Webspam Demotion: Low Complexity Node Aggregation Methods

Thomas Largillier
Univ Paris-Sud, LRI;
CNRS, INRIA;
Orsay, F-91405
thomas.largillier@lri.fr

Sylvain Peyronnet
Univ Paris-Sud, LRI;
CNRS, INRIA;
Orsay, F-91405
syp@lri.fr

***Abstract*—Search engines results pages (SERPs) for a specific query are constructed according to several mechanisms. One of them consists in ranking Web pages regarding their importance, regardless of their semantic. Indeed, relevance to a query is not enough to provide high quality results, and popularity is used to arbitrate between equally relevant Web pages. The most well-known algorithm that ranks Web pages according to their popularity is the PageRank.**

The term *Webspam* was coined to denotes Web pages created with the only purpose of fooling ranking algorithms such as the PageRank. Indeed, the goal of Webspam is to promote a target page by increasing its rank. It is an important issue for Web search engines to spot and discard Webspam to provide their users with a non biased list of results. Webspam techniques are evolving constantly to remain efficient but most of the time they still consist in creating a specific linking architecture around the target page to increase its rank.

In this paper we propose to study the effects of node aggregation on the well-known ranking algorithm of Google (the PageRank) in presence of Webspam. Our node aggregation methods have the purpose to construct clusters of nodes that are considered as a sole node in the PageRank computation. Since the Web graph is way to big to apply classic clustering techniques, we present four lightweight aggregation techniques suitable for its size. Experimental results on the WEBSPAM-UK2007 dataset show the interest of the approach, which is moreover confirmed by statistical evidence.

I. Introduction

Search engines are designed to provide users with results of the finest quality, *i.e.* Web pages highly relevant to the user's query. To achieve this task, Web pages are sorted with respect to their relevance towards the requests. Unfortunately, the result of this sorting step can be fooled since malicious webmasters (called spammers from now on) have the ability to index and make Web pages relevant to many (i.e. a really huge number of) requests. To avoid large scale manipulation over the relevance of Web pages, search engines use another metric to rank Web pages. This metric is often based on some kind of pages' popularity. Popularity together with relevance are used to sort results before presenting them to the users.

Nowadays it is really important for a website to have a good visibility to ensure its popularity and to attract some traffic. Being visible often means to be well ranked regarding requests on Web search engines. Without any surprise a site appearing on the first two result pages of Google will be more visited than a site on the 100^{th} page for the same request.

It is also important for a website to drain a sufficient amount of traffic since earnings on the Web is often proportional to the number of visitors. This leads webmasters, who wants to earn money on the Web, to take extra care of their ranking on Web search engines.

There are not many actions a webmaster can do to increase its relevance towards requests without quickly falling into spamming. Plus improving its relevance has its limits, once you are relevant towards the requests you're interested in, you will appear into the results list. Then you need to improve your rank in the list.

The popularity is often based on a structural criterion. Webmasters can improve their popularity by making some publicity for their site. It means that they are trying to achieve for other sites to "vote" for theirs in order to improve its popularity. There is a lot of mechanisms that they can use to ensure a high rank to their pages. Many of these mechanisms depend on the targeted ranking algorithms.

As soon as a ranking algorithm is known people will ask themselves how to maximize their score. This is also true for the PageRank. This question has been resolved years ago either for a single page by Gyongyi *et al* in [9] or for a whole website by De Kerchove *et al* in [6].

Orthodox, but borderline, techniques designed to increase the pagerank of some pages on the Web are regrouped under the name of *search engine optimization*. Not far from search engines optimizers (SEO) lie Web spammers. They are people whose goal is to promote a page or a site, regardless of the techniques used for that purpose. Many well-known techniques like link farms are well spread amongst Web spammers. These techniques evolve quickly making exact automatic detection hard in practice.

The frontier between SEO and Webspam is thin but creating a whole network of dull pages just to increase a target page or site pagerank can clearly be seen as Webspam

(moreover, according to Google, it is Webspam[1]).

One of the major challenge for search engines nowadays is the fight against Webspam. In order not to drive away frequent users, the results should not be polluted by spam pages. It is also important that Web pages are presented respecting a fair ranking since users that do not use a specific request may be more interested with genuinely popular pages.

The main goal of this paper is to propose an approach based on graph clustering to demote the effects of Web spamming and show its efficiency. We present statistical evidence of its viability at identifying Webspam in an automatic way. Our approach does not need any human assisted step.

This paper is an extended version of [14]. We propose in this extended version an extra clustering technique together with a complexity analysis of the four techniques. We also strengthen the experiments by adding a new dataset that contains more interesting nodes and that forces us to partly reconsider the conclusion we made in the previous article and a comparison to techniques presented in the Web spam challenge 2008[2].

The following of this paper is organized as follows, in section II we present work related to ours regarding Webspam detection and demotion. In section III we introduce the lightweight clustering methods we chose to use in order to demote Webspam. Section IV shows our experiments and their results. In section V we present statistical evidence of the viability of our approach. In section VI, we compare our approach to existing detection techniques, we also discuss the results we obtained on a peculiar subset of the dataset. Finally we conclude in section VII.

II. Related Work

Since the PageRank algorithm was introduced in [17], and became famous through its use within the search engine Google, Web spammers tried to figure out how to increase their rank. The question of how to maximise the rank of a target page has been answered in [9] where they also proposed an analysis of the Webspam. If someone wants to maximize the score of an entire Web site and not a single Web page it has to use a more complicated linking structure between the site pages. The optimal structure is described in details in [6].

With the apparition of Webspam, many techniques were developed to detect or demote the effects of Webspam in order to ensure the user with a fair ranking. These measures can be separated in three categories, demotion, detection and prevention as stated in [12].

[1]It can be read in Google webmaster guidelines: *"Don't participate in link schemes designed to increase your site's ranking or PageRank. In particular, avoid links to web spammers or "bad neighborhoods" on the web, as your own ranking may be affected adversely by those links"*.

