1

The Social Web and Privacy: Practices, Reciprocity and Conflict Detection in Social Networks

1.1 Introduction

In the Social Web, the Internet and Web are coming into their own, offering an infrastructure for self-expression, information and communication to everyone regardless of their technical expertise. The Social Web comprises many facets: wikis, blogs, photographs/video/etc.
portals, tagging systems, etc. Most of these have functionalities that allow “social networks” of people with shared interests or other forms of interdependency to grow and become manifest by links such as blogrolls, comments, etc. One group of applications and portals has made this network building into its core purpose: social network sites (SNS). SNS are web-based services that allow individuals to (1) construct a public or semi-public profile within a bounded system, (2) articulate a list of users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the system [11]. SNS are not only popular, but also highly effective at turning otherwise often passive users into active contributors: In 2008, more than 30% of Internet users were members of at least one SNS, and more than 80% of SNS users became active network members [18]. The publishing of personal information in a network and the ease with which such information travels through different sites and beyond, allows providers and users to profile each other based on personally and relationally revealed data. Thus, on the one hand SNS exemplify functionality that permeates the whole Social Web, on the other hand they are prime examples of profiling functionality.

More recently, privacy has shifted into the focus of social-network researchers and practitioners. A common view is that SNS play an active role in the (general) ‘privacy nightmare’ of the Internet. Empirical analyses of SNS support this view, showing that vast amounts of data are collected, often without a clearly defined purpose, that privacy settings are cumbersome and their use poorly communicated, and that privacy setting defaults reveal a lot of information and (see [8] for a detailed analysis of 45 major SNS). Interestingly, millions of users appear to think otherwise, delighting in the new possibilities for self-expression, finding new friends online and sharing resources. Nevertheless, a number of those users also complain about unexpected revelations of their personal information and related privacy concerns. How can these views, which at first sight appear to be opposed, be considered together in order to help users, companies, and society at large to reach common understandings and working implementations of privacy protection in a world in which information sharing is a desirable daily practice?

We argue that a solution to this problem lies in studying different approaches to privacy. The dominant approach to privacy in computer science is to define privacy as data confidentiality – hiding data in an environment in which SNS act as drivers of the privacy-nightmare seducing users to disclose personal information. In this view, solutions consist of formal approaches to ensuring confidentiality through access control methods, data perturbation and other modifications of data to counter unwanted information inferences and leakages. We show later that this approach is not sufficient to address users’ privacy concerns in social networks, is often inappropriate in collaborative sharing environments, and is solely preemptive – most confidentiality and anonymity models do not engage with the information that has already been revealed or leaked. Hence, in this chapter we categorize complementary approaches to privacy and show how the approaches can be used to address the different types of concerns.

Once we have sketched out the different approaches, we explore in depth one of these in the context of social networks, an approach we call privacy as practice. In order to be able to do that we step out of the privacy nightmare discourse, assume that SNS are an interesting space on the Internet for engaging in privacy [2] and that therefore a detailed study of user behavior and concerns in them will yield a more accurate description of the privacy concepts that are relevant. Further, we presume that privacy is not something concrete, in consensus and in constant danger. Rather, we conceive privacy as a set of practices to negotiate what should remain public or private in social contexts. Legal and other regulatory frameworks and various social mechanisms exist to ensure that individuals can practice their privacy. We hence argue that we also need approaches to defining and
developing technology that target the same.

Further, SNS provide a prime example for studying the privacy and related concerns manifest on the (Social) Web: by virtue of being public and popular, SNS make evident privacy problems elsewhere on the Internet such as emails, discussion forums, chats, or e-Commerce. At the same time, in no other web applications are the user communities so actively involved in privacy debates although similar concerns apply.

The chapter is organized as follows. In Section 1.2, we review some dominant approaches to privacy and emphasize the importance of methods other than confidentiality and anonymity for privacy practices. There we also show how the present chapter and Chapters 15 and 18 of this book, which give detailed overviews of mechanisms for protecting profile privacy and methods for private analysis of networks, complement one another. We also shortly contextualize the three approaches by analyzing privacy concerns articulated by users and other stakeholders of social networks. Based on those concerns, in Section 1.3 we categorize the types of conflicts that arise among users as a result of two characteristic features of SNS: relational information and transitive access control. We then introduce the initial concepts for a method to detect these conflicts. In Section 1.4, we construct a formal model of the conflicts and describe our conflict discovery method. We then apply our method to four cases which are typical in SNS and discuss their differences. In Section 1.5, we suggest that in order to negotiate the conflicts identified using our method, data-mining and feedback techniques as well as access control alternatives can be used. The method hence not only serves to assemble requirements and study interactions between those requirements; further, by showing outcomes and possible conflicts, it suggests how data-mining can be the core of awareness tools that help users better oversee consequences of their actions. An outlook is given in Section 1.6.

In the work presented here, the role of data mining changes. Today data mining is often seen as a technology that is at the core of privacy concerns and at the same time is the starting point of a group of methods (“privacy-preserving data mining”) that help avoid these. In this chapter, we show how data mining methods and technologies may also inform individuals and groups about the (possible) consequences of various privacy-related behaviors.

Finally, it is not the objective of this chapter to propose new access control models that limit how information can travel according to some specification. Rather, we develop a method to investigate the consequences and conflict potential of information travel resulting from common SNS designs. These detected conflicts can be used to elicit requirements for solutions – these requirements may in turn be useful in designing access control models.

### 1.2 Approaching Privacy in Social Networks

In this section we motivate and define different approaches to privacy. In order to do that, we start from the way the privacy problem is currently framed in privacy debates and in data protection legislation. We then proceed to describe the three relevant privacy approaches and the privacy definitions they rely on. For each approach we also state their advantages and disadvantages. Later, we return to the data question by identifying the privacy-relevant data most characteristic of social networks.

#### 1.2.1 Data I: Personal data

Since computers are about data and data processing, any concept of privacy in computational environments will concern (centrally or also) data, in particular “personal data”.
Personal data is “any information relating to an identified or identifiable natural person [...]”; an identifiable person is one who can be identified, directly or indirectly, in particular by reference to an identification number or to one or more factors specific to his physical, physiological, mental, economic, cultural or social identity” (EU Directive 95/46/EC [15], Art. 2 (a); this is defined in a similar way in other legal contexts). Notice the emphasis on identity, which is assumed to be unique and identifiable for one natural person; in line with this emphasis, US terminology talks about personally identifiable data. The standard types of personal data are profile data describing individuals, including name, address, health status, etc.

1.2.2 Privacy as hiding: Confidentiality

In one of its historical moments, privacy has been defined as “the right to be let alone” [50]. Although originally formulated as a right that protects individuals against gossip and slander, this construct has since then acquired a wider meaning. Namely, it refers to an individualistic liberal tradition in which an intrinsic pre-existing self is granted a sphere of autonomy free from intrusions from both an overbearing state and the pressure of social norms [39]. This definition has also been popularly used by computer scientists and has been interpreted as an autonomous (digital) sphere in which the data about persons is protected, outside of which the data remains confidential.

Data confidentiality – the protection of data from unauthorized access— is a strong and useful translation of such privacy concerns into digital space. After all, once data about a person exists in a digital form, it is very difficult to provide individuals with any guarantees on the control of that data. Data collected using current day technologies represent activities of individuals in social life that for many are assumed to be private. To preserve privacy, in that sense, is then to keep this data private, in other words confidential from a greater public. Not exchanging any data would preserve privacy but is inconvenient and probably also not desirable. Therefore, a lot of the privacy research in computer science is concerned with other forms of partial confidentiality, like anonymity.

Anonymity is achieved by unlinking the identity of the person from the traces that her activities leave in information systems. Anonymity keeps the identity of the persons in information systems confidential but is not necessarily concerned with how public the traces subsequently become. This is also reflected in data protection legislation which by definition cannot and does not protect anonymous data [21].

In technical terms, anonymity can be based on different models. In communications, anonymity is achieved when an individual is not identifiable within a limited set of users, called the anonymity set [38]. An individual carries out a transaction anonymously if she cannot be distinguished by an observer from others in that set. The observer, often also called the adversary, may obtain some additional information [12]. Depending on the observer’s capabilities, different models can be constructed with varying degrees of anonymity for the given anonymity set. Exactly what degree of anonymity is sufficient in a given context is dependent on legal and social consequences of a data breach and is an open question [12].

In databases and SNS, the conditions for establishing anonymity sets and the targeted objectives are different. Anonymity is a popular requirement when (SNS) data are to be analyzed (e.g. data-mined), especially when this is done by third parties. One difference to communications anonymity is that methods to anonymize databases aspire to protect the utility of the anonymized data for analysts. More specifically, it is an objective of these methods to provide the analyst with data that allow the inference of certain information, while forestalling the inference of certain other information (information that could lead to
privacy breaches). This type of research is called privacy preserving data mining (PPDM). PPDM research on relational databases as well as network data has shown that simple anonymization approaches do not work, because ill-meaning or even unsuspecting analysts (“adversaries”) may, through additional information based on the network structure, recover the supposedly-unlinked identities and/or find more information about these data subjects [46, 28, 30, 42, 13]. In Chapter 18 of this book, Hay, Miklau and Jensen give a detailed account of opportunities and challenges for network anonymization, again with the additional goal of preserving the utility of the network data.

These results demand that the scope of what is defined as ”personal data” be expanded. The expansion of the definition of personal data raises important research questions about the usability, efficiency, utility and practicability of privacy-as-confidentiality methods in social networks and in general. Further, where it is difficult to make confidentiality or anonymity guarantees, other approaches that address the privacy concerns raised may be considered. In the following two sections we motivate and describe two such approaches.

1.2.3 Privacy as control: Informational self-determination

A wider notion of privacy, appearing in many legal codifications, defines the term not only as a matter of concealment of personal information, but also as the ability to control what happens with it. One reason for this notion, which does not call for strict data parsimony, is that the revelation of data is necessary and beneficial under many circumstances – and that control may help to prevent abuses of data thus collected.

This idea is expressed in Westin’s [51] definition of (data) privacy: “the right of the individual to decide what information about himself should be communicated to others and under what circumstances” and in the term informational self-determination [9]. Informational self-determination is also expressed in international guidelines for data protection such as the OECD’s Guidelines on the Protection of Privacy and Transborder Flows of Personal Data [35], the Fair Information Practices (FIP) notice, choice, access, and security [49, 19], or the principles of the EU Data Protection Directives [15, 16]. As an example, consider the principles set up in the OECD Guidelines: collection limitation, data quality, purpose specification, use limitation, security safeguards, openness, individual participation, and accountability.

