Department of Applied Mathematics
Faculty of EEMCS

University of Twente
The Netherlands

P.O. Box 217
7500 AE Enschede
The Netherlands
Phone: +31-53-4893400
Fax: +31-53-4893114
Email: memo@math.utwente.nl
www.math.utwente.nl/publications

Memorandum No. 1754

Monte Carlo methods in PageRank computation: When one iteration is sufficient

K. Avrachenkov[1], N. Litvak,
D. Nemirovsky [2], and N. Osipova[3].

March, 2005

ISSN 0169-2690

[1]INRIA Sophia Antipolis, France
[2]St. Petersburg State University, Russia
[3]St. Petersburg State University, Russia

Monte Carlo methods in PageRank computation: When one iteration is sufficient

K.Avrachenkov[*], N. Litvak[†], D. Nemirovsky[‡], N. Osipova[§]

Abstract

PageRank is one of the principle criteria according to which Google ranks Web pages. PageRank can be interpreted as a frequency of visiting a Web page by a random surfer and thus it reflects the popularity of a Web page. Google computes the PageRank using the power iteration method which requires about one week of intensive computations. In the present work we propose and analyze Monte Carlo type methods for the PageRank computation. There are several advantages of the probabilistic Monte Carlo methods over the deterministic power iteration method: Monte Carlo methods provide good estimation of the PageRank for relatively important pages already after one iteration; Monte Carlo methods have natural parallel implementation; and finally, Monte Carlo methods allow to perform continuous update of the PageRank as the structure of the Web changes.

Keywords: Google, PageRank, Monte Carlo methods, absorbing Markov chains
AMS Subject Classification: 65C05, 68U35, 60J22, 62F25

1 Introduction

Surfers on the Internet frequently use search engines to find pages satisfying their query. However, there are typically hundreds or thousands of relevant pages available on the Web. Thus, listing them in a proper order is a crucial and non-trivial task. The original idea of Google presented in [5] is to list pages according to their PageRank which reflects popularity of a page. The PageRank is defined in the following way. Denote by n the total number of pages on the Web and define the $n \times n$ hyperlink matrix P as follows. Suppose that page i has $k > 0$ outgoing links. Then $p_{ij} = 1/k$ if j is one of the outgoing links and $p_{ij} = 0$ otherwise. If a page does not have outgoing links, the probability is spread among all pages of the Web, namely, $p_{ij} = 1/n$. In order to make the hyperlink graph connected, it is assumed that a random surfer goes with some probability to an arbitrary Web page with the uniform distribution. Thus, the PageRank is defined as a stationary distribution of a Markov chain whose state space is the set of all Web pages, and the transition matrix is

$$\tilde{P} = cP + (1 - c)(1/n)E, \tag{1}$$

[*]INRIA Sophia Antipolis, France, e-mail: k.avrachenkov@sophia.inria.fr
[†]University of Twente, The Netherlands, e-mail: n.litvak@ewi.utwente.nl
[‡]St.Petersburg State University, Russia, e-mail: nemd@newmail.ru
[§]St.Petersburg State University, Russia, e-mail: osipovanata@mail.ru

where E is a matrix whose all entries are equal to one and $c \in (0,1)$ is the probability of not jumping to a random page (it is chosen by Google to be 0.85). The Google matrix $\tilde{P}$ is stochastic, aperiodic, and irreducible, so there exists a unique row vector π such that

$$\pi\tilde{P} = \pi, \quad \pi\underline{1} = 1, \tag{2}$$

where $\underline{1}$ is a column vector of ones. The row vector π satisfying (2) is called a PageRank vector, or simply PageRank. If a surfer follows a hyperlink with probability c and jumps to a random page with probability $1-c$, then π_i can be interpreted as a stationary probability that the surfer is at page i. The PageRank also allows several different interpretations through expectations. For instance, in [2], the PageRank is seen as the average number of surfers navigating a given page at a given time instant provided that at each time instant $t \geq 0$, a surfer can cease from navigating with probability $(1-c)$ and on average $(1-c)$ surfers start navigating from each page. This interpretation is helpful for deeper understanding of the PageRank but it is hard to use in practice because it involves the time component. The interpretation via absorbing Markov chains that we explore in the present paper, is easier and it naturally leads to simple simulation algorithms for the computation of PageRank. The end-point of a random walk that starts from a random page and can be terminated at each step with probability $1-c$, appears to be a sample from the distribution π [4, 7, 9]. Thus, after repeating the process many times, the estimate of π_j for $j = 1, \ldots, n$, is determined as the number of times when a run terminated at j, divided by the total number of runs.

In order to keep up with constant modifications of the Web structure, Google updates its PageRank at least once per month. According to publicly available information Google still uses simple Power Iteration (PI) method to compute the PageRank. Starting from the initial approximation as the uniform distribution vector $\pi^{(0)} = (1/n)\underline{1}^T$, the k-th approximation vector is calculated by

$$\pi^{(k)} = \pi^{(k-1)}\tilde{P}, \quad k \geq 1. \tag{3}$$

The method stops when the required precision ε is achieved. The number of flops needed for the method to converge is of the order $\frac{\log \varepsilon}{\log c}\, nnz(P)$, where $nnz(P)$ is the number of non-zero elements of the matrix P [13]. We note that the relative error decreases uniformly for all pages. Several proposals [8, 11, 12, 14] (see also an extensive survey paper [13]) have recently been put forward to accelerate the power iteration algorithm.

In contrast, here we study Monte Carlo (MC) type methods for the PageRank computation. To our best knowledge only in two works [3, 7] the Monte Carlo methods are applied to the PageRank computation. The principle advantages of the probabilistic Monte Carlo type methods over the deterministic methods are: the PageRank of important pages is determined with high accuracy already after the first iteration; MC methods have natural parallel implementation; and MC methods allow continuous update of the PageRank as the structure of the Web changes.

