Towards a Gravity-Based Trust Model for Social Networking Systems

Muthucumaru Maheswaran, Hon Cheong Tang, and Ahmad Ghunaim
Advanced Networking Research Lab
School of Computer Science
McGill University
Montreal, QC, Canada
Email: {firstname.lastname}@mcgill.ca

Abstract - *Web-based social networks are emerging as the top applications on the Internet. With this immense popularity, many of the shortcomings of the current social network deployments are also coming to light. One of the glaring problems with existing web-based social networks is trust management. In this paper, we focus on trust modeling in social networks. Another allied issue that is not considered here is using trust in managing the activities within the social network. We introduce a gravity-based model for estimating trust. We present the complete model along with the trust computation algorithms. We present initial results from a simulation study that investigates the feasibility of the proposed scheme.*

1 Introduction

Trust modeling in large-scale distributed systems has been an active topic in the past several years in academia and industry. Most work in this topic have been motivated by two important applications: e-commerce and peer-to-peer (P2P) file sharing. In an e-commerce application, a buyer takes a certain amount of risk when she/he makes a purchase from an online vendor. One of the major contributor of risk is the fact that the goods cannot be physically inspected before proceeding with the purchase. This is a significant disadvantage for the e-retailers compared to their brick-and-mortar counterparts. To offset these concerns, simple trust modeling schemes in the form of rating systems have been successfully deployed by various e-retailers such as amazon.com and ebay.com. These rating schemes give buyers an idea of how the goods were perceived by previous buyers. Similar concerns also apply to P2P file-sharing systems where replicas of the content are offered by various servers. Some of these servers could be offering corrupted replicas of the content for malicious reasons. The clients would like to avoid servers hosting corrupted content to maximize their download efficiency. Trust modeling process is used in these systems to determine the reputations of the servers for providing genuine content.

Although local transactions among peers are the sources of trust in the above applications, the ultimate objective is to use these local values to compute global trust measures. The challenge in trust aggregation is to prevent malicious intervention by interested parties. For example, in e-commerce applications, vendors want to project highest possible ratings for their goods to attract buyers. To achieve this goal, malicious vendors can introduce bogus ratings into the system. Such bogus ratings can be of two types: positive ratings for the vendor's goods or negative ratings for the competitor's goods. Various schemes with their own limitations have been developed to address such attacks on trust aggregation processes.

Another application that has attracted lot of users, in recent years, besides e-commerce and P2P applications is social networking. In social networking systems such as myspace.com and linkedin.com users create profile pages and connect to their friends' profile pages. While some social networking systems such as myspace.com encourage users to make connections based on online encounters, other social networking

27th International Conference on Distributed Computing Systems Workshops (ICDCSW'07)
0-7695-2838-4/07 $20.00 © 2007 IEEE

systems such as linkedin.com discourage connections that are not supported by offline relations. Most social networking systems use a very simple trust model, where all friends are considered equal. Certain social networking systems such as orkut.com relax this restriction and allow users to annotate connections with trust levels. Although annotating connections adds flexibility to the social networking systems, it is still a *static* solution. For instance, the strength of a friendship between two users can change with time. We need trust inference schemes that can automatically change the strengths of the social connections to reflect the changes in friendships with time.

The trust aggregation (reputation management) schemes from e-commerce and P2P systems support dynamic trust variations by continually recomputing trust based on the feedback provided by the users. However, a straightforward adoption of such trust aggregation is not suitable for social networking systems. In particular, any trust computation process for the social networking systems should be able to handle the constraints imposed by the social relations.

In this paper, we present a new trust modeling process for social networks. The proposed trust model acts in two stages. In the first stage, the strengths of the friendships are recomputed along with the extent of the trusted social neighborhood for each user. In the second stage, the social neighborhood is used to compute the effective trust flow for users not in the social neighborhood.

The first stage allows a user to annotate the connections it has with others with trust values or constraints. A distributed optimization procedure is periodically invoked to re-compute the trust values on the connections. The trust values will be strengthened if the social connections reinforce the trust relations and constraints admit higher trust values for the connections. Similarly, trust values on the social connections can be weakened if the social connections contradict the trust relations and constraints admit lower trust values for the connections. Each user on the social network periodically runs the distributed optimization process to recompute the strengths of the social connections and the extent of the trusted social neighborhood.

