Web Search Engines and Linear Algebra

Špánek, Roman
2006

Dostupný z http://www.nusl.cz/ntk/nusl-36751

Institute of Computer Science
Academy of Sciences of the Czech Republic

Web Search Engines and Linear Algebra

Roman Špánek

Technical report No. 974

September 2006

Pod Vodárenskou věží 2, 182 07 Prague 8, phone: +420 266 051 111, fax: +420 286 585 789,
e-mail:ics@cs.cas.cz

Institute of Computer Science
Academy of Sciences of the Czech Republic

Web Search Engines and Linear Algebra[1]

Roman Špánek
[2]

Technical report No. 974

September 2006

Abstract:

The technical report presents a brief overview on web search engines with deeper insight into their linear algebra background. The linear algebra plays very important rule in modern web search algorithms (e.g. Google). The report presents two algorithms, particularly HITS and PageRank. The algorithms are discussed on their convergence problems and also some improvements to their personalization abilities. The computation complexity is also mentioned and briefly sketched.

Keywords:
Web searching engines, PapeRank, HITS, Power Method

[1]The work was supported by the project 1ET100300419 of the Program Information Society (of the Thematic Program II of the National Research Program of the Czech Republic) "Intelligent Models, Algorithms, Methods and Tools for the Semantic Web Realization", partly by the Institutional Research Plan AV0Z10300504 "Computer Science for the Information Society: Models, Algorithms, Applications" and by Ministry of Education, Youth and Sports of the Czech Republic, project No. 1M4674788502 Advanced Remedial Technology and Processes.

[2]Institute of Computer Science, Acad. of Sci. of the Czech Republic, P.O. Box 5, 182 07 Prague 8, Czech Republic

1 Introduction

Current web would not have been such a success, had it not for efficient search engines helping users to find requested resources. Therefore it is not a surprise that lot of research activities have been in the filed of web search engines. On the other hand, the search engines would not offer so good services if they did not employ some sophisticated mathematical methods.

The HITS [1] and PageRank [2],[3] are the most popular link analysis algorithms. Both were clearly described in [4]. These algorithms were proposed in 1998 and were a milestone in web search technologies. The state of the art before year 1998 was unacceptable, while web search engines produced a very long list of relevant pages and a user had to do all the confusing work of selecting the more and less relevant ones by simply clicking on the links. Moreover, it was quite easy to spam the pages and therefore the ordering of result web pages did not help a lot. Example of spamming techniques was a placing the most popular search terms printed out in white on a white background. This resulted in increasing rating of the page. Such situation was a real danger for future of the Internet, while the Internet was holding huge mass of useful information but with out efficient way to search for relevant ones, it might have let to decrease of users' attention.

This obvious space was then filled be many cleaver techniques to improve search engines accuracy and efficiency. In the rest of the report we will concentrate on the most important and popular ones, HITS and PageRank algorithms.

2 The HITS algorithm

The HITS [1] algorithm for link analysis was developed by Jon Kleinberg from Cornell University during his postdoctoral studies at IBM. The basis of the algorithm is quite simple and lies in understanding patterns that can be found in the web structure. Kleinberg noticed that some pages usually point to many other pages and on the other hand some pages are pointed by many others. Therefore he proposed two terms: *hubs* and *authorities*. The *hubs* are pages with many outlinks; the pages that point to many other. Pages pointed by many other pages, therefore having many inlinks, are *authorities*. Moreover, Kleinberg noticed two rules:

good hubs seemed to point to good authorities and good authorities were pointed to by good hubs

To record these rule he decided to compute and assign two scores to every web page i. Particularly, hub score h_i and authority score a_i. The scores were computed in iteration cycles. Therefore the hub score for page i at iteration k was labeled as $h_i^{(k)}$ and authority score as $a_i^{(k)}$. The definition of the scores follow:

$$\begin{aligned} a_i^{(k)} &= \sum_{j:e_{ji}\in E} h_j^{(k-1)} \\ h_i^{(k)} &= \sum_{j:e_{ij}\in E} a_j^{(k)} \quad \text{for } k = 1, 2, 3, \ldots, \end{aligned} \tag{2.1}$$

where e_i represents a hyperlink from page i to page j and E is the set of hyperlinks.
It is clear that at the very begin of the computation it is necessarily to have some starting values for both scores. Kleinberg decided to start computation simply with uniform scores for all pages:

$$\begin{aligned} a_i^{(1)} &= 1/n \\ h_i^{(1)} &= 1/n \end{aligned} \tag{2.2}$$

where n is total number of pages in a so-colled neighborhood set for the query list.
The neighborhood set contains all pages in the query list with addition of all pages pointing to and also from query pages. The amount of pages in neighborhood might be in size of hundred or hundred thousand of pages depending on the query. Latent association can be made in the neighborhood set.

