Gossip-based Reputation Aggregation for Unstructured Peer-to-Peer Networks*

Runfang Zhou and Kai Hwang

University of Southern California
Los Angeles, CA, 90089

Abstract: *Peer-to-Peer* (P2P) reputation systems are needed to evaluate the trustworthiness of participating peers and to combat selfish and malicious peer behaviors. The reputation system collects locally generated peer feedbacks and aggregates them to yield global reputation scores. Development of decentralized reputation system is in great demand for unstructured P2P networks since most P2P applications on the Internet are unstructured. In the absence of fast hashing and searching mechanisms, how to perform efficient reputation aggregation is a major challenge on unstructured P2P computing.

We propose a novel reputation aggregation scheme called *GossipTrust.* This system computes global reputation scores of all nodes concurrently. By resorting to a gossip protocol and leveraging the power nodes, GossipTrust is adapted to peer dynamics and robust to disturbance by malicious peers. Simulation experiments demonstrate the system as scalable, accurate, robust and fault-tolerant. These results prove the claimed advantages in low aggregation overhead, storage efficiency, and scoring accuracy in unstructured P2P networks. With minor modifications, the system is also applicable to structured P2P systems with projected better performance.

1. Introduction

Peer-to-peer (P2P) computing accounts for an large portion of the Internet traffic in recent years [13]. These include distributed file-sharing [6], [16], digital content delivery [3], [13], and Grid computing [7]. Despite the demand of robustness and scalability of P2P networks, the anonymous and dynamic nature of peer activities makes them very vulnerable to abuses by selfish and malicious peers [20]. As most P2P file-sharing networks, e.g. Gnutella, consist of autonomous peers with different self-interests [3] [8], their open and decentralized nature makes them extremely susceptible to malicious users spreading harmful content like viruses, fake files or just wasting others' resources.

To combat malicious peers and encourage resource sharing among peers, reputation management is essential for peers to assess the trustworthiness of others and to selectively interact with more reputable ones. Without an efficient reputation management facility, peers may hesitate to interact with unknown peers due to the concern of receiving corrupted or poisoned files or being exploited by malwares. Furthermore, identifying trustworthy peers is especially necessary in commercial P2P applications, such as P2P auctions, trusted content delivery, pay-per-transaction, and P2P service discovery. The mechanism through which online reputations are managed is extremely important for evolution and acceptance of these P2P services.

In a traditional reputation system, after a peer completes a transaction, e.g. downloading a music file, the peer will rate the other based on its experience in the transaction. The *reputation system* computes the global reputation score of a peer by aggregating the local rates (i.e. feedbacks) from those who have interacted with this peer. By making the global reputation scores publicly available, peers are able to make informed decisions about which peers to trust.

The field of P2P reputation systems is currently receiving a lot of attention as a means of helping peers avoid unreliable or malicious peers. In an open and decentralized P2P system, there is no centralized authority to maintain and distribute reputation data. Instead, P2P reputation systems calculate the global reputation scores by aggregating peer feedbacks in a distributed manner [12]. The most important issue involved in this process is the reputation aggregation from locally generated feedbacks to yield global reputation scores.

Most proposed reputation aggregation scheme, e.g., PowerTrust [22], EigenTrust [10] and PeerTrust [19] rely on the DHT mechanism to achieve scalability in aggregating and managing reputation data. However, the P2P architectures that are most prevalent in today's Internet are decentralized and unstructured, e.g. Gnutella, Kazaa and Freenet. To the best of our knowledge, there exists no specific reputation systems for unstructured P2P

* Presented in *IEEE International on Parallel and Distributed Processing Symposium* (IPDPS-2007), Long Beach, CA. March 27-29, 2007. This work was supported by NSF Grant ITR-0325409 at the University of Southern California. Corresponding author is Kai Hwang at kaihwang@usc.edu, Tel: 213 740 4470, and Fax: 213 740 4418.

1-4244-0910-1/07/$20.00 ©2007 IEEE

networks. How to perform efficient reputation aggregation is the major challenge in unstructured P2P networks due to lack of embedded fast hashing or searching mechanism.

In this paper we present a *GossipTrust system,* a scalable, robust and fault-tolerant reputation aggregation scheme for unstructured P2P networks. GossipTrust resorts to gossip protocols to aggregate global reputation scores. Each peer repeatedly contacts others at random, and exchanges reputation data with them.

The remaining parts of this paper are organized as follows. Section 2 reviews existing work on the development of reputation systems for both structured and unstructured P2P networks. Section 3 introduces the architecture of GossipTrust. The gossip protocol is described in Section 4. We present distributed gossip and aggregation algorithms in Section 5. Simulation experimental results are given in Section 6. Finally, we conclude with a summary of contributions and make suggestions for further research needed.

2. Related Work and Our Approach

Several reputation systems have been proposed to discourage maliciousness and motivate trustworthiness and cooperation in P2P networks. Xiong and Liu presented PeerTrust [19], which computes the trustworthiness of a given peer based on five factors. Srivatsa took a further step to improve the robustness of PeerTrust itself. They proposed methods to especially counter vulnerabilities in reputation management.

The EigenTrust algorithm [11] aggregates local scores by having peers perform a distributed calculation approaching the eigenvector of the trust matrix over the peers. We proposed the PowerTrust [22], which leverages the power law distribution of peer feedbacks to fast aggregate global reputations. All the above approaches are designed for DHT-based P2P networks.

Trust management was first studied by Abrerer and Despotovic 1] for unstructured P2P networks. Their approach is based on a decentralized storage method (P-Grid). The information provided by P-Grid is used to assess the probability that an agent will cheat in the future. This approach suffers from the fact trust is evaluated only by referrals from neighbors, not based on global information.

