A Tunable Mechanism for Identifying Trusted Nodes in Large Scale Distributed Networks

Joydeep Chandra*†, Ingo Scholtes*, Niloy Ganguly† and Frank Schweitzer*
*Chair of Systems Design
ETH Zurich
Kreuzplatz 5, CH-8032 Zurich, Switzerland
†Department of Computer Sc. & Engg.,
Indian Institute of Technology, Kharagpur, India

***Abstract*—In this paper, we propose a simple randomized protocol for identifying trusted nodes based on personalized trust in large scale distributed networks. The problem of identifying trusted nodes, based on personalized trust, in a large network setting stems from the huge computation and message overhead involved in exhaustively calculating and propagating the trust estimates by the remote nodes. However, in any practical scenario, nodes generally communicate with a small subset of nodes and thus exhaustively estimating the trust of all the nodes can lead to huge resource consumption. In contrast, our mechanism can be tuned to locate a desired subset of trusted nodes, based on the allowable overhead, with respect to a particular user. The mechanism is based on a simple exchange of random walk messages and nodes counting the number of times they are being hit by random walkers of nodes in their neighborhood. Simulation results to analyze the effectiveness of the algorithm show that using the proposed algorithm, nodes identify the top trusted nodes in the network with a very high probability by exploring only around 45% of the total nodes, and generating nearly 90% less overhead as compared to an exhaustive trust estimation mechanism, named TrustWebRank. Finally, we provide a measure of the global trustworthiness of a node; simulation results indicate that the measures generated using our mechanism differ by only around 0.6% as compared to TrustWebRank.**

***Keywords*-Trust; Distributed Systems; EigenTrust; Random Walks;**

I. Introduction

The Internet has witnessed a huge growth not only in the number of users but also in the volume of contents and resources being shared between them. In recent times, with the advent of peer-to-peer (p2p) based services, online social networks (OSN) as well as platforms that facilitate the sharing of user-generated content, the importance of user contributions has increased significantly. However, as the amount of these contributions is growing in the Internet community, their trustworthiness is becoming an important concern: In the context of file-sharing and content distribution systems, malicious participants that deliberately distribute malware, fake files or mislabeled content constitute a significant problem. At the same time, p2p based services frequently suffer from *freeriding* users which exploit the resources provided by other participants while not contributing resources in return. It has thus been acknowledged that it is important to consider trust and reputation management schemes in the design of any distributed system that relies on user contributions [1].

Typically, such trust and reputation management aims at globally estimating the trustworthiness of users based on a record of previous cooperative or defective behavior. In file-sharing scenarios, this typically refers to the quality of files previously provided to other users, while in settings prone to freeriders, the amount of previously shared resources may be considered for this purpose. There exist several examples for trust management mechanisms which approach this problem by estimating the trustworthiness of users by a global, typically scalar value [2], [3], [4], [5], [6].

A major drawback of such approaches to trustworthiness is the fact, that in general trust is a personalized concept, i.e. the level of trust users may have in the behavior of a particular participant will differ across different sets of users or user communities. At the same time, differences between user preferences may lead to situations where the trust in the quality of other users' content may vary significantly across different participants [3].

Due to the importance of providing *personalized notions of trust* in a variety of distributed computing scenarios, recently a number of works have addressed this question [7], [8], [9]. Typically, these models rely on a transitive notion of indirect trust, i.e. if a node A trusts node B and node B trusts node C, then it is assumed that node A also trusts node C to a certain extent. A major benefit of this approach is that, based on a very sparse network of direct trust relations, it allows to make statements about the *indirect trust between users that have not interacted so far*.

Interesting approaches have been studied which intend to capture this personalized and indirect trust. A major hurdle in applying them in practical settings is the huge communication and computation overhead that is necessary to exhaustively compute the indirect trust between all possible pairs of nodes. A particular observation that serves as a motivation for our scheme is however that the exhaustive computation of all pairs of indirect trust is often unnecessary.

Instead, there are numerous situations where it is sufficient to present users with a *personalized set containing only a limited number of most trusted peers*. Examples, where such a limited set of trusted peers can be useful are social recommender systems, peer selection schemes in file-sharing scenarios, as well as topology management and adaptation mechanisms in p2p-based distributed services. Furthermore, since the required number of trusted interaction partners can vary depending on the actual application, any practicable scheme to retrieve such a list should provide the designer of a distributed system with the possibility to make a trade-off between the required number of most trusted nodes and the communication and computation overhead entailed by the protocol.

