Finding the Bias and Prestige of Nodes in Networks based on Trust Scores

Abhinav Mishra
Dept. of Computer Science and Engineering,
Indian Institute of Technology, Kanpur,
India
abhinavm@iitk.ac.in

Arnab Bhattacharya
Dept. of Computer Science and Engineering,
Indian Institute of Technology, Kanpur,
India
arnabb@iitk.ac.in

ABSTRACT

Many real-life graphs such as social networks and peer-to-peer networks capture the relationships among the nodes by using trust scores to label the edges. Important usage of such networks includes trust prediction, finding the most reliable or trusted node in a local subgraph, etc. For many of these applications, it is crucial to assess the prestige and bias of a node. The bias of a node denotes its propensity to trust/mistrust its neighbours and is closely related to truthfulness. If a node trusts all its neighbours, its recommendation of another node as trustworthy is less reliable. It is based on the idea that the recommendation of a highly biased node should weigh less. In this paper, we propose an algorithm to compute the bias and prestige of nodes in networks where the edge weight denotes the trust score. Unlike most other graph-based algorithms, our method works even when the edge weights are not necessarily positive. The algorithm is iterative and runs in $O(km)$ time where k is the number of iterations and m is the total number of edges in the network. The algorithm exhibits several other desirable properties. It converges to a unique value very quickly. Also, the error in bias and prestige values at any particular iteration is bounded. Further, experiments show that our model conforms well to social theories such as the balance theory (enemy of a friend is an enemy, etc.).

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data Mining

General Terms

Algorithms, Design

Keywords

Trust-based networks, Bias, Prestige, Deserve, Social networks

1. INTRODUCTION

In online social networks such as Facebook (`www.facebook.com`), Youtube (`www.youtube.com`), Twitter (`www.twitter.com`), etc., people share different content such as messages, videos, songs, opinions, blogs, etc. In another important form of networks—peer-to-peer networks—files are shared.

In both these cases, reputation of a peer matters; otherwise a user may get exposed to objectionable content such as a virus. Thus, the trust that other users impart on a node is an important attribute of it. Slashdot (`www.slashdot.org`) and Epinions (`www.epinions.com`) are networks where explicit opinions of a user as trust and distrust are available.

A network based on trust is quite different from other networks. An explicit link in a network such as Facebook, Youtube signifies that two nodes are close. However, in a trust based network, two nodes may be close and may be connected but the link may show distrust. More importantly, a neutral opinion in a trust based network is quite different from a no-connection. Consider a simple example where node A has 1000 neutral opinions and 10 negative opinions about it and another node B has only 10 negative opinions about it. It is intuitive that node A has a higher reputation. However, if the neutral opinion is modeled by the absence of an edge, the two nodes behave the same. This indicates that neutral opinion is not the same as a no-connection. In other words, an edge with 0 weight is different from an edge that is absent. This poses new challenges for trust based networks as random-walk based approaches [7] cannot be directly used.

Traditionally, techniques like HITS [10] and PageRank [14] give high rank to nodes with better connectivity. The situation changes dramatically in trust-based networks as a highly disliked node may also be well connected but should have a low prestige. In a trust network, prestige of a node depends on the opinions of other nodes whereas trustworthiness of a node depends on how a node gives correct opinion about other nodes (as against the opinion from other nodes in the network). This raises many interesting questions about the trust network. Is a node with higher prestige as trustworthy as another node? In general, though, we expect the two attributes to be correlated.

In a rating system, such as IMDB (`www.imdb.com`), Slashdot (`www.slashdot.org`), Epinions (`www.epinions.com`), etc., users rate an object. For example, a movie is an object in case of IMDB, and a user could be an object in case of networks such as Epinions and Slashdot. In the latter case, every user essentially rates other users and receive ratings from them. The prestige of a user is based on the opinion of other users, i.e., opinion of other people in the form of inlinks a user gets. An opinion of an user about others is based on her ratings (outlinks). Considering a user in isolation, we can say that her prestige depends on the *quality* of inlinks and truthfulness of a user who has rated her, and not

on quantity.[1] The truthfulness, on the other hand, depends on the opinion she gives in the form of outlinks. We refer to *truthfulness* of a node as *bias*[2] and *prestige* of a node as *deserve*[3]. If a node is biased, then its opinion should not weigh significantly. Then, what another node deserves (prestige) relies more on nodes that are more truthful (i.e., have low bias). A node that gives only positive ratings irrespective of what other node deserves can be said to have a high bias. Similarly, a node receiving positive inlinks from highly biased nodes has a lower deserve value than a node receiving inlinks from truthful nodes.

We can observe that there is no direct relationship between bias and prestige of a node. For example, a node may receive all positive ratings (has high deserve value) and still gives an opinion about another node that differs from the opinions of all other nodes in the network.

In this paper, we present a model that computes the prestige and trustworthiness of nodes in a trust-based network. It is based on the idea that the opinion of trustworthy nodes weigh more. We obtain the trustworthiness of a node by how well it computes the prestige of its neighbours. We show that our algorithm is fast ($O(km)$, where m is the number of edges in the network and k is the number of iterations), parameter-free and converges to a unique solution. To the best of our knowledge, there is no other algorithm that accommodates both negative and positive links and is guaranteed to converge. We also bound the error which enables us to know the require number of iterations required to get the desired accuracy.

There might also be few special requirements in a trust-network such as high trustworthiness of special nodes [9] (e.g., celebrity profiles on Twitter, Facebook, etc.), resistant to malicious users where a user can try to disturb the network by giving wrong opinion [9], etc. Further, there are different problems such as trust-prediction [5] between two nodes who do not have any prior contact. Another similar problem is edge sign prediction [11], where we try to predict whether there is a trust or distrust. We show that our algorithm fits in these problems. We also observe that the algorithm adheres well to social theories such as theory of balance ("enemy of an friend is a enemy", etc.).

The paper is organized as follows. We present the related work in Section 2. Section 3 describes the algorithm. In Section 4, we present the proofs for convergence and error-bounds. We discuss various extensions of our method in Section 5. Section 6 presents the experimental results before Section 7 concludes.