TrustMe [17] is a secure and anonymous underlying protocol for trust management. TrustMe uses a random assignment of reputation-holding peers and employs smart Public Key mechanisms to prevent the loss of anonymity. When a peer intends to query the global reputation of another peer, it will broadcast the query message; when a peer wants to report its feedbacks, it will broadcast the reports. This method will impose a lot of message in the network. Besides, when network size is very large, it will take a long time to disseminate peer reports and to get a peer's global reputation.

In [12], peer reputation scores are not aggregated from global information, but from very limited local information, i.e. a peer's local ratings or incorporating neighbors' rating. XRe [4] provides a protocol complementing current Gnutella protocol by allowing peers to keep track of and share information about the reputation of other peers and resources. No guarantees are given with respect to computational efficiency and scalability.

The idea of gossiping in cyberspace communication is very similar to gossiping to reach consensus in the human society. Gossip protocols were proposed for randomized communication [2], [9] and for aggregation of large amounts of distributed information [11]. These protocols do not rely on specific network topologies. They support the computation of aggregate functions like weighted sum, average value and maximum over large collection of distributed numeric values [14].

We are the very first attempt to use gossip protocol for fast global reputation aggregation in unstructured P2P networks. This P2P protocol does not require error recovery mechanisms, and thus enjoys an advantage in simplicity, while causes only moderate overhead compared to a deterministic communication protocol. GossipTrust is adaptable to peer dynamics and robust to disturbance of malicious peers by leveraging gossip protocols and power nodes. The power nodes are dynamically chosen after each reputation aggregation [22].

3. The GossipTrust Architecture

The major challenge in designing an efficient reputation aggregation scheme is to achieve a nice tradeoff between the following two key performance metrics.

***Computational complexity*:** The computation overhead, especially the time to aggregate the global reputation scores, is the major concern. Small computation overhead means significant traffic reduction in the system and less work for all peers involved.

***Accuracy of estimation*:** We would like our estimation of the global reputation score to be very close to the actual value. In this work, our goal is that the probability should be high for the estimated reputation score v' to be within $[(1-\varepsilon)v, (1+\varepsilon)v]$, where v is the actual reputation score and ε is the small relative error.

A very high level of accuracy can be achieved if one is willing to incur more complexity in terms of computation. Therefore, there is an inherent tradeoff between the computational complexities and accuracy of estimation.

This tradeoff can be exploited through the choice of various design parameters, as explained in Section 5. In addition, listed below are six key issues that we take into account in the design of GossipTrust.

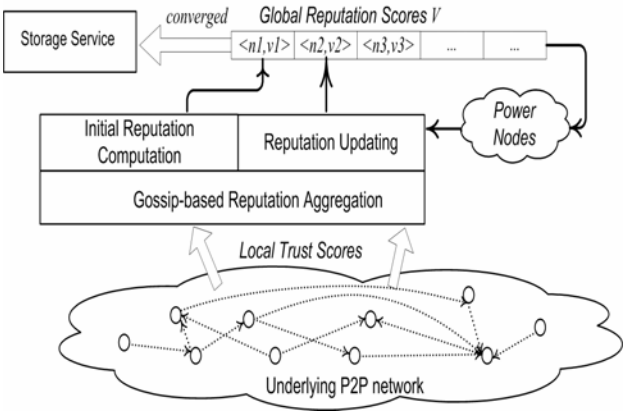
- ***System reliability*:** To help distinguish reputable from malicious peers, the system should calculate the reputation scores of peers as close to their real trustworthiness as possible.
- ***Simple to implement*:** The system should be simple to implement and maintain in a fully distributed manner.
- ***Light-weight communication***: The system should only consume limited computation and bandwidth resources for peer reputation monitory and evaluation.
- ***Adaptive to peer dynamics.*** Peer joins and leaves an open P2P network dynamically. The system should be adaptive and robust to peer dynamics.
- ***Robust to malicious peers***. The system should be robust to various attacks by both independent and collective malicious peers.
- ***System scalability***: The system should be able to scale to serve a large number of peers in term of accuracy, convergence rate, and extra overhead per peer.

We have developed a simulated *GossipTrust* reputation system at USC Internet and Grid Computing Laboratory. Figure 1(a) shows the architecture of the GossipTrust system running on a typical node N_i, where i=1,2,…,n. GossipTrust consists of three functional modules. The *Gossip-based Reputation Aggregation* module supports both *Initial Reputation Computation* and *Reputation Updating*. After each round of global reputation computation, GossipTrust will identify power nodes for the next round of reputation updating.

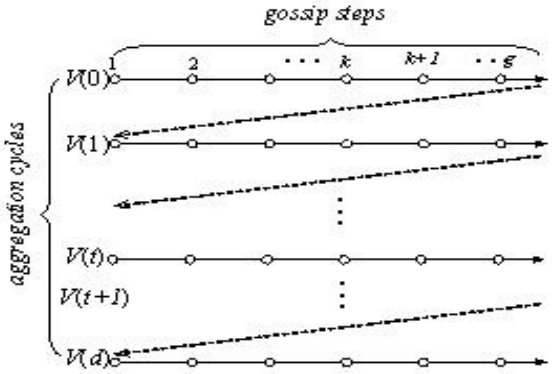
In this system, each node keeps a row vector of trust matrix S based on its outbound local trust scores. In addition, each node also maintains a global reputation vector $V(t)$ at aggregation cycle t. Internally, this vector is represented by a collection of <*node_id*, *score*> pairs. At the first aggregation cycle, $V(0)$ is initialized with equal global reputation scores, i.e. $v_i(0)=1/n$, i=1,2,…,n.

To compute the successive reputation vectors, GossipTrust uses a gossip-based protocol to perform the matrix-vector computation. Gossiping supports light-weight communications among nodes during the aggregation process. In GossipTrust, each aggregation cycle consists of several gossip steps as shown in Fig.1(b). In a gossip step, each node receives reputation vectors from others, selectively integrates the vectors with its current reputation vector, and then sends the updated one to a random node in the network.