The structure and the contributions of the paper are as follows. In Section 2, we describe different Monte Carlo algorithms. In particular, we propose an algorithm that takes into account not only the information about the last visited page (as in [3, 7]), but about all visited pages during the simulation run. In Section 3, we analyze and compare the convergence of Monte Carlo algorithms in terms of confidence intervals. We show that the PageRank of relatively important pages can be determined with high accuracy even after the first iteration. In Section 4, we show that experiments with real data from the Web confirm our theoretical analysis. Finally, we summarize the results of the present work in Section 5. Technical proofs we put in the Appendix.

2 Monte Carlo algorithms

Monte Carlo algorithms are motivated by the following convenient formula that follows directly from the definition of the PageRank:

$$\pi = \frac{1-c}{n}\underline{1}^T[I-cP]^{-1} = \frac{1-c}{n}\underline{1}^T\sum_{k=0}^{\infty}c^kP^k. \tag{4}$$

This formula suggests a simple way of sampling from the PageRank distribution [4, 7, 9]. Consider a random walk $\{X_t\}_{t\geq 0}$ that starts from a randomly chosen page. Assume that at each step, the random walk terminates with probability $(1-c)$, and makes a transition according to the matrix P with probability c. It follows from (4) that the end-point of such random walk has a distribution π. Hence, one can suggest the following algorithm employed in [3].

Algorithm 1 MC end-point with random start. *Simulate N runs of the random walk $\{X_t\}_{t\geq 0}$ initiated at a randomly chosen page. Evaluate π_j as a fraction of N random walks which end at page $j = 1, \ldots, n$.*

Let $\hat{\pi}_{j,N}$ be the estimator of π_j obtained by Algorithm 1. It is straightforward that

$$\mathbb{E}(\hat{\pi}_{j,N}) = \pi_j, \qquad Var(\hat{\pi}_{j,N}) = N^{-1}\pi_j(1-\pi_j).$$

A rough estimate $Var(\hat{\pi}_{j,N}) < 1/(4N)$ given in [3] results in a conclusion that the number of samples (random walks) needed to achieve a good relative accuracy with high probability, is of the order $O(n^2)$. In the ensuing Sections 3 and 4 we will show that this complexity evaluation is quite pessimistic. The number of required samples turns out to be linear in n. Moreover, a reasonable evaluation of the PageRank for popular pages can be obtain even with $N = n$, that is, one needs only as little as one run per page!

In order to improve the estimator $\hat{\pi}$, one can think of various ways of variance reduction. For instance, denoting $Z = [I-cP]^{-1}$ and writing π_j in (4) as

$$\pi_j = \frac{1-c}{n}\sum_{i=1}^{n} z_{ij}, \quad j = 1, \ldots, n,$$

we can view π_j as a given number $(1/n)$ multiplied by a sum of conditional probabilities $p_{ij} = (1-c)z_{ij}$ that the random walk ends at j given that it started at i. Since n is known, an unnecessary randomness in experiments can be avoided by taking $N = mn$ and initiating the random walk exactly m times from each page in a cyclic fashion, rather than jumping N times to a random page. This results in the following algorithm whose version was used in [7] for computing personalized PageRank.

Algorithm 2 MC end-point with cyclic start. *Simulate $N = mn$ runs of the random walk $\{X_t\}_{t\geq 0}$ initiated at each page exactly m times. Evaluate π_j as a fraction of N random walks which end at page $j = 1, \ldots, n$.*

Let $\hat{p}_{ij}$ be a fraction of m random walks initiated at i, that ended at j. Then the estimator for π_j suggested by Algorithm 2 can be expressed as

$$\hat{\hat{\pi}}_j = \frac{1}{n}\sum_{i=1}\hat{p}_{ij}.$$

For this estimator, we have

$$\mathbb{E}(\hat{\hat{\pi}}_j) = \pi_j,$$
$$Var(\hat{\hat{\pi}}_j) = (N)^{-1}[\pi_j - n^{-1}\sum_{i=1}^{n} p_{ij}^2] < Var(\hat{\pi}_j).$$

Besides the variance reduction, the estimator $\hat{\hat{\pi}}_i$ has important advantages in implementation because picking a page at random from a huge database is not a trivial problem [10]. This difficulty is completely avoided if the pages are visited in a cyclic fashion[1]. As the only advantage of the Monte Carlo with random start, note that it does not require the number of samples N to be a multiple of n.

Another and probably more promising way of reducing the variance is to look at formula (4) from yet another angle. Note that for all $i, j = 1, \ldots, n$, the element z_{ij} of the matrix

$$Z = [I - cP]^{-1} = \sum_{k=0}^{\infty} c^k P^k \tag{5}$$

can be regarded as the average number of times that the random walk $\{X_t\}_{t\geq 0}$ visits a page j given that this random walk started at page i. Thus, we can propose an estimator based on a complete path of the random walk $\{X_t\}_{t\geq 0}$ instead of taking into account only its end-point. The complete path version of the Monte Carlo method can be described as follows.

Algorithm 3 MC complete path. *Simulate the random walk $\{X_t\}_{t\geq 0}$ exactly m times from each page. For any page i, evaluate π_j as the total number of visits to page j multiplied by $(1-c)/(n*m)$.*

The Algorithm 3 can be further improved by getting rid of artifacts in the matrix P related to pages without outgoing links (so-called dangling nodes). When a random walk reaches a dangling node, it jumps with the uniform probability to an arbitrary page. Clearly, it is more efficient just to terminate the random walk once it reaches a dangling node. Thus, we aim to rewrite (4) in terms of the original hyperlink matrix Q defined as

$$Q_{ij} = \begin{cases} 1/k, & \text{if } i \text{ has } k > 0 \text{ outgoing links,} \\ & \text{and } j \text{ is one of the links;} \\ 0, & \text{otherwise.} \end{cases}$$

Denote by $\mathcal{I}_0$ a set of dangling pages and by $\mathcal{I}_1 = \{1, \ldots, n\} \backslash \mathcal{I}_0$ a set of pages which have at least one outgoing link. For all $j = 1, \ldots, n$, it follows from (1) and (2) that

$$\pi_j = c\sum_{i=1}^{n} P_{ij}\pi_i + \frac{(1-c)}{n}\sum_{i=1}^{n}\pi_i = c\sum_{i=1}^{n} Q_{ij}\pi_i + \gamma, \tag{6}$$

where γ is the same for each j:

$$\gamma = \frac{c}{n}\sum_{i\in\mathcal{I}_0}\pi_i + \frac{(1-c)}{n} < \frac{1}{n}. \tag{7}$$

[1]When referring to MC algorithms with cyclic start, we shall use the words "cycle" and "iteration" interchangeably.

