A Trust-Augmented Voting Scheme for Collaborative Privacy Management

Yanjie Sun[a,*], Chenyi Zhang[b], Jun Pang[a,**], Baptiste Alcalde[a], Sjouke Mauw[a,c]

[a] *Faculty of Sciences, Technology and Communication, University of Luxembourg, Luxembourg*
[b] *School of Information Technology and Electrical Engineering, University of Queensland, Australia*
[c] *Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, Luxembourg*

Abstract

Social networking sites have sprung up and become a hot issue of current society. In spite of the fact that these sites provide users with a variety of attractive features, much to users' dismay, however, they are prone to expose users' private information. In this paper, we propose an approach which addresses the problem of collaboratively deciding privacy policies for, but not limited to, shared photos. Our approach utilizes trust relations in social networks and combines them with Condorcet's preferential voting scheme. We study properties of our trust-augmented voting scheme and develop two approximations to improve its efficiency. Our algorithms are compared and justified by experimental results, which support the usability of our trust-augmented voting scheme.

Key words: Privacy, voting, trust, algorithms, social networks

1. Introduction

Social networking is one of the most influential inventions on the Internet during the past ten years. Social network sites provide users platforms to socialize both in the digital world and in the real world, for making friends, information exchange and retrieval, and entertainment. Some of the largest ones, such as Facebook [13], Google+ [17], Twitter [33], Kaixin001 [23] and RenRen [28] (the latter two mainly for Chinese users), provide services to hundreds of millions users world wide. However, partly due to the intention to attract as many users as possible for their commercial success, social networking sites

*Corresponding author
**Principal corresponding author
Email addresses: chenyi@uq.edu.au (Chenyi Zhang), jun.pang@uni.lu (Jun Pang), baptiste.alcalde@uni.lu (Baptiste Alcalde), sjouke.mauw@uni.lu (Sjouke Mauw)

tend to intentionally or unintentionally expose private information of existing users. User privacy has become an important and crucial research topic in social networks. A number of scholars have studied it from different viewpoints, e.g. [19, 11, 29, 5, 20, 30]. Moreover, the excessively expanded number of users also bring difficulties into the management of these sites, so that designing effective mechanisms to coordinate users' opinions over their privacy becomes an emerging issue.

As a shared platform, resources in a social network may be co-owned by a number of users. For instance, documents can be co-authored and several users may appear in the same photo. Determining the co-owners requires some content recognition or manual tagging mechanisms. The latter are supported by several popular social networking sites (e.g., FACEBOOK, KAIXIN001 and RENREN). Such co-ownership might cause a breach of privacy. For example, suppose user Alice wishes to publish on her personal page a picture which contains Bob's image, this action may cause exposure of Bob's privacy, regardless of Bob's personal will. In response to this issue, most of the social networking sites choose to place the burden of privacy setting solely on the owners of resources, to which we hold a different stance. Firstly, it is rather ineffective to make the owner solely responsible for the privacy setting of a picture. Secondly, co-owners of a picture may have different, or sometimes even conflicting, privacy concerns or preferences. Therefore, we believe it is more reasonable to let all co-owners participate in the privacy setting. In this paper, we mainly focus on the particular problem of how to merge privacy opinions from co-owners of shared resources.

Voting schemes are natural candidates for aggregating individual preferences into a joint decision that reflects the "general will" (or a social choice) of a group of people who share the piece of data. Siquicciarini et al. [31] propose a game theoretical method based on the Clarke-Tax mechanism [6], which can maximize the social utility function by encouraging truthfulness among people in the group. This induces a nice property that the final decision cannot be manipulated by individuals, as users express their true opinions about the privacy preference. However, their method is not as simple as it is claimed to be, as it requires each user to compute a value for each different preference and the user-input values are essential for their method to derive a joint decision. We argue that this requirement is not realistic and it makes users less interested in participating collaborative privacy control.

Instead, in this paper we propose a different but novel solution, by combining trust in social networks with the well-known Condorcet's preferential voting scheme. The trust relations are inherent in social networks and can be easily derived among users, for example, by comparing user profiles or computing the distance of users in a social networking site. We believe that trust should play an important role especially when users cooperate to decide the privacy policy on a shared resource. This also indicates that our solution to collaborative privacy control in social networks is applicable and effective when a trust relation on people in a friendship circle is established. Exploring trust helps to identify malicious users (co-owners with low trust values) when merging privacy opinions

from a group of people.[1] To exclude malicious users, one possible way is to set a threshold to filter out co-owners with low trust values. In a preferential voting scheme a user is required to provide a ranking — a linear order on the available privacy preferences, rather than only select a single choice. This allows users to express their opinions on different privacy policies in a more comprehensive way. The method of calculating a final ranking from a collection of rankings follows the consensus rule that was proposed at a conceptual level by Condorcet in the late eighteenth century, and rediscovered by Kemeney [24] and Young et al. [35, 34]. According to the rule (known as Kemeney's rule), the proposed method computes a final ranking that has the minimal sum of distances to all users' choices [24]. Our application enriches the method by associating a weight to each individual user's ranking, reflecting the degree of *trust* of that user as considered by the owner of the resource.

Comparing to the method of Siquicciarini et al. [31], ours is simple to use, in that our method only requires users to tick a ranking, and there are no values associated to each preference. The above discussion reflects two design rationales, *expressiveness* and *simplicity of use*, in our mind, and these considerations lead us to a method for collaborative privacy control which is as simple as possible without losing its expressive power.

Structure of the paper. The rest of the paper is organized as follows. Sect. 2 introduces notions of trust in social networks and Condorcet's preferential voting scheme. In Sect. 3 we propose a new algorithm that enhances Condorcet's voting scheme by taking trust relations in social networks into account, and discuss the properties of this trust-augmented voting scheme. In Sect. 4 we develop two approximations – one based on Borda's counting method and the other heuristic-based – to improve the efficiency of the proposed trust-augmented voting scheme, when the number of privacy policies is getting large. Sect. 5 presents experimental results on comparing the algorithms previously introduced, which also justify the correctness and efficiency of our approximations. We conclude the paper in Sect. 6.

This article is a revised and extended version of [32] that appears in the proceedings of the 6th International Workshop on Security and Trust Management (STM'10). In this version we have included a new discussion on the properties that are satisfied by our trust-augmented voting scheme and a new algorithm to approximate the scheme based on Borda count. We have also conducted new experiments to test all the proposed algorithms.

2. Preliminaries

2.1. Trust in social networks

Literature shows that social life is simply not possible without trust [16, 26]. In particular, trust relations are central to cooperation, i.e., the social process

[1] Trust is also shown useful to deal with malicious routers in anonymous communication [21].

aiming at the increase or preservation of the partners' power, wealth, etc. Online social networks reflect human social relations in the Internet, allowing users to connect to people they know, to share data (e.g., video, photos, text), and to have group activities (e.g., games, events). A number of social structures have been introduced in social networking sites, such as friendship, group membership and virtual family. The notion of trust is naturally present in social networks, and moreover, in contrast to real life, it can be quantified and made explicit.

