Making Eigenvector-Based Reputation Systems Robust to Collusion

Hui Zhang *  Ashish Goel †  Ramesh Govindan ‡  Kahn Mason §
Benjamin Van Roy ¶

May 20, 2004

* Department of Computer Science, University of Southern California. Email: huiz@usc.edu.
† Departments of Management Science and Engineering and (by courtesy) Computer Science, Stanford University. Email: ashishg@stanford.edu.
‡ Department of Computer Science, University of Southern California. Email: ramesh@usc.edu.
¶ Department of Management Science and Engineering, Stanford University. Email: kmason@stanford.edu.
§ Departments of Management Science and Engineering, Electrical Engineering, and (by courtesy) Computer Science, Stanford University. Email: bvr@stanford.edu.
Abstract

Eigenvector based methods in general, and Google’s PageRank algorithm for rating web pages in particular, have become an important component of information retrieval on the Web. In this paper, we study the efficacy of, and countermeasures for, collusions designed to improve user rating in such systems.

We define a metric, called the amplification factor, which captures the amount of PageRank-inflation obtained by a group due to collusions. We prove that the amplification factor can be at most $1/\epsilon$, where $\epsilon$ is the reset probability of the PageRank random walk. We show that colluding nodes (web-pages, blogs etc.) can achieve this amplification and increase their rank significantly in realistic settings; further, several natural schemes to address this problem are demonstrably inadequate.

We propose a relatively simple modification to PageRank which renders the algorithm insensitive to such collusion attempts. Our scheme is based on the observation that nodes which cheat do so by “stalling” the random walk in a small portion of the web graph and, hence, their PageRank must be especially sensitive to the reset probability $\epsilon$. We perform exhaustive simulations on the Web and Weblog graphs to demonstrate that our scheme successfully prevents colluding nodes from improving their rank, yielding an algorithm that is robust to gaming.

1 Introduction

Reputation systems are becoming an increasingly important component of information retrieval on the Web. Such systems are now ubiquitous in electronic commerce, and enable users to judge the reputation and trustworthiness of online merchants or auctioneers. In the near future, they may help counteract the free-rider phenomenon in peer-to-peer networks by rating users of these networks and thereby inducing social pressure to offer their resources for file-sharing [4, 8, 10]. Also, they may soon provide context for political opinion in the Web logging (blogging) world, enabling readers to calibrate the reliability of news and opinion sources.

It seems reasonable that reputation systems in these cases will be centralized, even if (as with peer-to-peer networks or the blogosphere) the underlying systems are distributed. Reputation is built or tarnished on a longer timescale than individual transactions, and the existence of centralized Web search engines clearly points to the feasibility of centralized reputation systems. What algorithms might such systems use to rate users? A simple, and commonly used, way to measure a user’s reputation is using a referential link structure, a graph where nodes represent entities (users, merchants, authors of blogs) and links represent endorsements of one user by another. A starting point for an algorithm to compute user reputations might then be the class of eigenvector or stationary distribution based reputation schemes exemplified by the PageRank algorithm.

Algorithms based on link structure are susceptible to collusions; we make the notion of collusion more precise later, but for now we loosely define it as a manipulation of the link structure by a group of users with the intent of improving the rating of one or more users in the group. The PageRank algorithm published in the literature has a simple “resetting” mechanism to alleviate the impact of collusions. The PageRank value assigned to a page can be modeled as the fraction of time spent at that page by a random walk over the link structure; to reduce the impact of collusions (in particular, rank “sinks”), the algorithm resets the random walk at each step with probability $\epsilon$.

In this paper, we define a quantity called the amplification factor that characterizes the amount of PageRank-inflation obtained by a group of colluding users. We show that nodes may increase their PageRank values by at most an amplification factor $\frac{1}{\epsilon}$; intuitively, a colluding group can “stall” the random walk for that duration before it resets. While this may not seem like much (a typical value for $\epsilon$ is 0.15), it turns out that the distribution of PageRank values is such that even this amplification is sufficient to significantly boost the rank of a node based on its PageRank value. What’s worse is that all users in a colluding group could and usually do benefit from the collusion, so there is significant incentive for users to collude. For example, we found that it was easy to modify the link structure of the Web by having a low (say 10,000-th) ranked user collude with a user of even lower rank to catapult themselves into the top-400. Similar results exist for links in the blogosphere.

There are two natural candidate solutions to this problem – identifying groups of colluding nodes, and identifying individual colluders by using detailed return time statistics from the PageRank random walk. The former is computationally intractable since the underlying optimization problems are NP-Hard. The latter does not solve the problem since we can identify scenarios where the return time statistics for the colluding nodes are nearly indistinguishable from those for an “honest” node.

