

# The Computational Complexity of Link Building

Martin Olsen

MADALGO\*

Department of Computer Science  
University of Aarhus  
Aabogade 34, DK 8200 Aarhus N, Denmark  
mo@madalgo.au.dk

**Abstract.** We study the problem of adding  $k$  new links to a directed graph  $G(V, E)$  in order to maximize the minimum PageRank value for a given subset of the nodes. We show that this problem is NP-hard if  $k$  is part of the input. We present a simple and *efficient* randomized algorithm for the simple case where the objective is to compute *one* new link pointing to a given node  $t$  producing the maximum increase in the PageRank value for  $t$ . The algorithm computes an approximation of the PageRank value for  $t$  in  $G(V, E \cup \{(v, t)\})$  for all nodes  $v$  with a running time corresponding to a *small* and *constant* number of PageRank computations.

## 1 Introduction

Google uses the PageRank algorithm [3, 10] to calculate a universal measure of the popularity of the web pages. The PageRank algorithm assigns a measure of popularity to each page based on the link structure of the web graph. The PageRank algorithm – or variants of the algorithm – can be used to assign a measure of popularity to the nodes in any directed graph. As an example it can also be used to rank scientific journals and publications [2, 4, 9] based on citation graphs.

An organization controlling a set  $T$  of web pages might try to identify potential new links that would produce the maximum increase in the PageRank values for the pages in  $T$ . Subsequently the organization could try to make sure that these links were added to the web graph. If for example a link from a page  $p$  not controlled by the organization is considered beneficial then the organization could simply contact the people controlling  $p$  and offer them money for the new link<sup>1</sup>. The problem of obtaining optimal new – typically incoming –

---

\* Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation.

<sup>1</sup> The author of this paper is fully aware that Google attempts to take counter measures to paid links as can be seen on the blog of Matt Cutts ([www.mattcutts.com/blog/text-links-and-pagerank/](http://www.mattcutts.com/blog/text-links-and-pagerank/)). Matt Cutts is the head of Google's Web spam team. The subject causes much debate which justifies looking at it from a theoretical standpoint

links is known as *link building* and this problem attracts much attention from the *Search Engine Optimization* (SEO) industry. In this paper we will look at the computational complexity of link building and try to answer the following question in a formal context: How hard is it to identify optimal new links using only information of the link structure?

Langville and Meyer [8] deal with the problem of updating PageRank efficiently without starting from scratch. Avrachenkov and Litvak [1] study the effect on PageRank if a given page establishes one or more links *to* other pages. Avrachenkov and Litvak show that an optimal linking strategy for a page is to establish links only to pages in the *community* of the page. When Avrachenkov and Litvak speak about a web community they mean “... a set of Web pages that a surfer can reach from one to another in a relatively small number of steps”. It should be stressed that Avrachenkov and Litvak look for optimal links in  $\{p\} \times V$  for a given page  $p$  where  $V$  denotes the nodes in the directed graph under consideration and that they conclude that  $p$  “... cannot significantly manipulate its PageRank value by changing its outgoing links”. In this paper we will look for optimal links in  $V \times V$  and  $V \times \{p\}$  respectively which could cause a significant increase in the PageRank value of  $p$ .

### 1.1 Contribution and Outline of the Paper

We briefly introduce the mathematical background and notation for the paper in Sect. 2 and present introductory examples in Sect. 3. A general formulation of the link building problem is considered in Sect. 4 where we show that this general variant of the problem is NP-hard. In Sect. 5 we look at the simplest case of the problem where we want to find *one* new link *pointing* to a given node  $t$  producing the maximum increase for the PageRank value of  $t$ . In contrast to the intractability of the general case we present a simple randomized algorithm solving the simplest case with a time complexity corresponding to a small and constant number of PageRank computations. Results of experiments with the algorithm on artificial computer generated graphs and a crawl of the Danish part of the web graph are also reported in Sect. 5.

## 2 Mathematical Background

This section gives the mathematical background for the PageRank algorithm. We refer to [6] for more details on Finite Markov Chains in general and to [7] for more details on the PageRank algorithm. All vectors throughout this paper are column vectors.

