Computing Trusted Authority Scores in Peer-to-Peer Web Search Networks*

Josiane Xavier Parreira[†] Debora Donato[‡] Carlos Castillo[‡] Gerhard Weikum[†]
[†] Max-Planck Institute for Informatics
Saarbrcken, Germany
{jparreir,weikum}@mpi-inf.mpg.de
[‡] Yahoo! Research
Barcelona, Spain
{debora, chato}@yahoo-inc.com

ABSTRACT

Peer-to-peer (P2P) networks have received great attention for sharing and searching information in large user communities. The open and anonymous nature of P2P networks is one of its main strengths, but it also opens doors to manipulation of the information and of the quality ratings.

In our previous work (J. X. Parreira, D. Donato, S. Michel and G. Weikum in VLDB 2006) we presented the JXP algorithm for distributed computing PageRank scores for information units (Web pages, sites, peers, social groups, etc.) within a link- or endorsement-based graph structure. The algorithm builds on local authority computations and bilateral peer meetings with exchanges of small data structures that are relevant for gradually learning about global properties and eventually converging towards global authority rankings.

In the current paper we address the important issue of cheating peers that attempt to distort the global authority values, by providing manipulated data during the peer meetings. Our approach to this problem enhances JXP with statistical techniques for detecting suspicious behavior. Our method, coined TrustJXP, is again completely decentralized, and we demonstrate its viability and robustness in experiments with real Web data.

Categories and Subject Descriptors: H.4 [Information Systems Applications]: Miscellaneous; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval; H.3.4 [Information Storage and Retrieval]: Systems and Software – *Distributed Systems*

General Terms: Algorithms, Security.

Keywords: P2P Networks, Trust, Decentralized PageRank

*Partially supported by the EU within the 6th Framework Programme under contract 001907 "Dynamically Evolving, Large Scale Information Systems" (DELIS).

AIRWeb '07, May 8, 2007 Banff, Alberta, Canada.

1. INTRODUCTION

Peer-to-peer (*P2P*) systems have emerged in the last years as a paradigm for storing and sharing information [31], using the computing and storage power of a myriad of relatively inexpensive computers (usually desktop PCs), to create large scale systems that would be very expensive or even unfeasible to build in a centralized manner.

So far the main application of such systems in practice is sharing files that are stored locally on each computer. A natural extension of that is global-scale P2P search systems, in which each peer crawls a sub-set of the Web, and queries are solved very efficiently in a distributed manner [32, 4]. The advantages are better scalability of course but also more privacy and resistance to censorship, among others.

The **PageRank algorithm** [26, 20] defines an authority score that considers hyperlinks on the Web as recommendation. For computing PageRank, global knowledge of the network is necessary. In our previous work [27, 28] we developed the **JXP algorithm** for computing decentralized PageRank-style authority scores in a P2P network. In the algorithm, each peer is responsible for computing the score for the pages it stores locally, and through a series of meetings, each peers gets an increasingly accurate approximation of the scores of the local pages without needing to compute scores for all pages in the network, which is good, as we are assuming that no peer can hold the entire network locally.

Since high authority scores can bring benefits for peers, it is expected that malicious peers would try to distort the correctness of the algorithm, by providing different (usually higher) scores for their local pages. In general P2P networks are vulnerable to malicious agents that can cheat in order to get more benefits, and need **reputation systems** [23] in place to be able to operate properly.

Our contribution: in this work we present a trust model that integrates decentralized authority scoring like JXP with an equally decentralized reputation system. Our approach is based on anomaly detection techniques, the allow us to detect a suspicious peer based on the deviation of its behavior from some common features that constitute the usual peer profile.

The method combines an analysis of the authority score distribution and a comparison of score rankings for a small set of pages. The original JXP algorithm is then extended to limit the impact of malicious peers. We call this extended version **TrustJXP**. This algorithm is completely decentralized, does not require storing any additional information about other peers, can operate anonymously, and involves only local computations. Also, TrustJXP does not require

any form of cooperation among peers, and the system works as long as the fraction of well behaving peers is significantly larger than the fraction of cheating peers.

We present the results of experiments that show how JXP scores are affected by the presence of malicious peers under different attack models, and how our TrustJXP method can help to avoid the distortion caused by the manipulated data of the cheating peers.

The rest of the document is organized as follows. Section 2 discusses related work. A brief review of the JXP algorithm is presented in Section 3. The *TrustJXP* algorithm, including our trust model, our methods for detecting anomalous behavior, and how to combine JXP with the trust model, is presented in Section 4. Experimental results are given in Section 5. Section 6 presents ideas for future work.

2. RELATED WORK

PageRank [5] and HITS [17] are two well known methods for link analysis on the Web graph, which can be used to derive authority scores for Web pages that reflect query-independent importance and community endorsements. [21] provides an in-depth surveys of follow-up and related work on link analysis.

A general framework for different types of trust and distrust propagation in a graph of Web pages, sites, or other entities is introduced in [13]. Detecting and combating Web link spam is a special, but highly important case of reasoning about trust and distrust on the Web. [14] gives a taxonomy of the most important Web spamming techniques. In this paper we do not deal with Web spam in that sense; Web spam detection systems are independent from the work we present here.