How then, can PageRank based reputation systems

\footnote{Although not viewed as such, PageRank may be thought of as a way of rating the “reputation” of web sites.}

\footnote{Collusion implies intent, and our schemes are not able to determine intent, of course. Some of the collusion structures are simple enough that they can occur quite by accident.}
protect themselves from such collusions? Observe that the ratings of colluding nodes are far more sensitive to $\epsilon$ than those of non-colluding nodes. This is because the PageRank values of colluding nodes are amplified by “stalling” the random walk as explained before, the amount of time a group can stall the random walk is roughly $1/\epsilon$. This suggests a simple modification to the PageRank algorithm (called the adaptive-epsilon scheme) that allows different nodes to have different values of the reset probability. We have not been able to formally prove the correctness of our scheme (and that’s not surprising given the hardness result), but we show, using extensive simulations on real-world link structures, that our scheme significantly reduces the benefit that users obtain from collusion in both the Web and Blog graphs. Furthermore, while there is substantial intuition behind our detection scheme, we do not have as good an understanding of the optimum policy for modifying the individual reset probabilities. We defer an exploration of this to future work.

While we focus on PageRank in our exposition, we believe that our scheme is also applicable to other eigenvector-based reputation systems (e.g. [8, 10]).

We should point out that the actual page ranking algorithms used by modern search engines (e.g., Google) have evolved significantly and incorporates other domain specific techniques to detect collisions that are not (and will not be, for some time to come) in the public domain. Our work is not addressed at algorithms that run on those systems—rather, our focus is on emerging public infrastructures (peer-to-peer systems and the blogosphere) whose reputation systems design are likely to be based on work in the public domain. Our work increases the awareness of the subtle implications of such algorithms.

The remainder of this paper is organized as follows. We discuss related work in Section 2. In Section 3 we study the impact of collisions on the PageRank algorithm, in the context of Web and Blogs. Section 4 shows the hardness of making PageRank robust to collisions. In Section 5 we describe the adaptive-epsilon scheme, and demonstrate its efficiency through exhaustive simulations on the Web and Blog graphs. Section 6 presents our conclusions.

2 Related Work

Reputation systems have been studied very heavily recently in non-collusive settings such as eBay [3, 6, 7, 11] — such systems are not the subject of study in this paper.

In the literature, there are at least two well-known eigenvector-based link analysis algorithms: HITS [9] and PageRank [13]. HITS was originally proposed to refine search outputs from Web search engines and discover the most influential web pages defined by the principal eigenvector of its link matrix. As discovering the principal eigenvector is the goal, original HITS doesn’t assign a total ordering on the input pages, and collusion is not a problem. On the contrary, PageRank was proposed to rank order input pages and handling clique-like subgraphs is a fundamental design issue.

Despite their difference, both algorithms have been applied into the design of reputation systems for distributed systems [8, 10]. These designs have mainly focused on the decentralization part, while their collusion-proofness still relies on the algorithm itself.

Ng et al. [12] studied the stability of HITS and PageRank algorithm with the following question in mind: when a small portion of the given graph is removed (e.g., due to incomplete crawling), how severely do the ranks of the remaining pages change, especially for those top ranked nodes? They show that HITS is sensitive to small perturbations, while PageRank is much more stable. They proposed to incorporate the PageRank’s “reset-to-uniform-distribution” into HITS to enhance its stability.

Finally, for context, we briefly describe the original PageRank algorithm with its random walk model. Given a directed graph, a random walk $W$ starts its journey on any node with the same probability. At the current node $x$, with probability $(1-\epsilon)$ $W$ jumps to one of the nodes that have links from $x$ (the choice of neighbor is uniform), and with probability $\epsilon$, $W$ decides to restart (reset) its journey and again choose any node in the graph with the same probability. Asymptotically, the stationary probability that $W$ is on node $x$ is called the PageRank value of $x$, and all nodes are ordered based on the PageRank values.

In the rest of the paper, we use the term weight to denote the PageRank (PR) value, and rank to denote the ordering. We use the convention that the node with the largest PR weight is ranked first.

3 Impact of Collusions on PageRank

In this section, we first show how a group of nodes could modify the referential link structure used by the PageRank algorithm in order to boost their PageRank weights by up to $1/\epsilon$. We then demonstrate that it is possible to induce simple collusions in real link structures (such as those in the Web and Blog graphs) in a manner that raises the ranking of colluding nodes significantly.