2. RELATED WORK

Graph theoretic methods for ranking nodes in a network have gained popularity since the introduction of HITS [10] and PageRank [14] algorithms. Subsequently, a number of other methods have also been proposed. Most of these

---

[1]In random-walk based techniques [7], net-quantity of in-links matters as well. However, in a trust-based network, we consider quality and truthfulness of an user as the main criteria.

[2]Actually, a node can have bias close to zero even if it is not fully truthful, but has given opposite opinions in both positive and negative manner; we ignore such cases for ease of discussion.

[3]In the rest of the paper, we use the pair of terms bias and truthfulness, and deserve and prestige interchangeably.

methods are usually a variant of eigenvector centrality measure [1]. For these, however, the edge weights needs to be non-negative as Perron-Frobenius [4] theorem relies on the non-negativity. In a network, where an object can be rated based on explicit rating such as IMDB, Epinions, etc., or where rating is derived as in the case of peer-to-peer networks, the non-negativity can no longer be enforced. Moreover, by including negative weights, a convergence cannot be guaranteed as the matrix is not stochastic. The algorithm *EigenTrust* [9, 15] removes negative entries by not considering negative ratings. Ranking has been done on trust based network as well while considering negative links, e.g. *PageTrust* [3]. However, convergence is not guaranteed as the matrix is not stochastic when negative links are included. It has been pointed out in [5] that by removing negative ratings one cannot distinguish between the no-connection and distrust. It was also argued that shifting is not an answer either due to many reasons including the distortion of the semantics of zero score. The authors of [5] proposed an algorithm to compute trust/distrust between two objects. There have been studies on the social aspects of trust-based networks. One important example is the *balance theory* [2] that considers relationships of type "enemy of an enemy" as a friend. Another popular theory is *status theory* [6], where a positive link denotes higher status. These theories have been well evaluated in [11, 12].

3. ALGORITHM TO COMPUTE BIAS AND PRESTIGE OF A NODE

In this section, we first define the terms "bias" and "prestige" precisely before describing and analyzing an algorithm to compute them. We consider two types of trust-based networks. In the first one, the users give both positive and negative scores (not necessarily discrete values). Here, a more positive rating signifies a higher confidence, 0 is a neutral opinion, and negative rating signals a low confidence. Examples of such networks are Slashdot and Epinions. In the second type of networks, only non-negative scores are given. In this case, a high positive score signifies more confidence and a low positive score less confidence. This type of rating is in practice for networks such as IMDB and Youtube.

In both the models, it is not difficult to normalize the ratings, and keep them in the range of $[-1, 1]$ for the former and $[0, 1]$ for the later. In our algorithm, a user can follow any model and can give rating either both positive and negative, or only non-negative rating. From here on, we cover both such systems to rate under a generalized range of $[-1, 1]$. For non-negative rating systems, rating would attain values (normalized) between 0 and 1.

We model the above trust networks using graphs where the edge weight indicates the user opinion. If a user does not rate, then there is no edge. If a user gives neutral rating, it is denoted by the edge weight 0. As discussed earlier, this is significantly different from a no-connection, as the absence of an edge indicates that the user did not rate. Likewise for the later case, where user has to give only positive scores, the edge weights attain only non-negative values. Here, an explicit 0 score implies very low confidence, but is again quite different from a no-connection that signifies no rating.

Formally, let $G = \{V, E\}$ be a graph, where an edge $e_{ij} \in E$ (directed from node i to node j) has weight $w_{ij} \in [-1, 1]$. We say that node i gives the trust-score of w_{ij} to node j.
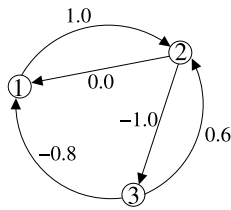
Figure 1: Graph with edge weights indicating trust.

Let $d^o(i)$ denotes the set of all outgoing links from node i and likewise, $d^i(i)$ denotes the set of all incoming links to node i. In this work, we measure two attributes of a node:

- *Bias* or *Trustworthiness*: This reflects the expected weight of an outgoing edge.

- *Deserve* or *Prestige*: This reflects the expected weight of an inlink from an unbiased node.

In the next section, we define these terms formally.

3.1 Definitions

The bias of a node is its propensity to trust/mistrust other nodes. Thus, the propensity or bias of a node can be measured by the difference between the rating a node provides to another node (i.e., the edge weight) and the "ground" truth, i.e., what the second node truly deserves (this takes into account the trust by other nodes). The bias of a node i is given by

$$bias(i) = \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \big(w_{ij} - deserve(j)\big). \qquad (1)$$

Normalization is done to maintain the value of bias in the range of $[-1, 1]$. A node is truly truthful if it has a bias of 0.

A node has a positive bias if it has a propensity to give positive outlinks, and a negative bias otherwise. A node giving a positive rating to other nodes that do not deserve such ratings values would attract a high bias. Using bias, the inclination of a node toward trusting/mistrusting is measured. It can also be used to understand the true nature of a node. If a highly biased node (either positive or negative) gives a rating, then such score should be given less importance. We can do so by reducing the effect of bias from each outlink a node gives. For example, if a node with a high positive bias gives a outlink with large positive weight, then the weight is reduced from this edge while calculating the rating of the other node. Similarly, negative weights from a negatively biased node are reduced. However, if a node has an edge whose weight has an opposite sign of that of the bias, we do not make any changes. Intuitively, when a person known to give a negative feedback in general, actually gives a positive feedback, then her opinion should weigh significantly. Therefore, if a node has a positive (negative) bias and has an edge with negative (positive) weight, then we do not make any change to the edge weight.

We introduce an auxiliary variable X_{kj} to measure the effect of bias of node k on its outgoing edge to node j per unit edge-weight:

$$X_{kj} = \begin{cases} 0 & \text{if } (bias(k) \times w_{kj}) \leq 0 \\ |bias(k)| & \text{otherwise} \end{cases} \qquad (2)$$

	Node 1	Node 2	Node 3
Bias	0.13	0.08	-0.14
Deserve	-0.33	0.73	-1.00

Table 1: Final bias and deserve values for the nodes in the example graph in Figure 1.