The problem of untrustworthy or manipulated content is felt even more in a P2P environment [31]. The complete lack of accountability of the resources that peers share on the network offers an almost ideal environment for malicious peers and forces the introduction of reputation systems that help to assess the quality and trustworthiness of peers. In [23], the authors present a complete overview of the issues related to the design of a decentralized reputation system. EigenTrust [16] is one of the first methods introduced to assign a global trust value to each peers, computed as the stationary distribution of the Markov chain defined by the normalized local trust matrix C where c_{ij} is the local trust value that a peer i assign to a peer j. A similar approach is presented in [30]. The authors describe an anomaly detection procedure that analyzes peer activity on the network in order to identify peers whose behavior deviates from peer profile. The peer profile is based on a number of parameters like the time and the duration of each connection, the number of bytes uploaded, and the number of query requests, but they require that peers identify themselves. In [25] the authors present SeAl, an infrastructure designed for addressing the problem of selfish peer behavior. It works by combining a monitoring/accounting subsystem, an auditing/verification subsystem, and incentive mechanisms.

Distributed link analysis in P2P networks has recently received great attention [15, 33, 28, 27, 8] including our own work. However, all of these methods, with the exception of our own JXP work [28, 27], assume that the global graph is partitioned into disjoint fragments across Web hosts or peers. In contrast, our JXP method emphasizes peer autonomy and allows each peer to host an arbitrarily compiled graph fragment at its discretion. This way, the fragments of different peers may overlap in an arbitrary manner, and this complicates the computation of global authority scores. Section 3 gives details on how this issue is addressed by JXP.

Mathematically, all these methods employ Markov-chain state lumping, aka. Markov-chain aggregation/disaggregation techniques [24]. Various algorithms have adopted such techniques for accelerating PageRank-like computations in centralized settings and for incremental updating of authority scores [6, 19, 7].

The analysis of social networks is another related area that has recently become very popular. The tasks considered there are mining social relationships and interactions, social tagging activities, community substructures, and other aspects of such networks (e.g., [10, 18, 11]).

In PeerTrust [34], there is a feedback mechanism in place that after each transaction, measures the level of satisfaction of a peer with respect to the transaction. These are explicit ratings that are stored in a distributed fashion and a time-window is applied to a peer does not gain much from behaving correctly from some time, and then starting to misbehave. The ratings are exchanged and a credibility factor is applied to each exchange of ratings about other peers. This credibility may be based on the rating of the peer that is generating the ranking, or in how similar his ratings are to what the receiving peer has observed (this is anomaly detection in the sense used in this paper). This requires to keep an identity at each peer, while in our system, all peers are anonymous.

In [1], a mechanism able to approximate EigenTrust [16] is proposed. The authors use mechanism design in order to develop a non-manipulable trust system, i.e., a system in which each peer has no incentive to manipulate its recommendation. The system is based on a cycling partitioning, that is, peers are divided into groups forming topologically a ring. Forcing each peer to download only from its successors in the ring, the trust system is proved to approximate EigenTrust with an error that decreases exponentially in the number of groups.

3. THE JXP ALGORITHM

The JXP algorithm dynamically computes an approximation of PageRank scores based on directed graphs that are arbitrarily spread over autonomous peers in a P2P collaboration. Each peer periodically, and independently of other peers, performs local PageRank score computations on its local graph fragment, where the local graph is augmented by a *world node* that represents the locally unknown part of the global graph. Mathematically, this is a state lumping or aggregation technique for the underlying Markov chain.

JXP uses meetings between pairs of peers for mutual exchange of information about their local graph fragments, to continuously improve each peer's knowledge about its world node. The meetings take place asynchronously, without any central planning, and in the basic version of the algorithm the peers for the meeting are picked uniformly at random.

At each peer, its *world node* is constructed by combining all edges from local pages (or other kinds of graph nodes) that point to external (i.e., non-local) pages into edges to the world node. Analogously, when a peer learns (by means of meeting another peer) about an edge from a non-local page to a local page, a corresponding edge from the world node to that local page is added to the local graph. Additionally,

the world node has a self-loop edge that represents all links among external documents.

During a meeting, a peer adds all relevant information given by the other peer into its world node and locally recomputes the local JXP scores by a standart PageRank power iteration on the local Web graph augmented by the world node.

Throughout these meetings, the JXP scores that are locally maintained at each peer for its graph fragment converge to the global authority scores that would be derived from the entire global graph. Figure 1 illustrates a meeting between two peers. Details about the JXP algorithm, including the proof of convergence, can be found in [27].

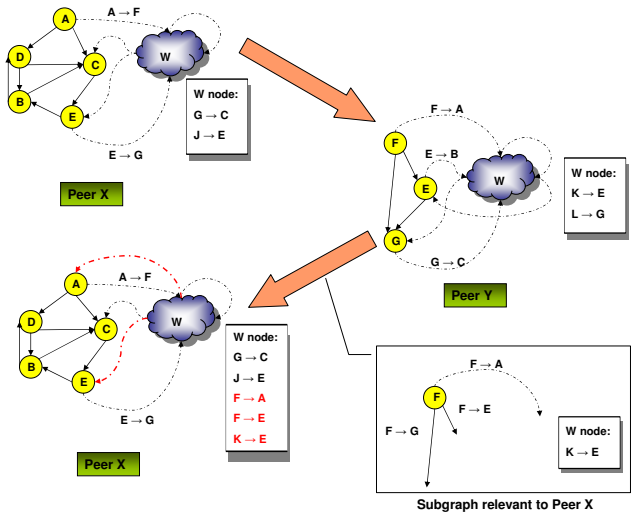


Figure 1: Improving Knowledge by Peer Meetings

The convergence rate of JXP scores depends on the choice of peers for meetings. Convergence can be accelerated by preferring to meet with nodes that have many edges to pages known in the local graph of the meeting initiator. [27] shows an efficient approach to identify such "good" peers based on statistical synopses and caching, without unduly increasing network bandwidth consumption.