Trust has been defined in several different ways. The definition of trust adopted here, first formulated by Gambetta [15], is often referred to as "reliability trust". Thus, we define *trust* as the belief or subjective probability of the *trustor* that the *trustee* will adequately perform a certain action on which the trustor's welfare depends. Trust is hence a quantifiable relation between two agents. There are mainly two approaches to trust quantification, namely by the evaluation of the similarity, and by an analysis of relevant past events (stored in a history) between two entities.

The similarity between two entities is a distance function that can take various attributes into account, such as *social* (e.g., gender, location, company, etc.), and *behavioral* (e.g., the way one ranks or buys) attributes. Intuitively, the closer the two entities are w.r.t. an attribute (i.e., the distance is small), the more likely the trust relation will be strong. In practice, a social networking site like FACEBOOK implements a *social similarity* mechanism in order to suggest to the user people that she might consider as friends. For instance, if Alice and Bob both have Clare and Danny in their friend list but Bob also has Elisabeth as a friend, then Elisabeth might be suggested to Alice.

In the second approach, a sequence of past events concerning a specific action[2] can be analyzed to predict the likelihood that the same action will be correctly performed if requested. This analysis can consist in checking a property over the history [25, 10], or the computation of a probability (e.g., Hidden Markov Models [12]). In the context of a social network, this requires to check the history of events or games that a given user has been participating in or playing during a certain time window. Based on this mechanism, a social networking site can suggest friends who have similar interests to the user. Moreover, trust transitivity can be used when an entity wants to evaluate the trust in an unacquainted entity, e.g., when the trustor does not have access to the trustee's profile, or has never interacted with her before. Trust transitivity is defined as the possibility for the trustor to use trust information from other entities in order to infer a trust evaluation towards the trustee, i.e., derive a trust value from a trust graph. In social networks, we can use the trust over friendship relations as transitive relations. Computational models for trust transitivity can be found in the literature [18, 22, 8, 2], and can be applied to social networks.

Our main motivation to incorporating trust in collective privacy management is that people's opinion in a social networking site can be evaluated by taking trust relations into account. For instance, when Alice rates a video that is

[2]In our setting, the action can be the collaborative decision making on privacy settings.

made by Bob, Clare evaluates this data item through the trust she has assigned to Alice (as a referee) and Bob (as a film-maker). Similarly, combining users' opinions is usually required to decide privacy policies of shared contents, which leads us to a trust-augmented voting scheme. In the following sections, we define trust as a function that assigns a value in $[0, 1]$ to every (ordered) pair of users, and assume that trust values among users can be efficiently computed in social networks by using the approaches as discussed above. The meaning of a value 0 is that a trustor fully distrusts a trustee such that his opinion will be completely disregarded, while 1 means that a trustor fully trusts a trustee.

A useful approach to derive such a trust value for a given trust network is Jøsang's Subjective Logic (SL) [22]. In this logic, a trust value is represented by a triplet (b, d, u), where $b \in [0, 1]$ denotes *belief*, $d \in [0, 1]$ denotes *disbelief*, and $u \in [0, 1]$ denotes *uncertainty*. It is required that the sum of these values equals 1, $b + d + u = 1$. The value $(1, 0, 0)$, for instance, represents full belief, while $(0.2, 0.2, 0.6)$ denotes an equal amount of belief and disbelief, with a relatively high uncertainty of 0.6.

Following this approach, a number r of positive experiences and a number s of negative experiences, which describe the experiences of agent A with respect to agent B, can be transformed into the SL triplet $(\frac{r}{r+s+2}, \frac{s}{r+s+2}, \frac{2}{r+s+2})$. This triplet describes the direct trust of agent A in agent B. The value 2 originates from the required correspondence between the Subjective Logic and beta distributions, where two *pseudo-experiments* are defined (one *success* and one *failure* experiment). Composition of such trust values then allows for the calculation of the indirect trust relations in a trust network. The two most relevant composition operators are *consensus* (notation $\oplus$) and *conjunction* (notation $\otimes$). Consensus is used for the fusion of two opinions concerning the same trustor and trustee. Conjunction is used to express the transitivity of trust, i.e. to calculate the trust of agent A in agent C, while knowing the trust of A in B and the trust of B in C. These two operators are defined as follows:

$$\begin{aligned}(b, d, u) \oplus (b', d', u') &= \left(\frac{bu' + b'u}{u + u' - uu'}, \frac{du' + d'u}{u + u' - uu'}, \frac{uu'}{u + u' - uu'}\right), \\ (b, d, u) \otimes (b', d', u') &= (bb', bd', d + u + bu').\end{aligned}$$

Finally, once the indirect trust relations in a trust network have been calculated, the relevant trust triplet (b, d, u) can be transformed into a one dimensional value $b + \frac{1}{2}u \in [0, 1]$, which is the probability expectation value of the opinion.

Example 1. *Suppose that Alice (A) takes part in a social networking site where she has two friends, Bob (B) and Clare (C). She knows Danny (D) who is a friend of both Bob and Clare. The resulting trust network is illustrated in Fig. 1, where straight lines refer to a direct trust relation and dashed lines to transitive trust relations. The double dashed line is the overall aggregated trust relation.*

Assume that Alice had 8 *interactions with Bob,* 5 *of which she considers successful, then the trust of Alice in Bob can be calculated as the triplet*

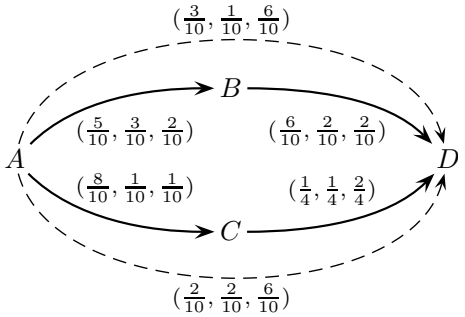

Figure 1: Example of a trust network with derived transitive trust.

$(\frac{5}{10}, \frac{3}{10}, \frac{2}{10})$. *If we assume that Bob had* 6 *successful interactions out of* 8 *with Danny, then the trust of Bob in Danny is* $(\frac{6}{10}, \frac{2}{10}, \frac{2}{10})$. *Applying the rules for conjunction, we can calculate the transitive trust of Alice in Danny via Bob by* $(\frac{5}{10}, \frac{3}{10}, \frac{2}{10}) \otimes (\frac{6}{10}, \frac{2}{10}, \frac{2}{10}) = (\frac{3}{10}, \frac{1}{10}, \frac{6}{10})$. *Likewise, we can calculate the transitive trust of Alice in Danny via Clare. We assume* 16 *out of* 18 *successful interactions between Alice and Clare, which gives* $(\frac{8}{10}, \frac{1}{10}, \frac{1}{10})$ *and* 1 *successful interaction out of* 2 *between Clare and Danny, which gives* $(\frac{1}{4}, \frac{1}{4}, \frac{2}{4})$. *Then the transitive trust of Alice in Danny via Clare is* $(\frac{8}{10}, \frac{1}{10}, \frac{1}{10}) \otimes (\frac{1}{4}, \frac{1}{4}, \frac{2}{4}) = (\frac{2}{10}, \frac{2}{10}, \frac{6}{10})$. *This allows us to calculate the consensus of the two transitive trust relations,* $(\frac{3}{10}, \frac{1}{10}, \frac{6}{10}) \oplus (\frac{2}{10}, \frac{2}{10}, \frac{6}{10}) = (\frac{5}{14}, \frac{3}{14}, \frac{6}{14})$. *Finally, after having calculated all relevant trust relations in the network, we obtain the one-dimensional trust values that A has in B* $(\frac{6}{10})$, *C* $(\frac{8.5}{10})$, *and D* $(\frac{8}{14})$. *Such calculated trust values will be used for subsequent calculations in our voting schemes.*