Following this motivation, in this article we introduce and evaluate a simple heuristic mechanism which attempts to solve the open problem of estimating indirect trust in a more efficient way. In particular, as will be argued in section IV, we show that the proposed protocol captures precisely the same transitive notion of personalized indirect trust as the *TrustWebRank* algorithm which has been proposed in [9]. We further show, that our protocol effectively constitutes a very simple and practical heuristic that allows to sample indirect trust values to a variable degree of precision and completeness which can be tuned to the needs and resources of a particular application. In its essence, the protocol is a random message passing scheme in which nodes bias message passing probabilities according to the level of direct trust they hold in their previous interaction partners. The proposed protocol, as well as its relation to the TrustWebRank algorithm will be described in more detail in sections II and III. We then evaluate the correctness of the proposed scheme with respect to how well it is able to recover the indirect trust computed by the exhaustive and analytical approach taken by the TrustWebRank algorithm. We further evaluate the efficiency in terms of the reduction of messages passed compared to the original analytical approach. Simulation results based on scale-free as well as Erdös-Rényi trust topologies indicate that with as much as a 90% reduction in terms of message overhead, the proposed sampling algorithm can produce almost equivalent results as the TrustWebRank mechanism in terms of the identification of the most trusted remote nodes for each user in the network. We also propose a measure of global trustworthiness of the users in section V and compare the same for the users when the trust values are calculated using our mechanism and TrustWebRank. We finally conclude our paper with some interesting directions for future research.

II. TrustWebRank

In this section we discuss about TrustWebRank, an existing personalized trust estimation mechanism, and discuss about the motivation of our proposed algorithm. A summary of the list of symbols used in this description as well as in the remainder of the paper is shown in table I.

T_{ij}	Normalized direct trust of node i in node j
$\mathbf{T}$	Direct trust matrix whose each element is t_{ij}
β	Damping factor used in TrustWebRank mechanism
γ	Damping factor used in our proposed mechanism
$N(i)$	Neighbor set of node i
$\mathbf{N}$	Diagonal matrix representing the number of random walkers sent by each node
$\mathbf{H}$	Matrix, whose entry h_{ij} represents the number of hits from source i to j
$\hat{\mathbf{H}}$	Matrix obtained by normalizing each entry of $\mathbf{H}$ over each row
$\mathbf{S}$	Matrix, whose entry s_{ij} represents the trust rating of source i in j calculated using TrustWebRank
$\hat{\mathbf{S}}$	Matrix obtained by normalizing each entry of $\mathbf{S}$ over each row

Table I: List of symbols

In TrustWebRank, each node, i, maintains individual trust opinions about certain other users (represented by $T_{ij} \in (0, 1]$) with which it has direct interaction or have similarity in the preferences. These users are termed as direct neighbors of the node and the measured trust in these users as the *direct trust*. The trust estimation for user j that is not a direct neighbor of node i is done by considering all the paths used to reach user j weighted by the direct trust values of each link used in the path. It is necessary to explore all the paths, as more the number of paths to a user j, the higher will be its trust. The trust value calculated using this mechanism is termed as the *indirect trust* of node i on user j and is represented as $\tilde{T}_{ij}$. The TrustWebRank mechanism is an exhaustive mechanism for estimating trust between every pair of nodes and can also be implemented in a distributed network as proposed in [10]. The indirect trust values of the nodes obtained using the TrustWebRank mechanism can be represented analytically as follows.

If $\mathbf{S}$ is a stochastic matrix such that the elements $S_{ij} = \frac{T_{ij}}{\sum_{k \in N(i)} T_{ik}}$, then the indirect trust of node i in a node l is given as

$$\tilde{T}_{il} = S_{ij} + \beta \sum_{k \in N(i)} S_{ik} \tilde{T}_{kl} \quad \forall i, l \tag{1}$$

where $\beta \in [0, 1)$ is a damping factor that reduces the trust value of node l with increasing distance from i. If $\tilde{\mathbf{T}}$ denotes the matrix representing the indirect trust values then $\tilde{\mathbf{T}}$ can be represented as

$$\tilde{\mathbf{T}} = (\mathbf{I} - \beta \mathbf{S})^{-\mathbf{1}} \mathbf{S}, \tag{2}$$

where $\mathbf{I}$ is the identity matrix. These indirect trust values can be normalized to form a row stochastic matrix $\hat{\mathbf{S}}$ whose elements are represented as

$$\hat{S}_{ij} = \frac{\tilde{T}_{ij}}{\sum_{k \in N(i)} \tilde{T}_{ik}} \tag{3}$$

However, for large graphs, performing these matrix operations require huge computational overhead. A practical

distributed implementation scheme for the TrustWebRank mechanism has been proposed in [10], where they proposed an iterative method to derive the trust values of the node pairs. In this approach, equation 1 is formulated as

$$\tilde{T}_{il}^{m+1} = S_{ij} + \beta \sum_{k \in N(i)} S_{ik} \tilde{T}_{kl}^{m}, \tag{4}$$