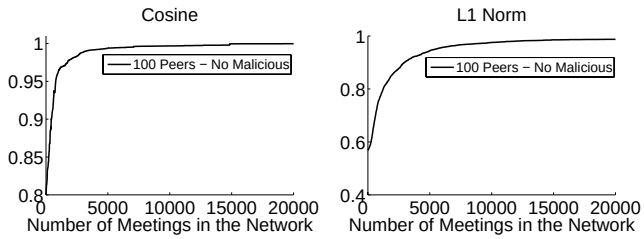
Figure 2 shows the convergence speed of JXP with 100 peers, to show the typical behavior of the scores generated by the JXP algorithm. The experimental scenario is described later in section 5. The curve for the L1-norm illustrates that the JXP scores are upper-bounded by the global PageRank scores, which agrees with the analysis given in [27].

4. THE TRUSTJXP ALGORITHM

This section describes our extension to JXP. There are many possible forms of attacks or manipulations in a P2P network. In this paper we deal with the group of attacks where peers want to distort the authority scores being computed by JXP, by reporting false scores for a set of pages at the meeting phase. We have modeled two general types of attack:

1. A cheating peer can report a higher score for a subset of its local pages, in an attempt to get its pages into high positions in the global ranking that JXP peers may perceive. In this form of manipulation, the peer would boost pages at the "expense" of reducing the total weight of its world node (giving lower score mass to all non-local pages).

2. A cheating peer can manipulate the scores of its local pages by modifying the scores, not necessarily increasing them. This way, some pages are boosted while others are downgraded. The score mass of the world node would stay unchanged. If the cheating peer wants to maintain the statistical distribution of the scores among local pages, he can just permute the scores of its local pages.



(a) (b)

Figure 2: JXP performance with 100 peers, all of them honest: (a) cosine similarity (b) L1 norm. With only about 20-30 meetings per peer the resulting scores are already close to the actual values.

In the following subsections we describe how we detect and eliminate or compensate the effects of these two forms of attack, or even from combined attacks that use both techniques.

4.1 Malicious Increase of Scores

As we mention earlier, for a peer, deceiving other peers to make them believe that he posseses documents with high authority scores can be benefical. To deter this kind of manipulation we use anomaly detection on the distribution of the scores reported by a peer. The hypothesis is that the local distribution of scores should resemble the global distribution of scores after a few iterations.

The justification for this hypothesis stems from the way the local graph fragments are built. In our P2P model, each peer gathers its data by performing focused Web crawls, starting from particular seeds and possibly using a thematically focused crawler in order to harvest pages that fit with the interest profile of the corresponding user (or user group). Given that the Web graph is self-similar [9, 2] under several partitioning schemes (topically, geographically, per-domain, etc.), the properties of the small graph fragment that a peer eventually compiles should be statistically indistinguishable from the properties of the full Web graph as seen by a centralized crawler.

Storing the typical profile: we keep a representation of the distribution of the scores in the system in histograms. Since scores are expected to follow a power-law distribution, we make the boundaries of the buckets also exponential, similar to what is used in [3]. More precisely, the bucket number i will have the boundaries

$$bucket(i) = [a \cdot b^{i-1}, a \cdot b^{i}) \ .$$

The precise values for a and b will depend on the distribution of PageRank values in the observed sample, which in turn depends basically on the number of pages in the entire network and the dampening factor for PageRank. The dampening factor for the computation is shared among all the nodes. The number of pages (at least its order of magnitude) can be initialized with an estimation that can be improved after a few meetings. The choice of the buckets is not relevant, as long as not all the pages fall in the same bucket. It is not necessary that all peers use the same buckets, and in the worst case, a peer can re-initialize its histograms with new parameters at any time (at the cost of slowing down its convergence). In our experiments we used $a = 0.005$ and $b = 0.3$ because those values cover the range of usual values for PageRank in our setting.

We create, at each peer, a histogram which is initially filled with the initial JXP scores of local pages. After each meeting, the distribution of the local scores of the other peer is added to the local histogram. We introduce a *novelty factor* to account for the dynamics of the scores across the meetings. Given the local histogram at meeting t, H^t, and the score distribution from the other peer D, the local histogram at meeting $(t+1)$ is updated as follows:

$$H^{(t+1)} = (1-\rho)H^t + \rho D$$

where the parameter ρ represents how much importance we give to the new values, and the precise choice only affects how fast the peer learns the global distribution of scores, which in turn changes the convergence speed for the scores in that particular peer. In our experiments we set $\rho = 0.6$.

Since we rely on the assumption that the number of honest peers is significantly bigger than the number of dishonest ones, we expect that the histogram always reflects the true distribution of the honest peers. If dishonest peers are reporting higher scores for some of their local pages, the distribution of their local scores would no longer resemble the distribution expected over all peers. Therefore, a comparison against the accumulated local histogram should give an indication of this deviation from normal behavior.

Detecting anomalies: given the accumulated local histogram of a peer i, H_i, and the histogram containing the scores distribution of another peer j, D_j, we want to compute how much D_j deviates from H_i. Since the distributions are expected to be similar [9], we believe that the distributions of honest peers should be very close to each other, and if D_j differs from H_i by a large margin, it is an indication that the peer is cheating about its local scores. For comparing the two distributions we have chosen the *Hellinger Distance*, which is defined as [22]:

$$HD_{i,j} = \frac{1}{\sqrt{2}}[\sum_k (\sqrt{H_i(k)} - \sqrt{D_j(k)})^2]^{\frac{1}{2}} \quad (1)$$

where, k is the total number of buckets and $H_i(k)$ and $D_j(k)$ are the number of elements at bucket k at the two distributions, both normalized by the total number of elements at each distribution. The factor $1/\sqrt{2}$ is introduced to normalize the range of possible values.

