in Figure 9.2(b), such that a score of 1 implies *Greedy-Dedup* outperformed *Force-Dedup*, a score or -1 implies vice versa, and a score of 0 implies there is no statistically significant different in the mean performance of the algorithms at the 99% confidence level.

For almost all data points neither algorithm outperforms the other. There are two exceptions, specifically at k=2 and k=49. At k = 2, we observe the *Force-Dedup* algorithm outperforms, with a mean percent of samples disclosed at approximately 1141 in comparison to 1140 for *Greedy-Dedup*. At k = 49, we observe the converse, such that *Greedy-Dedup* outperforms with a mean percent of samples disclosed at approximately 788 in comparison to 759 for *Force-Dedup*. However, such significance findings are few, and in Figure 9.14 we present the results for the other datasets, where we find there is no



(a) Percent Disclosed

(b) T-test Scores

Figure 9.2: 9.2(a) Average percent released of 1149 samples in the CF databases after executing the unlinking algorithms. All disclosed samples are k-unlinkable. 9.2(b) T-tests for significant difference in the two algorithms; a score of 1 implies *Greedy-Dedup* outperforms; a score of -1 implies *Force-Dedup* outperforms.

significant difference for the algorithms in the HD, PK, and TS algorithms. For the FA and HT datasets we observe similar findings to that of the CF dataset, such that there exist several occurrences in which *Force-Dedup* outperforms at lesser values of k (i.e., less than k = 10). Yet, we can not claim this is always so because in the SC dataset we find the converse. In the SC dataset, we observe a much stronger signal which suggests at lower values of k *Greedy-Dedup* outperforms, while at higher k, *Force-Dedup* outperforms. We will return to the implications of this finding when we consider the simulated population in the following section.

Returning to our analysis of the general trend of deduplication, it is also encouraging to note that the unlinking algorithms sustain a smaller rate of loss in disclosure ability than the rate of increased re-identification. This finding is depicted in Figure 9.3.

Continuing with the CF dataset, in Figure 9.4, we depict results for the task of maximizing the number of hospitals that are allocated non-null databases via the deduplication process. This plot depicts the average number of hospitals allocated data; the standard deviation was negligible ($\leq 1\%$) for almost all data points, so only the average is shown.

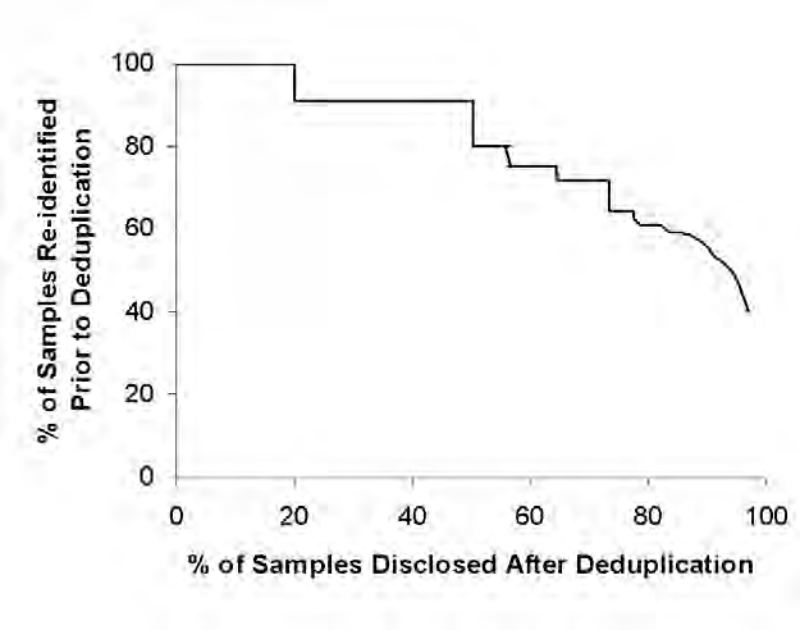


Figure 9.3: Rate of re-identification versus rate of protected disclosure for Force-Dedup. Range from k = 2 to 159.

Disease	Total # Locations	# of Locations Disclosing (Standard Deviation)			
		k=2		k=5	
		Greedy-Dedup	*Force-Dedup*	*Greedy-Dedup*	*Force-Dedup*
CF	174	136 (0.0)	137 (0.4)	82 (0.4)	83 (1.1)
FA	105	39 (0.6)	39 (0.6)	8 (0.0)	8 (0.20)
HD	172	120 (0.0)	119 (0.8)	52 (1.1)	52 (1.6)
HT	159	115 (0.5)	115 (0.6)	55 (1.0)	54 (1.6)
PK	57	23 (0.6)	23 (0.5)	4 (0.5)	5 (0.5)
RD	8	0 (0.0)	0 (0.0)	0 (0.0)	0 (0.0)
SC	207	184 (0.2)	184 (0.0)	155 (0.7)	154 (0.0)
TS	119	60 (0.9)	61 (0.8)	25 (0.9)	24 (0.9)

Table 9.3: The number of hospitals that disclose non-null DNA databases via Greedy-Dedup to satisfy 2-unlinkability and 5-unlinkability.

We observed an exponential decay in the average number of hospitals allocated non-null databases. This trend generalizes to the other datasets and is correlated with the theoretical maximum, calculated as $\min(|S|/k, |C|)$).

There is one exception to this trend, which corresponds to the SC dataset. We still find a generalized exponential decay trend from around k = 2 to 10. However, after this point the rate of change shifts to an approximately linear decay rate. This is indicative of the larger size of the SC dataset in comparison to the other datasets. More specifically, there is a larger number of samples per location, and as a result, as k increases the locations that cover significant portions of the sample population are suppressed.

Nonetheless, for all datasets we find the number of disclosing locations is far below the theoretical maximum. We also observe an interesting phenomenon quite contrary to initial expectations. Based on its two-phase design, we expected *Force-Dedup* to dominate over *Greedy-Dedup*. Instead we observe both algorithms are approximately equal in their capability. Again, there is no statistical significance in a point-by-point analysis. We suspected this phenomenon may be due in part to the distribution of people to locations. Specifically, the distribution of subjects per location follows a generalized power law.

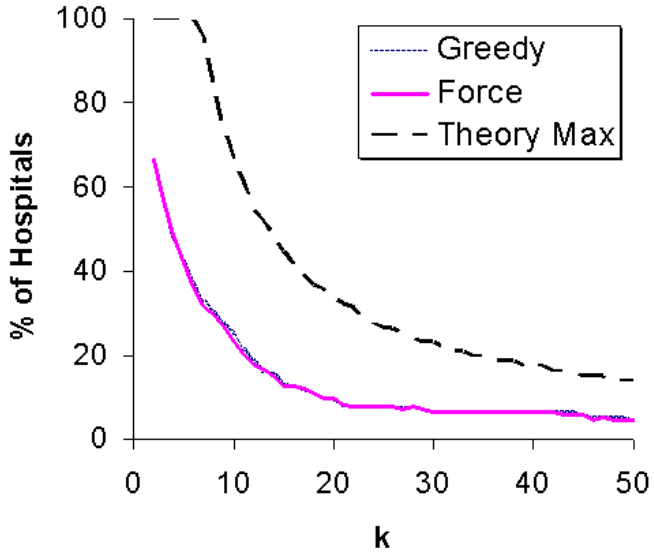
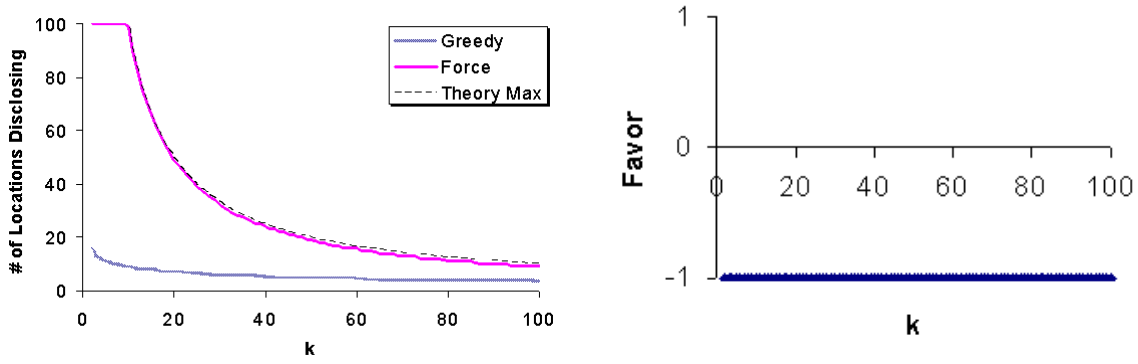


Figure 9.4: Percent of 174 locations in the CF dataset permitted to disclose data after deduplication.

9.2.2 Results with Simulated Populations

To explore our hypothesis we evaluated the algorithms in a simulated environment consisting of 1000 entities and 100 locations. Populations were generated according to a uniform distribution with $Pr(\text{patient } s \text{ visits location } c) = 0.5$ for all patients and locations. Twenty-five simulations were run for each level k from 2 to 100. In Figure 9.5(a), we plot the number of hospitals releasing non-null databases following execution of the deduplication algorithms as a function of the k protection parameter. Again, standard deviation was within approximately one location for all k. As expected, the *Force-Dedup* algorithm unequivocally dominates over *Greedy-Dedup* with respect to the average number of locations allocated samples, as depicted by its significance in the t-tests in Figure 9.5(b). Moreover, *Force-Dedup* is now much closer to the theoretical maximum in comparison to the CF dataset. Thus, in environments with lesser skewed distributions the heuristic employed in *Force-Dedup* appears to facilitate an increase in the number of locations disclosing non-null databases.



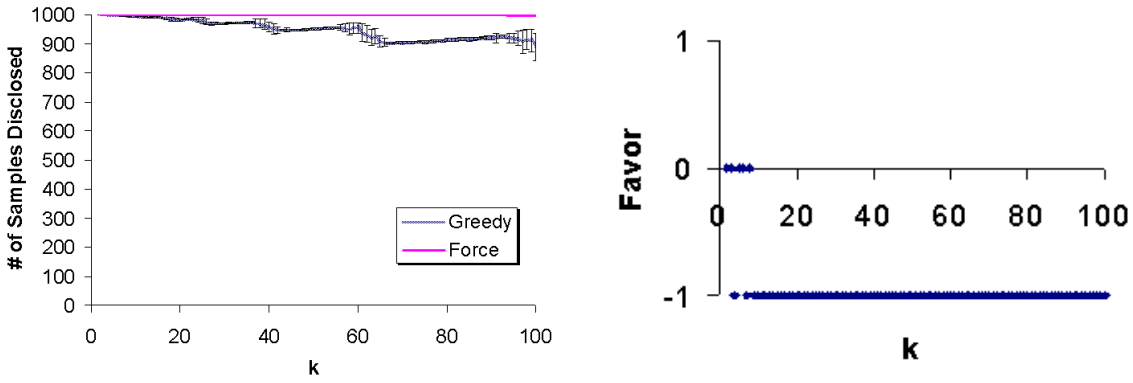
(a) Number Disclosed

(b) T-test Scores

Figure 9.5: 9.5(a) Number of locations allocated non-null databases in simulated populations. 9.5(b) T-tests for significant difference in the two algorithms; a score of 1 implies *Greedy-Dedup* outperforms; a score of -1 implies *Force-Dedup* outperforms.

Upon further investigation, it appears *Force-Dedup* may utilize a superior heuristic, in comparison to *Greedy-Dedup*, for deduplication as well. At every level k, 100% of the samples were k-re-identified. Moreover, in Figure 9.6(a) we depict the number of de-identified elements disclosed for varying levels of k. Note, the simulated populations are 100% re-identifiable for every level k via REIDIT-I-K . With respect to deduplication for

disclosure, we observe *Force-Dedup* completely dominates *Greedy-Dedup*. The expectation was the greedy heuristic *Greedy-Dedup* algorithm to be superior for this task, however, it seems that the initial allocation loop in *Force-Dedup*, oriented to boost the number of hospitals provides the added benefit of limiting the greedy behavior of *Greedy-Dedup*. This is statistically confirmed by significance in the t-tests shown in Figure 9.6(b). In future research, we intend to study the influence of data distribution on the two algorithms in more depth.



(a) Number Disclosed

(b) T-test Scores

Figure 9.6: 9.6(a) Number of samples disclosed from simulated populations. 9.6(b) T-tests for significant difference in the two algorithms; a score of 1 implies *Greedy-Dedup* outperforms; a score of -1 implies *Force-Dedup* outperforms.

9.2.3 Secure Results

The effect of the *Contributor-Clean* algorithms was evaluated on the genetic datasets of the previous chapter. As in the previous set of experiments, we investigated disclosure with respect to an unreserved environment. As expected, the disclosure capability of secure deduplication results in decreased amounts of disclosure, or loss in the number of samples that are disclosed and locations that are allocated non-null databases. First, in Table 9.4 we present a snapshot of the results with *Force-Dedup-Secure* for k = 5. As with the non-secure versions, we present the average number of samples disclosed from twenty-five runs of the algorithm. Similar standard deviations were observed, and thus are not presented in the table. From these experiments, it is apparent that there does exist loss for

the genetic datasets. The number of samples lost from ranges from approximately 5 to 20 samples. In Table 9.4, to characterize loss, we measured percent decrease and percent total loss. Percent decrease corresponds to:

$$100 * \frac{\text{(\# of samples in non-secure disclosure) - (\# of samples in secure disclosure)}}{\text{\# of samples in non-secure disclosure}}.$$

Similarly, we defined percent loss as:

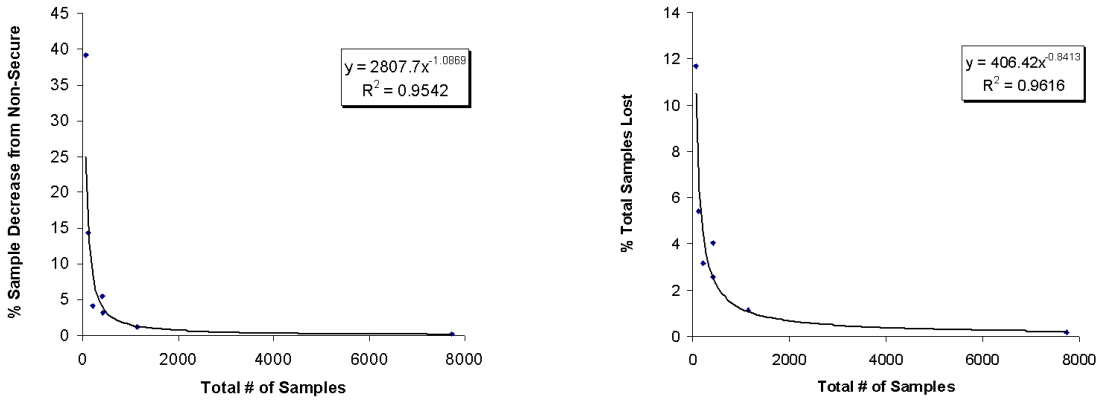
$$100 * \frac{\text{(\# of samples in non-secure disclosure) - (\# of samples in secure disclosure)}}{\text{total \# of samples}}.$$

For example, percent decrease for CF is calculated as 100*(1095 - 1082)/1095 = 1.18%. In contrast, percent loss for CF is calculated as 100*(1095 - 1082)/1149 = 1.13%. The raw number of additional samples that are suppressed by the secure algorithm ranges from around 5 to 20. When normalized by the number of samples that are initially disclosed by the non-secure samples, the percent decrease from the non-secure algorithms ranges from approximately 0.2% (the SF dataset) to almost 40% (PK dataset) loss. Yet, the percent loss has a smaller range of approximately 0.2% (the SF dataset) to around 12% (PK dataset). For the genetic datasets, both percent decrease and percent loss are dependent upon, and are inversely correlated with, the size of dataset. We summarize this finding in Figures 9.7(a) and 9.7(b) and show the inverse correlation appears to follow a power law.

Disease	**Total # Samples**	**# Disclosed**	**# Lost**	**% Decrease**	**% Loss**
CF	1149	1082	13	1.18%	1.13%
FA	129	42	7	14.29%	5.43%
HD	419	292	17	5.50%	4.05%
HT	429	325	11	3.27%	2.56%
PK	77	14	9	39.13%	11.69%
RD	4	0	0	0.00%	0.00%
SC	7730	7685	15	0.20%	0.19%
TS	220	162	7	4.32%	3.18%

Table 9.4: Samples lost due to execution of *Force-Dedup-Secure* in comparison to *Force-Dedup* for k = 5.

A more detailed analysis is shown in Figures 9.8(a), with a closeup in Figure 9.8(b) for the CF dataset. In these plots, we show the percent loss as a function of k. The final jump in



(a) Percent Decrease

(b) Percent Lost

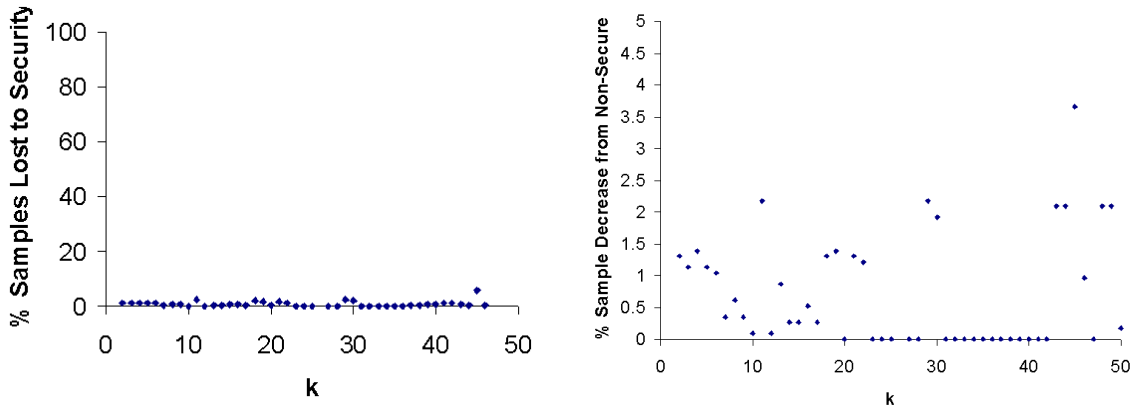
Figure 9.7: Trend summary in percent sample decrease and loss.

the percent of samples lost is due to the more stringent unlinkability constraints is imposed by the contributor k-unlinkability. Specifically, to satisfy contributor k-unlinkability, there must be at least k elements outside of the intersection of two locations' dataset in order to disclose such elements. Thus, this jump corresponds to the point at which the intersection of a large k no longer satisfies this requirement.

Nonetheless, the non-intersecting requirements as captured by the *Contributor-Clean* method shifts trend back by one k. We highlight this finding in Figure 9.9(a). In Figure 9.9(b) we plotted the shifted secure deduplication results (y-axis) against the original deduplication results. We find that there is a strong linear correlation between these two trends, with an r^2 correlation score of approximately 0.995. Note, the shift accounts for lower values of k, though as k increases, the explanatory power decreases such that the correlation is less defined.

Next, we investigated the effect of the increased constraints on our ability to distribute non-null databases to locations. In Tables 9.5 and 9.6, we show snapshots of the results for k = 2 and k = 5, respectively. For both tables, the percent decrease and percent loss measures where calculated as for the samples in Table 9.4.

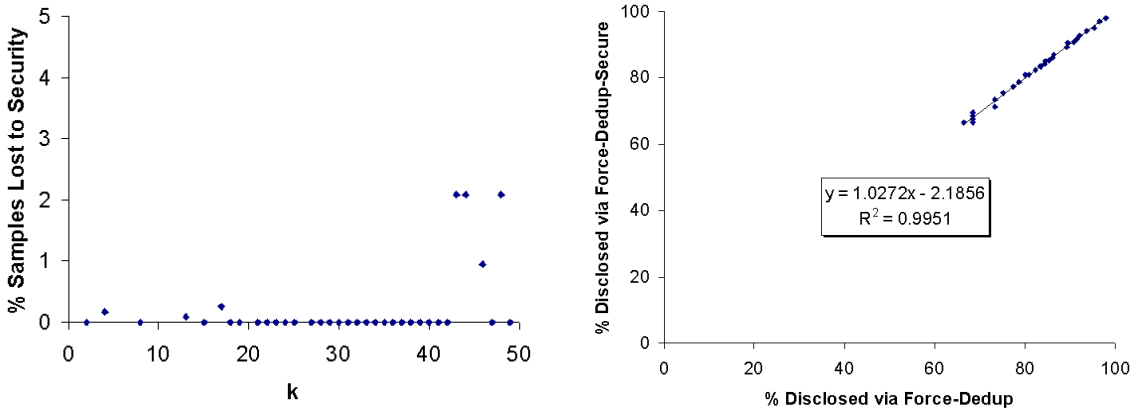
The two tables highlight the exponential decay effect an increasing k has on the rate of change in the number of locations. Specifically, as k grows, the rate decreases. Also, it appears that after a certain k (different for each dataset), the secure and non-secure algorithms appear to converge. We depict this effect with the CF dataset in Figure 9.10. The effect with respect to the other genetic datasets are provided in Figure 9.18.



(a) Loss compared to original algorithm.

(b) Zoomed in.

Figure 9.8: Difference in percent of CF samples disclosed by *Force-Dedup-Secure* in comparison to *Force-Dedup*.



(a) Shifted loss plot.

(b) Correlation.

Figure 9.9: Difference in percent of CF samples disclosed by *Force-Dedup-Secure* in comparison to *Force-Dedup* after shifting k by one.

In addition, this effect is highlighted by Figure 9.11(a), which depicts the percent of locations loss due to increased unlinkability constraints. Note that there is an exponential

Disease	Total # Locations	# Disclosing	# Lost	% Decrease	% Loss
CF	174	117	20	14.60%	11.49%
FA	105	23	16	41.03%	15.24%
HD	172	97	23	19.17%	13.37%
HT	159	93	22	19.13%	13.84%
PK	57	16	8	33.33%	14.04%
RD	8	0	0	0.00%	0.00%
SC	207	177	7	3.80%	3.38%
TS	119	53	9	14.52%	7.56%

Table 9.5: Locations lost due to execution of *Force-Dedup-Secure* in comparison to *Force-Dedup* for k = 2.

Disease	Total # Locations	# Disclosing	# Lost	% Decrease	% Loss
CF	174	73	9	10.97%	5.17%
FA	105	6	2	25.00%	1.90%
HD	172	47	4	7.84%	2.33%
HT	159	50	4	7.41%	2.52%
PK	57	2	2	50.00%	3.51%
RD	8	0	0	0.00%	0.00%
SC	207	149	6	3.87%	2.90%
TS	119	20	3	13.04%	2.52%

Table 9.6: Locations lost due to execution of *Force-Dedup-Secure* in comparison to *Force-Dedup* for k = 5.

drop in the difference between the difference in the percent of locations allocated non-null databases when comparing the non-secure and secure algorithms.

Furthermore, we observe a similar finding to our observation of shift with sample loss. In Figure 9.11(b) we show the effect of shifting k by one on the difference in the percent of locations not allocated any data. Akin to sample loss, we find that the difference in location loss is dampened by the shift. The dampening is not quite as strong as for sample loss. Furthermore, unlike our observations with sample loss we note that shifting allows the secure version to allocate to more locations than the non-secure. In this respect, it is clear that location loss due to increased unlinkability constraints is less predictable than sample loss. In future research, we intend to study the factors that influence this effect in more depth.

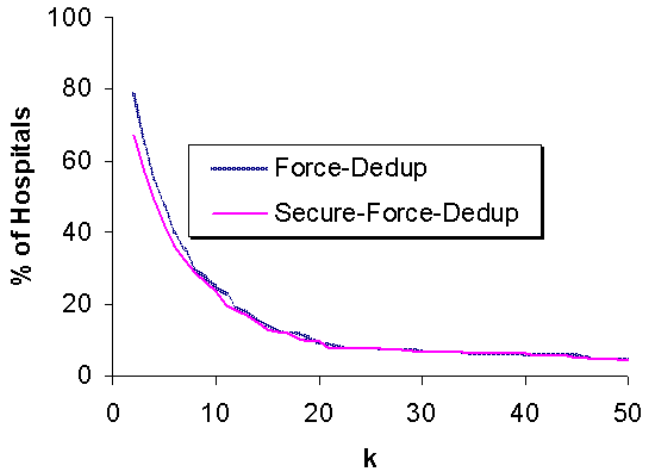
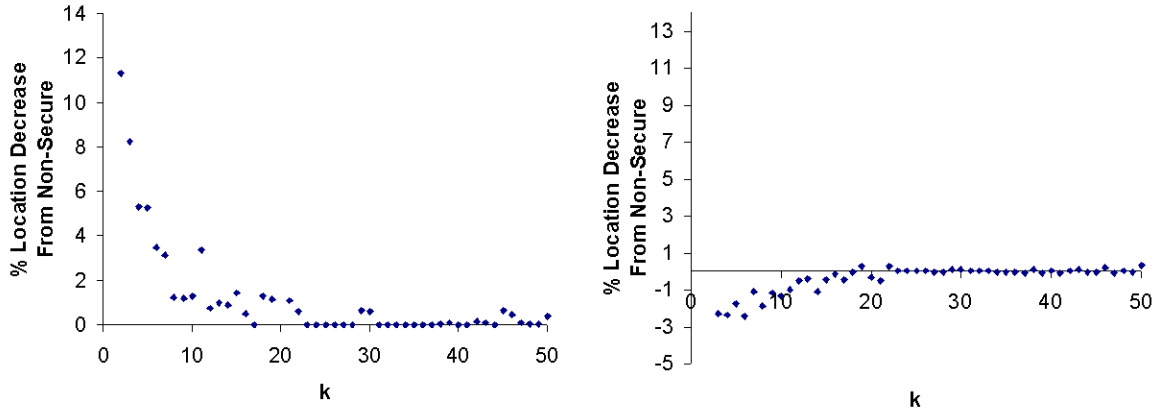


Figure 9.10: Comparison of locations allocated via *Force-Dedup-Secure* and *Force-Dedup*.