The process of computing of the both scores is then started in an iterative fashion until convergence to stationary values is reached.

To solve the problem of quite huge amount of pages in computation a reduction is necessarily. Therefore one can replace, using linear algebra, the summation equations with matrix equations. Let **h** and **a** be column vectors containing hub and authority scores, respectively. Adjacency matrix for the neighborhood is **L**. The L matrix is set up in that $\mathbf{L}_{ij} = 1$ if page i points to j, and 0, otherwise. Having these definition, the equations 2.1 can be rewritten as:

$$\begin{aligned} a^{(k)} &= L^T h^{(k-1)} \\ h^{(k)} &= L a^{(k)}. \end{aligned} \tag{2.3}$$

Using algebra it can be easily shown that

$$\begin{aligned} a^{(k)} &= L^T L a^{(k-1)} \\ h^{(k)} &= L L^T h^{(k-1)} \end{aligned} \tag{2.4}$$

From these equations it is clear that Kleiber's algorithm is a power method (a power method for solving eigenvalues is mentioned in section 4) applied to the positive semi-definite matrices $L^T L$ and LL^T. $L^T L$ is called hub matrix and LL^T is called authority matrix. The HITS is so reduced to solving the eigenvector problems:

$$\begin{aligned} L^T L a &= \lambda_1 a \\ L L^T h &= \lambda_1 h \end{aligned} \tag{2.5}$$

where λ_1 is the largest eigenvalue of the matrices, and **a** and **h** are corresponding eigenvectors.

Even though there are some issues like convergence, existence, uniqueness worth considering we will not take more attention on them. HITS algorithm has been also modified with various advantages and also disadvantages [5],[6],[7].

A variation of HITS's concept is the base of web search engine TEOMA (http://www.ask.com/).

3 The PageRank Algorithm

As was mentioned, before 1998 there were algorithms aimed to be the solution of the problem with unreliable web search engines. One of them, already mentioned, was HITS algorithm and the second was PageRank algorithm. Both were examples of link analysis algorithms. The PageRank algorithm is the search heart of well known Google web search engine and was proposed by Sergey Brin and Larry Page during their study at Stanford University. The PageRank algorithm also uses recursive schema as HITS does. On the other hand, the basis idea is a bit different:

"A page is important if it is pointed to by other important pages."

Although it might be seen a bit strange, your web page rank (its importance) is recursively calculated by summing rank of all pages pointing to yours. Very important aspect of the PageRank algorithm is that Brin and Page found that rank of pointing page cannot be used as whole, but it should be distributed proportionately. Particularly, assume that your page is pointed to by only Yahoo!, which also points to many other pages. Yahoo! surely has a big page rank, but the page rank given to the pointed page is not the Yahoo! rank, but it is its rank divided by amount of pages being pointed by Yahoo!. Assume that Yahoo! points to 1000 pages, the pointed page is then given by 1/1000 of Yahoo! page rank. The PageRank definition follows:

$$r_i^{(k+1)} = \sum_{j \in I_i} \frac{r_j^{(k)}}{|O_j|} \tag{3.1}$$

where $r_i^{(k)}$ is the PageRank of pointed page i at iteration k, I_i is the set of pages pointing to page i and finally $|O_j|$ is the amount of pages being pointed by page j. As was mentioned in case of HITS algorithm, also PageRank starts with uniform rank for all pages at zero iteration cycle. Therefore, the starting rank of page i at zero iteration is $r_i^{(0)} = 1/n$, where n is total number of Web pages. Before