[2]http://webspam.lip6.fr

A first kind of measures to appear was the propagation of Trust or Distrust proposed by Gyongyi *et al.* in [11][13]. Those methods need a human preprocessing step that help to split a small subset of nodes between good (to be trusted) nodes and bad (not to be trusted) nodes. Then starting from the seeds, either the Trust is propagated or the Anti-Trust is propagated backward.

Wu *et al.* propose in their paper [19] an improvement of the TrustRank algorithm where topicality is considered to increase the results. Their results outperform those given by the TrustRank algorithm but the authors are using a human powered preprocessing step, making the method difficult to use in practice (even harder than the TrustRank).

Gyongyi *et al.* propose an other approach. They present in [10] a framework where the fraction of pagerank coming from spam pages is computed for each Web page. This again requires a preprocessing human step where people label pages as spam or nonspam. The estimation of the fraction is calculated by evaluating the pagerank of each page when the source of pagerank is a subgraph composed only of good pages.

Ntoulas *et al.* present in [16] a classifier for Web pages based on their content. They chose many criteria going from the number of words in the title to the conditional n-grams likelihood. Their classifier has a high precision and recall but the problem is that the target page of spammers (that is the page whose pagerank is boosted by unorthodox techniques) is often a relevant page and thus falls out of the scope of the method. Moreover, using the classifier on every Web page is fastidious and not really tractable.

Abernethy *et al.* describe in [1] their method to identify Webspam pages using both content and hyperlinks analysis. Their algorithm, called WITCH, is based on machine learning and obtain really good results in differenciating spam pages from nonspam ones.

Martinez-Romo *et al.* propose in [15] a method to identify Webspam using Language Model Analysis. They compute several metrics associated with the language of the page, its title, the surrounding anchor text, *etc*. Then using a Kullback-Leibler divergence they separate genuine pages from the spam ones.

Finally, Benczur *et al.* (see the paper [3]) propose an fully automatic method to detect Webspam. This method proceeds by observing the distribution of suspected pages' contributors. Web pages with a biased distribution are considered spam.

All these methods fall into the scope of spam detection since they attempt to identify Webspam pages in order to reduce their influence. However, there is another way of fighting Webspam. Indeed, one can consider methods with a different goal: demoting the effects of Webspam without necessary detecting it.

Andersen *et al.* propose in [2] an algorithm called Robust PageRank. It is designed to fight link spam engineering.

They use the supporting sets of nodes (i.e. nodes contributing to the pagerank of a specific Web page) regarding the pagerank computation. They locally compute approximate features in order to demote the effects of Webspam. For instance, they examine the size of a node's supporting sets and the approximate l_2 norm of the PageRank contributions from other nodes.

Lu *et al.* prove in [18] that computing the PageRank using nonlinear coefficients regarding the correlation that may exists between links pointing to the same page. This could be use to reduce the importance of spam pages created to artificially increase the weight of the target page.

Chung *et al.* (see the paper [5]) have made a study of link farms during a period of two years to see how their distributions and compositions are evolving. They use the Strongly Connected Component (SCC) decomposition to find and study such link farms. Tarjan's algorithm complexity is $\mathcal{O}(n + m)$ where n is the number of nodes of the graph and m the number of edges. This is a good theoretical complexity but still too high to be used to fight Webspam since the algorithm should be iterated to find link farms (the graph is too large to be the input of an algorithm of this complexity). But the idea of clustering the graph to identify link farms is of the utmost interest. It is important to find a faster way to regroup nodes, this is the problem we address in this paper.

III. Clustering Methods

In this section we present the four graph clustering techniques we used in our experiments. First, it is important to notice that classical and efficient clustering methods for small graphs such as *the Markov Cluster Algorithm* (MCL, see [7]) and the *edge betweenness clustering* method (EBC, see [8]) are unsuitable for the web graph because of its size. Indeed MCL requires an explicit matrix representation of the graph which is totally infeasible in our case and EBC runs in time and space of $\mathcal{O}(nm)$ where n is the number of nodes in the graph and m the number of edges (these notations will be kept throughout the paper), which is in practice totally intractable. Moreover we are not interested in an exact clustering. Our interest is not the detection of Webspam but its demotion. If we can group enough Webspam with the target page, building big enough communities, we hope to stop a sufficient amount of incoming pagerank to nullify the Webspam's effects.

Google has indexed more than 1000 billion pages[3]. So every technique must have a low complexity, *i.e* linear at maximum. Indeed the PageRank has a linear complexity $\mathcal{O}(n + m)$. Since the PageRank must be calculated to offer a ranking to users, every method which purpose is to demote the effect of Webspam must add at most a constant amount of calculation to be effective. The ideal case would be a method that could be embedded with the PageRank computation with only a constant overhead and no memory usage.

All four methods we present below are local algorithms computed for every node in the graph. The idea is always to have a very simple criterion to group nodes together, starting from a peculiar node. It is also important to use only local knowledge to compute our clusters.

Fig. 1 shows example of how nodes are regrouped for each method. The red node labelled S is the starting point of the algorithm and the blue nodes are the ones regrouped with it after the computation.