---

3We use “pages”, “nodes” and “users” interchangeably when referring to entities that collude.
3.1 Amplifying PageRank Weights

In what follows, we will consider the PageRank algorithm as applied to a directed graph \( G = (V,E) \). \( N = |V| \) is the number of the nodes in \( G \). A node in \( G \) corresponds, for example, to a Web page in the Web graph, or a blog in the blog graph; an edge in \( G \) corresponds to a reference from one web page to another, or from one blog to another. Let \( d(i) \) be the out-degree of node \( i \), and \( W_e(i) \) be the weight that the PageRank algorithm computes for node \( i \). We define on each edge \( e_{ij} \in E \) the weight \( W_e(e_{ij}) = \frac{W_e(i \times (1-\epsilon))}{d(i)} \).

Let \( V' \subseteq V \) be a set of nodes in the graph, and let \( G' \) be the subgraph induced by \( V' \). \( E' \) is defined to be the set of all edges \( e_{ij} \) such that at least one out of \( i \) and \( j \) is in \( V' \). We classify the edges in \( E' \) into three groups:

**In links:** An edge \( e_{ij} \) is an in link for \( G' \) if \( i \notin V' \) and \( j \in V' \). \( E_{in}' \) denotes the set of in links of \( G' \).

**Internal links:** An edge \( e_{ij} \) is an internal link for \( G' \) if \( i \in V' \) and \( j \in V' \). \( E_{internal}' \) denotes the set of internal links of \( G' \).

**Out links** An edge \( e_{ij} \) is an out link for \( G' \) if \( i \in V' \) and \( j \notin V' \). \( E_{out}' \) denotes the set of out links of \( G' \).

One can then define two types of weights on \( G' \):

- \( W_{in}(G') = \sum_{e \in E_{in}} W_e(e) + N' \), \( N = |V|, N' = |V'| \).
- \( W_{out}(G') = \sum_{e \in E_{out}} W_e(e) \).

Intuitively, \( W_{in}(G') \) is, in some sense, the “actual” weight that should be assigned to \( G' \), when \( G' \) is regarded in its entirety (i.e., as one unit). The first term in \( W_{in}(G') \) represents the amount of reputation flowing into the group of nodes, while the second term makes an allowance for the size of the group. On the other hand, \( W_{out}(G') \) is the total “reputation” of the group that would be assigned by PageRank. Note that nodes within \( G' \) can boost this reputation by manipulating the link structure of the internal links or the out links.

Then, we can define a metric we call the amplification factor of a graph \( G \) as \( \text{Amp}(G) = \frac{W_{out}(G)}{W_{in}(G)} \).

4Given this definition, we prove (see Appendix A for the proof) the following theorem:

**Theorem 1** In the original PageRank system, \( \forall G', G' \subseteq G, \text{Amp}(G') < \frac{1}{\epsilon} \).

4While the term “amplification” might imply otherwise, it is possible for \( \text{Amp}(G') \leq 1 \). For example, a subgroup \( G' \) of random nodes having no internal links and few in links might have \( \text{Amp}(G') \approx \epsilon \).

3.2 PageRank Experiments on Real-World Graphs

This theorem says that given the value of \( \epsilon \), a group of nodes can amplify their total input weight \( W_{in} \) by at most \( \frac{1}{\epsilon} \). The literature [13] has assumed a default value for \( \epsilon \) of 0.15 , which translates into an amplification factor of about 7. We also expect practical \( \epsilon \) values to be about this large. Larger values of \( \epsilon \) will result in less differentiation among nodes since the random walk will reset more frequently. Smaller values will result in much longer convergence times for the PageRank algorithm.

Given this finding, we seek to understand what is the practical import of amplifying PageRank weights. Specifically, is it easy for a group of colluding nodes to achieve the upper bound of the amplification factor, \( \frac{1}{\epsilon} \)? Can nodes improve their ranking significantly?

To answer these questions, we obtained a large Web subgraph from the Stanford WebBase [15]. This graph contains upwards of 80 million URLs, and we call it \( W \) in the rest of the paper. For Weblogs, we extracted the “blogroll” structure for 72,428 blogs from Blogstreet [16]. A blogroll is a collection of links to other weblogs that are found on most weblogs [14], and the blogrolling relationships indicate the important references among blogs. We call the blogrolling structure \( B \) in the rest of the paper. We then modified one or more subgraphs in each of these graphs to simulate collaborations, and measured the resulting PageRank weights for each node. We tried a few different modifications, and report the results for one such experiment.

Our first experiment on \( W \) is called **Collusion200**. This models a small number of web pages simultaneously colluding. Each collusion consists of a pair of nodes with adjacent ranks. Such a choice is more meaningful than one between a low ranked node and a high ranked node, since the latter could have little incentive to collude. Each pair of nodes removes their original out links and adds one new out link to each other. In the experiment we report in this paper, we induce 100 such collusions at nodes originally ranked around 1000th, 2000th, . . . , 100000th.

