Inferring Binary Trust Relationships in Web-Based Social Networks

JENNIFER GOLBECK and JAMES HENDLER
University of Maryland, College Park

The growth of Web-based social networking and the properties of those networks have created great potential for producing intelligent software that integrates a user's social network and preferences. Our research looks particularly at assigning trust in Web-based social networks and investigates how trust information can be mined and integrated into applications. This article introduces a definition of trust suitable for use in Web-based social networks with a discussion of the properties that will influence its use in computation. We then present two algorithms for inferring trust relationships between individuals that are not directly connected in the network. Both algorithms are shown theoretically and through simulation to produce calculated trust values that are highly accurate.. We then present TrustMail, a prototype email client that uses variations on these algorithms to score email messages in the user's inbox based on the user's participation and ratings in a trust network.

Categories and Subject Descriptors: I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence; H.5.3 [**Information Interfaces and Presentation**]: Group and Organization Interfaces—*Web-based interaction*

General Terms: Algorithms, Design, Human Factors, Theory

Additional Key Words and Phrases: Social networks, trust, semantic Web, online communities, small worlds

1. INTRODUCTION

In recent years, social networking has moved from the constrained world of academia into the public consciousness. Movies (e.g., Six Degrees of Separation), games (the Kevin Bacon Game[1], and a new explosion of Web sites for

[1]http://oracleofbacon.org.

This work, conducted at the Maryland Information and Network Dynamics Laboratory Semantic Web Agents Project, was funded by Fujitsu Laboratories of America, Lockheed Martin Advanced Technology Laboratory, NTT Corp., Kervic Corp., SAIC, the National Science Foundation, the National Geospatial-Intelligence Agency, DARPA, US Army Research Laboratory, NIST, and other DoD sources.

Authors' address: Computer Science Department, University of Maryland, College Park, MD 20742; email: {golbeck,hendler}@cs.umd.edu.

personal social networking have been released to satisfy the interest in social networks and small worlds. Web sites have become particularly popular tools for social experimentation, with hundreds of millions of members dispersed over 100 sites.

The relationships in Web-based social networks are more complex than a simple indication that two people know one another. In Web-based social networks, users are given the opportunity to add a wide variety of metadata to describe their social relationships, and that data may be directed, representing a relationship that is not necessarily reciprocated. For example, users may state how well they know the person to whom they are connected or how much they trust that person. These expanded relationships mean that analysis of the networks can take the extra information into account to discover more about the nature of the relationships between people. For example, if a network allows users to state the type of relationship they have with another person, this can be used to determine the strength of an indirect connection (e.g., Bob is connected to Alice and the network tells Alice that Bob is the best friend of her roommate Jane).

This research looks particularly at the concept of trust in social networks. Some networks, like LinkedIn, have trust implied in the network connections. Their motto reflects this: "Find the people you need through the people you trust". Creating a link to a person on that Web site implies some amount of business trust for the person. Other networks have a more explicit notion of trust where users assign *trust ratings* to their friends. Orkut allows users to rate one another's trustworthiness with zero to three smiley faces. The Friend Of A Friend (FOAF) Project (specification at http://foaf-project.org), a Semantic Web-based social network, has a trust module developed as part of this project [Golbeck and Hendler 2004] with thousands of users rating each others' trustworthiness in general or with respect to a given topic on a 1-*10 scale.

When trust is explicitly rated on a numerical scale, this network data can be composed to produce information about the trust between two individuals without a direct connection. The specific problem we look at in this research is how to use the data in the network to accurately recommend to one user (the *source*) how much to trust another (the *sink*). With algorithms that can accurately calculate recommendations about trust using data in a social network, it is possible to build socially aware systems where users benefit from their trust and connections.

Our work is specifically designed to take advantage of the explicit trust ratings that users are currently asserting in Web-based social networks and use those ratings to infer the trust that may exist between two people who are not directly connected. We intentionally look at only the current state of the social network, without considering history, rules the user might assert, or learning approaches. We investigate what can be accomplished with only a momentary snapshot of the social network and its trust ratings. We believe that with no training or learning, no time needed to build a history, and no user-specific rules, useful computations can be made that are relatively accurate and useful, so those approaches to the problem of inferring trust are intentionally excluded from our work. This leaves room for the algorithms to be applied in different contexts and applications where trust information may be available on a social network and also broadens the spectrum of applications where the computational results can be incorporated.

There are two major goals of this article: to develop an efficient and accurate algorithm for inferring trust relationships that uses only the structure and trust ratings within a social network, and to integrate the results of the algorithm into applications to show its usefulness. In this article, we will present two variations on an algorithm to make this calculation in networks where users rate one another on a binary scale (trusted or not trusted). We begin by presenting a definition of trust and illustrating how it fits in with making trust ratings in Web-based social networks. For both algorithms, the objective is to infer trust values that are accurate to the person for whom they are calculated. We introduce each algorithm in detail, followed by a theoretical analysis that shows why highly accurate results can be expected. This is reinforced through simulation that demonstrates the correctness in simulated networks. Finally, we demonstrate the potential of using inferred trust values to create trust-aware applications through a prototype of TrustMail, an email client that uses trust ratings as a mechanism to filter email.

2. BACKGROUND AND PREVIOUS WORK

Our work is based on the premise that applications will access and utilize the trust data incorporated into Web-based social networks. In this section, we introduce the necessary background for successfully adding trust to networks on the Web and present some applications that have already begun to take advantage of Web-based trust.

2.1 Defining Trust

Before making any computations with trust in social networks, it is vitally important to know what trust is and the properties it has. Within computer science, trust has been co-opted by many subfields to mean many different things. It is a descriptor of security and encryption; a name for authentication methods or digital signatures; a measure of the quality of a peer in P2P systems; a factor in game theory; a model for agent interactions [Jonker and Treur 1999]; a gauge of attack resistance; a component of ubiquitous computing; a foundation for interactions in agent systems [Maes and Kozierok 1994; Barber and Kim 2000] and a motivation for online interaction and recommender systems. It is only in the last few years that the computing community has begun to look at the more social aspect of trust as a relationship between humans.

The difficulty of combining trust with algorithms and mathematical analyses is that trust is difficult to define, let alone to pin down in a quantifiable way. Intense theoretical analysis and complex models are one way to address this issue. However, the real boost to the union of these two topics has come in the form of Web-based social networks that force people to quantify trust.

Social trust depends on a host of factors which cannot be easily modeled in a computational system. Past experience with a person and with their friends, opinions of the actions a person has taken, psychological factors impacted by a lifetime of history and events (most completely unrelated to the person we are deciding to trust or not trust), rumor, influence by others' opinions, and motives to gain something extra by extending trust are just a few of these factors. For trust to be used as a rating between people in social networks, the definition must be focused and simplified.

Marsh [1994] addressed the issue of formalizing trust as a computational concept in his PhD dissertation at the University of Stirling. His model is complex and based on social and psychological factors. Although this work is often cited, the model is highly theoretical and difficult to implement. It is particularly inappropriate for use in social networks because his focus was on interacting agents that could maintain information about history and observed behaviors. In all Web-based social networks that implement trust, users assign a single rating without explicit context or history to their neighbors, and thus much of the information necessary for a system like Marsh's is missing. To capture the nuanced information required by an implementation of Marsh's system, users would need to log tremendous amounts of information, and even then the complexity of data required by Marsh's model would be unsatisfied. One of the stated goals of our work is to take advantage of social networks as they exist on the Web—with users adding simple expressions of trust as they determine it (rather than computing it for them). This differs from the agent system in Marsh's work, and that divergence in goals prevents an application of his model in this context.

Castelfranchi and Falcone [1998, 2002] present an excellent analysis of trust in multi-agent systems. They break trust down into its components, including the beliefs that must be held to develop trust, how that relates to previous experience, as well as giving descriptions of when trust is rational and irrational. Their work draws largely on the psychological literature and includes many psychological factors in developing a model for trust in multi-agent systems. This approach, however, does not carry over into our work with Web-based social networks. Our premise is to utilize only the explicitly stated information that users are already providing within these Web sites, and this does not include background information, nor does it track historical interactions. In this work, we want to show that useful results can be produced *without* this extensive background. Because the data that we have chosen as our focus is limited to "trust" or "no trust" labels on social connections, the more advanced and complex model of Castelfranchi and Falcone does not translate into our case.

Deutsch [1962] contains a frequently referenced definition of trust. He states that trusting behavior occurs when a person (say Alice) encounters a situation where she perceives an ambiguous path. The result of following the path can be good or bad, and the occurrence of the good or bad result is contingent on the action of another person (say Bob). Furthermore, the negative impact of the bad result is greater than the positive impact of the good result. This further motivates Alice to make the correct choice. If Alice chooses to go down the path, she has made a trusting choice. She trusts that Bob will take the steps necessary to ensure the good outcome. The requirement that the bad outcome must have greater negative implications than the good outcome has positive implications has been countered in other work [Golombiewski and McConkie 1975] which does not always require disparity.

Sztompka [1999] presents and justifies a simple, general definition of trust similar to that of Deutsch: "Trust is a bet about the future contingent actions of others." There are two main components of this definition: belief and commitment. First, a person believes that the trusted person will act in a certain way. The belief alone, however, is not enough to say there is trust. Trust occurs when that belief is used as the foundation for making a commitment to a particular action. These two components are also present in the core of Deutsch's definition we commit to take the ambiguous path if we believe that the trusted person will take the action that will produce the good outcome.

We adopt this as the definition of trust for our work: *trust in a person is a commitment to an action based on a belief that the future actions of that person will lead to a good outcome*. The action and commitment does not have to be significant. We could say Alice trusts Bob regarding email if she chooses to read a message (commits to an action) that Bob sends her (based on her belief that Bob will not waste her time).

2.2 Properties of Trust

There are three main properties of trust that are relevant to the development of algorithms for computing it, namely, *transitivity*, *asymmetry*, and *personalization*. If any of these properties did not exist, or had different behavior, the algorithms that are described in Section 3 would have to be changed if they were to remain in line with how trust behaves. In this section, we will describe each social property and how it impacts the computational features of our algorithm.

The primary property of trust that is used in our work is transitivity. Trust is not perfectly transitive in the mathematical sense, that is, if Alice highly trusts Bob, and Bob highly trusts Chuck, it does not always and exactly follow that Alice will highly trust Chuck. There is, however, a notion that trust can be passed between people. When we ask a trusted friend for an opinion about a plumber, we are taking the friend's opinion and incorporating that to help form a preliminary opinion of the plumber. In application, the transitivity of trust can work in two ways. First, a person can maintain two types of trust in another person: trust in the person, and trust in the person's recommendations of other people. Alice may trust Bob about movies but not trust him at all to recommend other people whose opinion about movies is worth considering.