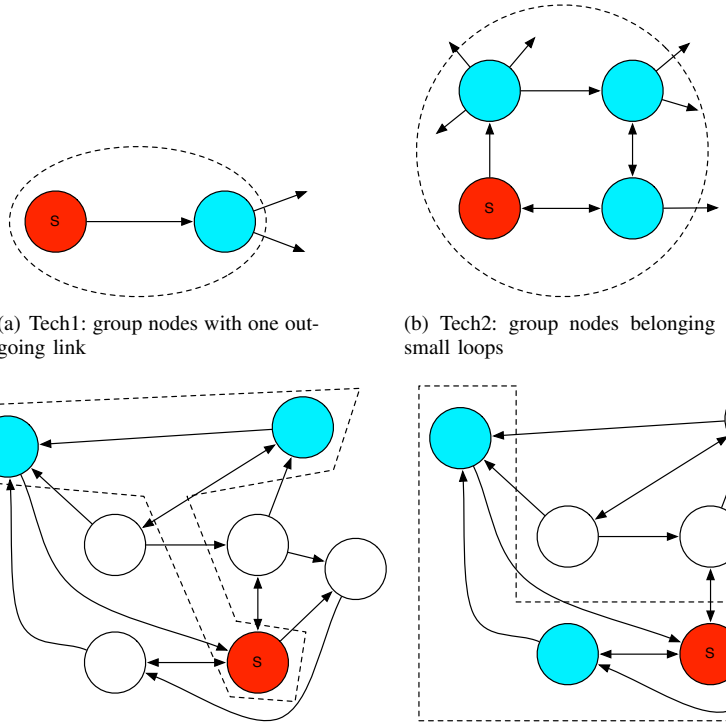
The first method we propose is very intuitive. We only want to group nodes with one outgoing link with the target of this link (see Fig. 1(a)). This means that we group people that give all their pagerank to one person with this person. A well-known technique to raise the pagerank of one page is to create a lot of dummy pages that will give all their pagerank to the target page. In the following this method will be referred to as Tech1. Note that it can be embedded for free during the computation of PageRank.

The second technique we want to test regroup nodes that belongs to short loops in the graph. In Fig. 1(b) the length of the loop is 4. For every node in the graph we compute every path of length k and if the path ends on the starting node then everybody in the loop goes in the same cluster. We know that spammers don't like to waste pagerank, thus many links coming out the target page should return to the target page in a few steps. In the experiments we chose to use a length $k = 3$. In the rest of this paper this method will be called Tech2.

For the third method we simply launch r random walks of length l from every node in the graph. If the number of random walks that ends on a particular node is higher than a threshold t then this node and the starting node are regrouped in the same cluster. With this approach we hope to regroup Webspam with their target page even if some links may lead elsewhere to avoid automatic detection of well known structures. Following links from a Webspam page will lead to the target page with high probability. Later in the article this technique will be named Tech3. During our experiments we launched 200 random walks of length 15. The threshold was fixed at 40 meaning that more than 1/5 of the walks must end on the same node for it to be regrouped with the starting node.

This threshold was chosen independently from the random walk length, we just wanted a number high enough to be suspicious. There exists a relationship between the best threshold and the length of the random walk but it implies some knowledge on the mixing time of the random walks, which we do not possess. It seems possible to compute this piece of information for some particular link farms, but it is not of interest here since we want our technique to be as general as possible to remain efficient while cheater's

[3]http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html



(a) Tech1: group nodes with one outgoing link

(b) Tech2: group nodes belonging to small loops

(c) Tech3: group nodes reached too often as ending point of RWs

(d) Tech4: group nodes leading to popular ending points of RWs

Figure 1. Lightweight clustering techniques grouping blue nodes with the starting point (red node labelled S)

techniques evolve.

At last, the forth technique is very similar to Tech3. It is based on the same principle, *i.e.* launching short random walks, but we group all nodes that lead to the often touched target with the starting point. This method creates bigger groups than the previous since for the same random walks, much more nodes will end up in the same clusters. We think it will merge close clusters that should belong together. This method is illustrated in Fig.1(d), where all nodes in the random walk leading from the starting node to the top left node are regrouped together.

More formally at the beginning of each algorithm every node belongs to its own cluster. When we regroup nodes we simply merge their clusters following the expression "Any friend of yours is a friend of mine". This has no impact if the starting point of the algorithm has no neighbors in the cluster of the node it wants to regroup with but, on the other hand if it wants to regroup with someone who is very close to one of its successors the starting node probably wants to associate itself with that particular neighbor. Thus two nodes can end up in the same cluster even if the method did not explicitly regroup them.

A. Complexity analysis

We will now analyse the complexity of all the methods presented above. The complexity of Tech1 noted $C(T_1)$ is obviously linear in the number of nodes of the graph since it only does one comparison by node, thus $C(T_1) = \Theta(n)$.

In order to compute the global complexity of Tech2 ($C(T_2)$), we will first look at the complexity of Tech2 for the node i ($C_i(T_2)$).

$$C_i(T_2) = 1 + |d_i^+| + \underbrace{\sum_{j \in d_i^+} (|d_j^+| + \sum_{l \in d_j^+} (|d_l^+| + \dots}_{k-1 \text{ steps}}$$

where d_i^+ represents the neighbors of node i and thus $|d_i^+|$ the outdegree of node i. With this number of operations we can only establish that one step of the algorithm may run in time $\mathcal{O}(n+m)$ since one node may be able to reach all the nodes in the graph within k steps. This assumption is obviously false for the Web graph and a small value of $k < 10$.

Let's take $d_{\max} = \max_i |d_i^+|$. We then have,

$$\begin{aligned} C_i(T_2) &\leq d_{\max} + \underbrace{\sum_{j=1}^{d_{\max}} (d_{\max} + \sum_{l=1}^{d_{\max}} (d_{\max} + \dots}_{k-1 \text{ steps}} \\ &\leq \sum_{j=0}^{k} d_{\max}^j \\ &\leq \frac{d_{\max}^{k+1} - 1}{d_{\max} - 1} \\ &= \mathcal{O}(d_{\max}^k) \end{aligned}$$

It is a fair assumption that $d_{\max}$ is a constant independent from the size of the graph. However one can set the value $d_{\max}$, and not consider nodes i with an outdegree $|d_i^+| > d_{\max}$ during the computation. Regarding the global complexity of Tech2,

$$\begin{aligned} C(T_2) &= \sum_i C_i(T_2) \\ &= \mathcal{O}(n \times d_{\max}^k) \end{aligned}$$

The complexity of the last two methods is linear in the number of nodes since the operations made by each node are random walks of fixed length. Indeed the complexity of Tech3 for node i is,

$$C_i(T_3) = \mathcal{O}(r \times l)$$

time required to launch r random walks of length l the $\mathcal{O}(r \times l)$ is here for the postprocessing of the random walks that requires $\mathcal{O}(r)$ steps: finding stop nodes above the threshold and regroup them with the starting point. Thus the total complexity of Tech3 is,

$$C(T_3) = \mathcal{O}(n \times r \times l)$$

The complexity of Tech4 for a node i is the same as Tech3's since is it also based on random walks. The postprocessing is a bit greater but is still in $\mathcal{O}(r \times l)$ leading to the following global complexity,

$$C(T_4) = \mathcal{O}(n \times r \times l)$$

It is important to notice here that since r and l are small constant, the last two techniques achieve a good complexity but Tech2 may take some time for nodes with high outdegrees. While having a linear complexity in terms of nodes, the constant $d_{\max}$ is huge so the technique may be a bit longer than the others.