Now, we rewrite equation (6) in the matrix form

$$\pi = \pi cQ + \gamma \underline{1}^T,$$

which leads to the new expression for π:

$$\pi = \gamma \underline{1}^T [I - cQ]^{-1}. \tag{8}$$

Note that the above equation is in accordance with the original definition of PageRank presented by Brin and Page [5]. The definition via the matrix P appeared later in order to develop the Markov chain formulation of the PageRank problem. The one-to-one correspondence between (4) and (8) was noticed and proved in [2] but we find the proof presented above more insightful in our context.

Consider now a random walk $\{Y_t\}_{t\geq 0}$ which follows hyperlinks exactly as $\{X_t\}_{t\geq 0}$ except the transitions are governed by the matrix Q instead of the matrix P. Thus, the random walk $\{Y_t\}_{t\geq 0}$ can be terminated at each step either with probability $(1-c)$ or when it reaches a dangling node. For all $i, j = 1, \ldots, n$, the element w_{ij} of the matrix $W = [I - cQ]^{-1}$, is the average number of visits of $\{Y_t\}_{t\geq 0}$ to page j given that the random walk started at page i. Denote

$$w_{\cdot j} = \sum_{i=1}^{n} w_{ij}.$$

Since the coordinates of π in (8) sum up to one, we have

$$\gamma = \left[\sum_{i,j=1}^{n} w_{ij} \right]^{-1} = \left[\sum_{j=1}^{n} w_{\cdot j} \right]^{-1} \tag{9}$$

and

$$\pi_j = w_{\cdot j} \left[\sum_{j=1}^{n} w_{\cdot j} \right]^{-1}. \tag{10}$$

This calls for another version of the complete path method.

Algorithm 4 MC complete path stopping at dangling nodes. *Simulate the random walk $\{Y_t\}_{t\geq 0}$ starting exactly m times from each page. For any page j, evaluate π_j as the total number of visits to page j divided by the total number of visited pages.*

Let W_{ij} be a random variable distributed as a number of visits to page $j = 1, \ldots, n$ by the random walk $\{Y_t\}_{t\geq 0}$ given that the random walk initiated at state $i = 1, \ldots, n$. Formally,

$$\mathbb{P}(W_{ij} = x) = \mathbb{P}\left(\left[\sum_{t=0}^{\infty} \mathbf{1}_{\{Y_t = j\}} \right] = x | Y_0 = i \right), \quad x = 0, 1, \ldots,$$

where $\mathbf{1}_{\{\cdot\}}$ is the indicator function. Let $W_{ij}^{(l)}$, $l \geq 1$, be independent random variables distributed as W_{ij}. Then the estimator produced by Algorithm 4 can be written as

$$\bar{\pi}_j = \left[\sum_{l=1}^{m} \sum_{i=1}^{n} W_{ij}^{(l)} \right] \left[\sum_{l=1}^{m} \sum_{i,j=1}^{n} W_{ij}^{(l)} \right]^{-1}. \tag{11}$$

In the next section we present the analysis of this estimator.

We note that the complete path versions of the Monte Carlo methods also admit a random start. The corresponding algorithm is as follows.

Algorithm 5 MC complete path with random start. *Simulate N samples of the random walk $\{Y_t\}_{t\geq 0}$ started at a random page. For any page j, evaluate π_j as the total number of visits to page i divided by the total number of visited pages.*

One can show however that Algorithm 4 provides an estimator with a smaller variance than Algorithm 5. Indeed, let W_{Uj} be the number of visits to page j from a randomly chosen page $U \in \{1, \ldots, n\}$. Then, we have

$$Var(W_{Uj}) = \frac{1}{n}\sum_{i=1}^{n} Var(W_{ij}) + \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}^2(W_{ij}) - \left[\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}(W_{ij})\right]^2 > \frac{1}{n}\sum_{i=1}^{n} Var(W_{ij}).$$

Now note that in n simulation runs, Algorithm 4 generates one sample of the sum $\sum_{i=1}^{n} W_{ij}$, whereas Algorithm 5 generates n samples of W_{Uj}. Hence, Algorithm 4 provides random variables with smaller variance in both numerator and denominator of (11).

3 Convergence Analysis

From the preliminary analysis of the previous section, we can already conclude that MC algorithms with cyclic start are preferable to the analogous MC algorithms with random start. In the present section we thoroughly analyze and compare MC complete path stopping at dangling nodes with MC end-point. We show that under natural conditions MC complete path stopping at dangling nodes outperforms MC end-point.

We start by studying the properties of W_{ij}'s. Denote by q_{ij} the probability that starting from page i, the random walk $\{Y_t\}_{t\geq 0}$ reaches page j:

$$q_{ij} = \mathbb{P}\left(\bigcup_{t\geq 1}\{Y_t = j\} | Y_0 = i\right), \quad i, j = 1, \ldots, n.$$

Note that in this definition, $q_{jj} < 1$ is a probability to return to state j if the process started at j. It follows from the strong Markov property that W_{jj} has a geometric distribution with parameter $1 - q_{jj} \geq 1 - c$:

$$\mathbb{P}(W_{jj} = x) = q_{jj}^{x-1}(1 - q_{jj}),\ x = 1, 2, \ldots,$$

which implies

$$\mathbb{E}(W_{jj}) = \frac{1}{1 - q_{jj}}; \quad Var(W_{jj}) = \frac{q_{jj}}{(1 - q_{jj})^2};$$

Further, applying again the strong Markov property, one can show that for all $i, j = 1, \ldots, n$, W_{ij} has a shifted geometric distribution:

$$\mathbb{P}(W_{ij} = x) = \begin{cases} 1 - q_{ij}, & x = 0, \\ q_{ij}\mathbb{P}(W_{jj} = x), & x = 1, 2, \ldots. \end{cases}$$