This gossiping process continues until the gossiped scores converge in g steps, where g is determined by a set *gossiping error threshold* ε. After the convergence of gossip steps, GossipTrust continues the next aggregation cycle until the global reputation vectors converge in d cycles, where d is determined by the *aggregation error threshold* δ. Figure 1(b) illustrates the process of reputation aggregation in GossipTrust.



(a) GossipTrust system architecture



b) Global reputation aggregation | through iterative gossip steps

Figure 1. The GossipTrust architecture and control flow among functional modules

4. Gossip-based Reputation Aggregation

In this section, we first characterize the global reputation aggregation problem. Then we present our gossip protocol for distributed reputation aggregation..

4.1 Global Reputation Aggregation

In a P2P network of n nodes, each node evaluates the trustworthiness of other nodes with *local trust scores* after conducting a P2P transaction, such as a file download. Consider a *trust matrix* $R=(r_{ij})$, $1\le i, j \le n$, where r_{ij} is the local score issued by node i for node j. If there is no feedback from node i to j, r_{ij} is set to 0. For global reputation aggregation, each node must normalize all local

scores issued by itself. The *normalized local score* s_{ij} is defined as follows:

$$s_{ij} = r_{ij} / \sum_j r_{ij} \quad \textbf{(1)}$$

Then we have a *normalized trust matrix* $S = (s_{ij})$. Note that $0 \le s_{ij} \le 1$ and each row sum $\sum_{j=1}^{n} s_{ij} = 1$ for all rows $i = 1, 2, \ldots, n$. In other words, the normalized trust matrix S is a stochastic matrix, in which all entries are fractions and all row entries add up to be 1.

Let $v_i(t)$ be the *global reputation score* of node i at *aggregation cycle t,* where $i = 1, 2, \ldots, n$ and $t = 0, 1, 2, \ldots, d$ for d cycles. The global scores of all nodes form a *normalized reputation vector* with n components $V(t) = \{v_i(t)\}^T$, where $\sum_i v_i(t) = 1$. The iterative method specified below calculates the $V(t)$ at cycle t. Let $V(0)$ be the initial reputation vector value. For all iterative cycles $t = 1, 2, \ldots, d$, we generate successive reputation vectors, recursively, by performing the following matrix-vector computations:

$$V(t+1) = S^T \times V(t) \quad \textbf{(2)}$$

Initially, all nodes are equally trusted, i.e. $v_i(0) = 1/n$, where $i = 1,2,\ldots,n$. The iterative computation in Eq.(2) continues until the average relative error between $V(d)$ and $V(d+1)$ is lower than δ for a given *aggregation error threshold* δ at the last cycle d. We have proved in [22] that $d \le \lceil \log_b \delta \rceil$ with $b = \lambda_2 / \lambda_1$, where λ_1 and λ_2 are the largest and second largest eigenvalues of the trust matrix S. The convergence threshold δ is often predefined by system designers. In other words, after d cycles, the global reputation vector converges to the eigenvector of trust matrix S.

This recursive process is motivated by Markov random walk among nodes, which is widely used in ranking web pages. Consider a random surfer hopping from nodes to nodes to search for a reputable node. At each surfing step, the surfer selects a neighbor according to the current distribution of local trust scores. The stationary distribution of the Markov chain is the converged global reputation vector.

Both EigenTrust and PowerTrust have developed scalable algorithms to calculate the global reputation vector $V(t)$ for DHT-based P2P networks. In this paper, we propose a GossipTrust system for global reputation aggregation in unstructured P2P networks. The new scheme is fully distributed without using any topological structure among the nodes. It is also proven fast in convergence rate and secure in dynamic peer participations.

4.2 Gossip Aggregation Protocol

Gossiping is done iteratively in a small number steps. We reserve the index k to indicate the gossip step. According to Kempe, et al [11], k is upper by a final step $g = O(\log_2 n)$. We use index t to refer to discrete times for aggregation cycles. The upper bound for t is d iterations specified in section 3.1. Associated with each peer node i is a *gossip pair* $\{x_i(k), w_i(k)\}$ at each gossip step k.

At time t, we have the *weighted score* $x_i(t) = s_{ij} \times v_i(t)$ as the local score s_{ij} weighted by the global score $v_i(t)$ of node i. The $w_i(k)$ is called the *consensus factor* of node i at step k. During each gossip step, every node i executes two computing threads: One thread sends the halved gossip pair $\{\frac{1}{2} x_i(k), \frac{1}{2} w_i(k)\}$ to itself (node i) and to a randomly selected node in the network. Another thread receives the halved pairs from other nodes and computes the updated $x_i(k+1)$ and $w_i(k+1)$ as follows, where r refers the index of remote nodes sending halved gossip pairs in step k:

$$x_i(k+1) = \Sigma_r \tfrac{1}{2} x_r(k) \quad \textbf{(3)}$$

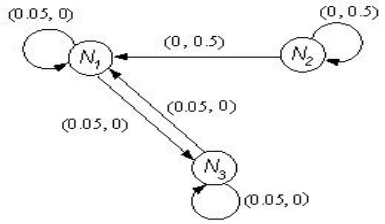
$$w_i(k+1) = \Sigma_r \tfrac{1}{2} w_r(k) \quad \textbf{(4)}$$

This process continues until the *consensus values* $\beta_i(k) = x_i(k)/w_i(k)$ agree on all nodes $i = 1, 2,\ldots, n$. The *global score* $v_j(t+1)$ is thus generated as follows on all n nodes at the final step g.

