Anonymity in Socio-Digital Systems

by

Aaron Beach

B.S., Northwestern University, 2006

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

2011

This thesis entitled:
Anonymity in Socio-Digital Systems
written by Aaron Beach
has been approved for the Department of Computer Science

_____

Richard Han

_____

John Black

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Beach, Aaron (Ph.D., Computer Science)

Anonymity in Socio-Digital Systems

Thesis directed by Prof. Richard Han

Social data is particularly interesting to anonymity research due to its personal nature and recent increase in occurrence and usage. Despite the personal nature and prevalence of social networks and their data little research has been devoted to their anonymization. This thesis presents work by the author to define and apply anonymity metrics to socio-digital systems.

This thesis discusses **(1)** new anonymity definitions that take into account the highly inter-related and often leaked nature of social network data, **(2)** algorithms to measure these anonymity metrics efficiently, **(3)** anonymization algorithms which balance information gain with anonymity, **(4)** a general data model connecting anonymity, information metrics, and knowledge models to make anonymity practical for real world usage and **(5)** a prototype system which anonymizes Facebook data released over an API to support anonymous socio-digital systems and applications.

Dedication

This thesis is dedicated to my family & friends.

Acknowledgements

Much of the research in this thesis was conducted in collaboration with Richard Han and Mike Gartrell. My PhD was primarily funded by the National Science Foundation and the University of Colorado.

Contents

Tables

Table

Figures

Figure

0.1 Thesis Statement

- **This thesis demonstrates that anonymity is practical for Socio-Digital applications** by defining anonymity metrics specifically for social data, designing measurement algorithms that scale well, identifying/defending against advanced re-identification attacks and implementing a prototype that demonstrates how applications can function with anonymized data.

0.1.1 Research Challenges

(1) Social Network data is highly personal, inter-related, and often publicly leaked, this makes it very hard to anonymize.

(2) Social Networks produce huge amounts of data that is used in real-time. Therefore, anonymization technologies must be scalable.

(3) Anonymization is often at odds with information gain. Anonymization technologies must find a balance with data utility.

(4) Knowledge & Trust models must be integrated with Anonymity to defend against attacks that use multiple accounts and queries.

0.1.2 Research Solutions

(1) New anonymity metrics (Social-K & q-Anon) were developed to support high levels of anonymity with high-levels of data utility.

(2) Advanced Logic-Minimization algorithms were modified and new algorithms developed to measure anonymity efficiently.

(3) A unifying data model (AnonyGraph) was designed to integrate anonymity metrics with various existing data sources, information metrics, and knowledge models.

(4) A prototype (Whoville) was developed to test anonymization metrics and algorithms in the real world.

Chapter 1

Introduction

This thesis considers whether anonymization technologies can practically be applied to social network data. There are three primary questions investigated by this Thesis work:

(1) Social data is often released from web APIs which require fast responses to support real-time user interaction. Can anonymization algorithms perform in real-time?

(2) Social data is highly inter-related and often publicly available. Can anonymization algorithms achieve a high level of anonymity with such data?

(3) Socio-digital systems benefit from the personal nature of social data. Can anonymization algorithms produce data that is both anonymous and still useful to socio-digital systems?

In order to address these questions, five major steps were taken by the researcher, these are:

(1) Modified existing anonymity metrics and developed new algorithms to achieve high-levels of anonymity with social data.

(2) Tested different methods for valuing information to help anonymization algorithms decide which data to release, generalize, and suppress.

(3) Designed and implemented a general framework, "AnonyGraph," which allows many different information metrics and anonymization algorithms to be integrated in real-time with social data sources.

(4) Developed a Facebook application, "Whoville," which serves as an anonymization layer on top of Facebook's web API, allowing applications to be run with anonymous data.

(5) Tested different social applications with "Whoville" to understand which types of applications function well with anonymized data.

Below we shortly summarize what the primary challenges were in each of the steps above and how these challenges were overcome.

Social informatics is one of the hottest and most active areas of informatics [35] and is particularly interesting in the context of anonymization research due to the fact that, by definition, it relates information to people's identity. However, most anonymity research assumes that data looks and acts like research data - well categorized as either private or public and generally released as one monolithic piece [55]. Also, most anonymity research assumes that the user of anonymized data is a researcher or analyst. Social networks and their data have not received much attention from anonymity researchers and this oversight may be attributed to the very recent explosion of social networks [36]. Facebook in particular, can now be considered the single largest source of personal data that has ever existed creating and indexing as many items every 3.5 hours as are in the entire library of congress [17] [2]. A primary motivation for this thesis is to gain a better understand of how anonymity relates to social networks and their data.

Social data tends to be very highly related and shared between users. Furthermore, privacy settings for much social data tend to be contextual and change over time meaning that assumptions cannot be made about whether or not data exists publicly or whether it might be considered private by its owner. As such, anonymity metrics must be designed with the assumption that any data may be public and yet designed to measure the anonymity of any piece of social data as it may be considered private by its owner [13]. Existing anonymity metrics such as K-Anonymity and l-Diversity were modified slightly to account for these assumptions. A version of K-Anonymity, Social-K, was developed by the author [9] which allows K-Anonymity to be measured quickly for social network without making assumptions about whether or not the data is public/private. However, for most types of social data this results in a significant loss in data utility.

Another anonymity metric, q-Anon [4] was developed which achieve equal levels of anonymity with a significant increase in data utility if applications can make queries using conditionals on social data rather than specifying users directly with identifiers. Furthermore, q-Anon can be measured in real-time using logic minimization algorithms. The author investigated which types of applications might benefit using q-Anon [5]. In general, applications that rely on demographic data to personalize a user's experience, but do not require identity to differentiate between users, can function at a very high level of personalization with anonymized data. However, anonymity metrics are only one piece in an anonymous data release system and must be integrated with many other data requirements to support a truly anonymous data service in the real-world.

When applying anonymity to a data release system in the real-world the designer or engineer must deal with a host of related questions that are not solved by any single anonymity metric. For instance, K-Anonymity guarantees that no data tuple maps uniquely to a user however it says nothing about the relational diversity between these tuples which must also be taken into account to protect user privacy. Furthermore, if the anonymized data is to be useful for its intended application it should integrate appropriate information metrics and if the anonymous data release system is to be resilient against attackers that use knowledge from many queries or accounts, the system must keep track of what knowledge it has revealed in the past (and to who it has released this knowledge). There is no one solution that answers all of these questions and their existence is probably one reason why most anonymity research has seen limited real-world application. While this thesis does not purport to solve every problem associated with real-world implementation of anonymous data systems, it does attempt to show how many anonymity metrics can be integrated in a real-world system. To do this, the author developed a general model on which anonymity, data utility, and knowledge models can be integrated.

AnonyGraph models social data and its relationships as a simple multi-graph, much like RDF uses triples to represent semantic relationships [37]. Using this simple model the author was able to define many anonymity metrics, information metrics, and create efficient records of what knowledge the system has ever released. With these many different types of informatics defined on the same data model, the only questions that a privacy engineer need concern themselves with is how to map their data into the AnonyGraph model

and which metrics or rules they are interested in using with their data. Using this model allowed the author to integrate the anonymity metrics designed for social data with the Facebook API, allowing for any Facebook application to be quickly ported to use anonymous data.

By implementing a mapping between Facebook's Graph API and the AnonyGraph model the author was able to route the Facebook Graph API through AnonyGraph before its results were returned to an application. This allowed the author to test AnonyGraph on a wealth of social data using many existing Facebook applications. To do this, a Facebook application, "Whoville" was implemented on top of the Facebook API which mirrored the entire API, however all information that passed through Whoville was anonymized depending on a set of metrics agreed upon by the user and application. To the user, Whoville is designed to look and feel like an application page in which "apps" can be added, deleted, or developed.

Using Whoville, it has been shown that a large class of social applications do not need to know the user's identity to function well. The most interesting examples in this class are advertising and recommendation applications that work on representative anonymous sets of data. The author has identified advertising and recommendation as the most practical and prevalent applications for use with anonymization. They are particularly interesting because many "social" applications do not use personal information for anything but advertising.

This thesis serves as a proof-of-concept to the growing industry of applications that use people's social information. It shows that such information is still beneficial when anonymized and that providing personalized recommendations or advertisements does not require the users identity. Hopefully, the work involved in this thesis will raise awareness of this alternative usage model for social data.

Chapter 2

Background & Related Work

This section summarizes the most pertinent developments, phenomena, and previous research as it relates to the thesis. First a set of recent developments, which function primarily as motivations, are discussed. Then terms, definitions, and techniques used in anonymity research are summarized to provide the reader with the appropriate vocabulary used throughout the paper.

2.1 Motivations

This section discusses the recent developments that have motivated the models and assumptions throughout this thesis. Many of these developments happened in only the last few years and some of them are just now happen**ing**.

2.1.0.1 Facebook

Every month over half a billion people log into their Facebook account [1]. Over 200 million of these people log in through their phones. Facebook users install over 20 million applications every day and over 2.5 million websites including most of the top 100 trafficked websites, are connected with Facebook databases through the Facebook API or Facebook Connect. three and a half years ago, when I began researching how Facebook data might be used to drive context-aware applications Facebook only had 50 million users, no mobile site, and had just launched its application platform. K-anonymity and much of the anonymity research was done well before social network data had grown to its current size and was open to the world through an API. Query-based Web APIs were not commonly used to publicly release private

data for many reasons. Open APIs tend toward looser management of who accesses the API and makes it much harder to draw clear conclusions about who has access to what data. However, this is the model that has flourished for many social networks including Linkdin, MeetUp, and most prominently Facebook. As such, the interactive API model is now the most common method used to publicly release private data and its widespread acceptance was so quick that research could not develop an anonymous alternative before Facebook had spread to include data from nearly half of all Internet users. As it seems that this API model is here to stay and if anything expanding to take a more central role in exchanging private data, this model and in particular, the graph model chosen by Facebook [33], should become an important motivating model for anonymity research.

One of the reasons Facebook data is so interesting for anonymization is that every piece of data created on it is mapped to people and their identities. Being as such, it is arguably the largest source of personal data that has ever existed, creating indexing, and mapping people to 42 Million new items an hour, which is equivalent to re-creating and indexing the entire library of congress every 3.5 hours. Based on the author's current surveys discussed in section 11, the 30 billion new items created in Facebook every month are connected into Facebook by an estimated Ĩ trillion relationships. When followed far enough every one of these relationships inevitably end up connecting to a person's identity, and as such are part of that person's internet footprint and could be used to identify that user.

2.1.0.2 Smart Phones

The past few years have seen explosive growth in mobile smartphones, and in particular "app phones" such as the iPhone, Droid, Nexus One, Palm Pre and Windows Phone. These phones mated to app stores have simplified loading of applications and resulted in commonplace use of third party applications. This trend toward "app phone" usage has accelerated the development of contextual mobile computing services. App phone users may browse near their location for friends (Latititude), events (Yelp), tweets (Twitter), or buzzes (Google Buzz). Also, apps allow people to connect virtual information with their surroundings through augmented reality apps such as virtual spray-painting (SimSpray) or a virtual wiki for the physical world (wikitude). Using digital cameras and/or other optical sensors, accelerometers, GPS, gyroscopes,

solid state compasses, RFID, wireless sensors and data connections to social networks, location databases or event lists the "app phone" is accessing an incredibly rich of set of data that is changing how the user sees and interacts with the world around them.

2.1.0.3 Mobile Social Networking

Mobile social networks combine location awareness from mobile phones with context awareness from social networks, e.g. friendship relationships and personal profile preferences, to create new socio-digital systems, such as viewing my friends nearby, or showing services that I might be interested in that are nearby. Commercial mobile social networks such as Loopt, Brightkite, and Foursquare have taken off - and over 100 million of the 500+ million Facebook users now access Facebook through their mobile devices. Many online data sources such as Facebook now have Web APIs that open up their information to cellular data connections. Research projects in mobile social networks have considered how mining social information about proximate individuals might enhance or drive their interaction [24, 50]. It has been shown that online social networks can affect how people physically see one another [10] and current projects are considering how large-scale pattern inference might "fuse" the many different types of sensors, location and social data to direct cohesive social environments [12].

2.1.0.4 Context-Aware Systems

Using information from the environment such as sensors and GPS or by accessing external information relevant to the environment such as social network data, a context may be inferred that characterizes how a user interacts with an application. These applications or systems may be called context-aware. Many credit the Active Badge System [60] as the first explicitly "context-aware" application. From these early systems that used simple "badges" to announce ones presence more complicated systems have evolved to utilize advanced devices on the person and in the environment interacting through wireless data connections to harvest the wealth of personal information online. The use of online databases to distribute social, location, activity, or messaging information is becoming ubiquitous in context-aware systems. Many of these context-aware systems are being used to drive social trends and interactions. This trend in context-aware

systems driving social interactions using online database mining, in particular that concerning private social data, is the focus of this paper. This project is largely motivated by the increasing number of context-aware research and industry applications that are driven by API access to online databases.

2.1.0.5 Privacy in Socio-Digital Systems

Privacy within the context of social networks is becoming a very hot topic in both research and among the public. This is largely due to an increase in use of Facebook and a set of high-profile incidents such as the de-anonymization of public data sets [48]. However, public concern about privacy has not necessarily translated into more responsible usage of the existing privacy mechanisms. It is suggested that this may be due to the complexity of translating real-world privacy concerns into online privacy policies, as such it has been suggested that machine learning techniques could automatically generate privacy policies for users [27].

Privacy and security issues have been considered with the realm of mobile, social, and context-aware system [57] [7]. Previous work at Duke University [18] [44] has dealt with privacy and anonymity questions as they apply to sharing presence information with other users and matching users with a shared location and time. This prior work in presence sharing relates to anonymity in socio-digital systems in that any presence or location sharing would need to also be anonymous, furthermore PP-anonymity may provide certain functions necessary to associate user preferences anonymously with user location for use by socio-digital systems. For instance, SmokeScreen [18] presents a protocol by which devices may broadcast identifiers that can be resolved to an identity through a trusted broker system. This identity could then be used to access anonymized personal information to drive socio-digital systems.

2.2 Previous Anonymity Work

Research into anonymizing data sets (or microdata releases) to protect privacy directly apply to the work in this proposal. Most of this research has taken place within the database research community. In 2001, Sweeney published a paper [56] describing the "re-identification" attack in which multiple public data sources may be combined to compromise the privacy of an individual. The paper proposes an anonymity

definition called k-anonymity. This definition was further developed and new approaches to anonymity were proposed that solved problems with the previous approaches. These later anonymity definitions include p-sensitivity [58], ℓ-diversity [42], t-closeness [40], differential privacy [21], and multi-dimensional k-anonymity [38]. All of these privacy approaches and their associated terms are discussed later in this section.

Methods have been developed that attempt to efficiently achieve anonymization of data sets under certain anonymity definitions. Initially simple methods such as suppression (removing data) and generalization have been used to anonymize data sets. Research has sought to optimize these methods using techniques such as minimizing information loss while maximizing anonymity [52]. One approach called "Incognito" considers all possible generalizations of data throughout the entire database and chooses the optimal generalization [39]. Another approach called "Injector" uses data mining to model background knowledge of a possible attacker [41] and then optimizes the anonymization based on this background knowledge. A few of these approaches will be discussed in more detail at the end of this section

2.2.1 Data Types

The data sets most often considered in anonymization research usually take the form of a table with at least three columns that usually include zip code, age, (sometimes gender), and a health condition. This data set is convenient for many reasons, it is very simple, but it also contains (and doesn't contain) certain data types that are important to anonymization. First, it does not contain any ***unique identifiers*** such as social security numbers. The obvious first step in anonymizing a data set is removing the unique identifiers. The most common unique identifiers discussed in this paper are social network user IDs (Facebook ID or username). The data set also contains a set of ***quasi-identifiers*** - age, zip code, and gender are the most common. Data may be considered a quasi-identifier if it can be matched with other data (external to the data set) that maps to some unique identifier. The re-identification attack consists of matching a set of quasi-identifiers from an anonymized data set to a public data set (such as census or voting records) effectively de-anonymizing the anonymous data set. It is important to note that quasi-identifiers are assumed to be public (or possibly public) by definition and as such are not the primary data to be protected. The data that

are to be protected from re-identification are termed ***sensitive attributes***. Sensitive attributes are not assumed to be publicly associated with a unique identifier and as such their relationship to the quasi-identifiers within a data set is what concerns most anonymity definitions. In most research examples health conditions or disease attributes are considered sensitive attributes. A set of sensitive attributes that share the same set of quasi-identifiers are, together with their quasi-identifier set, called an ***equivalence class***. For example, the health conditions associated with 25 year old males living in a particular zip code would be an equivalence class.

Furthermore, the $\{$*zip code, gender, age, disease*$\}$ data set is useful because its data exhibit different characteristics. Zip codes are structured hierarchically and ages are naturally ordered. The rightmost digits in zip codes can be removed for generalization and ages can be grouped into ranges. Gender presents a binary attribute that cannot be generalized because doing so would render it meaningless. Finally, using disease as the sensitive value is convenient since health records are generally considered to be private. Also, disease is usually represented as a text string that presents semantic challenges to anonymization such as understanding the relationship between different diseases.

2.2.2 Anonymity Definitions

K-Anonymity [56] states that a data set is k-anonymous if every equivalence class is of size k (includes at least k records). However, it was observed that if the sensitive attribute was the same for all records in an equivalence class then the size of the equivalence class did not provide anonymity since mapping a unique identifier to the equivalence class was sufficient to also map it to the sensitive attribute, this is called ***attribute disclosure***.

p-sensitivity [58] was suggested to defend against attribute disclosure while complementing k-anonymity. It states that along with k-anonymity there must also be at least p different values for each sensitive attribute within a given equivalence class. In this case, an attacker that mapped a unique identifier to an equivalence class would have at least p different values from which only one correctly applied to the unique identifier. One weakness of p-sensitivity is that the size and diversity of the anonymized data set is limited to the diversity of values in the sensitive attribute. If the values of the sensitive attribute are not uniformly distributed

across the equivalence classes there will be significant data loss even for small p values.

ℓ-diversity [42] was suggested to prevent attribute disclosure through either requiring a minimum of "entropy" in the values of the sensitive attribute or by placing a minimum and maximum on how often a particular value may occur within an equivalence class. While preventing direct attribute disclosure such an anonymization may result in the distribution of sensitive attribute values being significantly skewed. If the distribution of a sensitive attribute is known, this knowledge could be used to calculate the probability of a particular sensitive attribute value being associated with a unique identifier. For instance, while only 5/1000 records in a data set contain a particular disease, there may exist an equivalence class in the anonymized data set for which half the records contain the disease, implying that members of the equivalence class are 20 times more likely to have the disease.

t-closeness [40] approaches the problem of skewness by bounding the distance between the distribution of sensitive attribute values in the entire data set and their distribution within each equivalence class. The problem (or trade-off) with t-closeness is that it achieves anonymity by limiting the statistical difference between equivalence classes and in doing so minimizes any interesting correlations or statistics that could be drawn from the anonymized data set. Furthermore, it is not clear that there is any efficient way to enforce t-closeness on a large data set [20].

Defending against skewness attacks presents a paradox - data sets are useful because they contain correlations which say something about the world outside of the data set which is what a skewness attack does. In this sense the utility of a data set and its danger to privacy are correlated. Skewness attacks should therefore be approached practically considering the nature of the sensitive attributes in terms of the danger of their compromise and the utility they provide by being released.

Multi-Dimensional K-Anonymity [38] proposes a more flexible approach to K-anonymity in which equivalence classes are clustered or generalized across a table in more than one dimension. This flexibility allows for a higher degree of optimization than simply generalizing each column of a database separately. While optimizing the selection of equivalence classes is NP-hard a greedy approximation algorithm for multi-dimensional K-anonymity has been shown to outperform exhaustive optimal algorithms for a single dimension.

Differential Privacy [21] takes a different perspective on privacy than the other privacy models discussed in this thesis. Most interestingly, it assumes an interactive database model in which, as opposed to a non-interactive microdata release, the data collector provides an interface through which users may query and receive answers. As will be discussed in section 3, this model fits that currently used by many social network APIs and is much more practical for the types of data use associated with social networks. However, differential privacy focuses primarily on statistical databases, the queries on which are answered with added noise which guarantees a maximum level of danger to the privacy of anyone participating in the database. The difficulty in applying this to social networks is in appropriately measuring or defining "noise" in a way that meaningfully integrates with the data's use by social network applications. While interesting, this paper does not deal with the same problem. However, the interactive database model assumed by differential privacy is promoted as the appropriate model for anonymity mechanisms applied to social networks.

2.2.3 Anonymization Techniques

Finally, anonymization commonly consists of generalizing, perturbing, or suppressing data. Generalization of data requires some ordering or structure to the data type such that many specific values of data can be grouped as a related, but more general value. Perturbation involves distorting or adding noise to a value. Some types of data such as images may be perturbed and still useful. However, much social network data may not be useful when modified or generalized and as such must be removed or suppressed - as such suppression is the most generally applicable approach to anonymization when one cannot make assumptions about how generalization or perturbation will affect the utility of the data. Also, it should be noted that in social networks it is very common for a data field to have many values separated by commas. When items are suppressed from such data fields it could be said that the value of the data field has been generalized - however, this paper will refer to such an anonymization technique as suppression.

Chapter 3

Challenge: Applying Anonymity in Socio-Digital Systems

This section will highlight the challenges and motivations involved in applying existing anonymity definitions and models to social network data. This section will begin by discussing the difficulty in applying traditional anonymity measurements to socio-digital systems, highlighting the nature of social data and how it is commonly accessed and used. The section will go on to discuss the authors early investigations into how anonymity might be integrated into an anonymous socio-digital system.

3.1 Social Data and its Challenges to Anonymization

Most anonymity research assumes the same convenient data set discussed in section 2. This data set contains a few quasi-identifiers that are usually hierarchical or ordered such that they can be easily generalized along with a clearly identifiable sensitive attribute. Furthermore, anonymity research has generally assumed a rather research-centric non-interactive data release model. This section discusses these issues in detail, explaining why the vague nature and application-centric interactive usage of social data is incompatible with traditional anonymity measurements.