Iteration	Node 1		Node 2		Node 3	
	B	D	B	D	B	D
0	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00
1	0.10	0.00	-0.25	0.80	-0.25	0.00
2	0.12	-0.30	0.01	0.75	-0.16	-0.75
3	0.13	-0.33	0.08	0.73	-0.15	-1.00
4	0.13	-0.33	0.08	0.73	-0.14	-1.00
5	0.13	-0.33	0.08	0.73	-0.14	-1.00

Table 2: Example showing the values of the graph (Figure 1) after each iteration. (B denotes bias and D denotes deserve.)

From the above expression, we can see that when bias and edge weight are of opposite signs, X_{kj} becomes zero and there is no effect of the bias. Otherwise, X_{kj} becomes the absolute value of the bias. There is an equivalent formulation of X_{kj}:

$$X_{kj} = \max\{0, bias(k) \times sign(w_{kj})\}. \qquad (3)$$

We can now reduce the edge weight using the effect of bias, i.e., X_{kj}. The new weight w'_{kj} is scaled from the old weight as follows:

$$w'_{kj} = w_{kj}(1 - X_{kj}). \qquad (4)$$

If edge-weight and bias are of opposite signs, the new weight remains the same, otherwise it is reduced.

The deserve value of a node represents the true trust a node deserves. We can use bias to define deserve. Deserve is the expected weight of an incoming link from an unbiased node. The deserve value depends on the quality of the inlinks and not on the quantity: deserve of a node with one quality inlink is equivalent to a node with many quality inlinks. This definition differs from the usual random-walk based methods where the number of inlinks matter. For each inlink, we remove the effect of bias from the weight and then we compute the mean of all inlinks. The deserve of a node j is given by

$$deserve(j) = \frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} \big(w_{kj}(1 - X_{kj})\big). \qquad (5)$$

The deserve value lies in the range $[-1, 1]$.

An example network of three nodes is shown in Figure 1. Note that, as in the case of real networks, not all nodes rank all other nodes. In fact, the graphs are only locally dense. Table 1 shows the final bias and deserve values of the nodes. The next section describes the algorithm to compute these values.

The above formulations of bias and deserve enable our model to handle the two main design problems in a trust-based network, namely, handling negative weights and distinguishing edge with zero weight (i.e., neutral opinion) from a no-connection (i.e., no edge).

3.2 Computing bias and deserve

In this section, we describe an algorithm to find the bias and deserve values of all nodes in the network. Note that the definitions as given in Eq. (1) and Eq. (5) are mutually recursive. Bias of a node depends on the deserve of its neighbors which in turn depends on the bias of their neighbors and so on. Thus, to solve this, we use the method of fixed-point iteration.

We denote the bias and deserve of node i at iteration t by $bias^t(i)$ and $deserve^t(i)$ respectively. We use values obtained from iteration t to compute the values for iteration $t+1$. In subsequent sections, we prove that the system converges to an unique solution irrespective of the initial values. We also analyze error bounds that help in determining the number of iterations.

From the initial values of bias and deserve, deserve values at the next iteration are computed for all nodes. Then, using those values, the bias values are re-estimated. Thus, $deserve^{t+1}(i)$ depends on $bias^t(*)$ (actually, $X^t(*)$), which in turn is computed using $deserve^t(*)$. Eq. (1) and Eq. (5) can be now re-written as:

$$deserve^{t+1}(j) = \frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} \left(w_{kj}(1 - X_{kj}^t) \right) \qquad (6)$$

$$bias^{t+1}(i) = \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(w_{ij} - deserve^{t+1}(j) \right) \quad (7)$$

Table 2 demonstrates how the values of bias and deserve are updated after each iteration for the example graph shown in Figure 1.

3.3 Running time analysis

The complexity of computing bias of a single node in a single iteration is given by $O(|d^o||d^i|)$, where $|d^*|$ denotes the average degree of all nodes. Although this looks expensive, the amortized cost can be reduced substantially to $O(m)$, where m is the total number of edges in the network.

At the start of a new iteration t, we first compute $deserve^t(i)$ for all nodes using Eq. (6). If m is the total number of edges, then computing deserve value for all nodes requires a total of $O(m)$ time. Using these deserve values, we then compute $bias^t(i)$ for all nodes using Eq. (7). This again requires an amortized total time of $O(m)$. Thus, for k iterations, the total running time is $O(km)$. Only if the graph is dense, the time grows to $O(kn^2)$. However, for most trust-based networks such as Slashdot, Epinions, IMDB, Youtube, etc., the number of edges are more or less linear in the number of nodes. Thus, the algorithm runs in $O(kn)$ time.

4. PROPERTIES OF THE ALGORITHM

In this section, we highlight certain desirable properties of the algorithm. We start with the proof that shows that the difference between the bias of a node at any iteration and its true bias is bounded, i.e., the error at any iteration of the algorithm can be estimated up to a constant. Next, we show that this leads to a convergence of the algorithm to a unique value of bias. Since deserve of a node can be expressed in terms of biases of other nodes, this implies that the deserve values exhibit similar properties as well.

4.1 Error bounds

We first prove the error bound for bias. Suppose, $b^\infty(i)$ is the true bias value of node i. Assuming that the value converges to an unique number (proved in later sections), this is also the value of bias after infinite iterations. For any iteration t, the difference in the values is, therefore, $|b^\infty(j) - b^t(j)|$. Without loss of generality, among all nodes, assume that node i shows the largest deviation of the bias value, i.e., $|b^\infty(i) - b^t(i)| = \max_j |b^\infty(j) - b^t(j)|$. We will prove that this deviation is bounded by an inverse exponential function of the number of iterations:

$$|b^\infty(i) - b^t(i)| \leq \frac{1}{2^t}$$

The above claim will be proved using mathematical induction and the following simple observation:

OBSERVATION 1. *The difference between the values of the effect of bias between any two iterations is at most 1.*

PROOF. Consider the auxiliary variable X_{**}^* that measures the effect of bias. Since its value lies between 0 and 1, the proof is immediate. □

THEOREM 1. *The maximum deviation of bias of a node at any iteration t from its true value is bounded by an inverse exponential function of t:*

$$|b^\infty(i) - b^t(i)| \leq \frac{1}{2^t}. \qquad (8)$$

PROOF. We use mathematical induction for the proof. The bias values can be expressed in the following way:

$$b^\infty(i) = \left| \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(w_{ij} - \frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} w_{kj}(1 - X_{kj}^\infty) \right) \right|$$

$$b^{t+1}(i) = \left| \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(w_{ij} - \frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} w_{kj}(1 - X_{kj}^t) \right) \right|$$