2.2. Privacy policies

Relationships among people are complex. Users may have family members, relatives, colleagues, classmates, good friends, and so forth. To accommodate this diversity, most social networking sites have implemented the option to give users the ability to segregate their friends into abstract and self-defined groups. For example, the social networking site RenRen provides the user with predefined groups, while FACEBOOK allows users to create their own groups and to place a friend into one or several groups. Typically, privacy control in these sites is made easy with groups, which enables a user to have certain information (or data items such as pictures) exposed to specific groups of other users. One example is that FACEBOOK allows users to associate privacy control with groups, so that a user can manage access rights by associating his resources with existing groups. Once a data item is assigned to a group by the user with the 'visibility' right, all his friends in that group can see the item. Tab. 1 summarizes the allowed privacy control options in several popular social networking sites (i.e., FACEBOOK, RENREN and KAIXIN001). In practice, a user can select the options for his data item depending on his personal will.

In this paper, we simply refer to privacy policies as the set of users who are allowed read access to shared resources. For instance, for a co-owned picture,

Sites	Privacy Control Options
FACEBOOK	everyone, friends of friends, friends, customized
RENREN	everyone, subnetworks, friends, myself
KAIXIN001	everyone, friends, myself

Table 1: The privacy control options of three social networking sites.

the available policies are (i) visible only to the owner (P_1), (ii) visible only to those tagged, also called co-owners, in the picture (P_2), (iii) visible to friends of the tagged users (P_3) and (iv) available to everyone (P_4). Throughout the paper, we use $P_1, P_2, \ldots, P_n$ to range over such policies.

2.3. Condorcet's preferential voting scheme

There exist a number of voting systems in the literature. In single candidate systems each vote refers to a single choice (or candidate), which sometimes may not allow voters to express more comprehensive opinions. For example, a voter cannot express that he is initially willing to vote for C_1, but in case that C_1 fails to be elected, he will vote for C_2 among the rest of the candidates. In such a situation preferential systems can be applied to express more precise and more comprehensive ideas from the voters. In this paper we present a preferential voting scheme which is extended from a system that is originally proposed by Condorcet in late eighteenth century, and is rediscovered and advocated by Kemeny [24] and Young et al. [34]. For a complete description of Condorcet's voting system we refer to [34], which also explains why in a certain sense Condorcet's scheme may be regarded as 'optimal'.

In a preferential voting system, a ballot consists of a (preferential) list of all candidates.[3] For every pair of candidates appearing in the list, say C_1 and C_2, their relative positions reflect the voter's preference, e.g., the case that C_1 precedes C_2 in the list indicates that the voter prefers C_1 to C_2. Such a list implies a total order on the set of candidates expressing the complete opinion from a particular voter. A *voting profile* is a collection (or a multi-set) of all the cast ballots.

A voting profile may also be described as a *weighted matrix* of size $|\mathbb{C}| \times |\mathbb{C}|$, where $\mathbb{C}$ is the set of candidates. A cell in a weighted matrix with row C_1 and column C_2 is filled with $w(C_1, C_2)$, which is the number of votes that prefer C_1 to C_2. We further define a *Condorcet directed graph* $G = (V, E, W)$ such that $V = \mathbb{C}$ is the set of vertices, and E is the set of edges, which is defined as the set $\{(C_1, C_2) \in V \times V : w(C_1, C_2) \geq w(C_2, C_1)\}$. The function $W : E \to \mathbb{N}$, determines the labelings of the edges and is defined by $W(C_1, C_2) = w(C_1, C_2) - w(C_2, C_1)$, i.e., the difference between the votes preferring C_1 to C_2 and the votes that prefer C_2 to C_1. At the end of a voting procedure, a voting

[3] This is in contrast to so called single candidate voting systems where each vote only refers to a single choice.

	P_1	P_2	P_3	P_4
A	1	2	3	4
B	2	3	1	4
C	3	1	2	4

	P_1	P_2	P_3	P_4
P_1	–	2	1	3
P_2	1	–	2	3
P_3	2	1	–	3
P_4	0	0	0	–

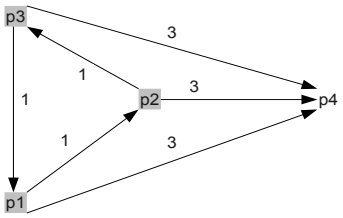

Figure 2: A Condorcet voting example: the voting profile (left), the weighted matrix (middle), and the Condorcet directed graph (right).

profile (or equivalently, a weighted matrix) needs to be evaluated in a certain way, in order to get a final result. There is a famous criterion applied in the calculation of a final winner as originally advocated by Condorcet.

Principle 1. *(Condorcet Winner) If there is a candidate that beats every other candidate in one-to-one comparison, that candidate should be the winner.*

Plainly, a Condorcet winner is a vertex in the Condorcet directed graph with out-degree $|\mathbb{C}| - 1$. We illustrate Condorcet's method in the following example.

Example 2. *Suppose there are three users Alice (A), Bob (B) and Clare (C) tagged in a picture owned by Alice. Alice wants to publish the picture on her personal page in the network, therefore she needs to negotiate with Bob and Clare to reach an agreement on the privacy policy associated to the picture. Suppose the available policies are defined in Sect. 2.2. A (preferential) voting form is made for Alice, Bob and Clare in which they fill in a total ranking on the available policies, as shown in the voting profile on the left of Fig. 2.*

The weighted matrix and the Condorcet directed graph are also sketched in Fig. 2. As one can see, P_4 is the least preferred by all users. However, there exists a *general tie* between policies P_1, P_2 and P_3. This situation is referred to as the *Condorcet paradox*, meaning that the generated Condorcet directed graph is cyclic on its top level vertices, and unfortunately, as we will show, the traditional method of Condorcet is unable to break such a tie in this example.