As an alternative to the Hellinger Distance, we also tried the χ^2 goodness-of-fit test or information-theoretic measures such as Kullback-Leibler divergence. The Hellinger Distance gave the most robust results, but the other methods worked fine, too. Also, since it is a metric, the Hellinger Distance has nice properties, besides the fact that values can be normalized, which makes it easier to be combined with other measures.

4.2 Malicious Permutation of Scores

Our histograms comparison is inherently unable to detect a cheating peer that reports a permutation of the current scores of its local pages, since both distributions would be statistically indistinguishable. For detecting this type of attack we use a different technique. In our experimental studies of the JXP algorithm, we have observed that, after a few meetings, although the local JXP scores do not correspond yet to the global authority scores, the relative ranking orderings of their local pages are already very close to the actual ordering. This is also exploited in [34] for a different task (testing if a feedback given by a peer makes sense).

What we do is to compare the rankings given by the two peers in a meeting for those pages fall into the overlap of both local graphs, and we measure what we refer to as the *Tolerant Kendall's Tau Distance* between those rankings, defined below.

We use a relaxation of Kendall's Tau since we need to tolerate small fluctuations in the scores of pages with almost identical global authority. To this end, we discount page pairs that have different relative orders in the two rankings if their score differences are below a tunable threshold Δ. In this case, we consider the page pair as incomparable and their rank order as arbitrary.

Our Tolerant Kendall's Tau Distance is therefore defined as:

$$\begin{aligned} K'_{i,j} = &|(a,b) : a < b \wedge score_i(a) - score_i(b) \geq \Delta \\ &\wedge \tau_i(a) < \tau_i(b) \wedge \tau_j(a) > \tau_j(b)| \end{aligned} \quad (2)$$

where $score_i(a)$ and $score_i(b)$ are the scores of pages a and b at peer i, $a < b$ refers to the lexicographical order of page URLs (to avoid double-counting), τ_i and τ_j are the rankings of pages in the overlapping set at peers i and j, and Δ is our tolerance threshold. A good choice of Δ can be derived from the dampening factor of the underlying PageRank model as follows. We consider as our threshold the minimum amount of authority mass one page can have, which is the score mass earned from the random jumps. Therefore, at each peer, Δ is set to

$$\Delta = \frac{(1-\epsilon)}{N} \quad (3)$$

where $1-\epsilon$ is the usual damping factor of PageRank $1-\epsilon = 0.15$ and N is the total number of pages in the network, or an approximation of it.

This approach assumes that whenever two peers meet, there is a sufficient overlap between their locally known pages to make this comparison statistically meaningful. In an application where such overlaps cannot be guaranteed with high probability, we would have to add artificial overlaps as "honesty witnesses". One way of designing such an additional set of witness pages would be to randomly draw a set of sample URLs and disseminate them in the network by an epidemic protocol or using the overlay network of the P2P system. This set of witnesses should be changed periodically to counter adaptation strategies of malicious peers.

4.3 Computing Trust Scores

We now use our trust model to assign trust scores to peers. The method is totally decentralized: each peer is responsible for assigning (its perception of) trust scores to other peers, based on interactions with them. During a meeting, peers exchange the scores of their local pages. These scores are used for computing both histograms divergence and the rank divergence for the overlapping pages. These two measures will determine the level of trust that should be given to the peer. A new trust score is assigned to a peer at every meeting, as scores are changing.

It is important to emphasize that our technique relies on comparing only the scores of the local pages without any further information about peer identity. As a matter of fact, the histograms allow each peer to maintain a global view of peers conduct without keeping track of the personal history of each of them. This characteristic makes the algorithm resilient to Sybil attack, since the identity of the peers is never considered by the anomaly detection techniques we integrate in TrustJXP.

For combining histograms divergence and rank divergence into one single trust score, we take a conservative choice: we always take the lower level of trust among the two measures. Thus, we define the trust score that a peer i gives to a peer j as

$$\theta_{i,j} = min(1 - HD_{i,j}, 1 - K'_{i,j}) \tag{4}$$

This is the trust score that will be used in the TrustJXP algorithm, which is presented in the following section. As shown in the experiments in section 5, the distribution of HD and K' are comparable and thus this simple combination makes sense in practice.

4.4 Integrating Trust Scores and JXP Scores

The idea of TrustJXP is to incorporate the trust measure θ into the JXP algorithm for computing more reliable and robust authority scores. Our approach is to use the trust measure at peer meetings when combining the scores lists. For combining the scores lists, in the original JXP algorithm, whenever a page is present in both lists, its score will be set to the average of both scores or the maximum of the two scores, depending on the approach chosen. More formally, the score of page i in the updated score list L' is given by

$$L'(i) = \begin{cases} (L_A(i) + L_B(i))/2 & \text{if "average"} \\ max(L_A(i), L_B(i)) & \text{if "maximum"} \end{cases} \tag{5}$$

where $L_A(i)$ and $L_B(i)$ are the scores of page i at the two peers. If the page is not in one of the lists, its value is set to zero on the respective list.