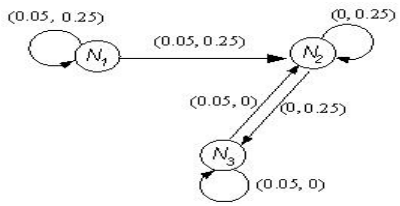
$$v_i(t+1) = x_i(g)/w_i(g) = \beta_i(g) \quad \textbf{(5)}$$

The above gossiping process is best illustrated by a small example in Fig.2. Consider a P2P network with three nodes. At time t, the global scores are given: $v_1(t) = 1/2$, $v_2(t) = 1/3$, and $v_3(t) = 1/6$. Given also normalized local score $s_{12} = 0.2$, $s_{12} = 0$, and $s_{32} = 0.6$. By Eq.(1), the updated global score of node $N2$ is calculated as:

$$\begin{aligned} v_2(t+1) &= v_1(t) \times 0.2 + v_2(t) \times 0 + v_3(t) \times 0.6 \\ &= (1/2) \times 0.2 + (1/6) \times 0.6 = 0.2 \end{aligned} \quad \textbf{(6)}$$



(a) First gossip step (k = 1)



(b) Second gossip step k = 2

Figure 2. Halved score sharing among 3 nodes in two gossiping steps to aggregate the global scores on all nodes, concurrently

We use Table 1 to illustrate the gossiping procedure in Fig.2. The end purpose is to generate the global score $v_2(t+1) = 0.2$ at all 3 nodes in 2 steps. In general, gossip protocols are used to calculate any aggregate function such as *sum, maximum,* or *average* of the numeric values distributed over many nodes. Here, we concentrate on the gossiped calculation of the global score of node *N2*. Initially at step 0, we thus assume $w_2(0) = 1$ and $w_1(0) = w_3(0) = 0$. The initial weighted scores $x_1(0) = (1/2)\times 0.2 = 0.1$, $x_2(0) = (1/3)\times 0 = 0$, and $x_3(0) = (1/6)\times 0.6 = 0.1$.

At the first gossip step, as shown in Fig.2(a), *N1* sends the pair $(\frac{1}{2}x_1(0), \frac{1}{2}w_1(0)) = (0.05, 0)$ to *N1* and to a randomly chosen node *N3* The node *N2* sends the pair (0, 0.5) to *N2* and a random node *N1*. Node *N3* sends the pair (0.05, 0) to *N3* and a random node *N1*. Then *N1* updates $x_1(1) = 0+0.05+0.05=0.1$ and updates $w_1(1) = 0+0.5+0 =0.5$. After first gossip step, *N1* has the pair $\{x_1(1), w_1(1)\}= (0.1, 0.5)$. The gossiped score $x_1/w_1 = 0.2$ on *N1*. Similarly, nodes *N2* and *N3* go through the same gossiping process to produce $x_2/w_2 =0$ and $x_3/w_3 = \infty$.

Table 1 Gossiped Scores Aggregated at Successive Steps on All Nodes in Fig.2, Concurrently

Node *N1*	$x_1(k)$	$w_1(k)$	$\beta_1(k) = x_1(k)/w_1(k)$
Step 1	0.1	0.5	0.2
Step 2	0.05	0.25	0.2
Node *N2*	$x_2(k)$	$w_2(k)$	$\beta_2(k) = x_2(k)/w_2(k)$
Step 1	0	0	0
Step 2	0.1	0.1	0.1
Node *N3*	$x_3(k)$	$w_3(k)$	$\beta_3(k) = x_3(k)/w_3(k)$
Step 1	0.1	0.1	0.1
Step 2	0.05	0.05	0.05

Figure 2(b) illustrates halved score sharing by the same gossiping process in step 2. After step 2, we have reached the consensus that $x_1/w_1 = x_2/w_2 = x_3/w_3 = 0.2$. Thus, we accept the updated global score for node *N2* as $v_2(t+1) = x_1/w_1 = x_2/w_2 = x_3/w_3 = 0.2$, which agree with the dot product calculation in Eq.(5). Thus, gossiped scores are equalized in all 3 nodes at the process end. Suppose we extend the gossiping process to another step, we will see no more changes in the consensus values $x_1/w_1 = x_2/w_2 = x_3/w_3 = 0.2$. Lacking centralized control, the consensus must be determined on distributed nodes locally. This last step determines the global consensus.

5. Distributed Aggregation in GossipTrust

The iterative method in Eq.(2) specifies global reputation aggregation in each cycle. Mathematically, we need to compute the weighted sum of all local scores s_{ij} for each peer $j= 1, 2, ...,n$ in Algorithm 1, where the normalized global scores $\{v_i(t-1)\}$ are the weights applied. The numerical computation in Eq.(6) for the *3*-node network example in Fig.2 is generalized for a general P2P network having n nodes indexed by $j = 1,2,\ldots, n$.

$$v_j(t) = \sum_{i=1}^{n} v_i(t-1) \times s_{ij} \qquad (7)$$

The above gossip protocol is generalized by the following Algorithm 1 for an n-node P2P network. The procedure shows how to perform the gossiping operation in g steps to reach the consensus on all nodes. Algorithm 1 is thus executed on all nodes, concurrently, even we specify the protocol only one node below.