The interpretation of Young and Levenglick [35] on Condorcet's method adopts what is known as *maximum likelihood estimation*. A philosophical assumption is that there exists an invisible total ranking, reflecting the true capabilities of all the candidates in an election. In this paper we adopt such a mechanism on selecting optimal privacy policies. For every pair of policies P_1 and P_2, such that P_1 precedes P_2 on the (invisible) ranking list, a user is more likely to vote for P_1 than P_2. As Condorcet assumes, each user will choose a better option (P_1 in this case) with some fixed probability p, where $\frac{1}{2} < p < 1$. Taking Example 2, the likelihood of the total order $P_1P_2P_3P_4$ to be the same as the true invisible ranking order, denoted by $L(P_1P_2P_3P_4)$, is calculated by combining the likelihood of every P_i beating P_j with $i < j$ (note that there are six preferential pairs in this case). If "$P_1P_2P_3P_4$" is the true ordering on

the candidates, then the chance that we get the current voting profile can be calculated as

$$\begin{aligned} L(P_1P_2P_3P_4) &= L(P_1P_2)\cdot L(P_2P_3)\cdot L(P_1P_3)\cdot L(P_1P_4)\cdot L(P_2P_4)\cdot L(P_3P_4) \\ &= \tbinom{3}{2}\left(p^2(1-p)^1\right)\cdot\tbinom{3}{2}\left(p^2(1-p)^1\right)\cdot\tbinom{3}{1}\left(p^1(1-p)^2\right)\cdot \\ &\quad \tbinom{3}{3}\left(p^3(1-p)^0\right)\cdot\tbinom{3}{3}\left(p^3(1-p)^0\right)\cdot\tbinom{3}{3}\left(p^3(1-p)^0\right) \end{aligned}$$

where $\binom{m}{n} = \frac{m!}{n!(m-n)!}$ for non-negative $m \geq n$. The expression for $L(P_1P_2)$, for example, follows from the fact that 2 out of 3 voters ranked policy P_1 higher than P_2, and thus voted in accordance with the hypothetical true ranking order $L(P_1P_2P_3P_4)$. The above method of calculating a final ranking was originally proposed by Condorcet, and rediscovered by Kemeney [24] and Young et al. [35, 34]. Nowadays, it is known as Kemeney's rule.

It has been pointed out that in practice when comparing the likelihood of two possible orderings, the combinatoric coefficients can be safely ignored [34], given the same voting profile. The only part that needs to be taken into account consists of the exponents over p (note that $\frac{1}{2} < p < 1$). In the case of $L(P_1P_2P_3P_4)$, the power over p is 14. One may also find that the power over p for $L(P_4P_3P_2P_1)$ is 4, thus $P_1P_2P_3P_4$ is more likely to be the true ordering over the privacy policies than $P_4P_3P_2P_1$ by Condorcet's method. As we have mentioned above, in this example we can compute the likelihood for every sequence that is a permutation of the set $\{P_1, P_2, P_3, P_4\}$. In fact, we have $L(P_1P_2P_3P_4) = L(P_2P_3P_1P_4) = L(P_3P_1P_2P_4)$ and it is larger than the likelihood of any other sequence. This means that Condorcet's method might select *multiple winners*, as P_1, P_2 and P_3 are all selected in Example 2.

2.4. Condorcet voting algorithm

The Condorcet voting algorithm is detailed as in Alg. 1. The algorithm takes a voting profile (*votingprofile*) as input and produces a set of winners as output (stored in *winners*). Function getCondorcetWeightedMatrix translates a voting profile into a weighted matrix, and then in the next step function getCondorcetDirectedGraph converts the weighted matrix into a Condorcet directed graph.[4] For example, as shown in Fig. 2, the algorithm translates the voting profile on the left part into the weighted matrix in the middle, and then into the Condorcet directed graph on the right. The rest of the algorithm focuses on how to select a set of top vertices in the Condorcet directed graph. By definition, the set of Condorcet winners are vertices that have an outgoing edge to every other vertex, although in the real world such sets are singletons in most cases. Such set will be returned by function getWinners. If getWinners returns an empty set, i.e., no Condorcet winners exist, the algorithm will compute the likelihood of all possible sequences and maintain the set of those with the maximal likelihood, and return their first elements as winners.

[4] We represent directed graphs as two-dimensional arrays. For example, if there is a directed edge from i to j with weight $n \geq 0$, then $cdg[i][j]$ has value n, and $cdg[j][k] = -1$ means that there is no edge from j to k.

Algorithm 1 The Condorcet voting algorithm.

```
input: votingprofile : VotingProfile;
output: winners : set ⟨string⟩;
var cwm: int[ ][ ] init null
    cdg: int[ ][ ] init null
    tlv: int[ ] init null
    sql: int init 0
    ml: int init 0
    ms: set ⟨string⟩ init ∅

begin
cwm := getCondorcetWeightedMatrix(votingprofile);
cdg := getCondorcetDirectedGraph(cwm);
winners := getWinners(cdg);
if winners = ∅ then
  tlv := findTopLevelVertices(cdg);
  for each sequence sq which is a permutation of tlv do
    sql := computeSequenceLikelihood(sq, cwm);
    if sql > ml then
      ml := sql;
      ms := {sq};
    else if sql = ml then
      ms := ms ∪ {sq};
    end if
  end for
  winners := getFirstElements(ms);
end if
end
```

In the above algorithm, not all sequences are required to be involved in the comparison of likelihoods. As a Condorcet directed graph imposes a topological order, the top level vertices compose a subgraph which is a strongly connected component (SCC) in the original graph. Function findTopLevelSCC returns the set of vertices that form the SCC. One may easily find that the set of winners can only come from the top-level SCC, thus we only need to compute the sequences initialized by permutations of vertices in the top-level SCC.[5] As in Example 2, only six three-element sequences need to be taken into account: $P_1P_2P_3$, $P_1P_3P_2$, $P_2P_1P_3$, $P_2P_3P_1$, $P_3P_1P_2$ and $P_3P_2P_1$, as P_4 can only be the least preferred. The algorithm will pick up three (total) sequences, $P_1P_2P_3P_4$, $P_2P_3P_1P_4$ and $P_3P_1P_2P_4$, which are with the most likelihood, and produce the

[5]There is no need to compute a whole sequence containing elements not in the SCC, as Condorcet's methods is *locally stable* [34], the particular order of less preferred candidates would not influence the final voting result.

winner set $\{P_1, P_2, P_3\}$ by taking the first elements (using function getFirstElements). The restriction to the top-level SCC effectively narrows the range of winners we need to consider, which greatly improves the performance of the selection procedure. It is easy to see that the running time of the algorithm has an upper bound in $O(|V|!)$, due to checking the likelihood of all possible sequences of nodes in the SCC.

3. A Trust-Augmented Voting Scheme

In this section, we describe our idea of combining trust in social networks with Condorcet's preferential voting scheme to merge privacy opinions from co-owners of shared resources. In Sect. 3.1, we motivate our idea of incorporating trust as weighted votes and explain its impact by examples. A few properties of our trust-augmented Condorcet voting scheme are discussed in Sect. 3.2, followed by a detailed algorithm in Sect. 3.3.

3.1. Incorporating trust as weighted votes

In some situations not all voters are equal. Typical examples include decision-makings in a shareholder's meeting where the weight of each voter corresponds to his volume of share. Likewise, in social networking sites users' opinions on deciding a privacy policy do not necessarily carry the same weight. For example, Alice has a picture in which there are Bob, Clare, Danny and Elisabeth, and she wants to publish that picture in her album. Before publishing it, Alice is willing to give right to the co-owners of the picture, i.e. Bob, Clare, Danny and Elisabeth, on deciding whether the privacy level of that picture is P_1 or P_2 (see their definitions in Sect. 2.2). We suppose that Bob is a friend of Alice and Clare is a friend of Bob but not a direct friend of Alice, i.e., Clare is a friend of a friend of Alice. Similarly, Danny is a friend of Alice and Elisabeth is a friend of a friend of Alice. In this case it seems more reasonable to give Bob's and Danny's opinion more weight than Clare's and Elisabeth's. We assume that the trust values of Alice in the other agents are as given in Fig. 3.