Basis: We first prove for $t = 1$.

$$|b^\infty(i) - b^1(i)|$$

$$= \left| \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} w_{kj}(X_{kj}^0 - X_{kj}^\infty) \right) \right|$$

$$\leq \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} |w_{kj}| \left| X_{kj}^0 - X_{kj}^\infty \right| \right)$$
$$[\because |x.y| \leq |x|.|y|]$$

$$\leq \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} \left| X_{kj}^0 - X_{kj}^\infty \right| \right)$$
$$[\because |w_{kj}| \leq 1]$$

$$\leq \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} 1 \right)$$
$$[\text{using Observation 1}]$$

$$= \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(\frac{1}{|d^i(j)|}.|d^i(j)| \right)$$

$$= \frac{1}{2}.$$

Induction step: We assume the bound to be true for $b^t(i)$,

i.e., the tth iteration. In the $(t+1)^{\text{th}}$ iteration,

$$|b^\infty(i) - b^{t+1}(i)| =$$

$$\left| \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} w_{kj}(X_{kj}^\infty - X_{kj}^t) \right) \right|$$

$$\leq \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} |X_{kj}^\infty - X_{kj}^t| \right)$$

$$= \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} |T| \right). \qquad (9)$$

We now analyze the quantity

$$T = \left| \max\{0, b^\infty(k) \times sign(w_{kj})\} - \max\{0, b^t(k) \times sign(w_{kj})\} \right|$$

Consider the following cases:

1. $b^\infty(k)$ and $b^t(k)$ are of opposite sign: One of the maximum terms in T is 0. Without loss of generality, assume $|T| = b^t(k)$. Since $b^\infty(k).b^t(k) < 0$, we have $|b^\infty(k) - b^t(k)| = |b^\infty(k)| + |b^t(k)|$. Therefore, $T \leq |b^\infty(k)| + |b^t(k)| \leq |b^\infty(k) - b^t(k)|$.

2. $b^\infty(k)$ and $b^t(k)$ are of same sign: Either $|T| = 0$ or $|T| = |b^\infty(k) - b^t(k)|$. Therefore, $|T| \leq |b^\infty(i) - b^t(i)|$.

From the above two facts, we can deduce that

$$|T| \leq |b^\infty(i) - b^t(i)| < \frac{1}{2^t}. \qquad \text{[induction assumption]}$$

Therefore, continuing from Eq. (9),

$$|b^\infty(i) - b^{t+1}(i)| \leq \frac{1}{2|d^o(i)|} \sum_{j \in d^o(i)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} \frac{1}{2^t} \right)$$

$$= \frac{1}{2^{t+1}}.$$

Thus, the error is bounded by an inverse exponential function of t. $\qquad \square$

4.2 Proof of convergence

Using the above error bounds, we now show that the value of bias of a node converges. If b is the true value of bias, then we need to show that after a particular number of iterations c, the value of bias assigned by our algorithm is as close to b as desired, i.e.,

$$|b - b^c(i)| \leq \epsilon$$

for some $\epsilon \to 0$. Using Eq. (8) in Theorem 1, we can set

$$c > \log_2 \left(\frac{1}{\epsilon} \right). \qquad (10)$$

Other iterative algorithms such as SimRank [8, 13] also exhibits similar convergence and error-bounding properties.

4.3 Proof of uniqueness

In this section, we show that the bias values obtained by the nodes at the end of the algorithm are unique. In other words, equation for bias has an unique solution. We prove the above claim using "proof by contradiction". If the equation for bias does not have an unique solution, then we can expect at least two different values of bias for a node.

Assume $b_1(i)$ and $b_2(i)$ to be the two bias values for node i. Denote the corresponding auxiliary variables by $X_{**}^{(n)}$ for $b_n(*)$. If $\delta(i) = b_1(i) - b_2(i)$ is the difference between the two values, let $M = \max_i |\delta(i)|$ be the maximum absolute value of difference for any node. The following theorem shows that $M = 0$.

THEOREM 2. *The difference between the two bias values of a node is zero, i.e., the node has a unique bias.*

PROOF. Without loss of generality, assume that node p displays the maximum difference in bias. Thus, the maximum difference M is:

$$M = |\delta(p)|$$

$$= \left| \frac{1}{2|d^o(p)|} \sum_{j \in d^o(p)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} w_{kj} \left(X_{kj}^{(2)} - X_{kj}^{(1)} \right) \right) \right|$$

$$\leq \frac{1}{2|d^o(p)|} \sum_{j \in d^o(p)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} |w_{kj}||X_{kj}^{(2)} - X_{kj}^{(1)}| \right)$$

$$[\because |x.y| \leq |x|.|y|] \qquad (11)$$

We now analyze the quantity

$$K = |X_{kj}^2 - X_{kj}^1|$$
$$= |\max\{0, b_1(k) \times sign(w_{kj})\} - \max\{0, b_2(k) \times sign(w_{kj})\}|$$

Consider the following cases:

1. $b_2(k).b_1(k) > 0$, $b_1(k).w_{kj} < 0$ and $b_2(k).w_{kj} < 0$: Here, $K = 0$.

2. $b_2(k).b_1(k) > 0$, $b_1(k).w_{kj} > 0$ and $b_2(k).w_{kj} > 0$: Here, $K = |b_1(k) - b_2(k)| \leq M$ (as M is the maximum difference).

3. $b_2(k).b_1(k) < 0$: Without loss of generality, assume $K = |b_1(k)|$. Also, $|b_1(k) - b_2(k)| = |b_1(k)| + |b_2(k)|$. Together, we have $M \geq |b_1(k) - b_2(k)| = |b_1(k)| + |b_2(k)| \geq K$.

Therefore, it is always the case that $K \leq M$. Using this in Eq. (11),

$$M = \frac{1}{2|d^o(p)|} \sum_{j \in d^o(p)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} |w_{kj}|K \right)$$

$$\leq \frac{1}{2|d^o(p)|} \sum_{j \in d^o(p)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} |w_{kj}|M \right)$$

$$\leq \frac{1}{2|d^o(p)|} \sum_{j \in d^o(p)} \left(\frac{1}{|d^i(j)|} \sum_{k \in d^i(j)} M \right)$$

$$= \frac{M}{2}.$$

The above inequality is possible if and only if $M = 0$ (remember that M must be non-negative). Hence, the bias of each node converges to a unique value. $\qquad \square$

5. EXTENSIONS

In this section, we discuss some extensions of the algorithm. This includes handling special requirements and applications.