An important class of tools to exercise (more) control over personal data are called Identity Management Systems (IDMS). There are different types of identity management systems, but here we refer to mechanisms that support separation of context-dependent virtual identities represented by pseudonyms of varying strength. IDMS allow individuals to establish and secure identities, describe those identities using attributes, follow the activities of their identities, and delete identities. They are often based on credentials, policies and access control methods.

In SNS an essential part of the configuration of ones profile is who gets to access what; and these whos are specified not on the basis of identity (only Mr. Ali gets to see my health data) or individual attributes (only people who are doctors get to see my health data), but on relationships, or in other words, the topology of the network (only my friends get to see my health data or only my friends and their friends get to see my health data). Thus, social network sites are in that sense identity management tools, which in turn are tools that implement the idea of privacy as control through topological vicinity.

A simple form of such access control defines access based on the path distance from the node that owns the data. Popular access models in current SNS comprise “friends” (only nodes one hop away from the data owner may see that profile) and “friends-of-friends” (only nodes at most two hops away may see the profile), or in some cases, friends at a longer path
length [8]. An overview of current work related to identity management in social networks and a detailed description of different access-control models that protect a user’s profile or relationships is given by Carminati, Ferrari, Kantarcioglu and Thuraisingham in Chapter 15 of this book.

Although informational self-determination principles are desirable, suggesting that individual control is always possible, desirable and effective can be misleading. This is the case for a number of reasons. First of all, collection limitation in one system does not protect against the aggregation of that data in many systems. Monitoring all revealed information in SNS may be overwhelming where the number of data controllers increase exponentially. A user may be overwhelmed by the difficulties of individual participation and unable to judge the risk of revealing information or using automated agents for such decision-making. Further, even if all these principles were implemented, it would be very difficult to identify violations. In the case of trusted parties, system security violations (i.e. hacked systems) or design failures (i.e. information leakages) or linking of different sources of safely released data may cause unwanted release of information. Last, data protection and most IDMS, by focusing on individual and identifiable data provides little protection with respect to aggregation of anonymized data, profiling based on correlations and patterns found in this aggregated data, and the consequent desirable or undesirable discriminations. In addition, privacy as control is an abstract concept that does not consider how people actually do and want to construct their identities. Hence, further approaches are needed, which assist in establishing an identity management practice. This is the topic of privacy as practice, to which we turn next.

1.2.4 Privacy as practice: Identity construction

Despite interesting research results in the area of privacy preserving methods and tools, individuals are daily confronted with the collection of massive amounts of data about them. This could be due to the market interests in collecting information; the lack of popular and usable privacy enhancing technologies; surveillance technologies that collect information on a mass level without consent; contexts in which identification is central to the services provided (i.e. in hospitals or in employment situations); or, simply due to the desire of individuals to reveal information about themselves with their names etc. By privacy as practice, we refer to the definition of the right to privacy as the freedom from unreasonable constraints on the construction of one’s own identity, be it by strategically being able to reveal or conceal data. This approach requires domain specific and sociological analysis of users’ and communities’ information revelation and concealment needs as in the recent examples given in [6, 7, 23]. Such diversity of user concerns resulting in or in tension with privacy are often not emphasized in the privacy-as-confidentiality and privacy-as-control approaches.

Privacy as practice demands the possibility to intervene in the flows of existing data and the re-negotiation of boundaries with respect to collected data. These two activities rest on, but extend the idea of privacy as informational self-determination in that they demand transparency with respect to aggregated data sets and the analysis methods and decisions applied to them. In this sense, these approaches define privacy not only as a right, but also as a public good [25].

Sociologists have investigated the idea that privacy is (social) practice from various viewpoints. Linking these discussions to the discussions on privacy on the Internet and concrete PETs, Phillips [39] distinguishes between four kinds of privacy (which are not mutually exclusive). These comprise the above-mentioned right to be let alone and the possibility of separating identities. The third type is the construction of the public/private divide. This
distinction concerns the social negotiation of what remains private (i.e. silent and out of the public discourse) and what becomes public. For example, in the case of voting, individuals may want to keep their choice private, and in the case of domestic violence, interest groups and individuals may have an interest in defining the “domestic” as a public issue. The fourth type of privacy is the protection from surveillance. In this context, surveillance refers to the creation and managing of social knowledge about population groups. This kind of privacy can easily be violated if individual observations are collated and used for statistical classification, which applied to individuals makes statements about their (non)compliance with norms, their belonging to groups with given properties and valuations, etc. Market segmentation is an example of the classification of population groups which may constitute a breach of this kind of privacy.

In a similar fashion, Palen and Dourish argue that “privacy management in everyday life involves combinations of social and technical arrangements that reflect, reproduce and engender social expectations, guide the interpretability of action, and evolve as both technologies and social practices change” [37]. Boyd and Ellison state that privacy in social networks sites is also implicated in users’ ability to control impressions and manage social contexts [11].

These definitions emphasize that confidentiality and individual control are part of privacy, but not all. Privacy includes strategic concealment, but also revelation of information in different contexts, and these decisions are based on – and part of – a process of collective negotiation. Tools should therefore support data concealment and revelation to help individuals practice privacy individually and collectively.

For example, Lederer et al. [29] suggest improving privacy sensitivity in systems through feedback that enhances users’ understanding of the privacy implications of their system use. This can be coupled with control mechanisms that allow users to conduct socially meaningful actions through them. These ideas have led to suggestions like the identityMirror [31] which learn and visualize a dynamic model of user’s identity and tastes. A similar approach is suggested in the concept of privacy mirrors [33]. The authors criticize purely technical privacy preservation solutions that do not take the social and physical environments in which the technical systems are embedded into consideration. Making the collected data visible would make the underlying systems more understandable, enabling users to better shape those socio-technical systems, not only technically, but also socially and physically. A first implementation of privacy mirrors exists in Facebook through which users can set controls on their profile information and then check how their profile is seen by their friends, but not by non-friends.

Hansen’s [24] proposal for linkage control in IDMS is a further example of these ideas. In her proposal to extend and improve the user experience of IDMS she suggests mechanisms that provide information about collected data to individuals and the general public. These mechanisms include informing users on possible and actual linkages, as well as de-linking options; communicating privacy breaches to individuals concerned; documenting the sources of data and algorithms used by data controllers as well as the recipients of analyzed data; making accessible personal data and also other data suitable to affect individuals; and providing effective tools to intervene in data linkages in order to execute corrections or deletions.

Social networks as a locus of Privacy as Control and Practice

We frame existing web-based social networks services as one of the first massively adopted IDMS (privacy as control) with the potential of providing insight into how improved practices can be developed (privacy as practice). SNS provide numerous and rich examples of
user-provider-negotiation with respect to privacy. This can be anywhere from the privacy settings which have evolved immensely in the last two years, via the introduction of usable and integrated privacy policies, to the introduction of some simple forms of “privacy mirrors” as a standard feature etc. The latter are exemplary of tools designed to raise awareness, on which we from now on we will focus.

Summing up, all three approaches to privacy: privacy as confidentiality, privacy as control and privacy as practice can and should be used in developing systems. Each type of privacy requires different kinds of research, the compositionality of which also demands further study.

In this Chapter, we emphasize the view of privacy as practice, investigate social networks as a locus of such practices, and describe methods to better support the identified processes and requirements.

1.2.5 Privacy in Social Network Sites: Deriving requirements from privacy concerns

Before we review the definition of personal data and suggest ways in which privacy as practice may be developed, we shortly summarize the results of a media study on privacy concerns that may or have lead to privacy breaches in SNS. Our objective is to understand what the privacy concerns are, how the underlying principles and design of SNS lead to these concerns, and how a practice around these problems can be developed. To understand privacy concerns in today’s SNS, we combined a literature study of a wide range of computer science, sociological and legal approaches, a study of large SNS’ design and privacy-strategy communications, and a manual content analysis of news and blogs during a time of intensive discussion of Facebook’s and Myspace’s privacy practices. The result was a categorization of privacy breaches in SNS into the following [22]:

Indeterminate visibility denotes the problem of a user’s profile information being visible to others without the user’s explicit knowledge or approval. Separation of digital identities and aggregation Separation of digital identities refers to the construction of social identities by individuals that selectively reveal and conceal information in specific contexts and roles [39, 34]; aggregation is the de-separation of these identities. Contested ownership: are explicit and implicit definitions of data ownership on SNS that may lead to privacy breaches.* Misappropriation is the use of SNS data out of context or for previously undefined purposes.

1.2.6 Data II: relational information and transitive access control

In the reports on privacy breaches in SNS that we analyzed in [22], a common theme arose throughout the different types of breaches: many concerns were raised about the practices in SNS with respect to what we term relational information and transitive access control.

In short, relational information (RI) is data that, unlike profile information, is regarded as

*While we called the third breach category “contested ownership”, we would like to point out that this term was chosen to reflect the wording used in the materials, and that we do not claim to make any statements about the associated legal concepts. To motivate our choice of terminology and put it into context, we will discuss the more general question of how a resource pertains to a person (such that the person has or should have “something to say” about its use). In this chapter, we cannot solve the legal issues, and we therefore avoid the term “ownership” and its derivatives wherever possible, preferring the (intentionally) underspecified “belonging to” or “related to”.
“belonging” or “related” to more than one user; for example, the information representing the “friends relationship” between A and B may be said to belong to both A and B. With relational information, both A and B have some control rights on that information. Which rights these are depends on if both have equal permissions on the relational information – as is the case with a friends relationship – or, if there are differences in the permissions distributed to the various related parties. The latter can, for example, depend on whose profile the information is placed: the owner of the profile usually has more rights than those who post information on that profile.

Transitive access control (TAC) refers to the fact that by defining topology based access control based on path lengths greater than or equal to two, a profile A allows her friends to co-define who can access her information; for example, if A specifies that “friends of friends” may see his profile, then a friend B – by her choice of friends – co-defines who may access A’s profile. Hence, effectively A’s friends are assigned the right to give and revoke to their friends access permissions to A’s selected objects. There again is a hierarchy in the permissions, a friend B can grant and revoke permissions to their friends that are not already assigned permissions by A. Further, if the access path length defined by A is $p > 2$, then friends at path length $p - 2$ are able to delegate the permission to co-define the access list of A to their friends.