In this section we propose an extension of Condorcet's voting system for weighted votes. The weight of each vote reflects the trust level of the owner of the shared resource having on the co-owners in their votes for setting a privacy policy for publishing the resource.

We sketch the voting results from this scenario in Fig. 3, and it is easy to find that P_1 is the winner according to the left table, as it receives three votes while P_2 only receives two. However, if weights (interpreted as the trust level of Alice in the co-owners to make the right decision on privacy preferences) are associated to votes, then P_2 is the winner, as it is supported by 1.7 weighted votes while P_1 is supported only by 1.5. This example clearly shows that trust relations in social networks, if carefully incorporated into decision making procedures, can affect the results in collaborative privacy management.

In the original Condorcet voting system all votes carry the same weight, and the preferential orders can be compared by only looking at the (integer)

Voter \ Policy	P_1	P_2
A	1	2
B	2	1
C	1	2
D	2	1
E	1	2
vote count	3	2

Voter(trust) \ Policy	P_1	P_2
A (1.0)	1	2
B (0.8)	2	1
C (0.2)	1	2
D (0.9)	2	1
E (0.3)	1	2
vote count	1.5	1.7

Figure 3: An example showing the effect of adding trust levels to voters.

exponents of p regarding to their likelihoods. In this paper we measure the likelihoods of these orders by allowing discounted votes to reflect the degree of trust of a user by the owner of a resource, so that each vote carries a real valued weight in $[0, 1]$ instead of always being an integer 1.

3.2. Properties of the trust-augmented Condorcet voting scheme

It has been discussed in [35, 34] that Condorcet's voting scheme satisfies most of the existing desirable properties for voting systems. In this section, we list a few properties which are satisfied by our trust-augmented Condorcet voting scheme.

Majority. This property says that if there exists a majority that ranks a candidate (privacy policy) higher than all the others then this candidate will be elected. This simply follows Condorcet's principle.

Reinforcement. This property, which is also known as *monotonicity*, says that if a ranking is approved by two groups of voters then it must also be approved by the union of the two.

Local stability. This property states that leaving out votes for less favored candidates does not change results on final winners.

Pareto. The Pareto property expresses that if there exist two candidates P_i and P_j such that no voter prefers P_j to P_i, and at least one voter prefers P_i to P_j, then P_j can never become the winner.

Condorcet's voting scheme suffers from the *no show paradox* [27] when there are at least four candidates and 25 voters [14]. This means that in some cases a voter is better off by not voting than by casting a sincere ballot, as this would lead to the election of a candidate whom he prefers. This effect applies in a similar way to our scheme, because the original Condorcet scheme can be considered as a special case of our trust-augmented scheme where all trust values

are taken to be 1.[6] However, there exist ways to circumvent this problem. For example, if a user does not explicitly vote, we let the system cast a "default vote" which can be calculated from the decision history of the user as an estimation of how a user is likely to behave in similar situations. In this way we can avoid in our application the negative effect inherited from the original Condorcet scheme.

3.3. The trust-augmented Condorcet voting algorithm

The trust-augmented Condorcet voting algorithm is detailed as in Alg. 2. The whole procedure of calculation is mostly the same as that of Alg. 1, except that now the sequence likelihood (sql) and maximal likelihood (ml) are of real valued type instead of integer type. A trust-based voting profile, which includes the preference lists and trust level for each participant, is taken as input by the algorithm (e.g., left part of Fig. 4), while the output, a set of winners, remains unchanged.

Algorithm 2 The trust-augmented Condorcet voting algorithm.

input: *trustvotingprofile* : VotingProfile;
output: *winners* : set $\langle string \rangle$;
var *cwm*: double[ ][ ] **init** null
cdg: int[ ][ ] **init** null
tlv: int[ ] **init** null
sql: double **init** 0.0
ml: double **init** 0.0
ms: set $\langle string \rangle$ **init** $\emptyset$

begin
cwm := getCondorcetWeightedMatrix(*trustvotingprofile*);
cdg := getCondorcetDirectedGraph(*cwm*);
winners := getWinners(*cdg*);
(* the rest is the same as Alg. 1 *)
end

The example in Fig. 4 illustrates our algorithm. Here we assume that A, as the owner of the picture, fully trusts himself, i.e., his trust value is 1.0. The trust of A in other participants (0.8 for B and 0.6 for C) is also shown in the table. From the trust-based voting profile, the revised Condorcet weighted matrix is obtained (the middle part of Fig. 4) by function getCondorcetWeightedMatrix. The weighted matrix is slightly different from that in Example 2 in the way that simply counted (integer) votes are replaced by accumulated trust values throughout the table. From the Condorcet directed graph (obtained by function getCondorcetDirectedGraph) in the right part of Fig. 4, we can find a unique winner P_1, after computing the likelihood of all sequences of nodes in the top

[6] That is, a defect in a system is also a defect in a more general system which may take this special form.

	P_1	P_2	P_3	P_4
A (1.0)	1	2	3	4
B (0.8)	2	3	1	4
C (0.6)	3	1	2	4

	P_1	P_2	P_3	P_4
P_1	–	1.8	1.0	2.4
P_2	0.6	–	1.6	2.4
P_3	1.4	0.8	–	2.4
P_4	0.0	0.0	0.0	–

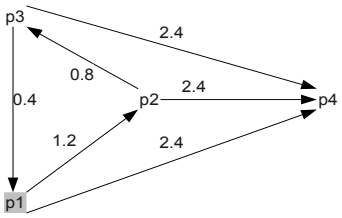

Figure 4: A trust-augmented Condorcet voting example: the voting profile (left), the weighted matrix (middle), and the Condorcet directed graph (right).

level SCC (containing P_1, P_2 and P_3). This can be easily verified since the likelihood of the sequence $P_1P_2P_3P_4$, as calculated as follows, is greater than the likelihood of every other sequence. Note that here we replace the number of votes as integer exponents over "p" and "$1-p$" by their corresponding sums of trust values.

$$\begin{aligned} L(P_1P_2P_3P_4) &= L(P_1P_2)\cdot L(P_2P_3)\cdot L(P_1P_3)\cdot L(P_1P_4)\cdot L(P_2P_4)\cdot L(P_3P_4) \\ &= \tbinom{3}{2}\left(p^{1.8}(1-p)^{0.6}\right)\cdot\tbinom{3}{2}\left(p^{1.6}(1-p)^{0.8}\right)\cdot\tbinom{3}{1}\left(p^{1.0}(1-p)^{1.4}\right)\cdot \\ &\quad \tbinom{3}{3}\left(p^{2.4}(1-p)^{0}\right)\cdot\tbinom{3}{3}\left(p^{2.4}(1-p)^{0}\right)\cdot\tbinom{3}{3}\left(p^{2.4}(1-p)^{0}\right) \end{aligned}$$

It is easy to see that this algorithm has the same time complexity upper bound $O(|V|!)$ as Alg. 1.