5.1 Ranking

Ranking a node in a graph is an important problem and has attracted a wide attention since the introduction of *HITS* [10] and *PageRank* [14]. Most of these ranking techniques are variant of the eigenvector centrality measure [1]. The eigenvector centrality measure relies on the computation of principal eigenvectors based on certain features of the node, e.g., presence/absence of an edge, random walk probability, etc. The measure is grounded on the direct implication of the Perron-Frobenius theorem [4] which depends on the fact that the matrix is non-negative. However, by introducing negative edges, we can not use techniques that uses Perron-Frobenius theorem.

As we have already seen, deserve provides a natural answer to this question. It is based on the principle that a node receiving positive ratings from unbiased or negatively biased nodes can be trusted more than the node receiving positive links from biased (positive) nodes. Thus, the deserve value of a node can be directly used for ranking.

5.2 Apriori notion of trust

In many networks, it is not uncommon to have few trusted nodes [9]. For example, in a peer-to-peer network, there may be a few trusted peers, such as "verified accounts" on Twitter (`www.twitter.com`), "official groups" on LinkedIn (`www.linkedin.com`), etc. Here, we want to have deserve as 1 or close to 1 for such nodes. At the same time, we may want to mark untrustworthy nodes with deserve value close to -1. Formally, we want to assign deserve value between -1 and $+1$ to a particular set of nodes that are chosen apriori, but it should be done automatically and the network should not influence this rating significantly.

This can be accommodated easily in our model in the following way. Assume that a large number of hypothetical nodes (around $O(n^2)$ where n is the total number of nodes) are present. We refer this set of nodes as the *trustset*. We construct a clique out of it such that all the nodes in the trustset are connected to each other with a weight of $+1$. This ensures that the bias of nodes in trustset is 0. Now, we connect each node of this trustset to nodes for which we want to fix the deserve value. The weight of such an edge should be the desired deserve value. Due to the large size of the trustset, the deserve value of the apriori chosen nodes will get sufficiently close to the edge weights assigned to them. The computation is simple in the sense that the existing algorithm can be used without any modification. Using the symmetry of the nodes in the trustset, the bias and deserve values need to be calculated only for a single additional node.

5.3 Trust propagation

Trust propagation is another important problem in a trust-based network. In trust/mistrust propagation, we compute the trust score between two nodes that may not be connected directly. It was proposed in [5] to separate trust and mistrust matrix and then perform operations on them to obtain the transitive trust between two nodes. We observe that our model fits naturally in this framework. We can remove the bias from each link, redefine these matrices and continue with the rest of the operations as proposed in [5].

5.4 Link classification

Another related problem is that of link classification where

Dataset	Slashdot	Epinions
Nodes	82140	131828
Edges	549202	841372
Number of weakly connected components	1	5816
Number of strongly connected components	53599	88609

Table 3: Parameters of the datasets.

we try to predict the label of a directed link between two nodes to be either positive or negative. Machine learning techniques were proposed where various features based on degree of the vertex (both in and out for positive and negative links) and one-step neighborhood properties were used for classification [11]. We can create additional features by removing bias from each edge. Similarly, we can remove bias while constructing features based on one-step neighborhood properties. Additionally, we can add deserve of a node whose link is to be classified as another feature.

5.5 Effect of adversarial nodes

In any network, a very important issue is that of adversarial or malicious nodes. In a trust-based network, for example, an adversarial node can maintain its bias artificially at 0, even though it is not truthful. For example, it can rate nodes that deserve high positive values negatively and nodes that deserve high negative values positively, then it is possible that the bias which reflects the expected weight of an outgoing edge may come close to 0. Such behavior cannot be directly tackled by our model.

Instead of bias, however, we can capture the *variance* that reflects the deviation of a node with respect to the network using higher order functions. We expect an adversarial node with bias artificially maintained at 0 to have a high variance.

$$variance(i) = \frac{1}{|d^o(i)|} \sum_{j \in d^o(i)} (deserve(j) - w_{ij})^2$$

A more sophisticated form of attack can be from a *colluding group* where nodes in the group give good ratings to each other and bad to rest. Under such conditions, it is possible for these nodes to keep their biases artificially close to zero and at the same time maintain a high deserve value. Once more, our base model cannot handle this attack and new strategies need to be developed for such attacks.

6. EXPERIMENTS

In this section, we describe in detail the different experiments done on real trust-based networks using our measures of bias and deserve.

6.1 Datasets

We used two real datasets, Epinions and Slashdot, for performing the experiments. The datasets are available from the Stanford Large Network Dataset Collection (SNAP, `http://snap.stanford.edu`). Table 3 shows the various statistics about the datasets. The Slashdot dataset is quite well connected while the Epinions dataset is not so with lots of
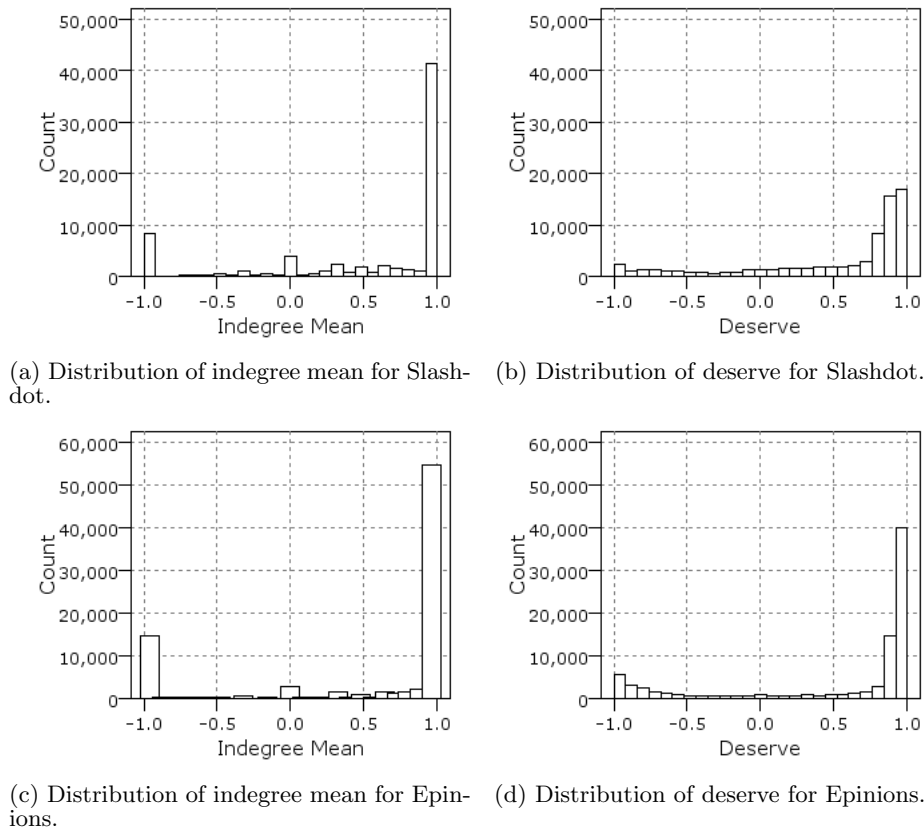
(a) Distribution of indegree mean for Slash-dot.