(a) Loss compared to original algorithm.

(b) Shifted by one.

Figure 9.11: Difference in percent of CF locations allocated non-null databases by *Force-Dedup-Secure* in comparison to *Force-Dedup*.

9.3 Discussion

From a computational perspective, the STRANON protocol permits centralized analysis and anonymization of distributed data trails. However, for the protocol to be deployed in

practice it must be framed in the context of policy. According to the U.S. Department of Health and Human Services (DHHS), who provides oversight for the HIPAA Privacy Rule, disclosed data is not sufficiently de-identified if it contains a keyed hash of an identifiable feature [32]. Thus, according to the DHHS, the *sTTP* is not legally permitted unfettered access or use of the encrypted databases. This policy constraint can be addressed if the *sTTP* functions as a business associate. with the participating hospitals [33]. Basically, by functioning as a business associate the *sTTP* is granted the rights to analyze a potentially re-identifiable health database provided it contractually agrees not to partake in any explicit re-identification attempts. Therefore, in combination with business associate status, the protocol allows the *sTTP* to 1) legally manage encrypted health data identifiers and 2) provably unlink trails without any ability to perform re-identification either during or after data disclosure.

Nonetheless, despite the certification of STRANON for HIPAA-regulated environments, there are certain limitations of the proposed protocol. Of specific concern is that though STRANON is based on a framework that is resistent to participants that act maliciously against the protocol (See Appendix C); STRANON does not guarantee that datasets submitted by the participating locations are truthful. From a security perspective, STRANON does not explicitly model the honesty of a location's behavior outside of the protocol. This is a concern and a direction for future research as well. Specifically, we are interested in extending STRANON to incorporate knowledge that permits the *sTTP* to validate the contents of submitted datasets. For instance, a tell tale indicator of dishonesty in data is if the *sTTP* detects trails that have no supertrails. Based on the reserved (and unreserved) environmental assumptions, such occurrences are impossible to achieve and therefore, some data holder must be lying.

In addition, the STRANON protocol can be extended to account for hospitals that collude with respect to their plaintext data. Basically, the *sTTP* can construct a table that incorporates trails as observed by a set of collaborators.

9.4 Limitations and Extensions

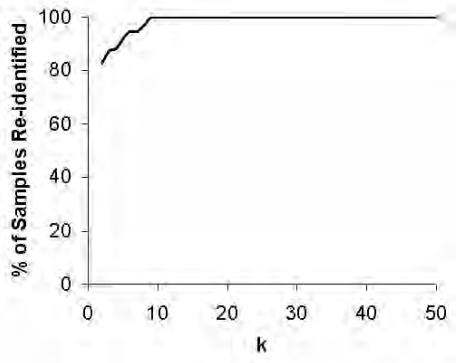
For this study, we adopted metrics based on deduplication. Yet, deduplication hinders certain types of analysis that can be performed. Specifically, we lose spatial and associative patterns with respect to individuals. In future research, we intend to study metrics that address different needs of the data. For instance, the traditional problem of minimize the total number of suppressions may useful for data mining purposes. In contrast, it may be useful to maximize the variance in the trail patterns. Each of these measures of utility addresses a different need and may require a different algorithmic approach.

The uncertainty that exists in the choice of utility metrics requires more targeted investigations into which types of metrics are appropriate for which types of usage. This may require expert input, which may be elicited via interviews with the experts (i.e., researchers) that have a vested interest in the use of such data. Alternatively, it may be possible to construct metrics that are based on information theoretic approaches for measuring the intrinsic value of the data. Nonetheless, this is a difficult and currently unresolved problem. Objective measures without any intuition into what the data is useful for can lead to inappropriate solutions.

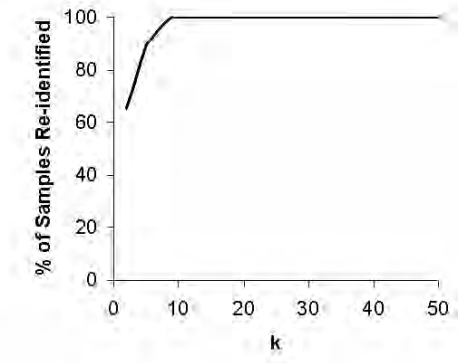
9.5 Conclusions

In this chapter we provided experimental validation on real world and simulated data, that for various goals of utility, significant quantities of data can be disclosed with zero risk of re-identification. We investigated the utility preservation capabilities of both non-secure and secure versions of the deduplication algorithms. Our findings with real world datasets suggest that the two-phase data allocation used by *Force-Dedup* and *Force-Dedup-Secure* tends to provide more preservation in comparision to the single-pass greedy algorithms. We confirmed this finding within a similulated population. Yet, simulated populations are not completely representatitive of the intricate correlations that manifest in the real world. In future research, we intend on evaluating the unlinking algorithms on databases derived from additional real world populations.

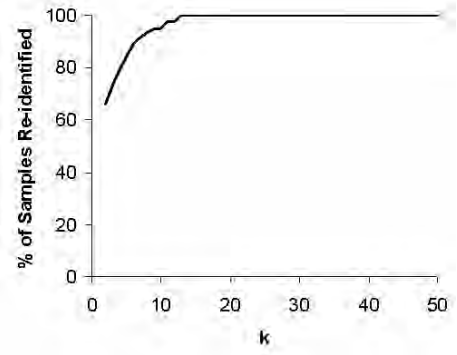
Referenced Figures



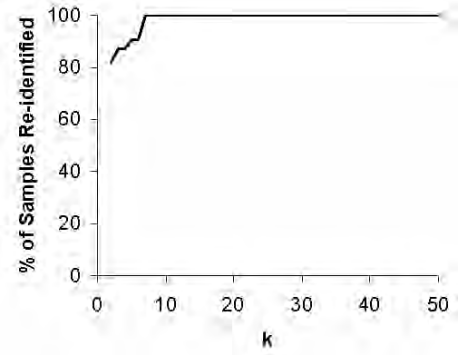
(a) FA



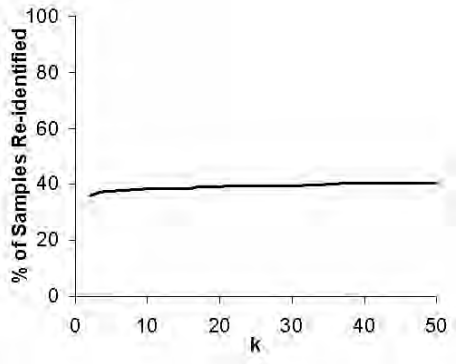
(b) HD



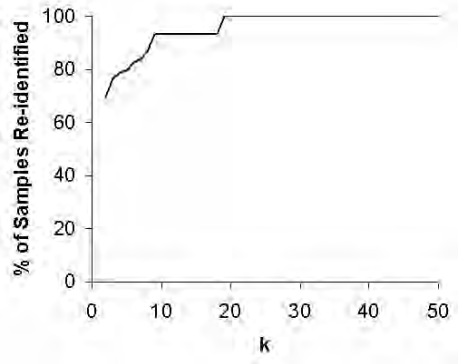
(c) HT



(d) PK

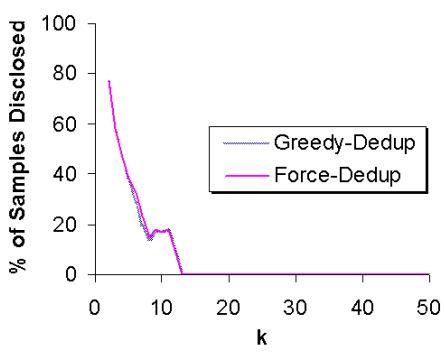


(e) SC

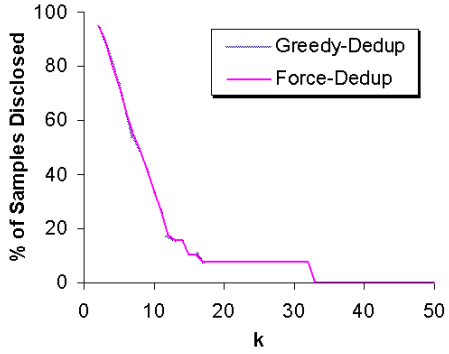


(f) TS

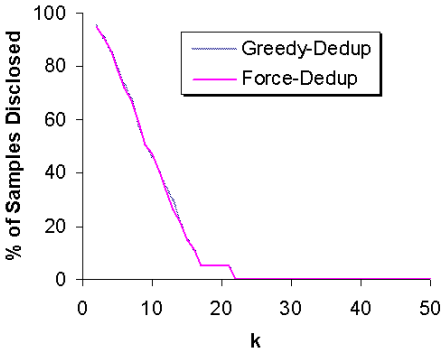
Figure 9.12: Re-identifications via REIDIT-I-K for genetic population datasets.



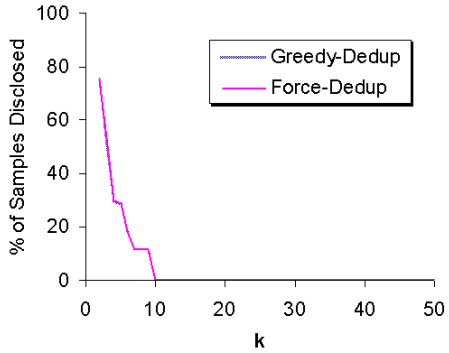
(a) FA



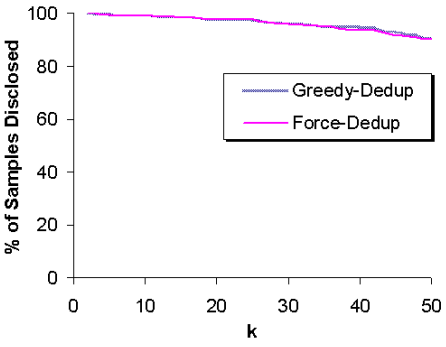
(b) HD



(c) HT



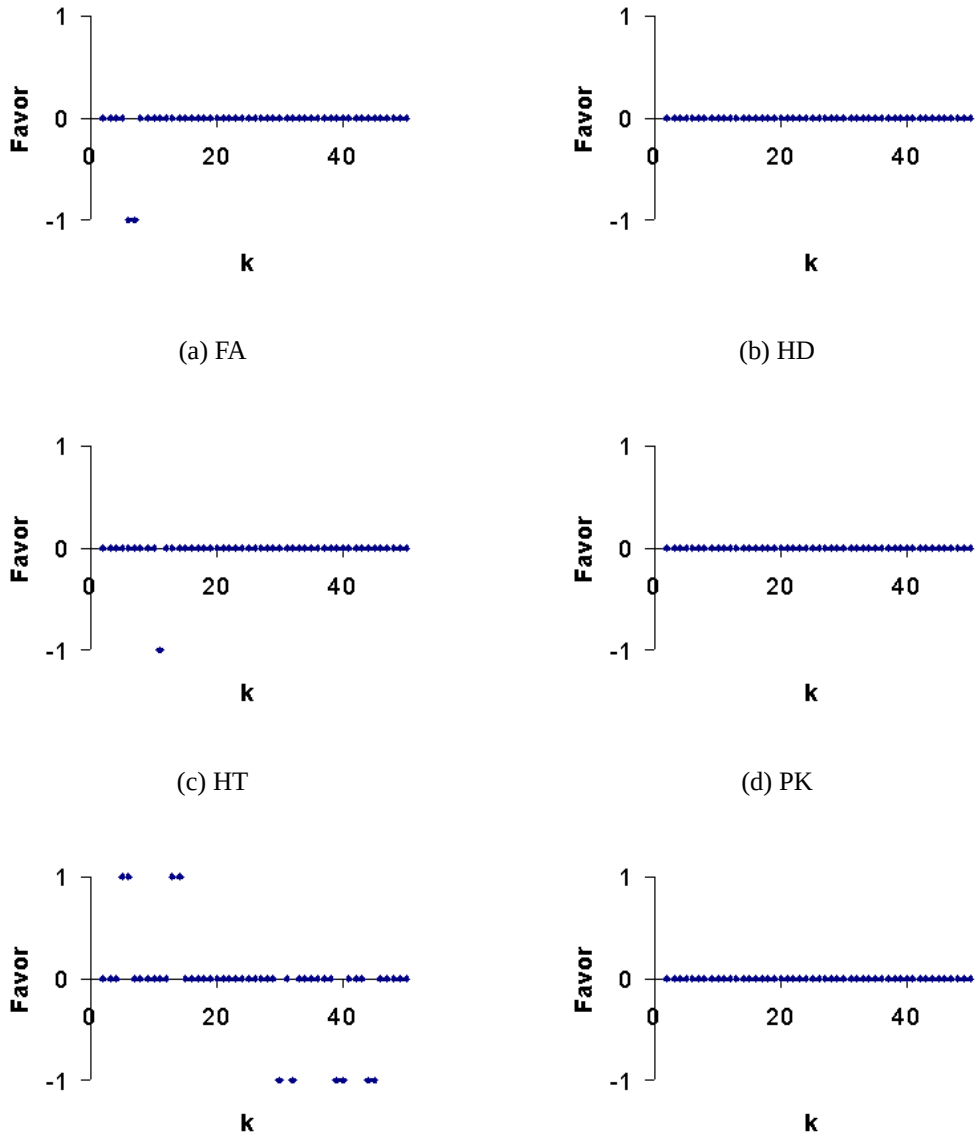
(d) PK



(e) SC

(f) TS

Figure 9.13: Percent of samples released for genetic datasets.

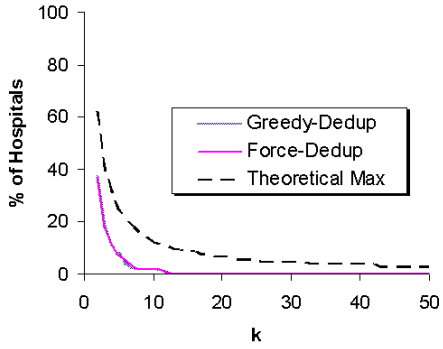


(a) FA (b) HD

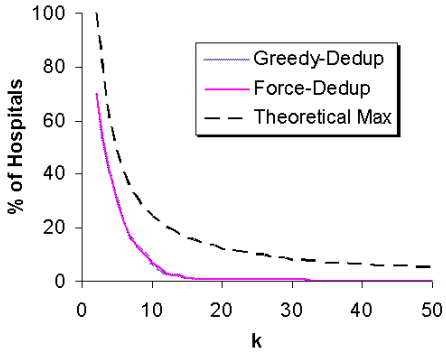
(c) HT (d) PK

(e) SC (f) TS

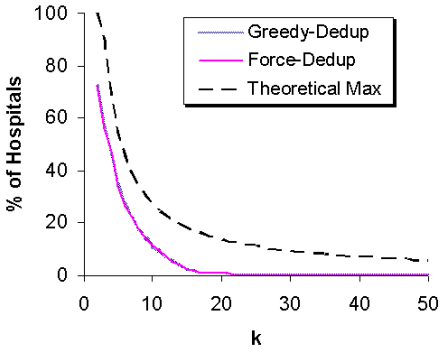
Figure 9.14: T-tests of percent of samples released for genetic datasets. A score of 1 implies *Greedy-Dedup* outperforms; a score of -1 implies *Force-Dedup* outperforms, and a score of 0 implies neither outperforms.



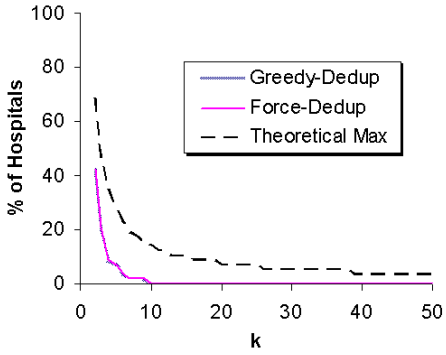
(a) FA



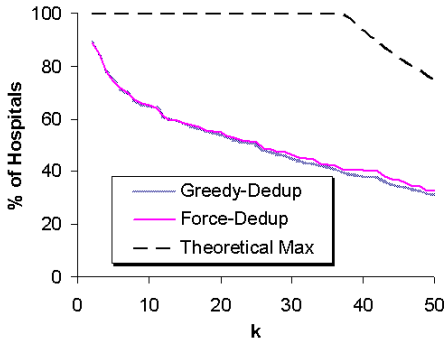
(b) HD



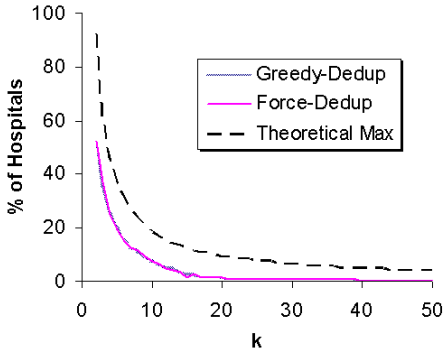
(c) HT



(d) PK

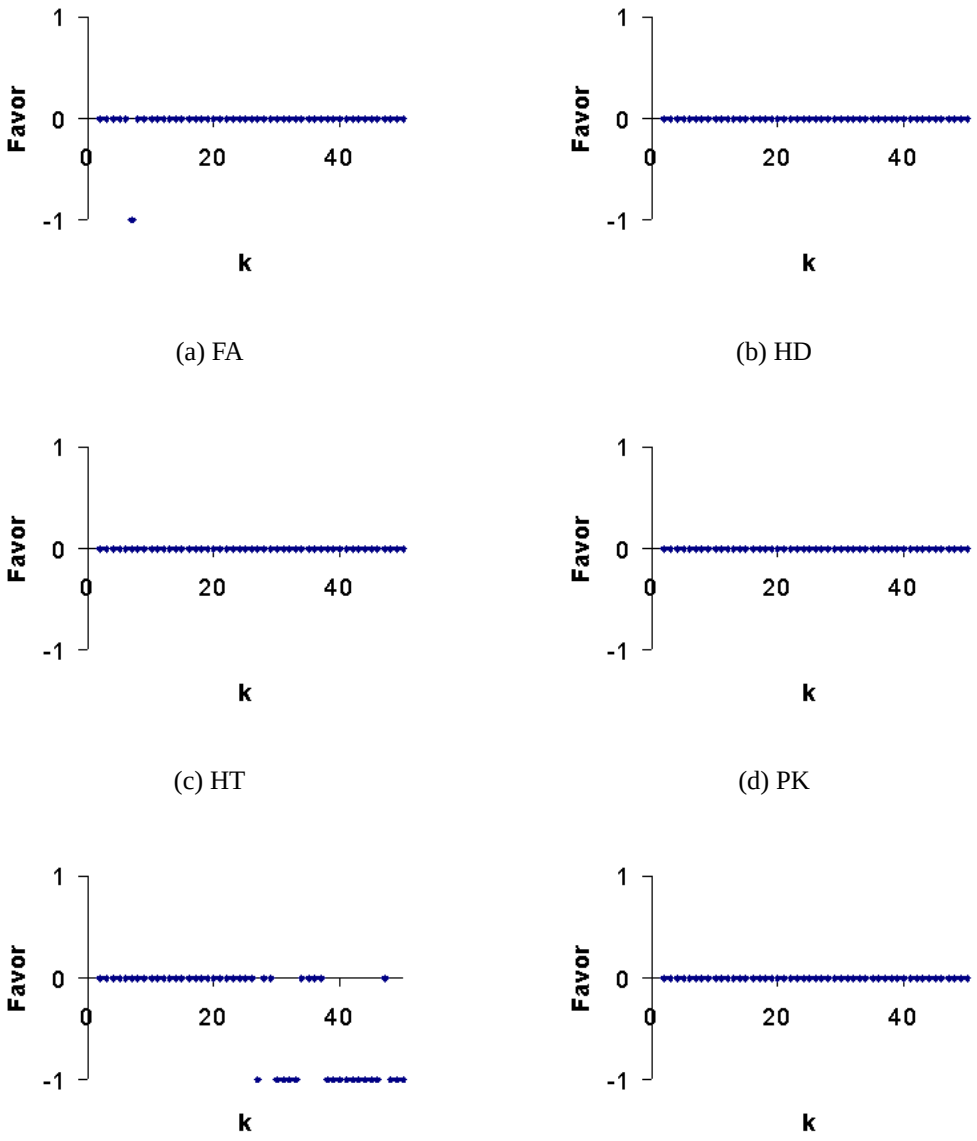


(e) SC



(f) TS

Figure 9.15: Percent of locations releasing non-null databases for genetic population datasets.



(a) FA

(b) HD

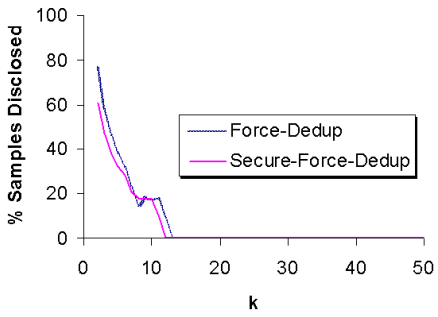
(c) HT

(d) PK

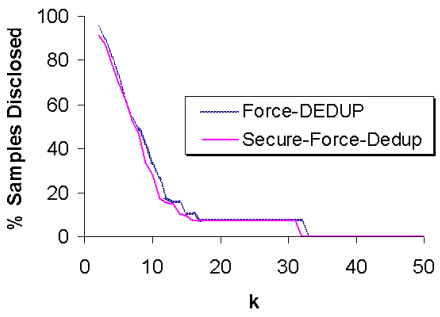
(e) SC

(f) TS

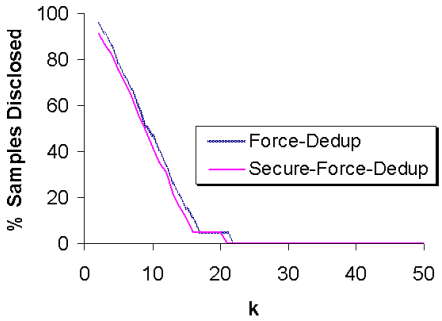
Figure 9.16: T-tests of percent of locations releasing non-null databases for genetic datasets. A score of 1 implies *Greedy-Dedup* outperforms; a score of -1 implies *Force-Dedup* outperforms, and a score of 0 implies neither outperforms.



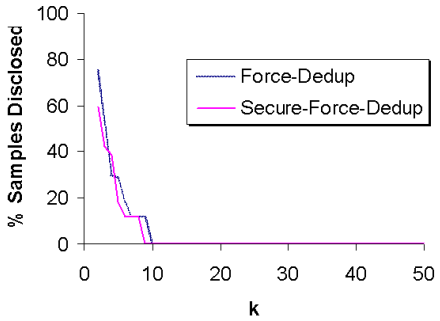
(a) FA



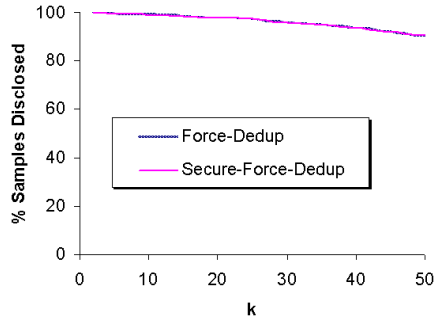
(b) HD



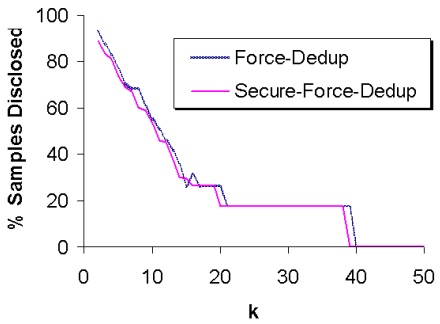
(c) HT



(d) PK

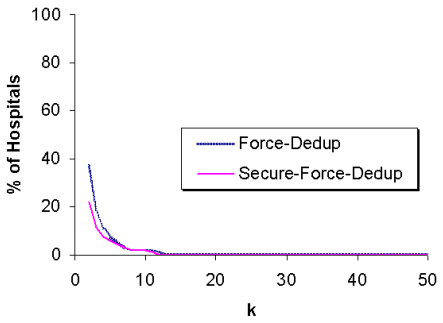


(e) SC

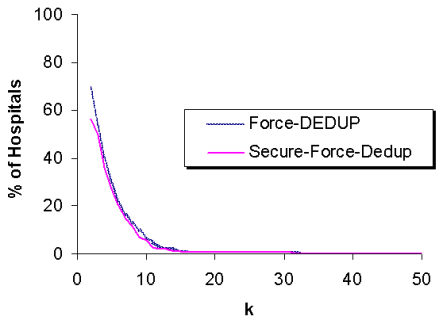


(f) TS

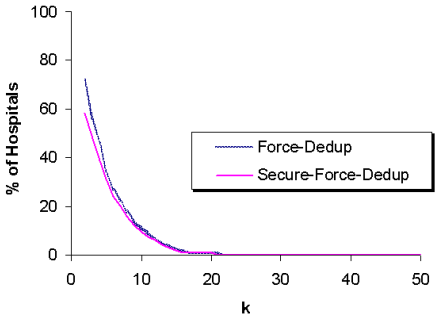
Figure 9.17: Difference in percent of samples disclosed by *Force-Dedup-Secure* in comparison to *Force-Dedup*.