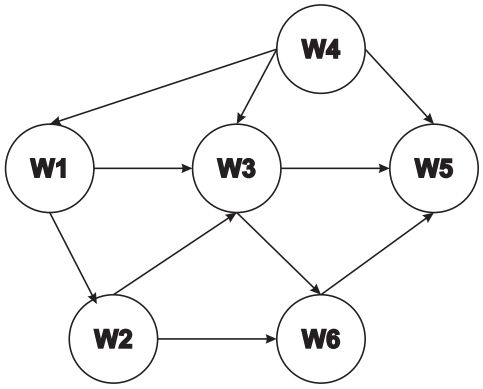

Figure 3.1: Example of web link-structure

we proceed, a matrix notation of the preceding formula should be defined as:

$$\pi^{(k+1)T} = \pi^{(k)T}\mathbf{H},$$

where $\pi^{(k)T}$ is PageRank vector (a row vector) at i^{th} iteration, $\mathbf{H}$ is row normalized hypermatrix. The $\mathbf{H}$ hyprermatrix stands for the structure of the Web and the elements of $\mathbf{H}$ are defined as:

$$\begin{aligned} h_{ij} &= 1/|O_i| \text{ if a link from page } i \text{ to page } j \text{ exists} \\ h_{ij} &= 0 \text{ otherwise.} \end{aligned}$$

In the Figure 3.1 one can see an example of web structure. Nodes stand for web pages and arcs denote hyperlinks between them. An example of matrix $\mathbf{H}$ of the web structure from Figure 3.1 given above follows:

$$\mathbf{H} = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 1/3 & 0 & 1/3 & 1/3 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

It is clear that matrix $\mathbf{H}$ from the example has dimensions $6x6$, while in the figure there are six webpages labeled as W1 to W6.

Unfortunately, this procedure suffers by convergence problems. Due to the convergence problems, Brin and Page a bit modified the proposed algorithm. The concept of hyperlink structure of the web was preserved. They build irreducible aperiodic Markov chain characterized by a primitive transition probability matrix. The irreducibility is the key factor influencing the convergence, while it guarantees existence of a unique stationary distribution vector - the new PageRank vector. In [8] authors showed that the power method applied on primitive stochastic iteration matrix will always converge to the PageRank vector independently of the starting vector, and moreover, the convergence rate is determined by the magnitude of the subdominant eigenvalue λ_2. Further, the power method guarantees not only the convergence independently of the starting vector, but it also speed up much the computation. For more information see section 4.

The power method requires a primitive stochastic matrix. Therefore, here one can see how the Google does the transformation of hyperlink structure of the Web. Firstly, stochastic matrix is a matrix where following holds: "Let I be a finite or countable set, and let $\mathbf{P} = (p_{ij} : i, j \in I)$ be a

matrix and let all p_{ij} be nonnegative. We say $\mathbf{P}$ is stochastic if:

$$\sum_{i \in I} p_{ij} = 1$$

for every $j \in I$. In other words, a matrix $\mathbf{P}$ is stochastic if every column is a distribution.
Google defines $\mathbf{H}$ matrix as an $n \times n$ (iff there is n pages in the Web) matrix whose elements h_{ij} are probabilities of moving from page i to page j in one mouse click (see stochastic matrix definition above). To do this transformation the simples way is to set $h_{ij} = 1/|O_i|$. To make this clear see the following example of $\mathbf{H}$:

$$\mathbf{H} = \underbrace{\begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 1/3 & 0 & 1/3 & 1/3 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{Probability of moving from page } i \text{ to page } j}$$

Assume, that a user is at page 4 (row four in the example above) and he/she is about to click on one of the existing hyperlinks (to pages 1,3, and 5). The probability that the user clicks on the hyperlink to page 3 is 1/3 while the total amount of outgoing links is 3.

Nevertheless, the $\mathbf{H}$ matrix can obviously contain some zero rows (row 5 in the example of matrix $\mathbf{H}$ above and web page W5 in the Figure 3.1), which means that it might not be stochastic. Such situation occurs for all pages having no outlinks and unfortunately there are many pages with this feature - we will call them *dangling nodes*. The dangling nodes can be fixed in an easy way by replacing all zeros rows with e^T/n, where e^T is a row vector containing all ones. A new stochastic matrix $\mathbf{S}$ created from $\mathbf{H}$ is then given as

$$\mathbf{S} = \mathbf{H} + ae^T/n.$$

$$a_i = \begin{cases} 1 & \text{if page i is a dangling node,} \\ 0 & \text{othervise} \end{cases}$$

An example of now stochastic matrix S created from example above:

$$S = \underbrace{\begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 1/3 & 0 & 1/3 & 1/3 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{Revised Stochastic matrix}}$$

Note, that uniform vector e^T/n can be replaced by any probability vector p^T if the following condition hold $p^T e = 1$.

