Identifying Link Farm Spam Pages

Baoning Wu and Brian D. Davison
Department of Computer Science & Engineering
Lehigh University
Bethlehem, PA 18015 USA
{baw4,davison}@cse.lehigh.edu

ABSTRACT

With the increasing importance of search in guiding today's web traffic, more and more effort has been spent to create search engine spam. Since link analysis is one of the most important factors in current commercial search engines' ranking systems, new kinds of spam aiming at links have appeared. Building link farms is one technique that can deteriorate link-based ranking algorithms. In this paper, we present algorithms for detecting these link farms automatically by first generating a seed set based on the common link set between incoming and outgoing links of Web pages and then expanding it. Links between identified pages are reweighted, providing a modified web graph to use in ranking page importance. Experimental results show that we can identify most link farm spam pages and the final ranking results are improved for almost all tested queries.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Performance

Keywords

Web search engine, link analysis, spam, HITS, PageRank

1. INTRODUCTION

Search is the dominant method of finding information on the Web. For most queries, only the top 10 web page results from a search engine are viewed. Since more traffic turns into more profit for most commercial web sites, content creators want their web pages to be ranked as high as possible in the search engine results.

Instead of making high quality web pages, some authors aim at making their pages rank highly by playing with the Web pages features that search engines' ranking algorithms base on. This behavior is usually called "search engine spam" [22, 15].

Many spamming techniques have been discovered. Originally the traditional textual information retrieval algorithms, such as TFIDF, played an important role in search engine ranking algorithms. So initial search engine spamming techniques focused on the content of the page, such as repeating keywords many times within a page or appending a dictionary at the bottom of a page. With the invention of link-based ranking algorithms, such as PageRank [8] and HITS [17], and their great success for current major search engines, new spamming techniques targeting links became important, and the link farm is one of them. A link farm is a network of web sites which are densely connected with each other. The link farm is one example of the tightly-knit community ("TKC") effect [20]. Since TKCs can have significant impact on ranking results [20, 7, 23], it is necessary to detect link farms and ameliorate their effect on the ranking process.

In this paper, we present ideas of generating a seed set of spam pages and then expanding it to identify link farms. First, we will use a simple but effective method based on the common link sets within the incoming and outgoing links of Web pages for selecting the seed set. Then an expansion step, *ParentPenalty*, can expand the seed set to include more pages within certain link farms. This spamming page set can be used together with ranking algorithms such as HITS or PageRank to generate new ranking lists. The experiments we have done show that this combination is quite resistant to link farms and the "TKC effect".

The rest of this paper is organized as follows: the background and related work will be introduced in Section 2. The impact of link farms on one link-based ranking algorithm will be shown in Section 3. Then the details of our algorithms for detecting link farms are given in Section 4. The experiments and results will be shown in Section 5. We conclude with a discussion and future work.

2. BACKGROUND AND RELATED WORK

2.1 Background

If we consider the Web as a graph, with the pages represented by nodes and the links between pages to be edges which connect nodes, we can use an adjacency matrix M to represent the Web, i.e., $M[i,j]$ is 1 if there is a link from page i to page j, otherwise it is 0.

Based on this adjacency matrix, many link-based ranking algorithms have been proposed and several of them showed good performance when they were proposed. PageRank and HITS are the most famous of these algorithms.

For PageRank: Suppose a page A has incoming links from pages $T_1, T_2, ..., T_n, C(A)$ is the number of links going out of page A, d is a constant with a value between 0 and

WWW 2005, May 10-14, 2005, Chiba, Japan.
ACM 1-59593-051-5/05/0005.

1. Then the PageRank [8] of A may be defined as follows:

$$PR(A) = \frac{(1-d)}{n} + d\sum_{i=1}^{n} \frac{PR(T_i)}{C(T_i)} \qquad (1)$$

In practice, each page's PageRank is combined with other features, such as a textual relevance score to determine a final ranking, such that for pages with equal relevance to a query, the one with highest PageRank will be ranked higher.

For HITS: The set of pages considered in HITS is only a small subset of the whole Web and closely related to a given query [17]. Each page u has two measures: one is the hub score h[u] and the other is the authority score a[u]. The higher authority score for a page, the higher indication that the page is a good authority for this given query. The higher hub score for a page, the more likely that the page points to many good authority pages. Suppose E is the adjacency matrix for this subset, we have the following equations:

$$\vec{a} = E^T\vec{h} \qquad (2)$$

$$\vec{h} = E\vec{a} \qquad (3)$$

After iteration of above two matrices, $\vec{a}$ will converge to the principal eigenvector of E^TE, and $\vec{h}$ will converge to the principal eigenvector of EE^T.

With the great success of link-based ranking algorithms in commercial search engines, a new industry has emerged, called search engine optimization (SEO) [24, 15]. The goal of this industry is to use some techniques to help websites to get better ranking in different search engines. Some examples of these techniques are: using significant titles for Web pages, giving descriptive words in the META tag, etc.

Unfortunately, there is often no clear boundary between legitimate SEO practices and "black-hat" spamming techniques. It may even be the case that some activities permitted at one search engine are against the rules at another. We are concerned with the efforts of some SEO practitioners that build link farms in order to help their web pages to be ranked highly.

2.2 Related work

The idea of "mutually reinforcing relationships" is carefully studied by Bharat and Henzinger in [6], which is the first work to address the issue that HITS can be dominated by special link patterns. A simple but effective algorithm is also given there — to assign an authority weight of $1/k$ if there are k pages from the same host pointing to a single document on a second host and assign a hub weight of $1/l$ if a single document on the first site has l links to a set of documents on a second host. We will call this method BHITS in this paper. Unfortunately, while this method neutralizes certain kinds of mutually reinforcing relationships, it cannot handle larger groups acting in cohort.

The "TKC effect" is first mentioned by Lempel and Moran in [20]. Pages within a tightly-knit community will get high rank value for iterative processes like HITS. A link farm exploits the TKC effect to improve their position in search engine rankings. The authors propose the SALSA algorithm which is more resistant to the TKC effect than HITS. SALSA acts much like a popularity ranking method [11, 7]), but does not incorporate the graph re-ranking that our approach uses to eliminate the effect of link farms.