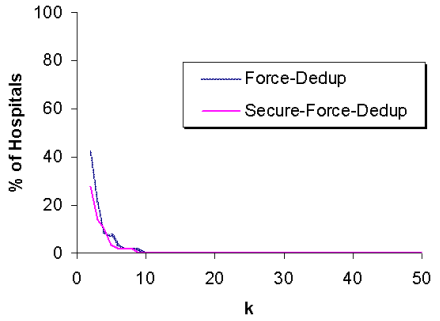
(a) FA



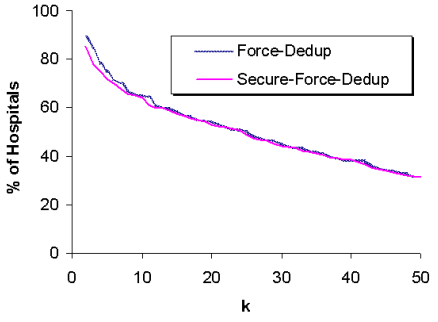
(b) HD



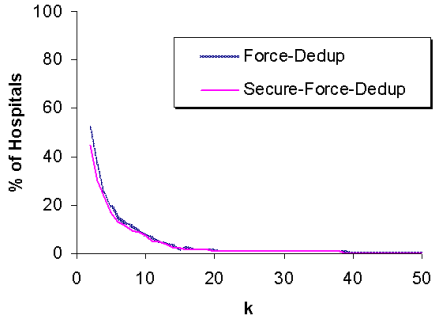
(c) HT



(d) PK



(e) SC



(f) TS

Figure 9.18: Difference in percent of hospitals disclosing by *Force-Dedup-Secure* in comparison to *Force-Dedup*.

Chapter 10

Conclusions and Future Research

This thesis set out to study and solve a data privacy challenge that arises in distributed systems. In this research, we investigated how an individual's location-visit pattern, or trail, can lead to the re-identification of seemingly anonymous data. In addition, this research investigated the degree to which formal models of privacy protection could be defined and implemented to prevent trail re-identification.

10.1 Summary of Contributions

The main contributions of this thesis are:

Formal Models of Trail Linkage. In Chapters 2 and 3 we presented a formal model of trails and how linkage between trails of disparate data types, such as personal names versus DNA sequences, can be achieved. The methodology involves constructing trails across locations from small amounts of seemingly anonymous or innocuous evidence that a person leaved behind at visited locations. Trails are also constructed on places where the person has left explicit information of their presence. Identifying uniqueness and inferences across these two sets of the trails relates information about where the person has been to who they are. In Chapter 2 we proved the maximum number of linkages can be discovered via basic graph theoretic techniques; however, these methods are inefficient . In contrast, In Chapter 4 we developed several automated pattern matching algorithms, collectively called REIDIT (RE-Identification of Data in Trails), that discover correct linkages in real time.

Formal Models to Prevent Trail Linkage. To prevent trail linkage, in Chapter 6 we proposed a novel formal protection model called k-unlinkability. The k-unlinkability

model is an implementation of the abstract k-map formal model introduced in earlier research by Sweeney [136]. Under the k-unlinkability model, it is guaranteed that every trail of one type of data, such as DNA sequences, is linkable to no less than k trails of another type of data, such as personal names. In comparison to earlier data protection models, we demonstrated that k-unlinkability protection allows for the retention of more variability with stronger claims of protection. In Chapter 7 we then developed several efficient algorithms to render trails k-unlinkable.

Evaluation in the Real World. Formal models of linkage and protection provide computational tools for compromising and upholding privacy. However, such tools are of little assistance if access to data is restricted by policy (i.e., there is no input into our tools) or if there is insufficient variability in data to permit sufficient linkage (i.e., output from our tools is of little assistance).

Re-identification Input. First, in Chapters 1 and 5 we analyzed several real world environments, including biomedical systems and the Internet, and demonstrated that data protection policies do not prevent the construction of trails. The results of this analysis provides a potential for trail re-identification.

Re-identification Output. Second, in Chapter 5 we performed experimental analysis of our re-identification algorithms on various real world and simulated populations. We empirically validated that there does exist enough variability in many populations' trails to lead to significant quantities of trail re-identification. These experiments justify trail linkage as a real data privacy threat.

On the flip side, we considered the degree to which we can protect trails from re-identification.

Protection Input. The basic data protection algorithms that are presented in Chapter 7 can be executed by the data holding locations, however, open collaboration is not always a plausible model given existing data protection policies. For example, data holders may not be permitted to view each others databases until data protection has been achieved. To address this issue, in Chapter 8 we extended the protection algorithms and the k-unlinkability model, so that protection could be performed and certified by a third party in an encrypted system. By doing so, data collectors do not have to reveal the plaintext contents of their databases until the disclosed trail matrices are guaranteed to be protected.

Protection Output. In Chapter 9 we applied our protection algorithms to the real world populations. Our results demonstrate that data holders can disclose large quantities of data that adhere to our formal protection models.

10.2 Limitations of this Thesis

This thesis defined a privacy risk as well as several solution to formally address the problem. However, the models and algorithms that are developed in this thesis are not without their limitations. In this section, we discuss several of the limitations of this work.

10.2.1 Geographic Constraints and Inference

One of the drawbacks to the k-unlinkability formal protection model is that it does not represent the probability that two elements are related. Rather it uses a boolean (i.e., could be related vs. could not related) characterization of linkage. As a consequence, the model neglects collocation patterns that can exist in the set of disclosed databases.

It may be possible that known collocation patterns, or geographic constraints, can limit the protection of k-unlinkability. If geographic or collocation constraints are appended to the existing boolean model, then an adversary may be able to learn that certain trails have a higher probability of being related than others. Keep in mind, when the adversary leverages collocation patterns that he observes, he may not be able to develop a computational proof that he has achieved a re-identification. The adversary will have to make a probabilistic argument regarding the certainty that two trails are related. Nonetheless, it is possible that such probabilities could lead to arbitrary degrees of certainty.

This is a different type of adversarial model than the k-unlinkability framework is designed to protect against. As it is currently defined, in a system that satisfies k-unlinkability, a data element can be linked to several data elements that have very different trails. When k-unlinkability can not rule out that two elements are related, based on their trails, then they are considered to be linkable.

10.2.2 Data Collection Assumptions

There are limitations that reach to the very core of our model. First of all, the REIDIT algorithms do not cover the space of data multiplicity and trail matrix reservation relationships. For example, REIDIT-I is correct only when one type of database is reserved to the other. In other words, if one location withholds genomic data, then our algorithms will only work when all locations withhold genomic data. Yet, if one location withholds

genomic data, and a different location withholds identified data, then neither trail matrix is reserved to the other. As a result, neither trail matrix may report the true number of entities in the system. The deterministic nature of the algorithms can not handle such a scenario. Use of the REIDIT algorithms can result in an increased number of false non-links. Even worse is that now the REIDIT algorithms can cause false links, which, given the current assumptions of our model, is impossible to achieve.

10.2.3 The Third Party

In the STRANON protocol, we used a trusted third party to analyze encrypted data and provide personalized responses to each of the data holders. In the realm of healthcare, trusted third parties are used for a number of functions, including data warehousing, data aggregration, and data brokering [59, 24, 30, 110, 99]. However, our model requires the trusted third party to act honestly with respect to private data.

Yet, if the third party does not act honestly, then there are a number of ways in which the third party can perform privacy compromising operations. We address some of these ways in Appendix C. The main concern is that such dishonest acts may be go undetected! Given the sensitive nature of the data in question, it is not necessarily the case that trusted third parties are an acceptable solution for preventing trail re-identification in the real world.

10.3 Directions for Future Work

Trail re-identification and protection is important to society simply because people seek safety without unnecessarily relinquishing their privacy. The *REIDIT* algorithms, and the associated re-identification experiments, exacerbate privacy concerns. The fact that trail re-identification can be done, as evidenced by the existence of this work, informs society, policy makers, and computer scientists of a real challenge to protecting privacy. However, hope is not all lost. The development of formal protection models, such as k-unlinkability and the *DEDUP* algorithms prove that data can be shared while defeating privacy compromises via trails. Our research provides models for trail linkage, protection, as well as validation. Nonetheless, to develop formal models for this research, we incorporated various assumptions and simplifications. As such, our models are a basis for the construction of more sophisticated systems. Throughout our presentation we touched upon various limitations and routes for extension. In this section, we restate and consider several potential future directions.

10.3.1 Probabilistic Linkage Models

The REIDIT and DEDUP algorithms are deterministic in nature, which is due to assumptions made regarding the static, traceable, uniqueness, and truthfulness of the data, as well as the manner by which data is collected. One of the core simplifying assumptions is in the use of trails of binary strings without any error. However, this is not always the case. In certain scenarios, typographical errors or false recordings of information in a database may occur. In this situation, not only can a * in a trail be replaced with a 0 or a 1, but a 0 may truthfully be a 1 and vice versa. As designed, the REIDIT algorithms can miss true re-identifications and cause false re-identifications.

An obvious extension of our research is in the design and evaluation of models that allow for the probabilistic qualification of trails. Additionally, in future research we must also address the case where neither trail matrix is reserved to other. For instance, in this case, one location may undercollect names on a certain portion of a population, but IP addresses on a different portion. This characterization of a trail opens up several branches of research.

There are several apparent extensions to our research. One possible direction is the development of trail re-identification methods based on record linkage models, such as those discussed in the related research section at the beginning of this chapter and in [95]. In this manner, we could phrase the trail re-identification problem as an optimization problem. Locations, or the combinations of such, could be afforded more weighting than others. In [57], a deterministic record linkage model is proposed, where feature selection of the best linkage attributes are determined. More complex record linkage model incorporate probabilistic models learn to account for typographical error [154, 158]. There is evidence of the successful application of such models by various federal agencies and healthcare organizations [13, 148]. For example, *John H. Smith* in database DB_1 and *Jon H. Smitth* in database DB_2 may both be the same individual, but neither *John* and *Jon*, nor *Smith* and *Smitth*, are equivalent.

Variations on these probabilistic methods may be useful for designing new trail matching models. Consider a simple reserved release: an identified track with two trails, Ali_Γ = [1,0,1] and Bob_Γ = [0,1,1], and a DNA track with two trails, $actg_\Omega$ = [0,0,1] and $ctga_\Omega$ = [1,1,1]. If each location has an equal amount of error in their released data, then no matches of identified to DNA trails can be made; both Ali_Γ and Bob_Γ differ from $actg_\Omega$ and $ctga_\Omega$ by 2 bits. However, when the first location is known to have a high rate of data error and the remaining locations have little or no error, then it is more probable that Ali_Γ and $actg_\Omega$ correspond to the same entity, and similarly for Bob_Γ and $ctga_\Omega$. Granted, the ability to make such a decision must be made in the context of the set of all trails in the matrices.

It may be that a combination of these methods will be necessary to ensure the maximum utility of the data. In order to do so, such techniques may be dependent on the data type considered and continued research will be able to answer this question for certain. Regardless, it is apparent that formal analysis of how to share data while maintaining privacy in distributed data is necessary.

10.3.2 Error, Ambiguity, and Variation

In a similar vein, we must address error, ambiguity, and variation that inherently exists in databases. For example, names are not unique identifiers for specific entities and, as a result, there exit many confounders to the construction of correct trails. Firstly, the data may consist of typographical error. In this case, the name ”John” may be accidentally represented as ”Jon” or ”Jhon”. There exist a number of string comparator metrics [154, 26, 151] to account for typographical errors, many of which are in practice by various federal statistical agencies, such as the U.S. Census Bureau. Yet, even when names are devoid of typographical errors, there are additional confounders to data correctness. For instance, there can exist name variation, where multiple names correctly reference the same entity. Or, more pertinent to our research, there can exist name ambiguity, such that the same name correctly references multiple entities.

In reality, before trails can be analyzed for linkage or protection purposes, we must account for the correctness of the data. Failure to ensure correctness can result in the inability to discover certain relationships or cause the learning of false knowledge. In this sense, our trail construction, linkage, and protection models would greatly benefit by incorporate probabilistic reasoning methods to help determine when multiple references correspond to the same entity or not. Therefore, even if resolution can not be completely achieved prior to trail construction, our models would benefit by reporting a likelihood that two trails correspond to the same entity, as opposed to our current “yes or no” reasoning. In [95] we review various resolution models which may be of assistance.

While all problems must be accounted for, in recent research, we have initiated investigations into how relational networks can assist in the resolution of ambiguities [88].

10.3.3 Fault Tolerant Hashing

The proposed secure multiparty protocol for trail protection (STRANON - see Chapter 8 is dependent on a fragile hash function that is not fault tolerant. In other words, if a patient’s data, such as a DNA sequence, is variable across data collectors, the modular exponentiation representation of the DNA will cause the third party to report a false negative match. Due to our assumptions discussed in the previous subsection, STRANON does not incor-

porate a metric to measure the similarity between two sequences. However, there exist an increasing number of proposed methods for measuring the similarity between strings in an encrypted environment [4, 7, 25, 27, 39]. There are competing models, but prior research mainly concentrated on two party string comparisons and are proof-of-concept. There are several models of interest. First, the model proposed by Atallah et al [7] returns an edit distance for two parties' strings. Second, the work by Cohen et al [27] propose using a secure dot product over a distribution of TF-IDF (term frequency - inverse document frequency) scores. Third, and most recently, several proposed models have computed distance based on encryptions of a string's piecewise components [4, 25]. We suspect the latter methods are scalable and amenable to our keyed cryptographic schema.

Yet, the incorporation of such methods into STRANON must be undertaken with caution. The use of such distance functions may provide the third party with additional knowledge to incorporate into its retained knowledge, and thus violate data use constraints of the HIPAA Privacy Rule. Specifically, beyond mapping encrypted to plaintext based on trails, the third party can reconstruct the distances of encrypted and plaintext data. When the set of distances are sufficiently similar, the third party can construct a more specific representation of retained knowledge to link de-identified data to its hashed values, and then finally to its identified data. More research into the protective capabilities of distance preserving encrypted string metrics is necessary before such techniques can be advocated.

Though the presented protocol is dependent on a hash function incapable of preserving string similarity, there exist several promising alternatives. In future research, we intend to evaluate their effectiveness within the multiparty protocol.

10.3.4 Alternatives to Trusted Third Parties

When we have sufficient confidence in the honesty of our third parties, there are a number of issues that must be be thought through before moving forward. For example, what are the properties that we should look for in a third party? In this sense, we must define criteria by which we can measure how much we believe a trusted third party will act honestly. Furthermore, if we are measuring honesty, then we need to figure out how, when, and what to audit with respect to the third parties procedures. This is by no means a trivial challenge.

A more attractive possibility is to remove the third party from the picture. Research in computational theory has shown that third parties can be removed without sacrificing the level of security in the original protocol [20]. However, the implementation of such systems are inefficient and unsuited for the real world applications that we address in this thesis. Therefore, if trail re-identification must be thwarted in an encrypted space, then a fruitful direction for future research is to construct secure multiparty computation

protocols that approximate the properties of the trusted third party without revealing the information the trusted third party receives.

10.3.5 Stopping Trail Re-identification Before it Starts

Trail re-identification is a real problem that can be thwarted after data has been collected. However, what if we could stop trail re-identification before it started? The research in this dissertation analyzed the trails of data post data collection, but if administrators and public officials wanted to minimize the threat of trails, then they should lay out locations in a manner that data collection minimizes the ability to perform trail re-identification. In this respect, the simulation studies in Chapter 5 provide insight into how we can organize a system of locations, such that it helps mitigate the re-identification risk for trails. For instance, if information is always to be released such that it is susceptible to REIDIT-C, then locations which capture data according to high skew distributions are more desirable, than locations that capture data according to less skew. In contrast, if the information has less certainty in the relations between trail matrices, then designing a system where location-based access is in the form of uniform distribution may be the best choice. Note that the word "may" is used because it is at this point where the brunt of the risk occurs. In a REIDIT-I environment, if the system falls into worst case location access scenario, such that the parametrization of the distribution maximizes re-identification, then the uniform distribution will reveal more re-identifications. However, if there is some doubt as to whether the parametrization will yield max re-identifiability, then one is actually better off in the uniform system. This is because of the finding that the average number of re-identifications is lesser in the uniform than in the high skew distribution. So, it appears that the question of which distribution will yield more re-identifications is a matter of how confident one predicts the parameter of the distribution by which subjects access locations. This raises several difficulties that extensions to this research should consider.

10.3.6 Systems Development and Adoption

We can not be complacent with the design and proof of formal models. For proposed data privacy solutions to be useful to society they must be adopted by real world systems. Trail linkage provides an architecture for mapping the trail of one type of data to the trail of another type of data. From a formal modelling perspective, trail linkage is well-defined and is amenable to various proofs.

The next step is to develop and integrate threat and protection models into existing systems. This will require adapting and integrating trail re-identification and unlinkability models into systems with varying idiosyncrasies and complexities that arise in specific ap-

plication domains in the real world. These issue, and others which touch on the semantics of privacy issues in society, must be addressed before trail protection models are deployed in distributed systems.

Part III

Appendices

Appendix A

Multiple Trail Re-identification

At times, the trail linkage problem as defined in Chapter 2 is inappropriate. The definition of the trail linkage problem in Definition 4 is acceptable only when we know each trail from one trail matrix has a corresponding trail in the other trail matrix. However, this definition is restrictive when a trail in one matrix has multiple corresponding trails in the other matrix. In other words this definition can not be used when an association relationship does not exist. Thus, we use the following definition for non-unique re-identification.

Again, the model has a relationship to graph theory. The definition corresponds to that of the strongly stable polygamous marriage problem [60]. In this problem, one of the genders is permitted to take multiple spouses; i.e., a polygamous marriage. A polygamous marriage is *strongly stable* if none of the multiple spouses have other partners with which they would be equally happy. In other words, let's say men are permitted to marry multiple women. If a woman is interested in a man besides her husband, then the marriage is not strongly stable. The strongly stable polygamous marriage can be defined directly from the link matrix (Definition 26).

Definition 26 (Strongly Stable Polygamous Marriage). *Let trails in X be permitted to marry multiple trails in Y. A trail $y \in Y$ is in a strongly stable polygamous marriage if only one cell in column y of L_{XY} is non-zero.*

A.1 Data Multiplicity

In this chapter we study data multiplicity characterized by a *one-to-many* relationship. In this type of relationship, a tuple of one type of data can correspond to multiple tuples of the other type. For example, imagine a scenario where a patient sheds different genomic sequences, each of which corresponds to a region of interest for differing hospitals' clinical

studies. The patient shares the same name with each hospital, but shares multiple unrelated DNA sequences (e.g., derived from different parts of the genome). In Definition 27 we functionally model the one-to-many environment. The right image in Figure A.1 depicts a one-to-many data relationship for the aforementioned example.

Definition 27 (One-to-Many Data). *Partitioned databases γ_c and ω_c are said to be one-to-many if there exists a surjective function $\beta : M \rightarrow N$.*

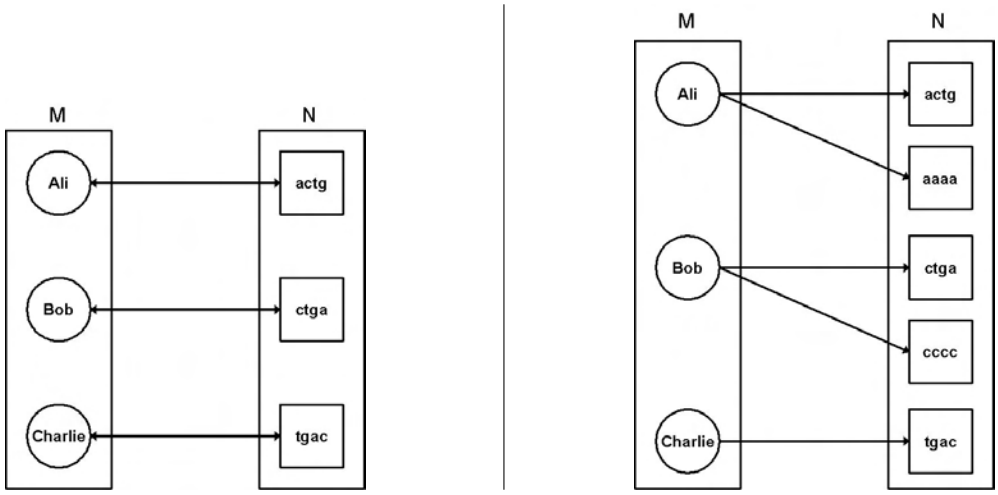


Figure A.1: *Left*) One-to-one data. There exists a bijective function $\alpha : M \rightarrow N$. ***Right***) One-to-many data. There exists a surjective function $\beta : M \rightarrow N$.

In Chapter 3, we presented two orthogonal features, specifically multiplicity and disclosure tactics, that influence the trail relations that can be utilized for linkage purposes. The REIDIT algorithms produce false re-identifications are made, but which algorithm is appropriate is dependent on the assumptions of the disclosure environment. Before delving into the details of the REIDIT algorithms, we summarize the appropriateness of each algorithm in the crossproduct of the feature space, which is shown in Table A.1.
In addition, we note REIDIT-C and REIDIT-I are designed to discover true loves, while REIDIT-M is designed to discover stable polygamous marriages. This issue will be addressed in the presentation of each algorithm.

A.2 REIDIT-Multiple

In Chapter 4 we introduced two trail re-identification algorithms, called REIDIT-C and REIDIT-I. In this chapter, we introduce a third trail re-identification algorithm that is called

		MULTIPLICITY of M-to-N	
		One-To-One	**One-To-Many**
DISCLOSURE TACTIC	**Unreserved**	*REIDIT-C#, -I*	*REIDIT-M*
	Γ_M **Reserved to** Ω_N	*REIDIT-I#, -M*	*REIDIT-M*
	Ω_N **Reserved to** Γ_M	N/A	N/A

Table A.1: Coverage of REIDIT algorithms over the space of disclosure variation, such that the algorithm produces true re-identifications and no false re-identifications. When multiple algorithms are applicable, a "#" indicates which algorithm is dominant, such that it guaranteed to find more trail re-identifications.

REIDIT-Multiple, or REIDIT-M. Pseudocode for the REIDIT-M algorithm is provided in Algorithm 16. It allows trails in N_Ω to be related to multiple trails in M_Γ. It performs subtrail matching akin to REIDIT-I, such that if a trail m_Γ is the subtrail of only one supertrail n_Ω, then a re-identification occurs between the corresponding tuples m and n. However, unlike REIDIT-I, the supertrail is not removed from further consideration.

Algorithm 16 REIDIT-M(M_Γ, N_Ω, M, N)

Input: See REIDIT-C.
Output: See REIDIT-C.

$R \leftarrow \{\}$
for each $m \in M$ **do**
  if there exists one and only one $n \in N$, such that $m_\Gamma \preceq n_\Omega$ **then**
    $R \leftarrow R \cup \{\langle TRAILDATA_\Gamma(m), TRAILDATA_\Omega(n)\rangle\}$
  end if
end for
return R

For example, imagine M corresponds to the names of webusers and N corresponds to the IP addresses of computers. Multiple individuals in a shared setting, such as a household in the Homenet dataset, can use the same computer. Online purchasers, in this case, would have multiple identities related to the same IP address. The reverse is also possible. One person could use more than one computer. In this case, one reference in N would relate to multiple references in M. The REIDIT-M algorithm addresses such issues of collocation.

REIDIT-M is appropriate when N to M is one-to-many and M_Γ is reserved to N_Ω. This is precisely why REIDIT-M does not remove elements from N, once they have been re-identified to elements in M. This remains the case even when $M = N$. To understand

why this is true, consider that it is possible $\beta(i) = \beta(j)$ for all $i, j \in M$ (See Definition 27). So, if $|N| > 1$, then it is unclear which tuples in N correspond to tuples in M, unless a tuple is a unique supertrail.

Theorem 9 (REIDIT-M Correctness in Reserved). *If trail matrix M_Γ is reserved to trail matrix N_Ω and N to M is one-to-many, then REIDIT-M outputs no false re-identifications.*

PROOF. Based on a similar argument made for REIDIT-I, since M_Γ is reserved to N_Ω it must be true that every trail $m_\Gamma \in M_\Gamma$ has a corresponding trail in $n_\Omega \in N_\Omega$, such that $\beta(m) = n$ (i.e., m and n were derived from a common tuple in a private database). Now, since Lemma 6 holds true for all $c \in C$, it must be true that the corresponding trail is a supertrail of m_Γ. REIDIT-M searches for re-identifications for m in the set of supertrails of m in N_Ω. Thus, for any two trails m_Γ and n_Ω, where m_Γ is not a subtrail of n_Ω, only true non-links will be recorded. In the event that there are multiple supertrails, no re-identification will be made and only false non-link will remain. Moreover, trails that are subtrails of m, but the elements do not belong together will be labelled as a true non-link. This accounts for all scenarios, and therefore REIDIT-M will never produce a false link. As a result, REIDIT-M produces the same contingency table as REIDIT-I in Figure 4.2. ■

Corollary 7 (REIDIT-M Correct in Unreserved). *If trail matrix M_Γ is unreserved to trail matrix N_Ω and N to M is one-to-many, then REIDIT-M outputs no false re-identifications.*

PROOF. This is a direct consequent of the fact that unreserved databases are a special case of reserved databases. ■

Note, Corollary 7 holds true when N to M is one-to-many, but it does not necessarily hold true when the reverse is true, i.e., M to N is one-to-many. This is so because Lemma 4 does not necessarily hold true. Thus, the corresponding cell in Table A.1 (i.e., N-to-M = one-to-many and M_Γ is reserved to N_Ω) does not contain REIDIT-M.

On another note, however, we point out that REIDIT-M achieves the same contingency table as REIDIT-I when M and N are one-to-one and M_Γ is reserved N_Ω. This is because one-to-one is a special case of one-to-many. As such, REIDIT-I dominates REIDIT-M in such an environment (Lemma 19).