There are various reasons why RI and TAC are useful for the different stakeholders of SNS. First, TAC makes it easier to share resources by allowing friends of a certain path length to grant further profiles in the network access to that resource. The TAC model hence overlaps with the SNS providers interest in achieving maximum sharing.∗ Second, many of the information objects in SNS, even at the time of creation, are related to many. The combination of TAC and RI addresses this problem by giving all users related to an information object access and some control permissions. It enables information objects to be collectively managed by multiple profiles who are granted a limited set of control permissions throughout the information objects’ lifetime. Third, TAC provides a collaborative solution to deal with the expansion and contraction of the network and maybe even a solution to deal with issues of privacy collectively.

Yet, while RI and TAC are cornerstones of the attractiveness of social networks for users, they also give rise to numerous privacy breaches [22]. Ownership of RI may be contested in various ways (what can be decided by A, what by B, and what happens when they grant and revoke different access rights with respect to a relational information object?), and TAC may lead to a wide and indeterminate visibility of (profile or relational) information. This usually has a consequence for any functionality to keep identities or audiences separated.

In addition, different profiles may have different preferences on how the consequences should be handled. The misappropriation of RI is a distinct and commercially attractive possibility in SNS whose business model rests on the “network value of customers”. Misappropriation may for example occur if, in order to reveal relational information to third parties, it is sufficient for one of the related profiles to agree to the third party’s terms. Finally, misappropriation may also occur because deletion of relational information is complicated as long as one or more of the related profiles want to keep their relational information and hence the integrity of their profiles.

Analyzing privacy concerns in SNS only with respect to “personal data” belonging to individual users falls short of analyzing many of the problems arising from relational infor-

∗A recent user study of Facebook users show that 87% of the users have default or permissive privacy settings [45] which reinforces privacy salience practices by the providers of SNS [8]
information and TAC. This is also a problem for existing data protection legislation which is focused on classical definitions of personal data and has no terms dealing with relationality of data or with the transparency of data collections and data mining methods.

As a step forward, in the next section we will model both of these concepts in detail. We will then use that model to develop a method with which we can identify types of conflicts that lead to privacy concerns. Later, we will discuss possible solutions to these conflicts using privacy awareness raising data mining techniques.

1.3 Relational Information, Transitive Access Control and Conflicts

Relational information objects raise a number of requirements and design questions: Which of the data subjects or controllers can access, distribute or delete such information? What conflicts may arise when relational information and Transitive Access Control are applied in an SNS, and how can these conflicts be systematically detected prior to implementation? How can these conflicts be addressed? In addition to keeping attributes of profiles confidential, as discussed in Chapter 15, what alternative solutions may data mining techniques offer for users of SNS? To answer these questions, we first define the notions in more detail.

1.3.1 Transitive Access Control and Relational Information

Access control in information systems defines which principals (persons, processes, machines, ...) have access to which resources in the system – which files they can read, which programs they can execute, how they share data with other principals, and so on [3]. Often a distinction is made between mandatory and discretionary access control. Mandatory access control occurs when there is a central instance defining access rules, where resources have trust levels that they require and principals have trust levels that they can prove – also called label-based access control. Mandatory access control is limited in responding to the needs of systems with dynamic principals and resources.

Under certain conditions, SNS make use of discretionary access control models. The specifics of these conditions are described below. Discretionary access control is a mode in which the creators or “owners” of files assign access rights, and a subject with a discretionary access to information can pass that information on to another subject [17].∗ In [27] the authors make a distinction between access permission and control permission. Access permission is the permission of a principal to access a resource in some manner. Control permission allows principals to grant or revoke access permissions. Sometimes, this includes the ability to pass a subset of the available control permission to other users [27].

Transitive access control is the type of discretionary access control that is activated in SNS when users grant and revoke both access permissions and a subset of their control permissions to others. Transitive access control is used when users manage relational information or allow friends-of-friends, or friends at longer path lengths to access their information objects.

We call each information resource of a profile in SNS an information object in that profile.

∗TCSEC (1985) defines discretionary access control without mention of “ownership”, but rather as “a means of restricting access to objects based on the identity of subjects or groups, or both, to which they belong. The controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission (perhaps indirectly) on to any other subject [...].”
If an information object starts to be controlled by more than one profile during its lifetime, it becomes a \textit{relational information object}. Some information objects have multiple controllers during initialization, i.e., they are relational information objects at creation.

Bidirectional relationships and commented photographs are two illustrations of relational information objects in SNS. In an SNS, profiles establish relationships with each other. In some SNS, these relationships are bidirectional: both profiles agree and then the friendship relationship is established. In SNS with bidirectional relationships between profiles, each friendship relationship between two profiles, by virtue of both profiles having control permissions on the relationship, is a relational information object from its conception. A photograph of a profile \(A\), once commented by another profile \(B\), becomes relational information, since by commenting, \(B\) henceforth receives permission to grant her friends of a desired path length access to both her comment and the photograph.\(^\ast\)

In our definition of relational information, we focus on explicit relatedness. We disregard the less clear-cut cases where the establishment of relatedness is technically difficult although legally relevant, such as the data subjects of a photograph published in an SNS.

**1.3.2 Inconsistency and Reciprocity Conflicts with TAC and RI**

Privacy concerns with respect to relational information and transitive access control, and resulting privacy breaches, have been expressed by users and experts in the media and the blogosphere as we discussed briefly in Section 1.2. With respect to the underlying design and configuration of TAC, two types of conflicts can occur among profiles that may lead to the articulated privacy concerns: inconsistencies among users’ TAC settings and reciprocity conflicts.

- **Conflicts due to inconsistency of related profiles’ TAC settings:** Inconsistency conflicts occur among profiles related to a relational information object \(R\) if there is a discrepancy between the privacy settings of a profile related to an information object \(R\), and the visibility of \(R\). Unless all profiles related to an RI have the same friends and the same TAC settings, inconsistency conflicts will occur among the related profiles with respect to the access permissions of the RI. As an example, consider that profile \(A\)'s friends are \(\{C, D, E\}\), and \(B\)'s friends are \(\{C, F, G\}\). If both \(A\) and \(B\) have set their access permissions to “friends” and then enter a relationship, the new relationship between \(A\) and \(B\) becomes visible to the union \(\{C, D, E, F, G\}\), a set greater than their respective set of friends. This leads to an inconsistency between the permissions granted by each profile and the actual visibility of their relationship information.

- **Conflicts due to lack of reciprocity among profiles’ access to information objects:** Differing privacy settings with varying sets of friends may be desirable in SNS. Hence, inconsistency among profiles’ TAC settings may be

\(^\ast\)When we first modeled RI in Facebook, by commenting a picture, a profile could simply disseminate the commented picture indefinitely. Recently Facebook has changed their privacy settings and has limited the control rights that a commenting profile is granted. As a result, now there is a hierarchy between the owner’s privacy settings and the privacy settings of those profiles who post information objects on the owner’s profiles, e.g., comments, photos, links. There are many varieties of how a hierarchy of permissions to control relational information are distributed on Facebook. The conflict detection model that we develop can be adapted to also address conflicts arising from the hierarchy of control permissions with respect to relational information objects.
acceptable to a certain level. In order to determine the acceptable transitive access control practices, feedback and transparency is necessary with respect to the effects of TAC. To increase transparency, the profiles related to a relational information object $r$ may be provided with feedback about the collective effect of their privacy settings, e.g., profiles related to $r$ should be informed about those profiles $r$ is visible to. However, such feedback can only be implemented if those profiles that $r$ is now visible to agree to being reported in the feedback mechanism. We see such an agreement to flow of information to provide feedback and hence transparency as a form of reciprocity.

The idea of reciprocity is derived from the theory of social translucence. Social translucence is an approach to designing digital systems that emphasizes making social information visible to the participants of a system. It is mapped to digital systems through the use of “social proxies”, minimalist graphical representations of the online presence and activities of people. Specifically, these systems work with the concepts of visibility, awareness and accountability [14]. The assumption is that if socially significant information about a person’s actions are visible to others using the system, then this raises awareness of that profile’s actions and enables social accountability. For social networks, social translucence can be applied through different degrees of reciprocity in information flow between profiles. For the time being, we distinguish between strong and weak reciprocity to increase social translucence in social networks.

- **Strong reciprocity**: is guaranteed in SNS when a profile $A$ can access profile $B$’s information, and that accessibility also holds in the opposite direction. For example in Facebook this usually holds when a relationship is established between two profiles: both profiles can access each other’s related information objects, e.g., profile information, list of friends relationships. SNS allow profiles to define strong reciprocity to only a subset of their profile information. For example, when a relationship is established strong reciprocity with respect to friends lists may hold, but this may not be the case for pictures. Further, in most SNS strong reciprocity is not guaranteed with friends of a greater path length. If the users expect strong reciprocity for their profile information and this is not guaranteed through the access control model then we say that there is a strong reciprocity conflict.

- **Weak reciprocity**: consists of $A$ knowing when a profile $B$ is a controller of some of her relational information, i.e., a co-controller, or when $B$ can access her profile information. Weak reciprocity with respect to co-controlled is available on Facebook but this is not always transparent, i.e. the ability of other profiles to set access permission to relational information is not always presented as such to the related profiles. In comparison, weak reciprocity with respect to who can access a profile’s information is not guaranteed in Facebook, but is a functionality available on Flickr. On Flickr relationships are not bidirectional and profiles can indicate when they are ‘following’ a profile. A feedback system that informs profile $A$ of the set $S$ of profiles who have access to her relational information $r$ can be implemented only if “weak” reciprocity is available. Not all members of the feedback set $S$ may agree with being listed in a feedback mechanism. We then say that there is a weak reciprocity conflict. Hence, depending on the degree of reciprocity
desired by the profiles, different types of conflicts may occur.∗

Depending on the privacy settings of different profiles and the way TAC is implemented, inconsistency and reciprocity conflicts may occur. Inconsistency conflicts and their unwanted consequences are pervasive in today’s SNS. For example, in a German social network with 120,000 users, more than two thirds of the users’ hidden relationships were disclosed to the public due to unilateral friendship disclosure by their immediate friends [41].