Consequently,

$$\mathbb{E}(W_{ij}) = w_{ij} = q_{ij}\mathbb{E}(W_{jj}) = \frac{q_{ij}}{1 - q_{jj}} \tag{12}$$

and

$$Var(W_{ij}) = \frac{1 + q_{jj}}{1 - q_{jj}}\, w_{ij} - w_{ij}^2. \tag{13}$$

Now, define

$$W_{\cdot j} = \sum_{i=1}^{n} W_{ij}, \quad j = 1, \ldots, n, \quad W = \sum_{j=1}^{n} W_{\cdot j}.$$

Assuming that all W_{ij}'s are independent, we immediately obtain

$$\begin{aligned}
\mathbb{E}(W_{\cdot j}) &= \sum_{i=1}^{n} w_{ij} = w_{\cdot i}, \\
Var(W_{\cdot j}) &= \frac{1 + q_{jj}}{1 - q_{jj}}\, w_{\cdot j} - \sum_{i=1}^{n} w_{ij}^2 < \frac{1 + q_{jj}}{1 - q_{jj}}\, w_{\cdot j}, \\
\mathbb{E}(W) &= \sum_{j=1}^{n} w_{\cdot j} = \gamma^{-1}.
\end{aligned}$$

For $i, j = 1, \ldots, n$, let the empirical mean

$$\bar{W}_{ij} = \frac{1}{m} \sum_{l=1}^{m} W_{ij}^{(l)}$$

be the estimator of w_{ij}, and view

$$\bar{W}_{\cdot j} = \sum_{i=1} \bar{W}_{ij}, \quad j = 1, \ldots, n,$$

and

$$\bar{W} = \sum_{j=1} \bar{W}_{\cdot j}$$

as estimators of $w_{\cdot j}$ and γ^{-1}, respectively. The estimator (11) can be then written as

$$\bar{\pi}_j = \bar{W}_{\cdot j} \bar{W}^{-1}. \tag{14}$$

Since the second multiplier in (14) is the same for all $j = 1, \ldots, n$, the estimator $\bar{\pi}_j$ is completely determined by $\bar{W}_{\cdot j}$. The following theorem states that the relative errors of $\bar{\pi}$ and $\bar{W}_{\cdot j}$ are similar.

Theorem 1 *Given the event that the estimator $\bar{W}_{\cdot j}$ satisfies*

$$|\bar{W}_{\cdot j} - w_{\cdot j}| \leq \varepsilon w_{\cdot j}, \tag{15}$$

the event

$$|\bar{\pi}_j - \pi_j| \leq \varepsilon_{n,\beta} \pi_j$$

occurs with probability at least $1-\beta$ *for any* $\beta > 0$ *and* $\varepsilon_{n,\beta}$ *satisfying*

$$|\varepsilon - \varepsilon_{n,\beta}| < \frac{C(\beta)(1+\varepsilon)}{\sqrt{nm}}.$$

The factor $C(\beta)$ *can be approximated as*

$$C(\beta) \approx x_{1-\beta/2}\sqrt{\frac{n-n_0}{n}\,(1+c^3)}\frac{c}{1-c},$$

where $x_{1-\beta/2}$ *is a* $(1-\beta/2)$*-quantile of the standard normal distribution and* n_0 *is the number of dangling nodes.*

Proof. See the Appendix.

Theorem 1 has two important consequences. First, it states that the estimator $\bar{\pi}_j$ converges to π_j in probability when m goes to infinity. Thus, the estimator $\bar{\pi}_j$ is consistent. Second, Theorem 1 states that the error in the estimate of π_j originates mainly from estimating $w_{\cdot j}$. The additional relative error caused by estimating γ as $\left[\sum \bar{W}_{\cdot j}\right]^{-1}$, is of the order $1/\sqrt{mn}$ with arbitrarily high probability, and thus this error can essentially be neglected.

It follows from the above analysis that the quality of the estimator $\bar{\pi}_j$ as well as the complexity of the algorithm can be evaluated by the estimator $\bar{W}_{\cdot j}$. We proceed by analyzing the confidence intervals. Consider the confidence interval for $\bar{W}_{\cdot j}$ defined as

$$\mathbb{P}(|\bar{W}_{\cdot j} - w_{\cdot j}| < \varepsilon w_{\cdot j}) \geq 1-\alpha. \tag{16}$$

From (12) and (13), we have

$$\mathbb{E}(\bar{W}_{\cdot j}) = w_{\cdot j}, \quad Var(\bar{W}_{\cdot j}) \leq \frac{1}{m}\,\frac{1+q_{jj}}{1-q_{jj}} w_{\cdot j}.$$

Since $\bar{W}_{\cdot j}$ is a sum of a large number of terms, the random variable $[\bar{W}_{\cdot j} - w_{\cdot j}]/\sqrt{Var(\bar{W}_{\cdot j})}$ has approximally a standard normal distribution. Thus, from (16) we deduce

$$\varepsilon w_{\cdot j}/\sqrt{Var(\bar{W}_{\cdot j})} \geq x_{1-\alpha/2},$$

which results in

$$m \geq \frac{1+q_{jj}}{1-q_{jj}}\frac{x^2_{1-\alpha/2}}{\varepsilon^2 w_{\cdot j}}.$$

Now applying $w_{\cdot j} = \gamma^{-1}\pi_j$, we get

$$m \approx \frac{1+q_{jj}}{1-q_{jj}}\frac{\gamma x^2_{1-\alpha/2}}{\varepsilon^2 \pi_j}. \tag{17}$$

Note that $\pi_j \geq \gamma$ for all $j = 1, \ldots, n$. Thus, with a high probability, a couple of hundreds iterations allows to evaluate the PageRank of all pages with relative error at most 0.1. In practice, however, it is essential to evaluate well the PageRank of important pages in a short time. We argue that a typical user of a search engine does not check more than a dozen of

first answers to his/her query. Therefore, let us evaluate the relative error ε for a given value of π_j. Using (7), from (17) we derive

$$\varepsilon \approx x_{1-\alpha/2}\sqrt{\frac{1+q_{jj}}{1-q_{jj}}}\frac{\sqrt{1-c+c\sum_{i\in\mathcal{I}_0}\pi_i}}{\sqrt{\pi_j}\sqrt{mn}}. \tag{18}$$