Let  $G(V, E)$  denote a directed graph and let  $|V| = n$  and  $|E| = m$ . We allow multiple occurrences of  $(u, v) \in E$  implying a *weighted* version of the PageRank algorithm as described in [2]. A *random surfer* visits the nodes in  $V$  according to the following rules: When visiting  $u$  the surfer picks a link  $(u, v) \in E$  uniformly at random and visits  $v$ . If  $u$  is a sink<sup>2</sup> then the next node to visit is chosen

<sup>2</sup> A sink is a node not linking to any node.

uniformly at random. The sequence of pages visited by the random surfer is a Finite Markov Chain with state space  $V$  and transition probability matrix  $P = \{p_{uv}\}$  given by  $p_{uv} = \frac{m(u,v)}{\text{outdeg}(u)}$  where  $m(u,v)$  is the multiplicity of link  $(u,v)$  in  $E$  and  $\text{outdeg}(u)$  is the out degree of  $u$ . If  $\text{outdeg}(u) = 0$  then  $p_{uv} = \frac{1}{n}$ .

Now we modify the behavior of the random surfer so that he behaves as described above with probability  $\alpha$  when visiting  $u$  but performs a *hyper jump* with probability  $1 - \alpha$  to a node  $v$  chosen uniformly at random from  $V$ . If  $E$  is the matrix with all 1's then the transition probability matrix  $Q$  for the modified Markov Chain is given by  $Q = \frac{1-\alpha}{n}E + \alpha P$ . The powers  $w^T Q^i$  converge to the same probability distribution  $\pi^T$  for any initial probability distribution  $w$  on  $V$  as  $i$  tends to infinity – implying  $\pi^T Q = \pi^T$ . The vector  $\pi = \{\pi_v\}_{v \in V}$  is known as the PageRank vector. Computing  $w^T Q^i$  can be done in time  $O((n+m)i)$  and according to [7] 50 - 100 iterations provide a useful approximation for  $\pi$  for  $\alpha = 0.85$ . Two interpretations of  $\pi$  are the following:

- $\pi_v$  is the probability that a random surfer visits  $v$  after  $i$  steps for large  $i$ .
- All nodes perform a vote to decide which node is the most popular and  $\pi$  is the result of the vote. The identity  $\pi^T Q = \pi^T$  shows that a node is popular if it is pointed to by popular nodes.

The matrix  $I - \alpha P$  is invertible and entry  $z_{uv}$  in  $Z = (I - \alpha P)^{-1}$  is the *expected* number of visits – preceding the first hyper jump – on page  $v$  for a random surfer starting at page  $u$ . If  $u = v$  then the initial visit is also included in the count.

In this paper we will typically look at the PageRank vector for the graph we obtain if we add a set of links  $E'$  to  $G(V, E)$ . We will let  $\tilde{\pi}_v(E')$  denote the PageRank value of  $v$  in  $G(V, E \cup E')$ . The argument  $E'$  may be omitted if  $E'$  is clear from the context.

### 3 Introductory Examples

We now present two examples of link building problems involving a small graph where the nodes are organized as a hexagon connected with one link to a clique consisting of two nodes. For simplicity we only allow links with multiplicity 1 in this section. Our objective is to identify new links pointing to node 1 maximizing  $\tilde{\pi}_1$  – the PageRank value for node 1 *after* insertion of the links. In this paper we will typically try to maximize the PageRank *value* for a node as opposed to try to achieve the maximum improvement in the *ranking* of the node in which case we also have to take the values of the competitors of the node into consideration.

Figure 1a shows an optimal new link if we only look for one new link and Fig. 1b shows an optimal set of two new links. The two most popular nodes in the set  $\{3, \dots, 7\}$  prior to the modification are the nodes 6 and 7. The examples show that adding links from the most popular nodes are not necessarily the optimal solution – even in the case where the most popular nodes have a low out degree. The examples show that the topology of the network has to be taken into consideration.