There is a subtlety in picking these nodes. We are given a real-world graph in which there might already be colluding groups (intentional or otherwise). For this reason, we carefully choose our nodes such that they are unlikely to be already colluding (the precise methodology for doing this will become clear in Section 5.2.1 when we describe how we can detect colluding groups in graphs).

We calculate the PageRank weights and ranks for all nodes before (called old rank and weight) and after (called new rank and weight) **Collusion200** on \( W \) with \( \epsilon = 0.15 \). Figures 1 & 2 show the rank and weight
change for those colluding nodes. In addition, we also plot in Figure 1 the rank that each colluding node could have achieved if its weight were amplified by $\frac{1}{r}$ while all other nodes remained unchanged in weight, which we call pseudo collusion.

As we can see, all colluding nodes increased their PR weight by at least 3.5 times, while the majority have a weight amplification over 5.5. More importantly, collusion boosts their ranks to be more than 10 times higher and close to the best achievable. For example, a colluding node originally ranked at 100025th had a new rank at 451th, while the 100005th node boosted its rank to 5033th by colluding with the 100009th node, which also boosted its rank to 5038th.

We repeated Collusion200 on B. In this graph however, we select nodes that are ranked around 100th, 200th, ..., 10000th given the smaller graph size. Figures 3 & 4 show the rank and weight change for those colluding nodes. As in W, colluding nodes boost their ranks significantly; the boosted ranks are close to those in pseudo collusion. For example, the 100th and 110th nodes in the original B successfully boosted their ranks to 7th and 8th positions with the collusion, while the 8100th node had its new rank at the 877th.

Thus, even concurrent, simple (2-node) collusions of nodes with comparable original ranks can result in significant rank inflation for the colluding nodes. For another view of this phenomenon in Figure 5 we plot the amplification factors achieved by the colluding groups in W and B. It clearly shows that almost all colluding groups attain the upper bound. Note that the actually achieved amplification factors for B are lower than what is theoretically possible, especially for these groups in the right side of Figure 5 which consist of low-ranked nodes. This is because $\frac{N_B}{N_W}$, the PageRank weight component that a colluding group $G'$ can't amplify, is a non-trivial component of $W_{in}(G')$ for these low PR weight nodes.
4 On the Hardness of Making
PageRank Robust to Collusions

We will now explore two natural approaches to detecting
colluding nodes, and demonstrate that neither of them

can be effective.

The first approach is to use finer statistics of the
PageRank random walk. Let the random variable $X_v$
denote the number of steps between two successive
visits to node $v$. It is easy to see that the PageRank value
of $v$ is exactly $1/E[X_v]$ where $E[X_v]$ denotes the
expectation of $X_v$. For the simplest collusion, where two
nodes $A$ and $B$ delete all their out-links and start pointing
only to each other, the random walk will consist of a
long alternating sequence of $A$'s and $B$'s, followed by a
long sojourn in the remaining graph, followed again by
a long alternating sequence of $A$'s and $B$'s, and so on $^5$.
Clearly, $X_A$ is going to be 2 most of the time, and very
large (with high probability) occasionally. Thus, the
ratio of the variance and the expectation of $X_A$ will be
disproportionately large. It is now tempting to suggest
using this ratio as an indicator of collusion.

Unfortunately, there exist simple examples (such as
large cycles) where this approach fails to detect collud-
ing nodes. We will present a more involved example
where not just the means and the variances, but the
entire distributions of $X_H$ and $X_C$ are nearly identical;
here $H$ is an "honest" node and $C$ is cheating to im-
prove its PageRank. The initial graph is a simple star
topology. Node 0 points to each of the nodes $1 \ldots N$ and
each of these nodes points back to node 0 in turn. Now,
node $N$ starts to cheat; it starts colluding with a new
node $N+1$ so that $N$ and $N+1$ now only point to each
other. The new distributions $X_0$ and $X_N$ can be explicitly
computed, but the calculation is tedious. Rather
than reproduce the calculation, we provide simulation
results for a specific case, where $N = 7$ and $\epsilon = 0.12$.
Figure 7 shows the revisit distribution for nodes 0 (the
original hub) and 7 (the cheating node). The distribu-

---

$^5$Incidentally, it is easy to show that this collusion mode can achieve the theoretical upper bound of $1/\epsilon$ on the amplification factor.
ions are nearly identical. Hence, any approach that relies solely on the detailed statistics of \( X_e \) is unlikely to succeed.

