Building Trust in Decentralized Peer-to-Peer Electronic Communities

Li Xiong and Ling Liu
College of Computing
Georgia Institute of Technology
{lxiong, lingliu}@cc.gatech.edu

Abstract

Many players in electronic markets have to cope with much higher amount of uncertainty as to quality and reliability of the products they buy and the information they obtain from other peers in the respective online business communities. One way to address this uncertainty problem is to use information such as feedbacks about past experiences to help making recommendation and judgment on product quality and information reliability. This paper presents PeerTrust, a simple yet effective reputation-based trust mechanism for quantifying and comparing the trustworthiness of peers in a decentralized peer-to-peer electronic marketplace. There are three main contributions in this paper. First, we argue that the trust models based solely on feedbacks from other peers in the community is inaccurate and ineffective. We introduce three basic trust parameters in computing trust within an electronic community. In addition to feedbacks in terms of amount of satisfaction, we incorporate the feedback context such as the total number of transactions and the credibility of the feedback sources into the PeerTrust model for evaluating the trustworthiness of peers. Second, we develop a basic trust metric that combines the three critical parameters to compare and quantify the trustworthiness of peers. Third, we present a concrete method to validate the proposed trust model and report the set of initial experiments, showing the feasibility, costs, and benefits of our approach.

Keywords Peer-to-Peer, Trust, Reputation, Credibility, Electronic Communities.

1 INTRODUCTION

The interest in peer-to-peer (P2P) computing continues to grow since the rise and fall of Napster. Some popular P2P communities include file sharing systems such as Gnutella [2] and Freenet [1], person-to-person online auction sites such as eBay, and many business to business (B2B) services such as supply-chain-management networks. A new P2P agent marketplace [25] was also proposed as a new approach that overcomes the shortcomings of existing electronic markets through software agent-driven, peer-to-peer, iterative negotiations.

It is widely recognized that players in such electronic communities have to cope with much higher amount of uncertainty regarding to the quality and reliability of the products they buy or the information they obtain from other players in the respective online business communities. Put differently, in a decentralized peer-to-peer electronic community, peers often have to interact with unknown or unfamiliar peers and need to manage the risk involved with the interactions (transactions) without any presence of trusted third parties or trust authorities. One way to address this uncertainty problem is to use information such as feedbacks about past experiences to help making recommendation and judgment on the potential players.

For example, in a buyer-seller market, buyers are vulnerable to risks because of potential incomplete or distorted information provided by sellers. A recent study [10] reported results from both an online experiment and an online auction

market, which confirmed that trust can mitigate information asymmetry (the difference between the amounts of information the two transacting parties possess) by reducing transaction-specific risks, therefore generating price premiums for reputable sellers.

In recognition of the importance of trust in electronic markets, an immediate question to ask is how to assess trust. Most existing trust models in trusted computing focus on how to compute indirect trust given the input of direct trust valuations. Establishing and assessing trust is considered subjective and is outside the scope of these models [24]. There is an extensive amount of research focused on building trust for electronic markets through trusted third parties or intermediaries [15, 22, 9]. Unfortunately, subjective trust valuations are not suitable for a P2P system where peers are equal in their roles in the community. Decentralized P2P systems need technical mechanisms for computing trust directly and revising the trustworthiness of a peer continuously at runtime based on the feedbacks from other peers.

Reputation systems [16] provide a way for building trust through social control without trusted third parties. Yet most existing reputation mechanisms require a central server for storing and distributing the reputation information. Centralized control of reputation data makes the reputation management systems vulnerable to a variety of failures. A decentralized P2P trust management system aims at reducing or avoiding single point of failure and increasing scalability of the system performance. One of the main challenges for building a reputation system in a P2P electronic community is to effectively cope with the potential malicious behavior of peers. Peers can misbehave in a number of ways, such as serving corrupted or low-quality trust data, and providing false feedbacks on other peers (e.g., filing false complaints on other peers). Therefore, a trust mechanism has to be robust against various misuse behaviors of the system.

Aberer et al. [8] are among the first to look at the trust issues in a P2P information system and introduced a complaint-only trust management method without trusted third parties. The complaint-only trust metric works in very limited cases and is over-sensitive to the skewed distribution of the community and to several misbehaviors of the system.

The effectiveness of a trust model depends not only on the factors for risk assessment but also on the factors that affect the performance of a P2P system that supports trust. A scalable trust management requires efficient storage of trust data (both the data that are used to assess and compute trust and the trust values of peers) in a decentralized P2P network, and fast lookup of trust data when assessing the risk of a peer in performing a task.

Furthermore, there is a need of methods for experimental evaluation of a given trust model. Most traditional trust models only give an analytical model without any experimental validation due to the subjective nature of trust. Few discuss the implementation issues of a trust mechanism in a distributed environment and how to evaluate the effectiveness of a trust mechanism.

With these research questions in mind, we develop PeerTrust, a reputation based trust management system for peers to quantify and compare the trustworthiness of other peers in P2P communities. The PeerTrust model has two unique features. First, we identify three important factors for evaluating the trustworthiness of a peer: the feedback in terms of amount of satisfaction a peer obtains from other peers through interactions, the feedback context such as the total number of interactions that a peer has with other peers, and the balancing factor of trust for the feedback source (see Section 2.1). We demonstrate that existing trust models which take into account only the first factor − the amount of satisfaction (feedbacks) that others peers have over the given peer, is inaccurate when applied to evaluating the trust of a peer. A basic trust function is presented to illustrate how to combine these three critical factors in computing trustworthiness of peers in an online community (see Section 2.2). Second, our system also extends the basic trust model to a series of adaptive trust metrics that addresses a variety of common problems encountered in today's electronic markets and online communities. Due to space limitations, this paper only briefly discusses the adaptive trust metrics and focuses on presenting and evaluating the basic trust parameters and metric. A concrete trust function is presented as an example application to illustrate the importance of the three trust assessment factors, and demonstrate the effectiveness and robustness of the PeerTrust approach.

The paper also includes a discussion on the trust information dissemination architecture and design considerations in PeerTrust, and a report on the results of our initial experiments, showing the accuracy, robustness, and costs of the PeerTrust approach.