For the TrustJXP algorithm, the contribution of the scores from the other peer are weighted based on how much that peer is considered to be trustworthy. The score of a page i in the updated scores list is now defined as

$$L'(i) = \begin{cases} (1 - \theta/2) * L_A(i) + \theta/2 * L_B(i) & \text{if "average"} \\ max(L_A(i), \theta * L_B(i)) & \text{if "maximum"} \end{cases} \tag{6}$$

After combining the scores lists, the JXP algorithm proceeds as usual: the relevant information learned from the other peer is added to the world node, and a PageRank computation is performed, leading to new JXP scores.

5. EXPERIMENTAL RESULTS

We conducted preliminary experiments on a small Web collection. Experiments with larger Web graphs are underway. Our Web collection was obtained in January 2005, using the Bingo! focused crawler [29]. We first trained the focused crawler with a manually selected set of seed pages for a Web crawl, and the fetched pages were automatically classified into one of 10 pre-defined topic categories like "sports", "music", etc.

We created 100 peers and assigned pages to these peers by simulating a crawler in each peer, starting with a set of random seed pages from one of the thematic categories and following the links and fetching pages in a breadth-first manner, up to a certain predefined depth. The category of a peer is defined as the category to which the initial seeds belong. During the crawling process, when the peer encounters a page that does not belong to its category, it randomly decides to follow links from this page or not with equal probabilities. In our setup, these 100 peers will correspond to the trustful peers and each one will hold its full graph fragment that was assigned to it. Thus, in the absence of malicious peers, our JXP scores can converge to the global PageRank scores of the complete graph for this Web collection, with a total of $N = 134,405$ pages and 1,915,401 links.

Different amounts of malicious peers were introduced in the system. Malicious peers perform meetings and local PageRank computations like any normal peer. The difference is that, when asked by another peer for its scores list, a malicious peers will lie about the scores of its local pages, according to one of the attack models described in the previous section: reporting a higher score for all or some of its local pages, or permuting the scores among its pages.

In these experiments, peers do not change their behavior during the TrustJXP computation; for example, if a peer chooses to permute its scores for the first meeting, it will do so for all subsequent meetings and it will apply always the same permutation. Having an inconsistent behavior (cheating sometimes, cooperating sometimes) does not help the peer as in [34], as there is no identity and no history is kept for each peer. Reporting good scores does not help the peer in building a reputation, but helps the system by accelerating its convergence towards the real values.

Evaluation metrics: we used four metrics, two over the top-1,000 pages (roughly top 1% of the pages) and two over the entire set. The idea is that the algorithm must give a good approximation of the actual scores for all pages, but particularly for those of high interest to the users. Over the top-1,000 pages we used Spearman's footrule distance and the Linear Score Error. Spearman's footrule distance is defined as [12]:

$$F(\sigma_1, \sigma_2) = \sum_{i=1}^{k} |\sigma_1(i) - \sigma_2(i)| \tag{7}$$

where $\sigma_1(i)$ and $\sigma_2(i)$ are the positions of the page i in the first and second ranking. In case a page is present in one of the top-k rankings and does not appear in the other, its position in the latter is considered to be $k + 1$. Spearman's footrule distance is normalized to obtain values between 0 and 1, with 0 meaning that the rankings are identical, and 1 meaning that the rankings have no pages in common. The Linear Score Error is defined as the average of the absolute

difference between the JXP score and the global PageRank score over the top-k pages in the centralized PageRank ranking. Additionally, we have computed the L1 norm for the JXP ranking vector and the cosine similarity between the vectors with local JXP and global PageRank ranks. Since the scores are normalized, the L1 norm for the global PageRank vector is always 1.

5.1 Effect of malicious peers in JXP

Starting from the setup with 100 peers shown in Figure 2 we then introduced 10 cheating peers, and later 50 cheating peers. Each of the malicious peers picks one of the following attacks:

- Report local JXP scores that are twice as their true values for all of their local pages.
- Report these falsely boosted scores for only half of their local pages (drawn randomly but used consistently throughout all meetings).
- Report permuted scores list (with a consistent permutation, otherwise it could be easily detected by two succesive meetings, as the peers are anonymous).

The results of this experiment are shown in Figure 3.

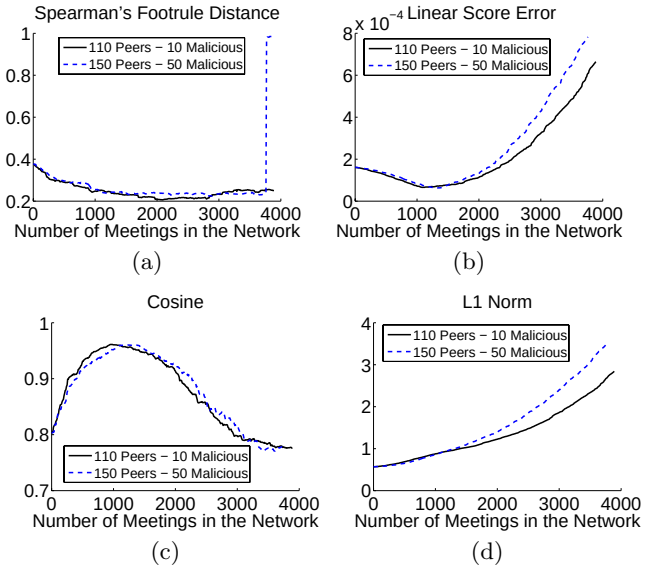


Figure 3: Impact of malicious peers in JXP.

With the introduction of malicious peers and without any trust mechanism, the JXP scores no longer converge to the true global PageRank values. The mathematical analysis of the JXP algorithm given in [27] proved that the JXP scores are upper-bounded by the true PageRank scores. With malicious peers reporting scores that are higher than the true ones, there is no bound for the scores. This effect can escalate: it distorts the world-node score and the transition probabilities from the world node to the local pages, and can even lead to a negative transition probability for the word node's self loop. At this point, scores start becoming undefined; this is the point where the linear-error, cosine, and L1-norm curves stop.