(b) Distribution of deserve for Slashdot.

(c) Distribution of indegree mean for Epin-ions.

(d) Distribution of deserve for Epinions.

Figure 2: Comparison of indegree mean and deserve values.

weakly connected components.[4] Many of the weakly con-nected components in the Epinions dataset are very small.

In a small graph, we can expect a high or low deserve value of a node, while the bias may still be zero. For example, consider a pair of nodes with just one directed edge. It is easy to see that the bias of one node will be zero and for the other node, it is undefined. Similarly, the deserve of one node will be the edge weight and for the other node, it is again undefined. While random-walk based techniques will give low scores to nodes in such components due to their connectivity, in our model, they may get high deserve value based on their inlinks. However, in general, bias and deserve values do not make much sense if the graph is very small.

6.2 Distribution of bias and deserve

The first set of experiments measure the distribution of bias and deserve values of the nodes. The distribution of deserve values are compared against that of the *indegree mean*. The indegree mean for a node is defined as the aver-age weight of incoming links. Figure 2 shows the histograms of the indegree mean and deserve value for both the datasets. In the datasets, the number of nodes with indegree mean 1 is very high; the count of nodes with indegree mean −1 is also significant in number. In both the datasets, count of nodes with indegree mean as 1 is very high. This is pri-

marily because more than 80% of the edges have positive weight.

However, the distribution of deserve is more smooth due to the removal of the effect of bias, especially for Slashdot. For Epinions, the distribution is not as smooth because of the presence of too many disconnected components of small size. In such small sized graphs, as previously discussed, if the edge weight is −1 or 1, the deserve values become close to that as well.

Figure 3 shows the distribution of bias values. It can be observed that the distribution is mostly concentrated around 0 and is positively skewed. This indicates that a significant fraction of the edges have positive weights. The bias values have little correlation with the mean of the outgoing edge weights.

6.3 Comparison of bias versus deserve

In this section, we see the effect of bias versus deserve. Typically, we expect a node with high prestige to be more trustworthy, i.e., have low bias. In case a node receives many positive inlinks but has outlinks that do not conform well with the opinion of other nodes, then we say that node has a high deserve score but has high bias.

In Figure 4, we plotted the histogram of deserve versus bias. We divided the bias into many bins of equal sizes, and then we computed mean of deserve lying in that bin. Here, we consider only those nodes that have both deserve and bias values (i.e., having both incoming and outgoing links).

In Figure 4(a), we can observe that nodes whose bias is

---

[4]A weakly connected component is a subgraph, where by replacing every directed edge with an undirected one, a path is obtained between every pair of vertices.
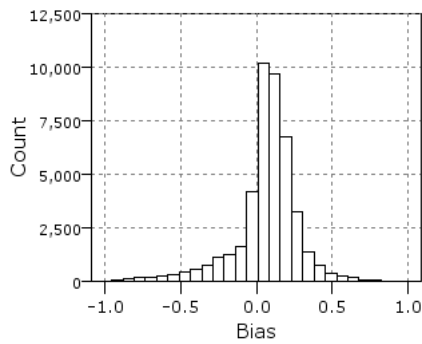
(a) Slashdot.
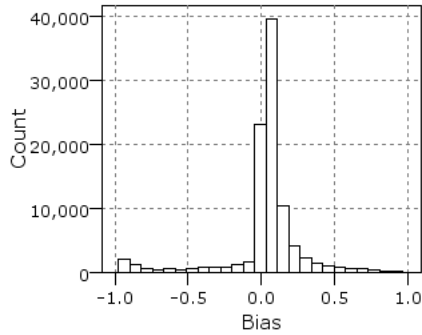
(b) Epinions.

Figure 3: Distribution of bias values.

(a) Epinions dataset.

(b) Slashdot dataset.

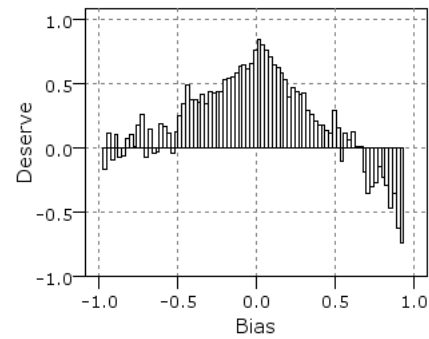**Figure 4: Comparison of bias and deserve values.**

close to zero have high deserve value, showing the strong relationship between the bias and prestige. However in Figure 4(b), nodes that have high bias (usually gives more positive links) also have high deserve value. It shows that a node giving positive ratings is respected as well.