![Figure 7: Frequency of revisit intervals for the cheating node (node 7) and the honest node (node 0) for the star-topology. The simulation was done over 1,000,000 steps.](image)

Thus, a more complete look at the graph structure is needed, one that factors in the various paths the random walk can take. One natural approach to identifying colluders would be to directly find the subgraph with the maximum amplification (since colluders are those with high amplification). However, it is very unlikely that this problem is tractable. Consider the intimately related problem of finding a group \( S \) of size \( k \) which maximizes the difference of the weights, \( W_G(S) - W_m(S) \), rather than the ratio. This problem is NP-hard via reduction to the densest \( k \)-subgraph problem [5]. Details of the reduction are in appendix C. There are no good approximation algorithms known for the densest \( k \)-subgraph problem (the best known is \( O(N^{1/2}) \)). The reduction is approximation preserving. Hence, identifying colluding groups is unlikely to be computationally tractable even in approximate settings.

This suggests that our goals should be more modest — rather than identifying the entire colluding group, we should just focus on finding individual nodes that are cheating. This is the approach we take in the next section.

5 Heuristics for Making PageRank Robust to Collusions

Given our discussion of the hardness of making PageRank robust to collusions, we now turn our attention to heuristics for achieving this. Our heuristic is based on an observation that can be explained using the following example. Consider a small (compared to the size of the original graph) group \( S \) of colluding nodes. These nodes can not influence links from \( V - S \) into \( S \). Hence, the only way these nodes can increase their stationary weight in the PageRank random walk is by stalling the random walk i.e. by not letting the random walk escape the group. But in the PageRank algorithm, the random walk resets at each node with probability \( \epsilon \). Hence, colluding nodes must suffer a significant drop in PageRank as \( \epsilon \) increases.

This forms the basis for our heuristic for detecting colluding nodes. We expect the stationary weight of colluding nodes to be highly correlated \(^6\) with \( 1/\epsilon \) and that of non-colluding nodes to be relatively insensitive to changes in \( \epsilon \).

To gain additional intuition, consider the very simple setting where each out of \( N \) nodes initially links to every other node. Suppose that a set \( S \) of \( K \) of these nodes starts colluding; each node in \( S \) removes all out-links to nodes outside \( S \). Let \( x(\epsilon) \) denote the stationary weight on one of the cheating nodes, and \( y(\epsilon) \) on one of the remaining \( N - K \) nodes. It is easy to compute \( x \) and \( y \); \( x(\epsilon) = 1/(K + (N - K)\epsilon) \) and \( y(\epsilon) = \epsilon/(K + (N - K)\epsilon) \). In the domain of interest, we get \( x(\epsilon) \approx 1/\epsilon N \) and \( y(\epsilon) \approx 1/N \). This gives co-co(\( x, 1/\epsilon \)) \( \approx 1 \) and co-co(\( y, 1/\epsilon \)) \( \approx 0 \), which strongly differentiates between the cheating and honest nodes.

In fact, a similar dichotomy is experimentally observed in the graphs \( W \) and \( B \). Figure 8 shows that most of the nodes have a correlation value close to or less than 0, whereas a small fraction have correlation value close to 1. This is a very promising indication, and we now present our adaptive-\( \epsilon \)-heuristic that tries to leverage this dichotomy. Our heuristic identifies nodes which have a high correlation with \( 1/\epsilon \) and increases the reset probability for those nodes — this diminishes the ability of colluding nodes to stall the random walk. Of course, the larger the correlation of a node with \( 1/\epsilon \), the larger the increase in its reset probability. Our heuristic requires only a relatively minor modification to the PageRank algorithm.

5.1 The adaptive-\( \epsilon \) heuristic

The central idea behind our heuristic for a collusion-proof PageRank algorithm is that the value of the reset probability is adapted, for each node, to the degree of

\(^6\)The correlation coefficient of a set of observations \( (x_i, y_i) : i = 1, \ldots, n \) is given by the formula:

\[
\text{co-co}(x, y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2 \sum_{i=1}^{n} (y_i - \bar{y})^2}}
\]
collusion that the node is perceived to be engaged in. This adaptive-\(\epsilon\) scheme consists of two phases:

1. Collusion detection
   
   \(\text{(a)}\) Given the topology, calculate the PR weight vector under different \(\epsilon\) values.
   
   \(\text{(b)}\) Calculate the correlation coefficient between the curve of each node \(x\)'s PR weight and the curve of \(\frac{1}{2}\). Label it as \(\text{co-co}(x)\), which represents the potential of \(x\) in collusion. \(\text{co-co}(x) = \text{co-co}(x) < 0 \Rightarrow \text{co-co}(x)\).

2. \(\epsilon\) Personalization
   
   \(\text{(a)}\) Now the node \(x\)'s out-link personalized-\(\epsilon\) = \(F(\epsilon_{defaul}, \text{co-co}(x))\).
   
   \(\text{(b)}\) The PageRank algorithm is repeated with these personalized-\(\epsilon\) values.