As we have seen so far, the trust level indeed has an impact on collaborative privacy management. Moreover, adding trust makes the algorithm better adopted in social networks, since in the real life, advices from different friends affect one's decision differently, depending on the social relationship between the person and his friends. Introducing a trust relation also expands the value space to avoid clashes, as we are going to show experimentally later. In Alg. 2 it is less likely to have unsolved cases as well as multiple winner cases than in Alg. 1. Similar to Alg. 1, the algorithm always selects Condorcet winners whenever they exist. However, in order to decide winners, both Alg. 1 and Alg. 2 may require a calculation of likelihoods for all permutations from the top-level SCC, which potentially takes running time exponential to the number of policies. This fact leads us to the search of more applicable algorithms.

4. Approximation Algorithms

It is clear from Sect. 2.4 that in order to resolve a general tie among policies (represented at the top-level SCC), all possible sequences of nodes in the SCC have to be checked. This gives rise to high complexity for Alg. 2, especially when the number of privacy policies gets large. This problem has been tackled in the literature as (nontrust-based) solutions to compute a final ranking satisfying Kemeny's rule in preferential voting schemes, and it has been shown as NP-hard even with four voters [3, 9]. Basically, Kemeny's rule is to minimize the number of pairwise disagreements with the given preferences on the

candidates. Numerous solutions are postulated for getting an approximation, which proximately minimize the number of pairwise disagreements. Ailon et al. [1] proposed a method to find Kemeny rankings in polynomial time which is approximately optimal. Later, Coppersmith et al. [7] presented a better approximation based on Borda's well-known counting rule. In this section, we introduce two approximation algorithms to improve the efficiency of Alg. 2: the first is based on Borda's counting method (an idea of Coppersmith et al. [7] which we enrich with weights), and the second is heuristic-based. Normally, good approximations should guarantee the quality of produced solutions and produce such solutions within reasonable run time bounds. A comparison between all the algorithms with respect to speed and precision is conducted in Sect. 5. which also justifies the precision and efficiency of our approximations.

4.1. An approximation based on Borda count

We adopt our idea of incorporating trust as weighted votes to Borda's counting rule following the work of Coppersmith et al. [7] and use it to resolve general ties in top SCC as produced from the trust-augmented Condorcet voting algorithm (Alg. 2).

In essence, Borda's counting method is a scoring system. Given a particular candidate, the system calculates a subjective value of how much a voter is in favor of that candidate. Then, the final score of the candidate is the sum of the scores from all the voters. To be precise, suppose there are n candidates. Given a vote that is a linear order on the voters, the least preferred candidate receives a score of zero from this vote, and the second least preferred receives one, etc. The top candidate receives $n-1$. By summing up the scores from all the votes, the one that receives the highest score gets elected.

We generalise this procedure to a *weighted Borda count*. The score that each security policy receives from a vote needs to be adjusted by a weight, which is essentially the *trust value* of the user who casts the vote. A score for each policy is thus the weighted sum of the scores from all the votes.

Example 3. *Consider the example given in Fig. 4, where policies P_1, P_2 and P_3 form a general tie (thus we do not consider P_4 as it is always least preferred). We apply weighted Borda count on each policy as follows.*

- *P_1 receives score 2 from A, 1 from B, and 0 from C, which gives $2 \times (1.0) + 1 \times (0.8) + 0 \times (0.6) = 2.8$.*
- *P_2 receives score 1 from A, 0 from B, and 2 from C, which gives $1 \times (1.0) + 0 \times (0.8) + 2 \times (0.6) = 2.2$.*
- *P_3 receives score 0 from A, 2 from B, and 1 from C, which gives $0 \times (1.0) + 2 \times (0.8) + 1 \times (0.6) = 2.2$.*

The final scores produce two results: $P_1P_2P_3P_4$ and $P_1P_3P_2P_4$, as the final rankings. In both cases we have P_1 as the unique winner, the same as what is produced by Alg. 2.

We develop an approximation based on Borda count as Alg. 3, which has a structure similar to Alg. 1. The main procedure is applied to the top-level SCC when a Condorect winner cannot be found. For each node in the SCC, a (weighted) count is calculated using function getWeightedBC as discussed above, and the result is stored in *weightedBC*. The final winners are determined by the highest score by applying function getMax.

Algorithm 3 An approximation based on Borda count.

```
input: trustvotingprofile: VotingProfile;
output: winners : set ⟨string⟩;
var cwm: int[ ][ ] init null
    cdg: int[ ][ ] init null
    tlv: int[ ][ ] init null
    count: double init 0.0
    weightedBC: map ⟨string, float⟩ init ∅

begin
cwm := getCondorcetWeightedMmatrix(trustvotingprofile);
cdg := getCondorcetDirectedGraph(cwm);
winners := getWinners(cdg);
if winners = ∅ then
  tlv := getTopLevelVertices(cdg);
  for all n ∈ tlv do
    count := getWeightedBC(n, tlv, trustvotingprofile);
    weightedBC := weightedBC + ⟨n, count⟩;
  end for
  winners := getMax(weightedBC);
  return
end if
end
```

Remark. It can be noticed that we could have used Borda count, or even other voting systems, as a basis to develop our trust-augmented voting scheme. This is certainly true, but we want to emphasize that the main reason we have chosen Condorcet's method is due to Condorcet's principle, i.e, the Condorcet winner has the nice property that when compared with every other candidate it is preferred by more voters and it would win a two-candidate election against the other candidates. It is known that Borda count sometimes produces a solution that does not satisfy Condorcet's principle, hence we adopt it solely as a tie-breaker in Alg. 3.

4.2. A heuristic-based approximation

The algorithm presented in this section provides another way to resolve general ties in the top-level SCC, as well as to reduce the computation time.

Taking a Condorcet directed graph, we start from the following observations. Suppose $w(P_i, P_j)$ is close to 0, then it is very likely that the voters are relatively indifferent with respect to the two policies P_i and P_j. Therefore, regarding to Condorcet's assumption, P_i does not have a significant chance to precede P_j in the underlying invisible order. This motivates and justifies our choice in the following algorithm to weaken such difference by adding another (reversing) edge in the graph from P_j to P_i. By doing this, we *equalize* the votes between policies P_i and P_j. Note that this is also consistent with the idea of proximately minimizing the number of pairwise disagreements when developing approximations. Technically, we only apply this operation within the top-level SCC, gradually by starting from the pairs (P_i, P_j) with least $w(P_i, P_j)$, then the pairs with second least weight, and so on. Each time we add new edges it is required to check whether a set of Condorcet winners have been generated in the new graph. The running time of the algorithm has an upper bound of $O(|V|^2)$, where V is the set of vertices of the Condorcet directed graph, which is much faster than Alg. 2. Nevertheless, the experimental results in Sect. 5 reveal strong similarity with respect to the results of Alg. 4 and Alg. 2, which provides a concrete support to the applicability of Alg. 4.