where $\tilde{T}_{kl}^{m}$ is computed at step m. Thus the algorithm is a flooding based algorithm that iteratively explores the trust values of the nodes with increasing steps m finally leading to a convergent solution. This scheme is exhaustive with little control over the number of nodes that are to be explored and hence also on the message overhead involved. *However from a designer's perspective a practical alternative would be to design a tunable algorithm that can provide the users an option to consider a trade-off between the required number of trusted nodes and the allowable message overhead.* Users willing to explore more number of trusted nodes will have to subsequently pay with a higher overhead. This is practical, as in realistic scenarios, e.g. bootstrapping in p2p networks like Gnutella [11], a newly arriving node need not evaluate the trust to all the nodes in the network. Rather it will preferably connect to a small number of trusted peers and hence need to explore the trust of only a small subset of the whole network. Thus, from the perspective of an individual node, it would be more useful to identify certain number of most trusted peers with which it can communicate. In this paper, we exploit this concept to propose a tunable mechanism based on random walks that preferentially samples a set of nodes based on their trust values (evaluated using the TrustWebRank mechanism) with respect to a user in a network. The parameters in the algorithm can be tuned to increase or decrease the number of trusted nodes that are explored based on the allowable message overhead. The proposed mechanism efficiently selects trusted nodes with high probability besides being lightweight and scalable. We next detail our proposed mechanism.

III. Proposed Random Walk Based Mechanism

Similar to the TrustWebRank, we initially normalize the direct trust values to all the neighbors of a node to obtain the stochastic matrix $\mathbf{S}$ given as

$$S_{ij} = \frac{T_{ij}}{\sum_{l \in N(i)} T_{il}} \tag{5}$$

The normalized trust values provide measure of the relative trust in the neighbors of each node. To derive the indirect trust of the remote nodes, each node initially sends W random walkers, each traversing through a link with a probability equal to the normalized direct trust value of the node connected through that link. Thus for a node i, a random walker visits a neighbor j with probability S_{ij}. Hence for W random walkers the average number of walkers that visit node j from node i is given as

$$V_j^{(i)} = W \cdot S_{ij}. \tag{6}$$

When a walker from a source node i reaches a node j on the h^{th} hop, node j stamps its ID on the walker message and does any of the following two steps.

1) With a probability, $\gamma^h S_{jk}$, it forwards the message to a neighbor node k, where γ is the damping factor, similar to β that is used in TrustWebRank, with a value that lies between $(0, 1)$.
2) With a probability $1 - \sum_{l \in N(j)} (\gamma)^h S_{jl}$, the random walker dies, i.e. j does not forward the message further, instead it sends back the message to the source node i.

Thus γ controls the number of hops of the random walker and hence we do not use an explicit Time-to-Live (TTL) value of these walkers. The major importance of using this damping factor is that nodes which are nearer to a source will receive more number of hits as compared to distant nodes. The rationale behind this is, it is difficult to predict the trust value of a node which is far away from a source. Thus the trust value of a node will decrease with path length from the source node.

On receiving the W messages back, the source node calculates the number of times each node has been hit that provides a measure of trust to the remote nodes. Thus the precision of the trust values depends on both the number of walkers as well as the damping factor γ and hence need to be carefully tuned so as to control the message overhead besides maintaining a certain level of precision. We discuss the effects of both these parameters later in section IV. Thus we can represent the probability that a random walker that has reached a node j on the h^{th} hop, traverses the link to a neighbor node k in the next hop by

$$H_{jk}^{(h)} = \gamma^h T_{jk} \tag{7}$$

and the probability that the random walker dies at node j in the h^{th} step by,

$$D_j^{(h)} = 1 - \sum_{l \in N(j)} H_{jl}^{(h)} = 1 - \sum_{l \in N(j)} \gamma^h T_{jl}, \tag{8}$$

Thus the average indirect trust values of each pair of peers for our proposed algorithm can be derived as follows. The stochastic matrix $\mathbf{S}$ provides the probability that a random walker from node i will reach a neighbor j in one step. The damping factor γ reduces the probability of a walker to continue further after each hop; thus the matrix representing the probabilities that a random walker from any node i reaches node j after exactly l hops (although it might have reached j in earlier hops) is given as $\gamma^{l-1}\mathbf{S}^l$. So the average number of times node j has been hit from source i by a

single random walker can be written using a matrix notation as

$$\begin{aligned} \mathbf{H_1} &= \frac{1}{\gamma}\left[(\gamma\mathbf{S})^1 + (\gamma\mathbf{S})^2 + \ldots \infty\right] \\ &= (\mathbf{I} - \gamma\mathbf{S})^{-1}\,\mathbf{S}, \end{aligned} \tag{9}$$

where $\mathbf{I}$ is the identity matrix. Note, equation 9 provides the indirect trust between any pair of nodes and is exactly similar to equation 2 of the TrustWebRank scheme. Thus this proves that our random walk based method produces consistent trust estimates with the TrustWebRank mechanism. To implement this mechanism, we send a number of random walkers from each node and if $\mathbf{N}$ be a diagonal matrix that represents the number of random walkers sent by each node, then, the average number of hits at each node for every source node can be represented as $\mathbf{H} = \mathbf{N}\mathbf{H_1}$.

