Link-based algorithms

Text Based Information Retrieval (H02C8A)

Li Quan

April 28, 2011

Contents

1 Introduction

In this paper, we investigate the link analysis algorithm PageRank [5, 18]. Various aspects of this algorithm have already been studied extensively (e.g., [3, 6, 14, 15, 16]). This paper answers the following questions:

1. What is the distribution of PageRank values as the number of iterations approaches infinity? We consider the convergence properties of the power iteration to solve the eigenvector problem with respect to the need of the damping factor.
2. What is the trade-off for the damping factor? In particular, the influence on the convergence rate and variability of PageRank values is explained.
3. What are methods to personalise the PageRank values? We also discuss their advantages and disadvantages.

Notation. *Matrices and vectors are denoted upright boldface and scalars italic. Vectors are assumed to be column vectors. E.g.,* $\mathbf{A}$ *is a matrix,* $\mathbf{v}$ *a vector, and* α *a scalar. Unless stated explicitely otherwise, symbol names are used as in [1]. The PageRank vector is denoted by* $\boldsymbol{\pi}^T$.

2 Calculating PageRank

2.1 Using no damping factor

We calculate the PageRank vector for the graph of *sample-tiny.txt*, visualised in Figure 1. First we use the basic power iteration with no damping factor, where we completely preserve the hyperlink matrix $\mathbf{H}$.

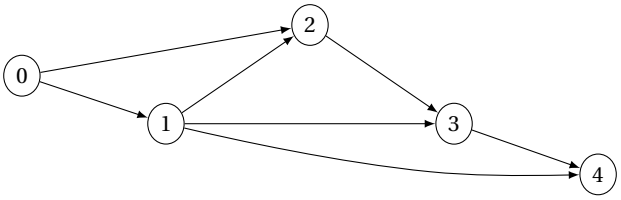


Figure 1: The web graph of *sample-tiny.txt*.

Table 1 shows the PageRank values after each iteration: after 4 iterations, the result is the null vector. This result is obviously meaningless. Node 4 drains all the importance from the web, in each iterative step taking some of the importance of nodes 1 and 3, but not passing it on to other nodes, as it has no outgoing links—it is a so-called dangling node [1].

Dangling nodes

Dangling nodes are certainly not an exception in real-world graphs: for some subsets of the web, dangling nodes make up 80 percent of the collection's page [15]. Various methods to deal with these nodes have therefore been proposed [7, 11, 15, 22].

	PageRank value of node				
k	0	1	2	3	4
0	0.2000	0.2000	0.2000	0.2000	0.2000
1	0.0000	0.1000	0.1667	0.2667	0.2667
2	0.0000	0.0000	0.0333	0.2000	0.3000
3	0.0000	0.0000	0.0000	0.0000	0.3000
4	0.0000	0.0000	0.0000	0.0000	0.0000
⋮	⋮	⋮	⋮	⋮	⋮

Table 1: Distribution of $\boldsymbol{\pi}^T$ of the graph *sample-tiny.txt* in the kth iteration.

A standard approach is to modify $\mathbf{H}$ as follows: $\mathbf{S} = \mathbf{H} + \mathbf{A}$, where $\mathbf{A} = \mathbf{d}\mathbf{w}^T$. The vector $\mathbf{d}$ identifies dangling nodes (d_i is 1 if the ith row-sum of $\mathbf{H}$ is 0, and 0 otherwise) and $\mathbf{w}$ is a probability distribution vector [22]. An obvious choice for $\mathbf{w}$ is the use of an uniform probability distribution: this is equivalent to adding artifical links from dangling nodes to all pages.