One of the key challenges for trust modeling is the "user acceptance" of the trust values generated by the modeling process. That is, a user can disregard the reputation measures because he/she does not understand how the trust management process works or does not agree with it. The trust model proposed here attempts to address this concern in different ways: (i) splitting the modeling process into two stages, (ii) providing facilities for closer inspection of the strengths of the friendships, and (iii) providing facilities to compute the trust flows to users beyond the trusted social neighborhood in a "qualified" manner. In particular, the proposed trust model can answer the following questions regarding the trust distributions among the users. (a) What is my trusted social neighborhood? (b) Why certain users are excluded or included in my trusted social neighborhood? (c) Why certain users are weakly or strongly connected within the social neighborhood? (d) How much trust do my k closest social neighbors have on a given target? We contend that by facilitating answers for these questions, the proposed trust model computes trust that better integrates with social networking systems.

Section 2 briefly describes related literature in trust management. The gravity-based trust model is introduced in Section 3 along with the algorithms for computing trust in this model. Section 4 presents some early stage results from a simulation study into the convergence properties of the trust estimation process.

2. Related Work

In this section, we provide a brief discussion of related research work. We group related work into four categories: structural, sensitive, anonymous, and fuzzy trust modeling.

The structural trust modeling works exploit some structural aspect of the social network in computing trust measures. The Tidal trust [Gol06] is a new trust metric for semantic web or friend-of-a-friend networks. The local trust values are assigned statically and these values are used for inferring trust between two nodes not directly connected on the social network. The inference process is based on two observations: recommendations from highly trusted peers are more accurate and trust inferred through shorter paths are more precise. The Tidal-

trust algorithm is simple to implement. However, it does not update the trust values dynamically or improve the accuracy based on direct experiences. The PowerTrust algorithm [ZhH06] is another structural trust modeling approach which is meant for P2P networks. It exploits the observation that most feedback comes from few "power" peers to construct a robust and scalable trust modeling scheme. In PowerTrust, peers evaluate each interaction and compute local trust values. The global trust measures are obtained via random walks that aggregate the local trust values. Once the power peers are identified through the reputation values, these peers are used in a subsequent look-ahead random walk that is based on Markov chain to update the global trust values.

The Advogato trust metric [Lev04] is meant for determining trust of nodes in a P2P community. Here trust is computed as a group instead of computing one peer at a time. The group trust computation problem is treated as a network flow problem. The source of trust is a "seed" who can be a domain expert with very high stature in the community. Advogato claims to prevent a common class of attacks where malicious peers create zoombie nodes and try to acquire additional trust through them. Another trust computation that is related to Advogato is the AppleSeed trust metric [ZiL05]. Unlike the Advogato metric that computes trust over the whole network, the AppleSeed can compute trust over part of a graph. This increases the overall scalability of the approach. The basic idea of AppleSeed is to use a spreading activation model borrowed from works that examine semantic search in contextual network graphs for computing trust.

Although most trust modeling approaches adapt output values with time, some techniques react quickly for changing trust conditions than the others. We refer to techniques that react in exceptionally short time durations to trust changes as sensitive trust metrics.

The PeerTrust [XiL04] is a reputation-based metric for P2P communities. The PeerTrust addresses several key issues with trust management systems: included contexts into trust computation, incorporated trust of recommenders into trust aggregation such that recommendations provided by trusted nodes are highly regarded, and a P2P system architecture for trust evaluation. To deal with time-based trust variations, PeerTrust uses a simple adaptive time window-based algorithm for tracking short-term node behavior. The reputation of nodes are rapidly dropped if malicious behavior is detected. The DynamicTrust [DuS05] is another sensitive trust modeling approach. It shares the same goals as the PeerTrust.

The anonymous trust metrics emphasize on maintaining the anonymity of the peers in the network. The motivation is that a trust metric with anonymity will allow peers to submit more accurate feedbacks. However, it is important to make sure that anonymity is not abused by malicious peers. TrustMe [SiL03] and P2PRep [ArD06] are examples of anonymous trust metrics.