The function \(F(\epsilon_{defaul}, \text{co-co}(x))\) provides a knob for a system designer to appropriately punish colluding nodes. In our experiments we tested two functions:

- **Exp. function** \(F_{\text{Exp}} = \epsilon_{\text{defaul}}(1.0 - \text{co-co}(x))\).

- **Linear function** \(F_{\text{Linear}} = \epsilon_{\text{defaul}} + (0.5 - \epsilon_{\text{defaul}}) \times \text{co-co}(x)\).

The function \(F_{\text{Exp}}\) punishes the colluding nodes severely enough that they have little ability to propagate their PR weight through the rest of the network. The function \(F_{\text{Linear}}\) is less severe and colluding nodes can still affect the weights of other nodes.

We have found that both algorithms achieve robustness to collusion and differ only to the extent that they more accurately represent the weights of non-colluding nodes. The choice of function is subjective and application-dependent, and given space limitations, we mostly present results based on \(F_{\text{Exp}}\).

### 5.2 Experiments

As in Section 3, we conducted experiments on the \(W\) and \(B\) graphs. In all experiments with our adaptive-\(\epsilon\) scheme, we chose seven \(\epsilon\) values in the collusion detection phase \(-0.6, 0.45, 0.3, 0.15, 0.075, 0.05,\) and \(0.0375\) – and used 0.15 as \(\epsilon_{\text{defaul}}\). While there are eight Pagerank calculations, the actual computational time for the adaptive-\(\epsilon\) scheme was only 2-3 times that of the original PageRank algorithm. This is because the computed PR weight vector for one \(\epsilon\) value is a good initial state for the next \(\epsilon\) value.

#### 5.2.1 Basic Experiment

We first repeated the experiment *Collusion200* from Section 3 for adaptive-\(\epsilon\) scheme. As mentioned in Section 3.2, all the colluding nodes are chosen from the nodes unlikely to be already colluding, and this is judged by their \(\text{co-co}\) values in the original topology. More precisely, we select nodes with \(\text{co-co}(x) < 0.1\). Choosing nodes with arbitrary \(\text{co-co}\) values doesn’t invalidate any of our conclusions in this paper (as discusses in Appendix B), but our selection methodology simplifies the exposition of our scheme.

We compared the original PageRank algorithm, the adaptive-\(\epsilon\) scheme using \(F_{\text{Exp}}\), and the adaptive-\(\epsilon\) scheme using \(F_{\text{Linear}}\). As shown in Figures 9 and 10 for \(W\) and \(B\) respectively, the adaptive-\(\epsilon\) scheme \(F_{\text{Exp}}\) restricted the amplification factors of the colluding groups to be very close to one, and \(F_{\text{Linear}}\) also did quite well compared to the original PageRank.

![Figure 9: \(W\): amplification factors of the 100 colluding groups in Collusion200](image)
Figure 10: $B$: amplification factors of the 100 colluding groups in Collusion200

In Figures 11 and 12, we compare the original PageRank and the adaptive-ε scheme using $F_{Exp}$ based on the old and new rank (resp. weight) before and after Collusion200 in $W$. For the original PageRank algorithm the rank and weight distributions clearly indicate how nodes benefit significantly from collusion. The curves for the adaptive-ε scheme nearly overlap, illustrating the robustness of our heuristic. Furthermore, note that the curves of the PageRank algorithm before collisions and the adaptive-ε before collisions are close to each other, which means the weight of non-colluding nodes is not affected noticeably when applying the adaptive-ε scheme instead of the original PageRank scheme.

We repeated the comparison for $B$, (Figures 13 and 14). Our observations above hold here as well.

5.2.2 Other collusion topologies

An experiment with miscellaneous collusion topologies: We tested adaptive-ε scheme under other collusion topologies in an experiment called Collusion22. In Collusion22 22 low co-co (≤ 0.1) nodes are selected for 3 colluding groups:

G1 G1 has 10 nodes, which remove their old out links and organize into a a single-link ring. All nodes have their original ranks at around 1000th.

G2 G2 has 10 nodes, which remove their old out links and organize into a star topology by one hub pointing to the other 9 nodes and vice versa. The hub node has its original rank at around 5000th, while the other nodes are ranked at around 10000th originally.

G3 G3 has 2 nodes, which remove the old out links and organize into a two-node circle. One node is originally ranked at around 50th, and the other at around 9000th.

We ran experiment Collusion22 on $W$ and $B$ using both original PageRank and adaptive-ε scheme. We first observed that the adaptive-ε scheme successfully detected all 22 colluding nodes by reporting high co-co values (> 0.96) for both $W$ and $B$.

In Figure 15, we compare the original PageRank algorithm, the adaptive-ε scheme with function $F_{Exp}$, and the adaptive-ε scheme with function $F_{Linear}$ based on the metric amplification factor under Collusion22 for $W$. As in Figures 11 and 12, the two adaptive-ε schemes successfully restricted the weight amplification for the colluding nodes.