As the trust in a node j by node i is estimated by the number of hits to node j of the random walkers from i, one should note that node j should be hit at least once from i to have a measure of the trust on it. Thus to ensure that at least one random walker reaches to all other nodes in the connected component, each node will have to theoretically send infinitely large number of random walkers. Hence, by sending infinitely large number of random walkers from each node, the nodes can obtain the same relative trust value in the other nodes as is measured using the TrustWebRank. However, it is obvious that sending such a large number of random walkers is infeasible due to the huge message overhead it would generate. Hence, the tunability of algorithm is achieved by controlling the number of random walkers, where a high number of random walkers will help to explore a large subset of nodes in order of their trust but at the expense of a higher message overhead.

We can normalize each row of the matrix, $\mathbf{H}$, to produce a row stochastic matrix, $\hat{\mathbf{H}}$, where each element $\hat{h}_{ij}$ will represent the relative trust of node i on j over all the nodes in the network. We use the matrix, $\hat{\mathbf{H}}$, along with $\hat{\mathbf{S}}$, the normalized trust matrix obtained in the TrustWebRank mechanism, to analyze the preciseness and efficiency of our proposed algorithm. We perform a comparative study in the next section based on the following issues that we need to analyze:

1) How does the trust values correlate with the trust values calculated using TrustWebRank for various kind of topologies?
2) How does the number of random walkers and the damping factor influence the preciseness of the trust values?
3) How much performance efficiency is achieved by using our algorithm as compared to the TrustWebRank?

We seek answers to these questions in the next section.

IV. Performance Analysis

In this section we critically analyze the proposed algorithm in terms of several metrics related to its correctness and efficiency. The initial analysis is based on the ability of the algorithm to identify the highly trusted nodes in the network with respect to a particular user and then evaluating the minimum overhead required to identify a critical fraction of a set of trusted nodes. We then analyze the total coverage achieved in terms of reaching a node by the random walkers and the efficiency achieved in terms of reducing message overhead. We observe the effect of the tuning parameters like the number of random walkers and the dampening factor γ on these metrics and compare the results with that of the TrustWebRank mechanism. We finally extend the concept of personalization to evaluate a global perspective of the nodes in a network and then analyze the efficiency of the algorithm in identification of the globally trusted peers in the network.

Experimental Setup: Our analysis of the accuracy and efficiency of the algorithm is done on two types of networks, scale-free and Erdös-Rényi networks comprising of 1000 nodes with an average degree of 10. The scale-free network is generated using the Barabasi-Albert preferential attachment model [12]. The number of random walkers in the simulations is varied from 60 to 460, whereas the value of γ is varied from 0.3 to 1.0. The value of β is fixed to 0.75. This is as $\beta = 0.75$ to 0.85 has been declared to be the best possible set of values by the proponents of TrustWebRank in [9]. We next discuss the performance of the algorithm for the above stated metrics.

A. Identification of Trusted Nodes

One of the major motivations of the algorithm is to sample the trusted nodes in the network with respect to an individual node. To test the efficiency of the algorithm with respect to this objective, we observe the ability of the algorithm to identify the *top p percentage of the most trusted nodes*. We formally define this metric as follows:

Definition 1: If $\mathcal{S}_H^p(i)$ and $\mathcal{S}_S^p(i)$ denote the set of top p percentage of the most trusted nodes of node i, obtained using our proposed algorithm and the TrustWebRank respectively, then the fraction of overlap of these two sets, given by $\frac{\mathcal{S}_H^p(i) \cap \mathcal{S}_S^p(i)}{|\mathcal{S}_H^p(i)|}$, provides a measure of efficiency of the proposed algorithm in identifying the highly trusted nodes in network by an individual node.

We initially analyze the average fraction of overlap of the node set of top 5% of the most trusted nodes for both TrustWebRank and our proposed mechanism for varying number of random walkers and for various values of γ. We also derive the rank correlation, $\mathcal{R}$, of the overlapping set of nodes, using the Spearman rank correlation coefficient, and the absolute trust difference of the non overlapping nodes. As stated earlier, the fraction of overlap provides a measure of correctness of the proposed algorithm in identifying