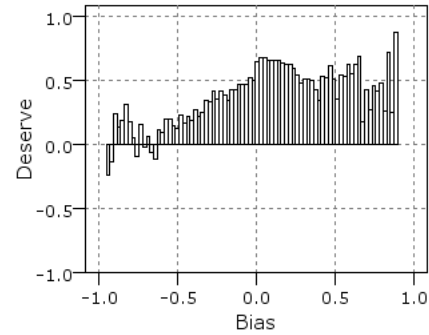## 6.4 Comparison of deserve with ranking

The next set of experiments compare the ranking of nodes using the deserve values against that produced by the popular ranking algorithms such as PageRank [14] and HITS [10]. Since HITS and PageRank cannot handle negative weights, we removed the edges with negative weights and then conducted the experiments. The caveat, of course, is the inability of HITS and PageRank to properly rank nodes having significant number of negatively-weighted links.

Figure 5 shows the comparison. Note that we have scaled the HITS and PageRank score by multiplying with 1000. One common trend we observe is that nodes with less prestige have low HITS and PageRank score, and those with high deserve have high score. This shows that the ranking determined by the deserve values conform to the perception that more popular nodes have more prestige.

However, towards the end (when deserve is also almost equal to 1), there is a drop in the scores. This is partly due to our model. Even if a node has few connections but has a quality inlink, it will attract high prestige. The same is not true for the other two algorithms. Moreover, the two datasets have large number of strongly connected components and most of them are very small in size. The nodes in these small components have high prestige, but due to their small sizes, they have low score.

6.5 Analysis of error and convergence

The next set of experiments analyze the properties of the algorithm in terms of error and convergence. We benchmark two kinds of errors for each of bias and deserve values.

Let $f^{(t)}(i)$ denote the value of a node i (can be either deserve or bias) at iteration t. The first kind of error is the maximum error for any node, i.e., it captures the worst-case scenario:

$$error(t) = \max_i |f^{(\infty)}(i) - f^{(t)}(i)|$$

The second kind of error is the mean error over all nodes:

$$error(t) = \frac{1}{|V|} \sum_{i \in V} |f^\infty(i) - f^{(t)}(i)|$$

where V denotes the set of all nodes.

Figure 6 shows how the error decreases for both bias and deserve for the two datasets. The figures are plotted only till 9 iterations as the value essentially converges after that. The plot also confirms the fact that the error decreases exponentially, as analyzed in Section 4.2.

6.6 Connection to balance theory

The final experiment shows how our model conforms with a well-established social theory—*balance theory*. In balance theory, four relationships are deduced: (i) "friend of a friend is a friend", (ii) "friend of an enemy is an enemy", (iii) "enemy of a friend is an enemy" and (iv) "enemy of an enemy is a friend". Interested readers are referred to [2, 6, 11, 12] for a more detailed discussion on status and balance theory with respect to social networks.
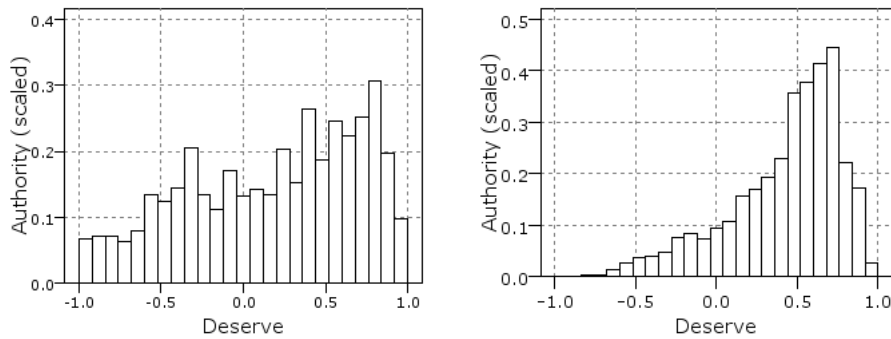
(a) HITS authority scores for Epinions dataset.

(b) HITS authority scores for Slashdot dataset.
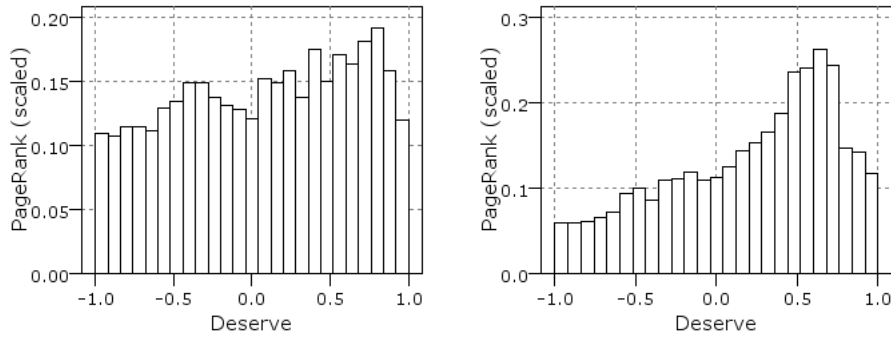
(c) PageRank scores for Epinions dataset.

(d) PageRank scores for Slashdot dataset.

Figure 5: Comparison of HITS and PageRank with deserve.

We consider an edge with a positive weight to represent friendship and similarly, an edge with negative weight to show animosity. To measure the conformity quantitatively, we utilize the concept of *transitive trust* [5]. It uses the four transitive relationships from the balance theory. For example, if i is a friend of j, and j is friend with k, then we expect i to be a friend of k. A direct connection from node i to node k is given by w_{ik}. For our experiments, this direct edge acts as the "ground truth". We compute transitive trust for each of the four relationships of the form $i \rightarrow j \rightarrow k$ given above for which a directed edge $i \rightarrow k$ exists. The weight of transitive trust from i to node k through node j is given by $w_{ij}w_{jk}$. If these two weights are close, then transitive trust matches with direct connection, and we say that the network adheres well with the balance theory.

Let S be the set of all occurrences of the form $i \rightarrow j \rightarrow k$ where the direct edge $i \rightarrow k$ exists. We compute the error using the following quadratic function:

$$\gamma = \frac{1}{4|S|} \sum_{\{i,j,k\} \in S} (w_{ij}w_{jk} - w_{ik})^2.$$

Here, the number 4 in the denominator acts as a normalizing constant.

In this experiment, we measure such adherence to the balance theory. Initially, we compute the conformity of the graph using the above equation on the original network. Then, we remove the bias from each edge using Eq. (2), and re-compute the error.