Chakrabarti proposed using Document Object Models together with hyperlinks to beat nepotistic "clique attacks" [10]. Compared to his idea, our method only takes link structure into account and does not need any other content analysis.

Li et al. [21] found that HITS is vulnerable to the "small-in-large-out" situation. The "small-in-large-out" is the root link that has few in-links but a large number of out-links. Usually the community associated with this root link will dominate the HITS results. They used a revised HITS algorithm that will give these "small-in-large-out" root links different weights to ameliorate their effects. Their approach is still vulnerable to link farms.

In many SEO discussion boards, participants discuss the latest ranking and spam-finding techniques employed by commercial search engines. One approach, called BadRank[1], is believed by some to be used by a commercial engine to combat link farms.[2] The idea is similar in spirit to our mechanism. BadRank is based on propagating negative value among pages. The philosophy of BadRank is that a page will get high BadRank value if it points to some pages with high BadRank value. So it is an inversion of the PageRank algorithm which believes that good pages will transfer their PageRank value to its outgoing links. The formula of BadRank is given as:

$$BR(A) = E(A)(1-d) + d\sum_{i=1}^{n} \frac{BR(T_i)}{C(T_i)} \qquad (4)$$

where $BR(A)$ is the BadRank of Page A. $BR(T_i)$ is the BadRank of page T_i, which is the outbound link of page A. $C(T_i)$ is the number of inbound links of page T_i and d is the damping factor. $E(A)$ is the initial BadRank value for page A and can be assigned by some spam filters. Since no algorithms of how to calculate $E(A)$ and how to combine BadRank value with other ranking methods such as PageRank are given in [1], we cannot tell the effectiveness of this approach. Our *ParentPenalty* has similar philosophy, but it is a more strict spam marking method based on a threshold, rather than a spreading activation.

Gyongyi et al. describe a new algorithm, TrustRank, to combat Web spam [16]. The basic assumption of TrustRank is that good pages usually point to good pages and seldom have links to spam pages. They first select a bunch of known good seed pages and assign high trust scores to them. They then follow an approach similar to PageRank; the trust score is propagated via out-links to other Web pages. Finally, after convergence, the pages with high trust scores are believed to be good pages. Although they used a great amount of data (several billion Web pages) in the experiment and claim that TrustRank can find more good pages than PageRank, there are still some problems with their algorithm. Their algorithm is not appropriate to the Web island pages; In contrast, we do not attempt to recognize good sites, but instead our technique uses a syntactic definition of link spamming behavior that can be automatically detected. In addition, we are able to prevent such behavior from affecting ranking, even when the behavior is found on otherwise known "good" sites.

Zhang et al. [25] propose a revised PageRank algorithm which uses different damping factors for different pages to ameliorate the effect of collusion of Web pages. The basis

[1] One description of BadRank can be found at [1].

[2] See, for example `http://www.webmasterworld.com/forum3/20281-22-15.htm`.

of their algorithm is that colluding pages will have a high correlation with the damping factor in the original PageRank. Although they did some simulation on different kinds of collusion, it is not apparent as to whether their algorithm can find spam which is not so sensitive to the damping factor. In addition, they need to calculate the PageRank values several times, which may be infeasible for large datasets.

Fetterly et al. use statistical analysis to find spam [14]. Several distribution graphs, such as the distribution of the number of different host-names mapping to the same IP address, or the distribution of out-degrees, are drawn. Most of these distributions are modeled well by some form of power law. The outliers of such datasets are marked as spam candidates. By manually checking these candidates, a majority of them are found to be spam. While this may reliably recognize very significant spam, many less significant spam pages or sites will survive.

Amitay et al. [4] propose categorization algorithms to detect website functionality. They recognized that the sites with similar roles exhibit similar structural patterns. After finding one special pattern for a certain kind of website, they can predict a new website to be in the same class if the site shows a similar pattern. For each host, they selected 16 features such as the average level of the page, in-links per page, out-links per leaf page, etc., to do the categorization. In the experiment results, they claimed to have identified 31 clusters, each of which appears to be a spam ring. But which of these 16 features are more significant for identifying spam is unclear.

The work by Kumar et al. [18] can be considered to have included some data cleansing. They eliminated mirror pages to improve quality of their results. Our work can also be considered to be kind of data cleaning, as we identify link farms and drop the links among them before applying a ranking algorithm.

Finally, in prior work [13] we examined the viability of learning to recognize whether a hyperlink would be considered nepotistic. Using a variety of page characteristics, initial experimental results on a small dataset showed significant promise, achieving classification accuracies of over 90%.

3. IMPACT OF LINK FARM

In this section, we will show the effect of the link farm technique on HITS results and the predominant existence of link farm spam in current popular search engine results.

3.1 Link farm effect on HITS

Following the data collecting process in Kleinberg's HITS algorithm, we collected data set for 412 queries. We used Google and Yahoo for getting the top 200 URLs as the root set and also used these search engines for getting the incoming links to expand the root set to generate the base set, then we applied HITS on them and find that usually HITS results are dominated by one or several link farms.

For example, for the query *wireless phone company*, the top 10 authorities are listed in Table 1(a). These 10 sites are strongly connected with each other and they dominate HITS results.

Another example is for the query *credit card application*; the top 10 authorities are listed in Table 1(b). A casino related community dominates this query simply because it is densely connected.

Rank	URL
1	http://www.lowcostwireless4u.com/
2	http://www.cellularratesonline.com/
3	http://www.lowcostwirelessrates.com/
4	http://www.cellphoneonlinerates.com/
5	http://www.cheapwireless4u.com/
6	http://www.newpurple2.com/
7	http://www.red4dir.com/
8	http://www.affordablecellphonerates.com/
9	http://www.affordablecellphonerates.com/ cellular-phone-company.html
10	http://www.lowcostwirelessrates.com/ cell-phone-company.html

(a) Top 10 HITS authorities for *wireless phone company* from Yahoo.