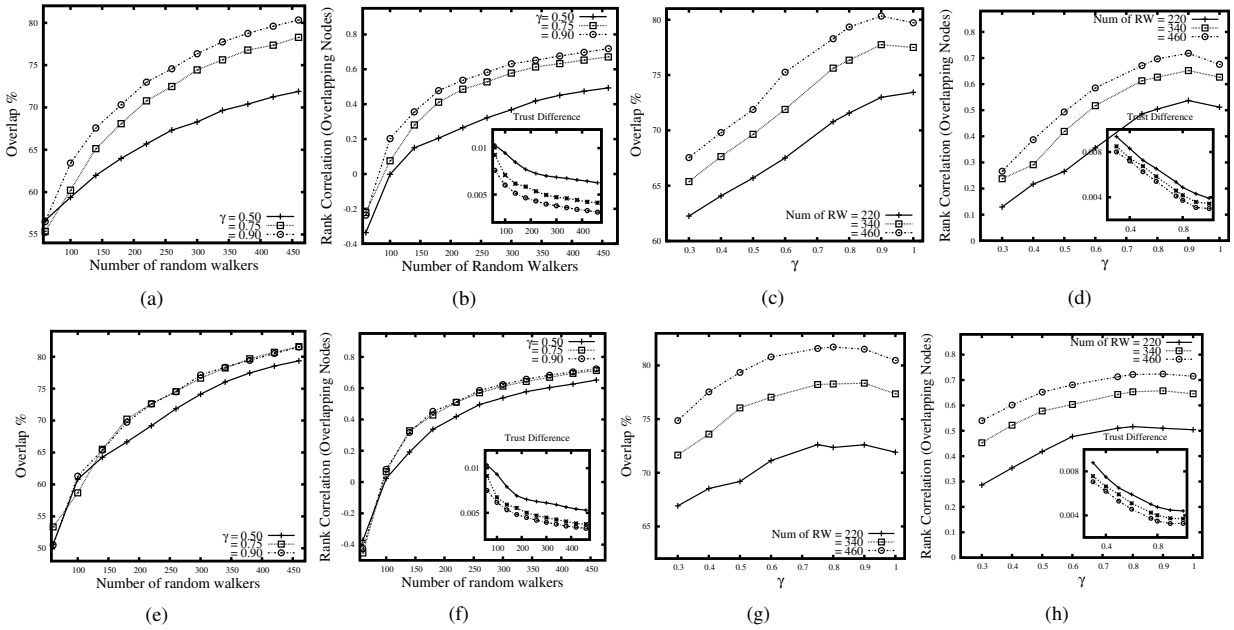


Figure 1: 1(a) – 1(d) show the simulation results for scale-free networks and 1(e) – 1(h) show the same results for Erdös-Rényi networks. 1(a) and 1(e) show the percentage overlap of the top 5% most trusted peers for TrustWebRank and our proposed algorithm for scale-free and Erdös-Rényi networks, respectively, for varying number of random walkers. 1(c) and 1(g) show the same results for scale-free and Erdös-Rényi networks, respectively, for various values of γ. 1(b) and 1(f) show the rank correlation of the overlapping peers for scale-free and Erdös-Rényi networks, respectively, and the figures in inset shows the trust difference of the non-overlapping peers, for varying number of random walkers. 1(d) and 1(h) show the same result for scale-free and Erdös-Rényi networks, respectively, for various values of γ. The networks considered are of 1000 nodes with an average degree of 10. The value of β considered for the TrustWebRank mechanism is 0.75.

the most trusted nodes by an individual node. The rank correlation $\mathcal{R}$ provides an indication that if for a node i, the set of its trusted nodes obtained using TrustWebRank can be sequenced in order of their trust values, then whether our algorithm can also rank the nodes in the same sequence. For the non-overlapping nodes, the difference in the normalized trust values (that we term as *Trust Difference* of the non-overlapping peers) is calculated to provide an indication of how close our algorithm has been in missing the trusted node. We discuss the effect of the number of random walkers and the dampening factor γ in this context.

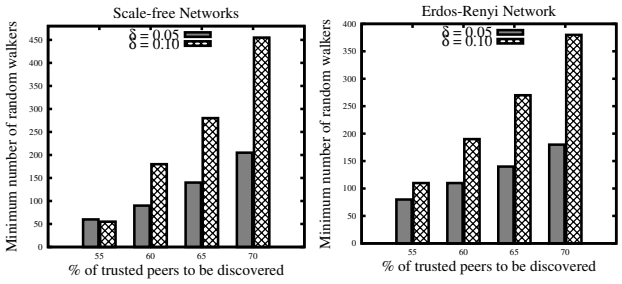
Effect of the number of random walkers: We initially discuss the effects of the number of random walkers on the above described parameter. Simulation results in figures 1(a) – 1(d) show the results for scale-free networks and the figures in 1(e) – 1(h) show the same for Erdös-Rényi networks, for increasing number of random walkers. Figures 1(a) and 1(e) show the average percentage overlap of the top 5% most trusted peers for TrustWebRank and our algorithm for scale-free and Erdös-Rényi networks respectively. The percentage overlap for Erdös-Rényi networks is higher as compared to the scale-free networks for low value of γ (0.50 in this case). This is due to the presence of a small number of hubs and large heterogeneity in the degree of nodes in scale-free networks; for low values of γ, the random walkers initially move towards the very few hub nodes and die out without exploring further trusted nodes. However, as can be observed that both the networks show similar trends, where with increasing number of random walkers, the percentage of overlap increases and is nearly 80% when the number of random walkers is 460, for higher values of γ. Further as shown in figures 1(b) and 1(f), the rank correlation of the overlapping set of nodes is nearly 0.8 for both scale-free and Erdös-Rényi networks indicating that the ordering of the nodes in terms of their trust values as observed in TrustWebRank is highly maintained in our algorithm. The figures in inset shows the trust differences of the non-overlapping nodes. It can be observed that although the differences is very high (nearly 0.01) for very low number of random walkers but it decreases rapidly to as low as 0.001, when the number of random walkers is increased to 460. This indicates that using our algorithm, nodes are able to identify the most trusted nodes in the network with a very high probability.