2 THE TRUST MODEL

In this section, we introduce the three trust parameters and a basic trust model that combines these three factors in evaluating trust of peers. To understand the usefulness and benefits of the PeerTrust model, we also provide a concrete form of the trust function to illustrate the PeerTrust model.

2.1 Three Basic Trust Parameters – Overcoming Inaccuracy

A variety of electronic markets and online community sites have reputation management built in, such as eBay, Slashdot, Amazon, Yahoo!Auction, and Auction Universe. However, to our knowledge, there is no comprehensive survey of all sites that use reputation management systems. From our experience with online auction sites, and the survey provided by Malaga in [16], most existing reputation-based trust management systems rely solely on the positive or negative feedbacks to evaluate and determine the trust of peers.

Let P denote the set of N peers in an electronic community built on top of a peer network, and $u, v \in P$ be peers in the community P. Note that at any given time, the number of active peers may be different, and not decidable in advance. Let $S(u, v, t)$ denote the amount of satisfaction peer u has with v up to transaction t. Let $T(u, t)$ be the evaluation of peer u's trust rating up to transaction t by the rest of peers in the community P. $T(u, t)$ in such systems can be defined as a function of $S(u, v, t)$ as follows:

$$T(u,t) = \sum_{v \in P, v \neq u} S(u,v,t) \tag{1}$$

We argue that the trust evaluation systems that aggregate individual peers' feedback scores are flawed in a number of ways.

Scenario 1: Consider a simple scenario where two peers obtain the same amount of satisfaction over the same time period but through different number of interactions. A peer A performs 10 interactions (services) with other peers up to time t and obtains satisfaction for each service it performs. Another peer B performs 100 interactions but only obtains satisfaction one out of ten services it performs. The two peers receive the same amount of satisfaction (feedback scores) through the services they perform, but it is obvious that if the trust is established in terms of the overall service quality, peer A will be considered more trustworthy than peer B. This example scenario shows that the amount of satisfaction will not be a fair measure without using the total number of interactions as the context, because the misbehaviors of peer A are not captured fairly.

Scenario 2: A peer happens to interact with only malicious peers that always badmouth other peers by falsely claim that another peer provides poor service. In this scenario, a trustworthy peer may end up getting a large number of false statements. Without a balance factor of trust built in, this peer will be evaluated incorrectly because of false statements even though it provides satisfactory service in every interaction. The role of a balance factor is used to filter out the false statements from the honest ones.

To address the above problems, in PeerTrust, a peer's trustworthiness is defined by an evaluation of the peer in terms of the degree of satisfaction it receives in providing service to other peers in the past. Such degree of satisfaction reflects to what extent the peer acts in other peers' best interests. We identify three important factors for such evaluation – the amount of satisfaction a peer obtains through interactions, the number of interactions the peer has with other peers, and a balance factor of trust. We show next that all three factors play an equally important role in evaluating the trustworthiness of a peer and how the problems with feedback-only methods are reduced or avoided.

Amount of Satisfaction
In a P2P network, the amount of satisfaction a peer receives regarding its service is usually resulting from the interactions other peers have had with this peer. Intuitively, during an interaction, the better a peer fulfills its part of the service agreement, the more satisfaction it will receive from the other peer. The larger the number of peers who are satisfied with a peer's service, the more it should be worthy of trust.

Number of Interactions
The total number of interactions a peer has with other peers in the P2P network is another important factor that affects the accuracy of feedback-based trust computation. It reflects over how many services the satisfaction is obtained. Put differently, the amount of satisfaction alone presents the feedbacks without a specific context. The number of interactions from which the feedbacks were derived is an important fairness measure with respect to the amount of satisfaction.

Balance factor of trust
The amount of satisfaction is simply a statement from a peer about another peer as to how satisfied it feels about the quality of the information/service provided by the other peer and is often collected from peers through a feedback system. Given that a peer may make false statements about another peer's service, the balance factor of trust is used to offset the potential of false feedback of peers and thus can be seen as a way to differentiate the credible amounts of satisfaction from the less credible ones.

Surprisingly, most existing trust mechanisms only takes into account the first factor, the amount of satisfaction that other peers have had over a given peer, when computing the trust value of this peer. Furthermore few have even considered the role of a balancing factor of trust in their trust computation models.

2.2 The Basic Trust Metric

The basic trust metric computes a trust measure by combining the three trust factors identified in the previous section. The goal of such a metric is to produce a fair measure of trust belief in the presence of unknown or unfamiliar peers and possible misbehaviors of such peers.

Let P denote the set of N peers in an electronic community built on top of a peer network, and $u, v \in P$ be peers in the community P. Let $I(u, v, t)$ denote the number of interactions that peer u has with v up to transaction t. Let $I(u, t)$ denote the total number of interactions peer u has with other peers in the network, i.e. $\sum_{v \in P, v \neq u} I(u, v, t)$. Let $S(u, v, t)$ denote the amount of satisfaction peer u has with v up to transaction t. Let $Cr(v, t)$ be the balance factor of trust that offsets the risk of non-credible feedbacks from v. The evaluation of peer u's trust rating up to transaction t by the rest of peers in the peer network P, $T(u, t)$, can be defined as a function of $S(u, v, t)$, $I(u, v, t)$, and $Cr(v, t)$. A simple and straightforward definition of $T(u, t)$ could be the ratio of the summary of a balanced amount of satisfaction that other peers have over u and the total number of interactions u has with other peers in the community P. That is:

$$T(u,t) = \frac{\sum_{v \in P, v \neq u} S(u,v,t) * Cr(v,t)}{\sum_{v \in P, v \neq u} I(u,v,t)} \tag{2}$$

The value of $T(u, t)$ is a global reputation measure of the peer u. In real life, reputation is the general estimation that a person is held by the public. It is often established through a good tractable history so that their behavior can be predicted reliably. A person consistently provides good service to others will in turn establish a good reputation. We are usually more willing to trust a person with a higher reputation. Similarly, in the equation 2, a higher value of $T(u, t)$ indicates that peer u is more trustworthy in terms of the collective evaluation of u by the peers who have had interactions with u. The value of $T(u, t)$ gives a measure that helps other peers to form a trust belief (opinion) about u. Each peer must consider to which degree the value of $T(u, t)$ will make it trust u. Different peers may have different perception over the same value of $T(u, t)$.