IV. Experiments

We present in this section experiments that have been conducted on the dataset WEBSPAM-UK2007[4]. This dataset is a crawl of the .uk domain made in May 2007. It is composed of 105 896 555 nodes. These nodes belong to 114 529 hosts and 6 478 of these hosts have been tagged. Please pay attention to the fact that hosts are tagged, not pages (e.g. entire domains instead of peculiar pages). We use the Webgraph [4] version of the dataset by Boldi and Vigna since it allows to manipulate huge graphs without using a lot of memory.

The tagged hostnames are separated in 3 user-evaluated categories: *spam* (690 972 nodes), *nonspam* (5 314 671 nodes) and *undecided* (201 205 nodes). Using this information we can construct 3 sets of web pages corresponding to the 3 categories. We also add another set to the three existing ones. We call it *spammed* and it is constituted of all pages at a maximum distance of 1 step from the set *spam* while not belonging to *spam*. It is composed of 49 193 nodes. But the intersection of this new set with the *nonspam* set is non empty. Thus we withdraw all nodes in this intersection from the *spammed* set. After that operation, it remains 48 164 nodes left in this set. Thus one should keep in mind that it may still be suspicious nodes in the *nonspam* set.

We first evaluate the pagerank of each node of our dataset. Then we sort each set *spam*, *nonspam*, *undecided* and *spammed* in decreasing pagerank value.

Then we apply each technique to the graph before computing a special version of the pagerank where i contributes to the pagerank of j iff $i \to j$ and i and j are in separate clusters. The contribution C_{ij} of i to j is the following: $C_{ij} = \frac{Pr(i)}{k_i}$ where $k_i = |\{j | i \to j, \text{cl}(i) \neq \text{cl}(j)\}|$ and $Pr(i)$ is the pagerank of page i. This is the same as running the PageRank on a graph where all intra clusters edges have been removed. Results can be found in Tab. I. We do not use the normalized version of the PageRank where they all add up to 1 since we make the computation over a huge graph and don't want to be limited by the machine precision. All $‰$ in Tab. I don't add up to one since we consider only a fraction ($\sim 5.91\%$) of all pages (the tagged web pages plus the *spammed* set). It can be seen in this table that every method make the pagerank of all four sets drop. This is easily understandable. Since many edges are removed from the graph, the pagerank can not spread as easily.

Tech1 reduces the whole pagerank of the graph of $\sim 18\%$, Tech2 reduces it by almost 43%, Tech3 by $\sim 10\%$ and Tech4 by $\sim 27\%$. These are the average demotions we register on the whole graph for each technique. The average demotion depends on the number of clusters made by the algorithm

[4]Yahoo! Research: "Web Spam Collections". http://barcelona.research.yahoo.net/webspam/datasets/ Crawled by the Laboratory of Web Algorithmics, University of Milan, http://law.dsi.unimi.it/. URLs retrieved 05 2007.

	Webgraph	*spam*		*nonspam*		*undecided*		*spammed*	
		value	‰	value	‰	value	‰	value	‰
PageRank	84 015 567.786	517 546.3795	6.16	4 230 292.491	50.35	167 809.751	2	876 574.001	10.43
Tech1	68 943 484.072	422 932.8076	6.13	3 449 440.644	50.03	141 221.9441	2.05	627 889.2	9.11
Tech2	48 431 264.361	294 940.5303	6.09	2 323 473.016	47.97	97 550.25296	2.01	821 191.761	16.96
Tech3	75 176 329.382	461 598.8212	6.14	3 809 062.589	50.67	150 273.0357	2.00	807 620.526	10.74
Tech4	62 371 681.151	350 894.3076	5.63	3 178 787.526	50.97	109 720.224	1.76	626 092.777	10.04

Table I
PAGERANKS OF EACH SET

	20%				30%			
	Total		Intersection		Total		Intersection	
	Nombre	Score	Nombre	Score	Nombre	Score	Nombre	Score
PageRank	56	104 580	46	89 726	186	155 798	147	128 351
Tech1	66	85 451.5	46	74 794.4	210	127 193	147	111 833
PageRank	56	104 580	36	87 877.8	186	155 798	142	136 765
Tech2	49	59 648.9	36	42 502.1	203	88 731	142	63 700.1
PageRank	56	104 580	52	99 496	186	155 798	161	143 803
Tech3	65	92 937.9	52	84 089.2	189	138 824	161	128 264
Pagerank	56	104 580	43	70 442.7	186	155 798	129	127 187
Tech4	122	70 712.6	43	37 777.6	343	105 399	129	70 212.1

Table II
EFFECTS OF DIFFERENTS TESTS ON SPAM TAGGED PAGES

	20%				30%			
	Total		Intersection		Total		Intersection	
	Nombre	Score	Nombre	Score	Nombre	Score	Nombre	Score
PageRank	958	846 644	799	699 145	2 433	1 269 460	1 901	1 062 620
Tech1	1 049	690 274	799	596 293	2 776	1 035 060	1 901	892 642
PageRank	958	846 644	312	389 323	2 433	1 269 460	1 030	745 638
Tech2	538	465 120	312	331 698	1 605	697 270	1 030	547 989
PageRank	958	846 644	763	714 624	2 433	1 269 460	2 028	1 087 260
Tech3	830	762 651	763	728 457	2 166	1 142 930	2 028	1 102 770
Pagerank	958	846 644	510	506 426	2 433	1 269 460	1 381	803 932
Tech4	859	636 190	510	452 925	2 289	953 812	1 381	704 729