Algorithm 1: Gossip Protocol to Compute Any Single Peer Score

1: **INPUT:** local score s_{ij}, global score $v_i(t-1)$ at time $t-1$, where $i = 1,2,\ldots,n$ and gossip threshold ε
2: **OUTPUT:** global reputation score $v_j(t)$ of node j at time t
3: **forall** $i = 1, 2, \ldots.n$ **do** { simultaneously }
4: $x_i \leftarrow s_{ij}\times v_i(t-1)$ { initialize weighted score x_i }
5: **if** $(i == j)$, set $w_i \leftarrow 1$, **else** $w_i = 0$ { initialize consensus factor w_i }
6 $k \leftarrow 0$ *{ initialize gossip step k}*
7: **repeat**
8: $u \leftarrow x_i/w_i$ { save previous score before convergence }
9: let $\{(x_r, w_r)\}$ be all gossip pairs sent to i in previous step
10: $x_i \leftarrow \sum_r x_r$, $w_i \leftarrow \sum_r w_r$ { update x_i and w_i }
11: choose a random node q
12: send the pair $(\frac{1}{2} x_i, \frac{1}{2} w_i)$ to node q and node i itself
13: $k \leftarrow k+1$ { increase gossip step by 1}
14: **until** $|x_i/w_i - u| \le \varepsilon$ { ε is a preset error tolerance or the gossip threshold}
15: **endfor**
16: **output** $v_j(t) \leftarrow x_i/w_i$

In GossipTrust, every node has a unique identifier and keeps a global reputation vector. The global-scale *gossip-based reputation aggregation* is specified in Algorithm 2. During each aggregation cycle t, each vector element is internally represented by a triplet (x_{id}, id, w_{id}), where id is a node identifier, x_{id} is the weighted score defined in Eq.(3) and w_{id} is the consensus factor defined in Eq.(4).

We aggregate all triplets in the reputation vector, concurrently. During each gossip step, every node i sends its reputation vector to a randomly chosen node, which can be a neighbor node or any other node. Upon receiving the global reputation vectors from others, node i updates x_{id} and w_{id} for every triplet in the reputation vector.

At the end of an aggregation cycle t, the gossip process converges to an equalized gossiped score x_{id}/w_{id} on all nodes. Node i checks the difference between the current global reputation vector and the one from aggregation cycle t-1. If the difference is larger than a pre-defined aggregation error threshold δ, node i will enter the next aggregation cycle t+1. Otherwise, the global reputation vector has converged, node i will replace the triplet $<x_j, j, w_j>$ with the pair $<v_j, j>$, where $v_j = \beta_j = x_j / w_j$ is the converged global score of node j.

Algorithm 2: Aggregation to Update All Peer Scores Concurrently

INPUT: local trust matrix $S=(s_{ij})$ and tolerable aggregation error δ
OUTPUT: Converged global reputation vector $V(t)$
forall $i = 1, 2, \ldots n$ **do** { Concurrently on n nodes }
  $t \leftarrow 0$ {initialize the aggregation cycle}
  Repeat
    forall local score s_{ij} **do**
      { initialize weighted score and consensus factor }
      if $(t == 0)$ **then** $x_j \leftarrow s_{ij} / n$
      else $x_j \leftarrow s_{ij} \times v_i(t\text{-}1)$ where $v_i(t\text{-}1)$ is node i's reputation at time instance $t\text{-}1$
      if $(j == i)$ **then** $w_j \leftarrow 1$ **else** $w_j \leftarrow 0$
      add the triplet $<x_j, j, w_j>$ to global reputation vector $V(t)$
    endfor
    repeat
      let $\{ V_r \}$ be all received vectors from previous gossip step
      forall j in V_r **do**
        $x_j \leftarrow \sum_r x_j$, $w_j \leftarrow \sum_r w_j$,
        update the triplet $<x_j, j, w_j>$ in $V(t)$
      endforall
      choose another node q randomly
      send ½ $V(t)$ to node q and to node i itself
    until all n gossiped scores $\{ \beta_i = x_j / w_j \}$ are equalized and converged to $v_i(t)$.
    forall j=1, 2, …, n **do**
      $v_j \leftarrow x_j / w_j$ and replace every triplet $<x_j, j, w_j>$ with the updated pair $<v_j, j>$
    endfor
    $t \leftarrow t + 1$ {increase time instance by 1}
  until $|V(t) - V(t\text{-}1)| < \delta$ {Test with tolerable convergence threshold}
endfor
output The updated global scores $V(t) = \{v_1(t), v_2(t), \ldots, v_n(t)\}$

6. Simulation Experimental Results

In this section, we evaluate the performance of the GossipTrust reputation aggregation scheme by analyzing *reputation convergence overhead*, *gossip propagation error* under different design parameters and *reputation aggregation error* with respect to different threat models. We assess the query success rate of using GossipTrust in simulated P2P file sharing applications.

6.1 Simulation Setup

We evaluate GossipTrust using our own discrete event driven simulator. In our experiments, we construct a Gnutella-like flat unstructured network initially consisting of 1000 nodes. The number of feedbacks every node issued is power law distributed. Initially the maximum feedback amount d_{max} is 200 and the average feedback amount d_{avg} is 20. This distribution better models feedback distribution in existing reputation systems [21], [22].

We choose greedy factor α = 0.15 as a default value. The system selects up to 1% of the total number of nodes as the power nodes. The base setting such as simulation parameters and default values used in most of the experiments are summarized in Table 2. δ and ε represent the threshold for global reputation convergence and threshold for gossip protocol convergence respectively.

We study two kinds of malicious behaviors, namely, independent setting and collusive setting. In independent setting, malicious peers cheat during transactions and issue dishonest feedbacks to others. They rate the peers who provide good service very low and rate those who provide bad service very high. In collusive setting, malicious peers collaborate with each other to boost up their own ratings. They may rate the peers in their collusion group very high and rate outsiders very low.

The simulated experiments were run on a dual-processor Dell server and the operation system installed on this machine is Linux with kernel 2.6.9. Each data point reported in the following sections represents the average of at least 10 simulation runs with different seeds.