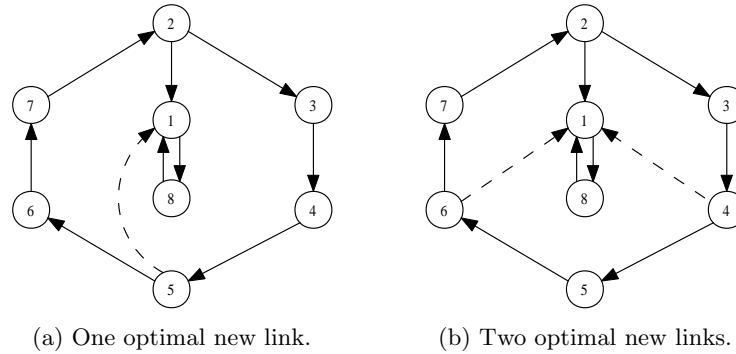


Fig. 1: The dotted links produce the maximum value of  $\tilde{\pi}_1$ . The PageRank values prior to the update are  $\pi_3 = 0.0595$ ,  $\pi_4 = 0.0693$ ,  $\pi_5 = 0.0777$ ,  $\pi_6 = 0.0848$  and  $\pi_7 = 0.0908$ .

## 4 A Result on Intractability

A natural question to ask for a set of pages  $T$  and numbers  $x$  and  $k$  is the following: “Is it possible for all the pages in  $T$  to achieve a PageRank value greater than  $x$  by adding  $k$  new links anywhere in the web graph?”. This is an informal way to phrase the decision version of the following optimization problem:

**Definition 1.** *MAX-MIN PAGERANK problem:*

- *Instance:* A directed graph  $G(V, E)$ , a subset of nodes  $T \subseteq V$  and a number  $k \in \mathbb{Z}^+$ .
- *Solution:* A set  $S \subseteq \{(u, v) \in V \times V : u \neq v\}$  with  $|S| = k$  maximizing  $\min_{t \in T} \tilde{\pi}_t(S)$ .

We allow multiple occurrences of  $(u, v)$  in  $E$  and  $S$ .

The MAX-MIN PAGERANK problem is solvable in polynomial time if  $k$  is a fixed constant in which case we can simply calculate  $\tilde{\pi}(S)$  for all possible  $S$ . If  $k$  is part of the input then the problem is NP-hard which is formally stated by the following theorem:

**Theorem 1.** *MAX-MIN PAGERANK is NP-hard.*

Theorem 1 is proved by reduction from the NP-complete balanced version of the PARTITION problem [5, page 223]. The rest of this section gives the proof in detail.

In order to prove that MAX-MIN PAGERANK is NP-hard when  $k$  is part of the input we need three lemmas concerning the graph in Fig. 2 where the weight of a link is the number of occurrences in  $E$ . The intuition behind the lemmas and the proof is the following: The nodes  $A$  and  $B$  are identical twins devoted to each other – the number of links  $x$  between them is big – and they share the

same view on the world by assigning the same weight  $w_i$  to any other node  $i$  in the network. Suppose that you would like to maximize  $\min(\tilde{\pi}_A, \tilde{\pi}_B)$  with  $n$  new links. The best you can do is to add one new link from every node in  $\{1, \dots, n\}$  to either  $A$  or  $B$  such that  $\tilde{\pi}_A = \tilde{\pi}_B$ . It turns out that we have to split the friends of  $A$  and  $B$  in two groups of equal cardinality and weight to achieve  $\tilde{\pi}_A = \tilde{\pi}_B$  and let one group link to  $A$  and the other group link to  $B$ . Splitting the friends is a well known NP-complete problem [5, page 223].

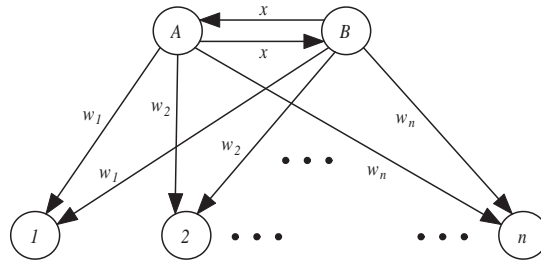


Fig. 2: A directed graph with weights indicating the number of occurrences of the links.