Table III
EFFECTS OF DIFFERENTS TESTS ON NONSPAM TAGGED PAGES

	20%				30%			
	Total		Intersection		Total		Intersection	
	Nombre	Score	Nombre	Score	Nombre	Score	Nombre	Score
PageRank	49	34 156.5	42	29 705.4	109	50 569.9	96	45 923.6
Tech1	48	28 748.5	42	26 154.3	118	42 566.2	96	39 577.6
PageRank	49	34 156.5	16	13 851.7	109	50 569.9	49	31 952.9
Tech2	23	19 875.7	16	14 271.6	66	29 427.1	49	25 522.6
PageRank	49	34 156.5	38	27 011.7	109	50 569.9	85	41 865.5
Tech3	45	30 774.8	38	26 741.8	111	45 230.8	85	41 002
Pagerank	49	34 156.5	32	20 373.9	109	50 569.9	57	28 535.1
Tech4	40	22 432.6	32	18 868.7	151	32 987.8	57	25 397.6

Table IV
EFFECTS OF DIFFERENTS TESTS ON UNDECIDED TAGGED PAGES

and their sizes. The more and the bigger the clusters are, the lower will be the sum of all pageranks since only inter-cluster edges are accounted for in our version of the PageRank algorithm. Then if we simply make the difference between the real and the inter-cluster pagerank, almost every will be demoted. Thus for a page to be considered demoted, its particular demotion should be greater than the average one observed on the whole graph otherwise it would be considered promoted.

We can see that proportions of each set is mostly respected

	20%				30%			
	Total		Intersection		Total		Intersection	
	Nombre	Score	Nombre	Score	Nombre	Score	Nombre	Score
PageRank	25	180 934	15	102 508	52	265 760	42	214 739
Tech1	29	129 179	15	81 326.8	63	191 141	42	152 871
PageRank	25	180 934	10	82 071.1	52	265 760	21	127 917
Tech2	12	176 016	10	145 938	27	251 537	21	210 996
PageRank	25	180 934	18	125 471	52	265 76O	38	191 167
Tech3	24	166 107	18	137 336	51	244 962	38	205 899
Pagerank	25	180 934	13	98 269.6	52	265 760	32	169 028
Tech4	24	129 895	13	83 338.5	52	190 841	32	142 621

Table V
EFFECTS OF DIFFERENTS TESTS ON SPAM LINKED PAGES

using whatever technique expect for the *spammed* set. Tech1 slightly reduces this set's importance, Tech3 and Tech4 do not really alter the mass of the *spammed* pages but Tech2 increases the importance of those pages by more than 60%. Those results are not precise enough to draw some conclusions and we need to look at each set in depth.

Let's take a closer look on the results. We focus on nodes with a proportionately high pagerank (nodes well ranked). We will concentrate our analysis on the first 20% (resp. 30%) of each set, meaning nodes representing 20% (resp. 30%) of the set pagerank. Tabs. II, III, IV and V represent for each technique the number of nodes and score for the whole 20 (resp 30) top percent of each set and the number of nodes and score for the intersection with the 20 (resp 30) top percent of the pagerank of the set. We want to ensure that the demotion observed at the whole graph level is not uniformly distributed amongst all nodes.

Tab. II presents the results for the *spam* set. Regarding the top 20% of this set we can see that for Tech1, Tech3 and Tech4 there are more nodes in this top 20 than for the PageRank meaning that each node is weaker. In the case of Tech4 it more than doubles the number of nodes in the top 20%. There are fewer nodes for the Tech2.

Looking at the intersection of each set we can observe that Tech1 demotes the intersection's pagerank by less than 17% which is less than the general demotion registered for this method. Tech2 demotes the pagerank by more than 51% which is better than the general reduction meaning that this spam is actually demoted. Tech3 performs a demotion of almost 15.5% on the intersection which is also better than the general demotion on the whole graph. Tech4 reduces the spam pages in the intersection's pagerank by more than 47% which is way more than the average demotion for this technique. Spam pages are hence strongly demoted by this technique.

On the top 30% of the *spam* set, Tech2 improves its results up to a 53% demotion more than 10 points above the average demotion. Tech1 results worsen to almost 13% and Tech3 efficiency falls to the average demotion. Tech4 demotion remains almost the same at just less than 46%. This demotion is again almost 20% higher than the average one proving the efficiency of this approach.

Now let us focus on the intersections for the 30 top percent. It is interesting for us to have big enough intersection in this case to be sure that strong demoted *spam* nodes are not replaced by stronger promoted *spam* nodes. The size of all intersections combined with the fraction of the set's pagerank they represent allow us to be sure that it is the case.

We can see in this table that only the first method fails at demoting *spam* pages. Indeed it has at tendency at increasing the pagerank of already highly ranked *spam* pages.

Tab. III shows the results for the *nonspam* set. It is important here to confirm the good results obtained by Tech2, Tech3 and Tech4.

We first study the results of Tech1. We observe that it has more pages in both top 20 and top 30 percent of the *nonspam* set. This shows that those pages are weaker and hence demoted. For the top 20% (resp. 30%), Tech1 demotes the intersection by 14.7% (resp. 16%) which represents a small promotion compared to the general demotion. The score on the top 30% is actually worse than the one for the top 30% of *spam* pages.

Tech2 has the smallest number of pages composing the top 20 and 30 percent of the *nonspam* set. This means that the pages are stronger after the application of this method than before. Tech2 demotes the intersection of the top 20% (resp. 30%) by 14.8% (resp. 26.5%) which is way less than the average demotion on the whole graph. This means that these *nonspam* pages are promoted compared to the rest of the graph.