Strikingly, it follows from (18) that the Monte Carlo method gives good results for important pages in one iteration only, that is, when $m = 1$. From the examples of PageRank values presented in [5], it follows that the PageRank of popular pages is at least 10^4 times greater than the PageRank of an average page. Since the PageRank value is bounded from below by $(1-c)/n$, the formula (18) implies that if the important pages have PageRank 10^4 times larger than the PageRank of the pages with the minimal PageRank value, the Monte Carlo method achieves an error of about 1% for the important pages already after the first iteration. In contrast, the power iteration method takes into account only the weighted sum of the number of incoming links after the first iteration.

Let us now compare the precision of the end-point version and the complete path version of the Monte Carlo method. According to Algorithm 1, the end-point version estimates π_j simply as a fraction of $N = mn$ random walks that ended at page j. Using standard techniques for such estimate, we construct a confidence interval

$$\mathbb{P}(|\hat{\pi}_{j,N} - \pi_{j,N}| < \varepsilon\pi_{j,N}) = 1 - \alpha.$$

Using again the standard normal distribution, we get

$$\varepsilon = x_{1-\alpha/2}\frac{\sqrt{1-\pi_j}}{\sqrt{\pi_j}\sqrt{mn}}. \tag{19}$$

Forgetting for a moment about slight corrections caused by the trade-off between random and cyclic start, we see that the choice between the end-point version and the complete-path version essentially depends on two factors: the total PageRank of dangling nodes and the probability of a cycle when a random walk started from j returns back to j. If the Web graph has many short cycles then the extra information from registering visits to every page is obtained at cost of a high extra variability which leads to a worse precision. If total rank of dangling nodes is high, the random walk will often reach dangling nodes and stop. This can have a negative impact on the complete path algorithm. The above mentioned two phenomena, if present, can make the difference between the end-point and the complete-path versions negligible. The experiments of the next section on the real data however indicate that the real Web structure is such that the complete path version is more efficient than the end-point version.

We remark that if the results of the first iteration are not satisfactory, it is hard to improve them by increasing m. After m iterations, the relative error of the Monte Carlo method will reduce on average only by the factor $1/\sqrt{m}$ whereas the error of the power iteration method decreases exponentially with m. However, because of simplicity in implementation (in particular, simplicity in parallel implementation), the Monte Carlo algorithms can be still advantageous even if a high precision is required.

Let us also evaluate a magnitude of π_j's for which a desired relative error ε is achieved. Rewriting (18), we get

$$\pi_j \approx x^2_{1-\alpha/2}\frac{1+q_{jj}}{1-q_{jj}}\frac{(1-c+c\sum_{i\in\mathcal{I}_0}\pi_i)}{\varepsilon^2 mn}. \tag{20}$$

Finally, we would like to emphasize that the Monte Carlo algorithms have natural parallel implementation and they allow to perform a continuous update of the PageRank vector. Indeed, each available processor can run an independent Monte Carlo simulation. Since the PageRank vector changes significantly during one month, Google prefers to recompute the PageRank vector starting from the uniform distribution rather than to use the PageRank vector of the previous month as the initial approximation [13]. Then, it takes about a week to compute a new PageRank vector. It is possible to update the PageRank vector using linear algebra methods [14]. However, one needs first to separate new nodes and links from the old ones. This is not necessary if one uses Monte Carlo algorithms. Specifically, we suggest to run Monte Carlo algorithms continuously while the database is updated with new data and hence to have an up-to-date estimation of the PageRank for relatively important pages with high accuracy. Then, once in a while one can run the power iteration method to have a good PageRank estimation for all pages. In particular, the continuous update should eliminate the negative reaction of users to the so-called "Google dance" [15].

4 Experiments

For our numerical experiments we have taken the Web site of INRIA Sophia Antipolis `http://www-sop.inria.fr`. It is a typical Web site with about 50000 Web pages and 200000 hyperlinks. Since the Web has a fractal structure [6], we expect that our dataset is enough representative. Accordingly, datasets of similar sizes have been extensively used in experimental studies of novel algorithms for PageRank computation [1, 13, 14]. To collect the Web graph data, we construct our own Web crawler which works with the Oracle database. The crawler consists of two parts: the first part is realized based on Java and is responsible for downloading pages from the Internet, parsing the pages and inserting their hyperlinks into the database; the second part is realized with the help of the stored procedures written in PL/SQL language and is responsible for the data management. The program allows to run several crawlers in parallel to use efficiently the network and computer resources. Since the multi-user access is already realized in Oracle database management system, it is relatively easy to organize the information collection by several crawlers and parallel implementation of Monte Carlo algorithms. We have also implemented the power iteration method and the following three Monte Carlo algorithms in PL/SQL language:

- MC complete path stopping in dangling nodes,
  MC comp path dangl nodes, for short;
- MC end-point with cyclic start,
  MC end-point cycl start, for short;
- MC complete path with random start,
  MC comp path rand start, for short.

First, we performed a sufficient number of power iterations to obtain the value of PageRank with 20 digits accuracy. We sorted the PageRank vector in the decreasing order and plotted it in the loglog scale (see Figure 1). It is interesting to observe that the PageRank vector follows very closely a power law. One can also see in Figure 2 how well the power low approximates the PageRank vector in linear scale starting from approximately the 100-th largest element.