Algorithm 4 A heuristic based approximation.

```
input: trustvotingprofile: VotingProfile;
output: winners : set ⟨string⟩;
var cwm: int[ ][ ] init null
    cdg: int[ ][ ] init null
    tls: int[ ][ ] init null
    lwes: set ⟨string⟩ init ∅

begin
cwm := getCondorcetWeightedMmatrix(trustvotingprofile);
cdg := getCondorcetDirectedGraph(cwm);
winners := getWinners(cdg);
if winners = ∅ then
  tls := getTopLevelSCC(cdg);
  while true do
    lwes := findLowestWeightEdges(tls);
    cdg := addReverseEdges(lwes, cdg);
    winners := getWinners(cdg);
    if winners ! = ∅ then
      return
    end if
  end while
end if
end
```

Similar to Alg. 2, Alg. 4 takes a voting profile as input and produces a set of winning privacy policies. The first part of the algorithm is exactly the same as in

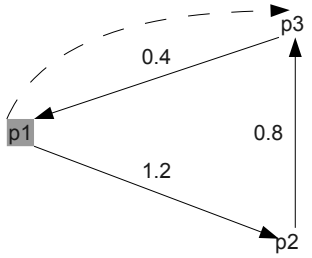

Figure 5: An example: adding a reverse edge.

the above two algorithms. However, if Alg. 4 cannot find Condorcet winners, it will extract the whole top-level SCC by function getTopLevelSCC into a subgraph tls. Then by starting from the lowest weighted edges, it adds reverse edges into the original Condorcet directed graph cdg using function addReverseEdges, and then searches for Condorcet winners in the modified graph. This will repeat until Condorcet winners are found in cdg. The algorithm is guaranteed to terminate before every pair of vertices in the top-level SCC has two connecting edges pointing to each other. Therefore it is bounded by $O(|V|^2)$ where V is the set of vertices in cdg. An application of Alg. 4 on Example 2 with additional trust values (as data shown in Fig. 4) has been depicted in Fig. 5.

5. Experimental Results

We have implemented a program to test the four algorithms: the original Condorcet voting algorithm (Alg. 1), the trust-augmented Condorcet voting algorithm (Alg. 2), the approximation algorithm based on Borda count (Alg. 3) and the heuristic-based approximation (Alg. 4).

In our test, the voting profiles and the trust levels are randomly generated.[7] The experimental results are obtained from 10,000 cases. We set the policy number in the range of $[3, 10]$, while the number of voters ranges in $[3, 100]$. We verify that the approximation algorithms (Alg. 3 and Alg. 4) have reasonably well performances, even in cases of large numbers of policies.

From Fig. 6, we can find that the number of cases where we cannot find a Condorcet winner in Alg. 2, Alg. 3 and Alg. 4 is much less (14%) than in Alg. 1. This is due to the incorporation of trust. Based on this we can conclude that trust can have a big impact on the voting results. It is also clear that the cases with multiple winners as output is on a steady decrease (about $13.7 \sim 13.95\%$), which means that the adoption of a trust relation, to a large extent, can effectively increase the possibility of having a unique winner. Moreover, we only see a slight increase in the number of cases with multiple winners for Alg. 4 compared to Alg. 2, while Alg. 3 has similar outputs as Alg. 2.

[7] Alg. 1 does not take the trust levels into account.

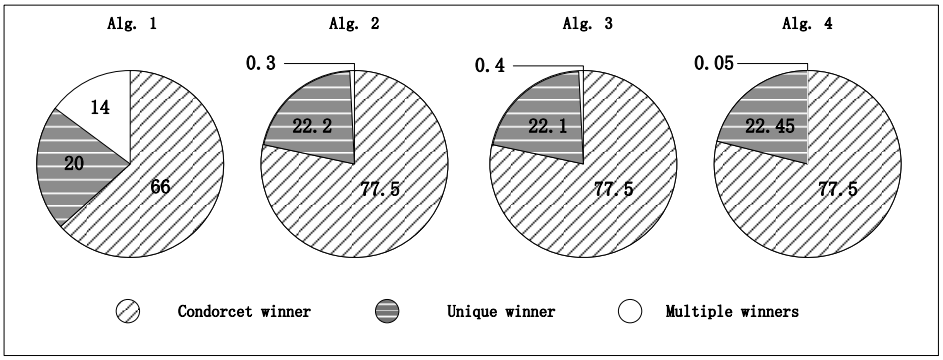

Figure 6: Case analysis on outputs of the algorithms.

Alg./#Policy	4	5	6	7	8	9
Alg. 2	5,300	8,366	15,874	75,864	713,358	101,141,105
Alg. 3	4,703	6,715	8,541	11,286	12,958	16,031
Alg. 4	6,083	9,407	12,486	18,521	21,992	29,217

Table 2: Total CPU time consumption (ms) w.r.t. policy numbers.

Besides, we measured the similarity among the outputs of Alg. 3, Alg. 4 and the outputs of Alg. 2. In 8,137 out of 10,000 cases (i.e., $> 81\%$) Alg. 2 and Alg. 3 produce the same results, while in 9,511 out of 10,000 cases (i.e., $> 95\%$) Alg. 2 and Alg. 4 produce the same results.[8] Since theoretically Alg. 2 always generates the best privacy policy, we can conclude that our heuristic based approximation (Alg. 4) can also produce the best privacy policy for most cases in practice. Moreover, in terms of producing the best privacy policy, Alg. 4 is better than Alg. 3 (about 14%). This shows that our approximations, and Alg. 4 in particular, can guarantee the quality of produced solutions.

Next, we have built a different set of experiments to compare the performance of Alg. 2, Alg. 3 and Alg. 4. For each number of policies $[4, \ldots, 9]$, we tested the three algorithms on 1,000 randomly generated voting profiles. We calculated the total CPU time for each algorithm to produce the outputs. From Tab. 2, it is clear that our approximation algorithms greatly improve the efficiency of Alg. 2 – the performance of Alg. 2 degrades largely when the number of policies increases. Hence, we can conclude that our approximations can produce solutions within reasonable run time bounds. We also observe that Alg. 3 performs even better than Alg. 4, while Alg. 4 has better precision in terms of similarity in outputs with Alg. 2. All experiments are conducted on a Lenovo laptop with Intel(R) Core(TM) 2 Duo CPU P8700 (2.53GHz) and 2.00GB of RAM.

[8] Alg. 3 and Alg. 4 produce 81,53 (out of 10,000 cases) similar results.

6. Discussion and Conclusion

Privacy in social networking sites is a rather complicated issue [4], from which many research questions have emerged in different areas such as economics, social science, computer science, and law. In this paper, we have proposed a trust-augmented voting scheme to solve the particular problem of collective privacy management for shared contents in social networking sites. Our main idea is to incorporate trust relations among users in social networks as vote weights in Condorcet's preferential voting algorithm. The motivation comes from the facts that trust is inherent in social networks and that a preferential voting scheme is an expressive and simple way for users to formulate their privacy concerns on shared contents. To make the algorithm both efficient and effective, we have developed two approximations for the algorithm to deal with situations where the number of privacy policies is large.

The algorithms in this paper have been developed mainly from a technical point of view and one may reasonably argue that voting is not the ideal approach to collective privacy management. Some people believe that the owner of a picture should have the right to decide whether and how the picture is to be published while taking into account the interests of concerned people. For example, a different approach proposed by Sarrouh et al. [30] gives a single user complete control over the picture that she owns. In our voting scheme, occasionally a group decision overrides an owner's decision. In this case, we believe that democracy should be a good solution in many situations, if not in all situations. (Note the owner does not have control over her trust in the other co-owners which is supposed to be computed automatically by the social networking site.) Our solution provides all co-owners of a shared resource a reasonable way to produce a group decision, which practically simulates real world decision making processes in the virtual world. If the owner does not feel comfortable with the outcome of the voting, in reality she may appeal to a trusted third party or the social networking site, which will then decide whether or not to refute the previous decision. In any case, a reasonable social decision is only required to be 'fair' with a certain degree of confidence. We believe that heuristic and algorithmic support to this process will result in a more transparent and hopefully more fair decision process.