Despite this dichotomy, in social networks, it is preferable to let a single value represent both of these ideas. If we say Alice trusts Bob with respect to movies, then her trust in Bob works in two ways. First, it applies to judging movies he recommends. That trust can also function as trust that extends to judging the trustworthiness of other people whom Bob says have trustworthy opinions in movies. Because we expect Bob to make a good recommendation about a film, and Bob trusts Chuck to make good recommendations, Alice can compose those values to develop an idea of Chuck's trustworthiness. This is not to say that Bob's opinion transfers directly to Alice—because she does not know Chuck, she may not trust him as much as Bob does. However, Bob's trust is evidence of shared opinion, and this allows Alice to use Bob's recommendation as evidence that Chuck may also be trustworthy with respect to movies. A single rating system is also more compatible with the traditional way users participate in social networks. Users are rarely, if ever, asked to express opinions of others with such subtle differences. Computationally, this idea of propagating trust along chains of connections (thus exploiting some form of transitivity) has been widely studied and implemented [Gray et al. 2003; Guha and Kumar 2004; Jøsang 1996; Jøsang et al. 2003; Richardson et al. 2003; Ziegler and Lausen 2004a].

In social networks, it is also important to note the asymmetry of trust. For two people involved in a relationship, trust is not necessarily identical in both directions. Because individuals have different experiences, psychological backgrounds, and histories, it is understandable why two people might trust each other to different degrees. While asymmetry occurs in all types of human relationships, it is documented more in situations where the two people are not of equal status. For example, employees typically say they trust their supervisors more than the supervisors trust the employees. This is seen in a variety of hierarchies [Yaniv and Kleinberger 2000]. Even outside of hierarchies, social situations can arise with asymmetric trust. One of the more extreme instances of this is one-way trust where circumstances force one person to trust the other, but there is no reciprocal trust [Hardin 2002; Cook 2001]. Because trust is naturally asymmetric, trust ratings in our system are also asymmetric and represented as directed edges in a network.

One property of trust that is important in social networks and has been frequently overlooked in the past is the personalization of trust. Trust is inherently a personal opinion. Two people often have very different opinions about the trustworthiness of the same person. For an example, we need only look to politics. In the United States, when asked "do you trust the current President to effectively lead the country?" the population will be about evenly split—half will trust him very highly, and the other half will have very little trust in his abilities.

Personalization plays into calculating trust recommendations by affecting the accuracy of a recommendation. If a person wants a recommendation about how much to trust the President, an algorithm that simply composes all of the values in the system can be expected to give an answer that falls almost directly in between very low trust and very high trust. Since most people have a strong opinion, this middle rating will not mean much. It reflects the opinion of the population, and is not a recommendation to the individual. Our algorithm is based on the perspective of the user. It looks at friends who the user trusts about their opinions on a topic, the people whom those friends trust, and so on. Thus, the opinions of people whom the user does not trust much are given very little consideration, and the opinions of people whom the user trusts highly are given more consideration.

The personal aspect of trust also highlights one of the areas where our algorithms and these definitions do not apply. We rely on trust to be entirely personal, which means there is no correct or incorrect value except when considered from the perspective of a given individual. As a result, these algorithms do not translate directly into systems with an absolute truth. As an example, consider a Web site where amateur astronomers submit data about their observations. A social network tied into this system could allow people to rate how much they trust an individual about astronomy based on the data they enter. In this type of system, trust is not just an expression of personal preference but a judgment on how frequently a person is correct. If a user wants to know how much to trust another, the information in the social network is not sufficient; the correctness of the data which can be determined without anyone's opinion is also important. The algorithms in this article are not designed to effectively handle cases where trust reflects something beyond a personal preference.

2.3 The Values of Trust

The way relationships are expressed in social networks, particularly in Web-based social networks, is through an explicit statement. Users rate their connections on a scale usually made available by the Web service.

There are many systems for rating how much one person trusts another; in the trust literature, the rating systems vary. The Advogato system uses a three-tiered system (apprentice, journeyman, master) for rating its members [Levin 1998]. Orkut, at http://Orkut.com, offers users the ability to rate many features of their friends, including their trustworthiness, with zero to three smiley faces. Semantic Web trust projects have used a scale of 1–9 [Golbeck and Hendler 2003] 1–10, or a value that can fall anywhere in the {0,1} range [Richardson et al. 2003].

For analytic purposes, we first explore networks that only allow a binary rating of 1 for trustworthy neighbors, or 0 for neighbors who are not trustworthy. Though the {0,1} scale is restrictive, it provides an excellent theoretical foundation for analyzing the effectiveness of the algorithms for inferring trust relationships.

One important note is that a 0 value does not indicate distrust. Recall that the definition of trust given states that: trust in a person is a commitment to an action based on a belief that the future actions of that person will lead to a good outcome. A 0 value in this system means that the person giving the rating does not believe that the person being rated can be relied upon to take the actions necessary to produce the good outcome. That may be because the person being rated explicitly tries to do bad things to harm the network or because of something less sinister like an uncertain user assigning somewhat arbitrary ratings.

2.4 Trust Algorithms

The algorithms presented in this article are designed to be used with any social network, but current implementations are semantic Web-based and use the Friend-Of-A-Friend (FOAF) vocabulary. The FOAF project defines a set of terms for letting users describe people and who they know. FOAF is one of the largest projects on the semantic Web which has an estimated 2–5 million users. Some of this data comes from individuals creating their own FOAF files and maintaining that information in their personal Web space, but, increasingly, it is coming from other Web-based social networks. LiveJournal (http://livejournal.

com), eCademy (http://ecademy.com), and Tribe (http://tribe.net) all publish their users' social network data in FOAF format. Other Web sites that have gathered social network data have chosen to make those connections available in FOAF format, for example, Howard Dean's Presidential campaign produced thousands of FOAF files representing the social network created when members used a feature of their Web site to share links with their friends. FOAF has become a recognized means of sharing social network data between social networking Web sites, and the ease of producing semantic Web data is encouraging this evolution. The FOAF community is actively rising to this challenge by formalizing their efforts in workshops and online meetings to create a stable core vocabulary that can be used by the widest range of people and applications.

Because it is a semantic Web ontology, FOAF can be easily extended to capture more detailed personal and relationship information. The Trust Module for FOAF [Golbeck and Hendler 2004] extends the FOAF vocabulary by adding a property where users state how much they trust one another. It has a scale of trust ratings that range from 1 (very little trust) to 10 (very high trust) and is used in several applications [Croucher 2004; Avestani et al. 2004; Golbeck and Hendler 2004]. The trust ratings available in this article are simpler, restricted to a binary scale (trust or no trust), but these can be modeled using the existing Trust Module for FOAF or with a separate extension of the vocabulary if a binary trust system is valuable to users.

The issue of sharing trust assessments on the semantic Web has also been addressed in contexts outside of explicit social networks. Gil and Ratnakar [2002] addressed the issue of trusting content and information sources on the semantic Web. Their TRELLIS system derives assessments about information sources based on individual feedback about the sources. Users of the system can annotate pieces of information and the annotations can include measures of credibility and reliability about a statement. These are later averaged and presented to the viewer. Using the TRELLIS system, users can view information, annotations (including averages of credibility, reliability, and other ratings), and then make an analysis. Our work uses this notion of augmenting data on the semantic Web (social network data in our case) with annotations about its trustworthiness.

Once a trust network has been properly modeled and represented, our attention moves to algorithms for calculating recommendations about trust in the network. In this work, the aim is to use trust ratings within the social network as the basis for making recommendations about the trustworthiness of unknown individuals. For this technique to be successful, there must be a correlation between trust and user similarity in trust opinions. That is, the technique relies on the fact that people we trust will share our opinions about the trustworthiness of others. This phenomenon has been investigated particularly in the context of recommender systems. Abdul-Rahman and Hailes [2000] showed that, in a predefined context such as movies, users develop social connections with people who have similar preferences. These results were extended in work by Ziegler and Lausen [2004b]. Their work showed a correlation between trust and user similarity in an empirical study of a real online community.

Developing algorithms for trust calculations in social networks has been addressed in several communities with a range of endpoint applications. The EigenTrust algorithm [Kamvar et al. 2003] is used in peer-to-peer systems and calculates trust with a variation on the PageRank algorithm [Page et al. 1998] used by Google for rating the relevance of Web pages to a search. A peer creates a direct trust rating for another peer based on its historical performance. In its simple form, the algorithm uses a matrix representation of the trust values within the system, and, over a series of iterations, it converges to a globally-accepted trust rating of each peer. Because of safeguards built into the system, EigenTrust has been shown to be highly resistant to attack. EigenTrust is designed for a peer-to-peer system, while ours is designed for use in human social networks, and thus there are differences in the approaches to analyzing trust. In the EigenTrust formulation, trust is a measure of performance (i.e., the number of corrupt files, the number of lost connections, etc.), and one would not expect a single peer's performance to differ much, based on the other peers with which it is interacting. If asked to provide information, P2P systems generally do not have a peer provide data where the quality varies based on the peer that requests it. Thus, we expect that a peer provides data that is of similar quality to all peers. Socially, though, two individuals can have dramatically different opinions about the trustworthiness of the same person. Our algorithms intentionally avoid moving toward a global trust value for each individual to preserve the personal aspects that are foundations of social trust.

Advogato is a Web site, at http://advogato.org, that serves as a community discussion board and resource for free software developers. It also is the testbed for Raph Levin's trust metrics research [1998]. Each user on the site has a single trust rating calculated from the perspective of designated *seeds* (authoritative nodes). Trust calculations are made using a network flow model. His metric composes certifications between members to determine the trust level of a person, and thus their membership within a group. Users can be certified at three levels, namely, apprentice, journeyer, and master. Access to post and edit Web site information is controlled by these certifications. Like EigenTrust, the Advogato metric is quite attack resistant. By identifying individual nodes as bad and finding any nodes that certify the bad nodes, the metric cuts out an unreliable portion of the network. Calculations are based primarily on the good nodes so the network as a whole remains secure. Because of its use of groups to determine who can post messages, Advogato is called a *group trust metric*. It is also a global trust algorithm because the same seeds are used to make calculations for every user. A common alteration to Advogato sets the user as the single seed, thus converting it to a local metric with personalized calculations.

Ziegler and Lausen [2004a] propose a trust algorithm called Appleseed. Like Advogato, it is a group trust metric. However, instead of using maximum flow, the basic intuition is motivated by spreading activation strategies. Like EigenTrust, Appleseed is based on finding the principal eigenvector. It is a local trust metric, and, given a network and a starting node, it returns a ranking of all the nodes in the network.