We argue that if the design options with respect to relational information and transitive access control can be handled during requirements engineering, such that the different inconsistencies and reciprocity conflicts are discussed with the different system stakeholders, then some of the privacy breaches mentioned in Section 1.2 can be avoided. Depending on the different stakeholders’ requirements, alternative TAC and relational information models can be designed.

Further, data mining tools can be used to filter and summarize the feedback to avoid information overload due to the application of social translucence. Data mining tools can hence improve users’ awareness of information distribution in order to support their privacy practices and negotiations. Data mining tools can only be implemented when weak reciprocity conflicts do not occur. After all, if profiles do not want to be mentioned in feedback systems (do not agree to weak reciprocity), then tools that mention their profile and access information to others would lead to a violation of their privacy needs.

In the following, we introduce the notation we use to describe RI and TAC. We then develop a method with which inconsistencies and reciprocity conflicts based on the RI and TAC design can be discovered. The output of this method can be used during requirements engineering to elicit privacy constraints towards the design of RI and TAC. As an example, we analyze four alternative TAC models with respect to one of the articulated privacy concerns: indeterminate visibility. We show how the possible solutions interact with the functional requirements of sharing information and enabling social networking.

1.3.3 Formal Definitions

In this section, we formalize SNS, access and control permissions, as well as inconsistencies and reciprocity conflicts. SNS are typically modeled as a graph. Commonly, when graphs are used to model SNS, each node stands for a user’s profile and all the information objects contained therein, while the set of edges stands for the relationships between the profiles. In such models, it is assumed that every information object is related to and controlled by a single profile, hence each profile \( p \) implicitly represents the information related to \( p \). The problem of who controls a relationship is often solved by modeling relationships as unidirectional: the source of each edge is the controller of the relationship information (see the graph on the left in Figure 1.3.3).

Since the focus of our study are relational information objects and the conflicts that arise as a result of the permissions attributed to them, we model information objects and use edges to indicate the profiles they are related to. Each information object, be it the relationship between two profiles, a photo or a comment, is represented explicitly by a node in the graph (See the graph on the right in Figure 1.3.3). Each edge models the relatedness

∗It is always possible to expect reciprocity of higher orders recursively: if \( B \) knows that \( A \) can access her profile, then \( A \) should know that \( B \) knows that \( A \) can access her profile. This is a typical problem of auditing the audits or awareness of awareness. For now we are only concerned with “first order” reciprocity.
FIGURE 1.1  Graphs for modeling SNS. The graph on the left models all information as belonging to a single profile. The graph on the right explicitly shows all the information objects related to a profile. The boxes with the label $r_{x,y}$ are nodes that model friendship relationships explicitly. The dotted lines between information objects $o_{1,2}$ and $o_{2,1}$ indicate that the two information objects are related e.g., a photo and a comment.

of that information object to other profiles* or other information objects. We assume that all edges are bidirectional. A profile node in our graph does not represent all the information that a user has put online, but rather is a root node that functions as a principal of all the related information objects. Next, we provide a formal definition of the graph we are using.

*From now on we refer to a “profile” in an SNS rather than to a “user”. This is for a number of reasons: firstly, a user may have multiple profiles; secondly, the profile of a user is not the user herself, but a set of information objects she collects, creates and distributes; and last, profiles may have no user behind them i.e. attacks on social networks may consist of injecting spam profiles into the network that may have serious implications for the functionality as well as the security of a social network.
The Social Web and Privacy: Practices, Reciprocity and Conflict Detection in Social Networks

information object, and not the right to edit, execute or delete an information object.

• control permissions: A principal with control permissions with respect to an information object o has access to o∗, and can grant and revoke additional access permissions by establishing or deleting relationships. In some cases, control permission also includes the permission for a principal to delegate the right to grant access permission to o to other profiles.

We assume that all profiles may create new information objects. The creators of an information object have control permissions for that object. The creator may permit other profiles to relate textual or media comments to existing information objects. Such information objects are then composed and become relational information objects. The related profiles then receive further control permissions depending on the underlying design.

Deletion and disconnection from information objects is an important functionality and a complicated matter in social networks. With relational information, deletion becomes a locus for conflicts and raises a number of privacy concerns. For now, we do not consider commenting or deletion in our models. Hence, in our simple model we assume that once control permissions are distributed, these can no longer be changed. Further, since we do not consider deletion, access permissions can be granted to further profiles, but revocation is not possible. We introduce the following notation in order to express the TAC model in SNS:

• C(o) and A(o) both subsets of P respectively denote the set of profiles that have control and access permissions on an information object o ∈ O.

• F(p) returns the set of profiles that established a relationship with p. For a subset of the profiles Q ⊆ P, F(Q) returns the union of F(p) for each p ∈ Q. p is not included in F(p) so that we can distinguish between the different sets “friends” and “friends-of-friends”. The union of those two sets is denoted with FoF(p).

• If p ∈ C(o) for some information object o, then it can assign further access or control permissions to that information object. The union of all the access permissions assigned to an object by profiles in C(o) define who can access o. S(p, o) returns the permissions a profile assigns to an information object o. The permissions return the tuple (controlPermission, accessPermission), both consisting of sets of profiles. Each permission set may vary between C(o) and P. An information object o is private if S(p, o).controlPermission = S(p, o).accessPermission = p. We also add a convention in order to keep profiles with no access permissions from being able to grant or revoke control permissions:

\[ S(p, o).controlPermission \subseteq S(p, o).accessPermission \subseteq P \]

• T(p) refers to all the permissions set by a profile p, i.e., \( T(p) = \{ S(p, o) : p \in C(o) \} \).
• s, t ∈ N are variables that state the length of a path in a graph.

*Note that when designing systems, ‘separation of duties’ may be a security concern. In such cases, it is possible to have control permissions for an information object and yet no access permissions. If separation of duties is a desirable requirement for the security of the social network, the definition of control permissions will have to be changed.*
In our simple SNS model, we concentrate on one type of relational information, namely, relationships themselves. Relationships are information objects that become relational as soon as they are established i.e., \( C(r_{ij}) = \{p_i, p_j\} \). The set \( C(o) \) of a relationship is static and never gets extended. Both profiles in \( C(r_{ij}) \) have the right to grant access rights to further profiles for relational information \( r_{i,j} \). We plan to study further types of relational information and address conflicts with respect to the assignment of hierarchical control permissions in future work.

**The graph:**

We can now construct our SNS graph \( G = (V, E) \) in which relationships and profiles are modeled explicitly using labels:

- \( V \): the vertices in \( G \) can be labeled in two different ways:
  - there is a labeling function \( l_{V,p} \) which maps some of the vertices to labels in \( P \).
  - there is a labeling function \( l_{V,r} \) which maps some of the vertices to labels in \( R \).
- \( E \): edges connect vertices labeled with an element of \( P \) and vertices labeled with an element of \( R \). When two profiles \( p_i \) and \( p_j \) establish a relationship, a vertex \( r_{ij} \) and edges \((p_i, r_{ij})\) and \((p_j, r_{ij})\) are inserted to the graph. The edges are undirected. If two profiles establish a relationship, then a path of length two exists between them.

**FIGURE 1.2** SNS graph \( H \). Each user is represented by a profile and her relationships.

**Example:**

Figure 1.3.3 is a depiction of a graph \( H \) which only includes profiles and relationships as information objects. The profiles in \( H \) are \( P = \{p_1, p_2, p_3, p_4\} \), and the relationships are \( R = \{r_{12}, r_{13}, r_{23}, r_{14}\} \). The \( C(r_{ij}) \) for each of the relationships are the relationship partners. So far, we have not represented permissions for information objects in \( H \).

If permission inconsistencies exist and are acceptable, and awareness mechanisms are desirable, then it is also necessary to determine existing reciprocity conflicts.

- **Reciprocity Conflicts:** In the case of relationships and profiles, the two information objects that we model, we distinguish between two kinds of reciprocity conflicts:
The Social Web and Privacy: Practices, Reciprocity and Conflict Detection in Social Networks

- **strong reciprocity conflicts**: Let us define a function that has a parameter `Scope` that determines the profile information objects that have to be reciprocated in the SNS. `Reciprocate(p, Scope)` returns all information objects `o ∈ O` related to profile `p` in the set `Scope`. `Scope` in our model can specify the profile itself and the friendship relationships of a profile. For example, `Reciprocate(p, R)` returns the set of all `r ∈ R` such that `p ∈ C(r)`. Strong reciprocity conflicts between two profiles `p_1` and `p_2` exists, if access to profile information objects defined by `Reciprocate(p, Scope)` has no mutuality. Concretely, if for the given `Reciprocate(p, Scope)` it is the case that `∃o_r ∈ Reciprocate(p_2, Scope)`, `p_1 ∈ A(o_r)`, and `∃o_j ∈ Reciprocate(p_1, Scope)` such that `p_2 ∈ A(o_j)` then there is a strong reciprocity conflict. An SNS guarantees strong reciprocity, if there are no strong reciprocity conflicts between any two profiles in the network for the information objects defined by `Scope`.

- **weak reciprocity conflicts**: We define a function `Feedback(p)` that returns the tuple `(profileControl, profileAccess)` which are respectively the profiles that co-control the relational information of a profile `p`, i.e., `profileControl = {C(o)|o_o, p ∈ C(o)}` and the profiles that can access information objects of a profile `p`, i.e., `profileAccess = {A(o)|o_o, p ∈ C(o)}`. A weak reciprocity conflict occurs between two profiles `p_1` and `p_2`, if `p_2 ∈ Feedback(p_1).profileControl` or `p_2 ∈ Feedback(p_1).profileAccess`, but `p_2` does not want `p_1` to know that this is the case. An SNS guarantees weak reciprocity if no weak reciprocity conflicts occurs between any two profiles.

It is also possible that for some profile information that reciprocity is not expected, e.g., for information that is made public.

Ultimately, we want to check how the permissions on information objects, some of which are relational information, behave on a social network represented by a graph `G` given the access control model of SNS. We want to verify if and when inconsistencies appear that are not evident from the individual definitions of access permissions. If inconsistencies exist and are acceptable, then we want to know if reciprocity conflicts exists and whether it is possible to implement feedback and awareness mechanisms in SNS despite these conflicts.

In order to analyze the different conflicts, we need to define some helping sets. Each profile can access other profiles and friendship relationships. Access to the latter also means that each profile may know something about other profiles by virtue of being able to access a subset of their relational information. For example, in our model, when profile `p_1` accesses the relationships of `p_2`, it may know the existence of a profile `p_3` and its relationship to `p_2`, although `p_1` may not access `p_3`’s profile. Therefore, for the analysis of conflicts we need to analyze and compare the information objects that each profile can access and knows about and the set of profiles that can access the information objects of and know about a given profile.