Remind that existence of a *unique* stationary distribution vector is required by well defined PageRank. Unfortunately, the modification of matrix $\mathbf{H}$ to matrix $\mathbf{S}$ does not guarantee the existence of the vector. Shortly, the irreducibility on top of stochasticity is required. In the other words, what we need is a matrix that describe strongly connected graph, and the Web doesn't have the strong connectivity property. So the adjustment to matrix $\mathbf{S}$ to make it irreducible creates the *Google matrix* $\mathbf{G}$, which is defined to be

$$\mathbf{G} = \alpha\mathbf{S} + (1 - \alpha)\mathbf{E}, \tag{3.2}$$

where $0 \leq \alpha \leq 1$ and $\mathbf{E} = e.e^T/n$ The easiest way how to guarantee the strong connectivity, is to take **S** matrix and to add a matrix that have no zeros elements, see the example below:

$$\mathbf{G} = \alpha \underbrace{\begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 1/3 & 0 & 1/3 & 1/3 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{S} \text{ with zero elements}} + (1-\alpha) \underbrace{\begin{bmatrix} 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \end{bmatrix}}_{ee^T/n} =$$

$$= \alpha \underbrace{\begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 1/3 & 0 & 1/3 & 1/3 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{S} \text{ with zero elements}} +$$

$$+ \underbrace{\begin{bmatrix} (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 \\ (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 \\ (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 \\ (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 \\ (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 \\ (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 & (1-\alpha)/6 \end{bmatrix}}_{(1-\alpha).ee^T/n}$$

In this moment, while **G** is a convex combination of the two stochastic matrices **S** and **E**, **G** is naturally both stochastic and irreducible. Note that in **G** every node is connected with all others. One might think that it is strange because the real Web does not have such a property, but note that the probability of moving from one site to the other is very small when there had not been hyperlink in **H**. Nevertheless, if you have ever used Google for searching then you might have noticed that Google sometimes does some PageRank correction - personalization. To allow the personalization feature, Google eventually replaced the uniform vector e^T/n with a general probability vector v^T, so that $E = ev^T$ instead. See [9], [10] for details on the personalization of Google and vector v^T.

4 Power Method

Having specified matrix **G**, we are now facing the problem of evaluating the PageRank vector $\pi^{(k)T}$ in an acceptable time. Further, take in mind that size of the matrix was about 4.3 billion pages in 2004 and has quickly risen to approximately 25 billion indexed pages in 2006. That the computation method of PageRank is the power method is very natural. The reasons follow. Consider iterates of the power method applied to completely dense matrix **G** assuming $\mathbf{E} = ev^T$, then

$$\begin{aligned} \pi^{(k)T} &= \pi^{(k-1)T}\mathbf{G} = \alpha\pi^{(k-1)T}\mathbf{S} + (1-\alpha)v^T = \\ & \alpha\pi^{(k-1)T}\mathbf{H} + (\alpha\pi^{(k-1)T}a + (1-\alpha))v^T \end{aligned}$$

The intent of the precedent equation is that now we can apply the power method to extremely sparse matrix **H**, and moreover dense matrices **G** and also **S** are never neither formed nor stored. Matrix free methods, like the power method is, are required due to the mentioned size of matrices and vectors involved in the computation.

Let us very briefly mentioned the computation complexity of the power method applied to very sparse matrix **H**. Each vector multiplication can be computed in *nzz(**H**)* flops, where *nzz(**H**)* is the number

of non-zeros in **H**. So the question is how many non-zeros can be found in **H**? Fortunately, the answer can be very easily found. First, mind how the **H** matrix was specified:

$$\begin{aligned} h_{i,j} &= 1/|O_i| \text{ if a link from page } i \text{ to page } j \text{ exists} \\ h_{i,j} &= 0 \text{ otherwise.} \end{aligned}$$

Then, each row stands for a page and every column stands for all other pages in the index (current PageRang indexes 25 billion pages). Therefore, if page has for example 25 outlinks, then 25 non-zeros elements are in i-th row. It is clear that each row contains much less non-zeros then the size of **H** matrix is, further the average number of non-zeros is less then 10. Therefore the complexity is $O(nnz(\mathbf{H})) \approx O(N)$. The last question concerns convergence problem. In [3] Brin and Page showed that only 50 to 100 iteration cycles are enough. The reason lies in the fact that it can be proven [10] that subdominant eigenvalue of **G** satisfies $|\lambda_2| \leq \alpha$, and Google originaly set $\alpha = 0.85$.