Rank	URL
1	http://www.1001casino.com/
2	http://www.yournetcasino.com/
3	http://www.casino-gambling-online.biz/
4	http://www.internet-gambling-online.biz/
5	http://www.on-line-casino.biz/
6	http://www.the-online-casino.biz/
7	http://www.gaming-zone.biz/
8	http://www.lucky-nugget-casino.com/
9	http://www.casino-game.biz/
10	http://www.luckynugget-online-casino.com/

(b) Top 10 HITS authorities for *credit card application* from Yahoo.

Table 1: Example of top 10 lists dominated by link farms.

3.2 Existence of link farm spam

To show the existence of link farm spam pages in the current search engine top results, we did a simple experiment on a data set of 140 queries, each of which is collected using the process that Kleinberg outlined for HITS, using the Yahoo search engine. We wish to measure how often link spam pages appear in search engine results.

Initially, we will mark a page as a link farm spam page if the domains of three of its out-links match the domains of three of its in-links. After establishing a seed set of such pages, we expand the spam page set by adding pages that have at least 3 outgoing links to the pages that have already been marked as spam pages. Here we only count links pointing to different domains (inter-site links). By this method, many spam pages within the top 200 list for these queries from Yahoo have been found as we describe below.

As shown in Figure 1, we use six levels to describe the existence of spam pages in the search engine results. The x-axis represents the minimum number of spam pages that are found in the top-n list, and the y-axis represents what percentage of the queries had results that included pages marked as spam. Six curves are drawn and each curve represents the top 10, 20, 30, 50, 100 and 200 results from the search engine respectively. The curve at the top is for the top 200. As we can see from the figure, about 68% of all queries have at least one spam page within the top 200 response list from the search engine. About 2% of queries have 7 or more spam pages. The curve at the bottom represents

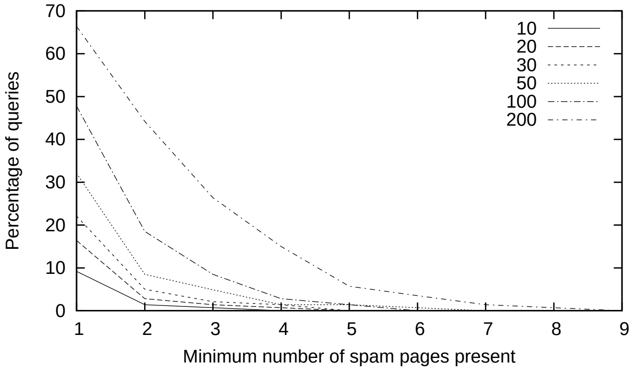


Figure 1: Percentage of queries containing spam pages within top-n results.

the statistics for the top 10 result from the search engine, which usually has the highest likelihood to be visited by search engine users. As we can tell from the figure, about 9% queries have at least one spam page within the top 10 list.

4. MOTIVATION AND DETAILS OF APPROACH

The motivation of our approach is that by observation, we found that pages within link farms are densely connected with each other and many common pages will exist both in the incoming and the outgoing link sets for a page in a link farm. Initially, if we can find some pages within a link farm as the seed set, then for each new page, it is quite possible for the page to be part of the same link farm if it has several incoming and outgoing links from and to the seed set. Then we can expand the seed set by adding the new page into it. This process can be iterative, so more and more pages within the link farm will be found and added. Finally the process will terminate when no more pages will be added. In fact, this mechanism works for a big data set containing many different link farms. The only difference is that we need to find several pages within each of these link farms to form the seed set, then to expand it iteratively to find additional pages within link farms. So the two key aspects of our algorithm are how to generate the seed set and how to expand the seed set.

As mentioned in Section 1 our algorithm has the following steps:

- Generate a seed set from the whole data set.
- The expansion step to propagate the initial badness value to additional pages.
- The ranking step which will combine the badness value together with normal link-based ranking algorithm, such as ranking by popularity, HITS, or PageRank.

4.1 Initial Step: IN-OUT common set

Like BadRank, described earlier in Section 2.2, a seed set is needed within which each page is judged by some spam filters to be a spam page.

Let p to denote the URL for a web page and $d(p)$ represents the domain name of p. Suppose we are given N pages initially. $IN(p)$ and $OUT(p)$ represent the sets of incoming and outgoing links of p, respectively.

1. For each URL i in $IN(p)$, if $d(i) \neq d(p)$ and $d(i)$ is not in $INdomain(p)$, then add $d(i)$ to the set $INdomain(p)$.
2. For each URL k in $OUT(p)$, if $d(k) \neq d(p)$ and $d(k)$ is not in $OUTdomain(p)$, then add $d(k)$ to the set $OUTdomain(p)$.
3. Calculate the intersection of $INdomain(p)$ and $OUTdomain(p)$. If the number of elements in the intersection set is equal to or bigger than the threshold T_{IO}, mark p as a bad page.
4. Repeat 1 to 3 for every page in the data set.
5. For all pages that have been marked bad during 1 to 4, place a 1 in the initial value array $A[N]$. Return A.

Figure 2: Algorithm for finding seed set.

How should one decide which pages should belong to the seed set? We used a simple method. A page usually has several incoming links which point to this page and outgoing links to which this page points. By observation, the pages within link farms usually have several common nodes between the incoming links set and outgoing links set. If only one or two common nodes exist, we will not mark this page as a problem page. But, if many such common nodes exist, it is quite possible that the page is a part of a link farm. A threshold T_{IO} is used here so that whenever the number of common links of the incoming links set and the outgoing links set is equal to or above this T_{IO}, we will mark the page as a bad page and put it into the seed set. In our experiment, we use 3 as the threshold so that we can be able to find link farms with as few as 4 elements.

When matching incoming and outgoing links, we are not limited to an exact match. That is, instead of requiring that two pages point to each other, we might wish to generalize this to allow a page to point to a site or domain, and vice versa. We use the granularity of domain-matching when generating the common set, i.e., if the number of distinct domains in the intersection of the domains of the incoming links and the domains within the outgoing links is equal to or above the threshold for a page, we will mark the page as a problem page and put it into the seed set.