Tech3 also has a smaller number of pages than the PageRank in its top 20 and 30 percent for the *nonspam* set ensuring that those pages have a higher pagerank on average. On this particular set, Tech3 realises negatives demotions *i.e.* promotions of respectively almost 2% for the top 20% and $\sim 1,43\%$ for the top 30%. These of course are better results than those observed on the whole graph since albeit the general graph lost some pagerank, those particular pages gained some.

Finally, Tech4, while having a bigger number of nodes in its top 20% of the *nonspam* pages, demotes those pages by only 11.56% which is actually a promotion of 15% compared to the average demotion. On the top 30%, this technique continues to perform well since it promotes the intersection of the top 30% by more than 10%.

Looking at the intersections we can see that Tech1 and Tech3 have large enough intersections but that Tech2 has a smaller one compared to its intersection on the *spam* set. This is of less harm here since we are less concerned about the promotion of *nonspam* nodes but we would like to keep the same sorting as the PageRank as much as possible. The size of this intersection can be explained by the fact that at the top level the fraction of pagerank represented by the *nonspam* set after Tech2 has been applied is smaller than the one of the PageRank, meaning that some important nodes have been demoted since the number of nodes is the same. We are allowed to think then that the ranking of Tech2 may preserve an important part of the PageRank ranking on the *nonspam* set.

Tech3 outperforms Tech2 and Tech4 on this table but it was the contrary on the *spam* table. It is of interest to see how they can be ranked and if the *undecided* and *spammed* sets can be helpful to do that.

Tab. IV concerns the *undecided* set. This set is the one that contains the least relevant information since pages contained in this set were not clearly identified as either *spam* or *nonspam* pages.

Our first method continues to produce the same effect previously seen on the first two sets. Meaning the demotion observed on the top 20% (resp. 30%) is $\sim$ 12% (resp. 13.8%), being inferior to the average demotion. We can then conclude that this technique slightly increase the pagerank of already highly ranked pages and demotes poorly ranked ones.

Tech2 promotes the top 20% intersection of the *undecided* set by more than 3% but if we consider the top 30% there actually is a demotion of 20% which is less than the average observed on the whole graph.

Tech3 practically does not touch to the pagerank of *undecided* nodes. The registered demotions for the top 20 and 30 percent are respectively of 1% and 2%. These results are again way above the average results of this method.

Tech4 slightly demotes the pagerank of the top 20 and 30% but less than the ones in the *nonspam* set. It means that those pages get a better promotion than the *nonspam* ones.

The results obtained by Tech2, Tech3 and Tech4 could be explained by the fact that these sites are borderline. it means that some may be *spam* while other are *nonspam* nodes. Thus, some nodes use the techniques tracked by our methods while others don't. This is clearly visible for Tech2 where we have a promotion on the top 20% but a demotion on the top 30%. Moreover we can see that the number of pages almost triple between the top 20 and 30 percent meaning that there is a gap in pagerank.

	Subset of *spammed*	
	Value	%
PageRank	18 645.802	100
Tech1	14 867.013	79,73
Tech2	10 269.059	55.08
Tech3	19 705.154	105.69
Tech4	12 963.077	69.53

Table VI
PAGERANK RESULTS FOR THE SPAMMED SUBSET OF NODES CONNECTED TO SPAM.

	Demotion		Promotion		Total
	O	E	O	E	
nonspam	458	488.63	572	541.37	1030
spam	98	67.37	44	74.63	142
Total	556		616		1172

Table VII
VALUES O(BTAINED) AND E(XPECTED) FOR TECH2

Tab. V deals with the *spammed* set. Since those pages where linked by the *spam* set, it is our assumption that they benefit from the Webspam and hence should be demoted by our techniques.

Looking at the table, we see that our techniques behave on this set like on the *nonspam* one. Indeed, on this set Tech1 is the only one that demotes the pagerank of those pages. Tech2 and Tech3 realizes absolute promotions on the top of those pages while Tech4 relatively promotes them. This is not the attended results. It is possible that *spam* pages make some outlinks to genuine strong pages. In that case it is natural that our techniques favor them. We want to be sure that those promoted pages are only connected to the *spam* set one-way.

In order to confirm our impression, let us focus on a subset of the *spammed* pages. We will concentrate on pages linked from the *spam* set that link back. Results are presented in Tab. VI.

We can see that on those particular nodes, except for Tech3, that promote those pages, all other techniques demote those pages tightly connected to the *spam* set. We will further discuss about the *spammed* set in section VI.

Analysing the effects of our 4 approaches on the *spam*, *nonspam*, *undecided* and *spammed* sets made us realise that Tech1 is not helpful but that Tech2, Tech3 and Tech4 succeeded in demoting the effects of Webspam while promoting honest pages. Tech4 outperforms both Tech2 and Tech3 concerning the demotion of Webspam and *vice versa* regarding *nonspam* promotion. In the next section we will use statistical tools to check whether these techniques are significantly efficient.

	Demotion		Promotion		Total
	O	E	O	E	
nonspam	86	86.16	1942	1941.84	2028
spam	7	6.84	154	154.16	161
Total	93		2096		2189

Table VIII
VALUES O(BTAINED) AND E(XPECTED) FOR TECH3

	Demotion		Promotion		Total
	O	E	O	E	
nonspam	174	179.26	1 207	1 201.74	1 381
spam	22	16.74	107	112.26	129
Total	196		1 314		1 510

Table IX
VALUES O(BTAINED) AND E(XPECTED) FOR TECH4

V. STATISTICAL TEST

In this section, we are looking for statistical evidence of the efficiency of our methods to ensure that they are more than just working heuristics. We saw that Tech2, Tech3 and Tech4 have different effects on pages based on their set of origin. We want to make sure that it is not just a huge coincidence but that it is in fact our methods that effectively help to separate Web pages. We will use a χ^2 independence test to verify that fact.