3 A New Trust Model

3.1 Overview

Social networks are graphs where the nodes represent users and the edges represent relations among the users. In most existing social networking systems, the edges represent trusted relations with fixed value. That is, if node X_i is directly connected to nodes X_j and X_k, then X_i trusts X_j and X_k at the same trust level. In some social networking systems, the edges can be annotated with the trust of the relationship to account for heterogeneity in friendship strengths. In either case, the effective trust between two users (or nodes) is given by the number of edges along the shortest path that connects them on the social network or a function of that parameter.

This simple trust model has several limitations including lack of support for time-based trust variations and trust contexts. Although the trust annotations associated with an edge can be changed to reflect varying trust levels, this process should be automated to be effective. If constant user input is required for trust updates, in most cases the trust values will be stale. The trust contexts are another important concern. For example, a node (or user) X_i may not trust node X_j for medical advise but could highly trust X_j for fixing cars. The idea that

trust is *context* dependent has been widely accepted in the literature [XiL04].

To model the contexts we propose a novel concept called a *trust space*. Let $C_0, C_1, C_2, \ldots C_{m-1}$ be a maximal set of independent contexts. We define a m-dimensional *trust space* (Γ) that is formed by a basis set derived from the independent set of contexts. The exact procedure for determining the independent contexts and deriving the basis for the trust space is a topic that needs further research.

All nodes that are part of the social network are placed on this trust space. The trust between two node X_i and X_j is quantified by a *trust distance* that is measurable in the above defined trust space. For example, the distance d_{ij} between X_i and X_j is an m-dimensional vector in the trust space. Two nodes that do not have any prior trust relations have a "neutral" distance $\bar{d}$ along all dimensions. This neutral distance corresponds to *no trust opinion*. As the nodes start interacting they form positive or negative trust opinions of the other. The trust distance between two nodes is decreased when the trust opinion between the nodes is favorable. Similarly, the distance is increased when the trust opinion between the nodes is unfavorable. We use bounded functions to ensure that the trust distance stay within a given range.

3.2 Computing Trusted Social Neighborhood

Social networks tend to have very large number of nodes (i.e., very large n values). However, a given node (or user) typically interacts with a very small fraction of the total population in the social network. Therefore, the number of interactions we see in an n node social network tend to be far less than the $n(n-1)/2$ possible interactions. This observation has two implications for the trusted social neighborhood computation process. First, a given node does not need to compute the trust distance to all nodes in the social network. Second, the trust distance computation process should work with partial information. That is, we should be able to compute the trust distances despite missing data.

To compute the trust distances with trust observations only on part of the relations, we adopt ideas from distributed virtual coordinate systems from the networking area [NgZ02, DaC04]. In networking, the virtual coordinates were meant to map the hosts onto a Cartesian space based on delay measurements obtained from the network. One of the primary advantages of the virtual coordinate system is its ability predict delay values for node pairs for which direct delay measurements do not exist using structural constraints.

We associate a virtual coordinate y_i from the trust space Γ with each node X_i. The initial value for the virtual coordinate is randomly assigned. Suppose the trust value between nodes X_i and X_j is d_{ij}. We can define the total error as a sum of a squared-error function as follows:

$$E = \sum_i \sum_i (d_{ij} - ||y_i - y_j||)^2 \tag{1}$$

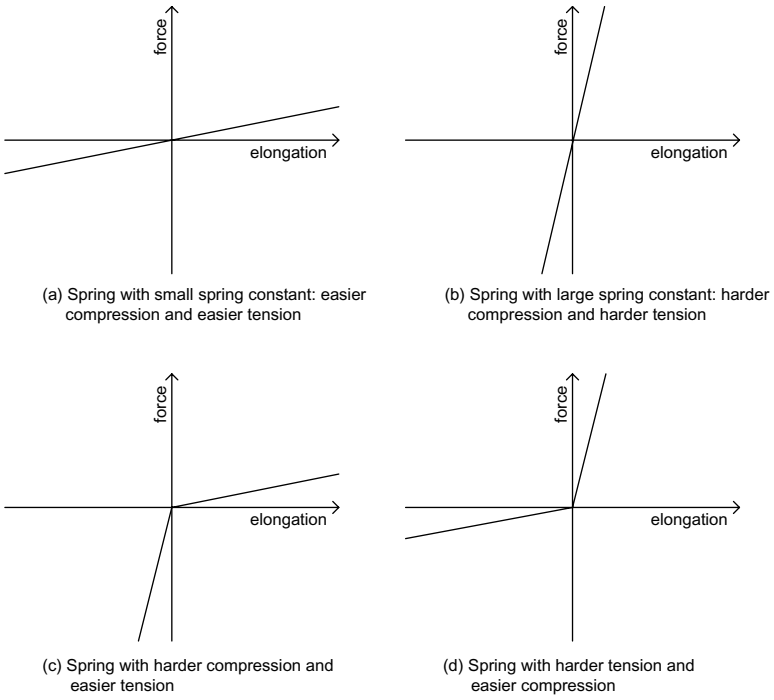
Prior work [DaC04] in virtual coordinates has shown that simulating a network of mechanical springs produces coordinates that correspond to the minimum of the error function E in Equation (1). We connect nodes X_i and X_j by a spring with natural length d_{ij}. The error is given by the elongation of the spring. The distributed algorithm we present below adjusts the springs so that the total elongation of the system of springs is minimized. From Hooke's' law, for a spring that is elongated by x and having a spring constant k, the restoring force acting on the spring is given by:

$$F = -kx \tag{2}$$

The potential energy stored in a spring with elongation x is proportional to x^2. Therefore, the total potential energy of the spring system is proportional to the total error E in Equation (1). Although the minimum energy coordinate position corresponds to the minimum error situation, there is no guarantee that the spring simulation will find the global minimum. Instead, it might converge to a local minimum.

The spring between nodes X_i and X_j is under tension if the elongation is positive and compression if the elongation is negative. Figure 3.2 shows the possible variation of the spring constant with the tension/compression

state of the spring. The natural length of the spring denotes the trust observation made by the nodes. If this trust observation is a strict constraint that should not be changed much by the error minimization process, then the spring constant should be set to a high value. If the trust constraint is a lower limit, then the spring constant should be setup as in Figure 3.2(c) and it should be setup as in Figure 3.2(d) for an upper limit. For example, if node X_i considers X_j as a *definite* good friend, then it can use the upper limit constraint which prevents the minimization process from stretching the distance between X_i and X_j too much. Similarly for the lower limit.



(a) Spring with small spring constant: easier compression and easier tension

(b) Spring with large spring constant: harder compression and harder tension

(c) Spring with harder compression and easier tension

(d) Spring with harder tension and easier compression

Let k_t be the spring constant when the spring is under tension and k_c be the spring constant when the spring is under compression. The restoration force between nodes X_i and X_j that is in the opposite direction of the elongation is given by:

$$F_{ij} = -k((y_i - y_j) - d_{ij}) \tag{3}$$

If $\|y_i - y_j\| > \|d_{ij}\|$ then $k = k_t$ and $k = k_c$ otherwise. In the spring simulation, the node X_i moves in the direction of the net force that acts on it. The distance moved by the node is determined by time step. If the time step is large, the algorithm will not converge. In Vivaldi [DaC04], the time step values were changed adaptively. Although the algorithm in Figure 1 uses a fixed time step, we could extend it to adaptively change the time step value.

Once the virtual coordinate computation process converges, each node X_i will have a coordinate value y_i. These coordinate values are used by X_i to identify other nodes in its trusted social neighborhood. Node X_i could discover new trusted neighbors as a result of this process. To compute the trusted social neighborhood, neighbors should exchange trust observations and virtual coordinates. At bootstrap, this exchange is performed among the initial set of "foundational" neighbors. In subsequent rounds, the neighborhood can evolve depending on the trust distances and trust observations.

While the above process allows us to quantify the trust within the trusted social neighborhood, in certain situations we want to measure the effective trust to nodes that lie outside the neighborhood. The basic idea is to some form of rating for this target from the nodes in the trusted social neighborhood. We explain this process next.

```
repeat
  for all node X_i do
    y_i ← virtual coordinate of node X_i
    for all node X_j do
      y_j ← virtual coordinate of node X_j
      d_ij ← observed trust value between X_i and X_j
      f_ij = ((y_i − y_j) − d_ij)
      if ||y_i − y_j|| > ||d_ij|| then
        k = k_t
      else
        k = k_c
      end if
      /* μ is the time step */
      y_i = y_i + μkf_ij
    end for
  end for
  errvar = error variation over last m iterations
until errvar < threshold
```

Figure 1: Pseudo code for centralized trust computation using coordinates.