In the following we let  $N = \{1, \dots, n\}$  and  $W = \sum_{i=1}^n w_i$ . We will write  $\tilde{\pi}_{AB}(E')$  as a shorthand for  $\tilde{\pi}_A(E') + \tilde{\pi}_B(E')$ . We will now formally introduce the term *sum-optimal* and justify this definition in the two subsequent lemmas.

**Definition 2.** A set of links  $E'$  is called *sum-optimal* if

$$\forall i \in N : (i, A) \in E' \vee (i, B) \in E' .$$

In Lemma 1 we show that we achieve the same value for  $\tilde{\pi}_A + \tilde{\pi}_B$  for all sum-optimal sets of  $n$  links. In Lemma 2 we show that we will achieve a lower value of  $\tilde{\pi}_A + \tilde{\pi}_B$  for any other set of links.

In Lemma 3 we show that we can achieve  $\tilde{\pi}_A = \tilde{\pi}_B$  for a sum-optimal set of  $n$  links *if and only if* we can split the friends of  $A$  and  $B$  in two groups of equal cardinality and weight. The three lemmas show that we can identify such a potential split by maximizing  $\min(\tilde{\pi}_A, \tilde{\pi}_B)$ .

**Lemma 1.** Consider the graph in Fig. 2. If  $E'_1$  and  $E'_2$  denote two arbitrary sum-optimal sets of  $n$  links then we have the following:

$$\tilde{\pi}_{AB}(E'_1) = \tilde{\pi}_{AB}(E'_2) . \tag{1}$$

*Proof.* Let  $E'$  be an arbitrary sum-optimal set of  $n$  links. The only nodes that link to the nodes in  $N$  are  $A$  and  $B$  and  $A$  and  $B$  both use a fraction of  $\frac{W}{W+x}$  of their links on  $N$ . Since no node in  $N$  is a sink and the sum of PageRank values of the nodes in  $N$  is  $1 - \tilde{\pi}_{AB}(E')$  we have the following:

$$1 - \tilde{\pi}_{AB}(E') = (1 - \alpha) \frac{n}{n+2} + \alpha \tilde{\pi}_{AB}(E') \frac{W}{W+x} . \quad (2)$$

From (2) we obtain an expression for  $\tilde{\pi}_{AB}(E')$  that proves (1):

$$\tilde{\pi}_{AB}(E') = \frac{1 - (1 - \alpha) \frac{n}{n+2}}{1 + \alpha \frac{W}{W+x}} .$$

□

**Lemma 2.** *Let  $x$  satisfy the following inequality:*

$$x > \frac{W(n+2)^2}{n(1-\alpha)} - W . \quad (3)$$

*If  $E'$  is an arbitrary sum-optimal set of  $n$  links and  $L$  is an arbitrary set of links which is not sum-optimal then we have that*

$$\tilde{\pi}_{AB}(E') > \tilde{\pi}_{AB}(L) . \quad (4)$$

*Proof.* There has to be at least one node  $u \in N$  that does not link to  $A$  and does not link to  $B$  since  $L$  is not sum-optimal. A fraction of  $1 - \alpha$  of the PageRank value of  $u$  is spread uniformly on all nodes. No matter whether  $u$  is a sink or not then it will spread at least a fraction  $\frac{n}{n+2}$  of the remaining part of its PageRank value to the other nodes in  $N$ . The PageRank value of  $u$  is greater than  $\frac{1-\alpha}{n+2}$  which enables us to establish the following inequality:

$$1 - \tilde{\pi}_{AB}(L) > (1 - \alpha) \frac{n}{n+2} + \alpha \frac{1 - \alpha}{n+2} \cdot \frac{n}{n+2} . \quad (5)$$

From (3) we get  $\frac{(1-\alpha)n}{(n+2)^2} > \frac{W}{W+x}$ . Now we use (2), (5) and  $\tilde{\pi}_{AB}(E') < 1$  to conclude that  $1 - \tilde{\pi}_{AB}(L) > 1 - \tilde{\pi}_{AB}(E')$  that proves (4). □