Specifically, for each source profile `p_i`, we need the set of profiles and the set of friends relationships that `p_i` can access. We also need the set of profiles that `p_i` knows about, by virtue of accessing their relationships. We call these the **outbound sets** of a profile. The outbound sets are determined by the permissions that profiles in the graph `G` give to their information objects. The **inbound sets** of a profile are those profiles that can access `p_i`’s profile and relationships, which includes those profiles that know `p_i` but cannot access its profile. These sets depend on the permissions set by `p_i` and by the profiles that co-control her relational information in graph `G`. By comparing the outbound and inbound sets, we can determine if conflicts occur between profiles in an SNS.
The relevant outbound sets and the inbound sets for each $p_i$ are as follows:

- **outbound sets:**
  - $\text{AccessibleProfiles}_{p_i}$: profiles that $p_i$ can access
  - $\text{AccessibleRelations}_{p_i}$: relationships $p_i$ can access
  - $\text{KnowsProfiles}_{p_i}$: profiles $p_i$ can access a subset of the relationships of, but whose profiles are not accessible to $p_i$

- **inbound sets:**
  - $\text{ProfilesAccessibleTo}_{p_i}$: those profiles that can access $p_i$’s profile
  - $\text{RelationsAccessibleTo}_{p_i}$: those profiles that can access $p_i$’s relationships
  - $\text{ProfileKnownTo}_{p_i}$: profiles that can access a subset of $p_i$’s relationships but not $p_i$’s profile

### 1.4 Social Networks Construction and Conflict Analysis

In the following, we show our method for detecting conflicts based on RI and TAC using the outbound and inbound sets. In order to do that, we define tokens that model permissions. Then we describe the behavior of the permission tokens, given the permitted activities in the SNS. We summarize four alternative TAC cases, for which we analyze the conflicts using our conflict detection method. Finally, we evaluate the advantages and disadvantages of each case with respect to the privacy concerns raised in Section 1.2 and the functionality of social networks.

In general, the way permissions behave in a graph are dynamic and are based on the activities of profiles. By *activities*, we mean any profile action that updates the graph. These include establishing new relationships, introducing new media objects, commenting on existing information objects, etc. Here, we solely study the establishment of relationships and study its effects on discovering information objects. The information discovery activity also returns the outbound and inbound sets necessary for conflict detection.

#### 1.4.1 Constructing the graph with tokens for permissions

In order to check the permission inconsistencies, as well as reciprocity conflicts, we use tokens, and we rename relationship nodes $r_{ij}$ as *relationship boxes*. Each relationship box $r_{ij}$ is labeled and the permission tokens are inserted to related boxes as relationships are established. These tokens enable the enforcement of the permissions set by the various profiles.

We focus on a basic SNS model. In this basic model, we fix control permissions to always be limited to the creators of information objects, where $o \in P \cup R$. Therefore, control permissions are static and are not further discussed in our model.

**Token distribution**

Each access permission token is a tuple $(p, s)$, where $p \in P$ and $s$ is a path length. In our model, each profile initially has a permission preference for all its information objects that is fixed (since we do not study deletion, it is also not possible to consider changes in permissions). Thus, all her information objects are accessible to a previously selected set based on path length, which can be defined as friends ($s=2$), friends-of-friends ($s=4$), etc. The size of the set changes as the profiles in the SNS establish relationships.
The existence of a token enables the traversal of certain paths. The information objects that can be traversed to starting from a given profile are the information objects accessible to that profile. The $p$ in a token $(p, s)$ can be used as both: permission for some other profile to access a profile $p$, and for $p$ to access the information objects of another profile.

When a relationship is established between two profiles $p_i$ and $p_j$, two edges and a node are inserted into the graph between the two profiles. We stipulate that when a relationship is established then at a minimum the profile information and the relationship box are accessible to the newly befriended profile. Therefore, two tokens $(p_i, s)$ and $(p_j, t)$ are inserted for each of the newly befriended profiles to enable access to the new relationship box and the corresponding profile.

Further, depending on the path lengths defined by tokens in the existing relationship boxes and the path lengths of the new tokens inserted into the newly created relationship box, propagation of tokens takes place. The path length of each token is reduced by two when propagated. The Propagation Algorithm consists of three steps given in the following algorithm:

[Propagate Step 1:] $p_i$ and $p_j$ propagate tokens from existing relationship boxes with $s > 0$ to the newly established relationship box $r_{ij}$.

[Propagate Step 2:] $p_i$ and $p_j$ propagate tokens in the established relationship box $r_{ij}$ with $s > 0$ to all their existing relationship boxes.

[Propagate Step 3:] depending on the maximum path length $s > 0$ of all the tokens propagated in Steps 1 and 2, profiles at a maximum path length $s - 2$ from $p_i$ and $p_j$ propagate the new tokens to further relationship boxes.

The tokens only contain information with respect to propagation path length. Further constraints with respect to distribution can be defined. For example, if “common friends only” is the preference in the network, then in Step 3 tokens are only propagated to the relationship boxes of common friends.

Further, paths to friends beyond friends-of-friends results in tokens traveling in cycles. This has an effect on if and how duplicate tokens are treated in relationship boxes. Since these cycles occur in none of the four cases we study, we erase all duplicate tokens (tokens referring to the same profile), keeping only those with maximum path length.

**Graph Traversal based on tokens**

The traversal rules based on the permission tokens are as follows:

1. traversal to a relationship box: a profile $p_i$ can traverse to a relationship box, if $p_i$ can traverse to a profile neighboring the relationship box, and if the relationship box contains a token $(p_i, s)$ where $s \geq 0$.

2. traversal to a profile: a profile $p_i$ can traverse to another profile $p_k$, $k \neq i$ if:
   - there is a token $(p_k, t)$ where $t > 0$ in every relationship box which $p_i$ has access to on a path between $p_i$ and $p_k$, and
   - in each token $(p_k, t)$ that $p_i$ collects on its path, $t$ is increasing.

**Example**

Figure 1.4.1 shows a graph that resulted from three profiles interacting with each other. $p_1$ and $p_2$ wish to give their friends access permissions to their information objects – these being $p_1, p_2, r_{12}, r_{23}$, while $p_3$ wishes to give her friends and friends-of-friends access permissions, written $FoF(p_3)) = F(p_3) \cup F(F(p_3))$. Their permissions result in the token settings shown in the figure.
• \( p_1 \) and \( p_2 \) can traverse to each other’s profiles and access all relationship boxes
• \( p_2 \) and \( p_3 \) can traverse to each other’s profiles and access all relationship boxes
• \( p_1 \) can traverse to \( p_3 \)’s profile, while the opposite is not the case. This is a result of the traversal to profiles defined above: there are tokens \((p_3, s)\) with increasing path lengths in the relationship boxes on the path to \( p_3 \), and both relationship boxes are accessible to \( p_1 \). In the other direction, there is no token \((p_1, s)\) where \( s > 0 \) in \( r_{23} \). \( p_3 \) can see the relationship box \( r_{12} \) because \( p_2 \) has allowed \( p_3 \) to do so according to the traversal rule to relationship boxes.

**FIGURE 1.3 Example: tokens define traversals to profiles and relationships boxes.**

In order to perform conflict analysis, we use the *Information-Objects Discovery Algorithm*. This algorithm traverses the graph once the permissions are distributed with an exhaustive traversal procedure such as breadth-first. For each profile node encountered during this traversal, a tuple \((\text{profile}, \text{inboundsets}, \text{outboundsets})\) is added to an initially empty set \( \text{Traversable InformationObjects} \). The set \( \text{Traversable InformationObjects} \) is the output of the algorithm. The inbound and outbound sets are then used to identify inconsistencies and reciprocity conflicts among profiles.

### 1.4.2 Relationship building and information discovery in different types of social networks

We now summarize conflict detection for four cases. The four cases involve making profiles and relationships accessible to others, i.e. beyond the profiles in \( C(o), o \in P \cup R \). As argued above, this type of access is necessary for a functioning social network – on the one hand, profiles should be accessible enough to make interesting discoveries, on the other hand, the profiles’ privacy preferences should be observed and undesired access should be minimized. We have selected four cases that are (a) as simple as possible and show the basic features of this problem, that (b) build on one another, that (c) reflect existing default and other possible settings in existing SNS [8].

In particular, in most existing SNS, establishing a relationship between two profiles \( p_i \) and \( p_j \) means that by default, it becomes visible to all friends, i.e. \( F(p_i) \cup F(p_j) \) (Case 1). An alternative, which is currently not available on SNS, is to make the relationships only visible to common friends, denoted \( F(p_i) \cap F(p_j) \) (Case 2). In some existing SNS, profiles can be made accessible to friends and friends-of-friends, shortly stated as \( FoF(p) \). This means relationships are accessible to \( FoF(p_i) \cup FoF(p_j) \) (Case 3). Case 4 investigates what happens when preferences differ, with some users preferring a Case-1 setting and others a Case-3 setting. More complex and differentiated settings can be constructed on this basis, also including information objects shareable by many; they are the subject of future work.

Here, we will describe in detail our model for Case 1, shortly summarize the other three cases, and move on to an evaluation of all four models with respect to inconsistencies, reciprocity conflicts and of the privacy concerns raised earlier – indeterminate visibility.
Case 1: Information objects accessible only to friends \((F(p_i) \cup F(p_j))\)

**Constructing the Model:** This is the most open model possible that is limited to friends of first degree. If a relationship is established, then both relationship partners can see each others’ profiles and list of all friends. \(F(p_i)\) can see \(p_i\)’s relationship with \(p_j\) but cannot access \(p_j\)’s profile, unless they are also in the set \(F(p_j)\). The opposite is also the case. We call this the SimpleFriendsOnlySNS.

**The algorithms** The algorithm consists of a straightforward application of the relationship box insertion and the Propagation Algorithm sketched above. Since tokens are propagated at most once (maximum \(s\) value is 2, and hence any token with \(s = 2\) can only be propagated once) by the newly related profiles, no cycles occur that affect the propagation of tokens. Hence, it is enough if two profiles that establish a relationship complete the first two propagation steps described above.