When the power iteration is now applied to the same example, the algorithm converges (for $\epsilon = 1 \times 10^{-4}$, after about 15 iterations) to the stationary vector

$$\boldsymbol{\pi}^T = (0.0769, 0.1154, 0.1539, 0.2692, 0.3846).$$

2.2 Using a damping factor

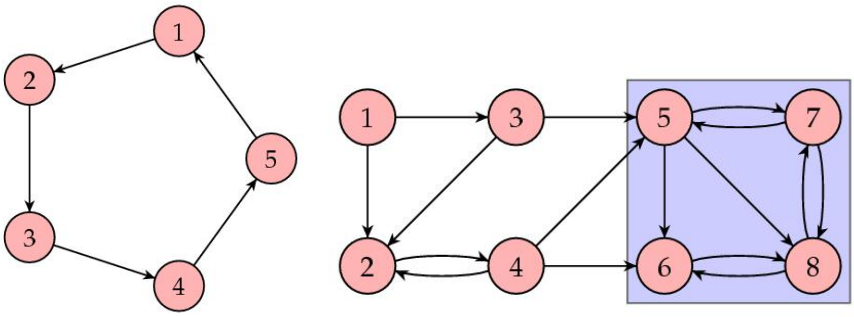
The matrix $\mathbf{S}$ is stochastic, it can be proven that stochastic matrices always have stationary vectors [8]. However, in general, we also have to use a damping factor α to *ensure convergence of the power iteration algorithm to an unique stationary probability vector, i.e., the PageRank, which has all strictly positive values* (proven by the Perron–Frobenius theorem) [1, 16, 22].

The need for the damping factor

Consider for instance the web graphs shown in Figure 2, for which our current version of the power iteration fails.

The graph in Figure 2a has a subdominant eigenvalue $|\lambda_2| = 1$ and fails to converge. A necessary condition for the convergence of the power iteration is that it is applied on a matrix that has subdominant eigenvalues with magnitude smaller than the dominant eigenvalue (i.e., $|\lambda_i| < |\lambda_1| = 1$, for $i = 2 \ldots n$). To satisfy the condition, the matrix $\mathbf{S}$ has to be *primitive* [1].

For the graph in Figure 2b, another problem arises. The power iteration converges to $\boldsymbol{\pi}^T = (0, 0, 0, 0, 0.12, 0.24, 0.24, 0.4)$. Clearly, nodes 1–4 should have some importance (there are nodes that link to them), but the importance sink (drawn in blue) drains the importance of these nodes in a way similar to dangling nodes as discussed above [1]. In general, we want $\boldsymbol{\pi}^T$ to have all strictly positive entries; a way of forcing this is making $\mathbf{S}$ *irreducible* which is equivalent to considering only strongly connected graphs [14, 16].



(a) Cyclic graph **(b)** Non strongly connected graph

Figure 2: Calculating PageRanks without a damping factor fails for certain graphs. (Examples are taken from [1].)

The Google matrix

To summarize, we have to modify $\mathbf{S}$ using the damping factor α in such a way that it is both primitive and irreducible [1]. This damping factor models the random surfer, who, with probability α, follows a link of the current page, and with probability $1-\alpha$ teleports to a random page using any other means (e.g., a bookmark).

This finally results in the *Google matrix*:

$$\mathbf{G} = \alpha\mathbf{S} + (1-\alpha)\mathbf{E} = \alpha\mathbf{H} + \alpha\mathbf{A} + (1-\alpha)\mathbf{E} \tag{1}$$

with the teleportation matrix $\mathbf{E} = \mathbf{e}\mathbf{v}^T$, where $\mathbf{e}$ is the vector of ones and $\mathbf{v}$ a probability distribution vector called the teleportation vector. A straight-forward choice—which we will use for the moment—is again the uniform distribution $\mathbf{v} = \frac{1}{n}\mathbf{e}$.

3 Damping factor trade-offs

3.1 Damping factor value

The damping factor α represents a weighting of the matrices $\mathbf{S}$ and $\mathbf{E}$ in the Google matrix. As $\alpha \to 0$, the web has a link between any two pages and we lose the original hyperlink structure of the web. On the other spectrum, as $\alpha \to 1$, the random surfer only uses the hyperlink structure: this has the consequence that important pages (i.e., pages that happen to be linked by many other pages, or by few important ones) will be visited more often by the random surfer and therefore will have higher PageRank scores [1].