In Figures 16 and 17, we compare original PageRank and adaptive-ε scheme with function $F_{Exp}$ based on the old and new rank (weight) before and after Collusion22 in $W$. The results for the graph $B$ were sim-
that might use the “waiting time” paradox to detect collusions. It is instructive to consider if the adaptive-$\epsilon$ scheme detects the collusion in this topology.

We ran the first phase of adaptive-$\epsilon$ scheme - collusion detection with varying $\epsilon$ - on one such topology: a 1000-node graph in which node 0 has out-links to nodes 1-998, node 1-997 have out links to node 0, and finally node 998 and 999 have out links to each other. (Note that, unlike previous experiments, this topology is not embedded into any of our real-world graphs.)

Figure 18 shows the PR weight variation of node 0 (the hub node), node 1 (a normal leaf node), nodes 998 and 999 (the two nodes in the dangling circle). The y-axis shows the PR weight normalized by each node’s PR weight at $\epsilon = 0.15$. It also includes, for calibration, the curve $\frac{0.15}{\epsilon}$. Clearly, the curves of node 998 and 999 have high correlation-coefficient with the curve $\frac{0.15}{\epsilon}$, while it is not so for the rest of the nodes. Therefore, adaptive-$\epsilon$ scheme can detect the dangling circle in this topology, another indication of the general applicability of our heuristic.

**Topology analysis – use as a detection scheme:**

In Figures 19 we plot the spanning trees (within 3 hops) rooted at node 30373768 which has a high rank and also a high $\alpha$-$\alpha$ value in $\mathcal{W}$. It turns out node 30373768 corresponds to the URL http://www.yahoo.com/, while node 3036967 corresponds to the URL http://messeger.yahoo.com/, and they organize into a circle 7. In Figure 20 we plot the spanning trees (within 3 hops) rooted at node 71311

---

7At the time when this link structure was obtained, all references from Yahoo, except for its link to http://messeger.yahoo.com/, contain URLs embedded as parameters to a CGI script, and these are not counted by the PageRank algorithm [13].
Figure 16: \( W \): PR rank comparison between original PageRank and Adaptive-\( \epsilon \) scheme in \textit{Collusion22}.

Figure 17: \( W \): PR weight comparison between original PageRank and Adaptive-\( \epsilon \) scheme in \textit{Collusion22}.

which has a high rank and also a high \textit{ca-co} value in \( B \). It turns out that this node is in a 7-node star-like topology with the hub node at node 65519.

\textbf{Validation}: Finally, we show in Table 1 the top-25 URL lists in \( W \) ranked with PageRank (called the old list) and with \( F_{Exp} \) based adaptive-\( \epsilon \) scheme (called the new list). Overall, we see in the new list many “intuitively” small pages drop out of the top 25 list while some well known URLs show up. Notice in particular that in the new list, \url{http://www.yahoo.com/} drops from the first to the second position, while \url{http://messenger.yahoo.com/} drops out of the top-25 list. This is fair since with original PageRank, the amplification factor for the Yahoo group is 4.8. The results for the top-50 list are qualitatively similar.

This serves as a “human-verifiable” sanity check that our scheme does not change the ranks of web pages in unexpected ways.

Figure 18: The change of PR weight with varying \( \epsilon \): a star+dangling circle topology.

6 Conclusion & Future work

In this paper we studied the robustness of one eigenvector-based rating algorithm: PageRank. We point out the importance of collusion detection in PageRank based reputation systems for real-world graphs, its hardness, and then a heuristic solution. Our solution involves detecting colluding nodes based on the sensitivity of their PageRank value to the reset probability \( \epsilon \) and then penalizing them by assigning them a higher reset probability. We have done extensive simulations on both the Web and Weblog graphs to demonstrate the efficacy of our heuristic.

While there is substantial intuition behind our detection scheme, we do not have as good an understanding of the optimum policy for modifying the individual reset probabilities. We believe this is an interesting topic which deserves further exploration. Similarly, studying the evolution of Web link structure under PageRank within the framework of game theory is an interesting research direction orthogonal to the work presented in this paper. This is motivated by the observation that PageRank, or more precisely, Google, has impacted the way web sites organize their links and “Google-bombing” is now a popular sport.