A simple decision rule can be conducted at peer w as follows:

$$\text{if } I(u,t) > c_1 \text{ and } T(u,t) > c_2, \text{ then u is trustworthy} \tag{3}$$

The threshold c_1 requires a minimum number of total interactions for a peer to make an informed decision. The threshold c_2 indicates how much the observing peer is willing to trust others. A more tolerant peer may have a lower threshold. It is a manifest of dispositional trust [18], which is the extent to which an entity has a consistent tendency to trust across a broad spectrum of situations and entities.

2.3 An Example Metric

We have presented a general trust metric for evaluating the trustworthiness of peers in a P2P electronic community. The next question one may ask immediately is how to measure the amount of satisfaction according to the service-level agreements. Different systems may use different measures and the metric may be adapted into different forms. We present a concrete form of our general metric in this section and will use it throughout the rest of the paper to demonstrate the feasibility, costs, and benefits of our metric.

Different feedback systems differ from each other in terms of the feedback mechanism and the feedback measure. In the first implementation of PeerTrust, we choose to use an interaction based complaint system.

With such a feedback system, if a peer receives a complaint from another peer after an interaction, it simply means the amount of satisfaction it gets through this interaction is 0. Otherwise, it gets 1. Let $C(u, v, t)$ denote the number of complaints a peer u receives from v up to transaction t. The amount of satisfaction $S(u, v, t)$ can be simply measured in terms of the number of interactions for which peer u does not receive a complaint. That is $I(u, v, t) - C(u, v, t)$.

Now let us consider the balancing factor of trust. It is obvious that peers cannot be blindly trusted for their feedbacks. Intuitively, a less trustworthy peer is more likely to file fake complaints to hide their own misbehavior and badmouth other peers. To make this problem tractable, in the first prototype of PeerTrust, we apply a simplifying assumption. We assume peers are rational in a game theoretic sense, i.e. trustworthy peers do not file fake complaints and untrustworthy peers file fake complaints when they misbehave during an interaction. Based on this assumption, complaints from a trustworthy peer can be trusted while complaints from an untrustworthy peer cannot be trusted. Therefore, it is reasonable to use the trust measure of a peer obtained from the collective evaluation of this peer in the network as the balance factor to offset the risk of its non-credible complaints. Thus, the balanced amount of satisfaction $S(u, v, t) * Cr(v, t)$ can be measured as $I(u, v, t) - C(u, v, t) * T(v, t)$ where $C(u, v, t) * T(v, t)$ denotes the credible complaints filed by v. The complaint-based trust metric can be defined as follows:

$$T(u, t) = 1 - \frac{\sum_{v \in P, v \neq u} C(u, v, t) * T(v, t)}{\sum_{v \in P, v \neq u} I(u, v, t)} \tag{4}$$

Note that the trust metric given above gives a value between 0 and 1. It determines how many complaints out of the total complaints a peer files against other peers will be counted. For example, complaints from a complete untrustworthy peer, i.e. who misbehaves during each interaction, will not be counted at all.

2.4 Discussions

We have presented the trust metric in its basic form and a concrete trust metric based on complaints and interactions. This section briefly discusses a few extensions we have introduced in the PeerTrust model to adapt to various commerce communities and to address some common problems observed [16] in supporting trust for P2P electronic communities. The detailed metrics are omitted due to space limitations. Interested readers may refer to [5] for details and a complete list of metrics.

Reacting to change in time
The basic metric evaluates a peer based on its consistent behavior and cannot react to behavioral changes of peers in a timely fashion. For example, a peer may join a network and behave reliably until it reaches a high reputation and then start committing fraud. Specifically, a peer may act completely trustworthy for the first X interactions (transactions) but start behaving completely untrustworthy for the next Y interactions. An easy extension of the metric in order to reflect such dynamic changes in time is to build in a sliding time window in the metric so that one can give more weight to the latest feedbacks and less weight to old feedbacks. The model is thus weighted toward current behavior and is able to reflect recent changes in time while still capturing the consistent behavior of the peers.

Incorporating Feedback Filters
We can also extend or customize the basic trust metric for different communities. As an example, if the community is

business savvy, we can incorporate the dollar amount of a transaction into the metric to weight the feedback. So the amount of satisfaction that is obtained through transactions that involve higher dollar amount of payment will carry a bigger weight than those that only involve smaller dollar amount. The total dollar amount of each transaction a peer has with other peers in the community can also be a determining factor when making trust decisions using the equation 3.

Evaluating trust from multiple context dimensions
Another extension to trust evaluation is to incorporate different context categories or dimensions into the basic trust metric with a weight function. Each of such categories is balanced by a weight value. For example for a buyer-seller community, peers can submit feedbacks about each other along different dimensions such as prompt shipment, product quality, information reliability, etc. and the final trust value is a weighted aggregation of feedback along these dimensions. An example application of this equation is to distinguish buyer transactions from seller transactions with different weighting. How to determine the weight function for all the parameters and how to integrate different extensions to the basic metric of trust evaluation need to be further investigated.

Overcoming Reentry and Entry Barrier
In order to prevent peers with bad reputation reentering the system with a new identity, a simple solution is to assign a default trust value that is fairly low, ideally lower than all the peers in the community, so the peers will have an incentive to keep their trust measure and keep on improving it instead of leaving the system and registering again to get a low default trust value. An obvious disadvantage is the entry barrier that new peers might be ignored completely because of lacking the feedbacks from past experiences. The challenge is balancing between ease of entry and meaningful trust scores. This is a topic of future research. A possible method is to use the mean or median of all peers' current trust values. Using a dynamically calculated initial trust value makes it harder to determine the starting trust score.

Creating incentives to rate
There are a number of remedies to the incentive problem. One way is to use reward schemes. Another is to build incentives into the reputation-based trust equation by adding a reward function. It is important to limit the amount of credits one can obtain by rating others in order to avoid the situations where peers may build a very high trust by simply rating others.