Effect of the Damping Factor γ: Simulation results for certain random walkers shown in figures 1(c) and 1(g)

indicates that the percentage overlap increases very rapidly with increasing values of γ, reaches a maximum point after which it starts decreasing. For $\gamma = 0.8$, the percentage overlap is nearly 80% for both scale-free and Erdös-Rényi networks for 460 number of random walkers. Beyond a threshold value of γ, the random walkers tend to forget their respective starting source node and the number of hits to a node become proportional to its degree. Thus the number of hits to a node gets biased by its degree and hence the percentage of overlap decreases. This fact causes similar trends for the rank correlation of the overlapping nodes and the trust differences for the non-overlapping nodes, as shown in figures 1(d) and 1(h).

Thus we observe that the damping factor γ plays an important role in our algorithm in determining the efficiency to predict the top trusted nodes in the network. By selecting an optimal value of γ (nearly 0.75-0.8 in this case) the algorithm can be highly efficient in identifying the trusted nodes in the network.

We next observe the variation of the minimum number of random walkers required in recovering a certain fraction of a set of most highly trusted nodes in the network.



(a) (b)

Figure 2: 2(a) shows the minimum number of random walkers required to identify a certain percentage of the nodes in a set of top 5% and 10% (δ = 0.05 and 0.10 respectively) most trusted nodes in the network, for scale-free networks. 2 shows the same for Erdö-Rényi networks. The number of nodes used in the simulation is 1000 and the average degree is 10 for both these networks.

B. Recoverability of Trusted Nodes

In this section we observe how the minimum number of random walkers can be tuned to recover a certain fraction of the most trusted nodes.

If δ denotes a threshold fraction of the top trusted nodes in a network, we observe the minimum number of random walkers required to identify a certain fraction of these most trusted nodes. This provides an indication of the minimum overhead required to identify a certain fraction of highly trusted nodes. We show the results for $\delta = 0.05$ and 0.10, i.e. the top 5 and 10% of the trusted peers and observe the number of random walkers required to recover a certain fraction of these highly trusted nodes.

As shown in figures 2(a) and 2(b) for scale-free and Erdös-Rényi network respectively, the minimum required number of random walkers increases rapidly for obtaining a higher fraction of the top trusted nodes for increasing values of δ. This confirms that for identifying a very high fraction of nodes in the network (i.e. $\delta \rightarrow 1.0$), the minimum number of random walkers required will be enormously high; however, for smaller values of δ, by appropriately tuning the number of random walkers, a certain fraction of the top trusted nodes can be easily identified by using much less number of random walkers.

C. Network Coverage

We now analyze the network coverage of the peers achieved in terms of the number of unique nodes explored by the random walkers from a source node and hence has an estimate of their trust values. We observe the network coverage for both the number of random walkers as well as the damping factor γ.

Effect of the number of random walkers: We simulated the coverage of the nodes, i.e. the number of nodes in the network that has been reached by the random walkers and hence has a non-zero trust value. The results shown in figure 3(a) for $\gamma = 0.75$ indicate that the coverage in case of scale-free network is higher as compared to Erdös-Rényi networks is higher; this is due to the lower diameter of the scale-free network as compared to the latter. However, in both cases, although the coverage of the nodes are small for low number of walkers, but the coverage of the nodes increases rapidly with small increase in the number of walkers. The coverage of the nodes is as high as 45% for the scale-free networks, when the number of walkers is 460. However for higher values of random walkers, the rate of increase of the coverage is slower for a fixed value of γ.

Effect of the damping factor: We also simulated the coverage of the nodes for various values of γ when the number of random walkers is equal to 460. The simulation results shown in 3(b) shows that the damping factor has a huge effect on the coverage of the peers. Although the coverage of the peers for lower values of γ is very low but with increasing values of the damping factor the coverage of the nodes increase rapidly and reaches almost 80% for a γ of 1.0. Thus from figures 3(a) and 3(b), we conclude that the coverage of the nodes can be tuned by a suitable controlling the number of random walkers and γ. Note that although we are sampling just 45% of the nodes, we are identifying almost all the most trusted nodes in the network.

We next analyze the efficiency of the algorithm in terms of the involved message overhead.