3.3 Computing Trust Flows

Another measure that is associated with each node is its age. For instance, the age (a_j) of node X_j denotes the time the node has spent in the social network. Node X_j could prove its age by producing certificates issued by other nodes that contain appropriate time-stamp values. It is up to other nodes to determine the age from this evidence. Therefore, the age of node X_j can be estimated differently by different nodes. The age of the social network as a whole is the age of the oldest node in the network. In a very large network with decentralized architecture, it is hard to locate the oldest node in the network. Therefore, we approximate the age of the system by computing the maximum of the ages of all nodes in the social neighborhood.

Using the trust distance and age parameters presented above, we formulate a computation strategy for trust. Our trust computation approach draws heavily on the gravity model used in force computation. *Further, we want the trust computation scheme to capture several common observations about trust relations in peer-to-peer social networks. The trust relations among long standing peers are well formed and are either largely positive or largely negative based on past experience. On the other hand, trust relations among newcomers are not well formed. The trust values on these relations are not fully formed due to inadequate past experience.*

Let X_i and X_j be two nodes. Suppose a_{ij} is the age of X_j as estimated by X_i based on evidence provided by X_j and a_i^s is the age of the system as estimated by X_i. Let $g(r)$ be a function defined on the distance between the two nodes, where r is the distance computed from the virtual coordinates computed above. Therefore, $r_{ij} = y_i - y_j$. The trust X_i places on X_j is defined by the following equation:

$$T_{ij} = \lambda \frac{a_i^s a_{ij}}{g(r_{ij})} \tag{4}$$

In this work, we want to define function $g()$ such that the effective trust value at $r = d_n$ is zero and trust takes negative values as r increases above d_n. The following function for $g(r) = \frac{\|r\|^3}{r - d_n}$ satisfies the above objective. Thus, we can re-express the trust X_i places on X_j as:

$$T_{ij} = \lambda \frac{a_i^s a_{ij}(d_n - r)}{\|r\|^3} \tag{5}$$

Equation (5) shows that T_{ij} takes positive values and negative values as r varies. The positive values denote favorable opinions and negative values, on the other hand, denote unfavorable opinions. Further, the trust values given by Equation (5) are dependent only on the estimates of system age, age of the target node, and trust distance.

4 Simulation Results and Discussion

Simulations were carried out to evaluate the convergence properties of the coordinate-based trust estimation processes. We generated 100 example social network configurations for number of nodes (n) ranging from 3 to 40. Each network configuration had only a single context. For each configuration each node had between 1 to 5 trusted neighbors. For each trusted neighbor, a trust value in the range [1..5] was assigned.

In Figure 2, we show the variation of the trust estimation error with the number of nodes. In this experiment, the spring constants were set to 1.0. That is, the springs have no additional stiffness. In Figure 3, we show the variation of the trust estimation error with the number of nodes for spring constants of $k = 2.5$ and $k = 0.4$. The spring constant value $k = 2.5$ indicates a scenario where the springs are stiffer than normal and $k = 0.4$ indicates a situation where the spring is weaker than normal. We can observe that the trust estimation error is higher for these conditions than the nominal case. In Figure 4, we show a situation where the springs are easier to compress or easier to extend.

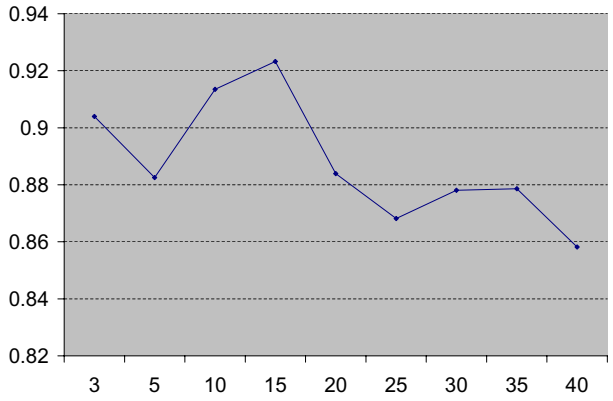


Figure 2: Trust estimation error variation with number of nodes.