One advantage of using domain name matching instead of URL matching is to catch the common web design technique in which a separate links page is created on which links from amateur or professional link exchanges are placed. Incoming links are mostly to the root page of the site, while links from the home page are typically internal links. Thus, the only way for an external site to recognize the situation is to permit the link target to be a different page within the domain name than for the return link. The complete algorithm is provided in Figure 2.

To make above steps clear, take a look at the simple ex-

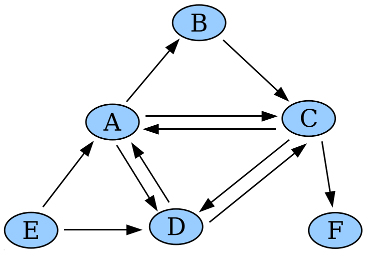


Figure 3: A simple example of 6 pages.

ample in Figure 3. Six pages form a network and we assume that they are all from different domains. For this simple example, we use 2 as the threshold. The incoming link set for node A is [C,D,E] and outgoing link set is [B,C,D]. The common set for node A is [C,D], which has 2 elements and is equal to the threshold, so we put A into the seed set. Similarly, node C and D will also be put into the seed set. Finally the seed set is [A,C,D].

4.2 Expansion Step

With the seed set, an expansion step is needed to find more bad pages within the data set. The intuition for this expansion step is that the structure of real link farms varies and the spam pages may survive the seed set detection. But to make a link farm work, usually pages within a link farm need to point to other pages within the same link farm. If a page only points to one spam page, we will not punish it. But if a page has many outgoing links to bad pages, it is likely that the page itself is with the same link farm. Since we have a seed set, we may enlarge it little by little by adding more pages that have too many outgoing links to these bad pages. Thresholds are used again for judging whether the page will be marked bad.

We use the *ParentPenalty* for the expansion step. The basic assumption is that if one page points to a bunch of bad pages, it is likely that this page itself is bad. Thus, like PageRank, the value of interest spreads from page to linked page, although here we are following incoming links rather than outgoing links. We will use another threshold (T_{PP}) to judge a page: if the number of outgoing links to bad pages meets or exceeds the threshold, the page will be also marked bad. Whenever a new page is marked as bad, the pages that link to it might now meet the threshold. Thus, an iterative process can be used until no more pages are marked bad. We again use a threshold of 3 links.

ParentPenalty:
Suppose we already have an array $A[N]$ from the initial step in which bad pages have value 1 in it and other pages have value 0, and a threshold T_{PP},

```
1. For each member $p$ s.t. $A[p] = 0$, fetch its outgoing links set $OUT(p)$.
2. Set $badnum$=0.
3. For each element $k$ in $OUT(p)$, if $A[k]$ is 1, then increase $badnum$ by 1.
4. If $badnum \geq T_{PP}$, set $A[p] = 1$.
```

Repeat 1 to 4 until the values of A do not change.

Figure 4: Algorithm of *ParentPenalty*.

For this method, we permit the target URL to be in the same domain as the source, and that the targets may all be within the same domain (even the same as the source domain). This enables us to expand the spam set to include other pages from the same site. The *ParentPenalty* algorithm is summarized in Figure 4.

For the simple example in Figure 3, the seed set is [A,C,D]. If we use 2 as the threshold, i.e., the page will be marked bad if it has 2 or more outgoing links pointing to the pages within the seed set. Node B only has one, node E has two, node F has zero. So node E will be marked bad and be assigned a badness value 1. Since no other page will be marked, the *ParentPenalty* algorithm ends up with the bad set as [A,C,D,E].

4.3 Ranking the marked graph

The goal of the previous steps is to find bad pages within a given data set. After finding these bad pages, we need a way to incorporate this information into a ranking algorithm. One obvious mechanism is to change the adjacency matrix of the web graph for this data set. The elements in the adjacency matrix can either be down-weighted or deleted.

Since we have pages marked as participants in link farms, one way is to penalize these pages strictly, i.e., to remove them from the graph. But in reality, this may be too much. For a simple example, consider a business that owns several different web sites for its products or subsidiaries. Often, these web sites will point to each other to form a link farm. So instead of penalizing these pages, we only penalize the links among these web sites, because the web pages themselves may still be good candidates for some real world queries. By penalizing these links, the pages can only get votes from other fair authors and (hopefully) cannot get votes from link farm collaborators.

Two possibilities are available to penalize these links. One is to down-weight them, such as assign a weight proportion to the total number of bad links to each of these links, e.g., a page contains 10 outgoing links related to link farms, then 0.1 is given to these links in the adjacency matrix. Another method is to delete all of them and set 0 to them in the adjacency matrix. Since we have not taken any page content into account in the ranking, and the bad links are crucial to our ranking process, we choose to penalize these links strictly. We delete all links between pages that have been marked as link farm participants from the above steps.

Although our algorithms can detect link farm spam, it is not immune to the "mutual reinforcement" [6] problem. So, we adopt a variant of Bharat and Henzinger's version of HITS in which we assign an edge a weight of $1/k$ if there are k pages from the same host pointing to a single document on a second host.

Now the adjacency matrix is ready for ranking. Several ranking algorithms can be used on this adjacency matrix, such as PageRank, HITS and weighted popularity.

Here "weighted popularity" is a generalization of the simpler approach ranking pages by the number of incoming links. Simple popularity has been explored by others (e.g., [7]) and found to have good performance. Since we have down-weighted the matrix (using a variant of B&H) and even set some link weights to 0 because of the spam pages we have marked, we use the sum of all the weighted values for the incoming links in the adjacency matrix as a metric and rank all pages according to this metric.

We consider this "weighted popularity" ranking algorithm because of its simplicity and speed, and in Section 5, we will show that it always gives better results than the original HITS formulation.