Lemma 19 (REIDIT-I Dominates REIDIT-M in One-to-One). *If trail matrix M_Γ is reserved to N_Ω and data in M and N are one-to-one, REIDIT-I dominates REIDIT-C.*

PROOF. Let P_m be the set of supertrails in N_Ω for m_Γ. Since one-to-one is a special case of one-to-many, both REIDIT-I and REIDIT-M will include the correct corresponding trail to m_Γ in P_m for every $m \in M$. Now, if $|P_m| > 1$, then

REIDIT-M will automatically classify m_Γ as a false non-link. However, REIDIT-I may reduce P_m during its iterative process, and thus in a subsequent iteration $|P_m|$ may be 1. Thus, REIDIT-I dominates REIDIT-M. ■

A.2.1 Complexity and Efficiency

First, the outer loop iterates over all of the tuples in M, which is $|M|$ iterations. Second, for each iteration, the algorithm iterates a maximum of $|N|$ times searching for the set of supertrails. Thus, the REIDIT-M algorithm is O($|N| \cdot |P|$).

We suspect that the REIDIT-M algorithm can be made more efficient using sorting strategies similar to those used in REIDIT-C. However, since M_Γ is no longer guaranteed to be equivalent to N_Ω, we must perform sorting strategies based on each attribute of the trail, as opposed to a single base 10 representation. Such strategies will be desired when REIDIT-M is applied to large datasets, but for now, we leave this to future research.

A.3 Experiments

The Homenet dataset lends itself toward a natural one-to-many trail linkage scenario. We acknowledge that a household may have multiple users of a particular computer and for the following experiments, we assume each website releases a list of customers who made a purchase at the website. This list includes the email address, not the mailing address, of the purchaser. Thus, an IP address of a computer can now relate to multiple email addresses. The attributes released with the de-identified databases were {*website, IP address*} and the attributes released with the identified databases were {*website, email address*} for each targeted website location. As such, the de-identified trail matrix consists of 30 rows and the identified trail consist of 26 rows. The number of locations remains 24.

A.4 Batch Analysis

Recall, there were 26 households and 30 individuals (2 households with 2 individuals, and 1 household with 3 individuals). REIDIT-M achieved re-identification for 15 individuals to households, or 50%. Furthermore, re-identification was achieved for all members of the three households with multiple members. Note, REIDIT-I, failed to re-identify these individuals. In the Homenet dataset, family members visited common sites, which under REIDIT-I remain ambiguous at the individual level, but not for REIDIT-M at the household level. Sensitivity of REIDIT-M to single locations was analyzed as described before and results are shown in Figure A.2.

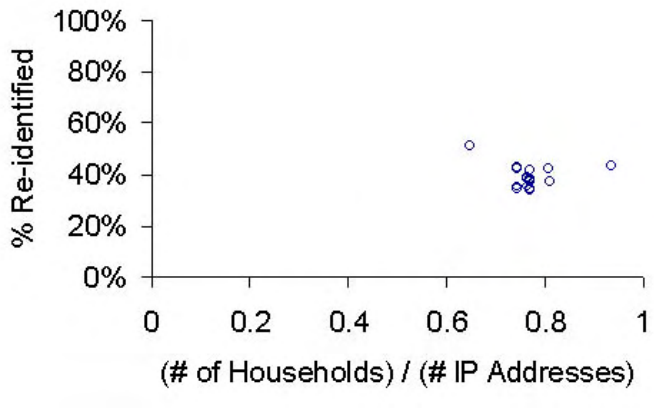


Figure A.2: Sensitivity of REIDIT-M re-identification to the removal of one domain's dataset removed. The values are slightly jittered for visual inspection.

A.4.1 Sensitivity Analysis

As stated above, there exist many households in the "general" Homenet population (30 of 86 households, or approximately 35%) with multiple users for a single IP address. In the following experiment, we allow an IP address to be re-identified to multiple identified individuals. We continue with the complete Homenet dataset for evaluation purposes.

In Figure A.3 we show the re-identification of multiple users to a single IP address as a function of the rank in website popularity. Initially, the number of people and the number of households re-identified are similar and grow at a similar rate as well. However, as depicted in Figure A.3 after approximately 60 websites, the re-identification of individuals begins to surpass the growth rate for households. From this test, it is apparent that members of the same household visit different sets of webpages. If this claim was not true, and members of the same household visited the same set of webpages, the growth rate of re-identification for individuals would be steeper than that of the household discovery rate at an earlier point.

While this analysis demonstrates that complete trails of IP addresses are complex enough to re-identify multiple individuals, more websites are needed for re-identification of the population than with REIDIT-C. This is not surprising because REIDIT-M allows searches for unique linkages whereas multiple linkages may require additional information.

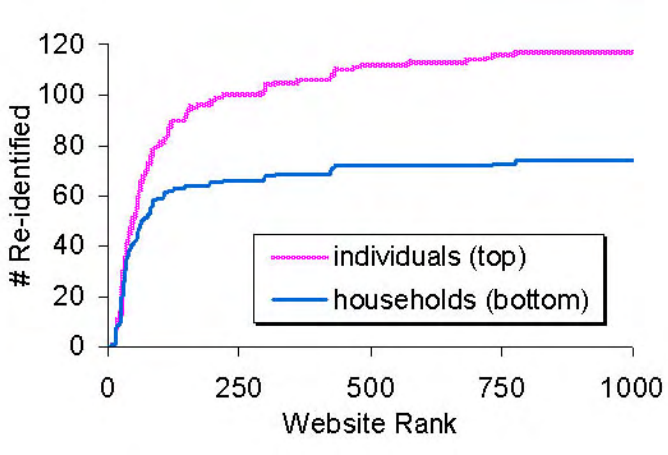


Figure A.3: Re-identification of individual users to IP addresses. The "individuals" line corresponds to the number of individuals re-identified, while the "households" line corresponds to the number of distinct households with a member re-identified.

A.4.2 Summary

This chapter extended the trail linkage model to account for one-to-many data relations. We developed a formal model of trail linkage when multiple data pieces of one type of data can be correctly linked to a single piece of another type of data. We extended the REIDIT family to include a new algorithm called REIDIT-Multiple, or REIDIT-M. We then applied the REIDIT-M algorithm to the Homenet dataset and demonstrated that multiple names could be correctly related to the same IP address. The REIDIT-M algorithm serves as an example of how formal trail re-identification models can be developed beyond traditional one-to-one data assumptions.

Appendix B

Genomic Privacy Protection Vulnerabilities

The biomedical community currently finds itself in the midst of a genomics revolution. Genomic data, combined with increasing computational capabilities, provides opportunities for health care that until recently were severely limited. Beyond gross diagnostics, mounting evidence suggests genomic variation influences disease susceptibility and the ability to metabolize drugs. As a result, genomic data is increasingly collected, stored, and shared in research and clinical environments [5].

The sharing and application of person-specific genomic data poses complex privacy issues and is considered one of the foremost challenges to the biomedical community [6, 147]. Many people fear knowledge gleaned from their genome will be misused, abused, or instigate social stigma for themselves or familial relations [116, 61]. This fear is exacerbated by the HIPAA Privacy Rule, under which genomic data is not considered an identifying patient attribute [32]. As such, genomic data may be released for public research purposes under HIPAA's safe harbor provision.[1] Yet, when genomic data is not publicly available, recipients may be subject to data use agreements. Though legally binding, there is no guarantee genomic data will be used according to specification. Thus, it is best that privacy laws are complemented with technology to assist in the enforcement of protections.

Privacy protection technologies for genomic data must address the question, "How can person-specific DNA be shared, such that a recipient can not sufficiently associate the DNA to its explicit identity (i.e., name, social security number, etc.)?" Though genome variation uniquely characterizes an individual, there exists no public registrar that maps

[1]For example, the PopSet database at National Center for Biotechnology Information contains publicly available DNA sequence data, which is not subject to oversight by an Institutional Review Board.

genomes to names of individuals. Over the past several years, many genomic data privacy protection systems have implicitly relied on this premise. These systems tend to separate DNA from explicit identifiers through methods ranging from simple removal of identifiers to strong cryptographic protocols.

This chapter addresses the extent to which current privacy enhancing technologies for genomic data are susceptible to compromise. Specifically, this work studies computational attacks that leverage information learned from shared genomic data, and additional resources, for linkage to named individuals. To summarize the findings: none of the systems analyzed are impregnable to re-identification. Rather, there exist patterns of flaws due to neglect of inferences that can be made from genomic data itself and the environment where the data is shared.

This chapter is organized as follows. In the following section, background on several published protection strategies for genomic data is provided. Each system is represented and discussed in a structured relational notation for comparative analysis. Next, computational re-identification methods for testing the protection systems are defined. With protection and re-identification methods presented, susceptibility analyses are performed and patterns of protection failures are discussed. This work concludes with a discussion on the need for research into formal anonymity protection schemas for genomic data and how such developments may proceed.

B.1 Current Privacy Protection Systems

In this section, four types of genomic data privacy protection systems are reviewed. Briefly, we review the following relational formalism to represent the systems. Person-specific data is organized as a table $T(A_1, A_2, \ldots, A_n)$ of rows and columns. Each column A_i is a semantic attribute, such as "date of birth", "DNA sequence", or "zip code". Each row is an n-tuple $t[a_1, a_2, \ldots, a_n]$, where a_i corresponds to a specific value of the i^{th} attribute. An identified table, T^+, includes explicitly identifiable information, such as name or Social Security number. Conversely, a de-identified table, T^-, is devoid of identifiable information. Figure B.1 provides examples of tables and tuples. For example the record t[***Bradley Malin***, ***000-00-0000***, ***BIGBM***, ***actg***] is a relevant tuple for table T(***Name***, ***Social Security Number***, ***Pseudonym***, ***DNA***). Adversaries are never provided with DNA in an identified table, so the DNA-identity mapping is unknown prior to receiving the de-identified table.

B.1.1 De-identification

The first type of protection system, adopted in a wide range of communities and environment, is based on *de-identification* (*DEID*)[17, 24, 160]. The data holder classifies

T			
T +		T -	
Name	Social Security Number	Pseudonym	DNA
John Doe	123-45-6789	JDFAKE	catg
Bradley Malin	000-00-0000	BIGBM	actg
Bob Smith	987-65-4321	FALSEBS	tgac

Figure B.1: The table *T*(*Name, Social Security Number, Pseudonym, DNA*) is data collected by a specific location. T^{+}(*Name, Social Security Number*) and T^{-}(*Pseudonym, DNA*), are identified and de-identified tables, respectively.

attributes into three types: explicit identifiers, quasi-identifiers, and non-identifying. Explicit identifiers consist of information that can directly reveal, or allow for contact with, an individual, such as name or Social Security number. A quasi-identifying attribute does not reveal identity by itself, but in combination with other attributes, can be used to link to other sources with explicit identifying attributes. For example, Sweeney demonstrated the values for date of birth, gender, and five-digit zip code uniquely characterize over 87% of the United States population [134]. Advocates of de-identification claim the corresponding identity of genomic data is sufficiently protected when explicit and quasi-identifying attributes are removed or generalized.

In *DEID*, the original table of a data holder takes the form T(*Explicit-identifiers, Quasi-identifiers, Non-identifiers*). When the data holder shares information, he removes Explicit-identifiers, generalizes values in Quasi-identifiers to prevent unique combinations. Thus, the data holder shares the dataset T'(*Quasi-identifiers, Non-identifiers*), where every value in the set of *Quasi-identifiers*$'$ is derivative of its corresponding value in the original set of *Quasi-identifiers*. In many situations, a unique identifier is assigned to a patient for linkage purposes. For instance, in the Utah Resource for Genetic and Epidemiologic Research (RGE) system, the unique identifier is a random number [160]. As a result, RGE data is released in a table T'(*Quasi-identifying Attributes, Other Attributes, Random Number*). Figure B.2 depicts a data release for a *DEID* system.

B.1.2 Denominalization

Systems based on denominalization (*DENOM*) are similar to *DEID*, except they incorporate structured coding, often for familial relationships [51]. In the original model, each patient is represented by six attributes $\{$*Individual, Family, Relation, Marriage, Sibling, Multiple*$\}$. *Individual* is a unique random number assigned to a patient, akin to the RGE system, which is used to manage the individual's clinical and biological samples. The

Attribute Class	Identifying	Quasi	Quasi	Quasi	Non
	Name	5-Digit Zip Code	Date of Birth	Gender	DNA
Original Table	John Doe	13579	1/1/1911	Male	catg
	Bradley Malin	24680	2/2/1922	Male	actg
	Bob Smith	12345	3/3/1933	Male	tgca

Attribute Class	Non	Quasi ′	Quasi ′	Quasi ′	Non
	Unique ID	5-Digit Zip Code	Date of Birth	Gender	DNA
Original Table	111111111	135**	1911	Male	catg
	222222222	246**	1922	Male	actg
	333333333	123**	1933	Male	tgca

Figure B.2: *Above*) Attributes of the original table are partitioned into explicit identifying (*identifying*), quasi-identifying (*quasi*), and non-identifying (*non*). *Below*) Identifying attributes are removed, the quasi attributes are generalized ($quasi'$). A unique ID has been added.

remaining attributes correspond to genealogical information. *Family* is a random number assigned to every member of the same family. *Relation* corresponds to the relationship of an individual to another family member, such as child or parent. *Sibling* denotes the birth order of a child (i.e., oldest, next oldest, etc.). *Marriage* specifies which marriage a child was born into. Multiple specifies which family a tuple pertains to when the individual is classified under multiple families.

The individual and family codes are managed independently. In the system description, it is claimed different levels of anonymity are achieved through the suppression, or withholding, of various attributes. For example, biological samples are considered to be sufficiently anonymous when stripped of the latter five attributes.

B.1.3 Trusted Third Parties

The third system (*TRUST*), introduced by deCode Genetics, Inc., facilitates data transfers via a trusted third party (TTP) intermediary empowered with full data encryption/decryption capability [59]. The full system consists of two protocols, both based on encryption and security. The first protocol facilitates discovery of research subjects, while the second specifies how biological samples are transferred to researchers. For brevity, we concentrate on the subject discovery protocol.[2]

Researchers initiate the protocol by communicating a specific disease of interest to

[2]Details on the second protocol and its mapping to this paper's formalism are available in reference 19.

physicians attending the patient population. The physicians create and send a population-based list $L\{$*Name, Social Security Number, Additional Demographic Features*[3], Disease$\}$ to the TTP. The TTP applies a reversible encryption function f to the Social Security Number (SSN) to derive an alphabet-derived pseudonym $f(SSN)$. Next, the TTP sends researchers the encrypted data, minus explicit identifiers, as a list $L'\{f($*SSN*$)$, *Disease*$\}$. Upon reception, the researchers match L' against f-encrypted genealogies linked to patient medical information. Based on this data, the researchers send a wish list of patients for further study, $N\{f($*SSN*$)\}$, back to the TTP. Finally, the TTP decrypts, appends the proper identifying information, and forwards the list $N'\{$*name, SSN*$\}$ to the appropriate attending physicians.

B.1.4 Semi-Trusted Third Parties

A fourth, and the most recent, system (*SEMITRUST*) was introduced by researchers at the University of Gent and affiliates [30]. Akin to *TRUST*, this system also employs third party, but one with restricted access to plaintext data, or a semi-trusted third party (sTTP). The third party is permitted to hold and distribute encrypted data only.

For the first step of the *SEMITRUST* protocol, the data holder constructs a list of identified individuals and their corresponding genomic data $L\{Identity, DNA\}$. The data holder applies public-key encryption function h to the Identity attribute and sends $L'\{h(Identity), DNA\}$ to the sTTP. Next, the sTTP applies its own public-key encryption function g to $h(Identity)$ to create $L''\{g(h($*Identity*$)), DNA\}$. In addition, the sTTP can act as a data broker for multiple data holders and can maintain a set of lists, $A\{g(h_A($*Identity*$)$,*DNA*$\}$, $B\{g(h_B($*Identity*$))$,*DNA*$\}$, ... , $Z\{g(h_Z($*Identity*$))$,*DNA*$\}$ for locations A, B, ... , Z. When researchers query the sTTP for data, they are supplied with doubly-encrypted lists. For additional data, researchers send requests onto the sTTP with a list of encrypted identities. In turn, the sTTP decrypts and sends the single-encrypted pairs onto the appropriate locations for additional data.

B.2 Re-identification Methods

In the following sections we briefly review four different types of re-identification techniques.

[3]The set of attributes Additional Demographic Features corresponds to demographic attributes deemed useful by deCode.

B.2.1 Family Structure

The first re-identification method (FAMILY) employs genealogical data accompanying genomic data. Genealogies, rich in depth and structure, permit the construction of complex familial relationships. Consider a simple family structure of two parents and one child. Since the parental genders are guaranteed, there exist 2 variants of this structure, since the child's gender is either male of female. When disease status is taken into account, it is represented as a Boolean variable; either an individual afflicted or not. In this aspect, all three family members can be represented as three attributes $\{$*Father*, *Mother*, *Child*$\}$, and there exist (father's disease status) * (mother's disease status) * (child disease status) * (child gender) = 2*2*4 = 16 possible family-disease combinations. In reality, pedigrees are much more robust than a simple nuclear family. For example, a three generation family of two children per family permits on the order of 10^5 distinct variants of the family-disease structure and 10^6 individuals that could be uniquely characterized. The number of combinations is larger when supplementary information, such as living status or medical/genetic features, is considered.

The ability to determine unique family structures is only one part of the re-identification process. These structures must be linked to identifiable information that, in many instances, is publicly available in the form of various genealogical databases. These databases are accessible both offline and via the World Wide Web. For example, genealogical records are available in many public databases, including Ancestry.com, Infospace.com, RootsWeb.com, GeneaNet.com, FamilySearch.org, and Genealogy.com. From such data, it is not difficult to construct family structures and, with such information in hand, an adversary can link disease labelled family structures to named individuals.

B.2.2 Genotype-Phenotype Inference

The second method relies on phenotype inferences for extracted from the genomic data (*GENPHEN*). Given two tables $X(A_1, A_2, \ldots, A_n)$ and $Y(B_1, B_2, \ldots, B_m)$ a set of relations is constructed and when a unique match is found between the two a re-identification is discovered. In the base case, this model is similar to the quasi-identifier based linkage model used in Sweeney's earlier work with health data re-identification [134, 135]. For example, consider *Health*(*Name, Address, Birthdate, Gender, Zip Code, Hospital Visit date, Diagnosis, Treatment*) and *Genomic*(*Age, Gender, Hospital Visit Date, DNA*). The set of extracted attribute relationships is $\{\langle$*Birthdate, Age*$\rangle$, $\langle$*Gender, Gender*$\rangle$, $\langle$*Hospital Visit Date, Hospital Visit Date*$\rangle\}$, but the set of relationships is expanded when relationships between clinical and genomic data are known.

To cope with the ever-increasing quantity of data produced from genetic research studies, and facilitate scientific discovery for clinical application, in 1987 the National

Center for Biotechnology Information (NCBI) established the Online Mendelian Inheritance in Man (OMIM) database [62]. The OMIM database is available online[4] and has been subdivided into several search mechanisms; for example, one feature permits specific searches corresponding to the keywords about genetic traits, another permits searches for the genome location of known disease genes. In general, the OMIM database is a catalogue of human genes and disorders containing textual information on data pertinent to a certain gene, as well as cytogenetic maps and reference information. In addition, the OMIM database lists the current resolution level of the chromosomal mapping of each gene or genetic locus entry, as well as important allelic variants that are known causes of clinical phenotype abnormalities (i.e., external qualities of a patient observed by physicians or other medical attendants).[5]

We cross-referenced ICD-9 codes[6] with key words from two publicly available resources provided by the National Center for Biotechnology Information (NCBI): a) the Genes and Diseases online book [49] and b) the Online Mendelian Inheritance in Man (OMIM) database [62]. Most of the genetic disorders listed on these websites are the direct result of mutations in a single gene. The ICD-9 codes represent disease that manifest as the result of mutations in single genes. The preliminary search has not been exhausted, since the names of some diseases are classified differently in the clinical information than its genetic counterpart. Examples of such well-defined diseases with different names in the database include diastrophic dysplasia, spinal muscular atrophy (SMA), and Angelman syndrome.

This is a non-exhaustive literature review and we discovered there exist at least 40 ICD-9 codes that can be related to over 35 DNA-mutation causing diseases. This information is depicted in Table B.2.2. Furthermore, pharmacogenomics continues to uncover relationships between genomic variation and the ability to process drugs and treatments [6, 147, 64]. Given such domain knowledge, it is possible to include $\{\langle \textit{Diagnosis}, \textit{DNA}\rangle, \langle \text{Treatment, DNA}\rangle\}$ relations.

In addition, by extending Sweeney's original work, it is possible to build systems that

[4]See `http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=OMIM`

[5]Other major central databases are specific to published mutations by gene and mutation type, such as the Human Gene Mutation Database [78] or by annotated sequence, such as the Database of Single Nucleotide Polymorphisms [129]. In addition to such databases, online websites, such as GeneClinics [42] at the University of Washington, have been implemented to facilitate the flow of information between the public and the medical genetics community. GeneClinics goals are to provide "disease-specific information on molecular genetic testing and its role in diagnosis, genetic counselling, and when appropriate, surveillance of at-risk relatives" [141]. Access to such information databases is critical to the progress of human mutation research [74].

[6]Available for download from the National Center for Health Statistics at `http://www.cdc.gov/nchs/about/otheract/icd9/abticd9.htm`.

Table B.1: Sample of diagnosis codes and gene counterparts derived from crossing the ICD-9 registry with NCBI's OMIM and Genes and Disease databases.

#	Disease Name	ICD-9 Code	Known Gene(s)
1	Adrenoleukodystrophy	3300	ALD
2	Amyotrophic Lateral Sclerosis (ALS)	33520	SOD1, ALS2, ALS4, ALS5
3	Burkitt's Lymphoma	2002	MYC
4	Chronic Myeloid Leukemia	2051, 20510, 20511	BCR, ABL
5	Cystic Fibrosis	27700, 27701, V181, V776	CFTR, CFM1
6	Duchenne's Muscular Dystrophy (paralysis)	33522	DMD
7	Ellis-van Creveld (chondroectodermal dysplasia)	75655	EVD
8	Essential Tremor (idiopathic) (autosomal dominant account for 1/2 of the cases)	3331	ETM1 (FET1), ETM2
9	Familial Mediterranean Fever (amyloidosis)	2773	FMF
10	Fragile X	75983	FMR1
11	Friedrich's Ataxia	3340	FRDA
12	Galactosemia	2711	GALT
13	Gaucher's disease (cerebroside lipidosis)	2727, 3302	GBA
14	Hemophilia Type A	2860	HEMA
15	Hereditary Hemorrhagic Telangiectasia	4480	HHT
16	Huntington's Chorea	3334	HD
17	Hyperphenylalaninemia (Phenylketonuria)	2701	PAH
18	Immunodeficiency with hyper-Igm (HIM)	27905	TNFSF5
19	Machado-Joseph Disease (Spinocerebellar Ataxia 3)	3348	MJD
20	Marfan Syndrome	75982	FBN1
21	Menkes Syndrome	75989	ATP7A
22	Methemoglobinemia	2897	HBB, HBA1, DIA1
23	Myotonic dystrophy	3592	DM
24	Pendred's syndrome	243	PDS
25	Prader-Willi Syndrome	75981	SNRPN
26	Refsum's Disease	3563	PAHX
27	Sickle Cell Anemia	28260	HBB
28	Spinocerebellar ataxias - or atrophy	3349	SCA1
29	Tangier disease	2725	ABC1
30	Tay-Sachs	3301	HEXA
31	Tuberous Sclerosis (Pringle's disease)	7595	TSC1, TSC2
32	Vitelliform Macular Dystrophy (Best Disease)	36276	VMD2
32	von Hippel-Lindau	7596	VHL
33	Werner's disease or syndrome	2598	WRN
34	Werdnig-Hoffmann disease	3350	SMA1
35	Kugelberg-Welander	33511	SMN/NAIP region
36	Wilson's Disease	2751	ATP7B

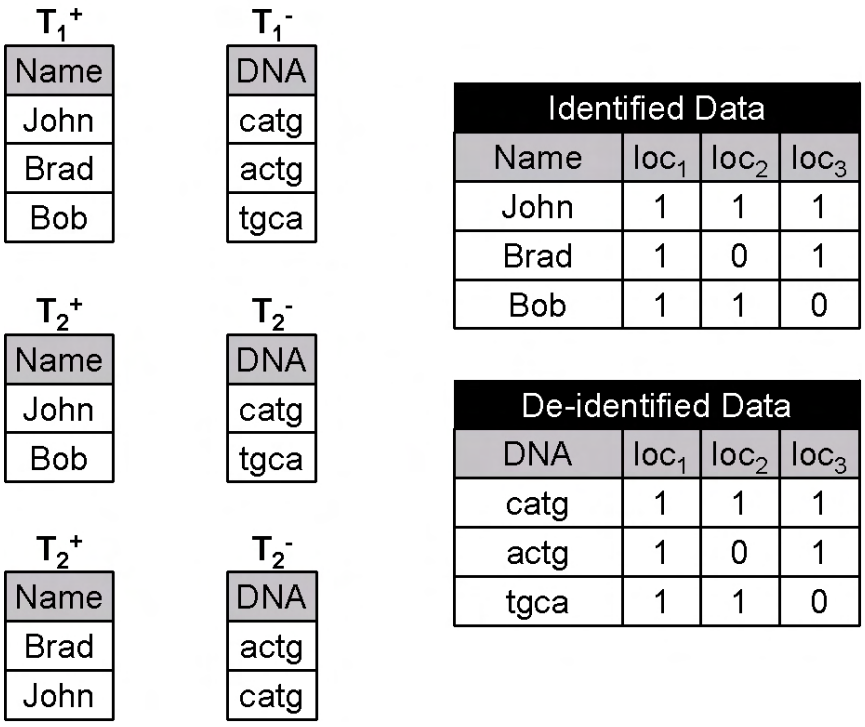
utilize attributes not observed in clinical or genomic information for linkage. When more complete clinical information is available, non-standard information, such as age of onset for progressive disorders, can be inferred. In previous research, we demonstrated how this could be achieved with longitudinal clinical information and Huntington's disease. Our system was able to infer age of onset within a 3 year period and subsequently match DNA to clinical data [93]. In its current implementation, this approach is applicable to any simple genetic disorder with defined clinical phenotypes.