In order to complete the conflict analysis, we run the Information-Objects Discovery Algorithm. For the model SimpleFriendsOnlySNS, the relevant outbound sets and the inbound sets for each \(p_i\) returned by the algorithm is as follows:

- **AccessibleProfiles\(_{p_i}\):** only \(p_i\)’s friends are accessible to \(p_i\).
- **AccessibleRelations\(_{p_i}\):** various subsets of the relationship boxes of \(F(p_i)\) are accessible to \(p_i\).
- **KnowsProfiles\(_{p_i}\):** through being able to access friends’ relationship boxes, \(p_i\) knows the friends of \(F(p_i)\), that is \(p_x \in F(F(p_i))\) such that \(p_x \notin \text{AccessibleProfiles}_{p_i}\).
- **ProfilesAccessibleTo\(_{p_i}\):** \(p_i\)’s profile is only accessible to her friends \(F(p_i)\).
- **RelationsAccessibleTo\(_{p_i}\):** \(p_i\)’s relationship boxes are accessible to \(F(p_i)\), and since friends of \(F(p_i)\) can access their relationship boxes, a subset of her relationship boxes are also accessible to friends of \(F(p_i)\). In total, different subsets of \(p_i\)’s relationship boxes are accessible to \(\text{FoF}(p_i)\).
- **ProfileKnownTo\(_{p_i}\):** although they are not mentioned in the permissions, by virtue of being able to access subsets of \(p_i\)’s relationship boxes, \(p_x \in F(F(p_i))\) knows \(p_i\), and belongs to the set \(\text{ProfileKnownTo}_{p_i}\), unless \(p_x \notin \text{AccessibleProfiles}_{p_i}\).

**Conflict analysis:**

We will now check for access permission inconsistencies and reciprocity conflicts in the SimpleFriendsOnlySNS model. Control permission inconsistencies are not an issue, since we decided to keep the control permissions fixed.

- access permission inconsistency: For every two profiles \(p_i\) and \(p_j\), if \(F(p_i) \neq F(p_j)\), then we have an access permission inconsistency. If \(F(p_i) \neq F(p_j)\), then some \(p_k \in F(p_j)\) and not in \(F(p_i)\) can access the relationship box \(r_{ij}\), although \(p_i\) had limited the access permission for \(r_{ij}\) to \(F(p_i)\). According to \(p_j\)’s settings, the access to \(r_{ij}\) for \(p_k\) is allowed. As long as there is one profile in the SNS with a different set of friends relationships, access permission inconsistencies occur in Case 1.
- reciprocity conflict: We check to see if strong reciprocity exists for different \(\text{Reciprocate}(p, \text{Scope})\) functions. Specifically, we check if \(\text{Reciprocate}(p, P)\) and \(\text{Reciprocate}(p, R)\) holds for Case 1. \(\text{Reciprocate}(p, P)\) means that if a profile is accessible to \(p_i\), that \(p_i\) profile should also be accessible. This holds in Case 1 since:

  \[
  \text{AccessibleProfiles}_{p_i} = \text{ProfilesAccessibleTo}_{p_i}
  \]

Here, \(\text{Reciprocate}(p, R)\) means that if \(p_i\) can traverse to the relationship box of
a profile, then the reverse should also hold. We defined the \textit{AccessibleRelations}_{p_i} to return the relationship boxes that \(p_i\) can access. The union of \textit{AccessibleProfiles}_{p_i} and \textit{KnowsProfiles}_{p_i} provides us with all profiles whose relationship boxes \(p_i\) can access. If we compare these two outbound sets with the inbound set \textit{RelationsAccessibleTo}_{p_i}, we get:

\[
\begin{align*}
\text{AccessibleProfiles}_{p_i} \cup \text{KnowsProfile}_{p_i} &= F(p_i) \cup \{p_x : p_x \in F(F(p_i)) \land p_x \notin \text{AccessibleProfiles}_{p_i}\} \\
&= F_{oF}(p_i) \\
&= \text{RelationsAccessibleTo}_{p_i}
\end{align*}
\]

From which we conclude that strong reciprocity holds in Case 1 with respect to friends relationships.

Finally, there are also no weak reciprocity conflicts, since all profiles know of the profiles that have access to their information objects because:

\[
\text{KnowsProfiles}_{p_i} = \text{ProfilesKnownTo}_{p_i}
\]

Strong reciprocity also implies weak reciprocity. Thus, there are no reciprocity conflicts in SimpleFriendsOnlySNS.

\textbf{Model evaluation:} We conclude that with the SimpleFriendsOnlySNS, the only conflict is the access permission inconsistency. We have also verified that there are no reciprocity conflicts. The access permission inconsistency can be eased by inserting feedback systems that show the difference between permissions and their behavior (see Section 1.5.1 below).

The SimpleFriendsOnlySNS model is open to socially translucent design, as it produces no reciprocity conflicts. As a design, SimpleFriendsOnlySNS is static and offers the users little flexibility with respect to preferences with alternative path lengths. The access to relational information is co-determined by friends, and can only be extended to friends-of-friends of a given profile. Hence, although the model is not optimal for sharing beyond immediate friends, it is a model which provides a good overview of how and how far information can travel in the network.

\textbf{Case 2: Information objects accessible only to common friends} \( (F(p_i) \cap F(p_j)) \)

\textbf{Constructing the model:} The SimpleCommonFriendsOnlySNS model is constrained in that only common friends are able to see that a relationship exists. This also means that access to a profile only reveals the list of common friends. If there is no central instance of the SNS, the SimpleCommonFriendsOnlySNS causes the problem of verifying \(p_i\) and \(p_j\)’s common friends without revealing information about the friends they do not have in common. This could be solved using zero-knowledge proofs or through a trusted third party that would do comparisons of encrypted lists of friends. This is not the focus of this chapter, so we simply assume that there is a function that returns the list of common friends.

\textbf{Model evaluation:} The SimpleCommonFriendsOnlySNS model avoids inconsistencies and reciprocity conflicts. At the same time, it is a very conservative model in which the network visibility of profiles is very small. As a design, SimpleCommonFriendsOnlySNS is static with respect to preference alternatives. Further, the model makes it difficult to discover and establish new communities where common friends are sparse.

\textbf{Case 3: Relationship accessible to} \( \text{FoF}(p_i) \cup \text{FoF}(p_j) \)

\textbf{Constructing the Model:}
The SimpleFoFOnlySNS allows relationships to be visible to friends of friends. If a relationship $r_{ij}$ is established, both $\text{FoF}(p_i) \cup \text{FoF}(p_j)$ can see that this relationship exists. The model is similar to the SimpleFriendsOnlySNS.

The relationship establishment algorithm is the same as that of the model SimpleFriendsOnlySNS up until the last propagation step (see Propagate Algorithm Step 3) in which friends of $p_i$ also update their relationship boxes, so that the access permissions also hold for friends-of-friends.

**Model evaluation:** Case 3 is a generalization of Case 1. There is one important difference: in SimpleFoFOnlySNS the permission tokens travel further. As a result, each profile $p_i$ can access relationship boxes of $F(F(p_i))$. In our model, by accessing relationship boxes, the profile $p_i$ can see the permission tokens in that relationship box. Relationship boxes in SimpleFoFOnlySNS contain tokens from friends-of-friends. Hence, any $p_i$ by accessing the relationship boxes of $F(F(p_i))$ has access to permission tokens of profiles at path length 8 away from $p_i$. This means that the $\text{KnowsProfile}_{p_i}$ contains profiles whose relationship boxes $p_i$ may access, or profiles whose permission tokens $p_i$ may access. Hence, we show that the SimpleFoFOnlySNS contains no strong reciprocity conflicts by reasoning from the outbound set $\text{AccessibleRelations}_{p_i}$ which returns $\text{FoF}(p_i)$. Hence, $p_i$ can also access relationship boxes of $F(\text{FoF}(p_i))$, which is equal to $\text{RelationsAccessibleTo}_{p_i}$ as shown in Table 1.1. We also see in the table that Access permission inconsistency $\text{KnowsProfiles}_{p_i} = \text{ProfilesKnownTo}_{p_i}$, hence there are also no weak reciprocity conflicts. Further, access permission inconsistencies holds in the SimpleFoFOnlySNS: each relationship is accessible by $\text{FoF}(p_i) \cup \text{FoF}(p_j)$. Hence, Case 3 is problematic from a usability and transparency perspective: it is probably more difficult for users to imagine and grasp access permission inconsistencies across longer path lengths. The model is more interesting from a sharing perspective then the previous two cases as information travels further. Finally, in common with the previous three models, no alternatives are available to the profiles with respect to determining the access permission path length for their information objects.

**Case 4: Relationship accessible to friends or to friends-of-friends**

**Constructing the Model:** Next, we construct a model that allows users some flexibility in their choice of permission path length. Users can choose between making their information objects available only to friends or to friends-of-friends. The model becomes complicated as soon as a profile $p_i$ wants to make her information objects accessible only to her friends, but a $p_j \in F(p_i)$ wants to make her information objects accessible to friends-of-friends.

**FIGURE 1.4** Graph with profiles setting access to either $F(p)$ or $\text{FoF}(p)$.

Consider the example graph in Figure 1.4.2. There are two options:

1. $\text{FoF}(p_i)$ overrides $F(p_j)$: $p_3$ sets access permissions for her information objects to
This means that $p_1$ can access $p_3$’s profile and her relationship boxes, although the opposite is not the case. Further, $p_4$ has set her access permission for her information objects to $F(p_4)$. However, according to $p_3$’s permissions, $p_1$ can access relationship box $r_{34}$. Hence, in this model we prioritize $p_3$’s permissions and suggest that they override $p_4$’s access permission that limits access to its relationship boxes to $F(p_4)$. We call it the FoFDominantSNS model.

2. $F(p_j)$ overrides FoF($p_i$): Alternatively, $p_4$’s access permissions can be prioritized, meaning that relationship box $r_{34}$ is not accessible to $p_1$. This we call the FDominantSNS model.

Model evaluation: The model FoFDominantSNS offers greater flexibility to users, since they can now choose between making their information objects accessible to friends or to friends-of-friends. Although this produces access permission inconsistencies, with appropriate feedback systems these can be seen as a feature of the model (see Section 1.5.1 below). The dominance of one permission over another does cause problems. Namely, in the worst case, when a profile $p_i$ with access permissions set to friends is surrounded by neighbors $p_j$ with access permissions set to friends-of-friends, then her permissions have no effect on the accessibility of her relationship boxes. Therefore, if users like the profile $p_i$ are not aware of the effects of the FoFDominantSNS model, they may harbor false perceptions of control. If they are aware of the effect, it may cause frustration or simply confusion.