We have advocated in the introduction that our solution to collaborative privacy management is more useful and effective when trust relations among users have been established. This helps to rule out extreme scenarios when a group of users act together to break a particular user's privacy, since malicious users are likely to be accrued with low trust values as time passes. One may further exclude malicious users by setting a threshold that filters out co-owners of low trust when deciding privacy policies. We leave assessment of the impact of such a threshold on the outcomes of our algorithms for the future. In this paper, we assume that privacy policies are independent from each other. Modeling and capturing dependencies among policies is not considered in the paper, but it is indeed an interesting topic to study for the future. As discussed in Sect. 3.2 our trust-augmented voting scheme implies the *no show paradox*, it is also an

interesting challenge to establish a bound following the results in [14].

Acknowledgements. The authors thank the anonymous referees for their helpful comments. Chenyi Zhang is supported by the Australian Research Council (ARC) Linkage Project (LP0989643). Baptiste Alcalde was supported by the grant TR-PDR BFR08-038 from the Fonds National de la Recherche, Luxembourg.

References

[1] N. Ailon, M. Charikar, and A. Newman. Aggregating inconsistent information: Ranking and clustering. In *Proc. 37th Annual ACM Symposium on Theory of Computing*, pages 684–693. ACM, 2005.

[2] B. Alcalde and S. Mauw. An algebra for trust dilution and trust fusion. In *Proc. 6th Workshop on Formal Aspects in Security and Trust*, volume 5983 of *LNCS*, pages 4–20. Springer, 2009.

[3] J. Bartholdi, C. A. Tovey, and M. A. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6(2):157–165, 1989.

[4] J. Bonneau and S. Preibusch. The privacy jungle: On the market for privacy in social networks. In *Proc. 8th Workshop on the Economics of Information Security*, 2009.

[5] B. Carminati and E. Ferrari. Privacy-aware collaborative access control in web-based social networks. In *Proc. 22nd Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, volume 5094 of *LNCS*, pages 81–96. Springer, 2008.

[6] E. H. Clarke. Multipart pricing of public goods. *Public Choice*, 11:17–33, 1971.

[7] D. Coppersmith, L. Fleischer, and A. Rudra. Ordering by weighted number of wins gives a good ranking for weighted tournaments. In *Proc. 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 344–353. ACM, 2006.

[8] C. Dong, G. Russello, and N. Dulay. Trust transfer in distributed systems. In *Proc. Joint iTrust and PST Conferences on Privacy, Trust Management and Security*, volume 238 of *IFIP*, pages 17–30. Springer, 2007.

[9] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proc. 10th International Conference on World Wide Web*, pages 6131–622. ACM, 2001.

[10] F. Eilers and U. Nestmann. Deriving trut from experience. In *Proc. 6th Workshop on Formal Aspects in Security and Trust*, volume 5983 of *LNCS*, pages 36–50. Springer, 2009.

[11] N. B. Ellison, C. Steinfield, and C. Lampe. Benefits of Facebook "Friends": Social capital and college students' use of online social network sites. *Journal of Computer Mediated Communication-Electronic*, 12(4), 2007.

[12] E. ElSalamouny, V. Sassone, and M. Nielsen. HMM-based trust model. In *Proc. 6th Workshop on Formal Aspects in Security and Trust*, volume 5983 of *LNCS*, pages 21–35. Springer, 2009.

[13] FACEBOOK, `www.facebook.com`.

[14] P. C. Fishburn and S. J. Brams. Paradoxes of preferential voting. *Mathematics Magazine*, 56(4):207–214, 1983.

[15] D. Gambetta, editor. *Trust: Making and breaking cooperative relations*. Department of Sociology, University of Oxford, 1988.

[16] D. Good. Individuals, interpersonal relations, and trust. In Diego Gambetta, editor, *Trust: Making and breaking cooperative relations*. Department of Sociology, University of Oxford, 1988.

[17] GOOGLE+, `plus.google.com`.

[18] E. Gray, J.-M. Seigneur, Y. Chen, and C. Jensen. Trust propagation in small worlds. In *Proc. 1st Conference on Trust Management*, volume 2692 of *LNCS*, pages 239–254. Springer, 2003.

[19] R. Gross, A. Acquisti, and H. John Heinz III. Information revelation and privacy in online social networks. In *Proc. 2005 ACM Workshop on Privacy in the Electronic Society*, pages 71–80. ACM, 2005.

[20] J. Grossklags, N. Christin, and J. Chuang. Secure or insure?: A game-theoretic analysis of information security games. In *Proc. 17th Conference on World Wide Web*, pages 209–218. ACM, 2008.

[21] A. Johnson, P. Syverson, R. Dingledine, and N. Mathewson. Trust-based anonymous communication: Adversary models and routing algorithms. In *Proc. 18th ACM Conference on Computer and Communications Security*, pages 175–186. ACM, 2011.

[22] A. Jøsang, S. Marsh, and S. Pope. Exploring different types of trust propagation. In *Proc. 4th Conference on Trust Management*, volume 3986 of *LNCS*, pages 179–192. Springer, 2006.

[23] KAIXIN001, `www.kaxin001.com`.

[24] J. G. Kemeny. Mathematics without numbers. *Daedalus*, 88(4):577–591, 1959.

[25] K. Krukow, M. Nielsen, and V. Sassone. A logical framework for history-based access control and reputation systems. *Journal of Computer Security*, 16(1):63–101, 2008.

[26] N. Luhmann. *Trust and Power*. John Wiley and Sons Inc, 1979.

[27] H. Moulin. Condorcet's principle implies the no show paradox. *Journal of Economic Theory*, 45(1):53–64, 1988.

[28] RenRen, `www.renren.com`.

[29] D. Rosenblum. What anyone can know: The privacy risks of social networking sites. *IEEE Security and Privacy*, 5(3):40–49, 2007.

[30] N. Sarrouh, F. Eilers, U. Nestmann, and I. Schieferdecker. Defamation-free networks through user-centered data control. In *Proc. 6th Workshop on Security and Trust Management*, volume 6710 of *LNCS*. Springer, 2010.

[31] A. C. Squicciarini, M. Shehab, and F. Paci. Collective privacy management in social networks. In *Proc. 18th International Conference on World Wide Web*, pages 521–530. ACM, 2009.

[32] Y. Sun, C. Zhang, J. Pang, B. Alcalde, and S. Mauw. A trust-augmented voting scheme for collaborative privacy management. In *Proc. 6th Workshop on Security and Trust Management*, volume 6710 of *LNCS*, pages 132–146. Springer, 2011.

[33] Twitter, `www.twitter.com/`.

[34] H. P. Young. Condorcet's theory of voting. *The American Political Science Review*, 82(4):1231–1244, 1988.

[35] H. P. Young and A. Levenglick. A consistent extension of condorcet's election principle. *SIAM Journal on Applied Mathematics*, 35(2):285–300, 1978.