An additional feature of the inference attack is it becomes more powerful with time. Since the goal of genomic medicine is to elicit the relationships between genomic data and clinical phenotype, the number of relations, and specificity of such, increase with advances in basic medical research. For example, the goal of the human genome diversity project and genomic anthropology is to pinpoint relationships between genomic variation and ethnicity. As a result, both the number, and specificity, of relations will expand, thus permitting an increasing capability for linkage.

B.2.3 Trails

The trail re-identification (*TRAIL*) methods utilizes location-specific information to match DNA to identity. To summarize chapter 2 of this dissertation, consider an environment with a set of locations, such as a set of hospitals, and a set of data subjects, such as a set of patients. Each location has the ability to collect multiple types of information, such as clinical and genomic data. To protect privacy when data is released, each hospital releases identified data and de-identified data separately. The first table released is T^{+}(*Demographic Information, Clinical Information*), where Demographic Information contains identifiable data. The second table released, T^{-}(*DNA*), consists of a list of genomic data samples.

An adversary retrieves data from a set of locations and creates two new tables, each one corresponding to location information for a particular data type. The first table consists of identified data, while the second consists of DNA data. The mapping of data to location is referred to as the data trail. In Figure B.3, trail are depicted as Boolean vectors; either a data value is observed at a location (1) or not (0). Details on trail matching algorithms and their application to real world populations can be found in chapters 2 and 5 of this dissertation. In short, genomic data left behind by an individual is matched to explicitly identifiable data based on the patterns of trails between the trail matrices.



T_1^+

Name
John
Brad
Bob

T_1^-

DNA
catg
actg
tgca

T_2^+

Name
John
Bob

T_2^-

DNA
catg
tgca

T_2^+

Name
Brad
John

T_2^-

DNA
actg
catg

Identified Data			
Name	loc_1	loc_2	loc_3
John	1	1	1
Brad	1	0	1
Bob	1	1	0

De-identified Data			
DNA	loc_1	loc_2	loc_3
catg	1	1	1
actg	1	0	1
tgca	1	1	0

Figure B.3: *Left*) Identified and de-identified data releases of locations $loc1$, $loc2$, and $loc3$. *Right*) Resulting Identified and DNA trail matrices created. When re-identification is based on exact trail matching, *John*, *Brad*, and *Bob* are re-identified to *catg*, *actg*, and *tgca*, respectively.

B.2.4 Dictionary Attack

The fourth re-identification method (*DICTIONARY*) is applicable when data is encrypted, or recoded, using non-random information. These methods, which obscure information can provide the basis for further erosion of patient privacy, beyond that of a susceptibility to the re-identification methods presented above. Consider a set of hospitals H, where each hospital $h \in H$ releases tables T_h^+ and T_h^- with attributes A_h^+ = {*name, date of birth, gender, zip code, clinical data*} and A_h^- = {*pseudonym*$_h$, *DNA*}. The attribute *pseudonym*$_h$ is generated through a reversible encryption function f_h, such as public-key encryption f_h(*Identity*, *key*$_h$) = *pseudonym*$_h$, where *Identity* is a tuple of patient information [*name, date of birth, gender, zip code*]. An adversary can use a trail attack to re-identify some of the patients released from a set of data releasing locations. Upon re-identification, a table with the attributes {*name, date of birth, gender, zip code, pseudonym*$_1$, *pseudonym*$_2$, . . . ,

pseudonym$_{|H|}$}, where *pseudonym*$_x$ is the pseudonym that hospital x uses for the identity of the patient. Thus, the adversary has achieved his goal of re-identifying the protected genomic data.

B.3 System Susceptibility Analysis

In this section, the general re-identification susceptibility for each of the protection methods is evaluated. The results are presented at a meta-level, such that either a system is considered susceptible or not. In Table B.2, a side-by-side comparison of protection model susceptibility is presented. Each of the protection models is susceptible to a minimum of three of the four re-identification attacks. Here, we discuss how each of the re-identification methods fares against the protection models in more detail.

Re-identification Attack	**Privacy Protection System**			
	TRUST	***SEMITRUST***	***DENOM***	***DEID***
FAMILY	Yes	No	Yes	Yes
TRAIL	No	Yes	No	Yes
GENPHEN	Yes	Yes	Yes	Yes
DICTIONARY	Yes	Yes	No	No

Table B.2: General susceptibility of privacy protection models to re-identification.

B.3.1 Family Susceptibility

The only model not susceptible to the family structure attack is the *SEMITRUST* system. Under this model, no familial relationships are considered in the genomic data. In specific cases, familial inferences may be possible, such as through haplotype analysis of DNA sequences. However, without more confidence regarding whether or not related family members are in the dataset, such analysis could create false family structures and familial relations.

It is interesting to note that the denominalization strategy behind *DENOM* strives to prevent the family attack almost explicitly. It provides protections by separating the individual from the family and using a local recoding of the identity. Yet, once this information is studied in a genealogical setting, the protections are minimal. Similarly, TRUST reveals genealogical information on a large scale, since this is how subject recruitment is performed.

In contrast, the RGE model of DEID is more difficult to analyze. As shown in Table B.2, the RGE model is susceptible to all re-identification attacks - though this may be somewhat deceiving. Since the RGE maintains a massive repository of diverse datasets, not all re-identification attacks can be performed on every dataset released. Thus, the analysis of re-identifiability for RGE released datasets is data dependent. Since RGE does have the ability to reveal genealogical information, and the only protection afforded to such data is de-identification and pseudonymization with random IDs, this model is susceptible to the family structure attack.

B.3.2 Trails Susceptibility

To construct a trail attack, two criteria must be satisfied. The first requirement is an individual's data is distributed over multiple locations. The second requirement is both genomic and identified data are available in partitions of the original collection. Table B.3 provides a characterization of which requirements the protection methods satisfy.

System	Multiple Locations	Partitioned Identified and DNA Data Available
TRUST	No	Yes
SEMITRUST	Yes	Yes
DENOM	No	Yes
DEID	Yes	Yes

Table B.3: General susceptibility of privacy protection models to trail re-identification.

The TRUST model does not satisfy the multiple location criteria. No location based information is revealed, nor is necessary. In addition, the *DENOM* model is not susceptible, since under the current version, genomic data is collected at one location only. Yet, if this model is applied to a distributed environment, then the trail attack is a feasible re-identification route.

In comparison, it can be verified that the *SEMITRUST* model does satisfy both criteria and is susceptible. The RGE model of de-identification is susceptible as well, since genomic data could be requested from multiple sources. The health-specific information could be either supplied directly as a separate source, or derived from various external resources, such as discharge information.

B.3.3 Genotype-Phenotype Susceptibility

This inference attack exploits relationships constructed between genomic data and known demographic or clinical information. As such, all four protection methods are susceptible to the attack, mainly due to the fact that the protection systems do not act directly on the genomic data. When considering simple versions of the inference attack, such as through direct ICD-9 linkage, with genomic data by itself, as is the case with the *SEMITRUST* model, this attack is dependent on the specificity of the known relationships between genomic data and clinical phenotype.

It is apparent that these methods can leak relationships that, though useful for research purposes and correlation studies, can allow for unique linkages to be constructed between identified and genomic data. This does not imply such relationships should not be inferable from shared data - rather the contrary. Yet, such inferences must be learnable, or communicated, in such a way that identities to which the data corresponds can not be determined. The concept of revealing inferences without revealing identity will be addressed below.

B.3.4 Dictionary Susceptibility

Models most susceptible to *DICTIONARY* use a single pseudonymization function, where pseudonyms are derived from patient-specific information. Since the RGE model uses random ID's for pseudonyms, a direct dictionary attack can not be achieved, regardless of the number of people re-identified through other means. In contrast, the other three systems are susceptible. The TRUST and *SEMITRUST* models are susceptible to a cryptographic dictionary attack . As an increasing number of people are re-identified, an adversary can collect a set of SSN, pseudonym pairs. Given enough pairs, the adversary may learn the key of the pseudonymizing function. In TRUST, the adversarial role can be played by any data requester. However, in the *SEMITRUST* model, this is not possible because the pseudonyms supplied to the researchers are doubly-encrypted. Though non-random, it is virtually impossible to discern the effects of the originating location's pseudonymizing function from the semi-trusted third party's (sTTP). Yet in the event the sTTP is corrupt, it can leverage the fact it receives single-encrypted pseudonyms from each of the submitting sources and attempt its own dictionary attack.

A modified version of the dictionary attack can be used to exploit familial relationship information released under the *DENOM* model. Given sufficient information to reconstruct and re-identify a certain amount of familial information, the recoding of familial relations can reveal additional information that may not have been learned in the family-structure attack, such as temporal information in the genealogy. For example, when a family has multiple children, the fifth cell of the family code, denotes what order of birth

a sibling is. Moreover, under the coding schema, this information is distinguishable for males, where the system uses even numbers, and females, where odd numbers are employed.

B.3.5 Compounding Re-identification

Many of the re-identification attacks presented in this paper are complementary. As a result, they can be combined to assemble more robust re-identification methods. For example, *FAMILY* can be used in combination with *GENPHEN* to construct more informative family structures, or with *DICTIONARY* when additional information about familial relationships is known. Moreover, an iterative process of alternating re-identification methods can be employed. Since different re-identification methods exploit different types of information, an adversary could use one method to re-identify a certain number of individuals in the population, then a second method to re-identify individuals not re-identified by the first or until certain confounding entities were removed from consideration. This process can continue as many methods necessary, or repeat with the same methods, until no more re-identifications are possible.

B.4 Discussion

To an extent the re-identification methods used in this study can be used to evaluate privacy protection technologies beyond those specifically designed for genomic data. The sole re-identification method directly dependent on genomic data is the *GENPHEN* attack, yet at its foundation, this method is based on the explicit representation of inferences between data types. As such, it is adaptable for other types of data relations. However, a note of caution. Before re-identification susceptibility for additional types of data can be claimed, a careful analysis of the social setting and attendant protections must be made. Though linkage of data types may be possible, it must be validated that access to such data is equally accessible. With respect to genomic data, its status as a lesser protected data type allows for re-identification using the above methods.

Given the current state of privacy protection systems, there exists a need for a new type of genomic data privacy protection model. In this sense, the results of this evaluation are a call to arms. Researchers must develop privacy protection methods that incorporate guarantees about the afforded protections. New methods must account for multiple environments of data sharing, as well as the type of inferences that can be gleaned from the shared data itself. These methods must be developed in a more scientific and logical manner, with formal proofs about the protection capabilities and limitations afforded by

the specific method. Though proofs may be difficult to derive in the face of uncertainties about the sharing environment, especially when the data itself holds latent knowledge to be learned at a later point in time, researchers can validate their approaches against known re-identification attacks in a logical manner.

B.4.1 Pseudonyms and Linkage

Based on the system analyses above, it is apparent the application of pseudonymization and naïve de-identification alone are not sufficient as proofs of identity protection. Mainly, this is because current systems tend to be narrow in their consideration of what is inferable from genomic data, as well as what additional information is available for relating genomic data to identified data. Yet, this does not imply pseudonyms and third party solutions are worthless in the pursuit of genomic data privacy protection. Rather, to some extent, these systems do provide a level of privacy protection and additional functionality for data sharing.

First, pseudonyms serve as a first-order protector and deterrent. It is conceivable an adversary, who approaches re-identification in a non-computational manner, will be deterred by the simple obscuring of explicitly identifiable information. Second, datasets devoid of linkage capabilities severely limit the types of research that can be performed. It is often the case where researchers may need to request additional information about a subject. Third, a subject may wish to remove their data from a research study or audit how their data has been accessed. Yet, if a pseudonym, or linkage value, is to be used as a primary key, it must be chosen appropriately. It should not be based on personal demographics as is currently the case with the *TRUST* and *SEMITRUST* models. A pseudonym based on this type of information is susceptible to dictionary attacks. Consequently, the RGE form of *DEID* and the *DENOM* models are more secure in their protection of linkage capabilities.

B.4.2 Accounting for Genomic Data

A common reason for re-identification susceptibility is the uniqueness of data that permit matching. One promising direction for research is the construction and analysis of systems based on formal computational models, such as k-anonymity [138]. Under the model of k-anonymity, every released record is indistinguishable from k-1 other records in the release. Within the genomics community, the k-anonymity model, under the term binning, has recently been adapted for the protection of single nucleotide polymorphism (SNP) data [83]. For example, consider the employment of the DNA generalization hierarchy in Figure B.4. If we wish to generalize the nucleotides C and G together, we only need to generalize up one level, and release R and R. To relate A and T, we must generalize to the

indeterminate character N.

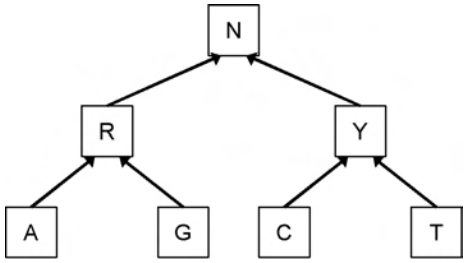


Figure B.4: SNP generalization hierarchy for purines and pyrimadines.

Though it has not been presented as a full system or for general genomic data, the binning method is a feasible solution, and worthwhile area of study, for genomic privacy protection. This is especially so, since such models are amenable to proofs of withstanding various re-identification attacks. However, this research is in a nascent stage and there are several deficiencies in the current binning model which researchers can build upon for more robust protection models. First, this model is restricted to SNP data and not more general genomic data. For a privacy protection system to function in the real world, it must be able to account for complex genomic features, such as nucleotide repeat structures and complex mutations.

Second, current binning models measure the amount of information lost via protection using an information theoretic perspective. While this is one way to characterize information loss, it does not take into account what the data is to be used for. Though formal protection methods, such as k-anonymity, advocate the direct manipulation of data values, there is no guarantee it will hinder applications or data usefulness. For example, in the statistics community, there has been much research into the design of formal protection methods that influence individual records but permit the recovery of aggregate statistics [41, 37]. More relevant to the genomics community though is recent research in privacy preserving data mining, where the methods privacy preserving methods are being validated with objective functions, such that logical rules or classifiers can be constructed with formal privacy guarantees about the data values shared [3, 2]. The development of genomic data privacy methods that incorporate models of utility is an open and fruitful direction of research.

From an opposing perspective, research should not remain content with their assumptions of how data sharing environments are organized and how re-identification can be performed. New re-identification attacks will be developed by those in the academic community, as well as adversaries outside the public realm. As such, researchers must continue to innovate and develop new methods re-identification for testing their protection tech-

niques. These methods may be new types of inferential or location-based techniques or completely new models yet to be discovered. Without the development of new protection and re-identification methods, researchers will continue to rely upon unfounded and possibly dangerous methods of privacy protection. The development of new identity protection strategies is paramount for continued data sharing and innovative research studies.

B.5 Conclusion

This chapter provided an analysis of the re-identification susceptibility of genomic data privacy protection methods for shared data. The results prove the current set of privacy protection methods do not guarantee the protection of the identities of the data subjects. This work stresses that a new direction in the research and advancement of anonymity protection methods for genomic data must be undertaken. The next generation of privacy protection methods must account for both social and computational interactions that occur in complex data sharing environments. In addition, privacy protection methods must provide proofs about what protections can and can not be afforded to genomic data, as well as the limits of research with protected data. The development of new identity protection strategies is paramount for continued data sharing and innovative research.

Appendix C

Secure Centralized Multiparty Computation

[1]

As technologies for collecting information infiltrate society, the ability to record and store personal information about specific individuals continues toward ubiquity. Knowingly and unknowingly, individuals shed data to a number of data collectors both within, as well as beyond, the confines of one's home. The information collection can be overt and apparent to the individual, such as when a consumer visits a retail store and completes a purchase with a personal credit card. Or data gathering can be less discernable, as when an individual's image is captured by an unforeseen video surveillance system. Regardless, the collection, storage, and sharing of personal information is becoming more widespread. [137]

In many instances, it is the interest of disparate data collecting locations to combine their data to learn more robust information. Though locations wish to collaborate, it is preferable not to reveal information that may compromise proprietary or strategic knowledge, overstep the boundaries set forth by legal statutes, or negatively affect individuals from whom the data was derived. Researchers in theoretical [161, 55, 19] and application-based multi-party computation [72, 85] have proposed methods to allow locations to collaborate by communicating only encrypted data. While the current techniques are useful for enabling encrypted data comparisons, they are hindered in their general applicability due to certain assumptions regarding the honesty of participating parties.

Over the past several years, various multi-party computation schemas have been applied to demonstrate how certain data mining endeavors, such as association rule learning, decision-tree construction, and basic machine learning methods can be achieved in

[1]This chapter was previously published as reference [89].

an encrypted setting. [85, 71, 146, 21] The specific type of multi-party computation this research generalizes is based on quasi-commutative cryptography, shown to be applicable for distributed frequent itemize mining. [71, 72] Though encrypted data analysis is achieved, it has been depicted in a proof of concept manner, rather than from a security perspective. Thus, in this paper we develop a protocol to perform distributed data analysis in a manner that adheres to more stringent security requirements. In addition to making the previous multi-party computation more secure, we provide intuition into how such a protocol can be configured for a number of different distributed data analysis. Most importantly, the protocol herein permits for each participating location to receive a differential response, which can be tailored to their data submissions.

Most secure multi-party computation schemes are designed under an expectation that the majority of participating parties are *semi-honest*. In the semi-honest model, participants are expected to follow protocol specifications, but they record intermediate values observed during the protocol that can be employed to compromise security. This is a widely used assumption in multi-party system analysis, however, it does not cover the space of adversarial models. When locations are *malicious* or corrupt, they attempt any number of techniques to gain an advantage over other locations, influence results, or simply wreak havoc. For example, consider multi-party protocols that require all locations to perform some action over every location's dataset [72]. When participating locations receive different data analysis results, a malicious participant can drop out of the protocol once it learns the contents of its results, thus preventing other location's from learning their own results. In previous multi-party models such problems were attended to by limiting the data analysis to a single global result that was broadcast to all participants. Yet, as will be shown, such limitations are unnecessary.

In this chapter, current methods will be extended and a protocol for secure centralized analysis of multiparty data that copes with malicious participants will be introduced. We provide proofs of additional security and integrity features that are not guaranteed in prior multi-party computation methods once semi-honest assumptions are relaxed. From a general perspective, the SCAD protocol allows for several new security features that garner special attention. First, the protocol is guaranteed to be collusion resistant. No location can collude with another location to bound or learn exactly the plaintext values in another location's dataset. Second, our model protects the integrity of every participating location's dataset. No location can maliciously target another location's dataset and tamper with values without being detected. This is a concern in previous models as will be discussed later. Third, we incorporate a component for a locking mechanism to prevent any location from observing plaintext data until all locations can correctly decrypt their own results. The level of protection afforded in the latter two features are probabilistic, but are specific to each location, such that each participant determines the appropriate amount of

security necessary for their own data.

The SCAMD protocol itself is not completely devoid of trust requirements. In order to achieve the aforementioned properties, a semi-trusted third party is incorporated to perform honest data analysis. The third party is trusted to receive and analyze encrypted data only. Yet, the use of a third party requires no more trust than in previous models, and actually permits the protocol to be more trustworthy. In comparison to previous models, where each participant must be viewed with a certain amount of skepticism, the third party model allows for participants to place their trust in a single party. This is especially useful since the lone trustworthy party has no data of its own at stake.

The remainder of this chapter is organized as follows. In the next section, relevant concepts from multi-party computation and encryption are reviewed. In section 3, we present the basic communications and data transfers that comprise the core of the protocol. In section 4, we develop protocol extensions particular to security and integrity, as well as prove their protective properties. Computational and bandwidth requirements of the protocol and its extensions are also addressed. In section 5, we demonstrate how the modular design of the protocol addresses computational concerns and permits different types of data analysis, such as differential encrypted response and centralized broadcasting. As an example, we map previous distributed analyses into the architecture of our protocol. Finally, limitations and possible extensions to this work are discussed.

C.1 Quasi-commutative Encryption

The protection protocol described below makes use of an interesting concept from cryptography known as the one way accumulator, or OWA. [10] In related research, OWAs were applied to a variety of distributed secure computations. For example, Zachary [162] demonstrates OWAs provide the necessary features for securely testing membership of nodes in distributed sensor networks. From another perspective, Faldella and Prandini [44] make use of OWAs for certificate authentication in a distributed public-key infrastructure. Most recently, and the work this research is closest to, Kantarcioglu and Clifton [71, 72] apply OWAs for data mining distributed association rules.

The protocol herein also employs OWAs for computation in a distributed environment. With respect to this research, the reader should view an OWA as a function to empower disparate locations, using different encryption keys, with the ability to reveal encrypted information from their local datasets, such that an encrypted piece of data is an equivalent primitive across locations. The OWA applied in this manner permits analysis and protection strategies to be executed over encrypted data. Plaintext information need not be revealed, unless it is desired by the owner of the data.

First, we review the general concepts of OWAs, then their transformation into keyed cryptosystems. Basically, an OWA is a hash function $h : X \times Y \rightarrow X$ that satisfies the *quasi-commutative* property. In equation (C.1), the following property holds for an arbitrary number and ordering of y_i.

$$h(h(x, y_1), y_2) = h(h(x, y_2), y_1) \tag{C.1}$$

In prior work, Benaloh and de Mare recognized that the modular exponentiation function $e_n(x, y_i) = x^{y_i} mod(n)$, as defined in RSA encryption, is an OWA. [10, 115] For appropriately chosen n, where n is the product of two large prime integers p, q, computing x from $e_n(x, y_i)$ and y can not be accomplished in polynomial time. Since repeated use of e_n may reveal hash collisions, values of n are further restricted to be chosen from the set of *rigid integers*, defined as the products of two *safe* primes p, q. A prime number p is safe if $p = 2p' + 1$, where p' is an odd prime. To provide some intuition, a safe prime is a large prime number that makes collisions of hashed values very unlikely to occur. Additional information about the features of p and q, such as congruency and collision-resistance requirements, can be found in [10] and [8].

While other types of accumulators exist [122], with an RSA basis, the quasi-commutative accumulator allows for trapdoor recovery of plaintext values. As a result, OWAs can be converted into asymmetric keyed cryptosystems. In order to do so, each encryption key y_i is paired with a decryption key z_i, where $y_i * z_i = 1 mod(\varphi(n))$, for some function $\varphi(\cdot)$. The term $\varphi(n)$, Euler's totient function, specifies the number of relatively prime positive integers less than n. When y_i and z_i are defined in this manner, decryption of an encrypted value v can proceed over m independent locations as

$$x = (h \ldots h(h(v, z_1), z_2), \ldots z_m) \tag{C.2}$$

Again, the ordering of the decryption keys $z_1, z_2, \ldots, z_m$ is of no consequence. Thus, the encrypted value v can be decrypted in a sequential manner using the same hash function as $h(x, z_i) = x^{z_i} mod(n)$.

C.2 Basic Communication Protocol

In this section, we introduce a protocol for the secure transfer and analysis of distributed data. The protocol is called SCAMD for secure centralized analysis of multi-party data. As the name implies, the current implementation requires a central authority, which we assume is semi-trusted. More specifically, it is trusted to receive and analyze encrypted

data, but not plaintext. The central party will collect encrypted data from each of the data releasing locations and is expected to return honest responses, to known questions and/or analyses, to each location. In previous research, others have proven that the responsibilities of a trusted third party can be distributed among the participants of the protocol. [161, 55, 19] However, when such a feat is achieved, it usually occurs via the sacrifice of computational complexity, such that the protocol may be infeasible to compute given temporal constraints. Moreover, most protocols deficient of a third party assume participants to act semi-honestly, which requires they follow the specifications of the protocol. With the incorporation of a semi-trusted third party, the central authority, the SCAMD protocol can account for any number of malicious locations. Though a certain amount of trust is still necessary with respect the central authority, the protocol shifts trust from each of the participating locations, to a single location with no data of its own at stake in the process.

We begin with a general overview of the protocol. A more in-depth description and formal treatment follows. First, each location encrypts every other location's datasets. Then, the central authority is provided with the encrypted datasets. The central authority performs some function over the submitted datasets and returns a list of encrypted values to each location. The encrypted values are decrypted by the set of locations, such that the final decrypter is the location the list was destined for.

More formally, the SCAMD protocol is defined as follows. Let there exist two types of participants, data locations $L = \{l_1, l_2, \ldots, l_{|L|}\}$ and a single central authority C. Each location $l \in L$ maintains three pairs of encryption-decryption keys, $\langle y_l^b, z_l^b \rangle$, $\langle y_l^r, z_l^r \rangle$, $\langle y_l^m, z_l^m \rangle$, for an agreed upon quasi-commutative hash function h as defined above. The function h is made public, however, all keys are kept private to each location. The first two key pairs are used for blinding purposes only by location l with its own dataset, akin to the blind signature process defined in Chaum's original description of untraceable payment systems. [22]. The first key pair blinds (superscript b) the data so that it can be digitally signed by every location with their multi-party encryption key (superscript m). The second key pair blinds the data after the central authority has returned its computation. Thus, this key pair serves for recollection (superscript r) of the plaintext data via decryption with every party's multi-party decryption key.

For simplicity, we represent location l's dataset as D_l and the set of encrypted values as $h(D_l, y)$. Additionally, the number of records in a dataset is represented as cardinality, or $|D_l|$. We now step through the basic protocol. A protocol over two locations is shown in Figure C.1.