Clearly, we want important pages to have higher PageRank scores; however, when we take a large value of α, the convergence of the power method will be very slow (it is easily proven that the convergence of the power iteration depends on the rate at which $\alpha^k \to 0$ [3, 16]).

Contrary to some popular belief, $\alpha \approx 1$ does *not* deliver "better" PageRank values [3]. Intuitively, this is easily explained: users do not surf on the web by just clicking on hyperlinks all the time. Additionally, sensitivity issues ($|\frac{d\pi_i(\alpha)}{d\alpha}| \leq \frac{1}{1-\alpha}$) can arise when the damping factor is chosen close to 1 [3, 16].

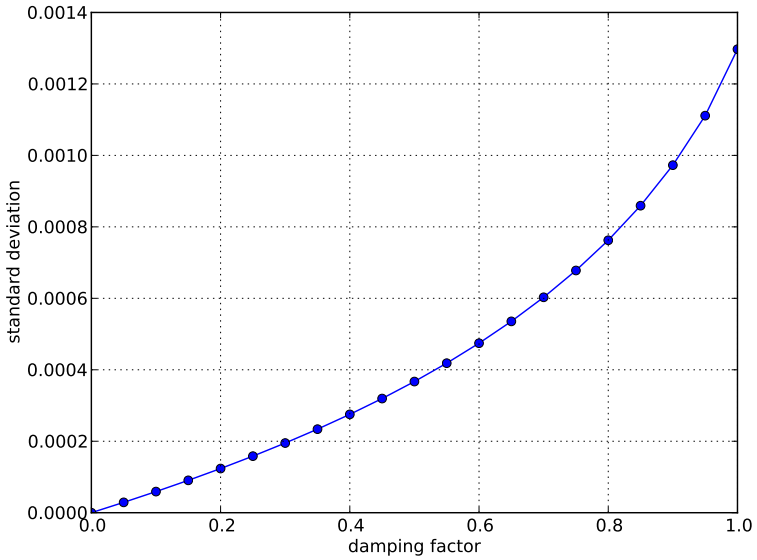


Figure 3: Damping value and standard deviation of PageRank scores.

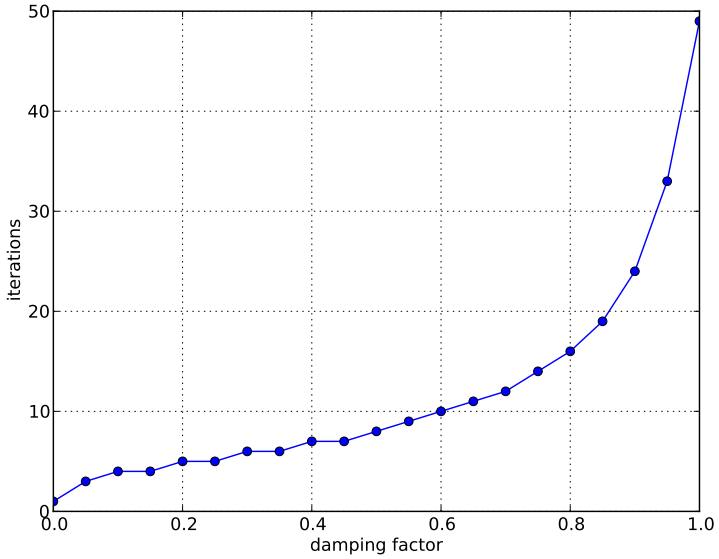


Figure 4: Damping value and number of iterations required for convergence of PageRank scores ($\epsilon = 1 \times 10^{-4}$).

3.2 Experiments

To illustrate these trade-offs, we calculate the PageRank values of the web graph given in *sample-large2.txt*, with various damping values from 0 to 1 in increments of 0.05.