While both concepts, HITS and PageRank, are simple it is good to note that the precedent gave only brief overview on the most important aspects. Many other important features and issues, like personalization, have not been mentioned but one can found some useful information in [11], [12], [13], [10], [14], [15].

5 Conclusion

The technical report summarizes the development in the area of web search engines algorithms, particularly, HITS and PageRank algorithms are discussed. The algorithms are described from the linear algebra point of view. Both algorithms are based on similar idea inspired by the Internet structure. HITS algorithm defines terms *hubs* and *authorities* that describe pages having many outlinks and pages with many inlinks, respectively. The PageRank algorithm, on the other hand, uses a slightly different approach how to describe the structure of the Internet. Further, both algorithms require efficient way of computing ranks, while amount of pages available on Internet is vast. Therefore some modifications are needed to achieve matrices properties required by the Power Method the computational method used. The modifications are described in details in case of the PageRank algorithm. The Power method is shown to be the right choice while it is able to converge in 50 to 100 iterations.

Bibliography

[1] J. Kleinberg, "Authoritative sources in a hyperlinked environment" in *Jurnal of the ACM*, 46, 1999.

[2] S. Brin, L. Page, "The anatomy of a large-scale hypertextual web search engine" in *Computer Networks and ISDN Systems*, 33:107-117, 1998.

[3] S. Brin, L. Page, R. Motwami, T. Winograd "The PageRank citation ranking: bringing order to the web" in *Technical Report*,Computer Science Department, Stratford University, 1998.

[4] A. N. Langville, C. D. Mayer, "The Use of Linear Algebra by Web Search Engines" in *Bulletin of the International Linear Algebra Society No. 33* , Dec., 2004, pp. 2-6.

[5] A. Ding, X. He, H. Zha, H. Simon, "PageRank, HITS and an unified framework for link analysis" in *Proceedings of the 25th ACM SIGIR Conference*, Tampere, Finland, August 2002, pp. 353-354.

[6] A. Farahat, T. Lafaro, J. C. Miller, G. Rae, L. A. Ward, "Modifications of Kleinberg's HITS algorithm using matrix exponentiation and web log records." in *ACM SIGIR Conference*, September, 2001, pp. 444-445.

[7] F. Fouss, J. Renders, M. Saerens, "Some relationships between Kleinberg's bubs and authorities, correspondence analysis, and the Salsa algorithm." in *Proceedings of the 7th Conference on the Statistical Analysis of Textual Data (JADT 2004)*, 2004, pp. 445-455.

[8] Carl D. Mayer, "Matrix Analysis and Applied Linear Algebra." in *SIAM*, Philadelphia, 2000.

[9] aher H. Haveliwala, Sepandar D. Kamvar, and Glen Jeh "An analytical comparison of approaches to personalizing PageRank" Technical Report, Stranford University, 2003.

[10] Amy N. Langville and Carl D. Mayer, "Deeper inside PageRank" in *Internet Mathematics Jurnal*, 2004.

[11] A. Farahat, T. Lofaro, J. C. Miller, G. Rae, and L. A. Ward, "Existence and uniqueness of ranking vectors for liner analysis." in *ACM SIGIR Conference*, September 2001.

[12] S. D. Kamvar and T. H. Haveliwala, "The condition number of the PageRank problem" *Technical report*, Stanford university, 2003.

[13] S. D. Kamvar, T. H. Haveliwala, and G. H. Golub, "Adaptive methods of web information retrieval" *Technical report*, Stanford university, 2003.

[14] C. Pan-Chi Lee, G. H. Golub, and S. A. Zenois, "Partial state space aggregation based on lumpability and its application to PageRank" *Technical report*, Stanford university, 2003.

[15] R. Lembel, S. Moran, "Rank-stability and rank-similarity of link-based web ranking algorithms in authority-connected graphs" in *Second Workshop on Algorithms and Models for the Web-Graph (WAW 2003)*, Budapest, Hungary, May 2003.