Step 1. (Blinding for Encryption) Each location l creates a dataset of "dummy" values and adds them to dataset D_l. The specifics of the dummy values will be made more clear below. However, for the curious reader, it should be noted that its purpose is for the control of a particular probability. Then, l encrypts each value in D_l using y_l^b. After this initial

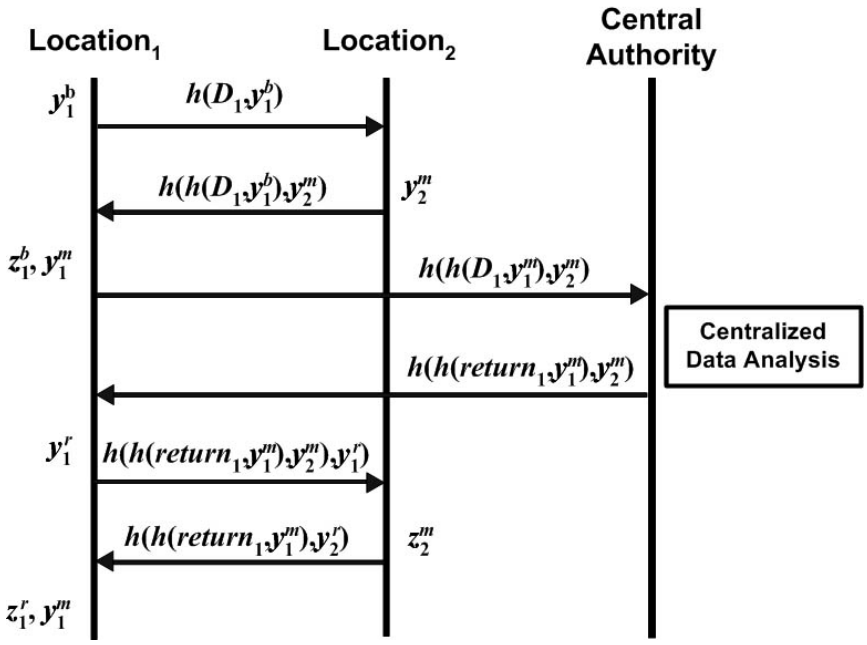


Figure C.1: Basic SCAMD protocol as executed by location 1, for scenario with two locations and central authority.

encryption, a blinded dataset $h(D_l, y_l^b)$ exists for, and is in the sole possession of, each location.

Step 2. (Full Encryption) Each location $l \in L$ shuffles and encrypts its own blinded dataset with y_l^m and sends it to other locations $x \in L$ in a sequential fashion. Each location x encrypts the received dataset with y_x^m and sends the dataset back to l. Once every location has encrypted the dataset, location l removes the blinding by decrypting with z_l^b. As a result, each location l is in the possession of $h(h(\ \ldots\ h(h(D_l, y_1^m), y_2^m)\ \ldots, y_{|L|-1}^m), y_{|L|}^m)$.

Step 3. (Encrypted Analysis) Each location sends the resulting dataset to the central authority C, who performs data analysis over the set of datasets. The central authority returns $return_l$ datasets to each l, which specifies values of interest to location l.

Step 4. (Blinding for Decryption) Upon reception, l blinds $return_l$ with the recollection encryption key y_l^r.

Step 5. (Full Decryption: Return) As in Step 2, for each location l, the encrypted $return_l$ datasets are shuffled and sent to each location $x \in L$ (including l). Now, location x decrypts the dataset with z_x^m and sends the dataset back to l. Once every location has decrypted the dataset, location l removes the blinding by decrypting with z_l^r.

C.3 Security and Integrity

In this section we address security and integrity properties of the SCAMD protocol. First, we prove SCAMD prevents any set of independent locations from learning the plaintext information of encrypted data held by honest locations through collusion. Next, we prove the SCAMD protocol can be extended, via novel configurable subprotocols, to incorporate additional crucial properties, such as the preservation of data integrity and guarantees of protocol completion for all participants.

Theorem 10 (Collusion Resistant). *Given any location $l \in L$, there exists no set of locations $U \subseteq L - \{l\}$, which can collude to determine the plaintext values of D_l.*

PROOF. In general, there are three ways by which plaintext values of D_l can be revealed. The first case is when D_l is sent to a colluding location. Since plaintext values are only directly revealed when l chooses so, this case never occurs. The second case is when both a hashed version of D_l and the appropriate decryption key is sent to a colluding location. Again, this never arises.

The third case is more subtle. It occurs by exploiting the definition of quasi-commutative encryption. When a colluding location $u \in U$ is in the possession of a hashed version of D_l which has been hashed by the same set of keys as D_u, then it can learn certain features of the data in D_l. The collection of hashed versions of D_l occurs during two points of the protocol: encryption and decryption. The first opportunity is via the encryption process before the dataset is submitted to the central authority. During this process, every version of D_l provided to colluding locations has been hashed with the blinding key y_l^b. Thus, for any colluder $u \in U$ to compare his dataset, it is necessary that l hashes D_u with y_l^b. However this never occurs, since l only uses y_l^b for his own dataset and no one else's. The second opportunity is via the decryption process, when D_l is sent as the $return_l$ list. Yet, as during encryption, the colluding locations only receive versions of $return_l$ that have been hashed with the recollection key, y_l^r, which is only used for l's datasets. ■

Now that simple security with respect to semi-honest behavior has been established, we concentrate on problems with respect to malicious adversaries.

C.3.1 Extensions to Basic SCAMD

The basic protocol prevents locations from making direct inferences about any particular location's dataset. However, the protocol is leaky in security, since colluding, or independently malicious, locations can influence the central authority's analysis and subsequent response, in the form of the $return_l$ datasets. Moreover, a location can perform certain functions that will go undetected. In order to control data representation, the malicious location must be able to make changes to another location's data in a manner that is undetected. Specifically, a malicious location can influence the central authority through several means. First, a location can lie about which values exist in their data collection. While blatant dishonesty regarding one's own data is a concern, the SCAMD protocol does not address issues regarding semantics of the data. Note, lying about one's dataset exists in the analysis of plaintext data as well. One manner by which dishonesty can be discovered is to validate data with external knowledge regarding the underlying truth. Yet, when dealing with proprietary knowledge, the construction of such a litmus test will be dependent on the data in question and may be impossible. This problem is beyond the scope of the current section. However, we will return to this concern when a specific implementation of this protocol for trail anonymization is introduced.

Second, a malicious location can attempt to control how data is represented during the execution of the protocol. In order to do so, the malicious location can employ a different multiparty key pair for another location's dataset. When a malicious location is using more than one multiparty key pair we term this action a *key switch*. We subclassify the key switch attack into two distinct, though related, types. The first type, called a *full* key switch, occurs when the malicious location applies a particular multi-party key pair to every value of a particular location's dataset. The second type, called a *partial* key switch, occurs when the malicious location partitions a location dataset into x parts and each part is encrypted/decrypted with a different multiparty key.

Now we turn to extensions of the basic protocol for integrity checks that detect key switch behavior. As will be proven, several extensions to the basic protocol guarantee that no set of malicious locations (even one location) can tamper with the encrypted data they receive at any stage without being detected. We analyze malicious data corruption in the form of both full and partial key switching. Theorem 11 covers the case of full key switching, which will be detected by the central authority, whereas Theorem 12 covers the case of partial key switching, which is more easily detected by the data providing locations. Intuitively, the probability that partial key switching is detected by a single location has a naturally low bound under general conditions, whereas the probability that partial key switching is detected by the central authority requires non-negligible effort (in terms of bandwidth, for example) to be controlled below the same bound.

Checks Performed by the Central Authority

The first type of detection for key switching is performed by the central authority. It accounts for the situation of "full" key switching, which occurs when Sally encrypts all of Alice's dataset with the "bad" keys. The extension works as follows. The central authority sends the same "dummy" value v_C to every participating location. Prior to Step 1 of the SCAMD protocol, every location adds the value to their dataset. The subprotocol for dummy data transfers and encryptions is shown in figure C.2.

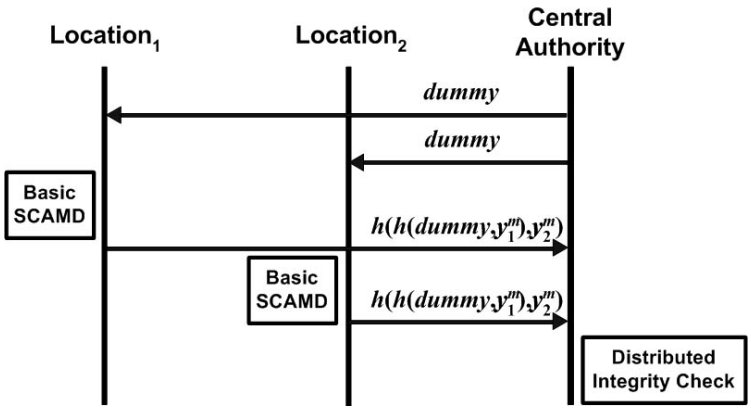


Figure C.2: Full key switch detection performed by central authority.

Let us call the switching location Sally and the owner of the switched dataset Alice. We consider the case when Sally uses two multi-key pairs. Instead of $\langle y^m_{Sally}, z^m_{Sally}\rangle$, Sally will use $bad = \langle y^{bad}_{Sally}, z^{bad}_{Sally}\rangle$ and $good = \langle y^{good}_{Sally}, z^{good}_{Sally}\rangle$, respectively. The latter key pair is used with every location's dataset, except for Alice for whom Sally uses the previous.

Theorem 11 (Full Key Switch Integrity). *The central authority is guaranteed to detect Sally's full key switch.*

PROOF. Assume Sally uses y^{bad}_{Sally} for all values in Alice's dataset. The only way that the full encrypted version of v_C, or any other value common to all datasets, will appear the same in all datasets is if Sally uses y^{bad}_{Sally} for every location's dataset including her own. Furthermore, if Sally only used y^{bad}_{Sally} during encryption, she must use z^{bad}_{Sally} with every location's dataset for decryption. However, if the latter is true, then Sally has only used one multi-party key pair and no key switching has occurred. ■

If the central authority does not detect a value that is the same at all locations this does not necessarily imply key switching. Rather, it may imply that a location failed to add v_C to its dataset. Regardless, when the latter is true then the central authority still detects that something has gone wrong during the execution of the protocol.

Checks Performed by the Single Locations

In addition to full key switching, Sally can perform "partial" key switching. In a partial key switch, Sally uses the bad multi-party key pair with a fraction of Alice's dataset and the $good$ multi-party key pair with the remainder. In order to prevent the partial key switch Alice introduces her own dummy data and uses an additional blinding key $\langle y_{Alice}^{check}, z_{Alice}^{check} \rangle$.

Prior to Step 1 of the protocol, Alice adds β dummy values to her dataset. After the final location has encrypted her data and prior to submitting the data to the central authority, Alice performs the following integrity check. After decrypting the data with the initial blinding key z_{Alice}^{b}, she re-encrypts her dataset with the new "check" key y_{Alice}^{check}, and then sends the dataset back to the other locations for decryption. If Sally is not performing a partial key switch, then she can correctly decrypt Alice's dataset without any problems. However, if Sally did perform a partial key switch then the probability she can correctly decrypt Alice's dataset is extremely small. In fact, Theorem 12 proves this happens with a naturally low probability, which can be further reduced by increasing β. Moreover, even if Alice believes that Sally randomly guessed the correct values to change, she can repeat the integrity check an arbitrary number α of times. For each repetition, Alice uses a new check key pair, again reducing at will the probability that Sally's cheating goes undetected. This process is depicted in Figure C.3.

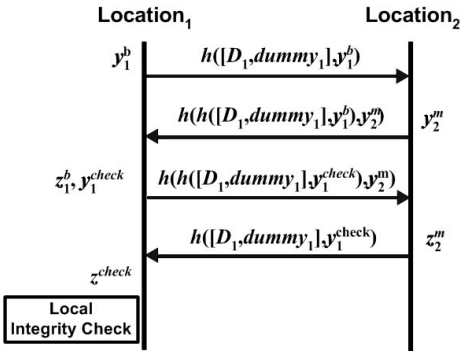


Figure C.3: Partial key switch detection as performed by location 1.

Theorem 12 (Partial Key Switch Integrity). *The probability Alice does not detect Sally's partial key switch is at most* $P_{\alpha,\beta} := 1 - (|D_{Alice}| + \beta)^{-\alpha}$.

PROOF. Assume Sally chooses f values in Alice's data to encrypt with y_{Sally}^{bad}. Now that Sally has performed her key switch, she must find those values in Alice's dataset during the decryption process. Yet, when Sally encrypted Alice's dataset, it was blinded by y_{Alice}^{b}, but now the data is blinded with y_{Alice}^{check}. As a result, unless Sally knows the new

blinding key pair, Sally must select the f records which need to be decrypted with z^{bad}_{Sally} at random. The probability of f successful guesses in our setting follows a hyper-geometric distribution with parameters $n - f$ (records encrypted by Sally with y^{good}_{Sally}) and f (records encrypted by Sally with y^{bad}_{Sally}) and can be written as:

$$Pr\begin{pmatrix} \textit{undetected} \\ \textit{key switch} \end{pmatrix} = \frac{\begin{pmatrix} f \\ f \end{pmatrix}\begin{pmatrix} n-f \\ n-f \end{pmatrix}}{\begin{pmatrix} n \\ f \end{pmatrix}} = \frac{f!(n-f)!}{n!}$$

This probability is maximized at $f = 1$ or $f = n - 1$. In Figure C.4 this is demonstrated for $n = 25$. As a result, Sally's best probability of remaining undetected is equal to $1/n$.

While f is chosen by Sally to maximize the probability of a partial key-switch being undetected, Alice can control $n = |D_{Alice}| + \beta$, the size of the dataset, to maximize the probability of detecting partial key switches. In particular, increasing the number of dummy records β, directly increases the probability of Sally's misbehavior being detected. However, Alice may wish to decrease β to save bandwidth during communication or total time necessary to complete decryption of the dataset. In this case she can still control the probability of detecting Sally's misbehavior by simply increasing the number of times that the decryption check is performed. Each decryption check is performed independently, since Alice uses a different blinding key for each check. Thus, the probability that Sally's partial key switch is detected by Alice is $P_{\alpha,\beta}$ = 1 - $(D_{Alice} + \beta)^{-\alpha}$ or less.

There are two possible scenarios. First, Alice chooses α (the number of checks to be performed) beforehand. In this scenario, the fact that the probability of detection is less than $P_{\alpha,\beta}$ is due to the fact that Sally's misbehavior can be detected before all α checks are performed. Second, Alice keeps on performing checks until the probability that a partial key switch was performed by Sally and was not detected falls below a certain threshold. In this latter scenario, the probability of detection is always equal to $P_{\alpha,\beta}$; in fact, Alice computes $P_{\alpha,\beta}$ after every check is performed and decides to stop when this probability is low enough. ■

In combination, the integrity checks performed by both Alice and the central party guarantee that the probability Sally performs a key switch is arbitrarily small. Both Alice and the central party are required to perform key switch detection. Appendix A provides proof that a) Alice can not perform full key switch detection as efficiently as the central authority and b) the central party can not perform partial key switch detection as Alice.

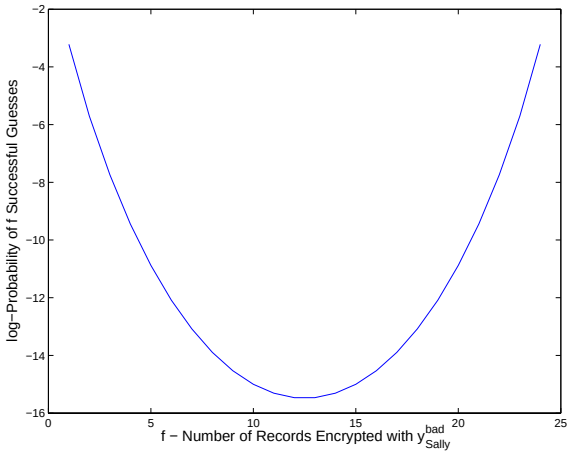


Figure C.4: Sample probabilities of exactly f successful guesses for the case of 25 records ($\log(1/25) \approx -3.219$).

Locking Out Malicious Locations

The implementation of the integrity checks performed in the previous section guarantee that no location can achieve a malicious action in the form of a key switch without being detected. In effect, the integrity of the cryptographic features are guaranteed, such that it is known that every location has both proper encryption and decryption multi-party key pair. However, the existence of such key pairs, does not imply that such key pairs will always be used. Neither the basic SCAMD protocol, nor the extensions for integrity discussed above, prevent Sally from achieving what we term a *grab-and-go*. Basically, Sally can recover the plaintext values of $return_{Sally}$ while simultaneously stopping Alice from recovering the plaintext values in $return_{Alice}$. This occurs when Alice decrypts Sally's dataset (*the grab*), but Sally refuses to decrypt Alice's dataset (*the go*).

In this section, we introduce a security feature that functions as a locking mechanism to prevent the grab-and-go. Basically, the central authority will guarantee that no location can recover their own values without acting honestly on behalf of all other locations datasets. Furthermore, the central authority will perform this validation without inspecting the plaintext values of any location's dataset.

The protection manifests in the form of a locking mechanism as follows. The central authority C creates a dummy dataset D_C at the very beginning. He then acts like an extra location performing Steps 1 and 2 of the SCAMD protocol, in other words, C sends D_C around through every location in L for full encryption, while all actual locations still perform the original Steps 1 and 2 with their own datasets. The lockout subprotocol is

depicted in figure C.5.

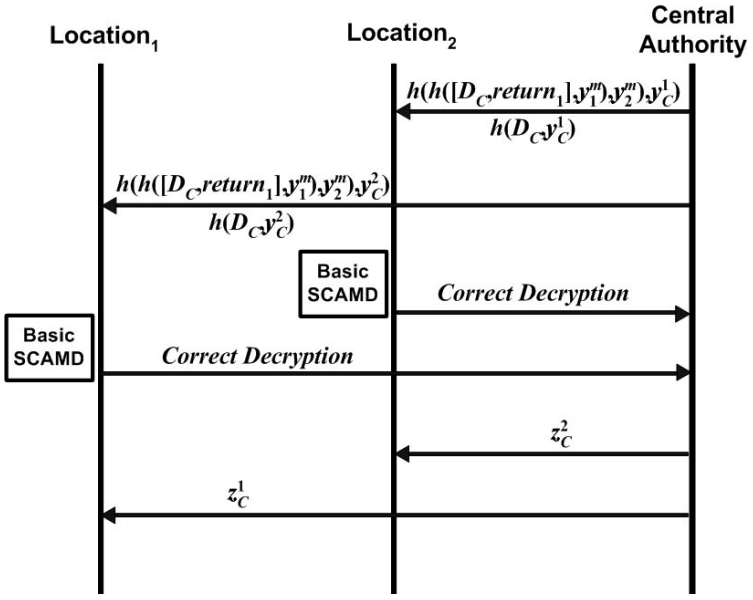


Figure C.5: Locking protocol.

For each location $l \in L$, C chooses a blinding key pair $\langle y_C^l, z_C^l \rangle$, blinds the mixture with y_C^l, and sends $h(\ldots h(h([D_C, return_l], y_C^b)y_1^m) \ldots, y_{|L|}^m)$ back to location l. In addition, C sends the blinded plaintext dummy dataset values, $h(D_C, y_C^l)$, to l. Next, each location performs full decryption as specified in SCAMD, and each location now possesses the central authority's blinded dataset $h([D_C, return_l], y_C^l)$. If the blinded dataset includes $h(D_C, y_C^l)$, then l tells C that decryption was performed honestly. Once all locations report honest decryptions, the central authority sends each $l \in L$ the appropriate blinding decryption key z_C^l. With the decryption key in hand, l decrypts the returned mixture and removes D_C.

Theorem 13 (Honest Decryption). *The probability Sally achieves a* grab-and-go *against Alice (A) is at most* $|return_A| \,/\, (|return_A| + |D_C|)$. PROOF. We assume that both Alice (A) and the central authority (C) have verified that no key switching behavior exists. Since Alice does not return the full decrypted dataset to C, the detection of an attempted grab-and-go is the sole responsibility of Alice. When Sally performs a grab-and-go, she needs to correctly decrypt D_C, while leaving $return_A$ in an encrypted state. Let f be the number of values Sally selects for a false decryption. Assuming Sally chooses to correctly decrypt a minimum of $|D_C|$ values (otherwise it is guaranteed her malicious behavior is

detected with probability 1), the probability of an undetected grab-and-go is equal to the probability all f values are selected from $return_A$:

$$\Pr\left(\begin{array}{c}\text{undetected}\\ \text{grab-and-go}\end{array}\right) = \frac{\left(\begin{array}{c}|D_C|\\ 0\end{array}\right)\left(\begin{array}{c}|return_A|\\ f\end{array}\right)}{\left(\begin{array}{c}|D_C|+|return_A|\\ f\end{array}\right)}$$

$$= \frac{|return_A|!(|D_C|+|return_A|-f)!}{(|return_A|-f)!(|D_C|+|return_A|)!}$$

The probability is monotonic and is maximized when $f = 1$. An example of this is shown in Figure C.6 for $|return_A| = 10$. When maximized, the probability Sally's action is undetected becomes $|return_A|$ / $(|return_A| + |D_C|)$. ■

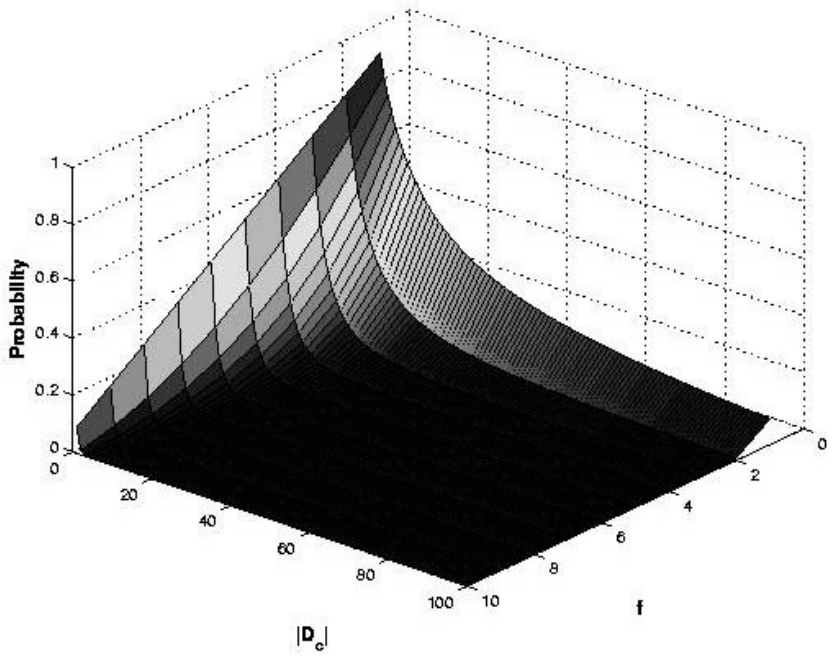


Figure C.6: Probability Sally's grab-and-go attack against Alice is successful; $|return_{Alice}|$ = 10. D_C is the size of the dummy dataset used by the central authority. f is the number of values Sally targeted.

In this locking mechanism, the central authority could initiate an extra integrity check for D_C in order to detect partial key switch attack by a malicious location. However,

this type of attack (even a full key switch against the whole D_C) can be detected in later comparisons between what is supposed to be $h([D_C, return_l], y_l^C)$, and $h(D_C, y_l^C)$, by each location $l \in L$. As a result, a standalone integrity check for D_C is not necessary.

C.3.2 Computational Overhead

Encryptions/Decryptions

For the following analyses, we assume each encryption or decryption operation on a dataset costs constant time (or can be bounded by it).

As described in section 3, in the basic protocol, each location $l \in L$ maintains three pairs of encryption and decryption keys. The first two pairs are only applied to l's own dataset, while the third pair is applied to every location's dataset.

The number of encryptions/decryptions that a specific location l needs to perform is:

$$\text{O}(\textit{encryption - basic}) = 2(1 + 1 + |L|) = \text{O}(|L|)$$

The total number of encryptions/decryptions performed by the whole system is $\text{O}(|L|^2)$. However, the encryption/de-cryption process for all locations is done in parallel, such that the total time necessary to complete this process remains $\text{O}(|L|)$.

In the protocol with integrity check, each location $l \in L$ is asked to perform α decryptions for integrity check initiated by each $l' \in L$ respectively, i.e., totally $\alpha|L|$ decryptions. l would also have α pair of keys applying to its own dataset for encryption and decryption. Now, the number of encryptions/decryptions that l needs to perform becomes:

$$\begin{aligned} \text{O}(\textit{encryption - with} & = 2(1 + 1 + |L|) + \alpha_l|L| + 2\alpha_l \\ \textit{integrity}) & = \text{O}(\alpha_l|L|) \end{aligned}$$

As a result, the total number of encryptions and decryptions for the whole system becomes $\text{O}(\overset{max}{l \in L} \alpha_l |L|^2)$. Yet, akin to the basic protocol, the integrity checks are performed in parallel, so the total time necessary for completion is $\text{O}(\overset{max}{l \in L} \alpha_l |L|)$.

When taking into account the locking mechanism to prevent the grab-and-go, the following additional encryptions and decryptions are needed due to dummy dataset D_C:

1. Each location needs to perform encryption one time to create the fully encrypted version of D_C;
2. The central authority needs to perform blinding encryptions for each location, a total of $|L|$ times;
3. Each location needs to apply a decryption key provided by the central authority to restore its returning dataset.

However, the encryptions/decryptions associated with recollection keys are spared at each location. Thus, the total time necessary is still $O(^{max}_{l\in L}\alpha_l|L|)$ because of parallelism of the protocol.

Bandwidth

Each location l provides a dataset of size s_l. To be encrypted by another location, the dataset needs to be sent from the originating location l to the destination location l', and sent back to l after encryption. A similar process is performed during the decryption phase. In addition, all datasets must be sent to the central authority; then a corresponding returning dataset of size r_l is sent back to the appropriate l.