3.1.0.1 Data Characterization

The traditional anonymity data set consists of some version of $\{$*zip code, gender, age, disease*$\}$. This data set is easy to understand and naturally translates to privacy examples since it uses traditionally accepted quasi-identifiers and sensitive data. Social networks do not provide such convenient data.

Social network data often consists of attributes such as name, unique ID, friendship links, favorite

movies/music/activities, birthdate, gender, hometown, group associations, guestbook or wall posts, pictures, videos, messages, status updates, and sometimes current location. To simplify discussion of social network data in this section we will assume the usage model and data types of the largest social network, Facebook.

The first task in anonymizing social network data is to specify which attributes are unique identifiers, quasi-identifiers and sensitive attributes. The unique ID is a unique identifier and some people may wish that their name be considered a unique identifier as well. There are then the traditional quasi-identifiers including city, birthdate, and gender - however these data types are often targeted by Facebook applications as the attributes of interest (such as birthday calendar applications), and may be considered sensitive attributes by some users. In fact, depending on a user's privacy settings nearly every data attribute may be publicly available or semi-public within a region or network. Furthermore, these privacy settings are constantly changing and the user's privacy expectations may change drastically depending on context. Given the lack of clear assumptions as to the public availability of most information on Facebook, all data types should be considered a quasi-identifier. Also, given the complex nature of social network applications, (e.g., calendars of friends' birthdays, context-aware music players, or location-aware games) all attributes may potentially be considered sensitive attributes within certain contexts and as such all data types should be considered sensitive attributes.

This poses significant problems to utilizing traditional anonymity solutions for social networks. If a single attribute is considered both quasi-identifiable and sensitive it renders k-anonymity incompatible with ℓ-diversity, p-sensitivity, and t-closeness. This is because equivalence classes must share the same quasi-identifier set (have the same values for all quasi-identifier attributes) and ℓ-diversity, p-sensitivity, and t-closeness require some variation of all sensitive attributes. t-closeness, ℓ-diversity, and p-sensitivity all assume equivalence classes defined by a shared quasi-identifier set. If some attribute was both sensitive and quasi-identifiable then it would be required to have the same value throughout the equivalence class (to avoid re-identification) and it would be required to have different values (to avoid attribute disclosure) rendering ℓ-diversity, p-sensitivity, and t-closeness [1] meaningless.

[1] being quasi-identifiable, the attribute must have a uniform distribution of one value

3.2 When is Data Private and Public?

Let us consider in more detail why social network privacy and anonymization is so contextual and diverse by considering a simple social network game and a public service application tracking disease spread. These examples involve a college student sharing their social graph with the two different Facebook applications. The first application is Farmville, a popular game in which users manage a farm and interact with their farm neighbors. However, users can only be neighbors with users who are also friends on Facebook. To verify that users are friends the user in our example is happy to share their social graph not seeing any reason why sharing such information should be considered a violation of their privacy. Later in the school year, the student chooses to take part in a campus wide STD (Sexually Transmitted Disease) screening program that anonymously screens students for STDs and then through a Facebook application uses student's social graph to direct further anonymous STD screening information toward those students at highest risk. The problem is that a Facebook user's social graph almost always identifies them uniquely. In this situation the user's social graph has privacy implications it didn't when playing Farmville and while the student is eager to help the screening program help other students, they are hesitant to give up their anonymity, which was one of the primary factors for why they were willing to participate in the program in the first place. Using an anonymous interface such as that described in secion 11 or in the paper on q-Anon [4], it is possible for the campus screening program to request an anonymized list of the users friends that does not uniquely identify the user [2] . Furthermore, if the anonymous interface supported more complex queries it would be possible for the screening program to request a list of the most at-risk or STD-connected students without violating the anonymity of those students that agreed to anonymously participate in the screening program. [3]

These two examples show how a user may consider the release of certain information a privacy violation in certain cases and not in others and highlights the diverse ways in which the same piece of social network data may be used. Due to the initial release of the students social network graph (along with the fact that their social graph is available to their friends) we must assume that the students social graph could

[2] The anonymous API would require that students are referenced using anonymous identifiers as described in this paper. [11].

[3] One might think that the list of at-risk users constitutes an anonymity violation, however the program could simply contact the full list of at-risk students, who could then decide to get tested or not and could be tested with their results being anonymous.

be used to re-identify anonymous information (STD screening results) with the user and as such must be anonymized.

Furthermore, the users perception of data as private may not only change depending on how it's used, it may change over time. For example, pictures, preferences, or test results that the student in our example did not consider private in college may suddenly be considered much more private when the student enters the job market or begins raising a family. As such, it should be clear that making public/private distinctions are rather complicated and context-dependent when it comes to social network data.

3.2.0.2 Data Usage

The obstacles to applying traditional anonymity approaches to social networks arise largely from the assumption that data is released in a non-interactive form as a database or table. Such a table results in each set of quasi-identifiers being uniquely associated with its sensitive attributes. However, this type of data association may not be necessary for many applications. Furthermore, a particular application may only be interested in a particular set of equivalence classes which include only a subset of quasi-identifiers. However, an alternative to the non-interactive data release model exists. In an interactive model, a "trusted" data collector provides an interface through which users or applications may query for information through an API or general language like SQL. Fortunately, Facebook already offers both a function API and a SQL-like language called FQL. Facebook's interface is currently used by over 2.5 million applications and most major websites on the internet. Recent developments, such as the announcement that Facebook will soon integrate users' location information and automatically release this information to "trusted" third-parties without the user's explicit approval, provide a clear motivation for solutions to anonymity within an interactive data release model.

3.3 Early Experience with Socio-Digital Systems

This section will discuss the author's early experience developing socio-digital systems. This experience motivated the author to investigate how anonymity might be integrated into such systems.

3.3.0.3 WhozThat

WhozThat was an early prototype Mobile Social Network which allowed users to search nearby for other users and filter those users based on their Facebook profile info. This allowed users to connect, chat, and share information with people and context-aware services around them through their social network, Facebook. A few applications were built on this system including a context-aware jukebox which automatically built playlists based on the Facebook profiles of the users in the room. Presence was shared from users phones to a local jukebox via bluetooth, giving the jukebox access to their Facebook profile. The music listed on the users profile was used as the basis for creating a music playlist. This prototype was presented at the poster session at Mobisys in 2008. This original system has motivated work into privacy concerning how presence and profile information can be shared without compromising anonymity. This idea was then used as the basis for the Social-K project which attempted to implement a related application (SocialAware Flicks) anonymously.

3.3.0.4 SocialAware Flicks

The SocialAwareFlicks application described in [30] was used as an example of a socio-digital system that queries Facebook profile information. SocialAwareFlicks displays recommended movie trailers that match the movie preferences of one or more users jointly watching a common display. SocialAware-Flicks could be deployed in locations such as video-rental establishments for marketing purposes, to make customers aware of new video rentals that match their interests. SocialAwareFlicks uses a stationary desktop or laptop computer to detect the presence of nearby mobile devices.It then obtains the favorite movie lists of these users and uses the Netflix REST API web service [49] to obtain a list of similar movies for each of the users' favorite movies. After obtaining these recommendations for each user, the SocialAware-Flicks uses heuristics to create an aggregate playlist of movie titles that accommodates the preferences of the detected users in the group. Finally, SocialAwareFlicks attempts to find a movie trailer file for each title on the playlist on the local filesystem, and begins playing the trailers on the screen. This application was integrated into a prototype framework for anonymous context-aware applications called Social-K, discussed

in section 7.

Chapter 4

Challenge: Scaling Anonymity

This section will discuss the challenge of scaling anonymity measurements to the size of large Socio-Digital systems. A major challenge to integrating anonymity into a socio-digital system is its negative impact on performance. Traditionally, anonymization required analyzing an entire data set and calculating optimal anonymity or perfect privacy is Π_2^p-Complete [43] [46]. One of the motivating factors in the design of q-Anon and AnonyGraph (discussed in sections 8 and 11 respectively) is that it allows global anonymity guarantees with only partial data set analysis. However, stronger anonymity guarantees require analysis of larger portions of the data set. If anonymity cannot scale to weaker/stronger gaurantees and/or larger data sets it will have limited applicability to socio-digital systems. This section discusses the primary challenges and metrics involved in scaling anonymity. It suggests a research agenda to design and implement algorithms that allow anonymity to scale to large data set sizes and privacy levels.

4.1 Memory Overhead

The first challenge in scaling anonymity will be efficient use of memory and storage resources. This section discusses how certain algorithms might be used to calculate anonymity for large data sets. Further, a set of research ideas are suggested that might lead to more efficient management of memory resources.

4.1.0.5 Algorithms

As discussed in section 8, q-Anon has been defined to take advantage of advanced logic minimization algorithms optimized to efficiently use time and space resources. However, optimal logic minimization is

hard problem which scales exponentials in time and space. For large and complex problems not even heuristic approximations such as Espresso, SIS, and BDS produce exponentially sub-optimal approximations. However, the two-level (SOP/POS) expressions created to represent the relationship between attributes values and users represent rather simple and "sparse" logic circuits in that most inputs are don't-cares and the desired output is also a two-level SOP expression. Preliminary results, discussed in later in section 8, are encouraging, demonstrating that q-Anon can be calculated for simple Socio-digital applications using Facebook for $q \leq 50$. However, these tests were done using the Quine-McCluskey algorithm with is over 50 years old and much less efficient than more advanced logic minimization techniques. The current q-Anon implementation used with the Whoville prototype is capable of calculating medium values of q in about second considering over 2.5 million different objects and nearly twice as many connections. The current q-Anon implementation is partially based on techniques developed by Petr Fiser at Czech Technical University. The current algorithm is modified from the algorithm described here [29].

4.1.0.6 Databases

In order for anonymity to scale to very large socio-digital systems (millions/billions+ of users), efficient databases will need to be integrated. In particular those that perform very well on the types of searches used to measure anonymity. The author has used SimpleDB, MySQL, Derby, and HSQLDB to store and calculate large social data sets. However, while not yet tested, the author has implemented the Anony-Graph model as a Triple representation like those used by TripleStore databases (those databases used to store semantic web relationships). TripleStore databases store all information as sets of "triples" or subject-predicate-object tuples. TripleStore databases are optimized for querying these simple tuples, which has allowed TripleStore databases to scale very well [34]. While current results have shown that reasonable anonymity levels can be achieved with those databases listed above, the applicability of anonymity would be significantly increased if it could be defined and implemented in terms of semantic relationships on a database designed for scaling and connecting large data throughout the world.

4.2 Response Time

Many socio-digital systems require real-time interaction with users. As such, to be practical anonymity must be computed within expected response time. This time will depend on the application and allow certain types of applications to provide stronger anonymity guarantees than others. Also, some users may value their privacy more than others and be willing to bare longer response times in exchange for greater anonymity. This section discusses how user studies and real-world application tests will allow the tailoring of logic minimization algorithms to support user demands in real-time.

Many logic-minimization algorithms are able to approximate an optimal minimization within certain bounds in much less time than it would take to calculate the optimal minimization. As mentioned, some users and applications may have hard requirements for anonymity while others may prefer anonymity, but not at all costs, allowing for "best-effort" anonymity approximations with certain bounds. To understand how different anonymity requirements affect application performance, different applications were implemented on top of the Whoville prototype. The author's experience with these applications is discussed in section 12.

4.3 Identifying Sufficient Problem Sizes

One method currently used to minimize the resources used in the Whoville prototype is that of bounding the input problem (number of objects considered). If a small value level of anonymity is required, the anonymity software need not find the maximum (or optimal) anonymity measure, rather it may only include a large enough subset of the data to prove that anonymity must be greater than that required by the application or user. Given this observation, Whoville uses simple heuristics to determine the size of a sufficient input problem or expression which allows anonymity to be approximated quickly before actually measuring a lower bound on anonymity. This approach reduces the time involved to calculate small anonymity values and significantly reduces calculation of large anonymity values.

4.4 Current Algorithmic Bounds and Real-world experience

A minimization method adapted from an algorithm developed at Czech Technical University has been shown to scale linearly for two-level logic minimization problems in which there are many terms (millions) and far fewer literals[29]. This algorithm is being evaluated with the Whoville prototype for calculating q. It was originally chosen due to the fact that when there are n users and m attribute values (or relationships/connections), the number of terms to be minimized is bounded by O($n^{\frac{n}{4}}$) when $m \geq \sqrt{n}$, otherwise when $m < \sqrt{n}$ the number of terms is bounded by O($(\frac{n}{m})^m$). While the number of terms has never been observed to scale anywhere near the maximum bound, it has been observed that as anonymity increases so too does the ratio between terms to be simplified and the number of users (literals) in those terms.

Chapter 5

Challenge: Balancing Anonymity & Information Utility

Further, this thesis will investigate how an anonymity systems chooses which data to anonymize and how to anonymize that data. Some information may be so unique to its associated user that it cannot be released without generalization under any anonymity guarantee. Whenever information is suppressed or generalized it reduces the amount of unique information released. This section discusses a research agenda focused around the trade-off between anonymity and information gain and the results of this investigation will be vital to designing a data filter component that preserves as much utility as possible when anonymizing data.

This section discusses the anonymity-information trade off as it relates to many traditional anonymity metrics such as K-Anonymity and l-diversity, as well as how the group anonymity metric (q-Anon), discussed in section 8, is affected by this trade off. Many of the results and observations in this section refer to the q-Anon metric in section 8, however one need not fully understand the algorithm to read this section, simply knowing that larger values of q imply greater anonymity and may be at odds with the utility of the information being anonymized.

5.1 The Anonymity-Information Trade-Off

To increase the anonymity of a query response certain values in the response are suppressed or modified in some way. In some cases the values may be modified without losing much information. For instance, section 8.4.2 discusses how user input movie strings are sanitized using the Internet Movie DataBase (IMDB). In this case, $\geq 95\%$ of strings were correctly sanitized, meaning that a standard string for the movie

referred to by the user input string was found and used in place of the user input string. This preserved most of the movie data while significantly increasing the anonymity of most query responses. However, depending on the data type and application anonymization often reduces the utility of the data significantly.

For instance, figure 8.2 the location of users is generalized to granularity of city. However, an socio-digital system which suggests nearby locations to eat. The application will need to know the location of a user within a few blocks such that it doesn't suggest places to eat that are on the other side of the city. This is a case of anonymization defeating the purpose of an application. In other cases, such as generalizing or perturbing user preference values, the anonymization may result in a more general but wrong preference being associated with a user. This may happen due to the anonymization assuming an incorrect hierarchy to the values or simply due to the fact that user has abnormal preferences (for instance a user that likes "The Beatles" does not necessarily prefer "Arthropods", an example of wrong semantic hierarchy).

Furthermore, the skewness attack discussed in regards to t-closeness [40], directly confronts this trade-off between anonymity and information gain by bounding the difference between the distribution of any equivalence class and the distribution of the entire data set. This effectively bounds the extent to which any subset of data may exhibit unique patterns, placing anonymity and information gain directly in opposition to one another. While the lack of direct mappings between users and their attribute values allows q-Anon to avoid such a direct opposition between anonymity and information gain there is still a trade-off that must take place during anonymization.

5.2 Information Gain & Entropy

One approach to balancing anonymity and information gain is to optimize some information gain metric (such as Inverse Frequency Count) within the anonymity requirements. The information gain measured at each attribute value could be used to decide which attributes are suppressed or generalized and to what extent.

For instance, consider the case that two values are equally distributed throughout a specified equivalence class, however one of them is equally distributed in the equivalence class and full data set while the other is rather unique to the equivalence class but distributed well enough throughout it to meet anonymity

requirements. Choosing to remove the more common value rather than the unique value would allow more unique information to be released with the same q-Anon measure. This would also release the fact that those in the equivalence class had a much higher probability of association with the data value than those not in the equivalence class - however, the probability of that association would still be bounded by the calculated q value (e.g., with $q = 10$ and the value only applying to $\frac{1}{100}$ users in the entire data set, the users in the equivalence class may be 10 times more likely to be associated with the value while still only having a $\frac{1}{10}$ probability of being associated with it). The last example of users being 10 times more likely, but still only $\frac{1}{10}$ probable, to be associated with a value, brings up an interesting point. Any anonymity measure must decide the goal of anonymization. q-Anon measures "absolute" anonymity and not "relative" anonymity. In the example above, releasing the unique value possibly reduces the anonymity of those in the equivalence class by 10 times, however it maintains an absolute bound on posterior probability of association at $\frac{1}{10}$. This distinction is often missed when comparing anonymity definitions that measure absolute anonymity (K-anonymity, p-diversity) and those that measure relative anonymity (t-closeness).

5.3 Inverse-Frequency or TF-IDF

TF-IDF or term frequencyinverse document frequency is a weight or metric commonly used for informatics. It is a statistical measure of how important a word is compared to a set of other possible values for any given data type in an information system.The importance increases as the word is used more often in a particular case than it is in general across the system. In other words, rarer words or values are more important than common values. This metric is used by the prototype Whoville as a baseline or default metric for all anonymization algorithms. It considers how often a particular object is connected across all of Facebook and makes it important or weight equal to the reciprocal of its node degree. This metric works well in general, however certain applications may be interested in particular fields or values and the Whoville prototype supports specifying these fields and values on a per-application basis.

5.4 Application Specific Metrics

While general metrics for measuring information gain may increase the utility of anonymized data in general, it may not increase the utility of the data for all applications. Due to the unique nature of socio-digital applications this project will develop an API through which applications may specify their own utility metric. This will require a simple specification (probably in XML) with which the application can translate its intentions into a description of the data in which it is most interested. These descriptions could be registered per application or per request with the Whoville prototype (section 11.

5.5 Suggested Future Work: Privacy Negotiation

A third option for optimizing the utility of query response data is to allow for negotiation between the anonymous API and the application and/or user or even negotiation between the user and the application.

If it is found that the user's privacy requirements are not compatible with the information metrics and requirements of the application then the anonymous API prototype may notify the user that their privacy requirements are not functional with the particular application. The system could also communicate what privacy levels are compatible with the application allowing the user to evaluate whether or not such a compromise is reasonable.

However, a more advanced method might be to develop a method by which q-Anon could be used to quantify the relationship (or difference) between the application's information requirements and the user's privacy expectations. Using this measure of privacy difference the user may be able to negotiate different service levels with the socio-digital application in return for their private information. Already, many people harness the value of their personal information to receive services such as e-mail, social networking, or multimedia. Privacy is monetized from the individual to the application or business, however, it is rare that this exchange flows in both directions. This is because people do not have the tools to negotiate their privacy to **AND** from the applications.

Due to the common nature of user-application interaction, privacy concerns tend to take a discrete boolean form: either a set of information is given away or not. If users had the ability to specify different

levels of privacy or anonymity using q-Anon application-user interaction could be optimized in real-time for the particular context. This will require a trusted or secure interface for users to specify privacy rules in real-time for applications which wish to access their data through the anonymous API.

Chapter 6

Challenge: Integrating Trust & Knowledge Models

This section touches on how different practical issues of anonymous data release relate to calculating anonymity. This thesis does not attempt to provide solutions to these issues so much as provide a perspective on how these issues relate to anonymization and how existing solutions can be integrated into an anonymous data release system. In particular, this section explains why ignoring these issues when designing anonymity solutions may render anonymity metrics or algorithms impractical in the real-world due to how they conflict with the other information metrics discussed in this section. This thesis suggests a simple set of questions that a privacy engineer should ask when considering their application.

6.1 Defining Trust as it Relates to Anonymity

This section defines user-application interaction to more clearly understand how a users interaction with one application affects their overall anonymity and potential for anonymity in the future. In particular, this section presents a few basic questions which allow the privacy engineer to decide how knowledge should be managed (or shared) within the anonymity model. These questions relate to trust, however not in the intuitive personal sense. These questions allow one to consider how the release of personal information to one entity may reduce the utility of anonymous data released to another entity. Often, such questions are confusing, for example: could sharing private data with your family lead to the de-anonymization of your medical records by some untrusted third-party? In order to answer such a question, we suggest that you start by asking to much simpler questions:

(1) Is the entity you are giving information to “trusted”?

(2) Is the exchange of information "anonymous"?

First we must define what we mean by "trusted" and "anonymous", we define them as follows:

> **Trusted:** An entity is *trusted* if you can bound or predict who will have access to the information you give the entity, or in other words you know whether or not any other entity received your information as a result of exchanging data with the trusted entity.

It is important to note that applying this definition recursively means that trusted entities can only exchange information with trusted entities. Also, this definition does not mean that trusted entities are trusted not to violate ones privacy or perform re-identification attacks, it simply means that they share ones data in a completely predictable way and only with trusted entities.

> **Anonymous:** An information exchange is *anonymous* if no combination of the information exchanged is unique and could have been accessed non-anonymously by an entity that may have access to the information being exchanged. In other words, the data exchange is NOT anonymous if any entity may have access to the anonymous data and other non-anonymous data that contain some uniquely matching sets of data. For simplicity we will refer to non-anonymous data exchange as being "personal".

6.2 How Lack of Trust Affects Anonymity

Now we can reexamine the assumption from section 3 that all social network data should be assumed public (but treated private). Leaving the question of whether or not social network sites like Facebook can be considered trusted entities for later we will focus on Facebook users and consider whether they could be trusted. We assume that all information on Facebook is shared with some other user otherwise its existence on Facebook is mostly meaningless to this discussion.

Consider that in order to "trust" a Facebook friend, a user would have to trust all their friends as well, following this logic to its end means that trusting one user requires trusting all Facebook users connected to them through the social graph. While there may exist a piece of data that is only shared within a closed social graph of trusted users, the fact that Facebook does not provide any guarantee as to who has access to ones data should be sufficient to imply that all Facebook users cannot be considered trusted by our definition.