4.4 Comparison of ParentPenalty and BadRank

As we mentioned before, BadRank uses the following philosophy: a page should be penalized for pointing to bad pages. However, it does not specify how far it should go. If page A points to page B, and B points to some known bad pages, it is intuitive to consider B to be bad, but should A be penalized just for pointing to B? For example, a computer science department's homepage points to a student's homepage and the student may join some link exchange program by adding some links within his homepage. It makes sense that the student's homepage should be penalized, but the department's homepage is innocent. In the BadRank algorithm, the department's homepage will also realize some non-zero badness value by propagation upward from one or more other pages with non-zero badness values.

Our *ParentPenalty* idea is more resistant to this issue. A threshold is used in Section 4.2 to decide whether the badness of the child pages should be propagated to a parent. If the number of bad children is equal to or larger than the threshold, then the parent should be penalized. Further, if the number of parents that should be penalized is meets or exceeds the threshold, then the grandparents should be penalized. This also makes sense in real life. So the threshold plays an important role in preventing the badness value from propagating upwards to too many generations.

4.5 Algorithm Complexity

While asymptotically expensive (because of an O(n^2) expansion step), in practice our technique is quite scalable. If we assume that the average number of incoming links and outgoing links are constants, then for each page we can compare the incoming link set and the outgoing link set in a constant number of steps. In that case, the time complexity for seed set selection over all n pages in the collection is just $\Theta(n)$.

We can reduce the cost of seed set selection further through a simple filter. For example, if we use the threshold 3 in seed set selection, then all the pages that have in-degree or out-degree less than 3 cannot be selected for the seed set, so we can mark only the pages with a bigger in-degree and out-degree than 3 as seed set candidates. By using this pre-cleaning method, we can decrease greatly the number of the candidates for the seed set. In our experiments, this pre-cleaning method typically filtered nearly half of the nodes. For the expansion step, we can similarly use a pre-cleaning method to get rid of pages that have fewer outgoing links than the threshold.

At its heart, the expansion process needs to repeatedly examine unmarked pages to see if they point to a sufficient number of marked pages. Fortunately, in practice, this process only needs to be iterated a few times (e.g., 5-6 times for our search.ch domain graph, described below).

5. EXPERIMENTS AND RESULTS

5.1 Data collection

To test our algorithm, we used two data sets. The first one is that we adopted the similar data collecting process as HITS to collect our experiment data. Initially we send a query to `search.yahoo.com` to get top 200 URLs for this query, then for each URL, we get the top 50 incoming links to this URL by querying Yahoo again. We also download all pages referenced by these top 200 URLs. The expanded data set is then used as the base data set for this query. For this experiment, we tried 412 queries and downloaded more than 2.1 million unique Web pages. These queries include queries used by previous researchers [17, 23, 10, 12, 20, 21], the names of the categories from dmoz.org [3], and popular queries from Lycos and Google [2, 19].

The second data set is courtesy of a search engine company in Switzerland: `search.ch`. The search.ch data set contains around 20M pages and 317,640 different domains. We build a domain graph for this data set and applied our link farm detection algorithm on it.

5.2 Sample experimental results

In this section we demonstrate some of the striking results that this technique makes possible.

5.2.1 Query: IBM research center

For the initial step to generate the seed set, only 10 URLs are found. They are shown in Table 2(a). Most URLs in this table are pages for good companies but strongly connected with each other. We also consider this kind of links as forming link farms.

For the expansion step, by using the *ParentPenalty*, 161 more URLs are marked as belonging to link farms. When we checked these URLs manually, all of them wee with one or more link farms based on the initial set. For example, URL number 5, 6 and 7 in Table 2(a) are all part of the same business. After using *ParentPenalty* extension step, several more sites are found which are also part of this link farm: `http://line56.bitpipe.com/`, `http://www.destinationcrm.com/`. For URL No. 1 and No. 2, we found `http://linuxtoday.com/` and `http://www.internet.com/` that are cross-linked with them. A lot of pages within these sites are also marked. After deleting all links within these bad pages and down-weighting all links from the same site to a single page, we get a new adjacency matrix for the data set of this query. By using the "weighted popularity" algorithm, the new top list for the query is generated and shown in Table 2(c).

5.2.2 Query: wireless phone company

This query aims at finding wireless service, such as `http://www.verizon.com/`. Without using any of our algorithm and only using HITS, the top 10 results are shown in Table 1(a). These results are dominated by a set URLs that are strongly connected.

By using our initial step, 16 URLs were identified as seed pages. They are listed in Table 3(a). Interestingly, most URLs from Table 1(a) are caught even in this initial step.

By using the *ParentPenalty* extension step, in total 122 URLs are marked bad. More members within the link farm of Top 10 URLs in Table 3(a) are found, e.g., `http://cellulardirectory.com/`,

ID	URL
1	http://www.phpbuilder.com/
2	http://alllinuxdevices.com/
3	http://www.earthweb.com/
4	http://www.jupitermedia.com/
5	http://www.destinationkm.com/
6	http://www.line56.com/
7	http://www.portalsmag.com/
8	http://www.nwfusion.com/
9	hhttp://www.cio.com/
10	http://www.cxo.com/

(a) Seed set of graph from query *IBM research center* from Google.

Rank	URL
1	http://www.watson.ibm.com/
2	http://www.research.ibm.com/
3	http://www.openoffice.org/
4	http://www.ibm.com/
5	http://www.opensecrets.org/
6	http://www.arborday.org/trees/aerialbenefits.html
7	http://www.nrdc.org/
8	http://www.bushin30seconds.org/
9	http://www-users.cs.umn.edu/ agupta/wsmp.html
10	http://www.almaden.ibm.com/

(b) Top 10 Popularity results before detecting link farms.

Rank	URL
1	http://www.watson.ibm.com/
2	http://www.almaden.ibm.com/
3	http://www.zurich.ibm.com/
4	http://www.ibm.com/
5	http://www.research.ibm.com/
6	http://www.research.ibm.com/nanoscience/
7	http://dimacs.rutgers.edu/
8	http://www.amazon.com/
9	http://www.symantec.com/
10	http://www.infoworld.com/

(c) Top 10 Popularity results after detecting link farms.