D. Message Overhead

In this section we discuss about the reduction in message overhead using our algorithm as compared to the distributed

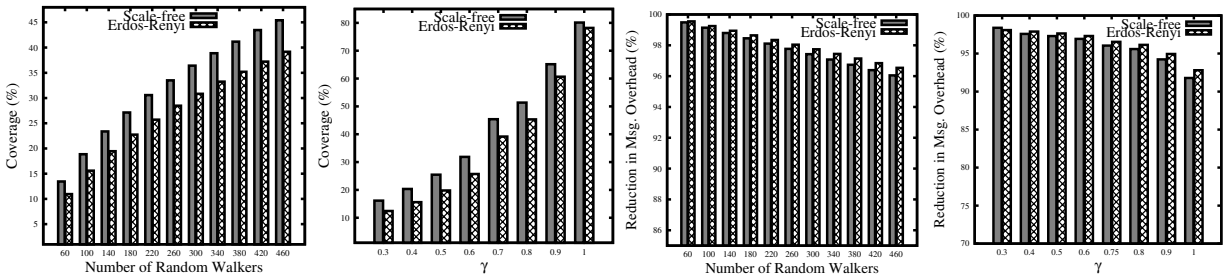


Figure 3: 3(a)and 3(b) compare the average coverage of the nodes for various random walkers and various values of γ, respectively, expressed in terms of the percentage of nodes with non-zero trust values per node for both scale-free and Erdös-Rényi networks. 3(c) and 3(d) show the percentage reduction in message redundancy by using the proposed algorithm for increasing number of random walkers and for various values of γ, respectively, as compared to the TrustWebRank mechanism for both scale-free and Erdös-Rényi networks. The number of nodes in each of the simulation results is 1000 and the average degree of the nodes is 10. The number of random walkers is varied from 60 to 460 and the value of both β and γ is considerd as 0.75.

implementation of TrustWebRank in [10]. We assume that a single unit of message is passed when a node communicates with other node for both the TrustWebRank as well as our proposed algorithm. We observe the effect of both the number of random walkers as well as the damping factor on the percentage reduction of number of messages passed by a node.

Effect of the number of random walkers: We simulated the reduction in the number of messages with increasing number of random walkers when the value of γ is 0.75. The simulation results shown in figure 3(c) indicate that for both scale-free and Erdös-Rényi networks, the proposed algorithm sends nearly 96% less number of messages as compared to the TrustWebRank, when the number of random walkers is 460. The overhead involved is extremely low due to the fact that by using a constant number of random walkers only a very small fraction of the edges are traversed.

Effect of the damping factor: We also simulated the effect of γ on the reduction of the number of messages when the number of random walkers is equal to 460. Simulation results for scale-free network shown in figure 3(d) indicate that by increasing the value of γ from 0.3 to 1.0 leads to an increase in the message overhead by around 62% (not shown in figure) but the reduction in message overhead as compared to TrustWebRank is still around 92%.

We next attempt to evaluate a global trust measure of a node, in light of how trusted it is perceived as trusted by other users in the network and observe whether the proposed mechanism can capture such a trust measure.

V. From Local to Global Trust

Personalized trust values can in turn be used to calculate the global trust of a node, i, by aggregating all the personal trust values of the other nodes in node i. Such a global trust can be helpful for a node in selecting initial trusted users in a community with whom the node had no prior interaction. To capture the global trust of the nodes, we observe the *global importance of the nodes* in the network that we define as follows:

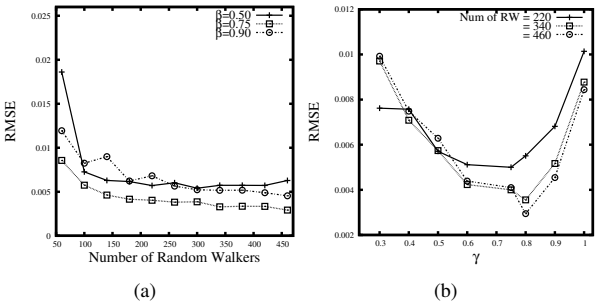


Figure 4: 4(a) and 4(b) show the simulation results for the RMSE values with respect to the number of random walkers and γ respectively. The threshold value τ is considered to be 0.01. The networks considered are of 1000 nodes with an average degree of 10. The value of β considered for the TrustWebRank mechanism is 0.75

Definition 2: We define the global importance, I_j, of a node, j, in the whole network as the fraction of the nodes in the network that have normalized trust value in node j, greater than a given threshold τ. Thus the global importance of a node gives a measure of what fraction of nodes in the network consider the node as trusted and have trust in it greater than the threshold τ.

We observe the effectiveness of the proposed algorithm in identifying the globally trusted nodes in the network. We analyze the root mean square error (RMSE) values of the global importance of all the nodes, for the TrustWebRank and our proposed mechanism. So if $I_i^{(T)}$ and $I_i^{(R)}$ denote

the global importance values of node i, calculated using the TrustWebRank and our proposed mechanism, respectively, then the RMSE is given as $\sqrt{\frac{\sum_i (I_i^{(T)} - I_i^{(R)})^2}{N}}$, where N is the number of nodes in the network. The threshold value τ in our simulations is considered as 0.01. The RMSE values indicate whether the fraction of nodes that considers a node as trusted differs when the trust values are calculated using our mechanism and TrustWebRank.