Then, we have chosen four elements from the sorted PageRank vector:

$$\begin{aligned} \pi_1 &= 0.004093834, \\ \pi_{10} &= 0.001035867, \\ \pi_{100} &= 0.000546446, \\ \pi_{1000} &= 0.000097785. \end{aligned} \tag{21}$$

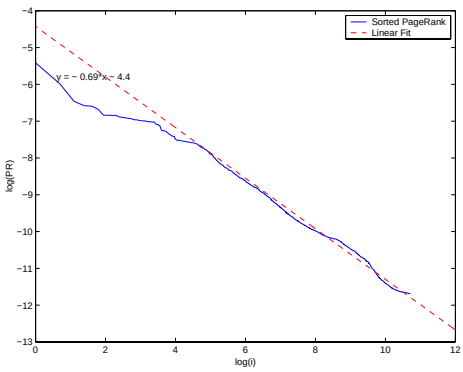

Fig. 1: Sorted PageRank, loglog scale

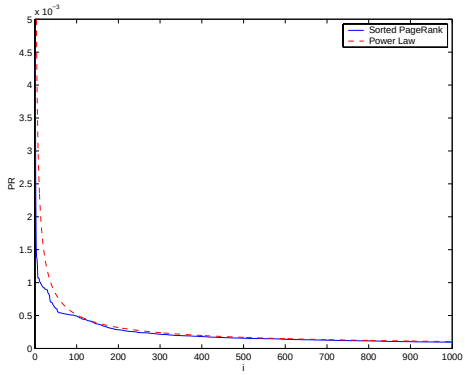

Fig. 2: Sorted PageRank, linear scale

We have performed 10 iterations of the PI method and 10 iterations of the three implemented MC algorithms. In Figures 3-6, we compare the results of 10 iterations of PI method and MC complete path stopping in dangling nodes method for the four chosen pages (21). Indeed, as predicted by formula (18), already the first iteration of MC complete path stopping in dangling nodes algorithm gives a small error for important Web pages. In fact, from Figures 3-6 one can see that MC complete path stopping in dangling nodes algorithm outperforms PI method even for the first 1000 most important pages. In Figures 3-6, we also plotted 95% confidence intervals for the MC method. As expected, there are some randomness in the convergence pattern of the Monte Carlo method and some points might fall outside of confidence intervals. However, as one can see from Figures 3-4, the PI method does not converge monotonously for the first few iterations as well.

At first sight, it looks surprising that one iteration gives a relative error of only 7% with 95% confidence for pages with high PageRank. On the other hand, such result is to be expected. Roughly speaking, we use $5 * 10^4$ independent samples in order to estimate the probability $\pi = 0.004$. A binomial random variable B with parameters $n = 5 * 10^4$, $p = 0.004$ has mean 200 and standard deviation 14.1, and thus, with a high probability, a relative error of a standard estimator $\tilde{\pi} = B/n$ will be less than 11%. The additional gain that we get in (18) is due to regular visits to every page and the usage of the complete path information.

Next, in Figures 7-10 we compare three versions of the Monte Carlo method: MC complete path stopping in dangling nodes, MC end-point with cyclic start, and MC complete path with random start. We plotted actual relative error and the estimated 95% confidence intervals. It turns out that on our dataset MC complete path stopping in dangling nodes performs the best, followed by MC complete path with random start.

MC end-point with cyclic start has the worst performance. The better performance of MC with cyclic start in respect to MC with random start was expected from the preliminary analysis of Section 2. MC is not trapped in cycles in our instance of the Web graph and the

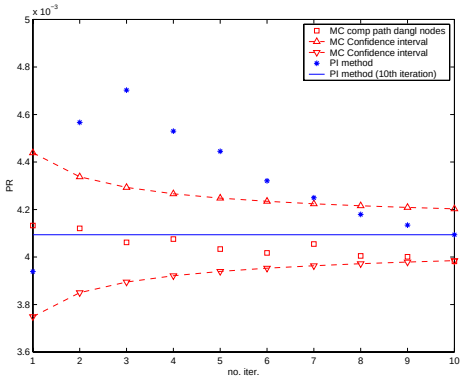

Fig. 3: PI vs. MC: π_1.

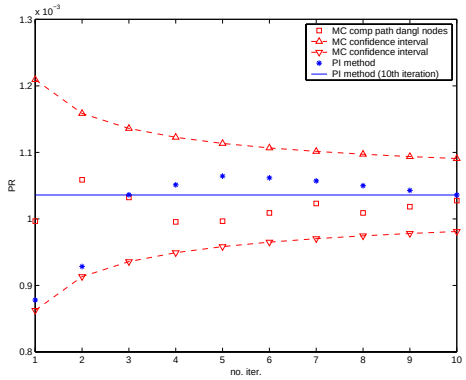

Fig. 4: PI vs. MC: π_{10}.

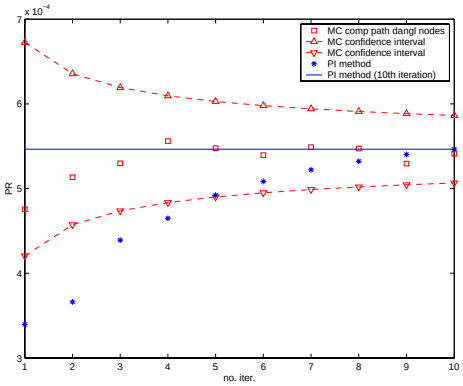

Fig. 5: PI vs. MC: π_{100}.

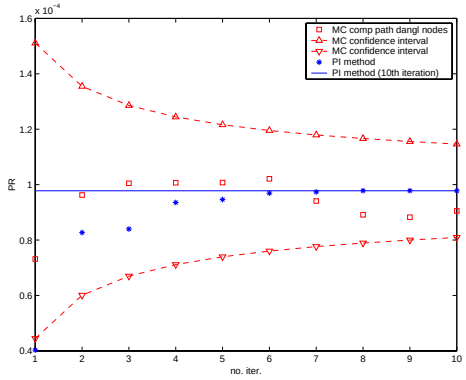

Fig. 6: PI vs. MC: π_{1000}.

total PageRank of dangling nodes is relatively small

$$\sum_{i \in \mathcal{I}_0} \pi_i = 0.23,$$

hence, we have

$$\varepsilon_{comp.path} \approx \sqrt{1 - c + c \sum_{i \in \mathcal{I}_0} \pi_i}\, \varepsilon_{end-point} \approx 0.59 \varepsilon_{end-point}.$$

To check if the presence of cycles hinder the convergence of the Monte Carlo methods, we took into account the intra-page hyperlinks. On the modified graph the Monte Carlo methods have shown a very slow convergence. It is thus fortunate for MC methods that the original definition of the PageRank excludes the intra-page hyperlinks.