Table 2: Results for query *IBM research center.*

ID	URL
1	http://www.cellphoneonlinerates.com/
2	http://www.newpurple2.com/
3	http://www.red4dir.com/
4	http://www.affordablecellphonerates.com/
5	http://www.cellularratesonline.com/
6	http://www.cheapwireless4u.com/
7	http://www.lowcostwireless4u.com/
8	http://www.lowcostwirelessrates.com/
9	http://www.lowcostwireless4u.com/ wireless-phone-service-24.html
10	http://www.cellphoneonlinerates.com/ wireless-handsfree-cell.html
11	http://www.cellphoneonlinerates.com/ wireless-phone-provider.html
12	http://www.cheapwireless4u.com/ cellular-phone-services.html
13	http://www.cellularratesonline.com/ cell-phone-services.html
14	http://www.nexteljobs.com/
15	http://www.cellphoneonlinerates.com/ cellular-phone-service.html
16	http://www.phone-telecom-wireless.net/

(a) Seed set for *wireless phone company* from Yahoo.

Rank	URL
1	http://www.nextel.com/
2	http://www.att.com/
3	http://www.attws.com/
4	http://www.verizon.com/
5	http://www.bellsouth.com/
6	http://www.alltel.com/
7	http://www.nokia.com/
8	http://www.airtouch.com/
9	http://www.sprintpcs.com/
10	http://www.voicestream.com/

(b) Top 10 Popularity results after detecting link farms.

Table 3: Results for query *wireless phone company.*

`http://www.getabooster.com/`, `http://www.affordablecellphonerates.com/`, `http://www.great-cellphone-deals.com/` and `http://www.phone-cards-4u.com/`.

As in the previous example, we delete all links among these pages. The top results for applying "weighted popularity" to the adjusted matrix is in Table 3(b).

5.2.3 Query: picnic

The HITS result for this query is shown in Table 4(a). Most of them are irrelevant to picnic and some of them are densely connected with each other.

Our seed set selection step identified 41 bad URLs which include No. 1, No. 2, No. 3 and No. 8 in Table 4(a). The *ParentPenalty* extension step found 368 bad URLs altogether which including No. 9 in Table 4(a).

After deleting all links among these pages identified by *ParentPenalty* step, the "weighted popularity" ranking algorithm generates the top 10 list shown in Table 4(b).

5.3 Evaluation of our data

To compare the relevance of results generated by our algorithms with the results generated by HITS and BHITS, an evaluation web interface was built in order to allow real users to judge the results.

For this evaluation, 20 queries were selected. For each query, the top 10 URLs generated from Kleinberg's HITS and our "weighted popularity" after dropping links among spam pages are mixed together for blind evaluation. Given a query and a subset of the mixed URL list to this query, real users have five relevancy choices: quite relevant, relevant, not sure, not relevant and totally irrelevant for each URL. Only one choice can be made for each URL, although a user can refrain from making a judgment.

Twenty-eight users participated, generating 541 preferences related to HITS, 469 entries for BHITS and 422 entries related to our "weighted popularity" algorithm. The distribution of preferences is shown is Table 5. Our approach

ID	URL
1	http://www.askthepreschoolteacher.com/
2	http://www.preschoolprintables.com/
3	http://www.preschoolcoloringbook.com/
4	http://www.lehighvalleykids.com/
5	http://www.theperfecttitle.com/
6	http://s12.sitemeter.com/ stats.asp?site=s12preschooledu
7	http://www.freesummervacation.com/to.cgi?l=5a
8	http://www.preschooleducation.com/
9	http://www.preschooleducation.com/ads.shtml
10	http://www.preschooleducation.com/newsletter.shtml

(a) Top 10 results from HITS on original data set.

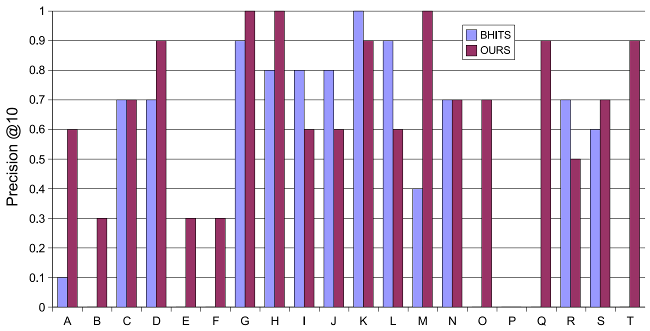
Rank	URL
1	http://www.picnicbackpacksgvshop.com/
2	http://www.google.com/
3	http://www.picnicplaza.com/
4	http://www.alanskitchen.com/
5	http://www.picnic-baskets.com/
6	http://www.fabulousfoods.com/ holidays/picnic/picnic.html
7	http://home.att.net/ cordelli/picnic.html
8	http://www.worldsoffun.com/
9	http://www.steamrollerpicnic.com/
10	http://www.recipeamerica.com/picnic.htm

(b) Top 10 results for weighted popularity after detecting link farms.

Table 4: Results for query *picnic*.

has significantly larger values in the "quite relevant" and "relevant" categories than Kleinberg's original HITS. If we combine the first two categories, we can say that our technique has a precision at rank 10 of 72.6% versus HITS with 23.6% and BHITS 42.8%. This shows that our algorithm generates more relevant top 10 ranking lists for the queries. Moreover, in Figure 5, we find that our approach out-scores BHITS for the majority of queries.

To test the correctness of our algorithm, we check how users think of the sites that our algorithm marked bad. From the evaluation data base, we select URLs that have been marked bad by our algorithm. The distribution of user opinions is in Table 6. While users marked approximately 85% of those nodes as irrelevant, we should point out that in some cases, a node that employes link spamming techniques may still be relevant (and perhaps even authoritative) for a given query. We discuss this issue further in subsection 5.4 where we also provide a few specific examples of this situation.



A. IBM research center
B. US open tennis
C. aerospace defense
D. art history
E. California lottery
F. hand games
G. healthcare
H. image processing
I. jennifer lopez
J. picnic
K. rental car
L. super bowl
M. table tennis
N. teen health
O. translation online
P. trim spa
Q. weather
R. web browser
S. web proxy
T. wine

Figure 5: Query-specific retrieval performance.