3 TRUST MANAGEMENT

The effectiveness of a trust model depends not only on the factors and the metric for computing trust measure, but also on implementation of the trust model in a P2P system. Typical issues in implementing a trust model include decentralized trust data management and trust computation execution. A decentralized trust management not only reduces bottleneck but also avoids a single point of failure. This section discusses the architecture and the design considerations in implementing the PeerTrust model in a decentralized P2P information system.

3.1 System Architecture

Figure 1 gives a sketch of the system architecture of PeerTrust. There is no central database. Trust data that are needed to compute the trust measure for peers are stored across the network in a distributed manner. If an individual database fails due to negligence or intentional corruption by some peers, it will be corrected by valid information in the rest of the community. In addition, the trust computation is executed in a dynamic and decentralized fashion at each peer. Instead of having a central server that computes each peer's trust value, a peer obtains another peer's trust data from the rest of the peers and computes the trust value of this peer on the fly. This allows peers to get an up-to-date evaluation of the peer by other peers.

The callout of the peer shows that each peer maintains a small database that stores a portion of the global trust data such as complaints filed. Each peer has a trust manager for submitting feedbacks, collecting feedbacks and computing trust measures. Each peer also has a data locator, a data placement and data location component for placement of trust data over multiple peers and managing the access to the trust data.

The data locator provides a P2P data location scheme for accessing and updating data in the network. Different applications may use different data placement and location schemes, which determine how and where the data can be inserted, updated, and accessed.

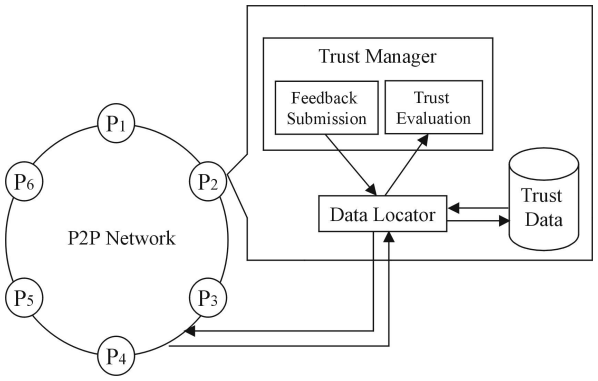

Figure 1: PeerTrust System Architecture

The trust manager has two main functions. First, it submits feedbacks to the network through the data locator, which will route the data to the appropriate peer for storage. Second, it is responsible for evaluating the trustworthiness of a particular peer. This task is performed in two steps. It first collects trust data about the target peer from the network through the data locator and then computes the trust value on the fly.

3.2 Trust Data Location

A number of P2P file sharing systems have emerged and each has its own data location scheme. Examples include Gnutella [2], Freenet [11], CAN [19], Chord [23], Pastry [21], and P-Grid [7]. Most of them are decentralized systems where there is no central server and data retrieval is self-organized by peers. Depending on the choice of a data location scheme, the implementation of the trust model may be a little different. Different schemes may also affect the overhead of the trust data management but should not affect the effectiveness of the trust metric. In the first version of the PeerTrust, we decide to use P-Grid primarily because we obtained the P-Grid source code.

P-Grid uses the approach of scalable replication of tree structures. Randomized algorithms based on local interactions among peers are used to partition the peers into a virtual binary search tree. By local interactions, peers successively partition the key space and retain routing information to other peers. Readers who are interested in the detail of the construction process of P-Grid may refer to [7]. Once the P-Grid is constructed, each peer maintains a small fragment of the data and maintains a routing table to other peers for the data it doesn't store locally.

Figure 2 shows a simple example of a PeerTrust network of 6 peers with constructed P-Grid. The callout at each peer shows the data keys each peer is responsible for and the routing table for those keys stored at other peers. The search keys are binary strings encoded from peer IDs. If $K_1K_2...K_n$ denotes the common prefix of the keys that a peer is responsible for, the peer keeps n rows in its routing table and the ith row indicates to which peer the query with a data key of a prefix $K_1K_2...\overline{K_i}$ will be routed. For example, P_4 is responsible for key 110, which means P_4 stores the trust data for P_6, including the number of complaints P_6 receives from other peers and the number of interactions P_6 has with other peers. It keeps routing entries for all the other data keys, i.e. the data keys that have a prefix 0 or 10.

When a peer receives a search or update request with a particular data key, it first determines if it is able to process this request locally. If yes, it will process the request with the data key; otherwise it will look up its routing table to find the matching prefix of the data key and route the request to the corresponding reference. Figure 2 shows an example of how a search request is processed in the PeerTrust network. Suppose that peer P_2 receives a query with key 110. P_2 cannot process this query. So it looks up its routing table and finds the row that has the common prefix with the query key 110, which is 1 in this case. According to the row 1, the query with key 110 is now routed to peer P_6. P_6 cannot serve this query locally and again looks up its routing table and finds the row that has the common prefix with the key 110, which is 11. P_6 now routes the query to P_4. P_4 can serve the query with key 110 locally and returns the trust data.

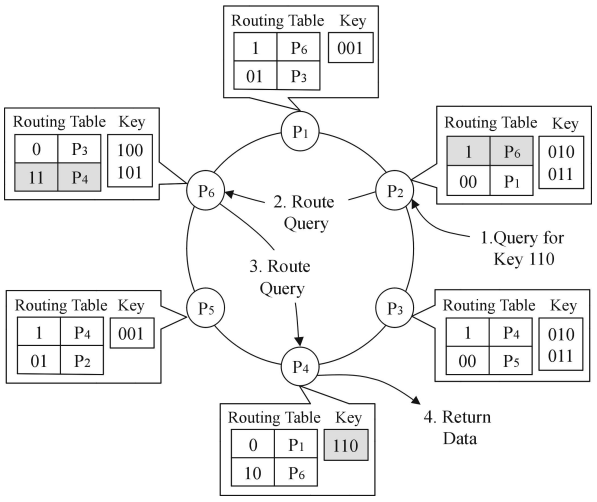

Figure 2: PeerTrust Data Location

For any given P2P data location scheme, there is a trust issue associated with it. Namely peers may misbehave by providing false data or random data when processing a search or update request for a data item it is responsible for. To address this issue, P-Grid can be configured to have multiple peers responsible for the same key. When a peer is searching for trust data about a particular peer, it simply issues the search request multiple times and finds multiple peers that are responsible for the same key, referred to as witnesses. The peer then combines the data from all witnesses and gets a consensus value by using a voting scheme [8] and taking into account the trust they put into the witnesses.