An alternative model like the FDominantSNS may solve these problems for profiles $p_i$, but then cause problems for profiles $p_j$ who have the feeling that their ability to share their resources with a greater community can be dampened by their very own friends. Therefore, although the two models in Case 4 provide the users with more flexibility, they also come along with their problems, especially with respect to reciprocity. As a result, a socially translucent design may be more difficult to implement with such models.

1.4.3 Comparing Models

In Table 1.1, we give an overview of the conflict analyses in the four cases (each case is listed in Column 1). In the second column, we have abbreviated $S(p_i,p_i).accessPermission$ to $S(p_i)$ since control permissions are fixed and are hence not used in the table. In columns two and three, we indicate whether permission inconsistencies or strong reciprocity conflicts occur. In the remaining columns, we list the outbound and inbound sets necessary to identify whether conflicts occur. In column $KnowsProfile_{p_i}$, after the first row we abbreviated $AccessibleProfiles_{p_i}$ to $AP_{p_i}$ to save space. In column $RelationsAccessibleTo_{p_i}$, we listed the profiles that have access, but did not specify which relationship boxes each profile can access. These are usually subsets of the profile $p_i$’s relationship boxes: relationships of $p_i$ known to another profile $p_x$ by virtue of accessing the profiles of another profile $p_j \in F(p_i)$. We also did not add a column for weak reciprocity conflicts. If no strong reciprocity conflicts occur, then weak reciprocity conflicts do not occur either. On the other hand, in Case 4, not only is strong reciprocity not guaranteed, but depending on the permutations of the neighbors preferences, different types of weak reciprocity conflicts occur. For the sake of readability and giving an overview, we chose not to list the different permutations of preferences that lead to weak reciprocity conflicts.

The most important conclusion to be drawn from the table is that the many privacy preferences offered by SNS providers produce inconsistent access permissions which are as a rule not communicated to the users of SNS. Further, it becomes evident from the table that the only case in which conflicts are avoided and information visibility is controlled is the model SimpleCommonFriendsOnlySNS (Case 2). This model is also the most conservative:
for cliques in a graph, access to information objects remains very open; for profiles that are not grouped in such close communities, access to information objects is very limited. Next, the problems that occur in Case 1, the model SimpleFriendsOnlySNS are simply amplified in Case 3, the model SimpleFoFOnlySNS.

We numbered the two sub-models of Case 4. Case 4.1 represents the mixed model with the FoFDominantSNS, whereas Case 4.2 is the FDominantSNS that privileges profiles \( p_i \) with access permissions set to friends. In Case 4.2, regardless of the access permissions of \( p_j \in F(p_i) \), \( p_i \) is guaranteed protection of its information objects equivalent to those of SimpleFriendsOnlySNS. If privacy can be seen as the ability to keep information close to a profile, then Case 4.2 can be favorable. However, Case 4.1 is the common model in existing SNS. This could be because it privileges sharing over privacy. Given the evaluation of the model of Case 4.1, it is no surprise that many users complain about indeterminate visibility of their information objects.

We conclude that already in a comparatively simple model in which profiles can choose between two permission settings, challenging complexities occur. In current SNS, there are also relational information objects for which the \( C(o) \) can be extended after the creation of the object. It is nevertheless clear in the four cases that combinations of permissions from networked profiles can produce unexpected visibility of information objects depending on the relationships and access permissions. Depending on the model, a profile’s permissions can be completely overridden by the permission of their friends at different path lengths. Access permission inconsistencies may be easier to determine, but determining reciprocity conflicts exhaustively can be a challenge.

Through relational information and transitive access control settings, it becomes evident that privacy in social networks is not only about individual decisions but also about collective ones. As simple as they may seem, the models for access control used in existing SNS are full of inconsistencies that are often brushed away. These become evident as users state indeterminate visibility and breaches of separation of identities as privacy concerns. Making the underlying relationality of access control evident and opening its design to users, or developing it to fit the users’ requirements, is a step forward in improving systems according to users’ differing privacy needs. The Information-Objects Discovery Algorithm hence can be used to detect conflicts, discuss requirements and design solutions with users and other stakeholders. In the next Section, we make proposals on how the conflicts made explicit through our models and analyses can be resolved using requirements engineering and data mining methods.

### 1.5 Data Mining and Feedback for Awareness Tools

Many privacy proponents observe that abstract warnings of the kind “If you disclose data, others may see and abuse them” are too abstract to have much effect on people’s behavior. Various authors have therefore proposed more concrete demonstrations of what is already known about the user, or about other people, in an attempt to be more effective by raising awareness.

Many of these techniques are based on retrieval. Tools allow a user to search for her name on various search engines and Social Web sites, and compile the results in one page (see www.123people.com as one example). A variant of this technique, which relies on pull activities and is thus more likely to reach users who are privacy-conscious anyway, is the “Identity Angel” [48] that employs a push technique. This is a specialized search engine that visits online job boards and other sources to look for either of three types of personally identifying information: a person’s name, address, and Social Security Number.
<table>
<thead>
<tr>
<th>Model</th>
<th>$S(p_i)$</th>
<th>Access Perm Conflict</th>
<th>Strong Recip. Conflict</th>
<th>Accessible Profiles $p_i$</th>
<th>Accessible Relations $p_i$</th>
<th>Knowns Profile $p_i$</th>
<th>Profile Accessible To $p_i$</th>
<th>Relations Accessible To $p_i$</th>
<th>Profile Known To $p_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.1</td>
<td>$F(p)$</td>
<td>yes</td>
<td>no</td>
<td>$F(p)$</td>
<td>${ \forall x: F(p_i) \in C(\forall x) }$</td>
<td>${ px: px \in F(F(p_i)) \land px \notin AccessibilityProfiles }$</td>
<td>$F(p_i)$</td>
<td>${ px: px \in F(F(p_i)) \land px \notin ProfileAccessibleTo }$</td>
<td></td>
</tr>
<tr>
<td>C.2</td>
<td>$F(p)$</td>
<td>no</td>
<td>no</td>
<td>$F(p)$</td>
<td>${ \forall x: px \in F(F(p_i)) } \cup { \forall x: px \notin (F(F(p_i)) \land F(F(p_i))) }$</td>
<td>$\emptyset$</td>
<td>$F(p)$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>C.3</td>
<td>$FoF(p)$</td>
<td>yes</td>
<td>no</td>
<td>$FoF(p)$</td>
<td>${ \forall x: FoF(p_i) \in C(\forall x) }$</td>
<td>${ px: px \in FoF(F(F(p_i))) \land px \notin AccessibilityProfiles }$</td>
<td>$FoF(p)$</td>
<td>${ px: px \in FoF(F(F(p_i))) \land px \notin ProfileAccessibleTo }$</td>
<td></td>
</tr>
<tr>
<td>C.4.1</td>
<td>$F(p)$</td>
<td>yes</td>
<td>yes</td>
<td>$F(p) \cup { px: px \in F(F(p_i)) \land S(px) = FoF(p_i) }$</td>
<td>${ \forall x: F(p_i) \in C(\forall x) } \cup { \forall x: px \in F(F(p_i)) \land S(px) = FoF(p_i) }$</td>
<td>$(F(p) \cup { FoF(p) \cup px \in F(p_i) \land S(px) = FoF(p_i) }) \cup { \forall x: S(px) = FoF(p_i) } \cup { F(p) \cup FoF(p) \cup S(px) = FoF(p_i) } \cup { \forall x: S(px) = FoF(p_i) }$</td>
<td>$F(p)$</td>
<td>${ F(p) \cup FoF(p) \cup px \in F(p_i) \land S(px) = FoF(p_i) } \cup { \forall x: S(px) = FoF(p_i) } \cup { S(px) = FoF(p_i) } \cup { S(px) = FoF(p_i) }$</td>
<td></td>
</tr>
<tr>
<td>C.4.2</td>
<td>$F(p)$</td>
<td>yes</td>
<td>yes</td>
<td>$F(p) \cup { px: px \in F(F(p_i)) \land S(px) = FoF(p_i) }$</td>
<td>${ \forall x: F(p_i) \in C(\forall x) } \cup { \forall x: px \in F(F(p_i)) \land S(px) = FoF(p_i) }$</td>
<td>$FoF(p) \cup { py: py \in FoF(p_i) \land S(py) = FoF(p_i) } \cup { py: py \notin S(py) } \cup { py: py \notin S(py) } \cup { py: py \notin S(py) } \cup { py: py \notin S(py) } \cup { py: py \notin S(py) }$</td>
<td>$F(p)$</td>
<td>${ px: px \in F(F(p_i)) \land px \notin ProfileAccessibleTo }$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$FoF(p)$</td>
<td>yes</td>
<td>yes</td>
<td>$F(p) \cup { px: px \in F(F(p_i)) \land S(px) = FoF(p_i) }$</td>
<td>${ \forall x: F(p_i) \in C(\forall x) } \cup { \forall x: px \in F(F(p_i)) \land S(px) = FoF(p_i) }$</td>
<td>$FoF(p) \cup { py: py \in FoF(p_i) \land S(py) = FoF(p_i) } \cup { py: py \notin S(py) } \cup { py: py \notin S(py) } \cup { py: py \notin S(py) } \cup { py: py \notin S(py) } \cup { py: py \notin S(py) }$</td>
<td>$F(p)$</td>
<td>${ px: px \in F(F(p_i)) \land px \notin ProfileAccessibleTo }$</td>
<td></td>
</tr>
</tbody>
</table>
Although the first two are often easy to find on the Web, finding all three is the gold standard for anyone who wants to commit fraud or steal someone’s identity. If the Identity Angel program finds all three, and can locate the person’s e-mail address, it will send an automated message to that person, warning that his/her identity may potentially be in danger.\footnote{Push techniques may work even without reference to specific information: \cite{32} found that sending teenagers on MySpace an email from “Dr. Meg”, mentioning that “you seem to be quite open about sexual issues or other behaviours such as drinking or smoking”, caused many of them to clean up their profiles and boost privacy settings.} Push techniques may work even without reference to specific information: \cite{32} found that sending teenagers on MySpace an email from “Dr. Meg”, mentioning that “you seem to be quite open about sexual issues or other behaviours such as drinking or smoking”, caused many of them to clean up their profiles and boost privacy settings.