Richardson et al. [2003] use social networks with trust to calculate the belief a user may have in a statement. This is done by finding paths (either through enumeration or probabilistic methods) from the source to any node which represents an opinion of the statement in question, concatenating trust values along the paths to come up with the recommended belief in the statement for that path and aggregating those values to come up with a final trust value for the statement. Current social network systems on the Web, however, primarily focus on trust values between one user and another, and thus their aggregation function would require some modification to be applied in these systems. Their article intentionally does not define a specific concatenation function for calculating trust between individuals, opting instead to present a general framework as their main result. To test their algorithms, they do choose a concatenation function (multiplication) and show accuracy results using the Epinions network. Grishchenko [2004] discusses some issues and potential applications related to the work from Richardson and others.

One problem that arises in algorithms that are based on finding the principal eigenvector, like Kamvar [2003], Zeigler and Lausen [2004], and Richardson et al. [2003], is that trust must first be normalized to work within the matrix. This means that the normalized trust value from a person who has made many trust ratings will be lower than if only one or two people had been rated. However, socially, trust is not a finite resource; it is possible to have very high trust for a large number of people, and that trust is not any weaker than the trust held by a person who only trusts one or two others.

Ultimately, we want to take the trust calculations made by the algorithms that we present in Section 4 and incorporate them into applications. Essentially, the inferred trust values are recommendations to one user about how much to trust another user. There is some research into more traditional recommender systems that suggest our approach to using trust calculations to make recommendations in software systems will be useful. Specifically, there is evidence to support that users will prefer systems with recommendations that rely on social networks and trust relationships over similarity measures commonly used for making recommendations. Research has shown that people prefer recommendations from friends to those made by recommender systems and that users prefer recommendations from systems they trust [Swearingen et al. 2001]. By producing recommendations through the use of trust in social networks, both of these user preferences are addressed. Recommendations come through a network of friends and are based on the explicit trust expressed by the user.

The algorithms here are generating a recommended trust rating for an unknown person, based on information from others in the network. In one sense, this is very similar to what collaborative filtering algorithms do. Collaborative filtering techniques generally compute a similarity measure between the user and others in the system, and then use that as a weight on the opinions of other users to develop a recommendation. Herlocker et al. [2004] present an excellent overview of the goals, datasets, and algorithms of collaborative filtering systems.

Table I. Average Error (User Rating | Recommended Rating), Measured in Number of Stars of Different Recommendation Methods

Minimum difference between user rating and average rating		Trust-Based Recommendation	ACF-Based Recommendation	Simple Average
	0	0.56	0.534	0.48
	0.5	0.705	0.705	0.79
	1	0.75	1.1	1.19
	1.5	0.82	1.71	1.76
	2	1	2.18	2.19

In the case of social networks, however, data is very sparse, presenting a challenge to collaborative filtering techniques Data sparseness is a known problem for collaborative filtering algorithms [Sarwar 2001]. This is supported by empirical evidence in Massa and Avesani [2004] and Massa and Bhattacharjee [2004]. These works address the use of trust within systems where the set of commonly rated items between users is sparse. This situation leads to a breakdown in correlation-based recommender system algorithms, and their work explores how incorporating even simple binary trust relationships can increase the coverage and thus the number of recommendations that can be made.

In this article, we do not extend our work into developing an integrated recommender system but rather propose new algorithms for computing trust that could then be incorporated in ways described in the existing research. However, our related work shows that, in the context of predictive movie recommendations, a trust-based approach can significantly outperform collaborative filtering algorithms in certain cases. Our hypothesis in that application is that trust recommendations will be useful when users have an opinion that is different from the average, that is, people rely on their trusted friends when they disagree with the general public. We measure error as the absolute difference between the user's rating of a movie and the recommendation predicted by different methods. Our results show that, as the minimum difference between the user's rating and the average rating is greater than 1 star, the recommendation produced using the trust-based method is significantly more accurate than the recommendations from both an automated collaborative filtering (ACF) method and from the simple average for $p < 0.01$ (see Table I). More details can be found in Golbeck and Hendler [2006].

Work in the Public Key Infrastructure (PKI) uses trust in a similar way to social networks. Mapping a name to a public key or, conversely, to finding the public key of a particular user is an important task for executing secure transactions. When there is no centralized authority to map keys and names, this sort of authentication can be done by combining information from a path of authorities. These chains can suffer if any of the intermediate authorities have poor information. Trust values between authorities can be combined over paths to determine confidence in the authority at the end point.

The inputs and outputs of these metrics vary, and some do not translate well to use in social networks. For example, Maurer [1996] makes calculations using several features in addition to trust. Confidence ratings are combined with explicit statements of trust and authenticity measures to infer authenticity

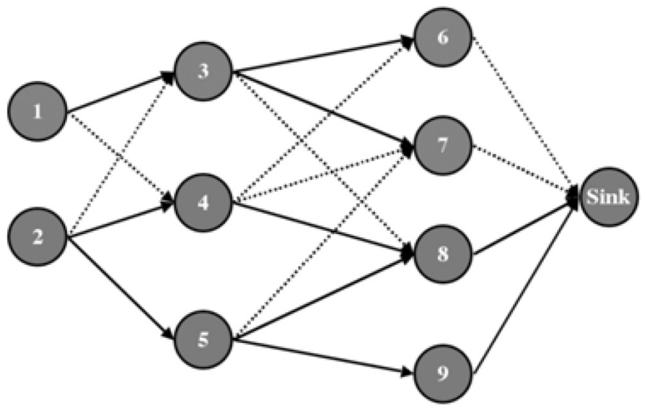


Fig. 1. Nodes consider ratings from the people they trust highly (indicated by solid edges). Nodes with low trust ratings (indicated with dashed edges) are only considered when they are a direct rating of the sink, but are not used in finding paths to the sink. The ratings made by trusted nodes that directly rated the sink are used in coming up with a recommendation about how much the source should trust the sink.

information. On the other hand, some metrics can be directly applied to trust in social networks. For example, Beth et al.'s metric [1994] is more directly applicable to Web-based social networks. However, these algorithms have poor scalability properties, owing to exponential time complexity [Reiter and Stubblebine 1997a]. When applied to the typical Web-based social networks, that have hundreds of thousands—or millions—of members, the complexity is prohibitive.

2.5 Trust Inference Example

To tie together all of the concepts of trust introduced in this section, Figure 1 illustrates the general process of making a trust inference in a social network.

Figure 1 depicts a sample social network. The solid lines indicate relationships with trust, and the dashed lines indicate no trust. The direction of the arrows indicates the direction of the trust (e.g., node 1 has given trust to node 3 but not the reverse). The node for which we are determining a trust rating, called the *sink*, is trusted by two nodes (8 and 9) and not trusted by two nodes (6 and 7). If a trust rating for the sink were calculated by composing all of the direct ratings of the sink, every node would get the same recommendation. However, if we take into account the information that we know about the structure of the network from the perspective of each node, a much more informative recommendation can be made. Node 1 can accept information only from it's trusted neighbors. In the end, only the trust ratings given by nodes 6 and 7 will propagate back to node 1. Both 6 and 7 do not trust the sink and only their opinion will be passed back to node 3 and then to node 1 who will calculate that the sink is not to be trusted. Similarly, node 2 also only considers trusted paths. At the end of these paths, nodes 8 and 9 both have directly rated the sink to be trustworthy. Their values are passed back along the network paths through nodes 4 and 5 to node 2. Node 2 will conclude that the sink is to be trusted. Thus, if perspective is taken into account, node 1 and node 2 can each receive relevant and accurate information about how much to trust the sink even though their opinions are diametrically opposed and the information in the network is mixed.

3. GENERATING SOCIAL NETWORKS

Naturally occurring networks take a long time to gain a large number of users, and the topological properties are fixed. To experiment with making trust inferences on networks with various properties, it is necessary to be able to automatically generate network models.

3.1 Building Networks with Correct Topology

It has been widely documented that social networks have the properties of small world networks [Watts 1999]. The properties of graph structures that define a small world network are clustering coefficient and average path length. The clustering coefficient (indicated by the variable γ) is the property of clustering in neighborhoods. Given a node n, γ is the fraction of edges between neighbors of n that actually exist compared to the total number of possible edges. Small world graphs have a high clustering coefficient. The average shortest path length between nodes (indicated with variable L) grows logarithmically with the size of the graph in small world networks.

The one difference between the networks in this research and traditional complex systems is that our network has directed edges. Although the L and γ values are usually calculated with undirected edges, they can easily be calculated with directed edges. The shortest path is calculated by following edges and respecting their direction. Clustering coefficient (γ) is calculated with twice as many possible edges since any pair of nodes has two possible directed edges that can connect them.

The work by Watts and Strogatz [1998] showed that graphs with small world properties can be generated by randomly rewiring a small number of nodes in a regular graph like a lattice. The variable p indicates what percentage of edges should be randomly selected, removed, and randomly reconnected. As p increases, the average path length drops off quickly. The average γ, on the other hand, remains high until p gets too large. Creating a lattice and choosing a p that produces a graph with a high clustering coefficient and low average path length will produce a small world graph.

Figure 2 shows how the average shortest path length and the clustering coefficient change as the p value changes. We can see here that the clustering coefficient remains relatively high until $p = 0.1$, at which point there is a sharp drop. The average path length was normalized to a range of 0–1 so it could be plotted with the clustering coefficient. In the figure, the average shortest path length decreases until it begins to level off when $p = 0.1$. It is at this point, $p = 0.1$, where the clustering coefficient is still high and the average path length has come down that the network has the topology of a small world network. While the results depicted in Figure 2 show an optimal p value of 0.1, this value will vary from network to network as the number of nodes and average

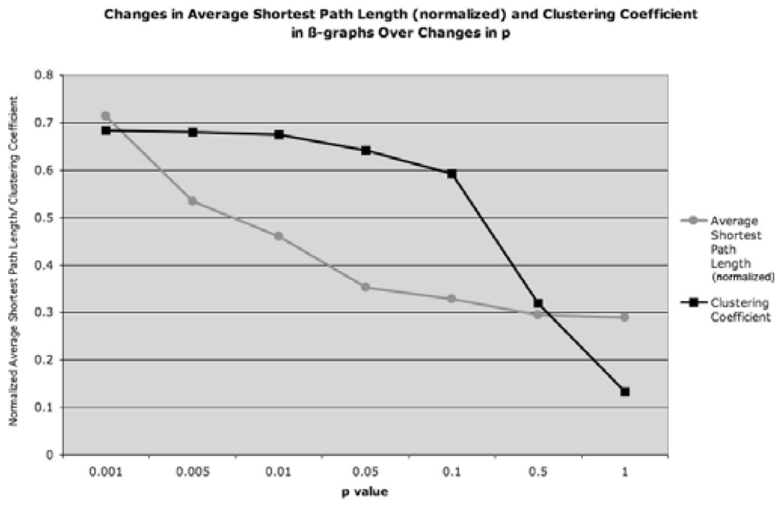


Fig. 2. The change in the normalized average shortest path length and the clustering coefficient as p changes. The pictured results are for graphs with 800 nodes and an average degree of 8.

degree changes. We analyzed the structure of our experimental networks to ensure that we used the appropriate value of p.