5 Conclusions

We have considered several Monte Carlo algorithms. In particular, we have proposed a new Monte Carlo algorithm that takes into account not only the information about the last visited page, but about all visited pages during the simulation run. We have shown that MC algorithms with cyclic start outperform MC algorithms with random start. Our theoretical

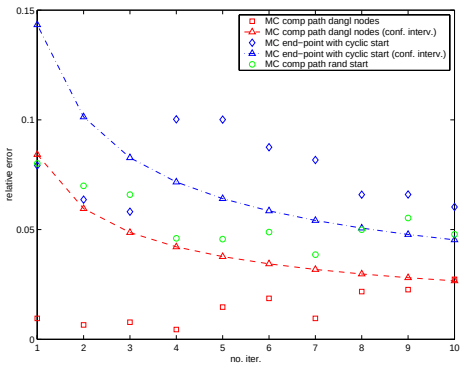

Fig. 7: MC algorithms: π_1.

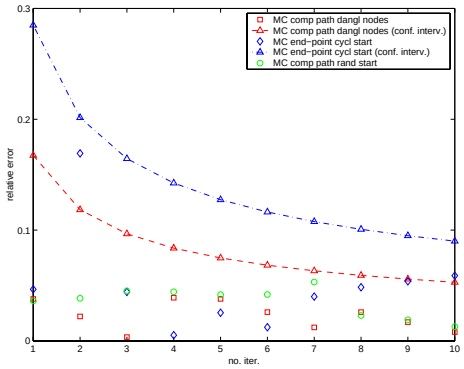

Fig. 8: MC algorithms: π_{10}.

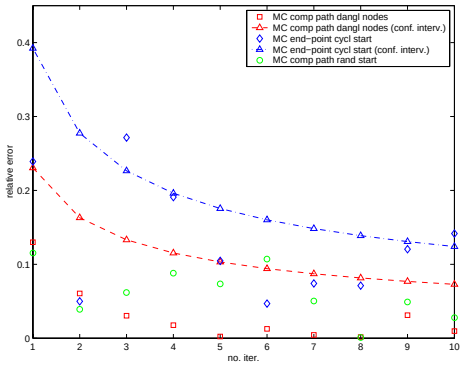

Fig. 9: MC algorithms: π_{100}.

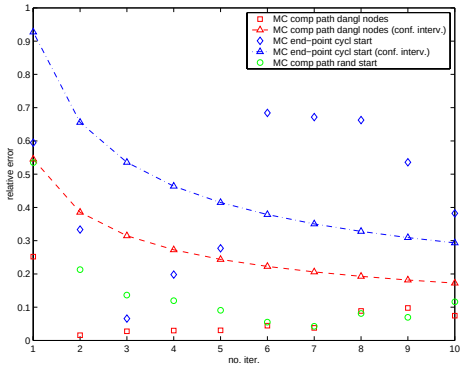

Fig. 10: MC algorithms: π_{1000}.

and experimental results have demonstrated that the Monte Carlo algorithms determine the PageRank of relatively important pages already after the first iteration. Here is a sharp contrast with the power iteration method that approximates the PageRank vector with the uniform relative error and takes into account only the weighted sum of the number of incoming links after the first iteration. The other advantages of MC algorithms are natural parallel implementation and the possibility of the continuous PageRank update while the crawler brings new data from the Web.

Appendix: The proof of Theorem 1

To prove Theorem 1 we need the following lemma.

Lemma 1 *Let $W_{i\cdot} = \sum_{j=1}^{n} W_{ij}$ be the length of the random walk $\{Y_t\}_{t\geq 0}$ initiated at page $i = 1, \ldots, n$. Then for all dangling nodes $i \in \mathcal{I}_0$, it holds $W_{i\cdot} \equiv 1$, and for non-dangling nodes $i \in \mathcal{I}_1$,*

$$\mathbb{E}(W_{i\cdot}) \leq \frac{1}{1-c}, \qquad Var(W_{i\cdot}) \leq \frac{c(1+c^3)}{(1-c)^2}. \tag{22}$$

Proof. The statement for dangling nodes is obvious. For non-dangling nodes, (22) essentially follows from the distributional identity

$$W_{i\cdot} \stackrel{d}{=} \min\{X, N_i\}, \quad i = 1, \ldots, n, \tag{23}$$

where N_i is a number of transitions needed to reach a dangling node from page i, and X has a geometric distribution with parameter $1 - c$. The mean and variance of X are given by

$$\mathbb{E}(X) = \frac{1}{1-c}; \quad Var(X) = \frac{c}{(1-c)^2}.$$

The upper bound for the expectation of $W_{i\cdot}$ follows now directly from (23). For the variance, we write

$$Var(W_{i\cdot}) = \mathbb{E}[Var(W_{i\cdot}|N_i)] + Var[\mathbb{E}(W_{i\cdot}|N_i)].$$

Conditioning on events $[N_i = k]$ and computing $Var(W_i|k)$ for $k = 1, 2, \ldots$, one can show that

$$\mathbb{E}[Var(W_{i\cdot}|N_i)] < Var(X).$$

Furthermore, we derive

$$\mathbb{E}(W_{i\cdot}|N_i) = \sum_{k=1}^{N_i} \mathbb{P}(X \geq k) = \sum_{k=1}^{N_i} c^k = \frac{c(1-c^{N_i})}{1-c},$$

and thus the variance of $\mathbb{E}(W_{i\cdot}|N_i)$ satisfies

$$Var(\mathbb{E}(W_{i\cdot}|N_i)) = c^2 Var(c^{N_i})/(1-c)^2 \leq c^4/(1-c)^2,$$

because for non-dangling nodes, the random variable c^{N_i} takes values only in the interval $[0, c]$. This completes the proof of the lemma. □

We are now ready to prove Theorem 1.