If no Facebook user is trusted then all data shared on Facebook is part of a personal-untrusted data exchange. It is in this sense that we say social network data should be assumed public and must be

considered when evaluating the anonymity of ANY data exchange. Furthermore, any other data that is associated with Facebook data may itself become personal if the associated Facebook data is not anonymized. This is one reason why the example of the "Anonymous" STD screening would require anonymization of the social graph in order to maintain the anonymity of the patient.

So if we can not trust any of our friends or even Facebook, **who can we trust and what benefit is received from this trust?** Consider that any anonymous source of Facebook data must maintain a record of what data it has released (or what queries it has answered) to protect against an adversary using the combined responses to multiple queries to perform a re-identification attack. However, if anonymous information is shared with a trusted entity or set of trusted entities, this information need only be considered when measuring the anonymity of an exchange that may reach one of those trusted entities. This allows for disjoint sets of trusted entities to receive separate sets of information from the same anonymous data source that may not be anonymous when taken together but are anonymous when separate. This allows for significantly greater anonymous information sharing with trusted entities, especially if the anonymous data source releases little or no information to untrusted entities.

Based on these observations a simple knowledge/trust-model was designed for the Whoville prototype. This knowledge/trust model is discussed in section 9.

Chapter 7

Solution: Social-K, K-Anonymity for Social Data

This section discusses the author's experience integrating anonymity into a socio-digital system. The system used a precursor to PP-anonymity much more like traditionally K-anonymity modified to measure query responses rather than entire data sets, however the project architecture presents how an anonymity layer may be integrated into an anonymous socio-digital system. While this system did not use a very mature or complete anonymity model it did show that anonymity could feasibly be integrated into a socio-digital system. For this project SocialAwareFlicks, discussed in section 3.3.0.4, was used as a test application.

7.1 Social-K Architecture

The system was called Social-K and tried to adapt the traditional K-anonymity definition to a social network API. Figure 7.2 depicts Social-K processing a query for a user's Facebook profile data. This figure also shows how each of the Social-K components interact in fulfilling the query. Each component is discussed briefly as it relates to supporting an anonymous socio-digital system. The system generally consisted of an identity management service tasked with sharing presence and identifiers to support personal queries without giving out the users true identity. Four other components in the system comprise the core of Social-K and cumulatively responsible for organizing the social network data, processing the data as necessary for measuring anonymity, measuring the anonymity, and finally for modifying the data set to meet the system's anonymity requirements.

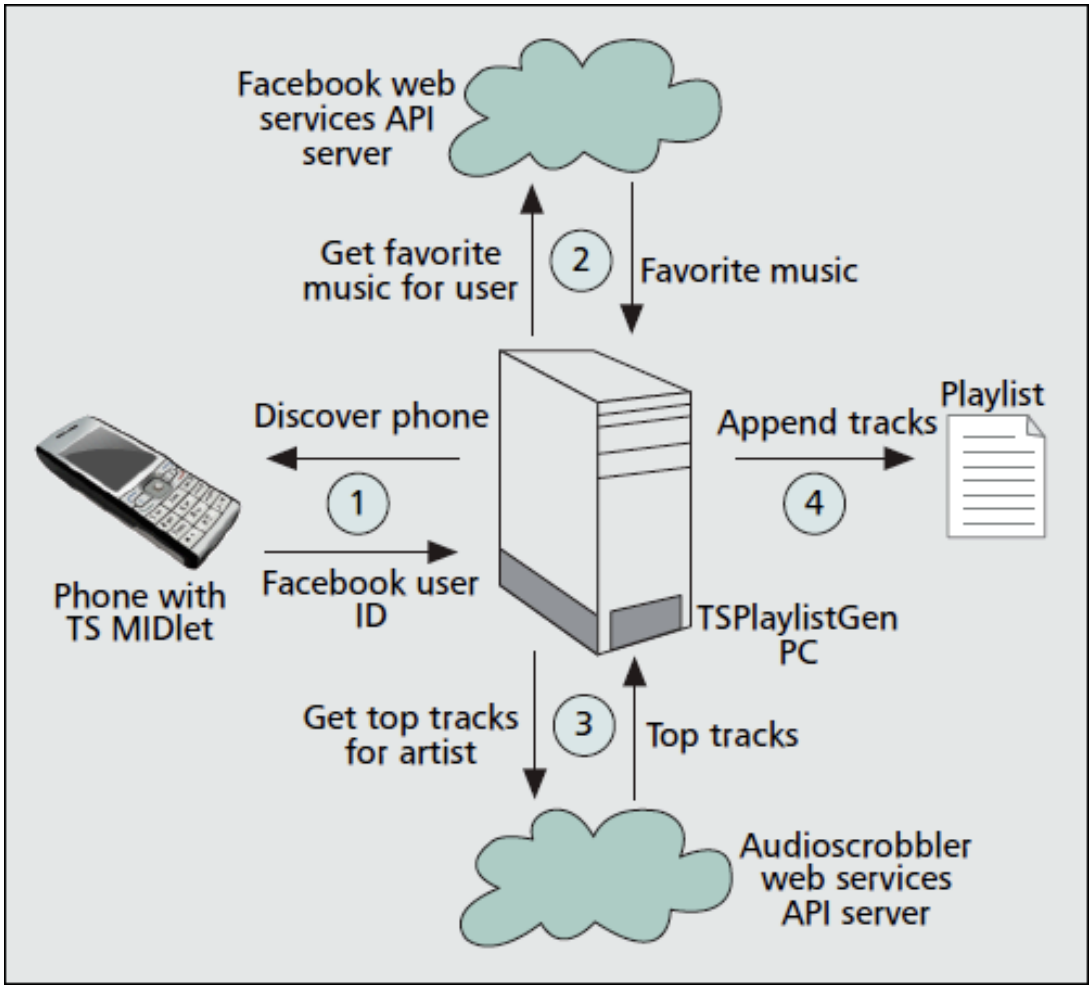


Figure 7.1: WhozThat Basic Architecture.

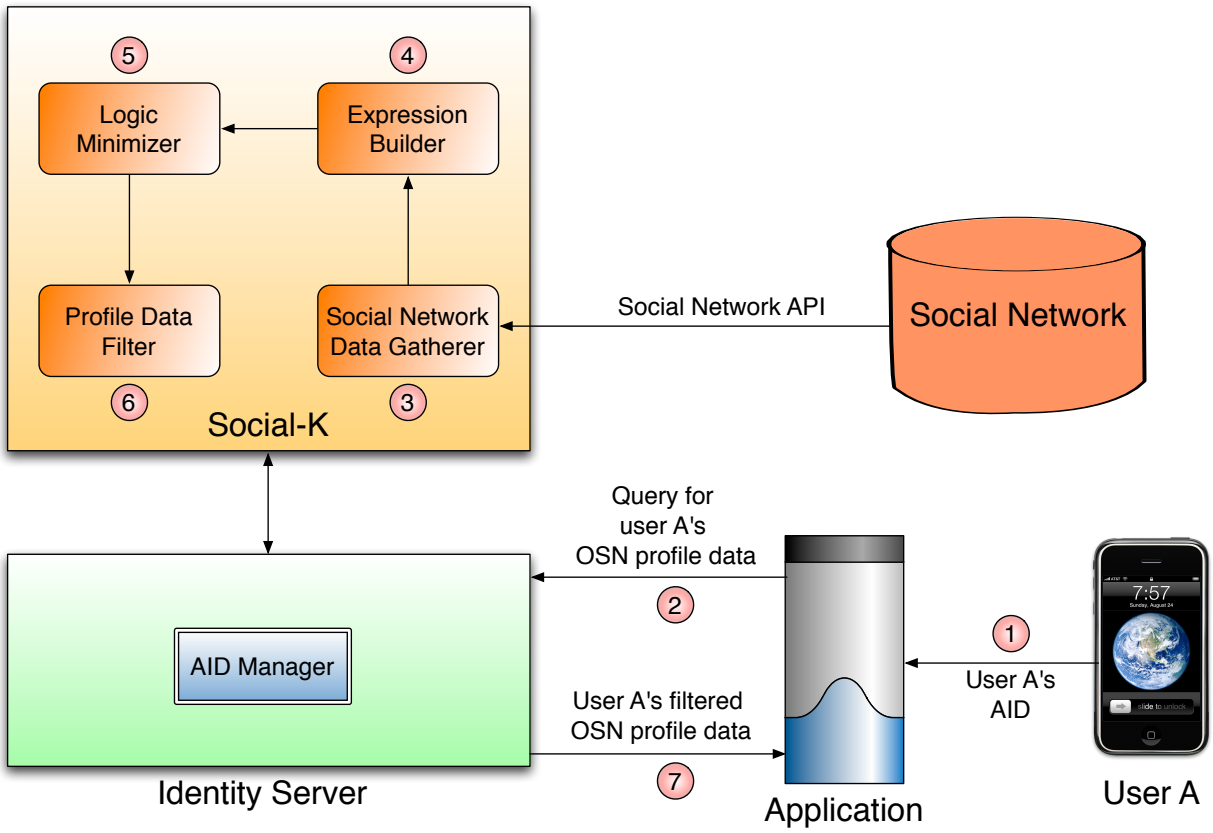


Figure 7.2: Social-K Architecture.

7.1.1 Identity Server

The Identity Server (IS) provides identity management services for social network application users. The IS generates and distributes anonymous identifiers (AIDs) to users through requests. An AID is a nonce which may be included with a query to anonymously identify the user (or users) associated with the query. A user's AID is mapped to the user's social network ID by the database on the IS.

7.1.2 Social Network Data Gatherer

After the IS receives a request for a user's social network profile data Social-K begins gathering pertinent user profile data for a large set of users. It was convenient to start with the user's friends on the social network since they were easiest to access through the Facebook API.

7.1.3 Expression Builder

These users and their relationships to their data are represented as a boolean expression as discussed in section 8.3.0.1. The output is an Sum-of-Products (SOP) expression containing every possible unique combination of users which could fully account for the data being released.

7.1.4 Logic Minimizer

Next, we applied logic minimization algorithms to simplify the Boolean expression provided by the Expression Builder. Several well known logic minimization algorithms exist, including ESPRESSO [51] and Quine-McCluskey [47]. For the initial implementation we used Quine-McCluskey.

7.1.5 Profile Data Filter

If the anonymity measurement implied that further anonymization was needed then the Profile Data Filter suppressed the least common values in the data set and remeasured anonymity. In the case that many values were equally common among the data set the value which had the highest inverse frequency count across the entire data set (not just those value being released) was chosen to maximize information gain.

This very simple approach to filtering social network data for anonymity worked but was inefficient and did not allow applications to specify which data was most valuable. However, it did prove that such an anonymization approach was feasible and could scale to social networks the size of most Facebook users' friend networks.

7.2 Preliminary Results: Scalability Testing with Social-K

To understand how anonymity calculations might scale with the size of a socio-digital system, the Social-K system (discussed in seciton 7) was tested with real Facebook data. All performance testing was performed using a Macbook notebook running Mac OS X 10.5, with a 2.0 GHz Core2Duo processor, 2 GB of RAM, and a university-provided high-speed Internet connection.

Our metrics were gathered from tests including 222 Facebook users and and focused on evaluating seven favorite movie values listed on their Facebook profiles. In a study conducted by Ellison et al. [25], the mean number of Facebook friends reported by the study participants was between 150 and 200. Therefore, this is a somewhat reasonable representation of a typical or average Facebook user, at least in terms of size.

The evaluation was conducted by submitting a query to the Social-K Identity Server requesting the list of favorite movies in the Facebook profile for user A. Social-K begins by processing the query, first gathering the favorite movie lists for each of the users then using the Expression Builder to construct a Boolean expression representing the relationship between user A's friends and their favorite movies.

Figure 7.3 shows how the time required to minimize the unsimplified Boolean expression in the Social-K Logic Minimizer component varied with the number of terms in the unsimplified Boolean expression. This plot shows that the Logic Minimizer component scales reasonably well for unsimplified Boolean expressions containing up to about 450 terms. It is expected that 450-term Boolean expressions will account for many typical usage scenarios, although this will vary based on the number of the user's favorite movies that match with his friends' favorite movies. There is a nonlinear relationship between the number of terms in the unsimplified Boolean expression and the number of terms in the simplified expression.

Figure 7.4 shows how the time required to minimize the unsimplified Boolean expression in the Social-K Logic Minimizer component varies with the number of terms in the simplified Boolean expres-

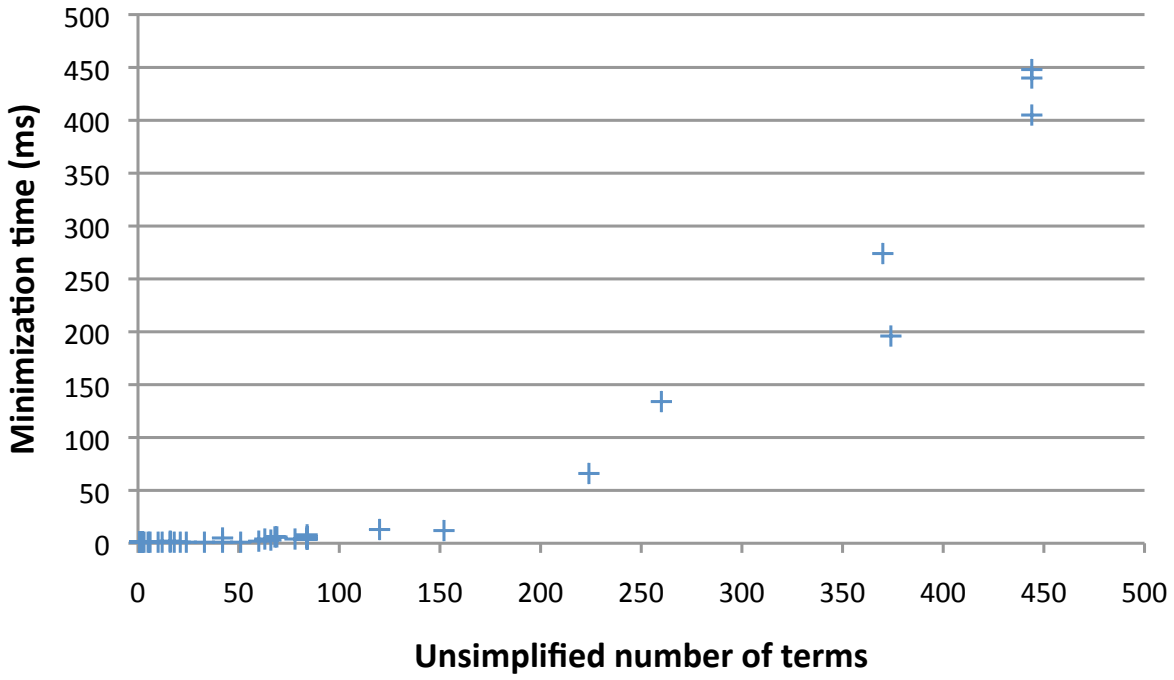


Figure 7.3: Minimization Scalability in Terms of Unsimplified Expression Size.

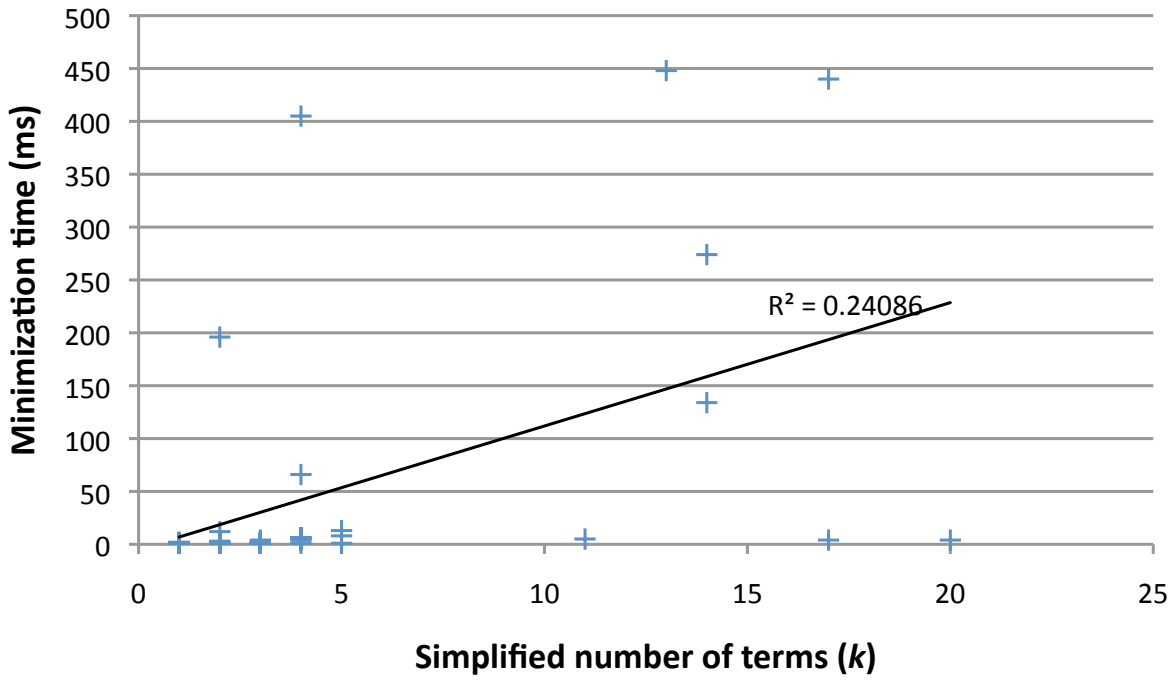


Figure 7.4: Minimization Scalability in Terms of Simplified Expression Size.

Component	**Mean run time (ms)**	**Run time standard error (ms)**
Social network data gatherer	852	145
Expression builder	11	1.7
Logic minimizer	297	8.29
Social-K total	1377	134.7

Figure 7.5: Social-K Component Performance

sion. We see from this plot that there is minimal correlation between the number of terms in the simplified expression and the time required to minimize the expression. We can conclude from figures 7.3 and 7.4 that the time to minimize the unsimplified Boolean expression is correlated with the size of the input to the Social-K Logic Minimizer component (the number of terms in the unsimplified expression), and is not correlated strongly with the size of the output (the number of terms in the simplified expression). This is the expected behavior for a logic minimizer and means that the data can be chosen before minimization to scale well not necessarily dependent on the level of anonymity desired by the system.

Table 7.5 shows the run times of each component of Social-K for the following example:

- Number of friends: 222
- Number of movie matches: 13
- Number of friend matches: 7
- Number of terms in unsimplified expression: 339
- Number of terms in simplified expression: 7

The Social-K total run time in table 7.5 is the time for our system to return anonymized favorite movie preferences for a user. In our tests, the run time of the Social Network Data Gatherer component dominates the run time of Social-K, since this component spends most of its time downloading data from Facebook. We expect that running Social-K with local access to Facebook's user database would significantly reduce the run time of this component. However, the current average total Social-K run time of 1377 ms for our

tests provides acceptable performance for applications such as SocialAwareFlicks.

Chapter 8

Solution: q-Anon, An Anonymity Metric for Group Queries

This section discusses q-Anon, a new measure for anonymity. In particular q-Anon measures the anonymity of responses to API calls. q-Anon measures the posterior probability that a private piece of data is associated with its correct user conditioned on a response to an API call. This section will describe the re-identification problem, as it applies to social network APIs, in more detail. Also, an example will be given to clarify the nature of this anonymity measure.

8.1 The Unique Anonymity Problem in Social Networks

This section discusses the anonymity (or re-identification) problem as it applies to q-Anon. The problem involves a trusted data collector who wishes to provide an interface through which applications may query user data without compromising the user's privacy beyond some threshold. Generally, a query is considered admissible if its released information does not result in any previously unknown mapping between data and identity having an implied posterior possibility greater than some threshold. We will refer to anonymous interactive interfaces which provide guarantees on such posterior probability as being "q-Anon". This section will also define how the interactive interface functions and the assumed background knowledge of the adversary.

q-Anon Definition: q-Anon focuses on measuring the ambiguity of released data in the face of a re-identification attack. Privacy is measured in terms of q, for which larger values represent greater ambiguity (privacy). q is measured by finding all unique user groups which could have accounted for the released data and then finding the largest fraction of those groups which include any one user. q is defined as the reciprocal

of this fraction. An example of this measurement is given in figure 8.3 and a step-by-step description of measuring q is given in section 8.3.

Adversary Definition: Furthermore, we assume that the adversary may have access to any entry in the data set. Therefore, in spite of its general impossibility for statistical databases [21] we will accept the intention of Dalenius' desideratum for statistical databases: "that nothing about an individual should be learnable from the database that cannot be learned without access to the database." We define learned in terms of posterior probability of existence.

Interactive Interface Definition: For the purposes of this paper, we define an interactive interface, using traditional anonymity terms, as an interface which allows applications to specify an equivalence class through a query along with attributes of interest. The interface returns the values of the attributes as one field or array not specifying any mapping between table entries and attribute values. The interface may return zero or more values from each entry in the equivalence class.

Privacy Compromise Definition: A privacy compromise has occurred if q-Anon, with a given q value, is violated by a response to some query.

It should be noted that if a query specifies an equivalence class which has fewer than q entries, the adversary may know that at least some of the released data is mapped to fewer than q identities, hence q-Anon implicitly requires all equivalence classes specified by a query to be at least size q. This requirement relates to the original K-anonymity definition.

Since the query response from the interactive interface does not include mappings between table entries and attribute values, the set of values may map to many different groups of individuals, each of which may fully account for all values in the query response. The next section discusses how the existence of these different groups, all of which account for a data release, may be measured to infer the probability that a set of private data is mapped to a specific individual.