This rewiring model, called the ß-model [Watts 1999], has been shown to successfully emulate the structure of several common social networks, including the co-authorship graph and co-actor graph [Davis et al. 2003; Foster et al. 1963; Newman 2001; Watts 1999]. We used the ß-model to create graphs for use in the analysis of the accuracy of algorithms that are presented in Section 4. The p value varied depending on the size and average degree of our graphs. We verified for each graph size and average degree that the p value produced graphs with L and γ values consistent with small world graphs.

3.2 Adding Trust Ratings to Graphs

The edges in the generated graphs represent connections between individuals. To generate a trust network, those edges must be augmented with values representing the trust relationship between individuals. This section describes the process of adding trust ratings into a generated graph. The method used for producing these trust values will also form the foundation for analyzing the trust inference algorithms presented in Section 4.

One node in the network is randomly chosen as the source. We then give each node a true rating. This rating determines whether the node is trustworthy (good) or not trustworthy (bad). The number of good and bad nodes are assigned at a predetermined ratio, and have two properties. First, this true value is treated as the source's opinion of the node, as though an all-knowing oracle could tell whether or not the source would consider this node was good or bad if there were an established relationship. The second property determined by the good/bad rating is the node's behavior. Good nodes agree with the source with a certain probability, while bad nodes always vote incorrectly; the bad nodes will say every good node is bad and every bad node is good.

It is important to note here that we are not studying the behavior of the nodes in the network to try to identify good or bad nodes. Clearly, with bad nodes always voting opposite the source, they would be relatively easy to track down. The bad nodes represent attackers—nodes that may be incorrectly called trustworthy and can then corrupt the system.

If a bad node, B, is accidentally marked as good in the network, all the information that comes from that node will be incorrect. Any good nodes that B has rated will be marked as bad, and thus excluded from further consideration. Any bad nodes that B has rated will be marked as good, and incorrect information will flow up from them as well. Recall that the ratings discussed here are all with respect to the opinion of the source, not of the individual node. That is to say, bad nodes always assign ratings that disagree with the source. As a result, there is no complex behavior that comes back from the bad nodes. They all consistently disagree with the source, so once a bad node is incorrectly allowed into the network, it will only accept information from other bad nodes, which will always be wrong from the perspective of the source. The behavior of good nodes cannot change this. A good node that correctly rates a bad node will prevent its ratings from ever being considered. A good node that incorrectly rates a bad node will allow that bad information into consideration. A change in the behavior of bad nodes, where they might introduce correct information part of the time, would only increase the accuracy of the results because there would be a chance for some correct information to get through. Thus, the always wrong behavior of the bad nodes represents a worst case for our analysis. While the behavior of bad nodes in these simulations—always assigning incorrect values—is worse than what we would expect in a real network, it is a useful analytical tool for seeing how effective an attack on the network could be.

Once each node has been given its true value, the trust ratings on the edges are assigned. Bad nodes rate every neighbor opposite its true value. Good nodes rate each neighbor correctly with a certain probability. For example, if we specify that the good nodes are accurate 70% of the time, each neighbor is rated independently with a rating corresponding to the true value with probability 0.7

4. MAKING TRUST INFERENCES

In this section, we present two algorithms for inferring trust ratings in the generated social networks described in Section 3. Given a social network represented as a directed graph (know to have a small world topology) with binary labels that represent the trust between nodes in the direction of the edge, a source node, a sink node, and a vector indicating whether each node is truly good or bad (represented as an n X 1 binary matrix), we will estimate how much the source node should trust the sink based on the paths and trust values connecting them in the network.

Since we have the true good or bad value for each node, we are able to calculate the recommended trust rating from source to sink and compare that

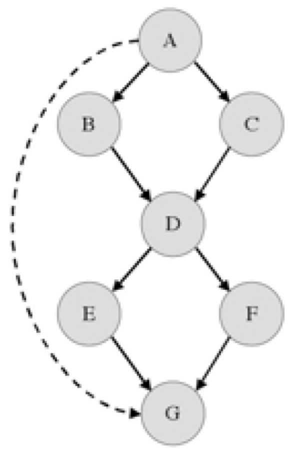


Fig. 3. An illustration of how nodes are used in the inference from node A to node G.

rating with the true value for the sink. This difference is a simple and natural measure of the accuracy of the recommendations made by an algorithm. We show that both theoretically and in simulation the algorithms presented here infer trust ratings that are highly accurate according to the beliefs of the source.

We will show two variations of the algorithm. The first uses {0,1} values exclusively—users assign {0,1} ratings, and, at every step of the algorithm, a node returns {0,1} values. The second algorithm also begins with users assigning {0,1} trust ratings, but as the calculations propagate through the network, nodes return average values that are continuous from [0,1]. Only in the final step is the value rounded to return either 0 or 1. A careful analysis of these results suggests that similar results can be expected with any scale, but the clarity analysis available with binary ratings allows us to provide insight into why other systems may work as well.

4.1 A Rounding Algorithm

In this algorithm, the source polls each of the neighbors to which it has given a positive reputation rating. Neighbors with zero (no trust) ratings are ignored, since their reputation means that they give unreliable information. Each of the source's trusted neighbors will return their rating for the sink. The source will then average these ratings and round the final value. This rounded value is the inferred reputation rating from source to sink.

Each of the source's neighbors will use this same process to come up with their reputation ratings for the sink. If there is a direct edge connecting them to the sink, the value of that edge is used; otherwise, the value is inferred. As shown in Figure 3, if a node is encountered in two paths from source to sink, it is considered in each path. node B and C will both return ratings calculated through D. When a reputation rating from D is first requested, D will average the ratings from E and F. The value is then cached at D so that the second time D's reputation rating is needed, no calculations are necessary.

While branches of the search tree can merge as in Figure 2, there are no cycles in the tree. For example, if in Figure 2, node E had rated node B as a neighbor, it would not consider node B in its computation of a trust value for the sink since node B is higher up in the search tree. To enforce this, each node must be marked to indicate its status in the search.

This process of finding a rating for the sink is essentially a modified breadth-first search (BFS). The search moves from the source along paths to find the sink. The only change that is made is that instead of stopping the search when the sink is found, the search continues until all paths terminate. Note, however, that we are not finding all paths to the sink, which would be exponential. As is the case with BFS, once a node has been searched, it is not re-searched again.

4.2 Nonrounding Algorithm

We altered the algorithm just presented by removing the rounding performed by each node before it returns a value. The only rounding is made in one final step, added to the end of the algorithm, where the original source rounds the average of the values returned by its neighbors, so the final inferred value is 0 or 1. Ratings are still assigned as 1 or 0 values (trusted or not trusted). With the algorithmic change, however, intermediate nodes on the path from source to sink return values in the range of {0,1} instead of returning rounded {0,1} values. Accuracy was determined by taking the difference between the rounded final inferred value and the true value.

4.3 Analysis of the Algorithms

There are two variables in the network, namely, percentage of good nodes, g, and the accuracy, p_a, of good nodes. The overall accuracy of the direct ratings initially assigned in the network is given by $g * p_a$.We call this the *initial accuracy* in the network and represent this with the variable a. Note that the initial accuracy is a measure of how frequently the direct ratings assigned to each node in the network agree with the true trust value that the source holds for each node. It is not a measure of how accurate a person's ratings are with respect to their own beliefs. The initial accuracy gives us a baseline to which the inferences can be compared. If the inferences offer no improvement, their accuracy should be approximately the same as the initial accuracy. However, if the accuracy of the inferences is significantly higher than the initial accuracy, it means that the algorithm filters out less accurate information and produces inferences that exceed the accuracy that would be expected from the base accuracy in the network.

By design in these algorithms, a node will make a correct inference if the majority of its neighbors return the correct rating for the sink. Since the bad nodes are always incorrect, the accuracy of the good nodes must compensate to obtain a correct inference from a majority vote. Thus, to obtain a correct inference, the initial accuracy ($g * p_a$.) must be at least 0.5. Let $a = g * p_a$.

A node will either give a correct or incorrect rating for the sink. This corresponds to a Bernoulli trial. Thus, for a graph with n nodes, the probability that

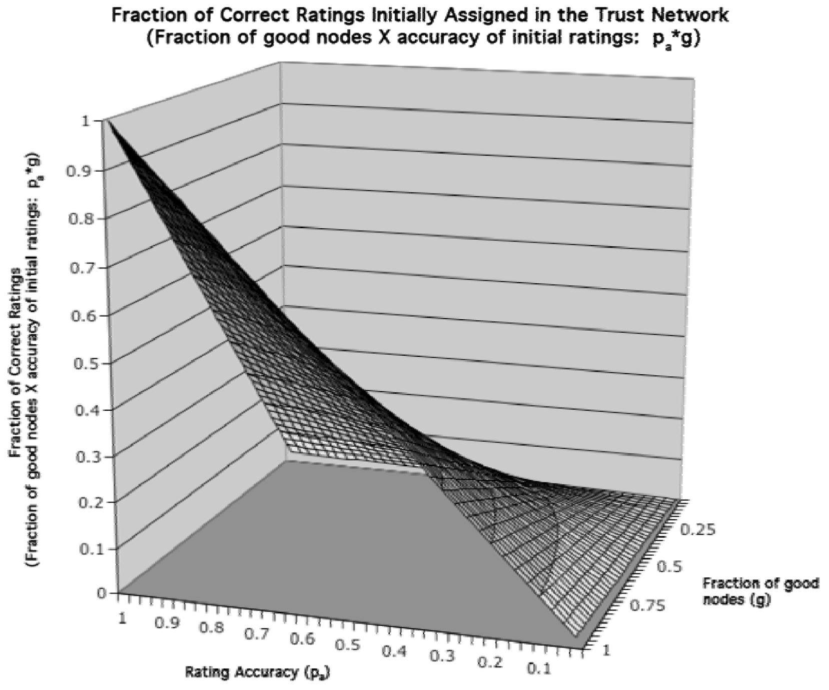


Fig. 4. A map of how the initial accuracy in the system changes with g and p_a.

exactly i nodes correctly rate the sink is given by a binomial distribution. This is given in Equation (1).

$$\binom{n}{i} a^i (1-a)^{n-i} \tag{1}$$

We will achieve a correct inference if at least half of the nodes give a correct answer. The probability that at least half the nodes correctly rate the sink is the sum of the binomial coefficients for all $i \geq$ n/2.