Since the JXP scores are no longer upper-bounded in the presence of malicious peers, it is not a good approach, when combining the score lists of two peers, to take the higher one of the two scores for pages that appear in both lists. We then ran another experiment where the lists are combined by taking the average of such scores. As we see in Figure 4, this method slows down the effects of malicious peers but cannot completely prevent the distortion. So a defense mechanism against cheating peers is indeed crucial.

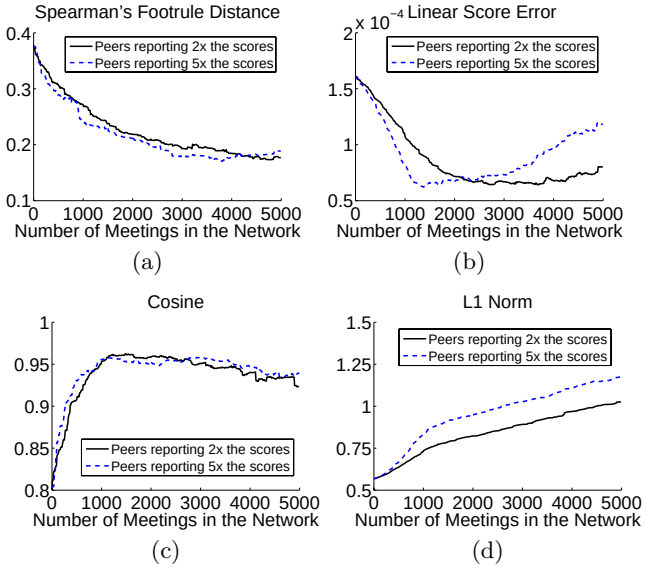


Figure 4: Impact of malicious peers with JXP and a naive defense mechanism: averaging the scores instead of picking the maximum.

5.2 Effect of malicious peers in TrustJXP

We proceeded by testing our trust model, measuring both histograms divergence and rank divergence for the overlapping pages. We again introduced 50 cheating peers, but now all peers performed the same type of attack. Figures 5 show the Hellinger Distance and the Tolerant Kendall's Tau for the case where cheating peers report scores five times higher than the true ones, and for the case where peers permute their scores, respectively.

The results confirm our hypothesis that comparing histograms can be an effective indicator of cheating behavior with increased scores. We can also see that, when scores are permuted, the histogram approach does no longer work, and the rank divergence provides a better indication of such malicious behavior.

Finally, we repeated this experiment with 50 malicious peers, and used our new TrustJXP method for computing local scores. The histograms and rank divergence, as well as the final TrustJXP scores are shown in Figure 6.

For comparison on how effective a trust model could be, we also simulated a best case, with an oracle-based defense mechanism that knows the class of each peer (honest vs. cheating) beforehand. The results for TrustJXP versus JXP and the oracle-based system are shown in Figure 7. For most of the metrics, our TrustJXP method is fairly close to the ideal case in terms of detecting and compensating malicious peers.

In the figure we can also see that JXP algorithm works good at the beginning as it moves in large steps towards

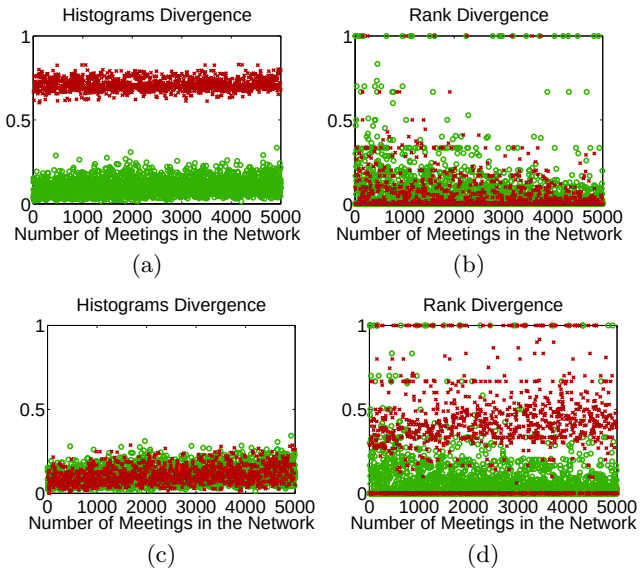
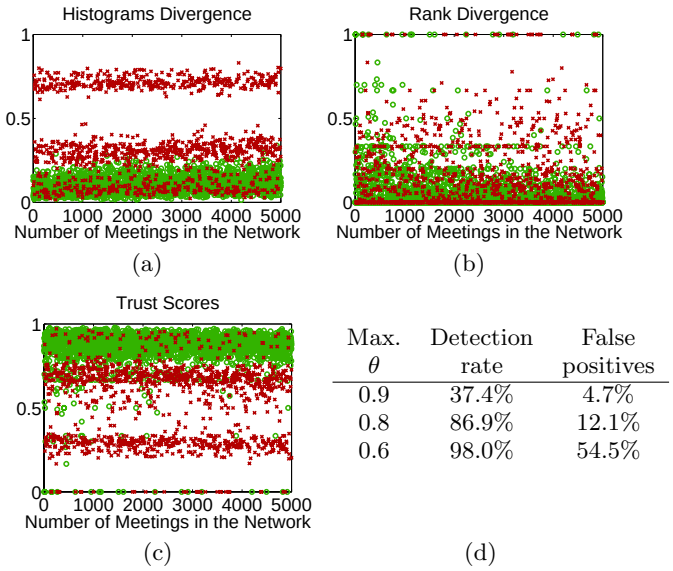


Figure 5: Increased-scores attack: (a) histogram divergence (b) rank divergence. Permuted-scores attack: (c) histogram divergence (d) rank divergence. A circle (○) represents a meeting between two honest peers, and a cross (×) a meeting between an honest and a dishonest peers. Meetings between two dishonest peers are not shown for clarity.