Other techniques emphasize inference: that the presence of such different explicit pieces of information on a person means that there is even more – implicit – information about them. With the help of a mashup that visualizes the homes (including addresses) of people interested in “subversive” books, \cite{36} demonstrated the simplicity of combining existing information to infer people’s identity. The technique is attribute matching, which was shown by Sweeney \cite{47} to be an easy way of circumventing simple anonymization schemes. In social networks, recent studies have shown that attributes revealed by friends in a profile’s vicinity may be used to infer confidential attributes \cite{53, 4}. These methods are powerful and at the same time controversial. For example, although inferencing attributes through social networks may return mathematically accurate results, the assumptions such inferences are based on are socially questionable. In \cite{52}, for example, the authors problematize some of the heuristics used to probabilistically infer information from the attributes of a profile’s network vicinity. They argue that the heuristics are comparable to notions like ”birds of a feather flock together”, ”judge a man by the company he keeps”, or ”guilty by association”.

Hence, such inference techniques should be made visible to users to raise awareness of both, the (in)appropriateness of their underlying heuristics, as well as the potential risks accruing if they are applied. Along these lines, in \cite{5}, we have presented a P3P extension and a Web-based service for helping businesses to avoid computing analytics that would indirectly violate their own P3P policies and therefore data-protection agreements with their customers. This service was based on computing inferences implicit in data configurations. We have proposed an analogous extension to P3P for different groups of “friends” in social network services in \cite{40}.

Data mining can extend the scope of these applications and simulations for information inference by employing more sophisticated forms of induction and deduction for demonstrating the possible consequences of a user’s actions. In Section 1.4, we have shown how the spread of visibility/accessibility of a user’s profile and relational data may be computed. The same could be done for the spread of control permissions over such data. In the remainder of this section, we first outline how inferencing could be employed in the SNS models we studied and then sketch more sophisticated approaches.

1.5.1 Towards conflict avoidance and resolution: Feedback and trust mechanisms

The analyses of the cases in Section 1.4 have shown, among other things, access permission inconsistencies: some information becomes visible beyond the group of people originally intended. A “feedback mechanism” could be implemented to make users aware of this.

\footnote{Related business models are becoming common practice. For example, an entrepreneur in the UK has launched Lucid Intelligence, a database of personal data available on phishing and hacking sites. Concerned individuals can pay to check whether their data security has been breached \cite{1}.}
It would signal, upon the intention of $p_i$ and $p_j$ to establish a relation, to them that the actual group of people who will be able to see the relationship will be larger than what they (probably) expect based on their individual permissions. A simple feedback mechanism could rest on showing $\text{AccessReachFeedback}(r_{ij}) = \bigcup_{p \in C(r_{ij})} S(p, r_{ij})$.

If such access permission inconsistencies are judged to be acceptable, there is no problem. However, if users disagree, other models will have to be considered that either avoid access permission conflicts or allow users to articulate conflicting requirements and find designs that allow users to negotiate these prior to design or during run time. It is important to underline the fact that users cannot and do not decide on their preferences alone as long as RI and TAC is implemented.

In Cases 1–3, these problems occur in a symmetric fashion. However, in Case 4, model 4.1, the access to information objects can become asymmetric. Assume again that $p_i$ is the more restrictive side of a relationship to be established. If $p_i$ is surrounded by profiles such that for all $p_x \in F(p_i)$ the permissions for profiles are $S(p_x, p_x).accessPermission = FoF(p_x)$, then $p_i$'s choice of limiting her access permissions to its information objects to $F(p_i)$ is close to meaningless. It only serves to protect her profile, which may be important, but nevertheless, may not be sufficient. The reverse, that $p_j$ is surrounded by profiles $p_y$ that all set their access permissions to friends might not be problematic, unless $p_j$ feels that her intentions to share her resources have been limited by her friends.

In order to avoid such cases, one could introduce trust models. In these trust models, $p_j$ may be allowed to make $r_{ij}$ accessible only to those friends-of-friends who can prove a threshold trust level that the two profiles $p_i$ and $p_j$ determine together (see Chapter 15 of this book for further uses of trust in social-network access modelling). In any case, $p_i$ should be provided with feedback similar to the feedback proposed in Case 1, informing her about how far her relationship information travels through the graph. This feedback can be coupled with collective privacy setting negotiation mechanisms, building on policy visualization techniques like the Expandable Grids [43].

1.5.2 Design choices in feedback mechanisms based on data mining

Going beyond straightforward what-if simulations, we believe that feedback for awareness-raising simulations should not only be limited to the application of data-mining models such as classifiers or graph inference results. Rather, it is vital to consider also the dynamics with which users’ data-related activities contribute to the learning of these models. Thus, we propose to integrate data mining more fully – by considering also statistical information, by considering also the learning stages of a model – into creating privacy awareness tools.

As one example in social networks, consider the problem of inserting structure into the set of “friends”. In current SNS, these sets have no internal structure, or friends can be assigned to predefined classes [8]. These sets grow too fast for many users and easily become unmanageable. This is reminiscent of the email structuring problem (which has been addressed by several machine-learning approaches such as [10]). In addition, it is however an increasing privacy problem, because profile and relational information is distributed either to all friends or to none. To improve on this situation, the user’s set of “friends” could be clustered by connectivity, a classifier could be learned from the user’s own past communication behavior with these different clusters, and a recommender could be derived from it to suggest that in the future, it might be advisable to withhold certain information from this group. Exactly such a mechanism was implemented in [20] based on tie strength characterized by multiple dimensions representing trust and closeness among friends. This type of clustering / classification / recommendation mining could be incremental, such that the effects of decisions such as accepting an invitation to become friends attain visibility.
This basic idea gives rise to a number of choices and questions: (a) The implications could be shown to users in pull or push fashion. Push has the advantage of potentially reaching more people, but the disadvantage of potentially becoming tiresome and ignored if too many warnings are issued. Machine learning could in turn be used to learn how and when to make proposals to a user to maximize effectiveness (cf. earlier work on desktop agents). (b) Inferences can be based on already-stored data or on what-if simulations. The latter have the advantage of warning people “before it’s too late”, but may therefore also create a false sense of security. This tradeoff remains an open issue for interaction design. (c) The target groups to whom inferences are shown can range from end users (natural persons in SNS applications, businesses in applications like [5]) to SNS providers.

Simulations issued to end users are particularly interesting when conflicts between end users are possible: Simulation runs in simultaneous interaction with several users may show a conflict about to evolve. This could pave the way for a negotiation between the users as a way of solving the conflict. Such solutions will often require a preference aggregation function. For example, a minimum strategy may be applicable in situations like that of Fig. 1.4.2, where it would imply that the conflict between \( p_1 \) and \( p_3 \) will be resolved in favor of the more restrictive preference of \( p_1 \), such that neither of them can see the other’s profile. This also involves the challenge of (d) how to best address groups (rather than individuals). Some of the issues involved (such as preference aggregation) will be similar to those in issuing recommendations to groups [26], but further ones will surface due to the as-yet little-explored nature of privacy seen as a collective good.

On the other hand, simulations can also be helpful for service providers to help them make well-informed choices when introducing new features into their applications. Choices made in response to public outrage at previous choices (such as Facebook’s alterations of the Beacon functionality from opt-out to opt-in after the widespread anger expressed in the blogosphere, cf. http://news.bbc.co.uk/2/hi/technology/7120916.stm) may show a certain responsiveness to user interests, but participatory design would preview and integrate concerns in a much earlier phase of software development and maintenance.

1.6 Conclusions and outlook

In this chapter, we have analyzed challenges for privacy on today’s Social Web. We have focused on social network services as Social Web applications that are both highly popular at the moment and regarded by many as prime players in the “privacy nightmare” of the Web. We have argued that this seeming contradiction shows that a re-investigation of the notion of privacy itself is necessary. This investigation has shown that privacy is not only about hiding or controlling information, but also about the practices with which collectives – such as the users of a social network service – constantly explore and (re-)negotiate what information to disclose and what to hide, and the construction of identities therein. We described empirically identified types of concerns and breaches in detail and derived relational information and transitive access control as central to both sharing and privacy concerns. The latter is the case because the individual profile privacy settings in SNS, when aggregated in a network with the privacy settings of other profiles, often lead to unexpected visibility of information objects and to privacy concerns. This shows that access to information objects is not determined solely through individual privacy settings but depends on the permissions of friends of different path lengths – depending on the SNS design. Hence, tools are necessary that make transparent collective data practices and access to information objects, as are tools for negotiating differences. All of these developments affect how personal data is constructed legally and question the viability of using existing
data protection frameworks in social networks.

We therefore proposed formal definitions of conflicts resulting from relational information and transitive access control, and a method for detecting the occurrence of these conflicts in various types of social networks. As an example, we provided an analysis of different design settings typical of SNS and studied the conflicts that may emerge in them. We also briefly described proposals for avoiding or resolving those conflicts. Concluding, we proposed to use this analysis to infer which conflicts can emerge, along with other inference methods from data mining, to create both techniques for requirements engineering and privacy awareness tools.

Many open issues remain for future work. Models should be considered in which control permissions may also be granted and deletion of both permissions and information objects is possible. We expect that the inbound and outbound sets will become impermeable as these parameters are included in the model, emphasizing the importance of data mining and visualization of collective permissions for privacy as practice. We plan to develop and implement simulation models and awareness tools which we will use to gather and analyze requirements, support negotiations between users and other stakeholders – that include service providers and privacy commissioners– prior to wide-spread privacy breaches and during run-time. In general, modeling relational information and transitive access control explicitly will enable different analyses on social network groups, affect the possible inferences made from networks, and change understandings of information, sharing and social ties in SNS.

In this chapter, our main objective was to treat data mining not only as a technology that can cause privacy breaches or that can help build new privacy-preserving algorithms, but also as a technology for privacy and information literacy. Data mining techniques can be used to build “privacy mirrors” that inform individuals and groups about the possible consequences of various privacy-related behaviors at the requirements phase, design phase and run time. Further collaboration between research on data mining, access control, interaction design, and other fields mentioned in this chapter may make data practices transparent and hence actively support the establishment of privacy-conscious behaviors on the Social Web.

References


35. OECD. Guidelines on the protection of privacy and transborder flows of personal data.