Here we are only interested in pages with high pagerank before and after the computation of one of our method on the graph. We want to see how these pages are treated by our last three techniques. We make two categories, pages that are demoted (meaning their particular demotion is greater than the average one) and pages that are promoted (their particular demotion is either negative or less or equal to the average one). Since we are only interested in pages with high pageranks, our sample for each set will be the top 30%.

The hypothesis $\mathcal{H}_0$ we want to test is that both *spam* and *nonspam* pages share the same distribution.

The values V_{ij} for each set and each category can be found in Tab. VII for Tech2, Tab. VIII for Tech3 and Tab. IX for Tech4. All categories fill the minimum requirements for the χ^2 test. These tables also show the expected values calculated with the following formula:

$$E_{ij} = \frac{S_{i*} * S_{*j}}{S_{**}}$$

where S_{i*} is the sum of the i^{th} line S_{*j} the sum of the j^{th} column and S_{**} the sum over the lines and columns.

Finally the χ^2 value for all tests can be found in Tabs X for Tech2 and XI for Tech4. Since this χ^2 test is made over 2 categories and 2 sets of values, the critical value to exceed is 3.84 if we want to reject the hypothesis $\mathcal{H}_0$ with a probability of error of 5%. The χ^2 value is calculated according to the following formula:

$$\chi^2 = \sum_{i,j} \chi^2_{ij} \qquad \text{where} \qquad \chi^2_{ij} = \frac{(V_{ij} - E_{ij})^2}{E_{ij}}$$

The χ^2 value obtained for Tech2 is **30.16** meaning that we can reject the hypothesis $\mathcal{H}_0$ with at most 0.5% chances to be wrong. Thus it can be stated that *spam* and *nonspam* pages do not share the same distribution in this case *i.e* Tech2 effectively separates *spam* pages from *nonspam* ones.

Looking at the above formula and the values in Tab. VIII, the score for Tech3 is easy to compute and leaves no doubt, **0** meaning that the two samples share the same distribution. As well as it seems to work in practice, there is no statistical evidence that Tech3 may able to tell the difference between *spam* pages and *nonspam* ones.

The χ^2 value for Tech4 is **2.07**. We can reject $\mathcal{H}_0$ with a probability of being wrong of 15%. It is only a weak evidence that Tech4 effectively makes the difference between *spam* an *nonspam* pages. Still it is promising for Tech4 with just a bit of refining the technique we should be able to obtain the statistical evidence we look for.

We were only able to show strong statistical evidence for the good behaviour of Tech2, leaving us with Tech3 just as an heuristic that seems to work and Tech4 that looks promising but probably requires a better tuning of its parameters since the χ^2 test was not conclusive enough.

	Demotion	Promotion	Total
nonspam	1.92	1.73	3.65
spam	13.93	12.57	26.51
Total	15.85	14.31	**30.16**

Table X
χ^2 VALUES FOR TECH2

	Demotion	Promotion	Total
nonspam	0.15	0.02	0.17
spam	1.65	0.25	1.9
Total	1.8	0.27	**2.07**

Table XI
χ^2 VALUES FOR TECH4

VI. COMPARAISON & DISCUSSION

In this section we will compare the results of our approach to other detection approaches on the same dataset. It is in fact a popular dataset that was used for the Web spam challenge 2008[5]. All the participants in this challenge used machine learning based techniques and their objective is to identify Webspam pages. We cannot directly compare our approach to those in this challenge since we have different objectives, theirs is to detect Webspam while ours is to minimize its influence.

[5]http://webspam.lip6.fr

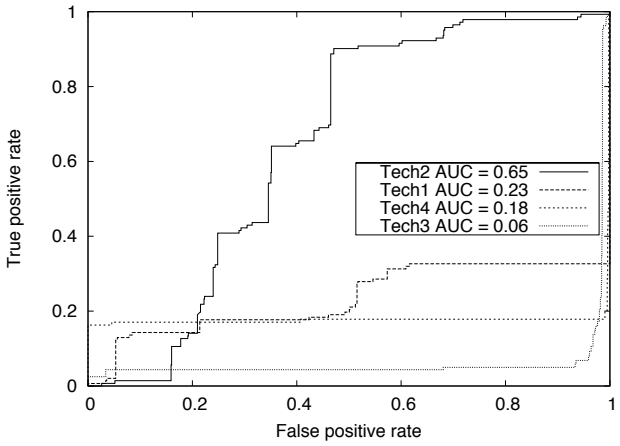


Figure 2. ROC curves for all four techniques

In order to compare our methods to those presented at the challenge, we will assume that their detection capabilities are uniformly distributed regarding the pagerank of said pages. All methods are classifiers that give a score in $[0, 1]$ to all pages. Any page whose score is strictly higher than 0.5 is considered spam while any page receiving a score strictly lower than 0.5 is considered nonspam. All pages with a score equal to 0.5 are not considered for the evaluation.

The only information we possess about our pages is the value of their demotion or promotion by a particular technique. For each page i we define the relative effect of technique k as $Re_k(i)$ s.a

$$\begin{array}{ll} Re_k(i) \in [-d, p] & \\ Re_k(i) < 0 & \text{Page i was demoted} \\ Re_k(i) > 0 & \text{Page i was promoted} \end{array}$$

where $-d$ is the maximum relative demotion, p is the maximum relative promotion. Then the classification score given by Tech$k$ to page i is,

$$Cs_k(i) = -\frac{Re_k(i) - m}{2 \cdot m}$$

where $m = \max(d, p)$. This value fulfills all the requirements for classification, $Cs_k(i) \in [0, 1], \forall i$, $Cs_k(i) > 0.5$ (demotion) page i is called spam and $Cs_k(i) < 0.5$ (promotion) page i is called nonspam.