Max. θ	Detection rate	False positives
0.9	37.4%	4.7%
0.8	86.9%	12.1%
0.6	98.0%	54.5%

(d)

Figure 6: Random forms of attack: (a) histograms divergence (b) rank divergence (c) trust scores (d) effect of the hypothetical usage of thresholds in the trust score.

the final scores, but it cannot counter the effect of malicious peers and it quickly degrades.

In Figure 6 (d), we show the fractions of good and malicious peers that receive a trust score θ above or equal to a given threshold t. These values can be used to measure the performance of a (hypothetical) system in which the trust

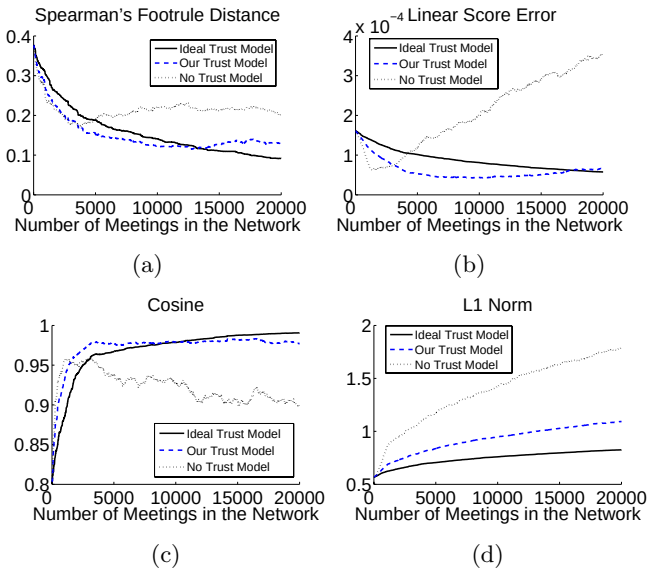


Figure 7: Impact of 50 malicious peers in a system of 150 peers.

score is measured and a meeting is rejected whenever the other peer's trust value is below the threshold. We can see that a threshold equals to 0.8 allows us to recognize and discard 81.93% of malicious peers.

Even though we have shown that TrustJXP is a good contribution for the problem of detecting malicious behavior, it is by no means sufficient in this task. Figure 8 shows the algorithm performance for different number of bad peers in the network. The number of good peers is fixed and equals to 100. We can see that, as the number of bad peers increases, TrustJXP becomes less effective in detecting all malicious behaviors. However, even with a high number of malicious peers, the algorithm is able to slow down the effects of the attacks.

6. CONCLUSIONS

We presented a method for identifying and reducing the impact of malicious peers on the computation of authority scores in a P2P network. We developed the TrustJXP algorithm, an extension of the JXP algorithm that supports the trust scores provided by our trust model. Experiments have demonstrated the viability and robustness of our method, ensuring the correctness of global authority scores.

The model combines an analysis or the authority score distribution and a comparison of score rankings from a small set of pages. It relies on the assumptions that score distributions at all honest peers should look alike, given that the Web graph is self-similar, and that there is sufficient overlap among peers' local graphs. In cases where these assumptions do not hold, honest peers might be unfairly punished, slowing down the scores convergence.

We showed that the normal JXP system can withstand a population of 10% of malicious peers using the attack models described on this paper, but not a population of 33%. We have seen that TrustJXP can work with such a high number of malicious peers. For bigger populations, we showed that

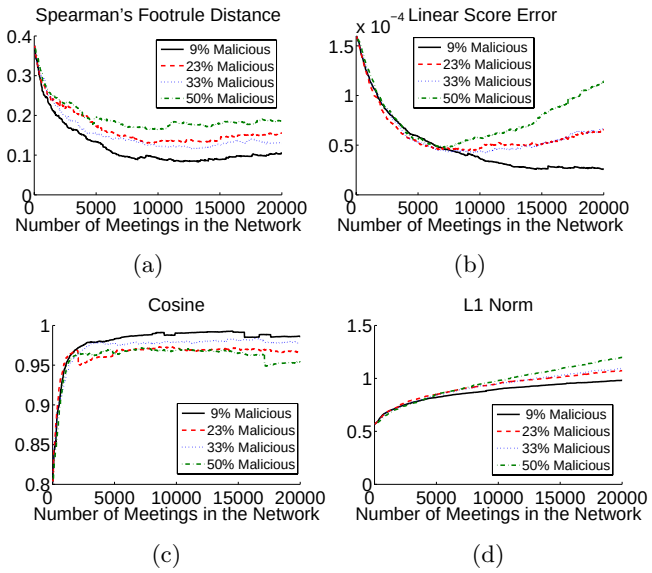


Figure 8: TrustJXP performance with varied number of malicious peers.

the algorithm becomes less effective, but it is still able to slow down malicious effects.

In P2P systems, dynamics is very important. It would be very important to verify in future work what happens if the population of pages changes too rapidly over time.

Our future work will mostly focus on further extension of the trust model to detect other types of malicious behaviors, for instance, where attacks are coordinated among malicious peers.