Figure 3 shows the standard deviation of the PageRank values and Figure 4 the number of iterations for convergence. As expected, for small values of α, the algorithm takes less iterations to converge and the standard deviation is smaller, compared to larger values of α.

4 Personalisation of PageRank

Personalised search is a hot area since some predict them as the future (e.g., [9, 17, 19, 21]). The papers [9] and [15] give a good overview on this subject in the context of PageRank.

The teleportation vector $\mathbf{v}$ in the Google matrix (Eq. 1) is also known as the *personalisation vector* [22]. As its name suggests, this vector can be adjusted to model the random surfer's tendencies by introducing a bias on the probabilities that certain pages are chosen.

While the concept of full personalisation sounds wonderful in theory, doing this in practice is still computationally infeasible [15]. The personalisation currently used by Google[1] is based on the user's web history and location. This restricts the way personalisation is done (users can't have direct influence on how their results are personalised) [20], but allows for reasonable efficient computations while allowing some degree of personalisation.

Nevertheless, the concept of the personalisation vector is useful and has also been used for other purposes such as controlling spamming done by link farms [15, 22]. We now discuss some proposed methods to personalise PageRank more in-depth.

4.1 Topic-Sensitive PageRank

First consider a more restricted version of full personalisation called Topic-Sensitive PageRank [9]. The basic idea is to precompute a set of importances for a page with respect to various topics (by setting the personalisation vector values higher corresponding to pages relevant to those topics) [10]. (Note that the damping factor α has a direct influence on the amount of biasing [9].)

At query-time, the PageRank value is weighted according to this set of importances. There should be enough topics to have a useful model; using too many topics will however have a negative impact on the efficiency [9, 10]. This approach is suitable for query-sensitive PageRanking, but does not allow for "true" user personalisation.

4.2 Real-time personalisation

In [12] a linear relationship between personalisation vectors and their corresponding personalised PageRank vectors is identified. This relationship allows the personalised PageRank vector to be expressed as a linear combination of vectors that are called basis vectors [15]. (The number of basis vectors is a parameter in the algorithm.)

[1] See `http://www.google.com/psearch`.

The computation of the basis vectors is done by a scalable dynamic programming approach. At query time, an *approximation* to the personalised PageRank vector is constructed from the precomputed basis vectors [10, 12, 15].

Another possible approach involves the BlockRank algorithm [13]. It was originally designed as a method for accelerating the computation of the standard PageRank vector by finding a good starting vector, but has also been suggested as a method to create personalised PageRank vectors [15].

BlockRank is an aggregation method that lumps sections of the web by hosts, using the natural structure of the web. BlockRank consists of three main steps [15]:

1. First, local PageRanks for pages in a host are computed independently using the link structure of the host.

2. In the next step, these local PageRanks are weighted by the importance of the corresponding host.

3. Finally, the usual PageRank algorithm is run using the weighted aggregate of the local PageRank vectors as the starting vector. By assuming a web surfer can only teleport to hosts (rather than individual pages), personalisation can be accounted for in the second step.

This algorithm gives the personalised PageRank vector, *not* an approximation in contrast to the previous approach, with relatively low overhead [15].

5 Conclusion

The damping factor is crucial for the workings of the PageRank algorithm, in particular for the convergence of the power iteration. The damping factor allows an intuitive model of a random surfer, who not only surfs on the web by clicking links, but can also teleport to some page (with probability $1-\alpha$).

An important question remains the value of α: from empirical experiments (and supposedly used by Google), $\alpha = 0.85$, seems a good trade-off between convergence rate and "true" page rankings [18, 22]. However, no analytical justifications have been given; some researchers have also advocated the use of $\alpha = 0.5$ [2, 3, 4].

Personalisation makes the once query-independent, user-independent PageRanking calculations query-dependent, user-dependent and more calculation-laden [15]. While significant research has been done to minimise the effort associated with personalised results, search engines are still far from producing real-time fully personalised results [20].

References

[1] AUSTIN, D. How Google finds your needle in the Web's haystack, 2011. http://www.ams.org/samplings/feature-column/fcarc-pagerank.