Acknowledgement

We would like to thank the Stanford Webbase group for making a pre-processed copy of the Web link structure available to us.
<table>
<thead>
<tr>
<th>Rank</th>
<th>Old list</th>
<th>New list</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td><a href="http://upload.tucows.com/contactus.html">http://upload.tucows.com/contactus.html</a></td>
<td><a href="http://www.w3.org/">http://www.w3.org/</a></td>
</tr>
<tr>
<td>18</td>
<td><a href="http://www.w3.org/">http://www.w3.org/</a></td>
<td><a href="http://www.worldwidemart.com/scripts/faq/wwwboard">http://www.worldwidemart.com/scripts/faq/wwwboard</a>...</td>
</tr>
</tbody>
</table>

Table 1: The old and new top-25 list of $W$

References


Figure 19: a small loop between the top 2 nodes in \( \mathcal{W} \)

Figure 20: a star topology in \( \mathcal{B} \)


A Proof of Theorem 1

Proof: Like calculating the PR weight of each individual node, we can calculate the PR weight $W_G(G')$ for the subgroup $G'$ as:

$$W_G(G') = \sum_{e \in E_{n}'} W_e(e) + \sum_{e \in E_{n}'} \frac{\varepsilon \times N'}{N}$$

$$\leq \sum_{e \in E_{n}'} W_e(e) + \sum_{v \in V'} (1 - \varepsilon) \times W_v(v)$$

(= holds when $E_{n+1} = \phi$)

$$\leq \sum_{e \in E_{n}'} W_e(e) + \frac{\varepsilon \times N'}{N} + (1 - \varepsilon) \times W_G(G')$$

Therefore,

$$W_G(G') \leq \frac{\sum_{e \in E_{n}'} W_e(e) + \frac{\varepsilon \times N'}{N}}{\varepsilon}$$

$$< \frac{W_G(G')}{\varepsilon}$$

B Collusions with Arbitrary Correlation Coefficients

In the paper, we presented experiments in which only low co-co nodes were chosen for collusions. The reason is to decouple the effects of pre-existing collusions (whether intentional or accidental) from new ones. Suppose a node $X$, which is already in collusion with another node $Y$ in the original link topology, is now chosen for a new collusion with node $Z$. In our experiments, we would remove all old outgoing links from $X$ and $Z$ and establish a new link from $X$ to $Z$. In particular, this disrupts the collusion between $X$ and $Y$. Depending on the scenario, $X$’s PageRank value using the adaptive-$\varepsilon$ scheme might actually go up, making it hard to compare the adaptive-$\varepsilon$ scheme to the original PageRank.

For completeness, we now present some experiments with nodes having arbitrary co-co values in the original topology. Due to the similarity in the results, we only show the results for the experiment Collusion200 on $W$.

In Figure. 21 and 22, we compare the original PageRank and the adaptive-$\varepsilon$ scheme using $F_{Exp}$ based on the old and new rank (resp. weight) before and after Collusion200. Figure. 21 and 22 have several similarities with those in Figures 11 and 12: the curves for the adaptive-$\varepsilon$ scheme nearly overlap, while the curves for the original PageRank are quite different from each other. However, for several nodes, the adaptive-$\varepsilon$ scheme differs significantly from the original PageRank in the absence of collusions, making it harder to do an “apples-to-apples” comparison. This is not surprising, since we know that close to 10% of the top 1 million nodes in $W$ have co-co values higher than 0.9 (shown in Figure 8) and will be penalized by adaptive-$\varepsilon$.

Even in these experiments, we can still clearly exhibit the main point of the adaptive-$\varepsilon$ scheme in Figure 23: no colluding group can achieve a large amplification factor and benefit from Collusion200.

C The NP-Hardness of Identifying Colluding Groups

Suppose $R(S)$ is the total stationary weight of a set $S$, $I(S)$ is the total weight of all the incoming transitions
Figure 23: Amplification factors of the 100 colluding groups in Collusion200 - arbitrary coco

(from $V - S$ into $S$), and $L(S)$ is the total weight of all the local transitions (from $S$ to $S$). The optimization problem that we formulated in section 4 was to maximize $W_G(S) - W_{in}(S)$ over sets $S$ of size $k$. This is the same as maximizing $R(S) - I(S)$, which in turn is the same as just maximizing $L(S)$.

Hence, our optimization problem is the following: Given a directed graph $G(V,E)$ with non-negative edge-weights, find the set $S \subset V$ of size $k$ which maximizes the sum of the internal edge-weights of $S$.

The densest $k$-subgraph problem [5] is the following: given an undirected graph $G'(V',E')$ with non-negative edge-weights, find the set $S \subset V'$ of size $k$ which maximizes the sum of the internal edge-weights of $S$.

The two problems are the same except for the difference between directed and undirected graphs. To obtain an approximation-preserving reduction of the densest $k$-subgraph problem to our problem, we just need to bidirect each edge in $E'$.

The densest $k$-subgraph problem is known to be NP-Hard via reduction from CLIQUE, so our optimization problem is also NP-Hard.