Table 2. Parameters and Default Values used

Parameter	Basic Definition	Default Value
n	No.of peers in P2P network	1000
α	Greedy factor of a peer	0.15
d_{max}	Max. peer feedback amount	200
d_{avg}	Average peer feedback amount	20
γ	Percentage of malicious peers	10%
q	Max. No. of power nodes	1%
δ	Global aggregation threshold	10^{-3}
ε	Gossip error threshold	10^{-4}

6.2 Gossip Overhead vs. Threshold Applied

The objective of this set of experiments is to evaluate computational efficiency and scalability of GossipTrust. There are two convergence processes in GossipTrust: one is the convergence of the reputation computation round and another is during every aggregation cycle, the convergence of gossip protocol to aggregate the weighted sum. Figure 3 shows the effects of various gossiping error threshold ε and network size n on gossip steps.

We measure both the number of reputation convergence cycles and the number of gossip aggregation iterations. Our studies prove that GossipTrust only runs a small number of aggregation cycles before the reputation scores converge, given an arbitrary trust matrix and a fixed greedy factor $\alpha > 0.1$. The larger the number of gossip steps, the higher is the convergence overhead.

The convergence overhead increases with the decrease of gossiping error threshold and growth of network size. When the gossiping error threshold ε is very small, as $\varepsilon < 10^{-4}$, the gossiping error threshold dominates the convergence overhead, regardless of the network size. While when ε is large, as $\varepsilon > 10^{-2}$, the network size dominates the convergence overhead. With a fixed small ε, the convergence overhead remains closes for different network sizes, which means that GossipTrust is able to scale well when reputation system grows.

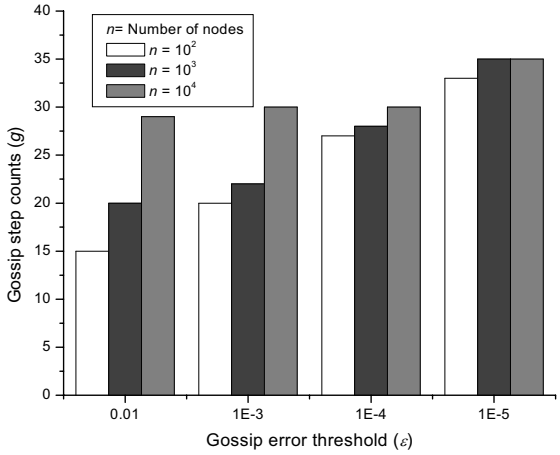


Figure 3 Gossip step counts of three P2P network configurations under various gossip error thresholds

6.3 Gosssip and Aggregation Error Analysis

The relative error incurred by gossip protocols in each aggregation cycle is propagated in the aggregation process. Table 3 reports both gossip error and aggregation error under three convergence settings for a P2P network of 1000 nodes. We consider the effects of varying both aggregation error threshold δ and gossiping error threshold ε. We assess the gossip error by measuring the relative error of the global reputation scores caused by gossip protocol. The aggregation error is measured as the distance between actual and estimated global reputation vectors.

Table 3 Gossip and Aggregation Errors under Three Convergence Threshold Settings for a 1000-Node P2P Network

ε	δ	Aggrega-tion Cycle	Gossip Step	Gossip Error	Aggregation Error
10^{-5}	10^{-4}	19	35	1×10^{-6}	1.6×10^{-4}
10^{-4}	10^{-3}	15	28	7×10^{-6}	7.3×10^{-4}
10^{-3}	10^{-2}	5	22	1.6×10^{-4}	3.8×10^{-3}

Tradeoffs exist between computational efficiency and accuracy: the smaller are the aggregation error threshold δ and gossiping error ε, the less the propagation of gossip error, but the larger the number of gossip steps and reputation aggregation cycles. Based on the result in Table 3, for a network of 1000 nodes, we choose $\varepsilon = 10^{-4}$

and $\delta = 10^{-3}$ to balance the tradeoff between the convergence overhead and computational accuracy

We evaluate the robustness of GossipTrust against malicious peer behaviors. The experiments were performed to compare non-collusive and collusive peer operations. In a non-collusive setting, malicious peers report dishonest trust scores independently. In a collusive setting, abusers collaborate with each other to boost up their own ratings.

They rate the peers in their collusion group very high and rate outsiders very low. The probability of a node behaving maliciously is inversely proportional to its global reputation, because a node providing corrupted services is highly likely to issue dishonest local trust scores.

We compute below the *root-mean-square* (RMS) *error* E in aggregated global scores under different percentage of malicious peers in a P2P network. Lower RMS error implies the system is more robust to attacks by malicious peers. The RMS error is defined by:

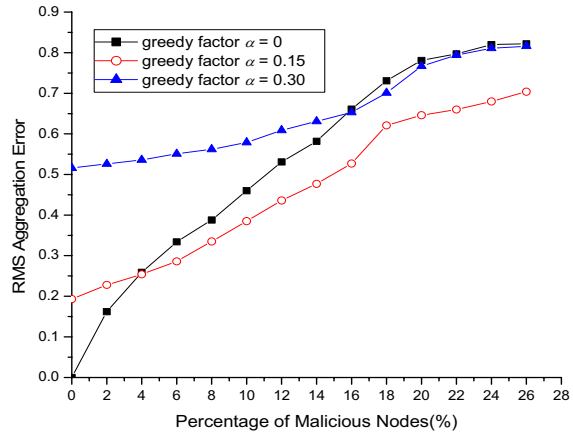
$$\textit{RMS aggregation error} \quad \boldsymbol{E} = \sqrt{\frac{\sum((v_i - u_i)/v_i)^2}{n}} \qquad \textbf{(8)}$$

where v_i and u_i are the calculated and gossiped global reputation scores of peer i, respectively.