Since a is the initial accuracy of the nodes, the proportion of nodes that correctly rate the sink should have a mean of a. The binomial distribution can be approximated by a normal distribution with a mean centered at a. By Central Limit Theorem, as n increases, the binomial distribution gets closer and closer to a normal distribution. That is, the binomial probability of any event approaches the normal probability of the same event. As n increases, the standard deviation of the normal distribution decreases, making a narrower curve, and thus the proportion of nodes that make correct rating of the sink is closer to the mean a. Thus, for $a > 0.5$, the probability that the mean is greater than 0.5 (and, correspondingly, more than half the nodes correctly rate the sink, thus producing a correct inference) approaches 1. Similarly, for $a < 0.5$, the probability of a correct inference goes to 0.

Thus, if the initial accuracy is > 0.5, the probability that the recommendation is correct goes to 1. This is a critical point. As long as $g * p_a$ is greater than

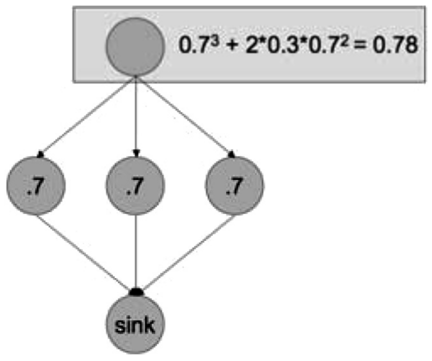


Fig. 5. This figure shows a simple network and demonstrates the increasing probability of accuracy. Beginning with a uniform accuracy of 0.7, the probability of properly classifying the sink increases as we move up the search tree from sink to source.

half, we can expect to have a highly accurate inference (see Figure 4).

$$\lim_{n\to\infty} \sum_{i=\lceil \frac{n}{2} \rceil}^{n} \binom{n}{i} a^i (1-a)^{n-i} \to 1 \quad \text{for} \quad a > 0.5 \tag{2}$$

In Equation (1), there is one case that must be considered where n is even, exactly half of the neighbors call the sink trustworthy, and the other half say it is not trustworthy. In that case, we are rounding up, to call the sink trustworthy. However, there is an error factor that must be subtracted, since in some cases the sink will not be trustworthy. The probability that this rounding is wrong is (1-a). We can determine the overall reduction in accuracy by multiplying (1−a) by the probability of an even split. This total reduction in accuracy is

$$\binom{n}{n/2} a^{\frac{n}{2}} (1-a)^{\frac{n}{2}+1}. \tag{3}$$

For $a \geq 0.5$, this value decreases as n increases. Since our algorithms are only effective for $a > 0.5$, and the probability of error decreases (Equation (3)) while the probability of accuracy increases (as shown in Equation (2)), this slight reduction in accuracy will not affect the overall quality of the results.

This analysis describes one step of rounding. With the nonrounding algorithm, where only the final inferred value is rounded, this analysis applies. In the rounding algorithm, where the average trust value is rounded at each step, the accuracy increases at each step. As the algorithm moves up from the immediate neighbors of the sink toward the source, a will vary from node to node, but it will increase at each level. Figure 5 illustrates this point where the network starts with all good nodes, accurate in 70% of their classifications. After moving up two levels from the sink, the accuracy of the inference will be approximately 78%.

This analysis suggests that the rounding algorithm will outperform the nonrounding algorithm. The nonrounding algorithm is an important intermediate step between a system with binary ratings and a system with continuous values. Since continuous values are used in the intermediate steps between source and sink, this algorithm nearly replicates what would be used when users assign values in a broader range. The analysis here not only shows that the final step of rounding gives good results, but also that accuracy is not lost when the internal rounding is eliminated. Future work will address the shift to a system with continuous values and how a slight variation on the nonrounding algorithm can be effective in such a network.

This analysis also shows that the algorithms will be more effective on larger networks with higher average degrees. Equation (2) shows that the probability of a correct inference increases at a given node as the number of neighbors increase. The propagation of this increased accuracy up through the network is more effective when the path length is longer. Even the smallest Web-based social networks have several hundred users, and most have many thousands of users. These numbers are large enough to clearly see the benefits of our algorithms (this is explored further in Section 4.4), but it should be noted that, on very small networks or in networks with a low average degree, the benefits of the algorithms illustrated in this analysis will not be as strong.

At its core, the algorithm is a modified version of breadth-first search (BFS). The search is carried out fully, and then each step of the search is traced back to the source as the values are averaged and rounded. Each step of the BFS is essentially performed twice: once in the search for paths to the sink, and once as the trust values propagate back up to the source. Since this is a constant multiplier of BFS, the complexity of this algorithm is the same as the complexity of BFS: O (V+E).

4.4 Simulations

When inferring the trust rating for a node, the accuracy of the inference is determined by comparing the inferred value to the true value. In this section, we present a series of simulations to compare with the theoretical analysis presented in Section 4.3. We show that, for both the rounding and nonrounding algorithms, the inferred trust rating is more accurate than the accuracy of the initial trust ratings in the network for $p_a * g > 0.5$.

In these simulations, both g and p_a were independently varied. Each variable was tested on 400 different values in the range {0,1} taken in increments of 0.025. When combined, this resulted in 1,600 (g, p_a) pairs. For each (g, p_a) pair, we generated 1,000 small-world graphs using the ß-model previously described. In those graphs, a source and sink were randomly-chosen. The trust inference from source to sink made on each graph was checked against the true value of the sink. This was performed with ten randomly-chosen pairs in each graph, for a total of 10,000 pairs for each (g, p_a). This experiment was repeated for graphs with 400, 800, and 1600 nodes with several values for the average degree for each. These variations in graph size and average degree affect the results of the simulations, but we are interested only in the general behavior with respect to the initial accuracy. For all of our simulations, the same pattern was observed: accuracy of the inferred values was higher than the initial accuracy for $a > 0.5$.

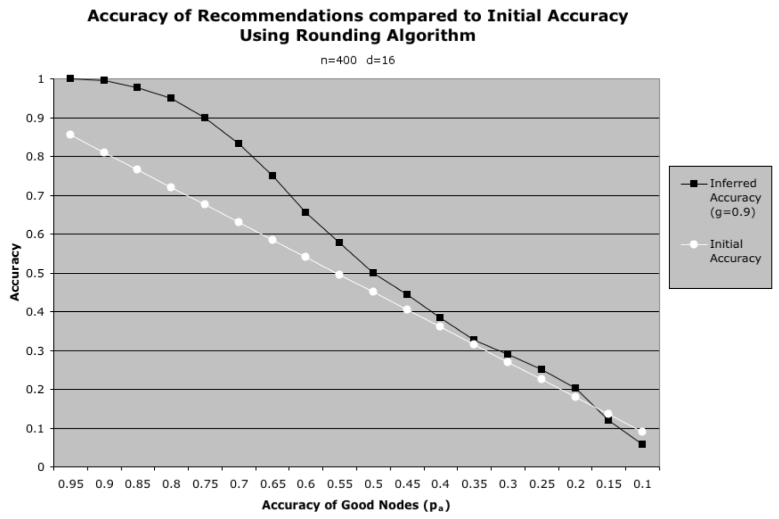


Fig. 6. A comparison of the initial accuracy of trust ratings with the accuracy of inferred ratings using the rounding algorithm for $n = 400$, $d = 16$, $g = 0.9$, and a variable p_a.

Beginning with the rounding algorithm, experiments showed that the accuracy of the inferred rating was significantly higher than the initial accuracy in the network. Figure 6 shows data for the inferred accuracy using the rounding algorithm on a set of graphs with 400 nodes and an average degree of 16. While the initial accuracy of ratings decreases linearly, the accuracy of the inferred ratings remains higher.

Overall, in the simulations, the accuracy of a recommendation remains high relative to the initial accuracy until $p_a * g$ nears 0.5. This is shown in Figure 7. The graph shown in Figure 7 has 400 nodes with an average degree of 16. Changes in both the size of the graph and the average degree will affect the search from source to sink. An increase in the number of nodes or a decrease in the average degree will lengthen the average path length. While this will increase the number of bad nodes that are encountered, it does not affect the accuracy of the inference because the percentage of bad nodes remains the same. Because the algorithm computes a result based on the percentage of a node's neighbors that rated the sink good or bad, the increased number of bad nodes encountered does not negatively impact the accuracy of the results.

The nonrounding algorithm produced results inline with the theoretical analysis. Even without the intermediate rounding, the inferred values were more accurate than the initial accuracy in the system. The results are less dramatic than for the rounding algorithm, but Figures 8 and 9 show that the increased accuracy is still present.

To directly compare the two algorithms, Figure 10 shows the accuracy of both the rounding and nonrounding algorithms together for $g = 1$ and $g = 0.9$ with $p_a > 0.5$. The results in Figure 10 are drawn from simulations on 400 node networks with and average degree of sixteen. The performance follows a similar pattern for both algorithms and both g values. As the theoretical results would

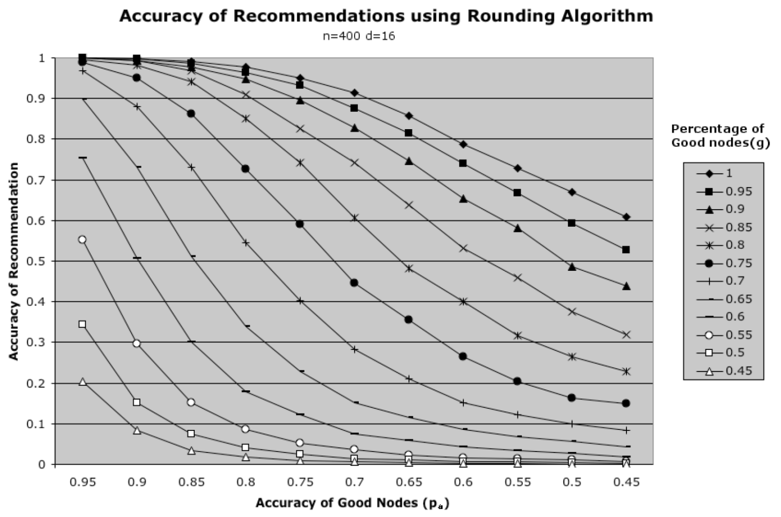


Fig. 7. The accuracy of inferred ratings are shown for various initial percentages of good nodes.