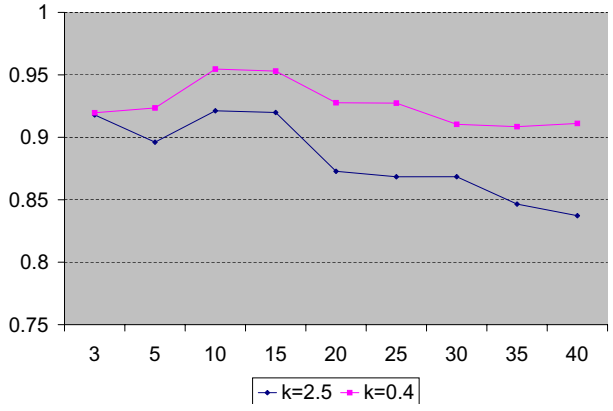


Figure 3: Trust estimation error variation with number of nodes.

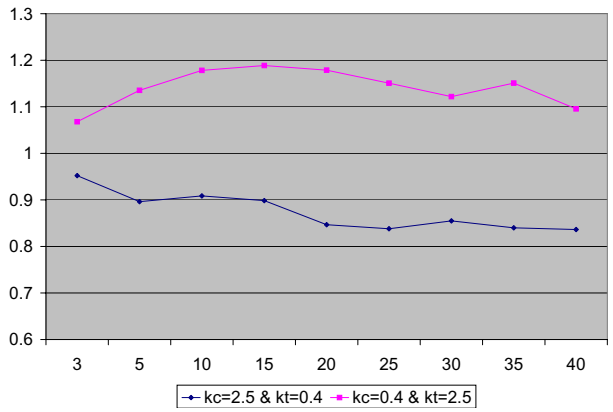


Figure 4: Trust estimation error variation with number of nodes.

5 Conclusions and Future Work

In this paper, we proposed a new trust model for social networks. We completely described the trust model along with the trust computation processes. Some early stage simulation results were also presented. The results indicate that the trust estimation processes converge within relatively low number of iterations.

This is part of an ongoing work to build a peer-to-peer social networking system at McGill University.

References

[ArD06] R. Aringhieri, E. Damiani, S. D. Capitani, D. Vimercati, S. Paraboschi, and P. Samarati, "Fuzzy techniques for trust and reputation management in anoymous peer-to-peer systems," *Journal of the American Society for Information Science and Technology*, Vol. 57, No. 4, Feb. 2006.

[DaC04] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," *Proc. of the ACM SIGCOMM'04 Conference*, Dec. 2004.

[DuS05] C. Duma, N. Shahmehri, and G. Caronni, "Trust metrics for peer-to-peer systems," *2nd Int'l Workshop on P2P Data Management, Security and Trust*, Aug. 2005.

[Gol06] J. Golbeck, "Combining provenance with trust in social networks for semantic web content filtering," *Int'l Provenance and Annotation Workshop*, 2006.

[Lev04] R. Levien, *Attack Resistant Trust Metrics*, PhD thesis, Dept. of Computer Science, University of California, Berkeley, 2004 (manuscript).

[NgZ02] T. S. E. Ng and H. Zhang, "Predicting internet network distance with coordinate-based approaches," *IEEE Infocom*, June 2002.

[SiL03] A. Singh and L. Liu, "Trustme: Anonymous management of trust relationships in decentralized p2p systems," *IEEE Int'l Conference on Peer-to-Peer Computing*, 2003.

[XiL04] L. Xiong and L. Liu, "Peertrust: Supporting reputation-based trust in peer-to-peer communities," *IEEE Transactions on Knowledge and Data Engineering: Special Issue on Peer-to-Peer Based Data Management*, Vol. 16, No. 7, July 2004, pp. 843–857.

[ZhH06] R. Zhou and K. Hwang, "Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing," *IEEE Transactions on Parallel and Distributed Systems*, 2006.

[ZiL05] C. Ziegler and G. Lausen, "Propagation models for trust and distrust in social network," *Information Systems Frontiers*, Vol. 7, No. 4-5, Dec. 2005, pp. 337–358.