In the basic protocol, each dataset proceeds through $|L|-1$ encryptions and decryptions by locations other than the originating one, and communicates with the central authority once, so total bandwidth required for a location l is:

$$\begin{aligned} \text{O(}\textit{bandwidth - basic}\text{)} &= 2s_l(|L|-1) + 2r_l(|L|-1) \\ &\quad +s_l + r_l \\ &= \text{O}((s_l + r_l)|L|), \end{aligned}$$

where s_l = $|D_l|$ and r_l = $|return_l|$. Because of parallelism, the total bandwidth required to finish the protocol is $O(^{max}_{l\in L}(s_l + r_l)|L|)$.

In the protocol with integrity check, s_l equals ($|D_l|$ + β_l + 1), where β_l is the number of dummy value added by l and "1" accounts for the dummy value provided by the central authority. Each location needs an extra α_l rounds of decryption from other locations, so the total bandwidth becomes:

$$\begin{aligned} \text{O(}\textit{bandwidth - with integrity}\text{)} &= 2s_l(|L|-1) + 2r_l(|L|-1) \\ &\quad +s_l + r_l + 2\alpha_l s_l(|L|-1) \end{aligned} \tag{C.3}$$

where s_l = $|D_l|$ + β_l + 1 and r_l = $|return_l|$. Thus, an upper bound on the bandwidth is $O(^{max}_{l\in L}\alpha_l(s_l + r_l)|L|)$.

When we consider the additional locking mechanism, r_l equals ($|D_C|$ + $|return_l|$) and the central authority sends an extra version of blinded D_C to l. Assuming the full encryption of D_C is performed in parallel, the total bandwidth required for a specific location l is:

$$\begin{aligned} \text{O(}\textit{bandwidth - with integrity \& locking}\text{)} &= 2s_l(|L|-1) + 2r_l(|L|-1) \\ &\quad +s_l + r_l + |D_C| \\ &\quad +2\alpha_l s_l(|L|-1), \end{aligned} \tag{C.4}$$

where s_l = $|D_l|$ + β_l + 1 and r_l = $|D_C|$ + $|return_l|$. Similarly, an upper bound is $O(^{max}_{l\in L}\alpha_l(s_l + r_l)|L|)$.

Integrity Check vs. Bandwidth

According to Theorem 12, the lower bound of the probability of location l detecting a partial key switch attack is $1-(|D_l|+\beta_l+1)^{-\alpha}$ (1 dummy value provided by the central authority considered). Assume each location l requires the protocol to satisfy a certain confidence requirement λ_l on this lower bound. In order to achieve the lowest bandwidth cost, we are actually solving a special non-linear optimization problem. Specifically, we need to solve for α_l's and β_l's as integers, when minimizing equation (C.3) and satisfying constraints on confidence in honesty for each $l \in L$:

$$1-(|D_l|+\beta_l+1)^{-\alpha_l} > \lambda_l,$$

as well as bandwidth and computational constraints.

Similarly, if the locking mechanism is also required, we need to simultaneously solve for α_l's and β_l's, as well as $|D_C|$ as integers, while minimizing equation (C.4) and satisfying additional constraints according to Theorem 13.

C.4 Protocol Application

The distributed operations the SCAMD protocol enable us to carry out in a secure manner are very different in nature. This has an impact on the format of the data needed for centralized data analysis, for example, a certain application may require plaintext data to be broadcasted by the *semi-trusted* third-party to the participating locations, whereas another application may require an analysis on the union of the encrypted data sets and only few sensitive records, still encrypted, may need be returned to the single locations. Thus, different scenarios require slightly different definitions of *semi-trusted*, which we now discuss. In the original definition, semi-trusted requires the central authority is never permitted to know the plaintext values it analyzes. However, if plaintext values need to be broadcast, the definition of semi-trusted can be relaxed. In the relaxed definition, the third-party is not allowed to know which participating location submitted which specific values, though it is permitted to see the plaintext values of the records it is going to broadcast.

In addition, other aspects must change to fit the SCAMD protocol to different scenarios. For example, in order to allow the central authority to have broadcasting powers, we would modify the *full decryption* section of the protocol. Instead of sending a $return_l$ dataset to each location l, the central authority only needs to send a single dataset, which we refer to as $return_C$, around for decryption. Briefly, the central authority performs Steps 4 and 5 of SCAMD with it's own recollection key pair $\langle y_C^r, z_C^r \rangle$. The central authority blinds and sends $h(return_C, y_C^r)$ to each of the locations, for full decryption. Once

fully decrypted, the central authority removes its blinding and broadcasts the plaintext data to all participating locations.

C.4.1 Configurability of the SCAMD Protocol

The SCAMD protocol provides security with provable probabilistic guarantees in carrying out distributed data mining tasks. The main ideas that enable functionality in a diversity of applications include: (a) the incorporation of dummy data, which allows for control over the detection of integrity tampering, (b) data shuffling, along with (c) forcing a malicious location to compete against its own behavior by decrypting its own encryption, and (d) a locking mechanism, which prevents malicious locations from learning information the protocol does not prescribe. Given a variety of practical tasks, we believe it is possible to combine these basic security enabling addenda to fit the SCAMD protocol to the specific scenario at hand. This is achieved without any assumptions about semi-honest behavior on the part of the participating locations. Hence, we say that SCAMD protocol is *configurable* to fit an ample spectrum of distributed data mining computations.

In this light, removing certain parts of the protocol harms neither the functionality, nor the security of our protocol. For example, when the definition of semi-trusted is relaxed to provide the central authority with plaintext broadcasting capabilities the locking mechanism is not needed. It can be validated that removal of the locking mechanism security component does not affect the overall security of the protocol. Rather, it is not necessary to carry out the specific distributed computation task.

The SCAMD protocol is the first step towards a formal modular architecture for secure, distributed data mining, with provable guarantees in environments where semi-honest behavior on the part of participating locations can not safely be made. The next step in the development of a modular protocol is to feature a description of distributed data mining tasks along relevant dimensions, and map them into a sequence of primitive sub-tasks, which the various modules of our protocol can address in a secure way. A major portion of the single modules will address primitive sub-tasks, whereas others will provide provable security guarantees that the modules will produce the expected results, even in the presence of malicious participating locations.

C.4.2 Example: Distributed Data Union

To illustrate SCAMD's flexibility and security, we map a distributed association rule learning algorithm [71, 72] into the SCAMD architecture. The previously defined algorithm is based on semi-honest assumptions, which we refer to as *SemiSecureUnion*, or *SSU*, is presented to find the secure union of distributed data without revealing which itemsets be-

long to a given location. The underlying mechanism *SSU* is as follows. Each location l sends D_l to all locations in L for encryption, such that another location $x \neq l$, receives the full encrypted D_l. Once completed, each location holds another location's full encrypted dataset. Next, two locations collect and union half the full encrypted datasets. Then, one location sends its union to the second location, who again performs a unions. Finally, the locations, send the dataset around for full decryption, such that a third location, not one of the previous two, broadcasts the plaintext union.

The *SSU* protocol is susceptible to collusion, which suggests it may not be practical in real-world settings - even in a semi-honest environment. Consider, for example, a situation where a regional association of large-size retail stores wants to provide some aggregate statistics about the market, in the form of large itemsets. Collusion is more likely to occur in such a network when several sites belong to the same umbrella company or to the same chain (*e.g.* Wal-Mart comprises 70% of the participating sites).

Algorithm 17 SCAMD-SemiSecureUnion: Secure union of itemsets with SCAMD

Phase 0*: Local plaintext association rule learning*
for each $l \in L$ **do**
Generate set of local association rules $rule_l$ as defined in distributed association rule mining algorithm (FDM) defined in [23]
end for

Phase 1*: Encryption by all sites*
Each $l \in L$ executes SCAMD encryption phase (Steps 1-2) //*Each location possesses full encrypted* $rule_l$*, denoted* $frule_l$

Phase 2*: Central itemset merge (Step 3 of SCAMD)*
All $l \in L$ send $frule_l$ to C
C creates $RuleSet = \bigcup_{l \in L} frule_l$

Phase 3*: Central Broadcast*
C sets $return_C = RuleSet$
C executes SCAMD decryption phase (Steps 4-5)
C broadcasts full decrypted dataset $return_C$

We present the *SCAMD-SemiSecureUnion* (*SCAMD-SSU*), pseudocode provided in Algorithm 1, which maps *SSU* into the SCAMD architecture. *SCAMD-SSU* achieves the same goal as *SSU* and provides in addition that it 1) prevents each location from seeing other locations' fully encrypted data, 2) prevents two participating locations from perform-

ing the union of all fully encrypted data sets, and 3) prescribes for the data to be broadcasted from a third party, not present in *SSU*, that has no data at stake. In the semi-honest environment, the main concern is collusion. Since *SCAMD-SSU* is an implementation of the SCAMD protocol, collusion among participating locations is no longer a concern (as shown in Theorem 10). In addition, *SCAMD-SSU* solves problems that *SSU* is susceptible to once implemented with malicious locations.

It is interesting to note that while *SSU* is susceptible to collusion, it is partially protective against a key switch attack. Again, consider the situation where Sally performs a key switch against Alice. When all of Alice's switched values are in any other location's dataset, then the key switch has no real influence on the union. In the alternative situation, if Alice's switched values are not within another location's submission, Alice can claim the integrity of her data was tampered with. However, this is why *SSU* provides only partial protection. Once the plaintext union is broadcast, Alice can not add her dataset to the union without every other location learning the unique values of her dataset. This problem is solved by *SCAMD-SSU* once the integrity checking is integrated.

C.4.3 Security Concerns

In this section, we briefly discuss several concerns and challenges regarding the current design of the SCAMD protocol. The first concern corresponds to the difficulty in detecting which location is malicious. The second concern addresses assumptions regarding the central authority.

Traitors Lost in the Crowd

Theorems 2-4 prove it is possible to control the probability of malicious actions going undetected. Yet despite these controls, we acknowledge it is not possible to detect the source of the irregularity. This is mainly due to the usage of an accumulator based on quasi-commutative encryption. It is possible that more complex schemas may be able to detect both mishaps, as well as their sources, however, this is beyond the scope of the current research. We expect to look into such extensions in future research.

The Semi-Trusted Assumption

With respect to the central authority, many of the protections afforded by the SCAMD protocol are dependent on the assumption that the central authority is honest; it neither collaborates with participating locations nor answers locations dishonestly. First, consider a central authority that is merely semi-honest. In this case, collusion resistance as proven in Theorem 10, no longer holds true. Specifically, locations that collude with the central

authority will be able to compare their full encrypted dataset against the full encrypted datasets of every non-colluding location. When an encrypted value v is found to be equivalent between the colluding and non-colluding datasets, then the colluder can bound the set of plaintext values for v. When multiple locations are colluding, the possibility exists that the colluder can learn the exact plaintext value for v. This occurs when the number the values in common between a set of colluders datasets is equivalent to the number of values in the non-colluders dataset.

Second, and of more grave concern, we consider a central authority that is actually malicious. When such an event occurs, Theorem 13 can be nullified in such a way that non-colluding locations fail to detect malicious behavior. Basically, the central authority can supply corrupt locations with its blinding decryption key regardless of if corrupt locations used the correct multiparty decryption keys with other locations. Moreover, the central party can violate Theorem 10 in such a manner that a location $x \in L$ colluding with the central authority can learn all plaintext values of a dataset for any arbitrary location $l \in L$ without being detected. This would occur if the central authority was to provide a colluder with $return_x = D_l$ and Steps 4 and 5 of SCAMD proceed as specified.

Recent research in theoretical multi-computation proves that third parties can be removed from the protocol while maintaining the same level of security. [20] However, the design of these systems are most often inefficient to the point of being intractable for practical application. Thus, in future research we expect to continue investigating models that incorporate third parties, but reduce the requirement of the semi-trusted model.

C.5 Conclusions

This chapter introduced a novel protocol, termed secure centralized analysis of multi-party data, or SCAMD. The protocol allows for multiple locations to conduct analyses over distributed data in a secure manner in the face of malicious behavior. The protocol supports location-specific responses, such that each location can learn information, of which other locations can not ascertain the contents. Moreover, parallelism of the protocol allows for execution in linear time and bandwidth. The protocol is amenable to different types of encrypted data analysis, and in this chapter we demonstrated how set unioning can be made more secure. In the dissertation, we demonstrate how the protocol can be used for trail unlinking, and in the future we expect to extend the protocol to a range of distributed computations in malicious environments.

Appendix D

k-Unlinkability Bounds

In Chapter 8, we proved that the secure deduplication algorithms satisfy various forms of k-unlinkability. However, in the encrypted space, the semi-trusted third party (*sTTP*) does not know which encrypted de-identified data sample corresponds to which identified data sample. As a consequence, the *Contributor-Clean* method always uses the minimum number of elements in γ_i that are available to protect de-identified elements disclosed in $\omega_i{}'$. In Lemma 20, we show the underestimation of the true number of elements has a bound dependent on regions ❷ and ❹.

Lemma 20 (*sTTP* Bounds Reserved By Minimum). *Given $h(\Gamma)$, $h(\Omega)$ and $h(\Omega')$, as output by Greedy-Dedup-Secure, the Contributor-Clean method underestimates the number of elements in γ_i that can protect elements in $h(\omega_i)$ with respect to c_j by* $\min(|h(\omega_j)|-|❸|, |\gamma_i \cap \gamma_j|-|h(\omega_i)\cap h(\omega_j)|, |\gamma_i|-|h(\omega_i)|, |\gamma_j \backslash \gamma_i|, |\gamma_j|-h(\omega_j)|, (|\gamma_i|+ |\gamma_j| - |\gamma_i \cap \gamma_j|) - (|h(\omega_i)| + |h(\omega_j)| - |h(\omega_i) \cap h(\omega_j)|))$

.

PROOF. The knowledge that c_j can use to assist in re-identifying elements from $h(\omega_i)$ resides in regions ❸ and ❹. The *Contributor-Clean* method learns the exact size of ❸, but uses an upper bound for the size of region ❹, which we represent as $❹_{max}$. The $❹_{max}$ region is the worst case scenario for the protection of c_i's de-identified data with respect to private knowledge held by c_j. In this scenario, the number of elements in $\gamma_i \cap \gamma_j$ that contribute to protection of $h(\omega_i)$ is $|\gamma_i \cap \gamma_j|$ - |❸| - $|❹_{max}|$.

In contrast, the best case scenario for the protection of c_i's data with respect to c_j occurs when the size of region ❹ is minimized, which we represent as $❹_{min}$. In this scenario, the size of region $❹_{min}$ is equivalent to that of region $❷_{min}$ from the perspective of c_j. By switching perspectives, we exchange the databases of c_i and

c_j. We find that the size of $❹_{min}$ is:

$$❹_{min} = \max\left(0, |h(\omega_j) - ❸| - |\gamma_j \backslash \gamma_i|\right) \tag{D.1}$$

The largest number of elements by which the *Contributor-Clean* method will underestimate protection is the worst case scenario minus the best case scenario, which is

$$|\gamma_i \cap \gamma_j| - |❸| - |❹_{min}| - (|\gamma_i \cap \gamma_j| - |❸| - |❹_{max}|)$$

By cancelling several terms, this quantity reduces to:

$$|❹_{max}| - |❹_{min}|$$

$$= (\min(|h(\omega_j)|, |\gamma_i \cap \gamma_j| - |❷_{min}|) - |❸|) - \max(0, |h(\omega_j) - ❸| - |\gamma_i \backslash \gamma_i|)$$

$$= \min(|h(\omega_j) - ❸|, |\gamma_i \cap \gamma_j| - |❷_{min}| - |❸|) - \max(0, |h(\omega_j) - ❸| - |\gamma_i \backslash \gamma_i|).$$

Let us relabel this equation as $\min(A, B)$ - $\max(C, D)$. By distributing and working through each of these terms, we find that there are six mutually exclusive and irreducible terms that comprise the space of possible values, which we will call (I) through (VI). Considering, each difference term:

$$\begin{aligned} A - C &= |h(\omega_j)| - |❸| && \textbf{(I)}, \\ B - C &= |\gamma_i \cap \gamma_j| - |❷_{min}| - |❸|, \end{aligned}$$

which due to uncertainty in the size of region $❷_{min}$ is expanded and becomes:

$$B - C = |\gamma_i \cap \gamma_j| - \max(0, |h(\omega_i) - ❸| - |❶|) - |❸|$$

$$= \min(|\gamma_i \cap \gamma_j|, |\gamma_i \cap \gamma_j| - |h(\omega_i) - ❸| + |❶|) - |❸|$$

Now, since the size of region ❸ must be ≥ 0, we can rule out the first term, and the formula further reduces to

$$= \min(|\gamma_i \cap \gamma_j| - |❸|, |\gamma_i \cap \gamma_j| - |h(\omega_i) + ❸| + |❶| - |❸|)$$

$$= \min(|\gamma_i \cap \gamma_j| - |❸|, |\gamma_i \cap \gamma_j| - |h(\omega_i)| + |❸| + |\gamma_i| - |\gamma_i \cap \gamma_j| - |❸|)$$

and ultimately reduces to the two terms:

$$\begin{aligned} &|\gamma_i \cap \gamma_j| - |h(\omega_i) \cap h(\omega_j)| && \textbf{(II)}, \\ &|\gamma_i| - |h(\omega_i)| && \textbf{(III)}, \end{aligned}$$

Continuing with the distributed terms, we find:

$$A - D = |\gamma_j \backslash \gamma_i| \qquad \textbf{(IV)},$$
$$B - D = |\gamma_i| - |❷_{min}| - |h(\omega_j)|,$$

which for the same reason as $B - C$ is expanded to:

$$B - D = |\gamma_i \cap \gamma_j| - |❷_{min}| - (|h(\omega_j)| - |\gamma_j \backslash \gamma_i|)$$

$$= |\gamma_i \cap \gamma_j| - \max(0, |h(\omega_j) - h(\omega_i) \cap h(\omega_j)| - |\gamma_i \backslash \gamma_j|) - (|h(\omega_j)| - |\gamma_i \backslash \gamma_j|)$$

$$= |\gamma_j| - |h(\omega_j)| - \max(0, |h(\omega_i) \backslash h(\omega_j)| - |\gamma_i \backslash \gamma_j|)$$

$$= \min(|\gamma_j| - |h(\omega_j)|, |\gamma_j| - |h(\omega_j)| - (|h(\omega_i) - h(\omega_i) \cap h(\omega_j)| - |\gamma_i \backslash \gamma_j|))$$

which reduces to:

$$|\gamma_j| - |h(\omega_j)| \qquad \textbf{(V)},$$
$$(|\gamma_i| + |\gamma_j| - |\gamma_i \cap \gamma_j|) - (|h(\omega_i)| + |h(\omega_j)| - |h(\omega_i) \cap h(\omega_j)|) \qquad \textbf{(VI)}.$$

Taking the minimum of terms (I) through (VI) provides the largest number of elements the *sTTP* can underestimate are available for protecting disclosed data. □

As noted earlier, in an unreserved environment, the *sTTP* will not underestimate the number of elements available for protection. This is because *Contributor-Clean* provides the exact number of elements available to protect elements in $\omega_i{}'$. This is stated as Lemma 21 and can be verified visually in Figure 8.10.

Lemma 21 (*sTTP* Does Not Underestimate in Unreserved). *Given an unreserved environment with Ω', as output by* Greedy-Dedup-Secure*, the Contributor-Clean method will not underestimate the protection capability of γ_i for elements in $h(\omega_i)$ with respect to c_j.*

PROOF. From Lemma 20, we know that the bounds for Contributor-Clean are due to the variable size of region ❹. Here, we show that when the environment is unreserved $❹_{min} = ❹_{max}$. First, consider $❹_{max}$, the size of which we know from Theorem 5 is $\min(|h(\omega_j)|, |\gamma_i \cap \gamma_j| - |❷_{min}| - |❸|)$. Now, region $❷_{min}$ has size $\max(0, |h(\omega_i) - ❸| - |\gamma_i \backslash \gamma_j|)$. By expanding and rearranging term, we find:

$$\begin{aligned} ❷_{min} &= \max(0, (|h(\omega_i)| - |h(\omega_i) \cap h(\omega_j)|) - (|\gamma_i| - |\gamma_i \cap \gamma_j|)) \\ &= \max(0, (|h(\omega_i)| - |\gamma_i|) + (|\gamma_i \cap \gamma_j| - |h(\omega_i) \cap h(\omega_j)|)) \end{aligned}$$

Since the environment is unreserved, we know that $|\gamma_i| = |h(\omega_i)|$ and that $|\gamma_i \cap \gamma_j| = |h(\omega_i) \cap h(\omega_j)|$. Using these facts, we can perform substitution and $|❷_{min}|$ reduces to $\max(0, 0)$, or 0. This means region ❷ is nonexistent in an unreserved environment. Thus, the size of $❹_{max}$ reduces to $\min(|h(\omega_i)|, |\gamma_i \cap \gamma_j|) - |❸|$.

Also, due to the unreserved environment, we know $|\gamma_j| = |h(\omega_j)|$. So, we can perform substitution and reduce $❹_{max}$ to

$$\min(|\gamma_j|, |\gamma_i \cap \gamma_j|) - |\gamma_i \cap \gamma_j|.$$

It must be true that $|\gamma_j| - |\gamma_i \cap \gamma_j| \geq 0$, so $❹_{max}$ reduces to $\min(0, 0)$, or 0.

If $0 \leq |❹_{min}| \leq |❹_{max}| \leq 0$, then $|❹_{min}| = 0$. Therefore, in an unreserved environment, $|❹_{max}| - |❹_{min}| = 0$. □

There are several useful findings with respect to Lemma 21, which Figure 8.10 helps demonstrates from a visual perspective. In an unreserved environment, the only regions that exist are ❶, ❸, and ❻. As a result, it is evident that $|\gamma_i| = |h(\omega_i)|$, $|\gamma_j| = |h(\omega_j)|$, and $|\gamma_i \cap \gamma_j| = |h(\omega_i) \cap h(\omega_j)|$ As a consequence, the *sTTP* can certify when contributor k-unlinkability is satisfied by studying only the published identified databases.

Bibliography

[1] G. Aggarwal, T. Feder, K. Kenthapadi, R. Motwani, R. Panigrahy, D. Thomas, and A. Zhu. Approximation algorithms for k-anonymity. *Journal of Privacy Technology*, page 20051120001, 2005.

[2] D. Agrawal and C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the ACM Symposium on Principles of Database Systems*, pages 247–255, 2001.

[3] R. Agrawal and R. Srikant. Privacy preserving data mining. In *Proceedings of the ACM Symposium on Principles of Database Systems*, pages 439–450, 2000.

[4] A. Al-Lawati, D. Lee, and P. McDaniel. Blocking aware private record linkage. In *Proceedings of the ACM SIGMOD Workshop on Information Quality in Information Systems*, Baltimore, MD, 2005.

[5] R. Altman. Bioinformatics in support of molecular medicine. In *Proceedings of the American Medical Informatics Association Annual Symposium*, pages 53–61, Miami Beach, FL, 1998.

[6] R. Altman and T. Klein. Challenges for biomedical informatics and pharmacogenomics. *Annual Review of Pharmacology and Toxicology*, 42:113–133, 2002.

[7] M. Atallah, F. Kerschbaum, and W. Du. Secure and private sequence computations. In *Proceedings of the $2^n d$ ACM Workshop on Privacy in the Electronic Society*, pages 39–44, 2003.

[8] N. Baric. and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In I. Ingemarsson, editor, *Lecture Notes in Computer Science 1233: Advances in Cryptology: Proceedings of EUROCRYPT*, pages 480–494. Springer-Verlag, New York, NY, 1997.

[9] R. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *Proceedings of the* 21^{st} *IEEE International Conference on Data Engineering*, pages 217–228, Tokyo, Japan, 2005.

[10] J. Benaloh and M. deMare. One-way accumulators: a decentralized alternative to digital signatures (extended abstract). In T. Hellsuth, editor, *Lecture Notes in Computer Science 765: Advances in Cryptology (EUROCRYPT '93)*, pages 274–285. Springer-Verlag, New York, NY, 1994.

[11] S. Bender, R. Brand, and J. Bacher. Re-identifying register data by survey data: an empirical study. *Statistical Journal of the United Nations ECE*, 18:373–381, 2001.

[12] M. Bilenko and R. Mooney. On evaluation and training-set construction for duplicate detection. In *Proceedings of the ACM SIGKDD Workshop on Data Cleaning, Record Linkage, and Object Consolidation*, Washington, DC, 2003.

[13] T. Blakely and C. Salmond. Probabilistic record linkage and a method to calculate the positive predictive value. *International Journal of Epidemiology*, 31(6):1246–52, 2002.

[14] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: Evidence and implications. In *Proceedings of IEEE INFOCOM*, pages 126–134, New York, NY, 1999.

[15] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the* 7^{th} *World Wide Web Conference*, pages 107–117, Brisbane, Australia, 1998.

[16] R. Bucklin and C. Sismeiro. A model of web site browsing behavior estimated on clickstream data. *Journal of Marketing Research*, 40:249–267, 2003.

[17] L. Burnett, K. Barlow-Stewart, A. Pros, and H. Aizenberg. The "genetrustee": a universal identification system that ensures privacy and confidentiality for human genetic databases. *Journal of Law and Medicine*, 10(4):506–513, 2003.

[18] L. Caenazzo, E. Ponzano, N. Greggio, and P. Cortivo. Prenatal sexing and sex determination in infants with ambiguous genitalia by polymerase chain reaction. *Genetic Testing*, 1(4):289–291, 1997–1998.

[19] R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multi-party computation. In *Proceedings of the* 28^{th} *ACM Symposium on Theory of Computing*, pages 639–648, Philadelphia, PA, 1996.