Table 4 shows the errors for both the datasets. Here, γ_o denotes the error for the original graph and γ_b denotes that

Relationship	Epinions		Slashdot	
	γ_o	γ_b	γ_o	γ_b
friend-friend-friend	0.01	0.01	0.02	0.02
friend-enemy-enemy	0.53	0.36	0.59	0.36
enemy-friend-enemy	0.40	0.31	0.32	0.24
enemy-enemy-friend	0.60	0.23	0.45	0.19

Table 4: Error of conformity with balance theory.

after removing the bias. Removing the bias always decreases the error except in the case of the relationship of the form "friend of a friend is a friend" where the error is negligible to start with. Thus, it can be concluded that capturing the bias of a node and taking action to remove it improves the conformity of the network with the balance theory.

7. CONCLUSIONS

For many applications involving trust-based networks, it is crucial to assess the prestige or bias of a node. Whereas bias denotes the propensity of a node to trust/mistrust its neighbours, prestige represents the true trust a node deserves. In this paper, we have proposed an algorithm to compute the bias and prestige of nodes in networks where the edge weight denotes the trust score. Unlike most other graph-based algorithms, our method works even the edge weights are not necessarily positive. The algorithm exhibits several desirable properties: it is efficient ($O(km)$ running time, where k is the number of iterations and m is the total number of edges in the network), the bias and prestige

(a) Bias for Epinions dataset.

(b) Bias for Slashdot dataset.

(c) Deserve for Epinions dataset.
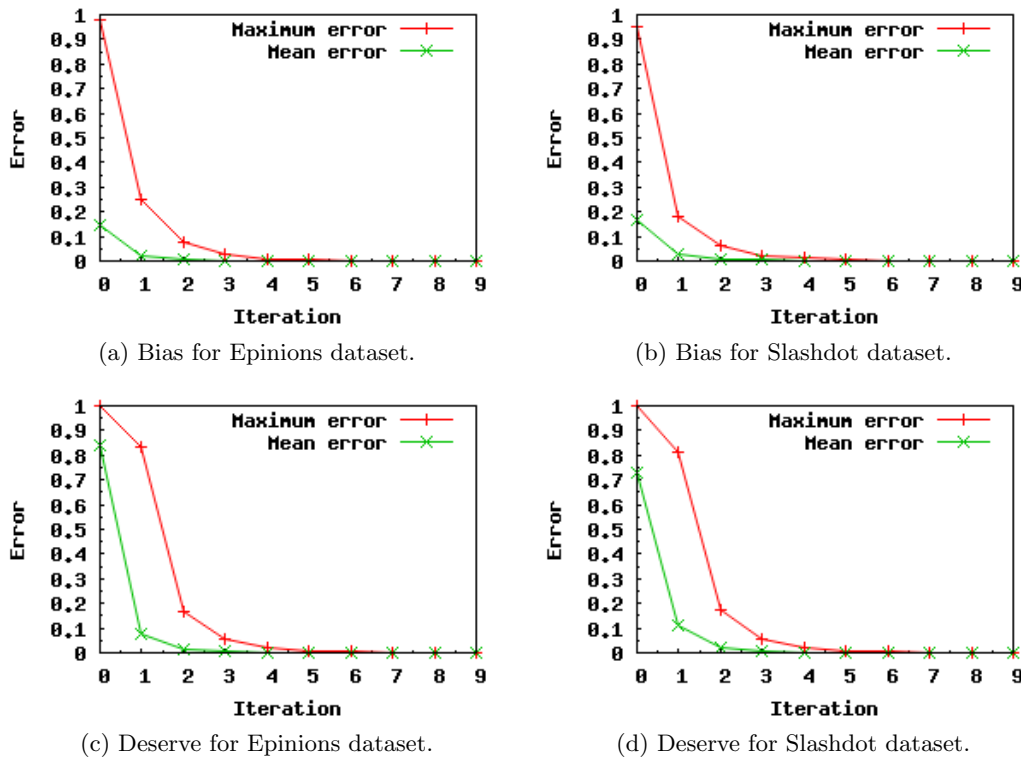
(d) Deserve for Slashdot dataset.

Figure 6: Exponentially decreasing errors.

values converge fast to unique values, and the error at any iteration is bounded. Experiments showed that our model conforms well to other graph ranking algorithms and social theories such as the balance theory. The algorithm, however, is prone to adversarial nodes and colluding groups. In future, we aim to tackle these and other forms of attacks. Also, the applicability of our method in a distributed setup will be explored in detail.

Acknowledgments: We thank the anonymous reviewers for their valuable comments that helped in improving the paper immensely. We also thank Praveen Reddy, Pavan Reddy, and Sumeet Khurana for stimulating discussions and incisive inputs.

8. REFERENCES

[1] P. Bonacich. Factoring and weighting approaches to status scores and clique identification. *J. Mathematical Sociology*, 2:113–120, 1972.

[2] D. Cartwright and F. Harary. Structural balance: A generalization of Heider's theory. *Psychological Review*, 63:277–293, 1956.

[3] C. de Kerchove and P. V. Dooren. The PageTrust algorithm: How to rank web pages when negative links are allowed? In *SDM*, pages 346–352, 2008.

[4] G. Golub and C. F. V. Loan. *Matrix Computations*. Johns Hopkins University Press, 1996.

[5] R. V. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *WWW*, pages 403–412, 2004.

[6] F. Heider. Attitudes and Cognitive Organization. *J. Psychology*, 21:107–112, 1946.

[7] B. D. Hughes. *Random Walks and Random Environments*. Oxford University Press, 1996.

[8] G. Jeh and J. Widom. Simrank: A measure of structural-context similarity. In *KDD*, pages 538–543, 2002.

[9] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The Eigentrust algorithm for reputation management in P2P networks. In *WWW*, pages 640–651, 2003.

[10] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[11] J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg. Predicting positive and negative links in online social networks. In *WWW*, pages 641–650, 2010.

[12] J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg. Signed networks in social media. In *CHI*, pages 1361–1370, 2010.

[13] D. Lizorkin, P. Velikhov, M. Grinev, and D. Turdakov. Accuracy estimate and optimization techniques for SimRank computation. *Proc. VLDB Endowment*, 1(1):422–433, 2008.

[14] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[15] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *ISWC*, pages 351–368, 2003.