8.2 Measuring Anonymity

This section will discuss a practical approach to calculating bounds on knowledge learned by an adversary that may know every data value in a data set except one (the one we are theoretically calculating

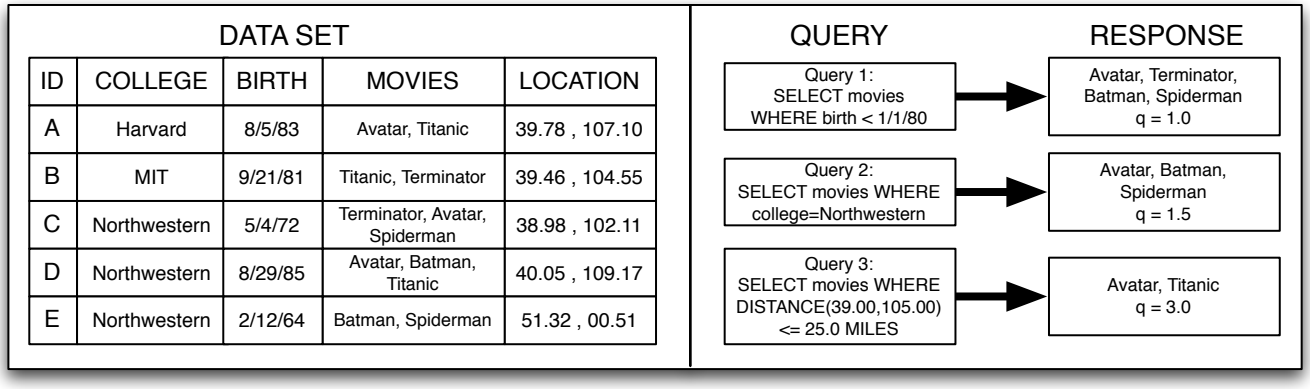


Figure 8.1: Example q values for different queries

the bound for). By "bounding knowledge" we mean not releasing information which implies a posterior probability of association between an an individual and their private data greater than $\frac{1}{q}$. In calculating this bound for a particular query response we say we are measuring the privacy of the query response. Hence, the larger the q value the greater the q-Anon.

This section will explain ways to measure q, starting from the perspective of the adversary. A possible attack will be described which will imply a way that q can be calculated. In this way, the value of q will be concretely connected with its privacy implications from the beginning. Each example will be connected with the example data set, queries, and responses presented in figure 8.1. For comparison, the example of the data set is also presented in figure 8.2 anonymized with traditional K-anonymity ($k = 2$). Note that optimal k-anonymization would depend on knowing the application intentions and attributes of interest beforehand. While this is not possible, the example data set is k-anonymized to retain as much useful data as possible for comparison. Furthermore, it should be noted that this data set cannot contain diversity (as discussed in section 3) and as such is trivially vulnerable to an attribute disclosure attack.

Rare Value Attack: Many strings found in social network data fields are very unique. For example, sometimes a user who likes the movie Avatar may include the string "Avatar totally rocks!!!" in their movie list, rather than simply including the string "Avatar". This string may uniquely map to the user and should never be released from a social network's anonymous interface. Assume the string does uniquely map to this user and also say an attacker knows this and wants to find the location of the user. The attacker also knows that the anonymous interface only returns queries that are admissible under q-Anon with $q = 6$. The attacker could create 5 fake users at a particular location and list "Avatar totally rocks!!!" in their movie list. Then the attacker could query for the movie lists of users within a geographic area including the location of his five users. If the query is responded to with the string "Avatar totally rocks!!!" the attacker would know that the victim user was also in that geographic area, since there must have been at least 6 users in that area with the unique movie string or else the query would not have included it in the response.

This presents a rather simple way to measure privacy. The anonymous interface could simply measure the privacy of a query response by taking the least common value from the query response and counting the

number of individuals in the equivalence class[1] that are associated with that value. We will refer to this value as the Least Common Value count (LCV count). As can be seen in Query 1 in figure 8.1, the response contains four movies, one of which (Batman) is only listed by one individual in the specified equivalence class (those born before 1980). Hence this response appropriately lists $q = 1$. However, the next example will show that this measure of privacy is only an upper bound on privacy knowledge and misrepresents what the attacker may logically deduce from a query response.

Logical Exclusion Attack: Consider the response to Query 2 in figure 8.1. Each of the three movies in the response has at least two individuals in whose movie lists they are included and might be considered q-Anon with $q = 2$. However this is not necessarily the correct bound on posterior probability of associating the set of movies with an individual in the data set. The attacker could logically deduce that one of three possibilities must exist, the group correctly accounting for the data must contain either persons C and D, D and E, or C and E. These three groups represent the minimal combinations of which at least one must exist within any group which fully accounts for the data released by the query. This could be expressed as the boolean expression $(CD + DE + CE)$. Given that each individual is in two of the three groups the attacker can assume that there is a $\frac{2}{3}$ (or 66%) possibility that one of those individuals is correctly associated with the data, hence $q = 1.5$.

This presents a stronger measure of privacy but is more complicated to measure. To measure privacy in this way requires that one find the minimal set of groups required to fully account for the data being released. Section 8.3 discusses how this can be done by representing the data values and their mapping to users as a boolean expression and finding the prime implicants using logic minimization. Representing social network data in this manner was originally suggested and discussed for use in a mobile social network application [9].

8.3 Real-Time Anonymization

This section discusses the practical steps involved in measuring the q value for q-Anon. The steps generally consist of **(1)** Building a logical Sum-of-Products (SOP) expression from the data to be released

[1] remember the equivalence class is specified by the query parameters

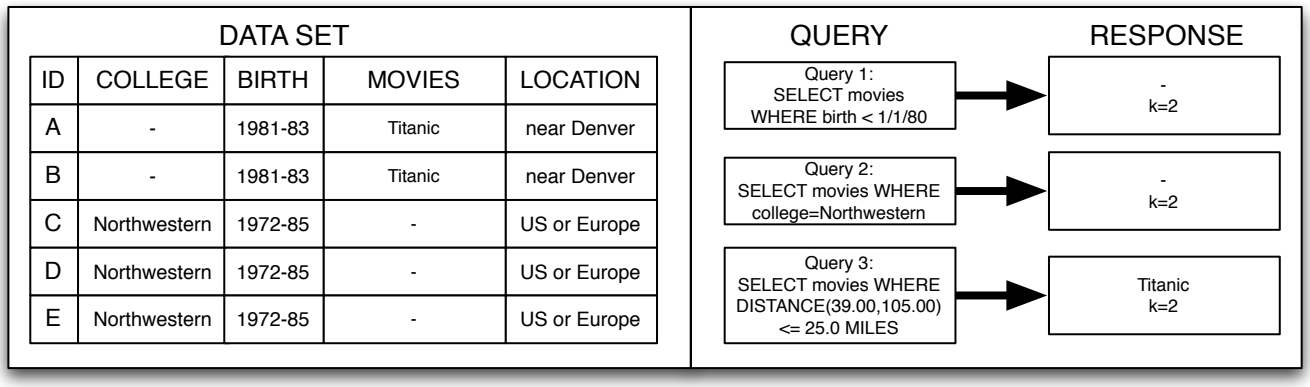


Figure 8.2: Example data set K-anonymized with $k = 2$

by a query. **(2)** Minimizing the SOP expression to find the essential prime implicants. **(3)** Calculating q which is the number of occurrences of the most common literal (variable) divided by the number of terms (clauses) in the minimized expression. Each of these steps will be explained and then the section will finish with a discussion of how this approach would perform in the real-world.

Remember that a query specifies an equivalence class (e.g., "men under 25 in Denver") and an attribute of interest within that class (e.g., "Movies"). The query response consists of a set of values from the attribute of interest within the equivalence class. It is this set of values in the query response and its relationship to the equivalence class that is measured to calculate q.

8.3.0.1 Step 1: Building an SOP Expression

This section describes how a product-of-sums (POS) expression is created from an equivalence class specifying a set of users and their data. The POS expression can then be converted to a Sum-of-Products (SOP) expression by De Morgan's law or more efficient method.

Given an equivalence class E consisting of n user entries and m attributes for which S is the set of all attribute values. We define a set $E_i \in E$ as the set of users which map to a certain value $S_i \in S$. A POS expression is formed from E_i whereby every set $E_i \in E$ is mapped into a sum term (clause) in which every user $e \in E_i$ is a literal (variable). This POS expression is then converted to an SOP expression for minimization or minimized directly, understanding that the output must be a two-level SOP expression.

8.3.0.2 Step 2: Minimizing the Expression

The SOP expression is then minimized using a logic minimization method which finds the essential prime implicants of the expression. A classic algorithm for doing this is the Quine-McCluskey algorithm, developed in 1950's. This algorithm was used for initial evaluation of a related problem [9] and those results are discussed in section 8.4 along with discussion of how more recent developments in logic minimization may allow this step to scale linearly with the number of individuals and data items.

8.3.0.3 Step 3: Calculating q

Given a minimized SOP expression with n product terms the maximum number of times that any literal occurs (m) is divided by n. Hence $q = \frac{n}{m}$. Because the SOP expression is expressed partially with only the on-set - all product terms contain at most one occurrence of any literal. Therefore $n \geq m$ meaning that $q \geq 1$ and since all literals must occur at least once $q \leq n$.

8.3.0.4 Example

To help explain the process of measuring q-Anon we will reconsider Query 3 from figure 8.1, however this time q will be calculated for a different query response. This example will be explained using figure 8.3. The query requests the "MOVIE" values for those individuals within a geographic area specified in latitude and longitude. This area includes individuals {A,B,C,D}. In this case, q is being calculated for the response {Avatar, Titanic, Terminator}.

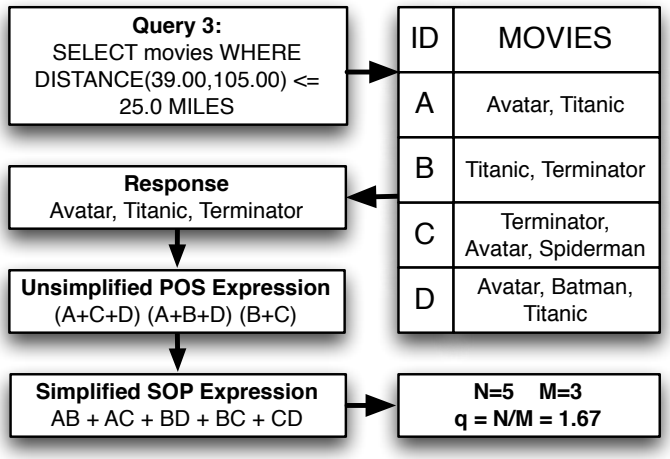


Figure 8.3: Example of using logic minimization to calculate q

As can be seen in figure 8.3: Avatar is mapped to {A,C,D}, Titanic to {A,B,D} and Terminator to {B,C}. This results in 9 possible groups which could possibly account for the entire response set, these groups are represented by the unsimplified expression in figure 8.3. Minimizing the unsimplified expression results in an expression with 5 product terms. Since there are 5 product terms in the unsimplified expression and no literal occurs more than 3 times $n = 5$ and $m = 3$, hence $\frac{n}{m} = q = 1.667$.

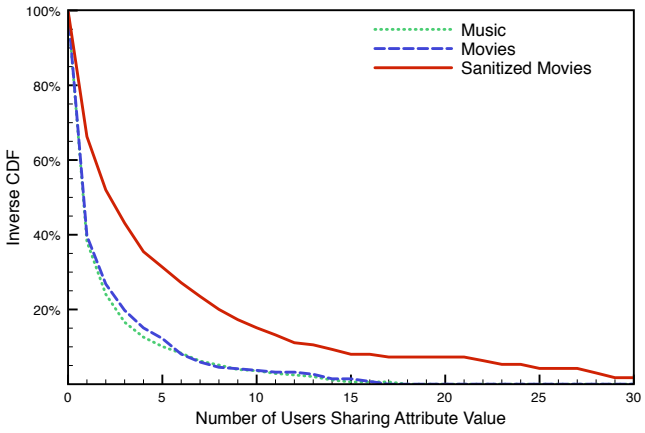


Figure 8.4: Distribution of Attribute Values.

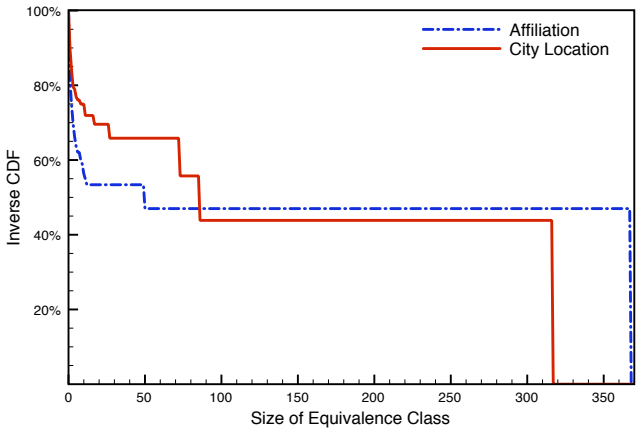


Figure 8.5: Equivalence Class Distribution.

8.4 Preliminary Results: q-Anon Feasibility Study

Once it had been shown that an anonymous API was feasible from an implementation standpoint (section 7) the data itself needed to be studied to gain a basic understanding of whether the data could effectively be anonymized, including understanding to what extent the data could be anonymized and how useful the data was when anonymized. This section presents the results of small study of Facebook data. The study evaluates a set of Facebook data including over 700 users' city location, network affiliation, music, and movie preferences. The users in data set are all friends with the same user and could be considered an equivalence class specified by this friendship. As such, the data samples are not random but may be considered realistic for socio-digital systems in which social connections are natural aggregators. The distributions of these data are presented to understand the practical size of some equivalence classes and the distribution of their attribute data. These values are discussed in terms of q and what values of q are actually practical for a social network such as Facebook. Finally, a basic prototype is presented to show the feasibility of measuring q for social network data applications with discussion of how advanced logic minimization algorithms might scale in relation to social networks the size of Facebook.

8.4.1 Size of Equivalence Classes

In an interactive interface, equivalence classes may be specified with conditionals using sql style interfaces or particular API calls. Examples evaluated in this section include selecting those users in a particular city or users affiliated with a particular network. The size of the equivalence class is a bound on

what q values may be calculated for its data. The distribution of city locations and network affiliations in the test data is shown in figure 8.5. A few locations and affiliations dominate the distribution with nearly half of the users sharing either a location or network affiliation, a few other values account for 5-10% of all values. Within a social application such a distribution should not be surprising considering that proximity and affiliation are the basis for many social connections. However, what can be noted is that about 20-40% of all values considered specify equivalence classes containing about 1% of the overall users. These smaller equivalence classes consist of only 5-10 users and may not be able to support q-Anon queries with q values greater than 3 or 4. As such, a q-Anon interface with $q \geq 10$ would not release information to queries specifying these smaller equivalence classes.

8.4.2 Distribution of Attribute Values

However, a large equivalence does not itself guarantee any anonymity as it may not contain significant overlap in the attributes for which the query is interested. For instance, the music and movie preferences listed by the users in the test data contain significant diversity as seen in figure 8.4. While many of the users may indeed list the same movies and music artists among their favorite they often will not use matching strings to refer to same artist or movie. For instance users may list "Lord of the Rings", "LOTR", "The Rings Trilogy", or even "I heart LOTR". This reduces the usefulness of the data and disallows release of over 95% of music and movie data values under q-Anon with $q \geq 10$. Facebook is currently moving toward standardizing these values (probably to increase their worth to Facebook) through a new feature called connections in which the user "likes" something (e.g., thumbs up, thumbs down) resulting in a canonical string for each artist or movie. To understand how this data might be distributed the movie values from the data set were partially sanitized by using google.com to search imdb.com for the un-sanitized string value and then replacing the value with the movie title of the first result. This approach found the correct movie for over 95% of the movie values. When sanitized, over four times as many movie values were able to be released under q-Anon with a $q = 10$ and the same amount of data released from the un-sanitized movie values with $q = 10$ could now be released with $q = 27$, significantly increasing anonymity and the utility of the data. Considering that these values of q were calculated over a small subset of Facebook data (less than

.001% of Facebook users) it is possible that much larger values of q could be calculated considering only 1% of Facebook users. Furthermore, since q generally grows with the number of users being evaluated, the number of users which need to be evaluated could be bounded given a target q values.

Chapter 9

Solution: A Simple Trust Model for Anonymity

Using the two questions discussed in section 6 we propose to specify using the four different possible ways of collectively answering the two questions. Remember, the two important trust questions related to anonymity are:

(1) Is the entity you are giving information to "trusted"?

(2) Is the exchange of information "anonymous"?

Based on these questions, there are four possible approaches to handling trust and knowledge in an anonymous socio-digital ifnormation exchange: trusted-anonymous, trusted-personal, untrusted-anonymous, and untrusted-personal. This section will discuss the different assumptions that can be made about each of these information exchange models. Also, we will discuss how information exchange within one model affects information exchange under another model. Below we will summarize the important things to note about each model as they relate to anonymization of social network data.

9.1 Untrusted-Personal

The Untrusted-Personal model is the model generally used by Social Networks, wherein the entities involved are not trusted and information is explicitly associated with an identity. The important thing to remember about this model is that it is the existence and prevalence of this type of exchange that makes re-identification attacks possible. The association between social network information and an identity allows such information in any anonymized data set to be de-anonymized using social network data. As such

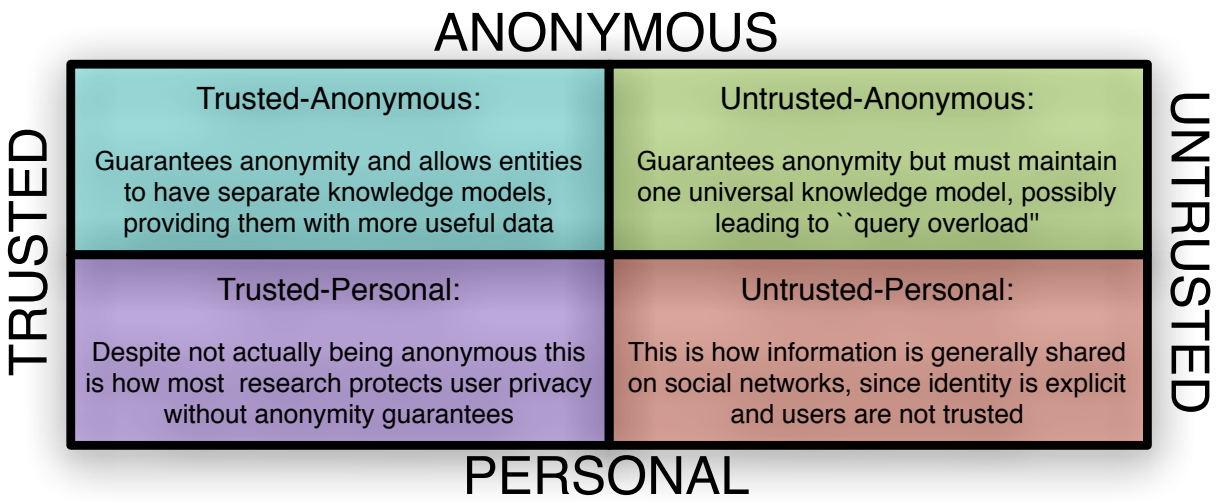


Figure 9.1: Four Assumption Models for Trust & Anonymity

any information released in this way must be considered "public" and for generality we assume all social network data is or has been released this way.

9.2 Untrusted-Anonymous

The Untrusted-Anonymous model is the most basic model for an anonymous social network API. Social network applications could be written to query this API and receive anonymized data through it. Users could then run these applications with guarantees and measurements as to how anonymous their data being released is. Furthermore, if a user only allowed their data to be accessed through one anonymous API, the anonymous API could build up a knowledge model for all new data created by the user after they began using the anonymous API and guarantee or measure the extent to which all API responses every released by the API affect that data. The limitation with this model is that, as the number of users and queries grows, the API's knowlege model must assume all data is possibly shared (entities are not trusted) and as such will reach a point when little or no new information may be released without compromising some users anonymity. For simplicity we'll call this state "query-overload" as the anonymous interface's knowledge model is effectively overloaded. Note that the existence of such a state points to a rather simple denial of service attack.

9.3 Trusted-Anonymous

Adding trust into an anonymous release model allows the anonymous interface to maintain non-universal knowledge models. This allows for the anonymous interface to respond to one set of trusted entities without limiting the information it can give to another trusted entities. Such an interface could be set up along with legal protections to support academic or government research using social network information.

However, one might ask why data being given to a trusted entity should be anonymized. This question can be approached from the perspective of the data provider (social network user) or trusted entity that may be a consulting company, government, or academic researcher under some contract ensuring legal support for the trust or within a system that enforces trust. Consider the example of the anonymous STD screening that uses anonymized social network data to predict STD spread, the user may trust that the hospital will handle their information as expected, however they may still desire to maintain their privacy. Often the guarantee of privacy is what convinces people to take part in STD screenings and studies. From the perspective of the trusted entity, avoiding data that could have been used to violate a contract or compromise a research participants privacy provides the trusted entity with some level of plausible deniability and avoids the chance of accidentally breaking the anonymity agreement. Such anonymization techniques could be alleviate requirements for human subject testing, streamlining the research process when unique or personally particular information is not needed.

9.4 Trusted-Personal

Despite it not being defined as “anonymous” according to our definition, the trusted-personal model is the traditional model for research and studies that guarantee protection of their participants anonymity or privacy. Existing anonymity metrics are often not utilized in “anonymous” studies due to lack of understanding or impracticality. Understanding this distinction is important as the lack of any real, measurable anonymity guarantee should be explicit when social network users allow access to trusted entities. Understanding that their data will not have any anonymity guarantee may lead the participants to evaluate the use

of their data more carefully as they should, considering whether they desire their identity to be revealed within the study and to those entities that have access to the data.

This discussion of different information exchange models has shown how important it is to appropriately understand whether or not your information is really anonymous and whether the person you give it to is really trusted. Furthermore, it has describe how participation in one type of data-exchange may affect the data utility in another type of data exchange. Any attempt to release social network data anonymously should be aware of these implications.