[20] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *Proceedings of the* 34^{th} *Symposium on Theory of Computing*, pages 494–503, New York, NY, 2002.

[21] J. Canny. Collaborative filtering with privacy. In *Proceedings of the IEEE Conference on Security and Privacy*, pages 238–245, Oakland, CA, 2002.

[22] D. Chaum. Blind signatures for untraceable payments. In *Advances in Cryptography, Crypto 1982*, pages 199–203. Plenum Press, New York, NY, 1983.

[23] D. Cheung, J. Han, V. Ng, W. Fu, and Y. Fu. A fast distributed algorithm for mining association rules. In *Proceedings of the International Conference on Parallel and Distributed Information Systems*, pages 31–42, Miami Beach, FL, 1996.

[24] T. Churches. A proposed architecture and method of operation for improving the protection of privacy and confidentiality in disease registers. *BMC Medical Research Methodology*, 3(1):1, 2003.

[25] T. Churches and P. Christen. Some methods for blindfolded record linkage. *BMC Medical Informatics and Decision Making*, 4(1):9, 2004.

[26] W. Cohen, P. Ravikumar, and S. Fienberg. A comparison of string distance metrics for name-matching tasks. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, 2003.

[27] W. Cohen, P. Ravikumar, and S. Fienberg. A secure protocol for computing string distance metrics. In *Proceedings of the* 3^{rd} *IEEE Workshop on Privacy and Security Aspects of Data Mining*, pages 40–46, Brighton, England, 2004.

[28] L. Cortesi, D. Turchetti, C. Bertoni, R. Bellei, L. Mangone, M. Vinceti, M. Federico, V. Silingardi, and S. Ferrari. Comparison between genotype and phenotype identifies a high-risk population carrying brca1 mutations. *Genes Chromosomes and Cancer*, 27(2):130–135, 2000.

[29] L. Cranor. *Web privacy with p3p*. O'Reilly & Associates, Sebastapol, CA, 2002.

[30] G. de Moor, B. Claerhout, and F. de Meyer. Privacy enhancing technologies: the key to secure communication and management of clinical and genomic data. *Methods of Information in Medicine*, 42:148–153, 2003.

[31] A. de Waal and L. Willenborg. A view on statistical disclosure control for microdata. *Survey Methodology*, 22:95–103, 1996.

[32] Department of Health and Human Services. 45 cfr (code of federal regulations), parts 160 - 164. standards for privacy of individually identifiable health information, final rule. *Federal Register*, 67(157):53182–53273, 2002.

[33] Dept of Health and Human Services. Standards for privacy of individually identifiable health information, final rule. *Federal Register*, 45 cfr, 160-164. aug 12, 2002.

[34] C. Diaz, J. Claessens, S. Seys, and B. Preneel. Information theory and anonymity. In *Proceedings of the* 23^{rd} *Symposium on Information Theory in the Benelux*, 2002.

[35] C. Diaz, J. Claessens, S. Seys, and B. Preneel. Towards measuring anonymity. In *Proceedings of the* 2^{nd} *Privacy Enhancing Technologies Workshop, Lecture Notes in Computer Science vol. 2482*, pages 54–68, Berlin, Germany, 2002. Springer-Verlag.

[36] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[37] J. Domingo-Ferrer and V. Torra. Disclosure control methods and information loss for microdata. In P. Doyle, J. Lane, J. Theeuwes, and L. Zayatz, editors, *Confidentiality, Disclosure and Data Access: Theory and Practical Applications for Statistical Agencies*, pages 93–112. North-Holland, Amsterdam, 2002.

[38] J. Domingo-Ferrer and V. Torra. Ordinal, continuous, and heterogeneous k-anonymity through microaggregation. *Data Mining and Knowledge Discovery*, pages 195–212, 2005.

[39] W. Du and M. Atallah. Protocols for secure remote database access with approximate matching. In *Proceedings of the* 1^{st} *ACM Workshop on Security and Privacy in E-Commerce*, Athens, Greece, 2000.

[40] M. Dugas, C. Schoch, S. Schnittger, W. Kern, T. Haferlach, D. Messerer, and U. Überla. Impact of integrating clinical and genetic information. *In Silico Biology*, 2:383–391, 2002.

[41] G. Duncan and S. Fienberg. A markov perturbation method for tabular data, turning administrative systems into information systems. *IRS Methodology Report Series 5*, pages 223–231, 1997.

[42] J. Edwards and P. Tarczy-Hornoch. Implementation of a classification hierarchy for the genetests/geneclinics genetic testing databases. In *Proceedings of the American*

Medical Informatics Association Annual Symposium, pages 235–239, San Antonio, TX, 2002.

[43] M. Elfeky, V. Verykios, and A. Elmagarmid. Tailor: a record linkage toolbox. In *Proceedings of the* 18^{th} *International Conference on Data Engineering*, San Jose, California, 2002.

[44] E. Faldella and M. Prandini. A novel approach to on-line status authentication of public-key certificates. In *Proceedings of the* 16^{th} *Annual Computer Security Applications Conference*, New Orleans, LA, 2000.

[45] J. Feigenbaum, M. Freedman, T. Sander, , and A. Shostack. Economic barriers to the deployment of existing privacy technology. In *Proceedings of the Workshop on Economics and Information Security*, Berkely, CA, 2002.

[46] I. Fellegi. On the question of statistical confidentiality. *Journal of the American Statistical Association*, 67:7–18, 1972.

[47] I. Fellegi and A. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(1183–1210), 1969.

[48] T. Ferris, G. Garrison, and H. Lowe. A proposed key escrow system for secure patient information disclosure in biomedical research databases. In *Proceedings of the American Medical Informatics Association Annual Symposium*, pages 245–249, San Antonio, TX, 2002.

[49] N. C. for Biotechnology Information. *Genes and Disease*. National Library of Medicine, Bethesda, MD, 2005.

[50] P. Franks and C. Clancy. Referrals of adult patients from primary care: demographic disparities and their relationships to hmo insurance. *Journal of Family Practice*, 45:47–53, 1997.

[51] D. Gaudet, S. Arsnault, C. Belanger, T. Hudson, P. Perron, M. Bernard, and P. Hamet. Procedure to protect confidentiality of familial data in community genetics and genomics research. *Clinical Genetics*, 55:259–264, 1999.

[52] T. Gelehrter, D. Ginsburg, and F. Collins. *Principles of Medical Genetics,* 2^{nd} *Edition*. Lippincott Williams & Wilkins, Philadelpha, PA, 1998.

[53] W. Gesler and M. Meade. Locational and population factors in health care-seeking behavior in savannah, georgia. *Health Services Research*, 23(3):443–62, 1988.

[54] M. Gold. Beyond coverage and supply: measuring access to healthcare in todays market. *Health Services Research*, 33(3):625–652, 1998.

[55] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game - or - a completeness theorem for protocols with honest majority. In *Proceedings of the* 19^{th} *Symposium on Theory of Computing*, pages 218–229, New York, NY, 1987.

[56] D. Goldschlag, M. Reed, and P. Syverson. Onion routing for anonymous and private internet connections. *Communications of the ACM*, 42(2):39–41, 1999.

[57] S. Grannis, M. Overhage, and C. McDonald. Analysis of identifier performance using a deterministic linkage algorithm. In *Proceedings of the American Medical Informatics Association Annual Symposium*, pages 305–309, San Antonio, TX, 2002.

[58] V. Griffith and M. Jakobsson. Messin' with texas: deriving mother's maiden name using public records. In *Proceedings of the Applied Cryptography and Network Security Conference*, New York, NY, 2005.

[59] J. Gulcher, K. Kristjansson, H. Gudbjartsson, and K. Stefansson. Protection of privacy by third-party encryption in genetic research. *European Journal of Human Genetics*, 8:739–742, 2000.

[60] D. Gusfield and R. W. Irving. *The stable marriage problem : structure and algorithms*. MIT Press, Cambridge, Massachusetts, 1989.

[61] M. Hall and S. Rich. Patients' fear of genetic discrimination by health insurers: the impact of legal protections. *Genetics in Medicine*, 2:214–221, 2000.

[62] A. Hamosh, A. Scott, J. Amberger, C. Bocchini, and V. McKusick. Online mendelian inheritance in man (omim), a knowledgebase of human genes and genetic disorders. *Human Mutation*, 15:57–61, 2000.

[63] K. Hara, K. Ohe, T. Kadowaki, N. Kato, Y. Imai, K. Tokunaga, R. Nagai, and M. Omata. Establishment of a method of anonymization of dna samples in genetic research. *Journal of Human Genetics*, 48:327–330, 2003.

[64] P. Hess and D. Cooper. Impact of pharmacogenomics on the clinical laboratory. *Molecular Diagnosis*, 4(4):289–298, 1999.

[65] C. Hoare. Quicksort. *Computer Journal*, 5:10–15, 1962.

[66] J. Hopcroft and R. Karp. An $n^{5/2}$ algorithm for maximum matching in bipartite graphs. *SIAM Journal of Computing*, 2:225–231, 1973.

[67] G. Iverson, S. Coleridge, K. Fulda, and J. Licciardone. What factors influence a family physician's decision to refer a patient to a specialist? *Rural and Remote Health Journal*, page Article No. 413, 2005.

[68] V. Iyengar. Transforming data to satisfy privacy constraints. In *Proceedings of the ACM SIGKDD International Conference on Data Mining and Knowledge Discovery*, pages 279–288, Edmonton, Canada, 2002.

[69] W. Jiang and C. Clifton. Privacy-preserving distributed k-anonymity. In *Proceedings of the* 19^{th} *Annual IFIP WG 11.3 Working Conference on Data and Applications Security, Lecture Notes in Computer Science vol. 3654*, pages 166–177, Storrs, CT, 2005.

[70] E. Johnson, W. Moe, P. Fader, S. Bellman, and Lohse. On the depth and dynamics of online search behavior. *Management Science*, 50(3):326–335, 2004.

[71] M. Kantarcioglu and C. Clifton. Privacy-preserving data mining of association rules on horizontally partitioned data. In *Proceedings of the ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, Madison, WI, 2002.

[72] M. Kantarcioglu and C. Clifton. Privacy-preserving data mining of association rules on horizontally partitioned data. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1026–1037, 2004.

[73] M. Kantarcioglu, J. Jin, and C. Clifton. When do data mining results violate privacy? In *Proceedings of the ACM SIGKDD International Conference on Data Mining and Knowledge Discovery*, pages 599–604, Seattle, WA, 2004.

[74] H. Kazazian. Overview: progress toward a new millennium of medical genetics. *Human Mutation*, 15:2–3, 2000.

[75] A. T. Kearney Inc. E-business spending now exceeds 20 percent of all i.t. expenditure, August 7, 2003. Available online at `http://www.atkearney.com/main.taf?p=1,5,1,135`.

[76] T. Klein, J. Chang, M. Cho, K. Easton, R. Fergerson, M. Hewett, Z. Lin, Y. Liu, S. Liu, D. Oliver, D. Rubin, F. Shafa, J. Stuart, and R. Altman. Integrating genotype and phenotype information: an overview of the pharmgkb project: Pharmacogenetics research network and knowledge base. *Pharmacogenomics Journal*, 1(3):167–70, 2001.

[77] R. Kraut, S. Kiesler, B. Boneva, J. Cummings, V. Helgeson, and A. Crawford. Internet paradox revisited. *Journal of Social Issues*, 58:49–74, 2002.

[78] M. Krawczak and D. Cooper. The human gene mutation database. *Trends in Genetics*, 13:121–122, 1997.

[79] R. Kruse, B. Ewigmen, and G. Tremblay. The zipper: a method for using personal identifiers to link data while preserving confidentiality. *Child Abuse & Neglect*, 25:1241–1248, 2001.

[80] K. LeFevre, D. DeWitt, and R. Ramakrishnan. Incognito: efficient full-domain k-anonymity. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 49–60, Baltimore, MD, 2005.

[81] C. Leonard, G. Chase, and B. Childs. Genetic counseling: A consumer's view. *New England Journal of Medicine*, 287:433–439, 1972.

[82] Y. Li, P. Ning, X. Wang, and S. Jajoda. Generating market basket data with temporal information. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, 2001.

[83] Z. Lin, M. Hewitt, and R. Altman. Using binning to maintain confidentiality of medical data. In *Proceedings of the American Medical Informatics Association Annual Symposium*, pages 454–458, San Antonio, TX, 2002.

[84] Z. Lin, A. Owen, and R. Altman. Genomic research and human subject privacy. *Science*, 305, 2004.

[85] Y. Lindell and B. Pinkas. Privacy preserving data mining. *Journal of Cryptology*, 15(3):177–206, 2002.

[86] B. Malin. Compromising privacy with trail re-identification: The reidit algorithms. Master's thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2002.

[87] B. Malin. Betrayed by my shadow: learning data identity via trail matching. *Journal of Privacy Technology*, page 20050609001, 2005.

[88] B. Malin, E. Airoldi, and K. Carley. A network analysis model for disambiguation of names in lists. *Computational and Mathematical Organization Theory*, 11:119–139, 2005.

[89] B. Malin, E. Airoldi, S. Edoho-Eket, and Y. Li. Configurable security protocols for multiparty data analysis with malicious participants. In *Proceedings of the* 21^{st} *IEEE International Conference on Data Engineering*, pages 533–544, Tokyo, Japan, 2005.

[90] B. Malin and L. Sweeney. Determining the identifiability of dna database entries. In *Proceedings of the American Medical Informatics Association Annual Symposium*, pages 537–541, Los Angeles, CA, 2000.

[91] B. Malin and L. Sweeney. Re-identification of dna through an automated linkage process. In *Proceedings of the American Medical Informatics Association Annual Symposium*, pages 423–427, Washington, DC, 2001.

[92] B. Malin and L. Sweeney. Compromising privacy in distributed population-based databases with trail matching: A dna example. Technical Report CMU-CS-02-192, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2002.

[93] B. Malin and L. Sweeney. Inferring genotype from clinical phenotype through a knowledge-based algorithm. In *Proceedings of the 2002 Pacific Symposium on Biocomputing*, pages 410–52, Lihue, HI, 2002.

[94] B. Malin and L. Sweeney. How (not) to protect genomic data privacy in a distributed network: using trail re-identification to evaluate and design anonymity protection systems. *Journal of Biomededical Informatics*, 37(3):179–192, 2004.

[95] B. Malin and L. Sweeney. ENRES: a semantic framework for entity resolution modelling. Technical Report CMU-ISRI-05-134, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2005.

[96] B. Malin and L. Sweeney. A secure protocol to distribute unlinkable health data. In *Proceedings of the American Medical Informatics Association Annual Symposium*, pages 485–489, Washington, DC, 2005.

[97] B. Malin and L. Sweeney. Composition and disclosure of unlinkable distributed databases. In *Proceedings of the* 22^{nd} *IEEE International Conference on Data Engineering*, Atlanta, GA, 2006.

[98] B. Malin, L. Sweeney, and E. Newton. Trail re-identification: learning who you are from where you have been. Technical Report LIDAP-WP12, Data Privacy Laboratory, Carnegie Mellon University, Pittsburgh, PA, 2003.

[99] P. Manasco. Ethical and legal aspects of applied genomic technologies: practical solutions. *Current Molecular Medicine*, 5:23–8, Feb 2005.

[100] A. Meyerson and R. Williams. On the complexity of optimal k-anonymity. In *Proceedings of the ACM SIGMOD-SIGACT-SIGART Principles of Database Systems*, pages 223–228, Paris, France, 2004.

[101] W. Moe and P. Fader. Capturing evolving visit behavior in clickstream data. *Journal of Interactive Marketing*, 18(1):5–19, 2004.

[102] W. Moe and P. Fader. Dynamic conversion behavior at e-commerce sites. *Management Science*, 50(3):326–335, 2004.

[103] A. Montgomery, S. Li, K. Srinivasan, and J. Liechty. Modeling online browsing and path analysis using clickstream data. *Marketing Science*, 23(4):579–595, 2004.

[104] National Association of Health Data Organizations. *NAHDO Inventory of Statewide Hospital Discharge Data Activities*. National Association of Health Data Organizations, Falls Church, VA, May 2000.

[105] H. Newcombe, J. Kennedy, S. Axford, and A. James. Automatic linkage of vital records. *Science*, 130:954–959, 1959.

[106] E. Newton, L. Sweeney, and B. Malin. Preserving privacy by de-identifying face images. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):232–243, 2005.

[107] A. Øhrn and L. Ohno-Machado. Using boolean reasoning to anonymize databases. *Artificial Intelligence in Medicine*, 15:235–254, 1999.

[108] L. O'Neill and J. Kuder. Explaining variation in physician practice patterns and their propensities to recommend services. *Medical Care and Research Review*, 62:339–357, 2005.

[109] Y. Park and P. Fader. Modeling browsing behavior at multiple websites. *Marketing Science*, 23(3):280–303, 2004.

[110] P. Pharow and B. Blobel. Security infrastructure services for electronic archives and electronic health records. In *Proceedings of Studies in Health Technolology and Informics*, volume 103, pages 434–440, 2004.

[111] A. Phua, R. Abdullah, and Z. M. Z. A pcr-based sex determination method for possible application in caprine gender selection by simultaneous amplification of the sry and aml-x genes. *The Journal of Reproduction and Development*, 49(4):307–311, 2003.

[112] S. Pitzek. Security - privacy on the internet. Technical report, Vienna University of Technology, Vienna, Austria, 2001. Available online at `http://www.vmars.tuwien.ac.at/courses/akti12/journal/01ws/article_01ws_%Pitzek.pdf`.

[113] S. Pohlig and M. Hellman. An improved algorithm for computing logarithms over $gf(p)$ and its cryptographic significance. *IEEE Transactions on Information Theory*, 24:106–110, 1978.

[114] M. Reiter and A. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, June 1998.

[115] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[116] M. Rothstein, editor. *Genetic secrets: protecting privacy and confidentiality in the genetic era*. Yale University Press, New Haven, CT, 1997.

[117] R. Roy and D. Steffens. Infrared fluorescent detection of pcr amplified gender identifying alleles. *Journal of Forensic Science*, 42(3):452–60, 1997.

[118] A. Rutherford, R. Botha, and M. Oliver. Towards a hippocratic log file architecture. In *ACM International Conference Proceeding Series; Volume 75: Proceedings of the 2004 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries*, pages 186–193, Stellenbosch, South Africa, 2004.

[119] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[120] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information (abstract). In *Proceedings of the 7^{th} ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, page 188, Seattle, Washington, 1998.

[121] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical Report Technical Report SRI-CSL-98-04, SRI Computer Science Laboratory, Palo Alto, CA, 1998.

[122] T. Sander. Efficient accumulators without trapdoor. In V. Varadharajan and Y. Mu, editors, *Lecture Notes in Computer Science 1726: Proceedings of the 2^{nd} International Conference on Information and Communications Security (ICICS '99)*, pages 252–262, New York, NY, 1999. Springer-Verlag.

[123] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Canada, 2002.

[124] U. Sax and S. Schmidt. Integration of genomic data in electronic health records opportunities and dilemmas. *Methods of Information in Medicine*, 44:546–550, 2005.

[125] S. Scheleur and C. King. Retail e-commerce sales in fourth quarter 2003 were 17.2 billion, up 25.1 percent from fourth quarter 2002, census bureau reports. *U.S. Department of Commerce News*, February 23, 2004. Available online at `http://www.census.gov/mrts/www/data/pdf/03Q4.pdf`.

[126] D. Selkoe. Alzheimer's disease: genes, proteins, and therapy. *Physiological Review*, 81(2):741–66, 2001.

[127] A. Serjantov and G. Danezis. Towards an information-theoretic metric for anonymity. In *Proceedings of the 2^{nd} Privacy Enhancing Technologies Workshop, Lecture Notes in Computer Science vol. 2482*, pages 41–53, Berlin, Germany, 2002. Springer-Verlag.

[128] C. Shannon and W. Weaver. *The mathematical theory of communication*. University of Illinois Press, Urbana, IL, 1949.

[129] S. Sherry. Use of molecular variation in the ncbi dbsnp database. *Human Mutation*, 15:105–113, 2000.

[130] A. Skowron and C. Rauszer. The discernibility matrices and functions in information systems. In *Intelligent decision support: Handbook of applications and advances of rough set theoryvolume 11 of System Theory, Knowledge Engineering and Problem Solving*, pages 331–362. Kluwer, Dordrecht, The Netherlands, 1992.

[131] Southern Illinoisan v. Department of Public Health and John Lumpkin. Circuit court of the first judicial circuit, no. 5-99-0568, 2001 wl 337191.

[132] State of Illinois Health Care Cost Containment Council. *Data release overview*. State of Illinois Health Care Cost Containment Council, Springfield, IL, March 1998.

[133] S. Steinbrecher and S. Köpsell. Modelling unlinkability. In *Proceedings of the* 3^{rd} *Privacy Enhancing Technologies Workshop, Lecture Notes in Computer Science vol. 2760*, pages 32–47, Berlin, Germany, 2003. Springer-Verlag.

[134] L. Sweeney. Weaving technology and policy together to maintain confidentiality. *Journal of Law, Medicine, and Ethics*, 25:98–110, 1997.

[135] L. Sweeney. Uniqueness of simple demographics in the us population. Technical Report LIDAP-WP04, Data Privacy Laboratory, Carnegie Mellon University, Pittsburgh, PA, 2000.

[136] L. Sweeney. *Computational Disclosure Control: Theory and Practice*. PhD thesis, Massachusetts Institute of Technology, 2001.

[137] L. Sweeney. Information explosion. In P. Doyle, J. Lane, J. Theeuwes, and L. Zayatz, editors, *Confidentiality, Disclosure, and Data Access: Theory and Practical Applications for Statistical Agencies*. Urban Institute, Washington, DC, 2001.

[138] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness, and Knowledge-based Systems*, 10(5):571–588, 2002.

[139] L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness, and Knowledge-based Systems*, 10(5):557–570, 2002.

[140] L. Sweeney and R. Gross. Mining images in publicly-available cameras for homeland security. In *Proceedings of the AAAI Spring Symposium on AI Technologies for Homeland Security*, Palo Alto, CA, 2005.

[141] P. Tarczy-Hornoch, M. Covington, J. Edwards, P. Shannon, S. Fuller, and R. Pagon. Creation and maintenance of helix, a web based database of medical genetics laboratories, to serve the needs of the genetics community. In *Proceedings of the American Medical Informatics Association Annual Symposium*, pages 341–345, Miami Beach, FL, 1998.

[142] V. Torra. Re-identifying individuals using OWA operators. In *Proceedings of the* 6^{th} *International Conference on Soft Computing*, Iizuka, Japan, 2000.

[143] V. Torra. Towards the re-identification of individuals in data files with non-common variables. In *Proceedings of the* 14^{th} *European Conference on Artificial Intelligence*, pages 326–330, Berlin, Germany, 2000.

[144] V. Torra. OWA operators in data modeling and re-identification. *IEEE Transactions on Fuzzy Systems*, 12(5):656–660, 2004.

[145] T. Uno. Algorithms for enumerating all perfect, maximum and maximal matchings in bipartite graphs. In *Proceedings of the* 8^{th} *International Symposium on Algorithms and Computation, LNCS vol 1350*, pages 92–101, London, UK, 1997. Springer-Verlag.

[146] J. Vaidya and C. Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the* 9^{th} *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, 2004.

[147] L. Vaszar, M. Cho, and T. Raffin. Privacy issues in personalized medicine. *Pharmacogenomics*, 4(2):107–112, 2003.

[148] T. Victor and R. Mera. Record linkage of healthcare insurance claims. In *Proceedings of MEDINFO*, volume 10, pages 1409–1413, 2001.

[149] S. Vinterbo. Privacy: a machine learning view. *IEEE Transactions on Knowledge and Data Engineering*, 16(8):939–948, 2004.

[150] S. Vinterbo, L. Ohno-Machado, and S. Dreiseitl. Hiding information by cell suppression. In *Proceedings of the American Medical Informatics Association Annual Symposium*, pages 726–730, Washington, DC, 2001.

[151] J. Wei. Markov edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:311–321, 2004.

[152] L. Willenborg and T. de Waal. *Statistical Disclosure Control in Practice*. Springer, New York, NY, 1996.

[153] B. Winkelmann. Pharmacogenomics, genetic testing and ethnic variability: tackling the ethical questions. *Pharmacogenomics*, 4(5):531–535, 2003.

[154] W. Winkler. Matching and record linkage. In B. Cox et al., editor, *Business Survey Methods*, pages 355–384. J. Wiley, New York, NY, 1995.

[155] W. Winkler. Re-identification methods for evaluating the confidentiality of analytically valid microdata. In J. Domingo-Ferrer, editor, *Statistical Data Protection*. Office for Official Publications of the EC, Luxemburg, 1999.

[156] W. Winkler. Methods for record linkage and bayesian networks. Technical Report Statistics-2002-05, Statistical Research Division, U.S. Census Bureau, Washington, DC, 2002.

[157] W. Winkler. Using simulated annealing for k-anonymity. Technical Report 2002-07, US Census Bureau Statistical Research Division, Washington, DC, 2002.

[158] W. Winkler and F. Scheuren. Recursive analysis of linked data files. In *Proc. Census Bureau Ann Res Conf*, pages 920–935, Washington, DC, 1996.

[159] World Health Organization. *International classification of diseases, 9th revision: clinical modification*. Edwards Brothers, Ann Arbor, MI, 1980.

[160] J. Wylie and G. Mineau. Biomedical databases: protecting privacy and promoting research. *Trends in Biotechnology*, 21(3):113–116, 2003.

[161] A. Yao. How to generate and exchange secrets. In *Proceedings of the* 27^{th} *IEEE Symposium on Foundations of Computer Science*, pages 162–167, Washington, DC, 1986.

[162] J. Zachary. A decentralized approach to secure group membership testing in distributed sensor networks. In *Proceedings of the Military Communications Conference*, Boston, MA, 2003.