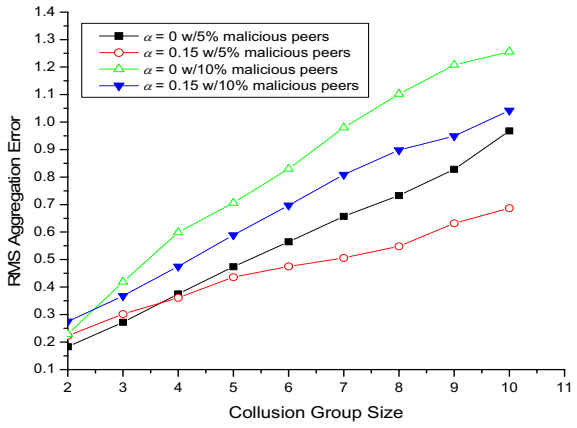
The *greedy factor* α indicates the eagerness for a peer to work with selected power nodes [22]. We plot in Fig.4(a) the RMS errors under different values of α and various percentages of independent malicious peers. By leveraging power nodes, we set the the greedy factor α = 0.15. This gives 20% less aggregation error than treating all peers equally with $\alpha = 0$.

However, increasing the greedy factor α to 0.3 does not lead to higher performance. This is because relying too much on the power nodes will miss the global view of the reputation data provided by majority nodes in the system. Therefore, setting α = 0.15 is indeed a very good choice. Figure 4(b) reports the RMS aggregation error under collusive peers working collectively to abuse the system.

The plot represents the effects of various *collusion group* sizes, defined by the number of malicious peers in a group. In all cases (5% and 10% collusive peers), leveraging the power nodes with a greedy factor α = 0.15 makes the system more robust against peer collusions. With 5% collusive peers, using power nodes has resulted in 30% less errors when collusion group size is greater than 6. The message being conveyed is proper use of power nodes are indeed effective to cope with peer collusions.



(a) Independent malicious peers



(b) Collusive malicious peers

Figure 4. Global aggregation errors from fake trust scores reported by malicious peers in a P2P network of 1,000 nodes

6.4 P2P File Sharing Experimental Results

We conducted extensive simulation experiments to measure the performance of using GossipTrust in P2P file-sharing applications. There are over 100,000 files simulated in these experiments. The number of copies of each file is determined by a Power-law distribution with a popularity rate φ = 1.2. Each peer is assigned with a number of files based on the Sarioiu distribution [12]. At each time step, a query is randomly generated at a peer and completely executed before the next query step. The query popularity reflects the file preferred by a peer.

We rank the queries according to their popularity. We use a power law distribution with a φ = 0.63 for queries ranked 1 to 250 and φ = 1.24 for lower-ranking queries. This distribution models the query popularity distribution in Gnutella. After a query for a file is issued and flooded over the entire P2P network, a list of nodes having this file is generated and the one with the highest global score is selected to download the file. The system

updates global reputation scores at all sites after 1,000 queries.

The *query success rate* is measured by the percentage of successful queries over the total number of queries issued. Every node has a rate to respond a query with inauthentic files. For simplicity, this rate is modeled inversely proportional to node's global reputation. We also consider the case of a *NoTrust system,* which randomly selects a node to download the desired file without considering node reputation. We plot in Fig.5 the results of using GossipTrust and NoTrust in simulated P2P file sharing applications.

In this experiment, the malicious peers issue unreliable scores and provide corrupted files. The performance of GossipTrust drops only slightly with the increase of the number of malicious peers, while performance of NoTrust drops sharply with more malicious peers. With the help of GossipTrust, even when the system has 20% malicious peers, it can still maintain around 80% query success rate. This experiment proves the effectiveness of using reputation-based selection scheme in unstructured P2P file sharing applications

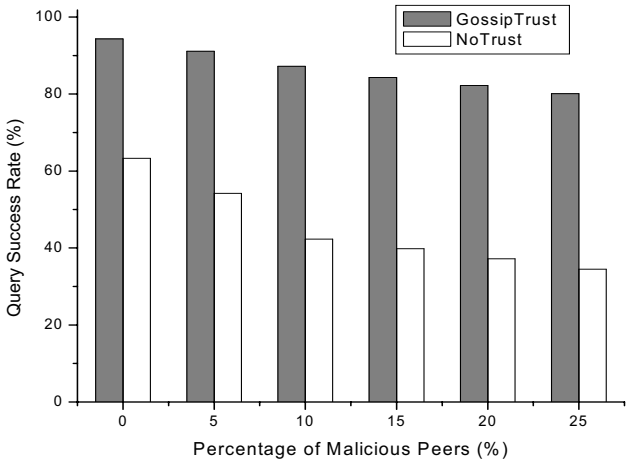


Figure 5 . Query success rate of simulated P2P file sharing applications on a 1000-node P2P network

7. Conclusions and Further Research

To our best knowledge, GossipTrust offers the very first attempt to extend the gossip protocol for reputation aggregation in P2P networks without any structured overlay support. GossipTrust is shown very fast in aggregating local trust scores into global reputation scores. The major innovations in GossipTrust development are summarized in three aspects: fast gossip-based aggregation algorithms, efficient reputation storage with Bloom filters, and secure communication with identity-based cryptography.

GossipTrust enables peers to compute global reputation scores in a fully distributed, secure, scalable and robust fashion. The simulation results show that the system scales well with the increase of network size. The system can also tolerates link failures and peer collusions. The benchmark experiments on P2P file-sharing applications demonstrate significant performance gains in using GossipTrust, compared with an unstructured P2P network without reputation services.

We have to point out that the GossipTrust system is not restricted to apply only in unstructured P2P systems exclusively. With minor modifications, the system can perform even better in a structured P2P system. The gossip steps and reputation aggregation process reported here can be further accelerated by the fast hashing and search mechanisms built in DHT-based overlay networks. A peer providing corrupted services is highly likely to issue dishonest reputation scores.

To probe further, we suggest to keep two kinds of reputation scores on each peer node: one to measure the *quality-of-service* (QoS) such as those performance measures reported here and another for *quality-of-feedback* (QoF) by participating peers. We suggest integrating these two scores together and address the tradeoffs between them in future research challenges.