Chapter 10

Solution: AnonyGraph, A General Data Model to Support Anonymity

This section will briefly introduce AnonyGraph. It will discuss the core AnonyGraph model, which allows the relationships between data to be abstracted from the content of data. It is this abstraction that allows anonymity and information metrics, trust, knowledge, and predictability models to be defined such that they are agnostic of the data source. This abstraction also allows data sources to take advantage of these anonymity techniques by only mapping their data model into AnonyGraph, rather than having to integrate these many models into particular applications.

10.1 AnonyGraph Relationship Model

AnonyGraph is both a system and a model. The model is a graph in which edges are undirected and typed (multi-dimensional). Nodes or vertices represent all of the objects in the graph (and any intermediate objects necessary to represent relationships at a finer granularity). Every edge has a unique id and content. Every relationship between objects must be represented in the graph, including unique IDs (an edge of type ”id” from one node to another node with the ID as its content) and node type (an edge of type ”type” from one node to another node with the type name as its content).

Anonymity metrics are defined in relation to a “protected” class, which is a set of nodes in the graph. In most cases the protected class is the “user” or whatever nodes represent the people in the data [1] . These protected classes are specified in AnonyGraph terms, for example the “user” class is specified as those nodes

[1] While it is common to anonymize the relationships of the data to people, there are cases when the relationships between non-people data types are considered private. However, this is rarer than one might think - for instance the private nature of correlating certain demographics with certain diseases is only private because of how it might relate to a person. Such relationships should be modeled with people as the protected class.

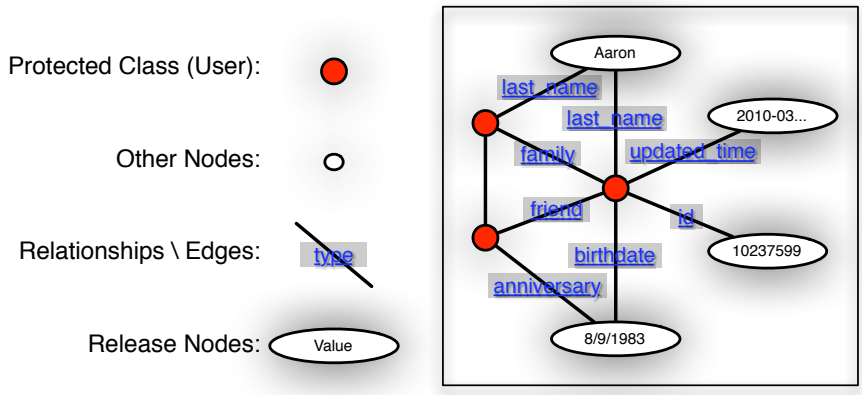


Figure 10.1: AnonyGraph model of attribute-user relationships.

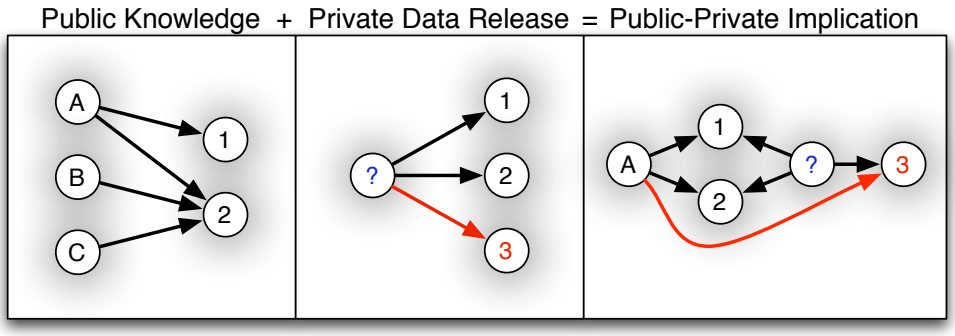


Figure 10.2: Basic Example of a Re-identification Attack

connected to the “user” node by a “type” connection. Nodes of the protected class are shown in the example in figure 10.1.

Also, while all AnonyGraph nodes have content, many of the nodes exist only to represent the relationship structure between different nodes - and as such their content is never directly released. To simplify examples, only those nodes whose content that is actually released (or anonymized) are shown with their content. Since the examples in this paper tend to show simple subgraphs, the nodes with content appear as leaf nodes on the edge of the graph. However in a large AnonyGraph these nodes are not leaves at all and may be highly connected (or “central” to the graph). These nodes are referred to as “release” nodes because taken together they include all of the original “content” in the dataset.

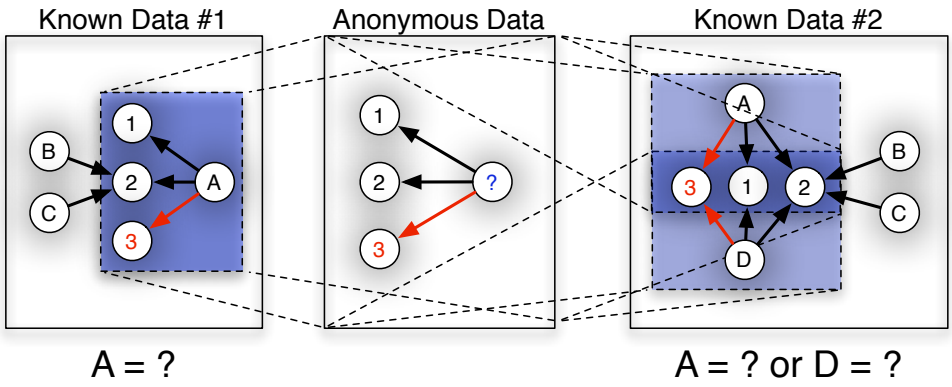
A small example AnonyGraph can be seen in figure 10.1. In this example there are three nodes of the protected class (”user”). The central user node is connected to four “release” nodes, which include the user’s last_name, updated_time, id, and birthdate attributes/fields. Also, the central node is connected to another user through a family connection, and this user maps to the same last_name attribute. The central node is also connected to a user through a friend connection and this node maps to the “8/9/1983” node. However, the two nodes map to the date differently, one node maps by “birthdate”, the other by “anniversary” - these nodes therefore do not share a mapping, but only map to the same node.

10.2 "Linkability" and AnonyGraph

The central question of anonymity is to measure or describe how one piece of data relates to an identity. Data is anonymous if it is "of unknown identity", in other words it can not be mapped to an identity. Anonymization attempts to render data anonymous. It is in this sense that "the central question of anonymity is to measure or describe how one piece of data relates to an identity". The simplest way to answer this question is to answer whether or not someone could take the data and map it to an identity. A re-identification attack is when known or public data is used to de-anonymize supposedly anonymous data. An example of such a de-anonymization is shown in figure 10.2. This figure shows how a set of information, whose relationship with a particular identity is public knowledge, allows supposedly anonymous information to be mapped to an identity (de-anonymized). In this example the identity 'A' is the only identity that maps to attributes '1' and '2'. The anonymous data release includes the attributes '1' and '2' together with another attribute '3'. Taken together with public or known attribute relationships, the private attribute '3' is related to the identity 'A'. However, if we know that there is another identity 'D' which also maps to attributes '1', '2', and '3' (as shown in figure 10.3), then we know that the anonymous data is not uniquely mapped to an identity. Therefore, we can conclude that the subgraph in this figure is anonymous in the sense that it does map to a unique subgraph within the entire AnonyGraph of known data.

The basic observation made by k-anonymity [56], and the view generally held by anonymity researchers, is that simply removing "unique identifiers" such as Social Security numbers is often not enough to render data anonymous. To render data anonymous, one must ensure that nothing about the data allows it to be mapped to an identity. In AnonyGraph terms, such a guarantee is defined as follows: *Data is anonymous, in relation to an AnonyGraph X, if its AnonyGraph representation Y does not uniquely map to one subgraph in the AnonyGraph X.* An example of such a unique mapping is shown in figure 10.3. Technically this definition is equivalent to proving Social-K [**?**] (generalized k-anonymity) with $k \geq 2$.

Figure 10.3: Examples of data which have or don't have ℓ-diversity > 1



10.3 Facebook as an Example of How AnonyGraph Encodes the Data Relationships

The classic anonymization problem is not understanding one's data (and its inherent relationships). This is the failure that motivated the original k-anonymity research and subsequent improvements (ℓ-diversity, p-sensitivity, t-closeness, Differential Privacy, &c,...). k-anonymity pointed out that many types of data may together by "unique identifiers", it termed these quasi-identifiers. ℓ-diversity and p-sensitivity observed that even if the relationship between the user and their data is ambiguous, the relationship between different data may not be and that unique relationship may be considered private as it relates to people (this observation was taken further when t-closeness observed that not only unique relationships, but even statistical correlations between data may constitute a privacy violation). Furthermore, Differential Privacy [**?**] showed that even non-involvement (or inclusion) of a person's data in a data release may increase the "risk" of their anonymity being compromised. Every one of these observations is related to properly understanding ones data and its inherent relationships. This work identifies two primary challenges to understanding and representing data appropriately via AnonyGraph. These two challenges are:

(1) **Explicating all implicit relationships** All relationships inherent in a data set must be explicated in the nodes, edges, and edge types of AnonyGraph.

(2) **Granularity of Data Atoms:** The system can only support anonymization and release of data to the extent that such data can be broken into its constituent parts and their relationships.

10.3.0.1 Explicating all implicit relationships

For AnonyGraph to anonymize data its relationships must be encoded as nodes and edges in an AnonyGraph. To help understand what this looks like we have included an example mapping, taking a dataset originally used by Latanya Sweeney to explain k-anonymity, we show the equivalent AnonyGraph representation, and the associated anonymity violation in figure 10.4. In this case the six rows and four columns of the database are mapped to 20 nodes and 24 edges of 4 different edge types. 14 of the nodes are "release" nodes and contain all of the values in the dataset while the other nodes encode the implicit relationship structure, albeit rather simple. The public dataset used for de-anonymization was not integrated into

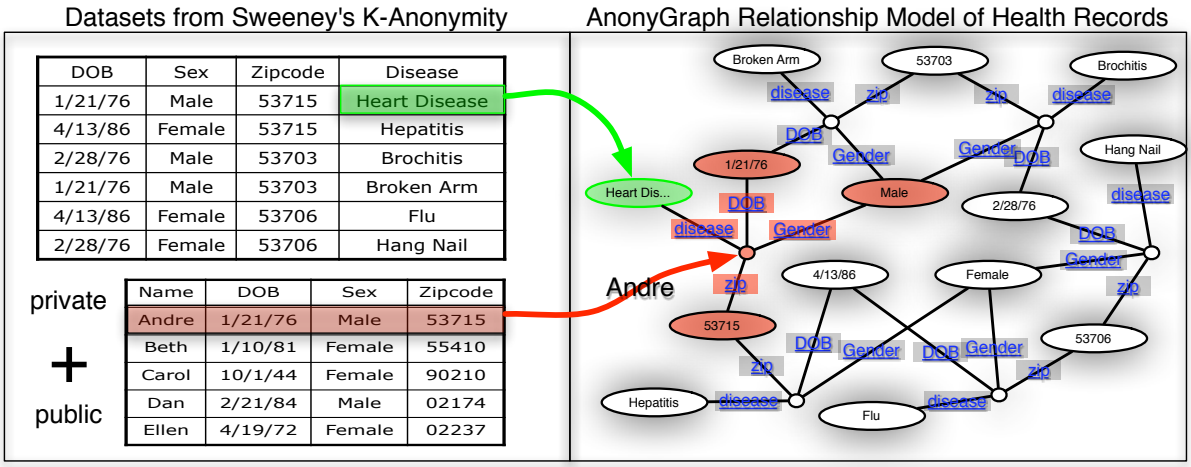


Figure 10.4: Example of Sweeney's Anonymity Violation in AnonyGraph

the AnonyGraph, however the sub-graph associated with its first row is specified within the AnonyGraph, revealing the inherent anonymity violation in the supposed “anonymous” health records.

A single table data set is not particularly complicated, and its relationships are simple. Likewise, translating such a data set to its AnonyGraph representation is not complicated. A node is created for every row and every unique value in the table. An edge is created for every cell in the table, connecting the row’s associated node with the cell value’s associated node by an edge type of the associated column name. In this way all of the tables implicit relationships are represented by the AnonyGraph. However, not all data sets are so straightforward. For instance other datasets may be databases with many tables related with join tables, etc... The schema may explain most of these relationships, however, this is assuming the schema appropriately reflects the nature of the data. The researchers propose to use Facebook and its Graph API to test and prove AnonyGraph; this requires mapping JSON responses to Facebook Graph API requests into the AnonyGraph structure.

The Facebook Graph API is of particular interest to the researchers due to the fact that Facebook recently adopted a simple Graph model with which to represent the majority of the relationships throughout its vast data set. For instance, objects such as users, events, groups, or photos are connected by connections such as friends, photo tags, group association, and event attendance. This shift results in Facebook having done a lot of the initial work that would be required to efficiently map Facebook data into the AnonyGraph model. However, Facebook has not converted all of its data into this model and the researchers had to convince themselves that this “leftover” data could also be mapped to AnonyGraph. The researchers have developed a prototype system which maps Facebook JSON responses to AnonyGraph, inferring the relationships from the JSON structure. An example JSON response to a request for a user object is shown in figure 10.5. This example has been stripped of most of its data to make it easier to understand, however the current AnonyGraph prototype is capable of processing the entire User JSON object returned by Facebook.

This example emphasizes how a rather simple set of data may be broken into many different relationships. In particular the structure of the “education” relationship that includes only two schools the user has attended (their undergraduate and graduate affiliations), relates to an AnonyGraph substructure including 24 edges and 24 nodes, 14 of which are release nodes containing the education content and 10 other nodes

http://graph.facebook.com/10237599


```
{
  "id": "10237599",
  "name": "Aaron Beach",
  "first_name": "Aaron",
  "last_name": "Beach",
  "education": [
    {
      "school": {
        "id": "228401243342",
        "name": "Northwestern University"
      },
      "year": {
        "id": "112527355447971",
        "name": "2006"
      },
      "concentration": [
        {
          "id": "128840923797111",
          "name": "Computer Science"
        },
        {
          "id": "137441329875717",
          "name": "History"
        }
      ],
      "type": "College"
    },
    {
      "school": {
        "id": "112055585473278",
        "name": "University of Colorado at Boulder"
      },
      "year": {
        "id": "112528265447971",
        "name": "2011"
      },
      "type": "Graduate School"
    }
  ],
  "updated_time": "2010-03-15T21:41:48+0000"
}
```


AnonyGraph Relationship Model (sub-graph)

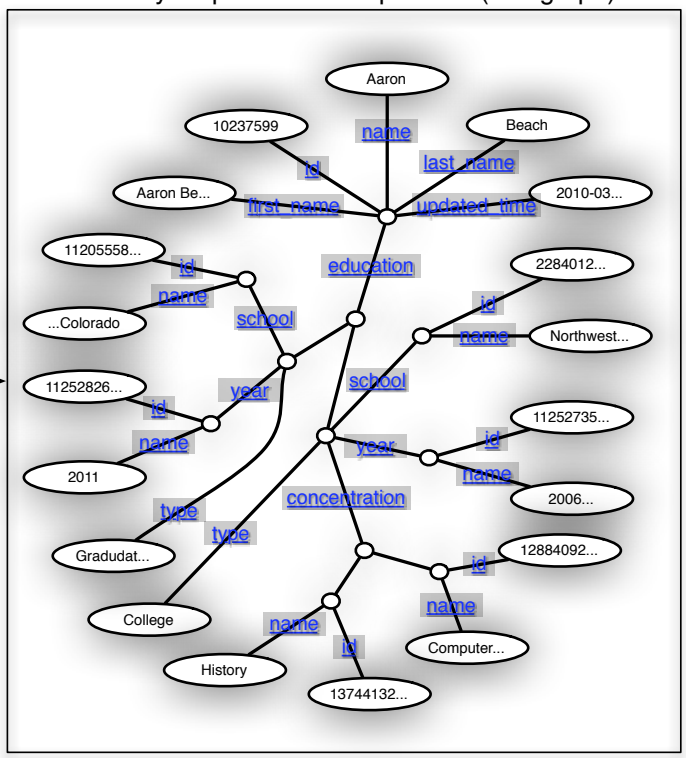


Figure 10.5: Example Facebook JSON response modeled in AnonyGraph

encoding the structural relationship between that data. If the data was not broken up in this way then it could not be distinguished by AnonyGraph. This is important because it allows AnonyGraph to understand that while two users may have attended the same school - the nature or context of their relationship to that school may have been different. In such a case, a piece of data indicating that a person actively attended a school in a particular year may be anonymous, while a piece of data that specifies a person graduated from the school in 2006 with a major in computer science may not be anonymous. Without breaking this data into its constituent parts, AnonyGraph would have to consider the entirety of the data as a single node and such a node may be very rare, very possibly unique, and therefore it and none of its included information could ever be released and considered anonymous. However, this information is encoded in the structure of the JSON response from Facebook.

10.3.0.2 Fragmenting or Breaking Data Into Its Constituent Parts

What about data that Facebook doesn't break into pieces for us? Many types of information such as education, familial relations, or item preferences (movies, books, &c,...) have relationship structures that are well understood (genre, location hierarchies, maternal/paternal relations, &c,...) with which they can be well structured. However, other types of information are harder for computers to break into their constituent parts, understanding them the way people might. Information such as photo, video, and free text are complicated not but not impossible for computers to understand. For instance image recognition allows for face identification, face recognition, and even action recognition (the new gaming systems such as Kinect). Also, new techniques in natural language processing allow computers to infer language structure, topic, and even context to some extent.

The researchers propose to explore how these new ways of understanding photos, videos, and in particular free text might support their anonymization without specialized anonymity metrics, but rather by being mapped to an AnonyGraph relationship model built on existing image recognition and NLP techniques. For instance, on Facebook many users will promote an event or cause by replacing their profile picture with a common picture. Currently we have no way of encoding this information (unless the picture is mapped to a common URL) and as such we do not release this information despite the fact that it may

not hurt anonymity. Also, if text could be understood and broken into a general structure, idiosyncrasies or proper nouns may be removed to retain some meaning of the original text while rendering it anonymous.

The researchers plan to collaborate with other researchers to understand how more advanced methods for understanding data might allow the inherent relationship structure of such data to be inferred and represented in AnonyGraph. In particular, the researchers plan to work with NLP researchers to process pieces of large text on Facebook such as comments, notes, and users' "about me" section. If such text can be effectively anonymized, it opens the door for many applications such as automatic anonymization of classified documents.

10.4 Defining Information and Anonymity Metrics on an AnonyGraph

Anonymity has been defined for AnonyGraph in its most basic terms, but the many different types of anonymity and their improvements must still be defined in AnonyGraph terms if they are to be generally applied to all AnonyGraph representations. This section discusses how a few basic anonymity metrics are defined in AnonyGraph terms along with examples. The researchers propose to implement and test these anonymity metrics with real Facebook applications.

If anonymity is to be practically achieved for Facebook applications, AnonyGraph will have to support not only anonymity metrics, but anonymization algorithms that "anonymize" data, changing or choosing certain data from an un-anonymized data set such that it meets the requirements of the anonymity metrics. Given a particular anonymity metric there may be many different AnonyGraph sub-graphs to choose from which are equally anonymous; the anonymization algorithm must choose between these. The best choice may depend upon the application, who will want the sub-graph that reveals the most useful information for that application. As such, different information metrics may be better for different applications. These metrics will need to be defined in AnonyGraph terms and integrated together with anonymity metrics to achieve data which is both anonymous and useful.

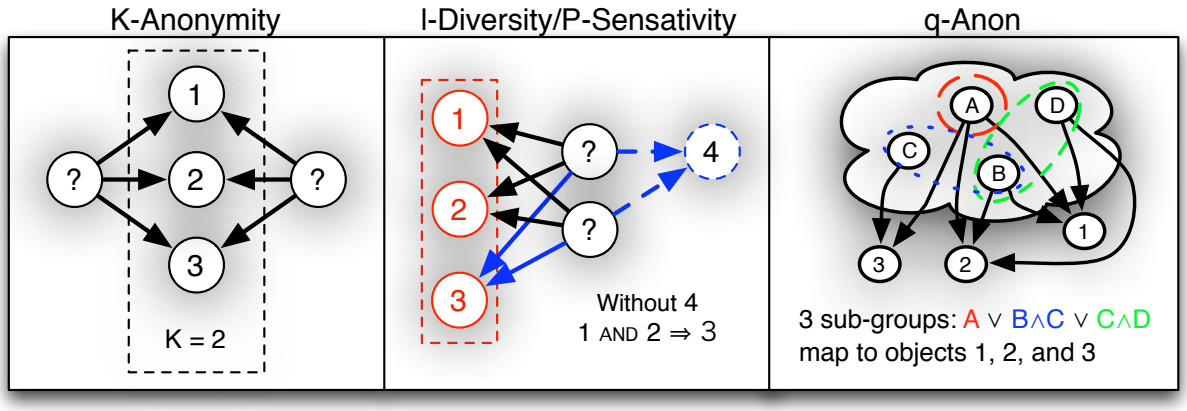


Figure 10.6: Different Anonymity metrics and their AnonyGraph Representations

10.4.0.3 Measuring Anonymity: k-anonymity, l-diversity, q-Anon, ...

This section describes how different existing anonymity metrics are defined in AnonyGraph focusing on Social-K (generalized k-anonymity), l-diversity, and q-Anon. These examples have been investigated by the researchers as examples of how many types of anonymity metrics might be defined for AnonyGraph.

Social-K: *Social-K defines anonymity in terms of a variable K. An AnonyGraph X is Social-K anonymous in relation to another AnonyGraph Y if it maps to at least K different sub-graphs in Y.*

Figure 10.6 shows an example of how two different nodes share three different mappings. The release of those mappings to an unknown node "?" is considered Social-K anonymous with $K = 2$.