3.3 Trust Computation

The trust computation component is responsible for computing the trust measure based on the complaints a peer has received from other peers and the number of interactions. We propose two implementations of the trust value computation in this section. One is called dynamic computation, which uses the fresh trust data collected at runtime to compute the trust value. The other is called approximate computation, which uses cache to speed up the trust computation process.

Dynamic Computation
Assume the same notations that are used in Section 2.2. Let $\mathbf{A} = (a_{uv})$ be a square matrix of order N (recall N is the number of peers in the network P) with rows and columns corresponding to the peers and a_{uv} is computed as follows:

$$a_{uv} = \begin{cases} \frac{C(u,v,t)}{I(u,t)} & \text{if } I(u,t) \neq 0 \\ 0 & \text{if } I(u,t) = 0 \end{cases} \tag{5}$$

Let $\mathbf{T} = (t_u)$ be a column vector of the trust values of all peers up to time t where $t_u = T(u,t)$. Based on the metric in equation 4, we have:

$$\mathbf{T} = 1 - \mathbf{AT} \tag{6}$$

We can simply start with a default trust value vector. As we obtain new trust data (such as complaints) for each peer, we repeatedly compute the trust vector until it converges. When this is done, all trust values of the peers in the network will be available.

We can easily see that this computation is very expensive as it retrieves the trust data of all peers in the network even when a peer is only interested in evaluating the trustworthiness of a particular peer or a small subset of peers. To address

the high communication cost involved in dynamic computation, we propose approximate computation, a more cost efficient computation at each peer using a trust cache.

Approximate Computation
Each peer maintains a trust cache that keeps the trust values it has computed for other peers in the past. Cache size can be determined based on different factors such as size of the overlay network, available resources at each peer, and so on. When a peer w needs to evaluate the trustworthiness of another peer u, it retrieves the trust data of u. Instead of computing the trust value for the peers who filed complaints against u as the balance factor, w looks for their trust values in its cache. It then uses the cache value in the case of cache hit and only computes the value on the fly in the case of a cache miss or simply uses a default value. Once the peer w computes the trust value for u, it can again add the trust value of u to the cache or replace an old value in the cache. When the cache is full, it uses an LRU-like cache replacement policy to evict the least recently used data items from the cache.

Let $T_{cache}(v, t')$ denote an old trust value of v up to transaction t' in peer w's cache and $T_{default}$ denote a default trust value. The approximate computation conducted at peer w computes the trust value of u up to transaction t as follows:

$$T(u,t) = 1 - \frac{\sum_{v \in P, v \neq u} C(u,v,t) * T'(v,t)}{\sum_{v \in P, v \neq u} I(u,v,t)} \tag{7}$$

Where

$$T'(v,t) = \begin{cases} T_{cache}(v,t') & \text{cache hit} \\ T(v,t) \text{ or } T_{default} & \text{cache miss} \end{cases} \tag{8}$$

Our simulation results later show this approximate computation still produces effective trust evaluation results and significantly reduces the communication cost for the trust computation.

4 INITIAL EXPERIMENTS AND PRELIMINARY RESULTS

This section presents our initial experiment design, including a few experiment evaluation metrics, and reports the set of initial experimental results.

4.1 Evaluation Metrics

We first define trust evaluation accuracy as a metric to evaluate how well the trust model helps peers in making trust decisions. A trust evaluation is considered correct when a trustworthy peer is evaluated as trustworthy or an untrustworthy peer is evaluated as untrustworthy. In contrast, a trust evaluation is considered incorrect if a trustworthy peer is evaluated as untrustworthy or an untrustworthy peer is evaluated as trustworthy. For the first type of incorrect evaluations, the peer may miss an opportunity to interact with a trustworthy peer. For the second type of incorrect evaluations, the peer may end up interacting with an untrustworthy peer and coping with the risk of misbehavior from the other peer.

Let E denote the total number of evaluations peers make in the network over a time period. Among these evaluations, let E_{tt} denote the number of evaluations during which trustworthy peers are evaluated as trustworthy, let E_{tu} denote the number of evaluations during which trustworthy peers are evaluated as untrustworthy, let E_{uu} denote the number of evaluations during which untrustworthy peers are evaluated as untrustworthy, and let E_{ut} denote the number of evaluations during which untrustworthy peers are evaluated as trustworthy. The trust evaluation accuracy can be simply defined as the ratio of the correct evaluations over the total number of evaluations. That is:

$$Accuracy = \frac{E_{tt} + E_{uu}}{E} \tag{9}$$

Other metrics can be defined for different application domains with different requirements. For example, we can define a false negative rate in terms of detecting the untrustworthy peers as $FNR = \frac{E_{ut}}{E}$. This may be important for evaluating the trust metric for application domains where it is critical to avoid untrustworthy peers, such as electronic commerce applications.

As performance is always a concern with additional security layer, we also define a metric that measures the overhead a trust model introduces to the application. A good trust model should be effective and robust against the misbehavior of peers in the P2P network without introducing significant overhead to the application. As the communication cost for trust data lookup is the main overhead, we use the average number of messages among peers that are needed for each evaluation as the metric for this purpose.

4.2 Simulation Setting

We implemented a simulator in Mathematica 4.0 to evaluate our approach and demonstrate the importance of the three trust parameters we have identified as against the conventional approach in which only the first parameter, i.e. the amount of satisfaction, is used to measure the trustworthiness of a peer.

Our simulated P2P network consists of 128 peers, among which some are trustworthy and some are untrustworthy. The experiment proceeds as follows. Peers perform interactions with each other. During the interactions, trustworthy peers always perform satisfactory services and only file complaint against the other peer if the other peer performs poor service; untrustworthy peers perform poor services and file a fake complaint against the other party one out of four times and perform satisfactory services in other times. After 6400 interactions, i.e. an average of 100 interactions for each peer, 4 peers evaluate the trustworthiness of 100 randomly chosen peers from the remaining peers.