[2] AVRACHENKOV, K., LITVAK, N., AND PHAM, K. S. A singular perturbation approach for choosing PageRank damping factor. Tech. rep., 2006.

[3] BOLDI, P., SANTINI, M., AND VIGNA, S. PageRank as a function of the damping factor. In *Proceedings of the 14th international conference on World Wide Web* (New York, NY, USA, 2005), WWW '05, ACM, pp. 557–566.

[4] BRESSAN, M., AND PESERICO, E. Choose the damping, choose the ranking? In *Proceedings of the 6th International Workshop on Algorithms and Models for the Web-Graph* (Berlin, Heidelberg, 2009), WAW '09, Springer-Verlag, pp. 76–89.

[5] BRIN, S., AND PAGE, L. The anatomy of a large-scale hypertextual web search engine. In *Seventh International World-Wide Web Conference (WWW 1998)* (1998).

[6] BRYAN, K., AND LEISE, T. The $25,000,000,000 eigenvector: The linear algebra behind Google. *SIAM Rev. 48* (March 2006), 569–581.

[7] CHI LEE, C. P. A fast two-stage algorithm for computing PageRank and its extensions. Tech. rep., 2003.

[8] GOLUB, G. H., AND SENETA, E. Computation of the stationary distribution of an infinite Markov matrix. Tech. rep., Stanford, CA, USA, 1973.

[9] HAVELIWALA, T., KAMVAR, S., KAMVAR, A., AND JEH, G. An analytical comparison of approaches to personalizing PageRank. Tech. rep., 2003.

[10] HAVELIWALA, T. H. Topic-sensitive PageRank: A context-sensitive ranking algorithm for web search. Technical Report 2003-29, Stanford InfoLab, 2003.

[11] IPSEN, I. C. F., AND SELEE, T. M. PageRank computation, with special attention to dangling nodes. *SIAM J. Matrix Anal. Appl. 29* (December 2007), 1281–1296.

[12] JEH, G., AND WIDOM, J. Scaling personalized web search. In *Proceedings of the 12th international conference on World Wide Web* (New York, NY, USA, 2003), WWW '03, ACM, pp. 271–279.

[13] KAMVAR, S., HAVELIWALA, T., MANNING, C., AND GOLUB, G. Exploiting the block structure of the web for computing PageRank. Tech. rep., 2003.

[14] LANGVILLE, A. N., AND MEYER, C. D. Fiddling with PageRank, 2003.

[15] LANGVILLE, A. N., AND MEYER, C. D. Deeper inside PageRank. *Internet Mathematics 1* (2004).

[16] LANGVILLE, A. N., AND MEYER, C. D. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, Princeton, NJ, USA, 2006.

[17] MATTHIJS, N., AND RADLINSKI, F. Personalizing web search using long term browsing history. In *Proceedings of the fourth ACM international conference on Web search and data mining* (New York, NY, USA, 2011), WSDM '11, ACM, pp. 25–34.

[18] PAGE, L., BRIN, S., MOTWANI, R., AND WINOGRAD, T. The PageRank citation ranking: Bringing order to the Web. Tech. rep., Stanford Digital Library Technologies Project, 1998. 17 p.

[19] STAMOU, S., AND NTOULAS, A. Search personalization through query and page topical analysis. *User Modeling and User-Adapted Interaction 19* (February 2009), 5–33.

[20] VAN REST, F. A mathematical approach to scalable personalized PageRank. Master's thesis, Universiteit Leiden, 2009.

[21] WEDIG, S., AND MADANI, O. A large-scale analysis of query logs for assessing personalization opportunities. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (New York, NY, USA, 2006), KDD '06, ACM, pp. 742–747.

[22] WILLS, R. S. Google's PageRank: The Math Behind the Search Engine, 2006. `http://www.emba.uvm.edu/~lakobati/AppliedUGMath/other_Google/Wills.pdf`.