P-sensitivity and l-diversity attempt to defend against the case when a set of attributes mapped to some person, imply another attribute. The problem is that even if anonymized data with such a property cannot be uniquely mapped to an identity, it does possibly disclose an unknown relationship for all identities fully mapped to a set of attributes. For instance, if all people of a certain age, race, and gender in a particular location are shown to have the same disease then every identity of those people is mapped to the disease due to the release of this information. In terms of AnonyGraph this is like a set of user nodes mapping to a set of attributes and one of the attributes is the only node in the entire AnonyGraph connected to the entire set of nodes by its particular edge type. In the example given above this would be a particular disease node connected by the "disease" connection. The following describes how this is defined generally for AnonyGraph.

ℓ-diversity: *ℓ-diversity defines anonymity in terms of a variable ℓ. An AnonyGraph X is ℓ-diverse in relation to another AnonyGraph Y $\iff$ $\forall$ connected vertex-edge pairs $(x_i, e_i) \in X$, have X' be the*

AnonyGraph X with vertex x_i anonymized. Given S' the set of sub-graphs that map to X' and set S_v the set of vertices in the sub-graphs of S' which are connected to edges of the same type as e_i: $|S_v| \geq \ell$

An example of ℓ-diversity in a sub-graph is shown in figure 10.6. In this example three attributes are ambiguously mapped to two different nodes. Attribute '3' happens to be mapped by an edge of some type (shown in blue; unless some other attribute (attribute '4') is also connected to the two central nodes by an edge of the same type (shown as blue), attribute '3' would have no diversity, i.e., attributes '1' and '2' would imply '3'.

Another practical example of a possible data diversity violation is shown in figure 10.7. In this case, two females that share the same birthdate and live in the same zip-code also share the same disease. The problem may be that there is no other knowledge of such demographics mapping to any other diseases, resulting in the "anonymous" data divulging a unique mapping that may have been unknown. The existence of such non-diverse nodes in a data release would constitute a privacy or anonymity violation.

Finally we will give an example of how an anonymity metric designed for group queries can be accommodated by the AnonyGraph model. q-Anon was developed by the PI to achieve strong anonymity while maintaining data utility for group queries such as requesting the movie-preferences for all people that live in a certain zipcode [**?**]. Given all possible people specified by the query criteria (e.g., people living in a particular zip-code), q-Anon guarantees that the data attributes released possibly map to many distinct sub -groups. Furthermore, that there is no common person throughout these sub-groups.

A note on **Differential Privacy:** Differential Privacy is defined in terms of learned statistical and distribution information about a data set whether described or directly released. However, many social network applications request a particular data field for some user, needing a discrete list of items *whether or not that list says anything in particular about the distribution of the users data*. They need this information to fill in the space between HTML tags in their application, not for research purposes. As such, many of these applications would benefit from anonymization algorithms that optimize for a particular information metric without providing any bounds or guarantee of how "representative" they data they receive is of the the real or complete distribution of the data. As such, we are not initially integrating a differential privacy generator, however we plan to integrate such a generator as an anonymization/information option targeted

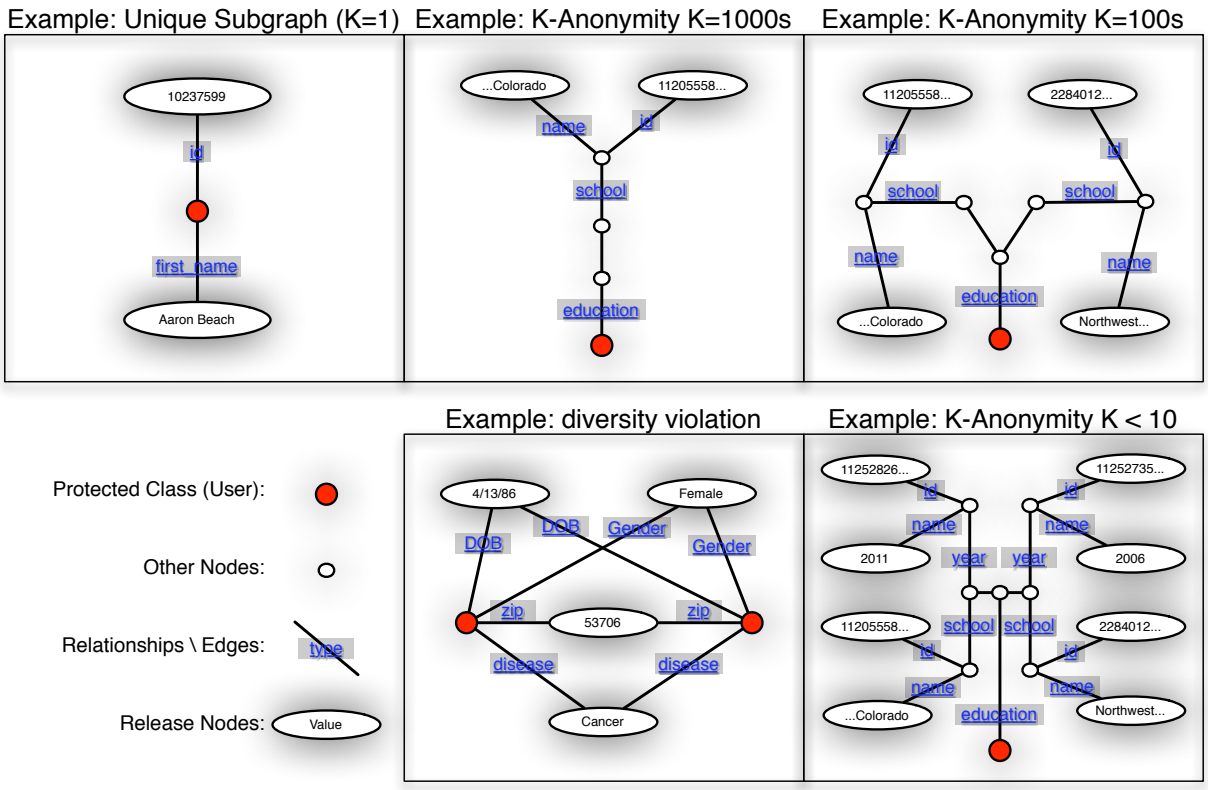


Figure 10.7: AnonyGraph Examples of Strong/Weaker Anonymity Levels

at researchers and analysts [2]

q-Anon: *q-Anon is defined in terms of a variable q. An AnonyGraph X is q-anonymous in relation to another AnonyGraph Y $\iff$ Given S_g the set of sub-graphs in Y which X maps to and S_v the set of vertices in the sub-graphs of S_g that are not in X, $\exists$ a set S_{S_v}, a set of all subsets in S_v that together map to all vertices in X and are not supersets of each other where $|S_{S_v}| \geq q$.*

An example of a q-Anon is given in figure 10.6. This figure shows three attributes: '1', '2', and '3', together map to three distinct sets of people, which are represented by the groups of nodes (A), (B,C), and (B,D). In this example the groups of nodes could be those users who live in a particular location and the attributes could be their movie preferences. The release of attributes '1', '2', and '3' do not constitute a privacy violation since their mapping to the protected class is ambiguous. However, what if there were many movies which together constituted a privacy violation and we had to choose which anonymous subset of those movies should be released, we would need to know which subset was the most useful to the application. The next section discusses how such choices might be made in the AnonyGraph system.

10.4.0.4 Measuring and Maximizing the Information Utility of Data

One of the reasons the researchers are proposing to test the AnonyGraph system with the Facebook API is its diversity of application. Facebook applications range from personalized recommendation and context-aware systems to simple games such as Farmville [3] and can even include research studies [54]. The information preferences of these applications vary widely. By specifying information metrics to be used during anonymization applications can maximize the utility of released data without violating anonymity. However, for AnonyGraph to do this these information metrics must be definable on the AnonyGraph structure.

An example of a simple information metrics is tf-idf [**?**] (term frequency-inverse document frequency). This metric gives high scores to information that is rare, preferring it over common information.

[2] It's interesting that a key assumption of most anonymity research is that users of anonymized data are most interested in the distribution and statistical relationships between data, however many applications are simply interested in the data atoms themselves for the purpose of filling out the content of the application and making it interesting to the user. In fact, in our initial experience most social network applications developers can say or care little about the actual distribution of data throughout Facebook and yet this assumption is almost always made by anonymity metrics developed by the research community

Such an information metric could be used to select those attributes that are rarer while removing common information to achieve greater anonymity in a data set. However, anonymity and the rarity of an attribute are inversely correlated. If many attributes are rare is it less likely that they are not unique when taken together, that is unless they are positively correlated with each other. Such rare, but positively correlated attributes, form clusters in the AnonyGraph. Therefore, the researchers will investigate how clustering algorithms might be used to define information metrics for AnonyGraph.

A clustering technique developed at Toulouse University is being targeted as an example of how clustering might be used as an information metric for annonymization [16]. This algorithm uses the Simpson coefficient to group nodes that are frequently associated with another common node. If these groups are considered "connected" by their association with the group, community detection (or k-clustering) can be performed on the implied graph, clustering nodes that share many of the same connections. These clusters could be used as the basis for deciding which connections are most "representative" for a particular node. Using such a metric, the information released by AnonyGraph would tend to look like personas of the people in the data set, increasing the utility of the data for applications that are interested in targeted demographics (market segmentation) or in generalizing the goals and behavior of their users.

However, the ultimate goal is to show that useful information metrics can be reasonably defined in terms of AnonyGraph such that future anonymous applications may even specify their own information metrics.

10.5 Integrating Knowledge, Trust, and Predictability Models

Guaranteeing the anonymity of a single data release requires the appropriate understanding of your data and application of anonymity metrics. However, even then multiple anonymous data releases may add up to an anonymity violation. Furthermore, social network applications often make thousands of requests every day (resulting in thousands of data releases). To protect the anonymity of those in a data set across many data releases the AnonyGraph system must guarantee that no one entity receives multiple data releases which it may use together to compromise any users anonymity.

Furthermore, understanding what one entity knows requires knowing what other entities' information

that they may have accessed. This requires being able to predict how an entity will share the information they receive from an anonymous data release. This is not such an easy thing to understand, as the many WikiLeaks scandals attest to. Even the US government cannot guarantee that its military data is secure. In the case that an entity is not predictable, measures must be taken to prevent leaking the wrong information to that entity.

Also, the anonymity metrics as discussed in this paper assume that the base AnonyGraph used by the system is trustworthy and does not contain false data inserted by an entity attempting to manipulate the AnonyGraph into falsely believing a set of information is anonymous when it really is not. This section discusses how knowledge, predictability, and trust can be integrated into the AnonyGraph system.

10.5.1 Integrating Knowledge

Integrating knowledge may increase whenever the AnonyGraph system releases data to an entity. Furthermore, the data from one release may be shared with many entities or leaked throughout the internet. As such, without some protection the system must assume that anyone has access to a data release. This results in single "universal" knowledge being assumed available to all entities accessing the AnonyGraph system. To model this knowledge a sub-graph should be maintained by the AnonyGraph system which is the union of all sub-graphs released by the system. If such a sub-graph ever constitutes a privacy violation then the system can be considered possibly compromised. Figure 10.8 shows how multiple Facebook queries for objects and their connections can be joined in a union to fill out a graph. Furthermore, it shows how the Social-K anonymity of such a graph decreases as more data is released by the system. In the example presented in figure 10.8 the sixth request results in a privacy violation as the cumulative knowledge gained from the six requests implies a sub-graph which uniquely maps to only one sub-graph in the AnonyGraph.

The first request in figure 10.8 only reveals that there is a person of uknown id, with the last name "Aaron" and birthdate "8/9/1983" the recently updated his account. Howsoever, assuming the last name is common this information may be rather anonymous given that there are over half-billion users on Facebook. Further requests reveal that the user has a friend and a family member also on Facebook. It turns out the family member shares the same name and the friend shares the same birthdate. This information turns

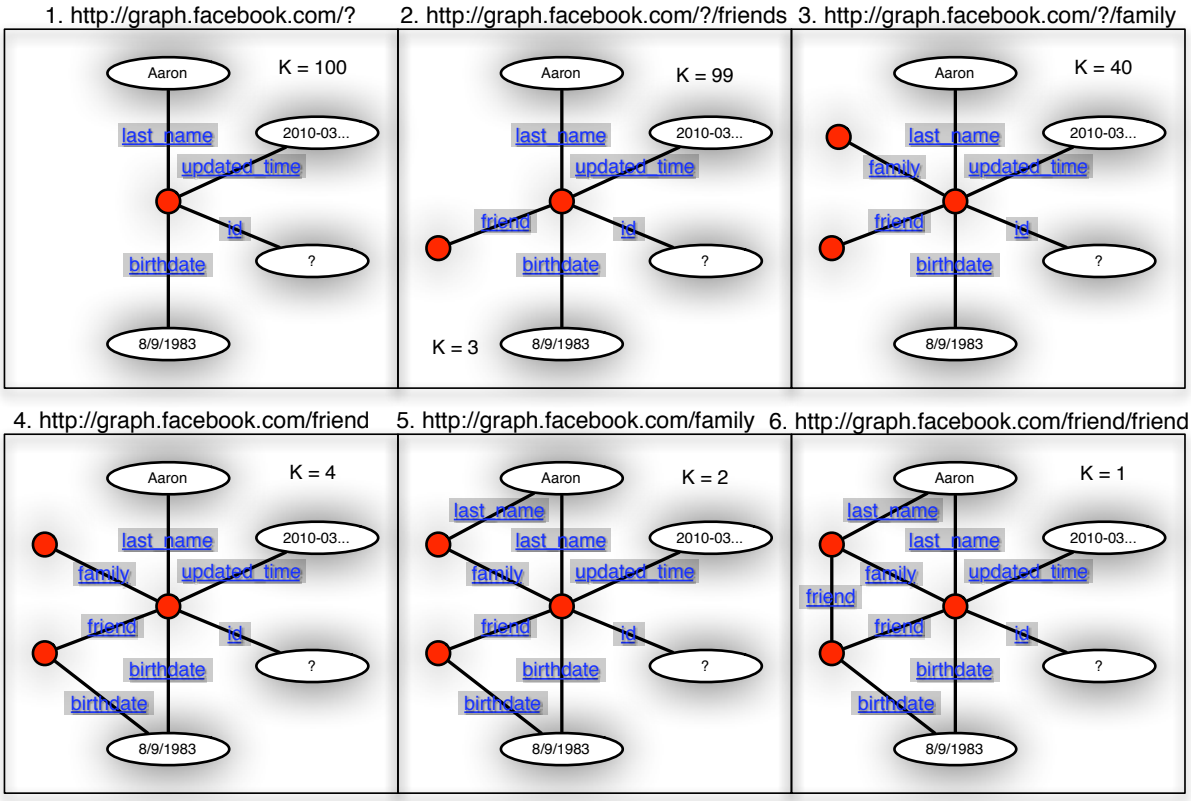


Figure 10.8: Example of Knowledge Gain over Multiple Queries

out to be rather rare; there is in fact only two Facebook users that match this profile and the fact that the friend and family member are themselves friends renders the match unique. In this fictitious example, the "anonymous" responses to these six requests constitute an anonymity violation. To prevent such a violation the last response should have been withheld. Maintaining a knowledge model would have allowed the system to detect and prevent this violation, however, defending against such violations significantly reduces the utility of the system over time. To allow for a greater amount of data to be released the system must have some guarantee (predictability) of how entities may share information with each other.

10.5.2 Integrating Predictability

Predictability takes the form of user-application trust, which may be implemented through some agreement and enforced or managed legally, through watermarks, or even physically enforced through a protected data access site. Examples of protected data access sites such as the Census Research Data Centers managed by the National Bureau of Economic Research are probably not a scalable solution for all but the most important types of data. In other cases, legal protections may deter would-be privacy violators with the threat of a tort and suit for damages. However, proving which entity is at fault would require tracing information back to its particular data release and doing so may require some form of watermarking in the data. The researchers are investigating how watermarking the data relationships themselves could be achieved through selective insertion and repression of data values in the annonymization process. If particular information is given to an entity it may be passed along if the entity "leaks" the information. If such information were systematically inserted such that it could be uniquely identified at some later time, the data leak could be traced to its source. Such techniques may deter some leaks such as those involved in the WikiLeaks scandals. However, in the end the most common form of predictability will be the users own agreement to trust a particular application. If the user is willing to take responsibility for trusting an application a separate knowledge model could be managed for the application, allowing for a greater amount of data optimized for the particular user-application release. Finally, we will discuss how existing trust relationships enforced by data providers such as Facebook may be integrated into AnonyGraph to defend against entities that would seek to fool AnonyGraph by inserting false data into the data set.

10.5.3 Integrating Trust

Trust exists between Facebook users to the extent that they give each other access to their information. A piece of information which no one can access is of little utility on Facebook. A user puts information on Facebook for the purpose of sharing it. Facebook allows users to specify which users have access to their information. This form of trust can be used to defend against attackers using false data insertion to access discover unknown private data relationships. This can be achieved by limiting what information is considered when calculating different anonymity metrics. For a particular protected node (user) and piece of information (x) being measured for anonymity, only information inserted by those nodes the user has has already granted access to the piece of information (x) will be considered when calculating anonymity. Such that the only way a privacy violator could fool AnonyGraph by inserting false data is by compromising an already trusted account, in which case the privacy violator would, by definition, already have access to the private piece of data and thus the release of such information based on its measured anonymity could not increase the risk of anonymity violation to the associated user. In short, if a privacy violator were capable of false data insertion they would already have access to the information being anonymized.

For any system to achieve anonymity for multiple data releases it must maintain knowledge models for all previous data releases, make predictions about how data may be shared once it is released, and not fail due to false data in its system. Achieving these goals presents many hard research problems, not the least of which includes watermarking the relationship structure of information such that might be traced to its source if leaked, however the researchers feel they have shown that the AnonyGraph model is capable of supporting models and techniques that solve some of these problems, showing promise of supporting future solutions to problems like the information watermarking problem discussed in this section.

Chapter 11

Solution: Whoville, a Prototype that Anonymizes Facebook

Whoville is a prototype anonymous social network API. In particular it mirrors the Facebook Graph API, however all information released is anonymized. Whoville allows applications that were written to use the Facebook API to easily be ported to use anonymous data. From the perspective of a Facebook App Developer, the Whoville application is anonymization service through which they can provide anonymity guarantees to their users and protecting themselves from having to worry about compromising their users privacy or anonymity.

This section discusses the author's experience developing an anonymous social network API. The technologies necessary to support such an API are discussed along with how they are integrated together. The general architecture for our design is given along with a step-by-step description of how an app developer would register their application and submit requests for anonymized data. Examples of how data changes when anonymized under different anonymity metrics are shown along with examples of how Whoville supports application specific anonymity guarantees. Finally a discussion of example applications that have been implemented and tested on Whoville shows how an application can optimize the anonymization process to achieve the highest level of data utility allowed while being anonymous.

11.1 Supporting Technologies

Whoville is an anonymity layer on top of Facebook's Graph API. Facebook's Graph API represents its social graph, including all types of Facebook connections and objects, with a simple rest API. Every object in the graph has a unique ID and can be accessed with this ID at the url:

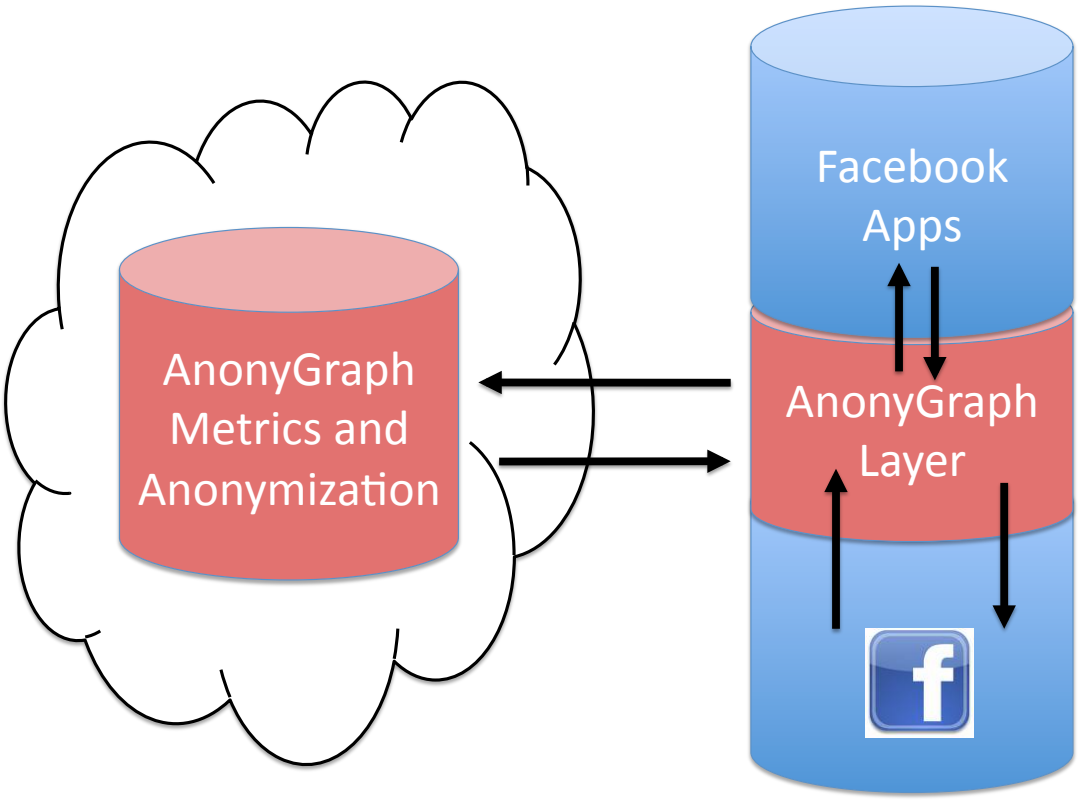


Figure 11.1: Diagram of how Whoville uses AnonyGraph to anonymize Facebook Graph API Queries

`https://graph.facebook.com/ID`

. All connections or relationships between Facebook objects are of a particular type, for instance two users may share a “friend” or “family” connection or a user may have a “fan” or “like” connection to a page for a music band. Connections of Facebook objects can be accessed with the object ID and the connection type at the url:

`https://graph.facebook.com/ID/CONNECTION_TYPE`

.