Since we have a seed selection step and expansion step, in order to show that the expansion step is necessary, we calculated the statistics for the data without using the expansion step. The results are in Table 7. The sum of "quite relevant" and "relevant" is only 70.7%, which is lower than 72.6%, generated with the expansion step.

Some may argue that the pages pointing to bad pages are not necessarily bad themselves. In order to test this argument, we also tried a new expansion step — in order to be marked bad, a node not only needs to point to at least the number of bad sites, but also should be pointed back by some of these bad sites. We did tests that require the node to be referenced by at least one bad site besides pointing to three bad sites for us to mark it bad in the expansion step. Intuitively, fewer sites will be marked bad this time because stronger conditions must be met for a node to be marked bad this time. But the evaluation shows that this new step does not generate better results and even generates slightly worse results (when the top 30 results are considered). So in our experiments we still use the expansion step algorithm shown in Figure 4.

The selection of a threshold is also a problem in our algorithms. In order to show the effect of different thresholds, we used different thresholds for the seed selection step and expansion step. For each of them, we tried six different values, from 2 to 7. Altogether, we tried 36 different combinations of thresholds. The user evaluation of the accuracy of using different thresholds are shown in Figure 6. It is obvious that with larger threshold, fewer sites will be marked and so more spam sites may occupy the top ranking positions. As we can tell, the accuracy is decreasing with the increasing value of the threshold.

Category	HITS	BHITS	OURS
quite relevant	12.9%	24.5%	47.1%
relevant	10.7%	18.3%	25.5%
not sure	6.6%	10.5%	7.3%
not relevant	26.8%	14.8%	12.0%
totally irrelevant	42.8%	31.9%	7.8%

Table 5: Evaluation results.

Opinion	Percentage
quite relevant	5.4%
relevant	4.8%
not sure	4.1%
not relevant	23.7%
totally irrelevant	61.7%

Table 6: User evaluation for sites marked as bad.

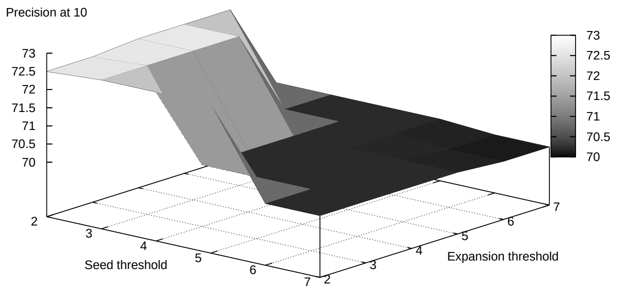


Figure 6: Retrieval performance as thresholds vary.

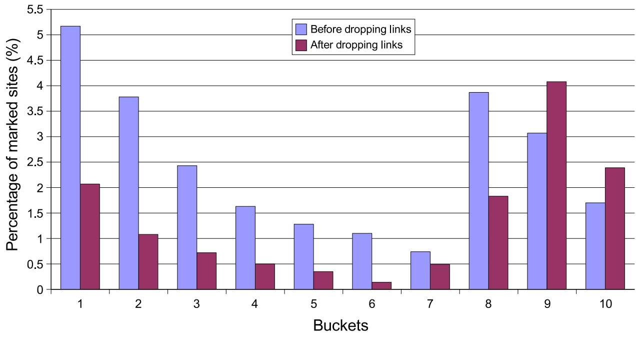


Figure 7: Distribution of sites marked by our algorithm.

5.4 Results for search.ch data

In order to show that our algorithm is also useful for global ranking algorithms, such as PageRank, we applied our algorithm on the search.ch data set.

If we set the threshold both to be 3 for the initial seed set selection step and the expansion step, 27,568 sites were identified in the seed set selection step and an additional 42,833 sites were marked during the expansion step.

We calculated PageRank for both the original domain matrix and the domain matrix after dropping the links between the nodes that have been marked by our algorithm.

First, we show the distribution of 70,401 bad sites for the two domain matrices. We put the nodes into 10 buckets according to their PageRank value, and the sum of the nodes PageRank value of each bucket equal to 10% the total PageRank value for the whole graph. We organize this with a decreasing order, i.e., the nodes in bucket 1 have the highest PageRank value and the nodes in bucket 10 have the lowest PageRank value. Figure 7 shows the distribution of the number of nodes within each bucket. The light bar represents the distribution for the original domain graph; the dark bar represents the distribution for the domain matrix after dropping links among the bad sites marked by our algorithm.

The philosophy of our algorithm is to find densely connected components of the web graph and drop links among them because we believe that the links from these densely connected components is not objective enough to be a good vote. If the Web site is truly good, the links from other sites out of the community will still make the site ranked high. In Figure 7, we see that there are nodes, though marked bad by our algorithm, in each bucket that do not move for the two domain matrix calculation. Ideally, these sites should be the ones that do not rely on the densely connected components to get rank. To verify this, we manually checked the 52 nodes that remain in the first bucket. Almost all of them are good sites. For example, the domain `http://www.admin.ch/` is the Authorities of the Swiss Confederation web site and should be ranked very high. Our algorithm marked it bad because it points to the sites like `http://www.snl.ch/` (Swiss National Library), `http://www.zivil-dienst.ch/` (Civilian service) and `http://www.swiss-energy.ch/` (Federal Office for Energy) and all these sites point back to `admin.ch`. Obviously all these sites are government sites and they link to each other for political and administration reasons, not necessarily for authority vote. Even though our algorithm dropped these links, the admin.ch is still ranked in the top bucket for the whole data set.

We also obtained a list of spamming domains from the search.ch company. They listed 732 sites as spam sites. Among them, 728 are found during the seed set selection step and two more are identified with the expansion step. We also draw a distribution of these 732 sites for the domain matrix before using our algorithm and after using our algorithm. Figure 8 has similar buckets as Figure 7, but we use percentage instead of absolute number. As we can tell, the light color bars, which represents the percentage of these spam sites for the original domain matrix, have a more even distribution among buckets from 5 to 8, while the dark color bars, which represents the percentage after using our algorithm, have much higher value in the last two buckets. This shows most of the spam sites are moved to the lowest level after using our algorithm.