**Lemma 3.** *Let  $E'$  denote an arbitrary sum-optimal set of  $n$  links and let  $x$  satisfy*

$$x > \frac{\alpha W(n+2)}{1-\alpha} - W . \quad (6)$$

*Let  $A_{\leftarrow} = \{i \in N : (i, A) \in E'\}$ . The set  $A_{\leftarrow}$  consists of the nodes in  $N$  that link to  $A$ . We define  $W_{A_{\leftarrow}} = \sum_{i \in A_{\leftarrow}} w_i$ . We also define  $B_{\leftarrow}$  and  $W_{B_{\leftarrow}}$  accordingly.*

*The following two statements are equivalent where  $E'$  is omitted as an argument for  $\tilde{\pi}_A$  and  $\tilde{\pi}_B$ :*

1.  $W_{A_{\leftarrow}} = W_{B_{\leftarrow}} \wedge |A_{\leftarrow}| = |B_{\leftarrow}|$  .
2.  $\tilde{\pi}_A = \tilde{\pi}_B$  .

*Proof.* Let  $\tilde{\pi}_{A_{\leftarrow}}$  and  $\tilde{\pi}_{B_{\leftarrow}}$  denote the sum of PageRank values for the two sets  $A_{\leftarrow}$  and  $B_{\leftarrow}$  respectively. Following the same line of reasoning as used in the proof of Lemma 1 we have the following:

$$\tilde{\pi}_A = \frac{1-\alpha}{n+2} + \alpha\tilde{\pi}_{A_{\leftarrow}} + \alpha\frac{x}{x+W}\tilde{\pi}_B \quad (7)$$

$$\tilde{\pi}_B = \frac{1-\alpha}{n+2} + \alpha\tilde{\pi}_{B_{\leftarrow}} + \alpha\frac{x}{x+W}\tilde{\pi}_A \quad (8)$$

$$\tilde{\pi}_{A_{\leftarrow}} = |A_{\leftarrow}| \frac{1-\alpha}{n+2} + \alpha \frac{W_{A_{\leftarrow}}}{W+x} (\tilde{\pi}_A + \tilde{\pi}_B) \quad (9)$$

$$\tilde{\pi}_{B_{\leftarrow}} = |B_{\leftarrow}| \frac{1-\alpha}{n+2} + \alpha \frac{W_{B_{\leftarrow}}}{W+x} (\tilde{\pi}_A + \tilde{\pi}_B) . \quad (10)$$

1  $\Rightarrow$  2: Assume that  $W_{A_{\leftarrow}} = W_{B_{\leftarrow}}$  and  $|A_{\leftarrow}| = |B_{\leftarrow}|$  for a sum-optimal set  $E'$  consisting of  $n$  links. By using (9) and (10) we conclude that  $\tilde{\pi}_{A_{\leftarrow}} = \tilde{\pi}_{B_{\leftarrow}}$ . By solving (7) and (8) we get that  $\tilde{\pi}_A = \tilde{\pi}_B$ .

2  $\Rightarrow$  1: Assume that  $\tilde{\pi}_A = \tilde{\pi}_B$  for a sum-optimal set  $E'$  of  $n$  links. In this case we can conclude that  $\tilde{\pi}_{A_{\leftarrow}} = \tilde{\pi}_{B_{\leftarrow}}$  by using (7) and (8). If  $x > \frac{\alpha W(n+2)}{1-\alpha} - W$  then  $\frac{1-\alpha}{n+2} > \alpha \frac{W}{W+x}$ . This means that the last term in (9) and (10) are smaller than  $\frac{1-\alpha}{n+2}$ . We conclude that  $|A_{\leftarrow}| = |B_{\leftarrow}|$  with  $W_{A_{\leftarrow}} = W_{B_{\leftarrow}}$  as a consequence.  $\square$

We are now in a position to prove Theorem 1.

*Proof.* We show how to solve an instance of the balanced version of the PARTITION problem [5, page 223] – which is known to be NP-complete – in polynomial time if we are allowed to consult an *oracle*<sup>3</sup> for solutions to the MAX-MIN PAGERANK problem.