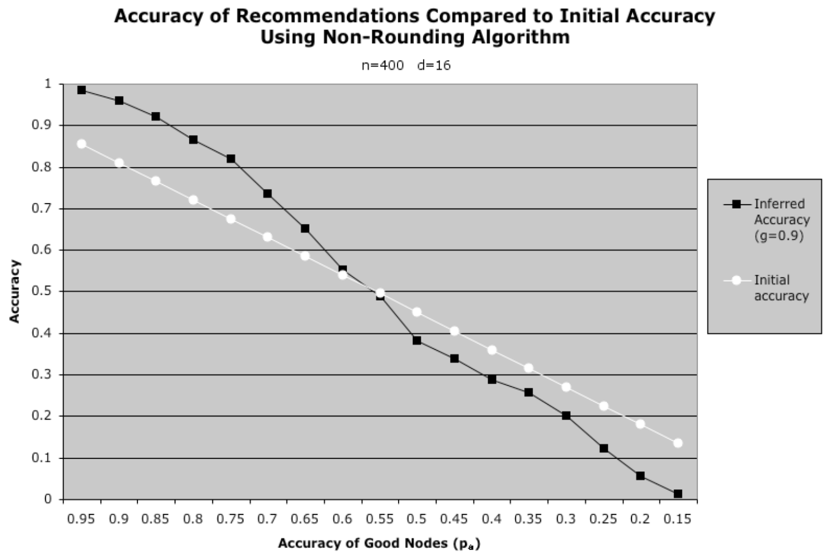


Fig. 8. This figure shows that for $a > 0.5$ the inferred accuracy using the nonrounding algorithm is higher than the accuracy of the initial ratings. This is the same effect as shown in Figure 5 for the rounding algorithm.

indicate, the rounding algorithm outperforms the nonrounding algorithm for both g values because the rounding at each step removes more error than is rounded out in the final step of the nonrounding algorithm.

With algorithms that are theoretically and experimentally verified to produce results much more accurate than the initial accuracy in the system, the next step is to integrate those algorithms into applications. In the end, the purpose of developing algorithms that can calculate trust recommendations from social

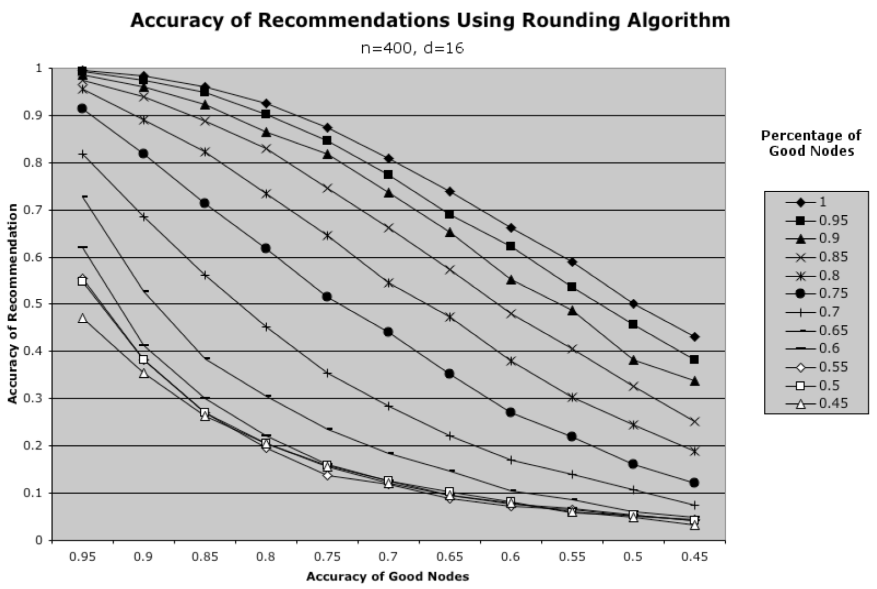


Fig. 9. This figure shows the accuracy of inferred trust values using the nonrounding algorithm for a range of g and p_a values. Though less pronounced than the results for the rounding algorithm shown in Figure 6, we can see that the results follow a similar pattern of remaining higher than the a value for $a > 0.5$ and lower for $a < 0.5$.

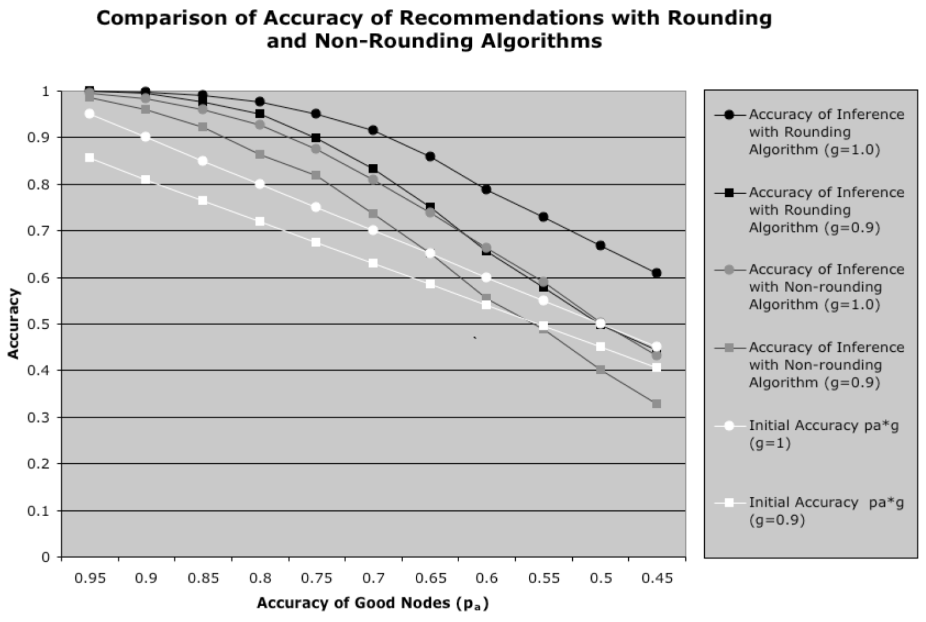


Fig. 10. Comparing the accuracy of trust inferences made with the rounding and nonrounding algorithms.

networks is to use them to create applications that are socially intelligent. By incorporating users' social preferences into software, features can be tailored to their current connections and preferences. The next section will present one application designed to illustrate the potential of this technique.

5. TRUST NETWORKS IN APPLICATIONS: TRUSTMAIL

Consider the case of two research groups working on a project together. The professors that head each group know one another, and each of the professors know the students in their own group. However, neither is familiar with the students from the other group. If, as part of the project, a student sends an email to the other group's professor, how will the professor know that the message is from someone worth paying attention to? Since the name is unfamiliar, the message is indistinguishable from other mail in the inbox. This scenario is exactly the type of situation that TrustMail strives to improve upon. The professors need only to rate their own students and the other professor. Since the trust algorithm looks for paths in the graph, there will be a path from the professor of one research group to the students of the other group through the professor-to-professor link. Thus, even though the student from one group and professor from another have never met or corresponded, the student gets a high rating from the network path. If it turns out that one of the students is sending junk type messages, but the network is producing a high rating, the professor who receives the messages can simply add a low direct rating for the student, downgrading the trust. The new rating assigned by the professor will appear next to messages from that student in the professor's inbox. Furthermore, the value will be used to infer the trustworthiness of the student for any user who has a path through the professor to the student. Thus, the professor's rating is not just blacklisting the student; it is used in computations in the network to compute trust inferences for other users.

5.1 Email Filtering with Trust Networks

The fact that spam has become such a ubiquitous problem with email has lead to much research and development of algorithms that try to identify spam and prevent it from even reaching the user's mailbox. Many of these techniques have been highly successful, catching and filtering the vast majority of Spam messages that a person receives.

Though work still continues to refine these methods, some focus has shifted to new mechanisms for blocking unwanted messages and highlighting good or valid messages. *Whitelist filters* are one of these methods. In these systems, users create a list of approved addresses from which they will accept messages. Any whitelisted messages are delivered to the user's inbox, and all others are filtered into a low-priority folder. These systems do not claim that all of the filtered messages will be spam, but rather that a whitelist makes the inbox more usable by only showing messages from known, valid senders. Though whitelists are nearly 100% effective at blocking unwanted email, there are two major problems with them. First, there is an extra burden placed on the user to maintain a whitelist, and second, some valid emails will almost certainly be filtered into the low-priority mailbox. If that box contains a lot of spam, the valid messages will be especially difficult to find.

Other approaches have used social networks for message filtering. Boykin and Roychowdhury [2004] create a social network from the messages that a user has received. Using the structural properties of social networks, particularly the propensity for local clustering, messages are labeled as spam, valid, or unknown, based on clustering thresholds. Their method is able to classify about 50% of a user's email into the spam or valid categories, leaving 50% to be filtered by other techniques.

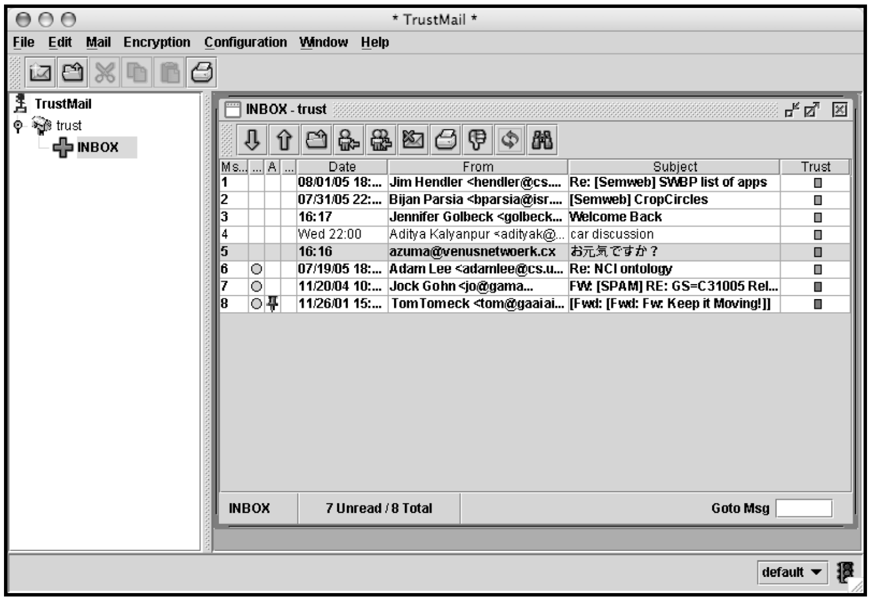
The LOAF Project is another social email filter. Using Bloom Filters [Bloom 1970], the user's address book is appended to the end of emails. The recipient can then check to see if they share any common friends in that list or if there has been previous direct communication. The limits of this technique are both in specificity and depth. First, LOAF only connects people who share common acquaintances with the recipient, limiting the reach of the technique to a depth of 2 in the social network. Second, there is no notion of good or bad communication. For example, a user could send a couple of emails to try to stop spam and if the recipient had also emailed to stop that spam, LOAF would show them sharing a common connection.

Our approach takes some of the basic premises of whitelisting and social network-based filtering and extends them. Unlike Boykin and Roychowdhury's technique [2004] that builds a social network from the user's own email folders, our technique uses a Web-based social network as described in previous sections. Trust ratings—direct or computed—are used to score messages based on the trust from the recipient to the sender.