Application developers who wish to use Facebook’s Graph API must register their application with Facebook and then support the OAuth 2.0 protocol in order to obtain a access token for each user. Access tokens, or “Oauth tokens” are necessary to gain access to users’ data. As part of the authentication process with Facebook the user must explicitly approve the application access to their data. After the user has approved an application, Facebook gives the application an OAuth token whenever the user uses the application.

Facebook Graph API responses encode Facebook data in JSON format. Each Facebook object has a set od default information that is included with it. This information is not encoded into Facebook’s social graph and any inherent relationships must be extracted. How these attributes and their inherent relationship structure are extracted is discussed in section 10.3.0.1.

Over the history of Facebook new features and access patterns often come up against negative user reactions. However, Facebook often responds by giving the user more fine grained control of how their data is shared. One area that has generally gone unquestioned is what Facebook deems a user’s “PAI”. “PAI” stands for Publicly Available Information, and by default this includes unique IDs that publicly map to their identity. This information is automatically shared with “pre-approved” third parties. This usually means that certain internet sites which have an agreement with Facebook can find out which user is accessing their site as long as that Facebook user is currently logged into Facebook from the same IP address. This allows sites that users don’t log into quite as often (cnn.com is an example) to identify exactly who it is that is using their site. This usage model tends to treat Facebook like an identity broker. This project seeks to hide a user’s

identity while releasing as much useful data as possible, whereas the current model of PAI / trusted 3rd parties, automatically releases very little useful information along with the users identity. In this sense the model promoted by Whoville is almost a polar opposite of the model Facebook is most actively pursuing.

Whoville uses AnonyGraph in order to parse Facebook data and anonymize it. Since AnonyGraph supports direct serialization to and from Facebook objects represented in JSON, integration is rather straightforward. Whenever a 3rd-party application submits a request to Whoville, the request is forwarded to Facebook then the response from Facebook is parsed and anonymized by AnonyGraph before being returned to the 3rd-party application. In order to support anonymization, AnonyGraph needs a base of knowledge with which to compare data for anonymization. Whoville uses its users information to seed AnonyGraph's knowledge model. Whenever a user add Whoville, their object information and all connections are parsed and inserted into AnonyGraph. AnonyGraph continues to follow all public connections it finds from the initial seed information for up to 10 hops through Facebook. This has been found to provide a reasonable knowledge base to support anonymization.

11.2 Application Architecture

This section discusses the design and basic function of the Whoville application. The basic structure of Whoville is shown in figure 11.1 in which all third party interaction is with Whoville, which manages access to Facebook and to AnonyGraph, through all data is first passed for anonymization.

Figure 11.2 shows a more detailed diagram of the major components of Whoville:

- **Authentication Manager** This Authentication manager handles all authentication and interaction with Facebook. This includes requesting the appropriate permissions from Facebook for new users and implementing the OAuth 2.0 protocol.

- **Application Manager** The Application Manager, manages all applications and anonymity settings in Whoville. It keeps track of which application have been added by each user and at what levels of anonymity they have agreed to allow applications to access their data.

- **Anonymization Layer** The Anonymization layer can be thought of as simply a wrapper around

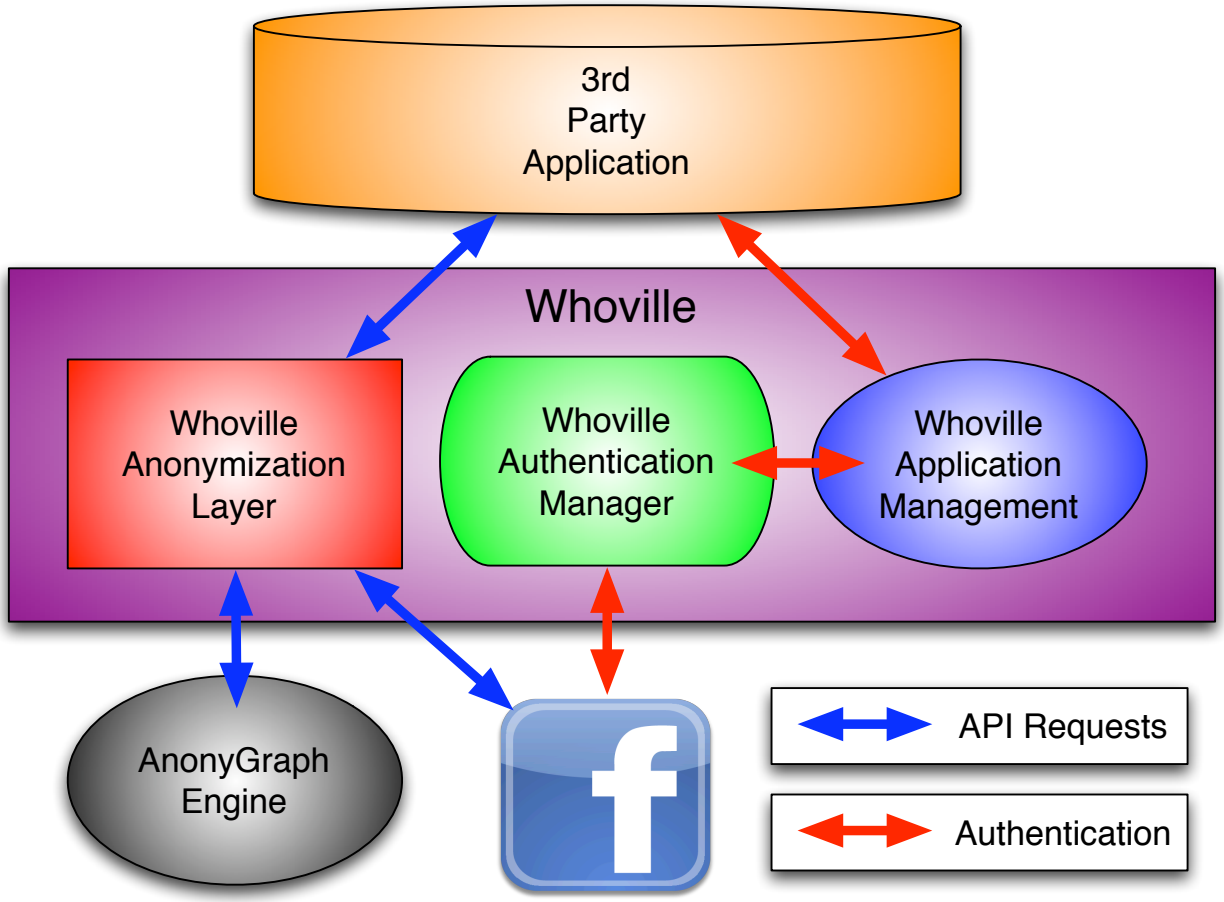


Figure 11.2: Diagram of how different software components of Whoville interact

the Facebook API and AnonyGraph. Whenever an application sends a request to Whoville, the Whoville access tokens are replaced with appropriate Facebook access tokens, a request is made to Facebook, the response is given to AnonyGraph for anonymization, and the anonymized result is returned to the 3rd party application.

In figure 11.2 you can see which components interact as part of authentication with Facebook and 3rd party applications represented with red arrows. You can also see how the components interact to support anonymized query responses represented with blue arrows. To understand this interaction in more detail, figure 11.3 goes through every action that takes place when a user adds Whoville, adds a third-party application, and that application requests anonymized data.

Following the example in figure 11.3, **(1)** the user first logs into Facebook. **(2)** They then add the Whoville application by clicking on a link to Whoville or going directly to the url:

```
http://apps.facebook.com/whoville
```

(3) Once at the whoville application they are presented with a basic description of Whoville along with a link that will **(4)** take them back to Facebook, requesting the necessary permissions from Facebook. If the user accepts and grants permissions to Whoville **(5)** they are returned to Whoville along with an OAuth access token which Whoville uses to access their information. OAuth access tokens expire when the user changes their Facebook password or deletes the application. Once the user has added the Whoville application **(6)** they are taken to the Application community page which allows them to add any existing applications or develop new applications. **(7)** If they add an application by clicking on that application's "add" button **(8)** they will be taken to screen which specifies the anonymity requirements that are applied to this application and asks whether or not they agree to use the application at that level of anonymity. If the user accepts and adds the application **(9)** the user is forwarded to the application's homepage along with a Whoville access token (WhoToken) which can be used to query Whoville. **(10)** When the 3rd-party application queries Whoville using their WhoToken, **(11)** Whoville replaces their WhoToken with a OAuth otken and **(12)** forwards their request to Facebook. When Facebook responds with a JSON repsonse, **(13)** the response is passed to AnonyGraph along with the anonymity settings required for the particular 3rd party application.

AnonyGraph anonymized the response under the appropriate anonymity requirements and then returns the response back to Whoville which **(14)** forwards the anonymized response on to the 3rd party application, **(15)** at which point the 3rd-party application may use the anonymized data to personalize their application or web-site.

11.3 Example API Request & Response

This section presents a very simple example of the data involved in an anonymized Whoville web request. In this example the application requests the "activities" connection for the user "aaronbeach". This correlates to the Facebook Graph API url:

```
https://graph.facebook.com/aaronbeach/activities
```

The corresponding Whoville url does not include the username because this ID is never given to the Whoville application. Rather, the application is told that the current user's ID is the same string as the WhoToken, generally an SHA-1 hash. So the corresponding Whoville url might look like:

```
https://whoville.cs.colorado.edu/6546...15b90/activities
```

In both cases an access token is included as a parameter in the request however this detail has been omitted from the example URLs for simplicity.

Figure 11.4 shows the un-anonymized and anonymized versions of the user's "activities" connection. The Facebook and Whoville APIs actually return data in JSON format, whereas the data in figure 11.4 has has been formatted for web-viewing, in fact it is from a screen-shot of the profile viewer application, however it is a complete version of what is returned by the APIs. The Un-Anonymized data is what Facebook returns the Graph API and the Anonymized data is that returned from the Whoville API. As you can see, not only are certain activity entries removed but also much of the meta-data such as "id" and "created_time" are also removed from the activities that are released. This is a rather simple and very anonymous example simply to show an example of anonymization. However, it does highlight how an anonymization system must consider all types of data not just those that are intuitively private. In this case the data was anonymized

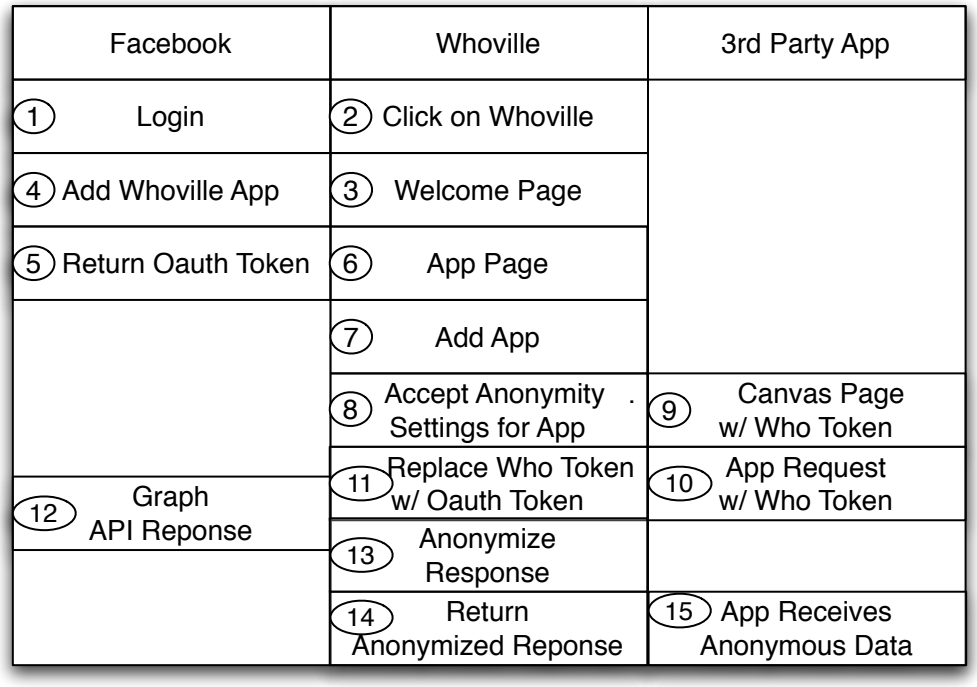


Figure 11.3: Step-by-step example of a user installing Whoville and running a 3rd party application

under Social-K with a very large K, releasing the fact that the user also participates in Bowling or that the particular "Travel" activity they are connected to in Facebook was created on March 2nd, 2001 would have constituted a violation of anonymity.

11.4 Supported Information & Anonymity Metrics

This section discusses how the different anonymity and information metrics are implemented by Whoville along with examples of how using one metric or another affects the data that is released by the system.

11.4.1 Anonymity Metrics

The initial current implementation of Whoville supports anonymization using Social-K [9] and q-Anon [4]. However, by default Social-K is used when anonymizing all normal Graph requests because q-Anon anonymizes in relation to a group of objects and the Facebook Graph API assumes a single object or connection to a single object.

For testing purposes, q-Anon was integrated into an internal application called the "Persona App". The q-Anon implementation is integrated into the API for use by 3rd-party applications by adding a special connection to each Facebook object called "persona". The information returned is intended to represent a summary or persona of the human users connected to that object. The Persona App is discussed more in section 11.5

By default the Whoville implementation of Social-K assumes $K = 2$ and uses only suppression for anonymization. To choose which data is suppressed and released Social-K defaults to TF-IDF or Inverse Frequency to value data.

The q-Anon implementation also uses TF-IDF by default and attempts to produce an anonymized result that is of a representative size of the average user included in its anonymization. Since q-Anon anonymizes the data of an entire group of users this means that the "object" created by the q-Anon is more an anonymous summary of the entire group, attempting to be both as descriptive/informative and anonymous as possible.

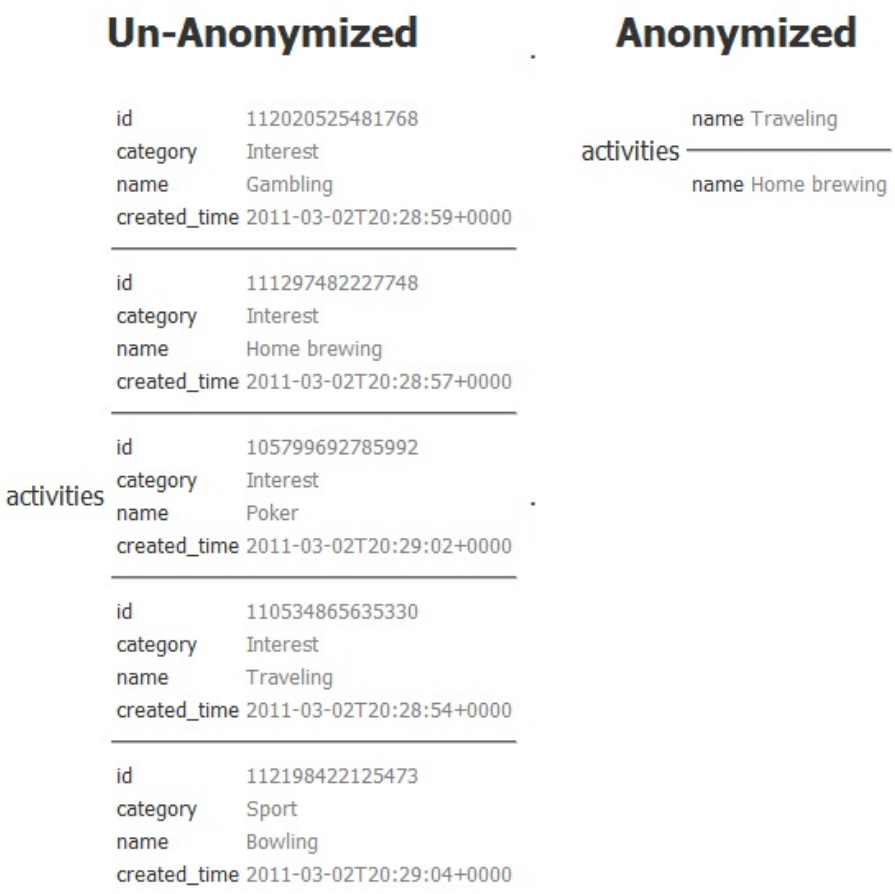


Figure 11.4: Example of the activities for a user un-anonymized and anonymized.

11.4.2 Information Metrics

Whoville support four different anonymity metrics, three of which are shown in figure 11.5, they are:

Maximize Data This metric weights all items equally effectively optimizing to maximize the amount of data that is released.

Maximize Sharers This metric weights items more valuable if they are shared by more users. Within the context of the Persona app or persona connection this means that anonymization will prefer data which is shared by the most users connected with the persona object, in effect maximizing commonality.

TF-IDF Whoville's implementation of Term-Frequency Inverse-Document-Frequency or TF-IDF weights an item as the reciprocal of the item's node degree for a particular connection type in Facebook (e.g., if a user has 700 friends, their friend TF-IDF weight would be $\frac{1}{700}$), this reciprocal is then multiplied by the frequency with which the item occurs in the subset of connections specified by the persona. For example, if 5000 people like the Beatles and 7 of a users friends like the Beatles, the connection "likes the beatles" would get a $\frac{7}{5000}$ weight when being considered for anonymization. This information metric effectively prefers information which occurs more frequently among a particular subset of users than it does in general across all of Facebook.

Application Specific Whoville allows applications to specify which values and fields in which they are most interested. All non-listed fields are excluded and values listed are preferred over those not listed.

These three different approaches result in anonymizations that are equally anonymous but produce different data.

Figure 11.5 shows how the different metrics compare to each other. The "quality" of a persona depends on the subject and the particular application that they are using. As such, it is well beyond the scope of this thesis to evaluate how much better one information metric is than another. However, we have found that self-testing the applications that using application specific field filters and TF-IDF tend to produce

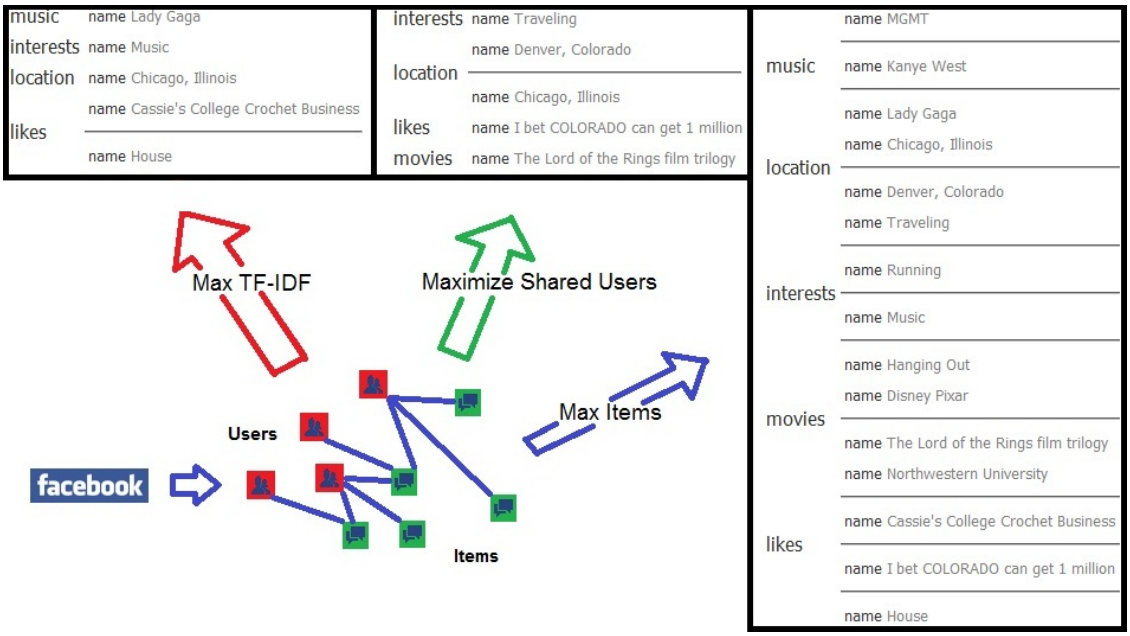


Figure 11.5: Example of persona data anonymized with three different information metrics.

very interesting personas. It is discussed in section 13 how advertisers might use these anonymous personas as an alternative to knowing the users identity.

11.5 WhoApps

This section discusses two internal apps created to help researchers and Whoville users understand how anonymization affects their data. The *profile* application allows them to compare anonymized and un-anonymized objects from the Facebook API. This application uses Social-K to anonymize each object or connection on Facebook with $K = 2$ for simplicity. The *Persona* application allows users to create "personas" of different objects on Facebook. This application was created to show how useful group anonymization can be.

11.5.1 Developer Application

The developer application is Whoville's analogous app to Facebook's developer application. The developer application is an application which allows Whoville users to create their own applications with Whoville. The users specify a subset of the options they would specify for a Facebook application such as the URL for the canvas page, the url for the applications home page, name of the application, description of the application, and URL to their terms of service. Once this is setup their application is available for other Whoville users to add. When a user adds an application that was developed by another user they are presented with a description of the application and a link to the application's terms of service. If they accept the privacy levels supported by the application they may choose to add the application. When the users "runs" the application they are forwarded along with an anonymous "WhoToken" which allows the application to make anonymous queries for the user.

11.5.2 Profile Application

The profile application was designed to help users understand how anonymization affects their data. The profile application makes a direct request to Facebook for the users GraphObejct, the object in which Facebook stores their non-connection data (includes things like name, location, education history, employ-

ments, but not friends, notes, likes, etc...). This object is then anonymized using Social-K with $K = 2$ for a minimum anonymization. The original Facebook object and its anonymized version are shown next to each other so that the user can see what their anonymized information looks like.