For an instance of the balanced version of PARTITION we have a  $w_i \in \mathbb{Z}^+$  for each  $i \in N$ . The question is whether a subset  $N' \subset N$  exists such that  $\sum_{i \in N'} w_i = \sum_{i \in N-N'} w_i$  and  $|N'| = |N - N'|$ .

In polynomial time we transform this instance into an instance of MAX-MIN PAGERANK given by the graph  $G$  in Fig. 2 with  $x = \frac{W(n+2)^2}{n(1-\alpha)}$ ,  $T = \{A, B\}$  and  $k = n$ . We claim that the following two statements are equivalent:

1.  $N' \subset N$  exists such that  $\sum_{i \in N'} w_i = \sum_{i \in N-N'} w_i$  and  $|N'| = |N - N'|$ .
2. The solution  $S$  to the MAX-MIN PAGERANK instance is a sum-optimal set of links with  $W_{A_{\leftarrow}} = W_{B_{\leftarrow}}$  and  $|A_{\leftarrow}| = |B_{\leftarrow}|$ .

1  $\Rightarrow$  2: Let  $E' = [N' \times \{A\}] \cup [(N - N') \times \{B\}]$ . According to Lemma 1 and Lemma 2 then  $\tilde{\pi}_{AB}(E')$  is at its maximum compared to any other set of  $n$  new links. According to Lemma 3 we also have that  $\tilde{\pi}_A(E') = \tilde{\pi}_B(E')$ . This means that  $\min(\tilde{\pi}_A(E'), \tilde{\pi}_B(E'))$  is at its maximum. The solution  $S$  to the MAX-MIN PAGERANK instance must match this value so  $S$  must be sum-optimal

<sup>3</sup> An oracle is a hypothetical computing device that can compute a solution in a *single step* of computation.

(Lemma 2) with  $\tilde{\pi}_A(S) = \tilde{\pi}_B(S)$ . According to Lemma 3 then  $W_{A_{\leftarrow}} = W_{B_{\leftarrow}}$  and  $|A_{\leftarrow}| = |B_{\leftarrow}|$  for  $S$ .

2  $\Rightarrow$  1: Take  $N' = A_{\leftarrow}$ .

We can now solve the PARTITION instance by checking whether 2) is satisfied in the solution of the MAX-MIN PAGERANK instance. The checking procedure can be done in polynomial time.  $\square$

## 5 An Efficient Algorithm for the Simplest Case

We now turn to the simplest variant of the link building problem where the objective is to pick *one* link pointing to a *given page* in order to achieve the maximum increase in the PageRank value for the page. This problem can be solved naively in polynomial time using  $n$  PageRank computations. We present an efficient randomized algorithm that solves this problem with a running time corresponding to a *small* and *constant* number of PageRank computations. The main message is that if we have the machinery capable of calculating the PageRank vector for the network we can also solve the simple link building problem.

If page  $j \neq 1$  establishes a link to page 1 then we have the following according to [1, Theorem 3.1]:

$$\tilde{\pi}_1 = \pi_1 + \pi_j \frac{\alpha z_{11} - z_{j1}}{k_j + z_{jj} - \alpha z_{1j}} . \quad (11)$$

The central idea for the link building algorithm is to avoid an expensive matrix inversion and only calculate the entries of  $Z$  playing a role in (11) for all  $j \neq 1$ . We approximate  $z_{11}$ ,  $z_{1j}$  and  $z_{j1}$  for *all*  $j \neq 1$  by performing two calculations where each calculation has a running time comparable to one PageRank computation. The diagonal elements  $z_{jj}$  are approximated by a randomized scheme tracking a random surfer. When we have obtained approximations of all *relevant* entries of  $Z$  then we can calculate (11) in constant time for any given page  $j$ .