The scoring system preserves the whitelist benefit of making the inbox more usable by making good messages prominent via high scores. The added benefit is that scores will also appear next to messages from people with whom the user has never had contact before. This is because, if they are connected through a mutual acquaintance in the trust network, we can infer a rating. This diminishes some of the problems with whitelists. There is still a burden for users to create an initial set of trust ratings. However, the properties of the Web-based trust network connect users to a much larger population for whom trust values can be calculated. Since scores are available for so many more people than the user explicitly rates, fewer valid messages will be filtered into a low-priority mail folder, lessening the burden on users to find these messages.

It is important to note that the goal of this scoring system is not to give low ratings to bad senders or spam. The main premise is to provide higher ratings to non-spam senders, so that users are able to identify messages of interest that they might not otherwise have recognized.

Trust scores are not intended to be a stand-alone solution to spam. We envision that the mail-scoring technique will be used in conjunction with a variety of other antispam mechanisms. There are also some spam issues that particularly effect this algorithm. Forged email headers, where the "From" line of a message is altered to look like a valid address, is one such issue. Because our technique is designed to identify good messages that make it past spam filters, we do not address the case where a person has a virus sending messages from



Ms...	...	A	...	Date	From	Subject	Trust
1				08/01/05 18:...	Jim Hendler <hendler@cs....	Re: [Semweb] SWBP list of apps	▨
2				07/31/05 22:...	Bijan Parsia <bparsia@isr....	[Semweb] CropCircles	▨
3				16:17	Jennifer Golbeck <golbeck...	Welcome Back	▨
4				Wed 22:00	Aditya Kalyanpur <adityak@...	car discussion	▨
5				16:16	azuma@venusnetwoerk.cx	お元気ですか？	▨
6	○			07/19/05 18:...	Adam Lee <adamlee@cs.u...	Re: NCI ontology	▨
7	○			11/20/04 10:...	Jock Gohn <jo@gama...	FW: [SPAM] RE: GS=C31005 Rel...	▨
8	○			11/26/01 15:...	TomTomeck <tom@gaaiai...	[Fwd: [Fwd: Fw: Keep it Moving!]]	▨

Fig. 11. The TrustMail Interface. The Trust column to the far right contains an indicator of the trustworthiness of the person who sent the message. Green indicates trusted senders, red indicates untrusted senders, and grey is used when no path to the sender can be found. The trust values of the senders is computed using the trust inference algorithms presented in this article.

their account. Our work is not designed to address these problems; rather, the trust features are designed to work in connection with more traditional spam filters that filter content and can detect spam and spoofed email headers.

Essentially, trust is integrated into the email client to serve as a tool for ranking and filtering messages according to their presumed importance. This is not the first technique developed for this task. Maxims [Lasharki et al. 1994] is an agent integrated with an email client that learns how to filter, delete, and archive messages based on user actions. While this work takes a social network-based approach to the problem of message filtering instead of an agent-based approach, the two methods are not contradictory. They could, in fact, be integrated into a complementary system to ease email overload.

5.2 The TrustMail Prototype

TrustMail is a prototype email client that adds trust ratings to each message in the folder view. This allows a user to see their trust rating for each individual and sort messages accordingly. This is essentially a message-scoring system. The benefit to users is that relevant and potentially important messages can be highlighted even if the user does not know the sender. The trustworthiness of the sender is computed with the trust inference algorithms presented in Section 4, using the user's own perspective on the trust network, and thus scores will be personalized to and under the control of each user.

Figure 11 shows a screen shot of a user's inbox in TrustMail. The program extracts the sender of each email, and then uses the trust algorithms to compute whether that person is trusted or not trusted. The Trust column to the far right contains an indicator of the trustworthiness of the person who sent the message. Green indicates trusted senders, red indicates untrusted senders, and grey is used when no path to the sender can be found. The trust values of the senders is computed using the trust inference algorithms represented here.

Because the network has a relatively small number of users, we have not thoroughly analyzed at what level TrustMail may affect a user's mailboxes. However, a small preliminary investigation has shown that, for the strongly connected individuals in the trust network, the client is catching and labeling many emails. We were able to obtain basic information from the inboxes of 10 core users in the trust network. Approximately 40% of the emails received trust values, which means the senders were rated in the network. Nearly all the messages with ratings were not spam. Of the emails with trust values, a majority (about 60%) were displayed with inferred ratings as opposed to direct ratings. Due to privacy concerns, we were not able to access the actual mail folders of these users, so information about the distribution of senders and messages was not available to us. Future work will expand this study to a large group of users with a more detailed analysis of which emails and senders are rated, but based on this anecdotal evidence, we expect that users who actively participate in social networks may see significant benefits from these techniques.

The trust information alongside messages is useful, not only for indicating trust or no trust, but also because they basically replicate the way trust relationships work in social settings. For example, it would be sensible and polite for a student emailing a professor she has never met to start the email with some indication of the relationships between the student and the two professors, for example, "My advisor has collaborated with you on this topic in the past and she suggested I contact you." The professor may chose to verify the validity of this statement by contacting the student's advisor or finding information that verifies the claim. These ratings are developed by consulting the social network and ratings within it and serve as evidence of mutual, trusted acquaintances.

TrustMail replaces the process of searching for information about a recipient by utilizing the data in Web-based social networks. Because calculations are made from the perspective of the email recipient, high ratings will necessarily have come through people the recipient trusts. This allows the trust network-based system to complement spam filters by identifying good messages that might otherwise be indistinguishable from unwanted messages and carrying the validation of a rating drawn from the users own network of trusted acquaintances.

Techniques that build social networks from messages that the user has sent or received can identify whether or not a message has come from someone in the network. However, because they are built only from the user's local mail folders, no information is available about people that the user has not previously seen. If the user's personal network is connected to a larger social network with information from many other users, much more data is available. Previously unseen senders can be identified as part of the network.

To understand what benefit TrustMail might offer to users, it is important to understand what percentage of messages we can expect to have ratings. The next section uses a real email corpus to gain some insight into this question.

5.3 The Enron Email Corpus

One common question about the TrustMail system is how strongly the trust network might impact the users mail folders. If user behavior is such that only a small fraction of messages would receive ratings, the application provides little motivation for users to take the time to add trust ratings for people they know and with whom they exchange mail. To gain some insight into how TrustMail might impact a user's mailbox, a large network with many users is required. Although the Trust Project had about 2,000 members, it is not ideal for this type of analysis because it only connects a small community of users, and thus it would only be possible to analyze the mailboxes of a few users. The ultimate application of TrustMail would involve a much larger network or a better connected community. Since this type of social network with trust ratings was not available to test TrustMail, it had to be generated from other existing data. Our analysis uses the Enron email dataset, a publicly available, real collection of email messages, to understand the social connections within a community and what that indicates for the potential success of TrustMail.

The Enron email dataset is a collection of the mail folders of 150 Enron employees, and it contains over 1.5 million messages both sent and received. There are over 6,000 unique senders in the corpus, and over 28,000 unique recipients. These numbers are much greater than the number of users whose mailboxes have been collected because they represent everyone who has sent a message to the users, everyone who has been cc'd on a message to the users, and everyone the users have emailed. The collection was made available by the Federal Energy Regulatory Commission in late 2003 in response to the legal investigation of the company. Because the messages represent a single community, they are ideal for analyzing the potential of TrustMail. Each message in the corpus was read, and an edge was added from the sender to each of the recipients. This produced an initial social network, although the connections are weak. To be sure that the links between people represented a relationship, connections were removed for any interactions that occurred only once; edges were only added from source to sink when the source had emailed the sink at least twice.

This social network is obviously lacking trust values. While the strength of relationships could be derived from the corpus of messages, this measure would not correlate to trust as it is defined here. Instead, we use the structure of the social connections in the email collection to indicate what percentage of messages might receive a rating in a social network-based system like TrustMail.

The Enron data allows us to see exactly who has emailed each user. A list of all individuals who sent mail to a given user is compiled. The network is searched for a path from the recipient to each sender. These calculations allow us to determine, in this email corpus, what percentage of senders could be given trust ratings if there were actually a trust network supporting the Enron users.

An analysis of the Enron network revealed the following statistics.

—37% of recipients had direct connections to people who sent them email in the social network. In other words, 37% of the time, the recipient had emailed the sender of a received message.

—55% of senders who were not directly connected to the recipient could be reached through paths in the social network.

—Thus, a total of 92% of all senders could be rated if trust values were present in the social network.

While there are no trust values in this network, the analysis does suggest that the social network has a structure that would allow a vast majority of messages to be rated. Of course, this is a best-case analysis for TrustMail; it is unlikely that everyone would give a trust rating to every person with whom two or more emails had been exchanged. However, we take two points from this understanding of the Enron collection. First, because so many senders could potentially be rated with a trust system, users would see a benefit from making trust statements. If the interface supports quick, easy annotation of the trustworthiness of email partners, there is potential for a self-supporting cycle; users add annotations, see the effects, and are encouraged to add more annotations. Second, even if users are not all adding trust statements about their email partners, the fact that the social network connects such a large percentage of people suggests that alternate techniques that use heuristics when trust data is not available may be worth considering as a lower-effort method for supplementing trust inferences to rate messages.

5.4 Implementation

TrustMail raises issues of implementation that are relevant to many other applications that might integrate trust data. There are several major concerns such as where is the social network data stored and where is computation handled. We believe the most effective system will have a centralized service (or several centralized services) that users can access via a Web service or other network protocol. Centralization has benefits with respect to the amount of data transfer and privacy.

The semantic Web nature of the trust networks we describe here means that the data will be distributed across the Web. However, that data must be collected into a centralized model in order to perform the computations over it. This collection process is something that is beyond the scope of the resources the general user is willing to allocate. It requires spidering files, downloading large amounts of data, and allocating resources to determine when people have been found in multiple files. Furthermore, the potential for frequent additions and updates to the data would require the model-building process to be repeated often. For these reasons alone, users would benefit from having a centralized service collect this data and build a model of it.

Centralized data collection does not necessarily mean there must be centralized computation. Users could download the model and make the trust computations locally. However, this still would require a large download. The average medium-sized social network size has tens of thousands of users, and overall the average social network has over one million users. Adding in the relationships and trust information produces several megabytes of data.

File size aside, there are more compelling grounds for moving computation off of the user's computer: privacy. For the trust system to work successfully, users must provide accurate trust ratings. If those ratings can be seen by anyone in the network, users might be dissuaded from assigning bad ratings for fear of insulting people. To avoid the social pressures that lead to less-than-honest ratings, it is important that they be kept private. However, the need for privacy must be reconciled with the nature of the semantic Web which is based on publicly-accessible files.