Opinion	Percentage
quite relevant	45.9%
relevant	24.8%
not sure	7.8%
not relevant	13%
totally irrelevant	8.3%

Table 7: Users evaluation for results calculated with only seed set.

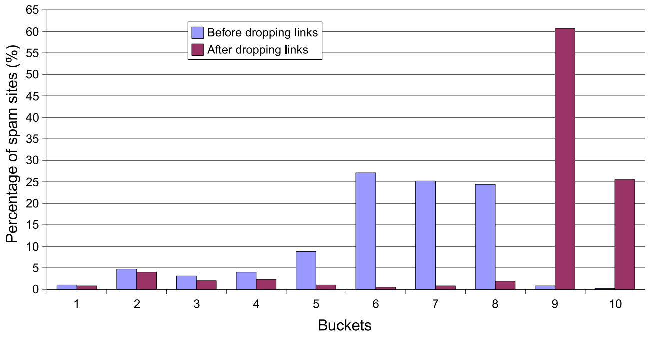


Figure 8: Distribution of sites on the blacklist.

6. CONCLUSION AND DISCUSSION

This paper has demonstrated the utility of our technique of finding and expanding upon a seed set of likely spamming pages.

One issue to discuss is that it is appropriate to punish links among link farms, but not good to punish some web pages. For example, most of the URLs in Table 2(c) are well-known companies. They are not good for the query *IBM research center*, but they are quite good for a query like *jupiter media*. So, we should not remove these web sites from the index just because they link with each other, but it makes sense to delete links among the pages in Table 2(c) to prevent them from getting authority value from collaborators. These pages should only get votes from pages outside their business in the ranking system. If many other pages point to them, they will still be ranked high and be hit for queries like *jupiter media*.

One weakness of the algorithms in this paper is that duplicate pages cannot be detected. For example, often pages in `dmoz.com` will be copied many times, but these pages do not connect with each other so we do not mark them as spam. As a result of the duplication, the targets cited by these pages will be ranked highly. In general, duplicate pages should be eliminated before using our algorithm to generate better ranking (e.g., using methods from [9, 5]).

Acknowledgments

We are grateful to Urban Müller for helpful discussions and for providing access to the search.ch dataset.

7. REFERENCES

[1] Pr0 - Google's PageRank 0, 2002. http://pr.efactory.de/e-pr0.shtml.

[2] Lycos 50, 2005. http://50.lycos.com/.

[3] Open directory project, 2005. http://dmoz.org/.

[4] E. Amitay, D. Carmel, A. Darlow, R. Lempel, and A. Soffer. The connectivity sonar: Detecting site functionality by structural patterns. In *Proceedings of the Fourteenth ACM Conference on Hypertext and Hypermedia*, pages 38–47, Aug. 2003.

[5] K. Bharat, A. Z. Broder, J. Dean, and M. R. Henzinger. A comparison of techniques to find mirrored hosts on the WWW. *Journal of the American Society of Information Science*, 51(12):1114–1122, 2000.

[6] K. Bharat and M. R. Henzinger. Improved algorithms for topic distillation in a hyperlinked environment. In *Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 104–111, Melbourne, AU, Aug. 1998.

[7] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Finding authorities and hubs from link structures on the world wide web. In *Proceedings of the 10th International World Wide Web Conference*, pages 415–429, May 2001.

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

[9] A. Broder, S. Glassman, M. Manasse, and G. Zweig. Syntactic clustering of the web. In *Proceedings of the Sixth International World Wide Web Conference*, pages 1157–1166, 1997.

[10] S. Chakrabarti. Integrating the document object model with hyperlinks for enhanced topic distillation and information extraction. In *Proceedings of the 10th International World Wide Web Conference*, pages 211–220, 2001.

[11] S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, San Francisco, CA, 2003.

[12] D. Cohn and H. Chang. Learning to probabilistically identify authoritative documents. In *Proc. 17th International Conf. on Machine Learning*, pages 167–174. Morgan Kaufmann, San Francisco, CA, 2000.

[13] B. D. Davison. Recognizing nepotistic links on the Web. In *Artificial Intelligence for Web Search*, pages 23–28. AAAI Press, Jul 2000. Technical Report WS-00-01.

[14] D. Fetterly, M. Manasse, and M. Najork. Spam, damn spam, and statistics: Using statistical analysis to locate spam web pages. In *Proceedings of WebDB*, pages 1–6, June 2004.

[15] Z. Gyongyi and H. Garcia-Molina. Web spam taxonomy. Technical report, Stanford Digital Library Technologies Project, Mar. 2004.

[16] Z. Gyongyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with TrustRank. In *Proceedings of the 30th VLDB Conference*, Sept. 2004.

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

[18] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the Web for emerging cyber-communities. *Computer Networks*, 31(11–16):1481–1493, 1999.

[19] D. Lemin. Google Zeitgeist, Dec. 2004. http://www.google.com/press/zeitgeist.html.

[20] R. Lempel and S. Moran. The stochastic approach for link-structure analysis (SALSA) and the TKC effect. *Computer Networks*, 33(1–6):387–401, 2000.

[21] L. Li, Y. Shang, and W. Zhang. Improvement of HITS-based algorithms on web documents. In *The eleventh International World Wide Web Conference*, pages 527–535. ACM Press, 2002.

[22] A. Perkins. White paper: The classification of search engine spam, Sept. 2001. Online at http://www.silverdisc.co.uk/articles/spam-classification/.

[23] G. O. Roberts and J. S. Rosenthal. Downweighting tightly knit communities in world wide web rankings. *Advances and Applications in Statistics*, 3(3):199–216, Dec. 2003.

[24] D. Sullivan. Search engine optimization, Apr. 2000. Online at http://searchenginewatch.com/resources /article.php/2156511.

[25] H. Zhang, A. Goel, R. Govindan, K. Mason, and B. V. Roy. Making eigenvector-based reputation systems robust to collusions. In *Proceedings of the Third Workshop on Algorithms and Models for the Web Graph*, Oct. 2004.