### 5.1 Approximating Rows and Columns of $Z$

We will use the following expression for  $Z$  [6]:

$$Z = (I - \alpha P)^{-1} = \sum_{i=0}^{+\infty} (\alpha P)^i . \quad (12)$$

In order to get row 1 from  $Z$  we multiply (12) with  $e_1^T$  from the left where  $e_1$  is a vector with a 1 at coordinate 1 and 0's elsewhere:

$$e_1^T Z = \sum_{i=0}^{+\infty} e_1^T (\alpha P)^i = e_1^T + e_1^T \alpha P + (e_1^T \alpha P) \alpha P + \dots . \quad (13)$$

Equation (13) shows how to *approximate* row 1 in  $Z$  with a simple iterative scheme using the fact that each term in (13) is a row vector obtained by multiplying  $\alpha P$  with the previous term from the left. We simply track a group of

random surfers starting at page 1 and count the number of hits they produce on other pages preceding the first hyper jump.

The elements appearing in a term are non negative and the sum of the elements in the  $i$ 'th term is  $\alpha^{i-1}$  which can be shown by using the fact that  $Pe = e$  where  $e$  is the vector with all 1's so the iterative scheme converges quickly for  $\alpha = 0.85$ . The iterative scheme has roughly the same running time as the power method for calculating PageRank and 50-100 iterations gives adequate precision for approximating the fraction in (11) since  $z_{jj} \geq 1$  for all  $j$ .

By multiplying (12) with  $e_1$  from the right we will obtain an iterative scheme for calculating the first column in  $Z$  with similar arguments for the convergence.

## 5.2 Approximating the Diagonal of $Z$

Now we only have to find a way to approximate  $z_{jj}$  for  $j \neq 1$ . In order to do this we will keep track of a *single* random surfer. Each time the surfer decides *not* to follow a link the surfer changes identity and continues surfing from a new page – we chose the new page to start from by adding 1 (cyclically) to the previous start page. For each page  $p$  we record the identity of the surfer who made the most recent visit, the total number of visits to  $p$  and the number of different surfers who have visited  $p$ . The total number of visits divided by the number of different surfers will most likely be close to  $z_{pp}$  if the number of visits is large.

If  $Z_{pp}$  denotes the stochastic variable denoting the number of visits on page  $p$  for a random surfer starting at page  $p$  prior to the first hyper jump then we have the following [6]:

$$\text{Var}(Z_{pp}) = z_{pp}^2 - z_{pp} = z_{pp}(z_{pp} - 1) . \quad (14)$$

where  $\text{Var}(\cdot)$  denotes the variance. Since we will obtain the highest value of  $z_{pp}$  if all the nodes pointed to by  $p$  had only one link back to  $p$  then we have that

$$z_{pp} \leq 1 + \alpha^2 + \alpha^4 + \dots = \frac{1}{1 - \alpha^2} . \quad (15)$$

Combining (14) and (15) we have that  $\text{Var}(Z_{pp}) = O(1)$  so according to *The Central Limit Theorem* we roughly need a *constant* number of visits per node of the random surfer to achieve a certain level of certainty of our approximation of  $z_{pp}$ .

Our main interest is to calculate  $z_{pp}$  for pages with high values of  $\pi_p$  – luckily  $k\pi_p$  is the expected number of visits to page  $p$  if the random surfer visits  $k$  pages for large  $k$  [6] so our approximation of  $z_{pp}$  tends to be more precise for pages with high values of  $\pi_p$ . We also note that it is easy to parallelize the algorithm described above simply by tracking several random surfers in parallel.

## 5.3 Experiments

Experiments with the algorithm were carried out on artificial computer generated graphs and on a crawl of the Danish part of the web graph. Running the

algorithm on a subgraph of the web graph might seem to be a bad idea but if the subgraph is a community it actually makes sense. In this case we are trying to find optimal link modifications only involving our direct competitors. Locating the community in question by cutting away irrelevant nodes seems to be a reasonable preprocessing step for the algorithm.

**Experiments on Artificial Graphs** The algorithm was tested on 10 computer generated graphs each with 500 nodes numbered from 1 to 500 and 5000 links with multiplicity 1 inserted totally at random. For each graph  $G(V, E)$  and for each  $v \in V$  such that  $(v, 1) \notin E$  we computed  $\tilde{\pi}_1(\{(v, 1)\})$ . The new PageRank value  $\tilde{\pi}_1$  of node 1 was computed in two ways: 1) by the algorithm described in this section and 2) by the power method. We used 50 terms when calculating the rows and columns of the  $Z$ -matrix and 50 moves per edge for the random surfer when calculating the diagonal of  $Z$ . For the PageRank power method computation we used 50 iterations. For all graphs and all  $v$  the *relative difference* of the two values of  $\tilde{\pi}_1$  was less than 0.1%.