For the trust evaluation, P-Grid is used as the data location scheme to store and retrieve trust data about peers. We simulated the misbehavior of peers for processing queries, namely, untrustworthy peers provide random data for a query for which it is responsible one out of four times. During each evaluation, a peer issues a search request for the trust data 15 times and then uses the mean value computed from data returned by different witnesses. Both the dynamic computation and approximate computation are simulated for computing the trust value. A simple decision rule with a threshold 0.8 is used to decide whether a peer is trustworthy or not based on peer's trust value.

In order to see how our mechanism performs in different scenarios, we simulated random interactions and the two special scenarios listed in Section 2. Specifically, in random scenario, peers perform interactions with each other in a random pattern so each peer ends up having about the same number of interactions. In scenario 1, half randomly chosen peers perform 2 times more interactions than the other half. In scenario 2, half randomly chosen trustworthy peers perform interactions only with untrustworthy peers. The probability of scenario 2 happening in the real world may be very low. We use this setup to test the reliability of our approach in the worse case scenario.

We also simulated different peer communities. In each scenario, we vary the number of untrustworthy peers from 4 to 128.

For comparison purpose, we use the compliant-based trust method described in [8] as an example of the conventional method. The number of complaints is used as the trust measure. It only supports a binary trust output, i.e. whether the peer is trustworthy or not. We refer to this method as the complaint-only approach. Two implementations are proposed in [8] using P-Grid, referred to as simple and complex algorithm. The simple algorithm takes a majority decision from all the trust data returned by multiple witnesses. The complex algorithm checks the trustworthiness of the witnesses first and removes untrustworthy witnesses before taking a majority decision.

4.3 Simulation Results

First we compare the effectiveness of the two models. Figure 3 represents the trust evaluation accuracy of the two implementations of each model with respect to the percentage of untrustworthy peers in different interaction scenarios.

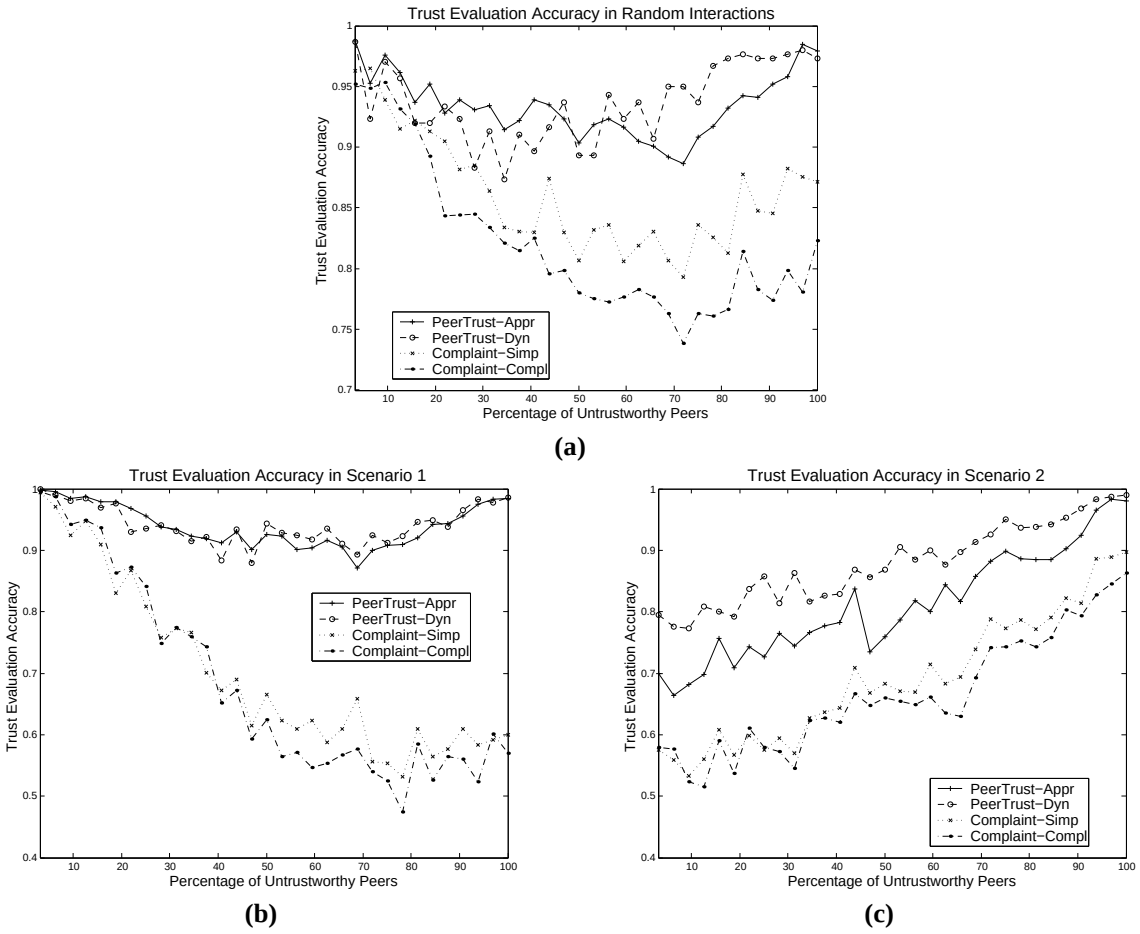

Figure 3: Trust Evaluation Accuracy

In the random scenario (Figure 3a), we can see the PeerTrust approximate computation is comparable to the dynamic computation in most cases. Interestingly, the complex implementation of complaint-only approach does not achieve higher trust evaluation accuracy than the simple implementation in most cases. Without an effective metric that takes into account all the necessary trust parameters, the complex implementation does not help by injecting trust considerations into the implementation.