11.5.3 Persona Application

The persona application was built to showcase more advanced use of anonymization. Since Facebook does not integrate a sense of group queries directly into its Graph API (which targets objects and their connections). While the Graph API does allow for piggy backing multiple object queries at one time, this requires knowing the Facebook IDs of multiple users and returns multiple results hence it is not appropriate nor allowed to support this kind of functionality in our Whoville app. So we created a new kind of connection that we supported at the Whoville layer above Facebook. We called this connection "persona" and it is accessed like any other connection by specifying the object id and its "persona" connection. For example:

```
http://facebook.com/OBJECT_ID/persona
```

The persona application allows the user to test or learn about the persona connection within the Whoville application before running a third party application that may access their persona. A persona is an anonymous persona of a particular object in Facebook. A persona is created for an object by finding all users connected to the object. Users may be connected to an object by a "like" or "location" connection and are connected to each other by "friend" and "family" connections. The group of connected users are then anonymized together using q-Anon for group anonymization. Results generally filter out non-interesting or non-informative fields, such as time stamps and object meta-information.

By default, the personas use q-Anon (with $q = 3$ for anonymization and TF-IDF to decide which information to keep/supress. However, personas can be created using the different information metrics discussed in section 11.4.2. An example of how personas change under the same anonymity settings can be seen in figure 11.5.

11.6 Performance Enhancements

This section discusses practical enhancements that were made to help Whoville scale and run efficiently. These aspects of the implementation are included for practical purposes of information someone trying to do the same thing as they don't constitute novel solutions to their respective problems, however they do present a few interesting observations real-world anonymous APIs.

11.6.1 Caching & Concurrency

When the Whoville project began Facebook's terms of service did not allow caching of their information for more than 24 hours. However, Facebook's policy on caching did a complete reversal a few months later, not only allowing for indefinite caching but supporting it through a service that pushes user changes to the application. Facebook maintains that the applications should attempt to keep information "up to date" however there are no particular requirements. This change in policy made the large-scale and independent anonymization much more practical. Whereas before, anonymization would require a large number of Facebook web-requests and was limited to locally using whatever information had been downloaded in the last day, now an application could continue to collect user information over time and process anonymization locally over all information the application had learned. As such, while Whoville would work best as a centralized service (preferably run by a trusted service connected directly into Facebook databases), the Whoville code will be released such that it can be run as an anonymization service or integrated directly into an application that chooses to anonymize the data that it receives from Facebook.

When implementing a service that processes many web requests at once, as occurs when an application is processing all connections for a particular Facebook object, it is very common to parallelize the web requests and the processing of their web responses. Aside from Facebook's web traffic throttling and request limits, Facebook data presents a very interesting challenge for being processed in parallel due to its highly interconnected data. Facebook's objects are highly connected, and those being processed together are often being processed together because they relate by a friend or family connection, within which users are densely connected by many connections of different type.

At first, we ran into many stale data errors while processing Facebook data in parallel. The addition of locking on the objects found that legitimate concurrency challenges existed as circular dead-locks would often appear. As such, only hierarchical locking or synchronous processing worked without moving to a more complete solution designed for high levels of parallelism.

11.6.2 Batch Processing

While Social-K and q-Anon have been shown to scale linearly, there is still a reasonable compute/memory cost involved in doing "good" or "strong" anonymization. Also, since we do not have direct access to Facebook's databases we often had to make many web-requests to get the data necessary to anonymize one single request. This often made requests take much longer than would be expected for a web API. While this would not be a problem for Facebook or another entity that managed the data directly, we had to deal with it to practically test the system. In order to do this, a thread could batch process anonymizations, pre-computing them and storing the results. When a user requested a particular object or connection the pre-computed anonymized result would be returned. This results in anonymizations being slightly stale, however, Facebook users tend to add/change only small parts of their data within the span of a week. It would be interesting to relate the rate of data change on Facebook to the necessary compute power needed to anonymize all of Facebook every week. Based on experience with Whoville, weekly anonymization takes much less compute power than providing an active API for many applications around the clock. The Facebook Graph API is much more active and should therefore already be using a significant amount of compute and memory resources. The author is interested in quantifying this in the future.

Chapter 12

Observations: Example Applications

The utility of certain applications may depend on the personal nature of data or use data which cannot easily be anonymized without complete loss of utility. This section breaks applications into four main categories for which anonymization is of increasing difficulty. These applications are then discussed in light of real-world examples using the Whoville prototype.

12.1 Classifying Application by Anonymization Difficulty

Different classes of applications present very different anonymization problems, some very simple, some impossible. While simple game applications may function completely without any personal information, certain types of applications may not allow for any anonymization without completely losing their utility to the user. Understanding which applications these are and what properties make these applications "anonymization-phobic" is important considering how anonymity might be integrated into an application you are designing or planning to use. The following breaks applications into four major classes in terms of how difficult it is to reconcile anonymity with application requirements:

12.1.1 Easy: Anonymous Identifiers

This class of applications require only an identifier by which to distinguish between users and with which application information can be stored. These applications generally provide non personalized services and encompass applications such as games, productivity tools, and other forms of entertainment. In fact, if one considers that applications that automatically personalize their function or services based on social

Easy: Games and Entertainment

Medium: Research and Market Studies

Figure 12.1: Basic games and application that use general informatics are often little affected by anonymization

network data are rather new it becomes obvious that most applications (even social network applications) do not require any personal information to function just fine.

12.1.2 Medium: Basic Demographics

This class of applications is particularly interesting to researchers and analysts who use this information as indicators to study social or economic trends. This type of information includes gender, race, nationality, sexual preference, and in some cases more granular location information such as city or zip code. The difficulty to anonymizing this type of data depends heavily on the level of granularity and type of trends in which the researcher is interested. If the researcher is interested in general trends, the data is easily anonymized, however this will tend to remove outlier information in the data. As such, studies that require or focus on outlier statistics may not be able to use anonymized data. Algorithms optimized for anonymizing data for research and analysts purposes could be low hanging fruit that despite their simplicity unlock huge potential value connecting private data with consulting firms without violating user privacy.

Figure 12.2: Applications that require personal information or data that cannot be well understand are hard to anonymize.

12.1.3 Difficult: Common Information

This type of data includes data that is particular to a person but not unique such as media preferences, health conditions, and places or named locations with which one has been or is associated. This type of data is commonly used by mobile-social applications, context-aware applications [11], and basic recommender systems. However, most of these applications benefit from data that differentiates users. Often, the information that best differentiates users (for recommendation engines for instance) is the information that is most particular to the user. When this is the case for an application it puts the utility of the application at odds with the anonymization process, leading to a rather difficult optimization problem. This is probably the type of anonymization that is most interesting from the research perspective due to its natural difficulty as a trade off between two competing interesting and due to the varied and specialized nature of context-aware applications and recommendation engines.

12.1.4 Near-Impossible: Unique and Incomprehensible Information

Certain pieces of information are absolutely unique to a user (e.g., social security numbers) and can not be released along with anonymous data. Note that names are not always (or often) unique identifiers as much of the worlds population shares many names. Furthermore, if we can't understand or comprehend a piece of data we can not measure how particularly it relates to an identity, as such we cannot meaningfully anonymize it. We could suppress, destroy or remove it, but we can not provide any anonymity guarantees.

In order to anonymize a piece of data while maintaining some of its utility, the data must be understood by the computer. GPS coordinates can be generalized because we can encode the spatial relationship between coordinates within the Geographic coordinate system. However, certain types of data can only be partially understood by computers. For example, while computers can identify and recognize faces in pictures, current technology may not be capable of fully understanding what is going on in a picture and as such cannot anonymize (or measure the anonymity of) such information. Furthermore, the anonymization system may not have important information necessary to understand a piece of data. Consider that even if a computer could recognize the actions and identities in a photo (Jon kissing Jane) it may not be able to

correctly infer Jon and Jane's relationship and how it relates to their interaction in the picture.

Chapter 13

Future Work & Suggested Research Directions

This section discusses a few interesting research questions and ideas that arose from the author's work on this thesis. In the process of designing and implementing a prototype application that used anonymized social data the author realized that social data needed to be treated differently to be anonymized which lead to the development of Social-K. While developing Social-K the author realized that much greater data utility could be preserved if users were queried in groups resulting in q-Anon. This section discusses the most recent findings and directions the author has found while working on sections 10 and 11 of this thesis.

13.1 Semantic Anonymity

While integrating the many different parts of AnonyGraph, the author realized that the level at which relationships must be defined for proper anonymization is the same level at which relationships are described by semantic markup languages such as RDF [19]. Semantic information is stored in efficient TripleStore databases [61]. Semantic data can be queried using the SparQL query language [15] [32]. Semantic information can be published as structured data using "Linked data" which allows machine-readable data to be automatically linked between databases around the world [14]. The possibility that anonymization could be defined and integrated within the very core of semantic web technologies is very interesting. Furthermore, many of the problems encountered while implementing anonymization for Facebook would be simplified within the real of the semantic web.

13.2 Anonymous Personas as PAI

Investigation of how different applications and sites use social data revealed that large web sites regularly share identifiable information about their users through back channels or sometimes through the browser [8]. Facebook has its "pre-approved" third party clause with which it shares a user's "PAI" or publicly available information which just so happens to require that PII (personally identifiable information) such as name and ID be included. This observation among others, has motivated the author and fellow researchers to consider how anonymized personas might be promoted as an alternative to the increasing practice of simply sharing identity by default.

13.3 Anonymizing Energy Data

With the governments recent support for the smart grid with both stimulus funding and regulations, millions of smart meters will be installed at residential locations over the next few years [59]. Among other things, smart meters provide fine grained records of power usage to the utility. These energy records can be used to identify how the energy is being used [45]. This could lead to utility companies understand their requirements better. However, it could also be used to what their customers are doing. Furthermore, if the energy records are considered property of the person, they may be released to other companies such as Google. As We have seen in the past, consumers may not understand the implications of releasing their energy records, rendering them public domain and accessible to other that they may not want knowing how they use their energy.

However, anonymizing energy traces from smart meters presents many unique research challenges. For one, the utility provider must have some way of knowing who to send the energy bill to and enough details to bill them correctly. Furthermore, energy traces contain information about the user encoded as recognizable patterns which are not so easily anonymized. Smart meter privacy will be a challenging and very important area in the coming years.

Bibliography

[1] Facebook. http://www.facebook.com.

[2] Facebook statistics. http://www.facebook.com/press/info.php?statistics.

[3] Farmville. Facebook.com/Farmville.

[4] M. Gartrell A. Beach and R. Han. q-anon: Rethinking anonymity for social networks. In 2nd International Conference on Social Computing (SocialCom 2010), 2010.

[5] R. Han A. Beach, M. Gartrell. q-anon: Practical anonymity for social networks. International Journal of Social Computing and Cyber-Physical Systems, 2011, to appear.

[6] A. Acquisti and R. Gross. Imagined communities: Awareness, information sharing, and privacy on the Facebook. In Privacy Enhancing Technologies, pages 36–58. Springer, 2006.

[7] Shane Ahern, Dean Eckles, Nathaniel S. Good, Simon King, Mor Naaman, and Rahul Nair. Over-exposed?: privacy patterns and considerations in online and mobile photo sharing. In CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems, pages 357–366. ACM, 2007.

[8] A.U. Asuncion and M.T. Goodrich. Turning privacy leaks into floods: surreptitious discovery of social network friendships and other sensitive binary attribute vectors. In Proceedings of the 9th annual ACM workshop on Privacy in the electronic society, pages 21–30. ACM, 2010.

[9] A Beach, M Gartrell, and Richard Han. Social-k: Real-time k-anonymity guarantees for social network applications. In IEEE International Workshop on SECurity and SOCial Networking (SESOC) at PerCom 2010, 2010.

[10] A. Beach, B. Ray, and L. Buechley. Touch me wear: Getting physical with social networks. In CSE '09: Proceedings of the 2009 International Conference on Computational Science and Engineering (Workshop on Social Computing with Mobile Phones and Sensors: Modeling, Sensing and Sharing 2009, associated with SocialCom 2009), volume 4, pages 960–965, Washington, DC, USA, 2009. IEEE Computer Society.

[11] Aaron Beach, Mike Gartrell, and Richard Han. Solutions to security and privacy issues in mobile social networking. In SMW09: Workshop on Social Mobile Web at SocialCom 2009 in CSE '09: Proceedings of the 2009 International Conference on Computational Science and Engineering, pages 1036–1042, Washington, DC, USA, 2009. IEEE Computer Society.

[12] Aaron Beach, Mike Gartrell, Xinyu Xing, Richard Han, Qin Lv, Shivakant Mishra, and Karim Seada. Fusing mobile, sensor, and social data to fully enable context-aware computing. In HotMobile '10: Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications, pages 60–65, New York, NY, USA, 2010. ACM.

[13] Aaron Beach, Baishakhi Ray, and Leah Buechley. Touch me wear: Getting physical with social networks. Workshop on Social Computing with Mobile Phones & Sensors: Modeling, Sensing and Sharing (SCMPS09) at SocialCom09 in IEEE International Conference on Computational Science and Engineering, 4:960–965, 2009.

[14] T. Berners-Lee. Linked data. International Journal on Semantic Web and Information Systems, 4(2), 2006.

[15] C. Bizer and R. Cyganiak. D2r server-publishing relational databases on the semantic web. In 5th International Semantic Web Conference. Citeseer, 2006.

[16] R. Cazabet, F. Amblard, and C. Hanachi. Detection of overlapping communities in dynamical social networks.

[17] US Congress. Library of Congress. Congressional Re.

[18] Landon P. Cox, Angela Dalton, and Varun Marupadi. Smokescreen: flexible privacy controls for presence-sharing. In MobiSys '07: Proceedings of the 5th international conference on Mobile systems, applications and services, pages 233–245, New York, NY, USA, 2007. ACM.

[19] S. Decker, S. Melnik, F. Van Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, and I. Horrocks. The semantic web: The roles of XML and RDF. Internet Computing, IEEE, 4(5):63–73, 2000.

[20] Josep Domingo-Ferrer and Vicenç Torra. A critique of k-anonymity and some of its enhancements. In ARES '08: Proceedings of the 2008 Third International Conference on Availability, Reliability and Security, pages 990–993, Washington, DC, USA, 2008. IEEE Computer Society.

[21] C. Dwork. Differential privacy. Automata, languages and programming, pages 1–12, 2006.

[22] C. Dwork and A. Smith. Differential privacy for statistics: What we know and what we want to learn. Journal of Privacy and Confidentiality, 1(2):2, 2010.

[23] C. Dwyer, S.R. Hiltz, and K. Passerini. Trust and privacy concern within social networking sites: A comparison of Facebook and MySpace. In Proceedings of AMCIS. Citeseer, 2007.

[24] N. Eagle and A. Pentland. Social serendipity: Mobilizing social software. IEEE Pervasive Computing, 4(2):28–34, 2005.

[25] N Ellison, C Steinfield, and C Lampe. The benefits of facebook "friends:" social capital and college students' use of online social network sites. Journal of Computer-Mediated Communication, 12(4), 2007. http://jcmc.indiana.edu/vol12/issue4/ellison.html.

[26] N.B. Ellison, C. Steinfield, and C. Lampe. The benefits of Facebook friends: Social capital and college students use of online social network sites. Journal of Computer-Mediated Communication, 12(4):1143–1168, 2007.

[27] Lujun Fang and Kristen LeFevre. Privacy wizards for social networking sites. In WWW '10: Proceedings of the 19th International World Wide Web Conference, 2010.

[28] P. Fišer and H. Kubátová. Two-level boolean minimizer boom-ii. In Proc. 6th Int. Workshop on Boolean Problems (IWSBP'04), Freiberg, Germany, volume 23. Citeseer, 2004.

[29] P. Fiser and D. Toman. A Fast SOP Minimizer for Logic Funcions Described by Many Product Terms. In Proceedings of the 2009 12th Euromicro Conference on Digital System Design, Architectures, Methods and Tools, pages 757–764. IEEE Computer Society, 2009.

[30] Charles M. Gartrell. Socialaware: Context-aware multimedia presentation via mobile social networks. Master's thesis, University of Colorado at Boulder, December 2008. `http://www.cs.colorado.edu/~rhan/Papers/Mike_Gartrell_CU_MS_thesis-final.pdf`.

[31] R. Gross and A. Acquisti. Information revelation and privacy in online social networks. In Proceedings of the 2005 ACM workshop on Privacy in the electronic society, pages 71–80. ACM, 2005.

[32] S. Harris and N. Shadbolt. SPARQL query processing with conventional relational database systems. In Web Information Systems Engineering–WISE 2005 Workshops, pages 235–244. Springer, 2005.

[33] K. Heyman. The move to make social data portable. Computer, 41(4):13–15, 2008.

[34] A. Kiryakov, B. Bishop, D. Ognyanoff, I. Peikov, Z. Tashev, and R. Velkov. The Features of BigOWLIM that Enabled the BBCs World Cup Website. 2010.

[35] R. Kling. What is social informatics and why does it matter? The Information Society, 23(4):205–220, 2007.

[36] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. Link Mining: Models, Algorithms, and Applications, pages 337–357, 2010.

[37] O. Lassila and R.R. Swick. Resource description framework (RDF) model and syntax. World Wide Web Consortium, http://www. w3. org/TR/WD-rdf-syntax.

[38] K. LeFevre, DJ DeWitt, and R. Ramakrishnan. Mondrian multidimensional k-anonymity. In Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on, pages 25–25, 2006.

[39] Kristen LeFevre, David J. DeWitt, and Raghu Ramakrishnan. Incognito: efficient full-domain k-anonymity. In SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data, pages 49–60, New York, NY, USA, 2005. ACM.

[40] Ninghui Li, Tiancheng Li, and Surech Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In Proceedings of the IEEE ICDE 2007, 2007.

[41] Tiancheng Li and Ninghui Li. Injector: Mining background knowledge for data anonymization. In ICDE '08: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, pages 446–455, Washington, DC, USA, 2008. IEEE Computer Society.

[42] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data (TKDD), 1(1):3, 2007.

[43] Ashwin Machanavajjhala and Johannes Gehrke. On the efficiency of checking perfect privacy. In PODS '06: Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, pages 163–172, New York, NY, USA, 2006. ACM.

[44] Justin Manweiler, Ryan Scudellari, Zachary Cancio, and Landon P. Cox. We saw each other on the subway: secure, anonymous proximity-based missed connections. In HotMobile '09: Proceedings of the 10th workshop on Mobile Computing Systems and Applications, pages 1–6, New York, NY, USA, 2009. ACM.

[45] P. McDaniel and S. McLaughlin. Security and privacy challenges in the smart grid. IEEE Security & Privacy, pages 75–77, 2009.

[46] Adam Meyerson and Ryan Williams. On the complexity of optimal k-anonymity. In PODS '04: Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, pages 223–228, New York, NY, USA, 2004. ACM.

[47] S. Muroga. Logic Design and Switching Theory. Wiley, New York, 1979.

[48] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In SP '08: Proceedings of the 2008 IEEE Symposium on Security and Privacy, pages 111–125, Washington, DC, USA, 2008. IEEE Computer Society.

[49] Netflix. Netflix api. `http://developer.netflix.com`.

[50] A.K. Pietilainen, E. Oliver, J. LeBrun, G. Varghese, and C. Diot. MobiClique: middleware for mobile social networking. In Proceedings of the 2nd ACM workshop on Online social networks, pages 49–54. ACM, 2009.

[51] Richard L. Rudell. Multiple-valued logic minimization for pla synthesis. Technical Report UCB/ERL M86/65, EECS Department, University of California, Berkeley, 1986.

[52] P. Samarati. Protecting respondents' identities in microdata release. IEEE Trans. on Knowl. and Data Eng., 13(6):1010–1027, 2001.

[53] S. Sapra, M. Theobald, and E. Clarke. SAT-based algorithms for logic minimization. 2003.

[54] S. Staab, P. Domingos, P. Mike, J. Golbeck, L. Ding, T. Finin, A. Joshi, A. Nowak, and R.R. Vallacher. Social networks applied. IEEE Intelligent systems, 20(1):80–93, 2005.

[55] L. Sweeney. Guaranteeing anonymity when sharing medical data, the Datafly System. In Proceedings of the AMIA Annual Fall Symposium, page 51. American Medical Informatics Association, 1997.

[56] Latanya Sweeney. k-anonymity: a model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10(5):557–570, October 2002.

[57] Amin Tootoonchian, Kiran Kumar Gollu, Stefan Saroiu, Yashar Ganjali, and Alec Wolman. Lockr: social access control for web 2.0. In WOSP '08: Proceedings of the first workshop on Online social networks, pages 43–48, New York, NY, USA, 2008. ACM.

[58] TM Truta and B. Vinay. Privacy protection: p-sensitive k-anonymity property. In Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on, pages 94–94, 2006.

[59] D. Von Dollen. Report to NIST on the smart grid interoperability standards roadmap. EPRI, Contract No. SB1341-09-CN-0031–Deliverable, 7.

[60] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. ACM Transactions on Information Systems (TOIS), 10(1):102, 1992.

[61] K. Wilkinson, C. Sayers, H. Kuno, D. Reynolds, et al. Efficient RDF storage and retrieval in Jena2. In Proceedings of SWDB, volume 3, pages 7–8. Citeseer, 2003.

[62] M. Wu and X. Ye. Towards the Diversity of Sensitive Attributes in k-Anonymity. In Proceedings of the 2006 IEEE/WIC/ACM international conference on Web Intelligence and Intelligent Agent Technology, pages 98–104. IEEE Computer Society, 2006.

[63] B. Zhou and J. Pei. Preserving Privacy in Social Networks Against Neighborhood Attacks. In Proceedings of the 2008 IEEE 24th International Conference on Data Engineering, pages 506–515. IEEE Computer Society, 2008.