Proof of Theorem 1. Using (9) and (10), we derive

$$\begin{aligned}\bar{\pi}_j - \pi_j &= \bar{W}_{\cdot j}\bar{W}^{-1} - \pi_j \\ &= \gamma(\bar{W}_{\cdot j} - w_{\cdot j})(\gamma\bar{W})^{-1} + \left((\gamma\bar{W})^{-1} - 1\right)\pi_j.\end{aligned}$$

Given the event (15), the last equation together with (9) and (10) yields

$$|\bar{\pi}_j - \pi_j| \leq \varepsilon\pi_j + \left|(\gamma\bar{W})^{-1} - 1\right|(1+\varepsilon)\pi_j. \tag{24}$$

Let us now investigate the magnitude of the term $(\gamma\bar{W})^{-1}$. First, note that the random variables

$$\bar{W}_{i\cdot} = \sum_{j=1}^{n} \bar{W}_{ij}, \quad i \in \mathcal{I}_1,$$

are independent because they are determined by simulation runs initiated at different pages. Further, for a non-dangling node i, using Lemma 1, we find

$$\begin{aligned}\mathbb{E}(\bar{W}_{i\cdot}) &= \sum_{j=1}^{n} w_{ij}, \\ Var(\bar{W}_{i\cdot}) &= \frac{1}{m} Var(W_{i\cdot}) \leq \frac{1}{m}\,\frac{c(1+c^3)}{(1-c)^2}.\end{aligned}$$

Thus, $\bar{W}$ equals the number of dangling nodes n_0 plus the sum of $n-n_0$ independent random variables $\hat{W}_{i\cdot}$, $i \in \mathcal{I}_1$. Since the number $n-n_0$ is obviously very large, $\bar{W}$ is approximately normally distributed with mean γ^{-1} and variance

$$Var(\bar{W}) = \sum_{i\in\mathcal{I}_1} Var(\hat{W}_{i\cdot}) \leq (n-n_0)\frac{c(1+c^3)}{m(1-c)^2}.$$

Hence, $\gamma\bar{W}$ is approximately normally distributed with mean 1 and variance

$$Var(\gamma\bar{W}) \leq \gamma^2(n-n_0)\frac{c(1+c^3)}{m(1-c)^2} < \frac{n-n_0}{n^2}\,\frac{c(1+c^3)}{m(1-c)^2}, \tag{25}$$

which is a value of the order $(nm)^{-1}$. Now, let us consider a $(1-\beta)$-confidence interval defined as

$$\mathbb{P}\left(\left|(\gamma\bar{W})^{-1}-1\right| < \epsilon\right) > 1-\beta \tag{26}$$

for some small positive β and ϵ. If ϵ is small enough so that $1/(1-\epsilon) \approx 1+\epsilon$ and $1/(1+\epsilon) \approx 1-\epsilon$, then the above probability approximately equals $\mathbb{P}\left(\left|\gamma\bar{W}-1\right| < \epsilon\right)$, and because of (25), the inequality (26) holds for all ϵ satisfying

$$\epsilon \geq x_{1-\beta/2}\frac{c}{1-c}\sqrt{\frac{n-n_0}{n}\,(1+c^3)}\frac{1}{\sqrt{nm}}. \tag{27}$$

The right-hand side of (27) constitutes the additional relative error in estimating π_j. For any $\beta > 0$, this additional error can be exceeded with probability at most β. This completes the proof of the theorem. □

References

[1] S. Abiteboul, M. Preda and G. Cobena, Adaptive on-line page importance computation, in Proceedings of the 12-th International World Wide Web Conference, WWW 2003.

[2] M. Bianchini, M. Gori and F. Scarselli, Inside PageRank, Technical report, 2002. To appear in ACM Trans. Internet Technology.

[3] L.A. Breyer, Markovian page ranking distributions: some theory and simulations, Technical report, 2002, available at `http://www.lbreyer.com/preprints.html`.

[4] L.A. Breyer and G.O. Roberts, Catalytic perfect simulation, Methodol. Comput. in Appl. Probab., v.3, pp.161-177, 2001.

[5] S. Brin, L. Page, R. Motwami and T. Winograd, The PageRank citation ranking: bringing order to the Web, Stanford University Technical Report 1998, available at `http://dbpubs.stanford.edu:8090/pub/1999-66`.

[6] S. Dill, R. Kumar, K. McCurley, S. Rajagopalan, D. Sivakumar, and A. Tomkins, Self-similarity in the Web, ACM Trans. Internet Technology, v.2, no.3, pp.205-223, 2002.

[7] D. Fogaras and B. Racz, Towards scaling fully personalized PageRank, in Proceedings of the 3-rd International Workshop on Algorithms and Models for the Web-Graph, WAW 2004.

[8] I.C.F. Ipsen and S. Kirkland, Convergence Analysis of an improved PageRank algorithm. North Carolina State University Technical Report CRSC-TR04-02, 2004, available at `http://www.ncsu.edu/crsc/reports/reports04.htm`

[9] G. Jeh and J. Widom. Scaling personalized web search. In Proceedings of the 12th World Wide Web Conference, 2003.

[10] M.R. Henzinger, Algorithmic Challenges in Web Search Engines. Internet Mathematics, v.1, no.1, 2003.

[11] S.D. Kamvar, T.H. Haveliwala and G.H. Golub, Adaptive methods for the computation of PageRank, Lin. Alg. Appl., v.386, pp.51-65, 2004.

[12] S.D. Kamvar, T.H. Haveliwala, C.D. Manning and G.H. Golub, Extrapolation Methods for Accelerating PageRank Computations, In Proceedings of the 12-th International World Wide Web Conference, 2003.

[13] A.N. Langville and C.D. Meyer, Deeper Inside PageRank, Internet Mathematics, v.1, no.3, pp.335-400, 2004, also available at `http://www4.ncsu.edu/ anlangvi/`.

[14] A.N. Langville and C.D. Meyer, Updating PageRank with Iterative Aggregation. In Proceedings of the 13-th World Wide Web Conference, 2004.

[15] `http://dance.efactory.de`