The metric use to rank the different methods is the Area Under the ROC Curve (AUC). This quantity is equal to the probability that a spam page will receive a higher score by the classifier than a nonspam page. Fig. 2 presents the ROC curves for all techniques together with their AUC. We can see on this figure that the only technique that behave well according to the AUC is Tech2 with a score of 0.65. This is far from the score of the 6 contestants of the Web challenge 2008 whose results[6] are between 0.73 and 0.85. But one should remember that those techniques use machine learning and were specifically trained on this set while the parameters in our techniques were not tuned. We cannot hope to achieve the same quality of results using only heuristics, that is why the results obtained by Tech2 are satisfying and prove the interest of lightweight approaches for Webspam demotion.

In this section we also discuss the results our techniques obtained on the *spammed* set. This set should contain pages possessed by spammers since it is unlikely that a spammer will make an outlink to a page that is not his. The difference of behavior may be caused by the fact that spammers also possess genuine pages and use mixed techniques to increase their target page pagerank.

Since we possess statistical evidence for the good behavior of Tech2 we should be able to use it as a reference regarding the nature of the pages in this set. It seems from the results presented in Tab. V that the top pages of this set are genuine and use genuine techniques to obtain their high pagerank since Tech2 promotes those pages. While focusing on the pages that link back to the *spam* set, Tech2 demotes them. Thus its behavior is coherent and we misjudged the content of *spammed* set. Indeed it appears it contains many genuine pages despite the way it was constructed.

VII. Conclusion

In this paper we have presented different clustering methods for the demotion of the effects of Webspam on the PageRank algorithm. All four approximate methods are fast to compute and need only a small amount of memory. Tech2, Tech3 and Tech4 based respectively on the identification of small circuits in the graph and random walks, are shown to have good results on Webspam demotion.

Moreover, for the method Tech2, we have strong statistical evidence that it can separate spam and nonspam nodes. The complexity of this method is $\mathcal{O}(n)$. Thus this fully automatic method could be effectively added to the already existing arsenal for the Webspam detection and demotion of a search engine. When compared to the best detection methods on the same dataset, Tech2 performs a little worse but does not require any learning phase thus having really encouraging results. It is still of interest to investigate other methods to perform approximate clustering on the Web graph.

Since Tech3 and Tech4 show promising results, it is interesting to look further on how the tuning of their parameter may influence the obtained results. In particular, the length of the random walk should be analyzed in more depth together with the interest of having random walks with different length during the computation.

References

[1] Jacob Abernethy, Olivier Chapelle, and Carlos Castillo. Web spam identification through content and hyperlinks. In *Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, AIRWeb '08, pages 41–44, New York, NY, USA, 2008. ACM.

[6] http://webspam.lip6.fr/wiki/pmwiki.php?n=Main.PhaseIIIResults

[2] Reid Andersen, Christian Borgs, Jennifer Chayes, John Hopcroft, Kamal Jain, Vahab Mirrokni, and Shanghua Teng. Robust pagerank and locally computable spam detection features. In *AIRWeb '08: Proceedings of the 4th international workshop on Adversarial information retrieval on the web*, pages 69–76, New York, NY, USA, 2008. ACM.

[3] Andras A. Benczur, Karoly Csalogany, Tamas Sarlos, Mate Uher, and Máté Uher. Spamrank - fully automatic link spam detection. In *In Proceedings of the First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb*, 2005.

[4] Paolo Boldi and Sebastiano Vigna. The webgraph framework I: Compression techniques. In *In Proc. of the Thirteenth International World Wide Web Conference*, pages 595–601. ACM Press, 2003.

[5] Young-joo Chung, Masashi Toyoda, and Masaru Kitsuregawa. A study of link farm distribution and evolution using a time series of web snapshots. In *AIRWeb '09: Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pages 9–16, New York, NY, USA, 2009. ACM.

[6] C. de Kerchove, L. Ninove, and P. Van Dooren. Maximizing PageRank via outlinks. *Linear Algebra and its Applications*, 429(5-6):1254–1276, 2008.

[7] Stijn Dongen. A cluster algorithm for graphs. Technical Report 10, 2000.

[8] Michelle Girvan and M. E. J. Newman. Community structure in social and biological networks. *PROC.NATL.ACAD.SCI.USA*, 99:7821, 2002.

[9] Z. Gyöngyi and H. Garcia-Molina. Web spam taxonomy. *Adversarial Information Retrieval on the Web*, 2005.

[10] Zoltan Gyongyi, Pavel Berkhin, Hector Garcia-Molina, and Jan Pedersen. Link spam detection based on mass estimation. In *VLDB '06: Proceedings of the 32nd international conference on Very large data bases*, pages 439–450. VLDB Endowment, 2006.

[11] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating web spam with trustrank. In *VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases*, pages 576–587. VLDB Endowment, 2004.

[12] Paul Heymann, Georgia Koutrika, and Hector Garcia-Molina. Fighting spam on social web sites: A survey of approaches and future challenges. *IEEE Internet Computing*, 11(6):36–45, 2007.

[13] Vijay Krishnan and Rashmi Raj. Web Spam Detection with Anti-Trust Rank. *AIRWeb 2006 Program*, page 37, 2006.

[14] Thomas Largillier and Sylvain Peyronnet. Lightweight clustering methods for webspam demotion. In *In Proceedings of the Ninth international Conference on Web Intelligence*. IEEE Press, 2010.

[15] J. Martinez-Romo and L. Araujo. Web spam identification through language model analysis. In *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pages 21–28. ACM, 2009.

[16] Alexandros Ntoulas, Marc Najork, Mark Manasse, and Dennis Fetterly. Detecting spam web pages through content analysis. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 83–92, New York, NY, USA, 2006. ACM.

[17] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web, 1999.

[18] S. Shi, B. Lu, Y. Ma, and J.R. Wen. Nonlinear static-rank computation. In *Proceeding of the 18th ACM conference on Information and knowledge management*, pages 807–816. ACM, 2009.

[19] Baoning Wu, Vinay Goel, and Brian D. Davison. Topical trustrank: using topicality to combat web spam. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 63–72, New York, NY, USA, 2006. ACM.