We can make a few observations when comparing PeerTrust and the complaint-only approach. First, the two approaches perform almost equally well with a small percentage of untrustworthy peers. Second, when the percentage of untrustworthy peers increase, PeerTrust stays effective while the performance of complaint-only approach deteriorates. Last, when the number of untrustworthy peers is close to the total number of the peers, both approaches gain a little accuracy. These observations can be explained as follows. With a small percentage of untrustworthy peers, the complaint-only approach relies on there being a large number of trustworthy peers who offer honest statements to override the effect of the false statement provided by the untrustworthy peers and thus achieves a high accuracy. When the percentage of untrustworthy peers increases, the chances for trustworthy peers to interact with untrustworthy peers and receive fake complaints increase. The complaint-only approach uses the number of complaints only for assessing the trust of peers without taking into account the credibility of the complaints. Therefore, the trustworthy peers with fake complaints will likely be evaluated as untrustworthy incorrectly in the complaint-only approach. On the contrary, PeerTrust uses a balance factor of trust to offset the risk of fake complaints and thus is less sensitive to the misbehavior of untrustworthy peers. When the number of untrustworthy peers is close to the total number of peers, there are actually few trustworthy peers that can be evaluated incorrectly. Consequently, the trust accuracy of both approaches goes up at the end.

In scenario 1 (Figure 3b), there is also no significant difference between the two implementations of each model. However, PeerTrust performs significantly better than the complaint-only approach. The difference between the two models increases dramatically as the percentage of untrustworthy peers increases. This can be well explained by the lack of both number of interactions and the balance factor of trust as the trust parameters in the complaint-only approach.

In scenario 2 (Figure 3c), the approximate computation suffers a lower accuracy than the dynamic computation because of the approximation it uses when computing the balance factor. Both approaches are not performing well with a small percentage of untrustworthy peers. This can be explained as follows. As defined in our simulation, half of the trustworthy peers only interact with untrustworthy peers. As a result, with a small percentage of untrustworthy peers, more trustworthy peers will interact with untrustworthy peers only and will be likely evaluated incorrectly. However, PeerTrust is still consistently better than the complaint-only approach.

In summary, we verified that PeerTrust is much more stable and performs significantly better than the complaint-only approach in various scenarios regardless of the percentage of untrustworthy peers in the community. The complaint-only model is overly sensitive to the misuse behavior in the network and only performs well in limited cases.

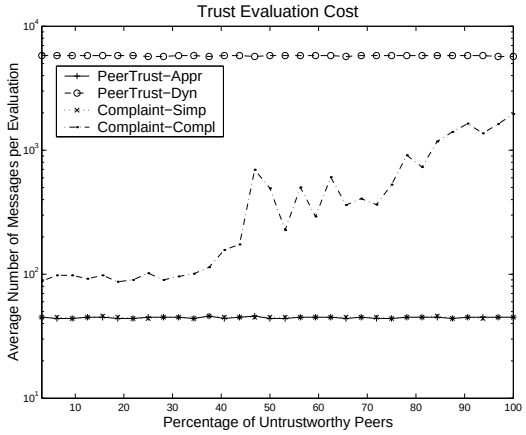

Figure 4: Trust Evaluation Cost

Next we compare the communication cost that the trust evaluations incur in the two models. Figure 4 represents the average number of messages exchanged for each evaluation of each implementation of the two models with respect to the percentage of untrustworthy peers. As the communication cost does not relate to the interaction scenarios, we only show the random scenario. It can be easily seen that the PeerTrust approximate computation and the complaint-only simple implementation are very efficient and the messages needed do not change with the percentage of untrustworthy peers. On the other hand, the complaint-only complex method has a significant overhead as the percentage of untrustworthy peers increases. The reason is that the method evaluates the trustworthiness of the witness recursively. When the percentage of untrustworthy peers increases, there are more complaints in the system and hence it causes more communications. As we expected, the cost of the dynamic computation of PeerTrust is very high. It is interesting to note, however, if a peer wishes to evaluate the trustworthiness of a large number of peers at the same time, the cost will be amortized.

5 RELATED WORK

Marsh [17] was one of the first who presented a formal trust model based on social models of interactions and social properties of trust. With a strong sociological foundation, the model is complex and is not suitable as an automatic trust mechanism for a P2P community.

A few other models were proposed that can be seen as an adaptation of Marsh's model to today's online environments and agent systems [27, 6, 26, 16]. However, they are not suitable for P2P environments for a number of reasons. First, the

models still tend to be complex with a lot of parameters and operations. In contrast, each peer only donates limited resources to the network in a P2P network and cannot afford to maintain complex information or perform complex computations locally. Second, they all rely on a central server or database or some kind of global knowledge to be maintained at each agent, which is also not suitable for a decentralized P2P environment. Last and most importantly, little attention was given for whether participant can conspire to provide false ratings. Most of them assume the feedback is always given honestly and with no bias and thus cannot adequately deal with malicious behaviors of agents.

There are also a few existing reputation systems as we discussed in Section 2.1. Most of these systems use the single factor of feedbacks as the reputation measure. Malaga [16] summarized some of the problems with these systems. All these systems use a centralized approach. They do not address trust management issues in decentralized P2P environment.

Another thread of work on trust models aims at building a high level trust model or framework for both trust between agent entities and trust in regards to the system itself [14, 13, 20]. In contrast, our work focuses on trust between entities and effective evaluation of the trustworthiness of peers based on the peer's past behaviors.

A number of research projects have engaged in P2P computing. Most of them have been focused on efficient resource location and load balancing and very few have addressed the need of trust in P2P systems or have incorporated trust into their resource placement and allocation algorithms.

The most relevant work is the trust method proposed by Aberer et al [8] as discussed above. The important contribution of their work is to emphasize the question of choosing the right model to assess trust and the question of obtaining the data needed to compute trust cannot be investigated in separation. They also utilize a complicated probability based threshold for trust decisions. Yet again their trust metric only takes into account the number of complaints and thus only works for limited scenarios and is very sensitive to the misbehavior of peers, as we have shown through our simulations.

Another relevant work is the P2PRep proposed by Cornelli et al [12]. It is a P2P protocol where servents can keep track, and share with others, information about the reputation of their peers. Their focus is to provide a protocol complementing existing P2P protocols, as demonstrated on top of Gnutella. There are no experimental results in the paper validating their approach.