Further research is also encouraged to apply reputation systems to enforce copyright protection in P2P systems. With the help of object reputation [18], a client can validate the authenticity of an object before initiating parallel file download from multiple peers. This opens up a meaningful direction to extend gossip-based systems for managing object reputations.

Acknowledgements: This work was fully supported by NSF ITR Grant ACI-0325409 at the Internet and Grid Research Laboratory, University of Southern California.

References:

[1] K. Aberer and Z. Despotovic, "Managing Trust in a Peer-2-Peer Information System", *Tenth International Conference on Information and Knowledge Management,* New York, 2001

[2] S. Boyd, A. Ghosh, B. Prabhakar, D. Shah, "Randomized Gossip Algorithms", *IEEE Trans. on Information Theory*, June 2006, 52(6):2508-2530.

[3] N. Christin, A.S.Weigend, and J. Chuang, "Content Availability, Pollution and Poisoning in File Sharing Peer-to-Peer Networks", *ACM Conf. on E-Commerce*, Vancouver, June 2005.

[4] E. Damiani, S. Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A Reputation-based Approach for Choosing Reliable Resources in Peer-to-Peer

Networks", *ACM Symposium on Computer Commmunication Security,* 2002

[5] C. Gkantsidis, M. Mihail, and A. Saberi, "Conductance and Congestion in Power Law Graphs", *ACM/IEEE SIGMETRICS,* San Diego, June. 2003

[6] K.Gummadi, R.Dunn, R. Dunn, "Measurement, Modeling and Analysis of a Peer-to-Peer File-Sharing Workload", *Proceedings of the 19th ACM Symposium on Operating Systems Principles*, Bolton Landing, NY, 2003.

[7] J. Hu and R. Klefstad, "Decentralized Load Balancing on Unsructured Peer-2-Peer Computing Grids", *IEEE Int'l Symp. on Network Computing Applicatons (NCA'06)*, Boston, July, 2006

[8] D. Hughes, G. Coulson, and J. Walkerdine, "Free Riding on Gnutella Revisited: The Bell Tolls?", *IEEE Distributed Systems Online*, *Vol. 6,* June 2005

[9] M. Jelasity, A. Montresor and O.Babaoglu, "Gossip-Based Aggregation in Large Dynamic Networks", *ACM Trans. on Computer Systems*, *Vol.23, No.3*, August 2005

[10] S. Kamvar, M. Schlosser, and H. Garcia-Molina, "The Eigentrust Algorithm for Reputation Management in P2P Networks", *ACM WWW'03*, Budapest, Hungary, May 2003

[11] D. Kempe, A. Dobra and J. Gehrke, "Gossip-Based Computation of Aggregate Information", *IEEE Symp. on Foundations of Computer Science*, Cambridge, MA, Oct.2003

[12] S. Marti and H. Garcia-Molina, "Limited Reputation Sharing in P2P Systems", *Proc. of ACM Conference on Electronic Commerce,* New York, May 2004

[13] J. Meserve, "P2P Traffic Still Dominates the 'Net", *Network World*, 2005.

[14] S. Nandy, L. Carter and J. Ferrante, "GUARD: Gossip Used for Autonomous Resource Detection", *19th Int'l Parallel &Distributed Processing Symposium*, Colorado, Apr. 2005.

[15] R. L. Page, S.Brin and T. Winograd, "the Pagerank Citation Ranking: Bringing Order to the Web", *Technical report*, Stanford Digital Library Technologies Project, 1998

[16] D. Qiu and R.Srikant, "Modeling and Performance Analysis of Bit Torrent-Like Peer-to-Peer Networks", *Sigcomm 2004*, Portland, Aug-Sep, 2004

[17] A. Singh and L. Liu, "TrustMe: Anonymous Management of Trust Relationships in Decentralized P2P Systems", *IEEE Intl. Conf. on Peer-to-Peer Computing*, Sep. 2003

[18] K. Walsh and E. Sirer, "Experience with an Object Reputation System for Peer-to-Peer File-sharing", *NSDI' Symp.on Networked Systems Design & Implementation*, San Jose, May 8-10, 2006

[19] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-based Trust for Peer-to-Peer Electronic Communities", *IEEE Trans. Knowledge and Data Engineering,* Vol.16, No.7, 2004, pp. 843-857

[20] B. Yang, T.Condie, S. Kamvar and H. Garcia-Molina, "Non-Cooperation in Competitive P2P Networks", *Proceedings of the 25th IEEE Int'l Conference on Distributed Computing Systems* (*ICDCS'05*), Columbus, Ohio, 2005

[21] M. Yang, Z. Zhang, X. Li and Y. Dai, "An Empirical Study of Free-Riding Behavior in the Maze P2P File-Sharing System", *Proc. of IPTPS*, Ithaca, Feb, 2005.

[22] R. Zhou and K. Hwang, "PowerTrust: A Robust and Scalable Reputation System for Trusted P2P Computing", *IEEE Trans. on Parallel and Distributed Systems*, accepted to appear 2007.

Biographical Sketches

Runfang Zhou is currently pursuing Ph.D. degree in Computer Science at the University of Southern California. Her research activities cover Peer-to-Peer reputation systems, overlay network design, web services performance improvement, and trust and secure collaboration in Grid computing. She can be reached at: rzhou@usc.edu.

Kai Hwang is a Professor of Electrical Engineering and Computer Science and Director of Internet and Grid Research Laboratory, University of Southern California. He received the Ph.D. from University of California, Berkeley in 1972. An IEEE Fellow, he specializes in computer architecture, parallel processing, Grid and cluster computing, and distributed computing systems. He has published over 200 technical papers and 8 books in these areas. His research group at USC has developed security-binding techniques, peer reputation systems, and distributed defense systems against worms and DDoS attacks for trusted Grid, P2P, and Internet computing. Contact him at kaihwang@usc.edu.