Effect of the number of random walkers: Simulation results for scale-free networks, shown in figure 4(a) indicates that when the value of γ is greater than 0.75, for moderately large number of random walkers, the RMSE values are very low — around 0.006. Since the global importance of a node indicates a fraction, a RMSE value of 0.006 indicates that the percentage of nodes that consider a node as trusted, when the trust values are calculated using TrustWebRank and our mechanism, differ by only 0.6%, which is very low.

Effect of the damping factor: The effect of γ on the RMSE values of global importance of the nodes, for scale-free network, shown in figure 4(b), indicates that with increasing γ, nodes are able to identify the globally trusted nodes more effectively. However beyond a threshold value of γ (which is near to $\gamma = 0.8$ in the present scenario), the RMSE values start increasing. The reason behind this trend is, as explained earlier, the coverage of the nodes increases with increasing γ thus leading to decreasing RMSE; however, with further increase the number of hits at a node become biased by its degree leading to a decrease in RMSE values. Thus for a suitable value of γ, the algorithm effectively identifies the globally trusted peers in the network.

VI. Conclusion

In this paper, we have proposed a random walk based mechanism to identify the trusted nodes in a network with respect to a user in a distributed network. We have shown that preferentially sampling the nodes in the network based on their direct trust values identifies the highly trusted nodes in the network, by exploring a very small subset of the total nodes and with much less overhead as compared to an exhaustive search. However the present analysis do not consider the dynamicity of the nodes in the network. But it may be assumed that the trusted nodes in the network are relatively more stable as compared to other nodes. Hence these nodes can be easily located using our algorithm, as sampling in presence of such dynamicity will preferentially lead to the more relatively stable nodes. Further, this algorithm can be naturally extended to explore the non-transitive relations among the nodes (as discussed in [13]) and compute trust based on these relations.

References

[1] A. Wierzbicki and A. Wierzbicki, "Trust management," in *Trust and Fairness in Open, Distributed Systems*, ser. Studies in Computational Intelligence. Springer Berlin / Heidelberg, 2010, vol. 298, pp. 71–143.

[2] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proceedings of the 2007 ACM conference on Recommender systems*, ser. RecSys '07. New York, NY, USA: ACM, 2007, pp. 17–24.

[3] Y. Matsuo and H. Yamamoto, "Community gravity: measuring bidirectional effects by trust and rating on online social networks," in *Proceedings of the 18th international conference on World wide web*, ser. WWW '09. New York, NY, USA: ACM, 2009, pp. 751–760.

[4] H. Zhao and X. Li, "VectorTrust: Trust vector aggregation scheme for trust management in peer-to-peer networks," in *Proceedings of 18th Internatonal Conference on Computer Communications and Networks, 2009*, ser. ICCCN '09, Aug. 2009, pp. 1 –6.

[5] R. Zhou and K. Hwang, "Trust overlay networks for global reputation aggregation in P2P grid computing," in *Proceedings of 20th International Conference on Parallel and Distributed Processing Symposium*, ser. IPDPS '06, April 2006, p. 10 pp.

[6] Y. Bernard, L. Klejnowski, J. a. H andhner, and C. M andller Schloer, "Towards trust in desktop grid systems," in *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*, may 2010, pp. 637 –642.

[7] N. Li, S. El Helou, and D. Gillet, "Trust-based rating prediction for recommendation in web 2.0 collaborative learning social software," in *Proceedings of the 9th international conference on Information technology based higher education and training*, ser. ITHET'10. Piscataway, NJ, USA: IEEE Press, 2010, pp. 197–201.

[8] X. Liu, A. Datta, K. Rzadca, and E.-P. Lim, "StereoTrust: A group based personalized trust model," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, ser. CIKM '09. New York, NY, USA: ACM, 2009, pp. 7–16.

[9] F. E. Walter, S. Battiston, and F. Schweitzer, "Personalised and dynamic trust in social networks," in *Proceedings of the 3rd ACM Conference on Recommender systems*, ser. RecSys '09. New York, NY, USA: ACM, 2009, pp. 197–204.

[10] V. Carchiolo, A. Longheu, M. Malgeri, and G. Mangioni, "A distributed algorithm for personalized trust evaluation in social networks," in *Intelligent Distributed Computing IV*, ser. Studies in Computational Intelligence, M. Essaaidi, M. Malgeri, and C. Badica, Eds. Springer Berlin / Heidelberg, 2010, vol. 315, pp. 99–108.

[11] P. Karbhari, M. Ammar, A. Dhamdhere, H. Raj, G. Riley, and E. Zegura, "Bootstrapping in gnutella: A measurement study," in *Passive and Active Network Measurement*, ser. Lecture Notes in Computer Science, C. Barakat and I. Pratt, Eds. Springer Berlin / Heidelberg, 2004, vol. 3015, pp. 22–32.

[12] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, Oct 1999.

[13] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," in *Proceedings of the 13th international conference on World Wide Web*, ser. WWW '04. New York, NY, USA: ACM, 2004, pp. 403–412.