7. REFERENCES

[1] Zoë Abrams, Robert Mcgrew, and Serge Plotkin. A non-manipulable trust system based on eigentrust. *SIGecom Exch.*, 5(4):21–30, 2005.

[2] Luca Becchetti, Carlos Castillo, Debora Donato, and Adriano Fazzone. A comparison of sampling techniques for web characterization. In *LinkKDD*, 2006.

[3] András A. Benczúr, Károly Csalogány, Tamás Sarlós, and Máté Uher. Spamrank: fully automatic link spam detection. In *AIRWeb*, 2005.

[4] Matthias Bender, Sebastian Michel, Peter Triantafillou, Gerhard Weikum, and Christian Zimmer. Minerva: Collaborative p2p search. In *VLDB*, 2005.

[5] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engines. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[6] Andrei Z. Broder, Ronny Lempel, Farzin Maghoul, and Jan Pedersen. Efficient pagerank approximation via graph aggregation. In *WWW Conf. on Alternate track papers & posters*, 2004. ACM Press.

[7] Steve Chien, Cynthia Dwork, Ravi Kumar, Daniel R. Simon, and D. Sivakumar. Link evolution: Analysis and algorithm. *Internet Mathematics*, 1(3):277–304, 2004.

[8] Jason V. Davis and Inderjit S. Dhillon. Estimating the global pagerank of web communities. In *KDD*, pages 116–125, 2006. ACM Press.

[9] Stephen Dill, Ravi Kumar, Kevin S. Mccurley, Sridhar Rajagopalan, D. Sivakumar, and Andrew Tomkins. Self-similarity in the web. *ACM Trans. Inter. Tech.*, 2(3):205–223, 2002.

[10] AnHai Doan, Raghu Ramakrishnan, Fei Chen, Pedro DeRose, Yoonkyong Lee, Robert McCann, Mayssam Sayyadian, and Warren Shen. Community information management. *IEEE Data Eng. Bull.*, 29(1):64–72, 2006.

[11] Micah Dubinko, Ravi Kumar, Joseph Magnani, Jasmine Novak, Prabhakar Raghavan, and Andrew Tomkins. Visualizing tags over time. In *WWW*, 2006. ACM Press.

[12] R. Fagin, R. Kumar, and D. Sivakumar. Comparing top k lists. In *SIAM Discrete Algorithms*, 2003.

[13] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *WWW*, 2004. ACM Press.

[14] Zoltán Gyöngyi and Hector Garcia-Molina. Web spam taxonomy. In *AIRWeb*, 2005.

[15] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub. Exploiting the block structure of the web for computing pagerank. Technical report, Stanford University, 2003.

[16] Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *WWW*, 2003. ACM Press.

[17] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. In *SIAM Discrete Algorithms*, pages 668–677, 1998.

[18] Ravi Kumar, Jasmine Novak, and Andrew Tomkins. Structure and evolution of online social networks. In *KDD*, 2006. ACM Press.

[19] A. Langville and C. Meyer. Updating the stationary vector of an irreducible markov chain with an eye on google's pagerank. In *SIMAX*, 2005.

[20] Amy N. Langville and Carl D. Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–380, 2003.

[21] Amy Nicole Langville and Carl Dean Meyer. Deeper inside pagerank. *Internet Mathematics*, 1(3):335–400, 2004.

[22] L. Le Cam. *Asymptotic Methods in Statistical Theory*. Springer-Verlag, New York, 1986.

[23] Sergio Marti and Hector Garcia-Molina. Taxonomy of trust: Categorizing p2p reputation systems. *Computer Networks*, 50(4):472–484, March 2006.

[24] C. D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, 2000.

[25] Nikos Ntarmos and Peter Triantafillou. Seal: Managing accesses and data in peer-to-peer sharing networks. *Peer-to-Peer Computing*, 00:116–123, 2004.

[26] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: bringing order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.

[27] Josiane Xavier Parreira, Debora Donato, and Gerhard Weikum. Efficient and decentralized pagerank approximation in a peer-to-peer web search network. In *VLDB*,2006.

[28] Josiane Xavier Parreira and Gerhard Weikum. Jxp: Global authority scores in a p2p network. In *WebDB*, 2005.

[29] Sergej Sizov, Martin Theobald, Stefan Siersdorfer, Gerhard Weikum, Jens Graupmann, Michael Biwer, and Patrick Zimmer. The bingo! system for information portal generation and expert web search. In *CIDR*, 2003.

[30] Natalia Stakhanova, Samik Basu, Johnny Wong, and Oleg Stakhanov. Trust framework for p2p networks using peer-profile based anomaly technique. In *ICDCSW*, 2005. IEEE Computer Society.

[31] Ralf Steinmetz and Klaus Wehrle, editors. *Peer-to-Peer Systems and Applications*, volume 3485 of *Lecture Notes in Computer Science*. Springer, 2005.

[32] Torsten Suel, Chandan Mathur, Jo wen Wu, Jiangong Zhang, Alex Delis, Mehdi Kharrazi, Xiaohui Long, and Kulesh Shanmugasundaram. Odissea: A peer-to-peer architecture for scalable web search and information retrieval. In *WebDB*, 2003.

[33] Yuan Wang and David J. DeWitt. Computing pagerank in a distributed internet search system. In *VLDB*, 2004.

[34] Li Xiong and Ling Liu. Peertrust: supporting reputation-based trust for peer-to-peer electronic communities. *Knowledge and Data Engineering, IEEE Transactions on*, 16(7):843–857, 2004.