**Experiments on the Web Graph** Experiments were also carried out on a crawl from spring 2005 of the Danish part of the web graph with approximately 9.2 million pages and 160 millions links. For each page  $v$  in the crawl we used the algorithm to compute the new PageRank value for [www.daimi.au.dk](http://www.daimi.au.dk) – the homepage of the Department of Computer Science at University of Aarhus, Denmark – obtained after adding a link from  $v$  to [www.daimi.au.dk](http://www.daimi.au.dk). The list of potential new PageRank values was sorted in decreasing order.

The PageRank vector and the row and column of  $Z$  corresponding to [www.daimi.au.dk](http://www.daimi.au.dk) was calculated using 50 iterations/terms and the diagonal of  $Z$  was computed using 300 moves of the random surfer per edge. The computation took a few hours on standard PC's using no effort on optimization. The links were stored on a file that was read for each iteration/term in the computation of the PageRank vector and the rows and columns of  $Z$ .

As can be seen from Equation (11) then the diagonal element of  $Z$  plays an important role for a potential source with a low out degree. As an example we will look at the pages [www.kmdkv.dk/kdk.htm](http://www.kmdkv.dk/kdk.htm) and [news.sunsite.dk](http://news.sunsite.dk) which we will denote as page  $a$  and  $b$  respectively in the following. The pages  $a$  and  $b$  are ranked 22 and 23 respectively in the crawl with  $\pi_a$  only approximately 3.5% bigger than  $\pi_b$ . Page  $a$  has out degree 2 and page  $b$  has out degree 1 so based on the information on  $\pi_a$ ,  $\pi_b$  and the out degrees it would seem reasonable for [www.daimi.au.dk](http://www.daimi.au.dk) to go for a link from page  $b$  because of the difference on the out degrees. The results from the experiment show that it is a better idea to go for a link from page  $a$ : If we obtain a link to [www.daimi.au.dk](http://www.daimi.au.dk) from page  $a$  we will achieve a PageRank value approximately 32% bigger than if we obtain a link from page  $b$ . The reason is that  $z_{bb}$  is relatively big producing a relatively big denominator in the fraction in (11).

**Acknowledgments** The author of this paper would like to thank Torsten Suel and his colleagues at Polytechnic University in New York for a crawl of the Danish part of the web graph and Gerth S. Brodal from University of Aarhus for valuable comments and constructive criticism.

## References

1. Avrachenkov, K., Litvak, N.: The effect of new links on Google Pagerank. *Stochastic Models*. 22(2), 319–331 (2006)
2. Bollen, J., Rodriguez, M.A., Van de Sompel, H.: Journal Status. *Scientometrics*. 69(3), 669–687 (2006)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*. 30(1–7), 107–117 (1998)
4. Chen, P., Xie, H., Maslov, S., Redner, S.: Finding scientific gems with Google's PageRank algorithm. *Informetrics*. 1(1), 8–15 (2007)
5. Garey, M.R., Johnson, D.S.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman (1979)
6. Kemeny, J.G., Snell, J.L.: *Finite Markov Chains*. Van Nostrand (1960)
7. Langville, A.N., Meyer, C.D.: *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press (2006)
8. Langville, A.N., Meyer, C.D.: Updating Markov Chains with an Eye on Google's PageRank. *SIAM J. Matrix Anal. Appl.* 27(4), 968–987 (2006)
9. Meho, L.I., Yang, K.: *Multi-faceted Approach to Citation-based Quality Assessment for Knowledge Management*. World Library and Information Congress: 72nd IFLA General Conference and Council (2006)
10. Page, L., Brin, S., Motwani, R., Winograd, T.: *The PageRank Citation Ranking: Bringing Order to the Web*. Technical report, Stanford Digital Library Technologies Project (1998)