One approach that has been implemented with the Trust Project network used in this work is an encryption scheme. Users maintain a public FOAF file with their basic social networking information. Then, the private data that they do not want shared with the general public is placed in a separate file. In the current configuration, the users encrypt their files with the public key of the service at http://foaf.dk/hosting/. This service, in turn, collects the public keys of other services that may want to access the private data. Based on the user's preferences, the files are made available to different services encrypted with the correct public key. Relevant to this work, the public key of our trust service is available at the hosting site so users can encrypt their trust files and allow us to access them and decrypt them to get the data. This model of posting files encrypted with public keys offers a solution to the privacy problem while still maintaining the open, public nature of the semantic Web.

With this centralized solution, applications can query the trust network via a Web service. Depending on the security required by the application, this can implement authentication features or other measures to protect the confidentiality of the data. A central service can also improve the computational aspects of computing trust. As computations are made, the service can cache results for later use.

6. CONCLUSION AND FUTURE WORK

In this article, we presented the foundations necessary for understanding trust in Web-based social networks and for using trust networks in applications. We presented a definition of trust and illustrated how it could be incorporated into current models of social networks. We then described two algorithms for calculating recommendations about how much one person should trust another based on a personal perspective on the trust network. A theoretical analysis showed that we can expect the recommendations made by these algorithms to be highly accurate relative to the initial accuracy of trust ratings in the system, and this analysis is supported by experimental results. We then described the potential for using these algorithms and their variations in applications to create software that is aware of the user's social preferences by presenting a prototype of TrustMail, an email client that incorporates direct and inferred trust values.

This work used binary trust ratings (trusted or not trusted) for nodes to allow for precise analyses. In reality, trust is not so simple as to be expressed with only two values. A much wider spectrum of values is necessary to capture the complexity of trust. The nonrounding algorithm was a first step in this direction. Nodes assigned values with binary ratings, but values in the range of {0,1}, were used as the algorithm passed values back up toward the sink. By changing the initially assigned values from binary to a more continuous system, users will be able to express more nuanced and accurate trust values for their friends. Future work in this area will require a more theoretical analysis of results with a wider range of values as well as a comparison with other algorithms. Our existing work extends into this space, and our current results show that algorithms similar to the ones presented here work well in social networks with continuous trust values.

These new algorithms also contain refinements that take into account other features of the network's topology. For example, the algorithms presented in this article use a uniform value for p_a among good nodes. In reality, we can expect that nodes closer to the source will be more likely to agree with it than nodes many steps away. This might mean that limiting the length of paths followed from the source to the sink could lead to better results. In small-world networks, the average shortest path between two nodes is logarithmic with the size of the network. Thus suggests that limiting the search to a short radius around the source will still find a number of paths. Our ongoing research in this space is encouraging, and length-limited trust recommendations seem to be more accurate than nonlimited ones.

The range of applications where these inferred trust values can be applied is also much wider than the email context shown here with TrustMail. We are currently investigating the use of trust inferences as a type of collaborative filtering algorithm for generating predictive movie ratings. To this end, we have set up a social network called FilmTrust that combines trust, social networks, and movie ratings. Our preliminary results show that, in certain cases, the trust-based predictions outperform most other systems. We are actively working in this space to push the boundaries of how trust inferences can be applied.

REFERENCES

ABDUL-RAHMAN, A. AND HAILES, S. 2000. Supporting trust in virtual communities. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*. Maui, HW.

AVESANI, P., MASSA, P., AND TIELLA, R. 2004. Moleskiing: a Trust-aware decentralized recommender system. In *Proceedings of the 1st Workshop on Friend of a Friend, Social Networking and the (Semantic) Web*. Galway, Ireland.

BADRUL, S., KARYPIS, G., KONSTAN, J., AND RIEDL, J. 2001. Item-based collaborative filtering recommendation algorithms. *The Proceedings of WWW Conference (May), Hong Kong*.

BARBER, K. S. AND KIM, J. 2000. Belief revision process based on trust: Agents evaluating reputation of information sources. *Lecture Notes In Computer Science*; vol. 2246, 73–82.

BETH, T., BORCHERDING, M., AND KLEIN, B. 1994. Valuation of trust in open networks. In *Proceedings of ESORICS 94* (Nov.) Brighton, UK.

BLOOM, B. 1970. Space/time trade-offs in hash coding with allowable errors. *Comm. ACM 13*, 7, 422–426.

BOYKIN, P. O. AND ROYCHOWDHURY, V. 2004. Personal email networks: an effective anti-spam tool. http://www.arxiv.org/abs/cond-mat/0402143.

CASTELFRANCHI, C. AND FALCONE, R. 1998. Principles of trust for MAS: Cognitive anatomy, social importance, and quantification. In *Proceedings of the 3rd International Conference on Multi Agent Systems.*

CASTELFRANCHI, C. AND FALCONE, R. 2002. Social trust: A cognitive approach. In *Trust and Deception in Virtual Societies*. C. Casteleranchi and Y.-H. Tan, Eds. Kluwer Academic Publishers, Dordrecht, Holland.

COOK, K. ED.. 2001. *Trust in Society*, Russell Sage Foundation, New York, NY.

CROUCHER, T. 2004. A model of trust and anonymity in a content rating system for e-learning systems. In *Proceedings of 1st Workshop on Friend of a Friend, Social Networking and the (Semantic) Web*. Galway, Ireland.

DAVIS, G., YOO, M., AND BAKER, W. 2003. The small world of the american corporate elite. *Strategic Organization 1*, 3, 301–326.

DEUTSCH, M. 1962. Cooperation and trust. Some theoretical notes. In *Nebraska Symposium on Motivation* Jones, M. R. Jones Ed. Nebraska University Press.

DUDEK, C. 2003. Visual appeal and the formation of trust in e-commerce Web sites. Masters Thesis, Carleton University, Ottawa, Canada.

FOSTER, C. C., RAPOPORT, A., AND ORWANT, C. J. 1963. A study of a large cociogram: Elimination of free parameters. *Behav. Science 8*, 56–65.

GIL, Y. AND RATNAKAR, V. 2002. Trusting information sources one citizen at a time. In *Proceedings of the 1st International Semantic Web Conference (ISWC)*. Sardinia, Italy.

GOLBECK, J., PARSIA, B., AND HENDLER, J. 2003. Trust networks on the semantic Web. In *Proceedings of Cooperative Information Agents*. Helsinki, Finland.

GOLBECK, J. AND HENDLER, J. 2004. Reputation network analysis for email filtering. In *Proceedings of the 1st Conference on Email and Anti-Spam*. Mountain View, CA.

GOLBECK, J. AND HENDLER, J. 2004. Accuracy of metrics for inferring trust and reputation. In *Proceedings of 14th International Conference on Knowledge Engineering and Knowledge Management*. Northamptonshire, UK.

GOLBECK, J. AND HENDLER, J. 2006. FilmTrust: Movie recommendations using trust in Web-based social networks. In *Proceedings of the IEEE Consumer Communications and Networking Conference*. Las Vegas, NV.

GOLEMBIEWSKI, R. T. AND MCCONKIE, M. 1975. The centrality of interpersonal trust in group processes. In *Theories of Group Processes*, C. Cooper, Ed. Wiley, Hoboken, NJ.

HARDIN, R. 2002. *Trust & Trustworthiness*. Russell Sage Foundation. New York, NY.

HERLOCKER , J. L., KONSTAN, J. A., TERVEEN, L. G., AND T. RIEDL, J. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inform. Syst. 22*, 1, 5–53.

JONKER, C. AND TREUR, J. 1999. Formal analysis of models for the dynamics of trust based on experiences. In *Multi-Agent System Engineering, Proceedings of the 9th European Workshop on Modeling Autonomous Agents in a Multi-Agent World*.

JØSANG, A. 1996. The right type of trust for distributed systems. In *Proceedings of the 1996 New Security Paradigms Workshop*.

KAMVAR, S. D., SCHLOSSER, M. T., AND GARCIA-MOLINA, H. 2003. The EigenTrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th International World Wide Web Conference*. Budapest, Hungary.

LEVIN, R. AND AIKEN, A. 1998. Attack resistant trust metrics for public key certification. In *7th USENIX Security Symposium*. San Antonio, TX.

MAES, P. AND KOZIEROK, R. 1994. Agents that reduce work and information overload. *Comm. ACM*. *37*, 7, 30-40.

MARSH, S. 1994. Formalising trust as a computational concept. PhD thesis, Department of Mathematics and Computer Science, University of Stirling.

MASSA, P. AND AVESANI, P. 2004. Trust-aware collaborative filtering for recommender systems. In *Proceedings of the International Conference on Cooperative Information Systems (CoopIS)*.

MAURER, U. 1996. Modeling a public-key infrastructure. *Proceedings of Computer Security (ESORICS '96)*.

NEWMAN, M. E. J. 2001. The structure of scientific collaboration networks. In *Proceedings of the National Academy of Sciences*, (Jan.) *98*, 404–409.

PAGE, L., BRIN, S., MOTWANI, R., AND WINOGRAD, T. 1998. The PageRank citation ranking: Bringing order to the Web. Tech. Rep. Stanford University, Stanford, CA.

RICHARDSON, M., AGRAWAL, R., AND DOMINGOS, P. 2003. Trust management for the semantic Web, *Proceedings of the 2nd International Semantic Web Conference*. Sanibel Island, FL.

SINHA, R. AND SWEARINGEN, K. 2001. Comparing recommendations made by online systems and friends. In *Proceedings of the DELOS-NSF Workshop on Personalization and Recommender Systems in Digital Libraries*. Dublin, Ireland.

SWEARINGEN, K. AND SINHA, R. 2001. Beyond algorithms: An HCI perspective on recommender systems. In *Proceedings of the ACM SIGIR 2001 Workshop on Recommender Systems*. New Orleans, LA.

SZTOMPKA, P. 1999. *Trust: A Sociological Theory*. Cambridge University Press, Cambridge, UK.

WATTS, D. 1999, *Small Worlds: The Dynamics of Networks Between Rrder and Rrandomness*. Princeton University Press, Princeton, NJ.

WATTS, D. AND STROGATZ, S. H. 1998. Collective dynamics of small-world networks. *Nature*, *393*, 440–442.

YANIV, I., AND KLEINBERGER, E. 2000. Advice taking in decision making: Egocentric discounting and reputation formation. *Organizat. Behav. Human Decision Process. 83*, 2, 260–281.

ZIEGLER, C. N. AND LAUSEN, G. 2004a. Spreading activation models for trust propagation. *Proceedings of the IEEE International Conference on* E-*Technology,* E-*Commerce, and* E-*Service*, Taipei, Taiwan.

ZIEGLER, C.-N. AND LAUSEN, G. 2004b. Analyzing correlation between trust and user similarity in online communities. *Proceedings of 2nd International Conference on Trust Management*.

Received December 2004; revised October 2005; accepted November 2005