Our approach differs from all these work in two significant ways. First, we identify three important trust parameters and argue that they are equally important for computing trust measure of peers in a decentralized P2P community. We also present the experimental results, verifying our argument. Second, we introduce a general trust metric that incorporate all three trust parameters in the trust computation and describe a complaint-based trust metric as a concrete form to illustrate the usefulness and benefit of our trust metric. We also validated our metric through a set of experiments, demonstrating the significant gain in accuracy when using our trust mechanism, compared with the complain-only methods.

6 CONCLUSIONS AND FUTURE WORK

We presented PeerTrust, a simple yet effective trust mechanism for building trust in P2P electronic communities. We identified three important trust parameters and developed a basic trust metric that combines these parameters for quantifying and comparing the trustworthiness of peers. We also discussed a number of extensions of the basic trust model to address special concerns in decentralized P2P electronic communities. We provided a brief discussion on implementation considerations of the trust model in a decentralized P2P environment. A set of initial experimental results have been reported to demonstrate the effectiveness, costs, and benefits of our approach in comparison with the complain-only approach.

Trust in P2P electronic communities is a new research area and there are many open questions and interesting issues to be addressed. Our research on PeerTrust continues along several directions. First, we are looking at ways to make the approach more robust against malicious behaviors, such as collusions among peers. We are also interested in combining trust management with intrusion detection to address concerns of sudden and malicious attacks. Another issue to address is how to uniquely identify peers over time and associate their histories with them. Without this ability, peers with bad reputations can simply shed their history by leaving and reentering the system and the only control is to assign a low reputation for new peers, which will also penalize innocent new peers. We are also continuing developing an extensive set of adaptive metrics, using the test bed we have to evaluate them, and finally testing the approach with real workload data.

Finally, we are working towards incorporating PeerTrust into two P2P applications that are currently under development in our research group, namely PeerCQ [4] and HyperBee [3]. PeerCQ is a P2P system for information monitoring on the web based on continual queries. HyperBee is a peer-to-peer search engine that looks to solve the problems that challenge today's search engines by utilizing peers to crawl the web page faster and cheaper. Both systems are faced with trust issues such as whether the peer can be trusted for performing a particular task, namely executing a continual query in PeerCQ and crawling the web page in HyperBee.

ACKNOWLEDGEMENTS

We would like to thank Karl Aberer and Zoran Despotovic for providing us the source code of P-Grid and their trust model for our comparison. This research is supported partially by a NSF ITR grant. The second author would like to acknowledge the partial support from a NSF CCR grant, a DOE SciDAC grant, and a DARPA ITO grant.

References

[1] Freenet. http://freenetproject.org.

[2] Gnutella. http://www.gnutella.com.

[3] HyperBee. http://www.hyperbee.com.

[4] PeerCQ. http://disl.cc.gatech.edu/PeerCQ.

[5] PeerTrust. http://disl.cc.gatech.edu/PeerTrust.

[6] A. Abdul-Rahman and S. Hailes. Supporting trust in virtual communities. In *33rd Annual Hawaii International Conference on System Sciences (HICSS-33)*, 2000.

[7] K. Aberer. P-grid: A self-organizing access structure for p2p information systems. In *Cooperative Information Systems, 9th International Conference, CoopIS 2001*, 2001.

[8] K. Aberer and Z. Despotovic. Managing trust in a peer-to-peer information system. In *2001 ACM CIKM International Conference on Information and Knowledge Management*, 2001.

[9] Y. Atif. Building trust in e-commerce. *IEEE Internet Computing*, 6(1), 2002.

[10] S. Ba and P. A. Pavlou. Evidence of the effect of trust building technology in electronic markets: Price premiums and buyer behavior. *MIS Quarterly*, 26(3), 2002.

[11] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Workshop on Design Issues in Anonymity and Unobservability*, 2000.

[12] F. Cornelli, E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati. Choosing reputable servents in a P2P network. In *Eleventh International World Wide Web Conference*, 2002.

[13] T. Dimitrakos and J. Bicarregui. Towards a framework for managing trust in e-services. In *Fourth International Conference on Electronic Commerce Research (ICECR-4)*, 2001.

[14] F. Egger. Towards a model of trust for e-commerce system design. In *CHI2000 Workshop Designing Interactive Systems for 1-to-1 E-commerce*, 2000.

[15] S. Ketchpel and H. García-Molina. Making trust explicit in distributed commerce transactions. In *16th International Conference on Distributed Computing Systems*, 1996.

[16] R. A. Malaga. Web-based reputation management systems: Problems and suggested solutions. *Electronic Commerce Research*, 1(4), 2001.

[17] S. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, Department of Mathematics and Computer Science, University of Stirling, 1994.

[18] D. H. McKnight and N. L. Chervany. The meanings of trust. Technical Report WP9604, University of Minnesota Management Information Systems Research Center, 1996.

[19] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content addressable network. In *ACM SIGCOMM*, 2001.

[20] S. Robles, S. Poslad, J. Borrell, and J. Bigham. A practical trust model for agent-oriented electronic business applications. In *Fourth International Conference on Electronic Commerce Research (ICECR-4)*, 2001.

[21] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *Middleware 2001, IFIP/ACM International Conference on Distributed Systems Platforms*, 2001.

[22] M. Shepherd, A. Dhonde, and C. Watters. Building trust for e-commerce: Collaborating label bureaus. In *Second International Symposium on Electronic Commerce, ISEC 2001*, 2001.

[23] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *ACM SIGCOMM*, 2001.

[24] V. Swarup and J. T. Fabrega. Trust: Benefits, models, and mechanisms. In *Secure Internet Programming: Security Issues for Mobile and Distributed Objects, Lecture Notes in Computer Science*. New York: Springer-Verlag, 1999.

[25] J. E. Youll. Peer to peer transactions in agent-mediated electronic commerce. Master's thesis, Massachusetts Institute of Technology, 2001.

[26] B. Yu and M. P. Singh. A social mechanism of reputation management in electronic communities. In *Cooperative Information Agents, 7th International Conference, CoopIS 2000*, 2000.

[27] G. Zacharia, A. Moukas, and P. Maes. Collaborative reputation mechanisms in electronic marketplaces. In *32nd Annual Hawaii International Conference on System Sciences (HICSS-32)*, 1999.