Algorithms on the Web Graph

Dániel Fogaras*

Department of Computer Science and
Information Theory
Budapest University of Technology and
Economics
Magyar tudósok körútja 2., I.B.132.
Hungary, H-1117
fd@cs.bme.hu

Abstract: Apart from the textual information, the World Wide is a set of empty web pages with hyperlinks between them, and this structure is referred to as the web graph. We summarize the graph algorithms designed for the web graph to support browsing the Internet or to explore the structure of the web. The solutions are usually based on some spectral heuristics derivated from some intuitions. These intuitions assume the existence of some underlying model, which controls the evolution of the hyperlinks of the web.

Keywords: link-analysis, web graph, web algorithm

1 Introduction

The World Wide Web (WWW) is one of the phenomenona appeared in the last decades greatly effecting our everyday life. If someone has a problem to solve or a piece of information to access, with high probability thousands of web pages are related to this problem. The bottleneck is not the existence of information, but processing such an enormous database and finding the most relevant pages. Hyperlinks support the navigation, which are incorporated within the information.

On a more abstract level the web is a directed graph $G(V, E)$ referred to as the *web graph*. Each web page corresponds to a node $v \in V$, and $(u, v) \in E$ if there exists a hyperlink from the page of u to that of v. The web graph is interesting to investigate both from theoretical and practical point of view. Some experiments reveal the most important graph parameters, such as the diameter, the average degree, degree distribution and the size of the largest strongly connected component of the web graph [20, 13, 6]. Furthermore, there is a large effort on the research area, which tries to explain the result of the experiments by simplified models yielding to the already measured properties [22, 2, 29, 19]. This may give a new direction of random graph theory in the future, since the classical Erdős–Rényi random graph model [15] differs from the web in many aspects. For instance the degree distribution of the web graph follows a power-law, i.e., the probability of a page with outdegree k is proportional to $k^{-\alpha}$ with $\alpha \approx 2$ [6]. On

*Research is supported by ???

the contrary, if a random graph has n nodes and each pair of nodes is connected with probability p, then the degree distribution is binomial close to Poissonian for large n.

From the practical point of view, the structure of the web can act as input for *web algorithms*. Typical applications aid human users to navigate, which is the most challenging task of using the web. The most well-known examples of web algorithms are implemented in search engines, such as Google or Yahoo. Web algorithms may use textual information written in the web pages: keywords, titles, and most frequently used expressions can be analyzed by softwares without any kind of human interaction. The web graph is also a rich source of information for web algorithms, although it contains only the following information type: *one page recommends the other to visit*. The textual information has a large variance depending on the author's language, style and programming skill. The linkage information, however, is more homogenous and independent of the language. Furthermore, the number of links on a page is significantly smaller than that of words.

In practical applications we need to pay attention to the difficulties of implementing web algorithms. First of all, the input of a web algorithm can never be complete neither accurate, since the web graph is available through downloading the pages by *crawlers*. These softwares visit and download the contents of the web pages. The URLs to visit by a crawler can be controlled by a programmer, but the main source of URLs is collected by the crawler itself. Thus, a crawler performs a walk on the web graph and the input of a web algorithm is the structure obtained from the walk of the crawler. This structure can only be a perturbed version of some part of the always changing web graph. Controlling the walk of a crawler is far from being trivial, see [10, 11]. The second problem of realization comes from the incredibly large size of data to deal with. The downloaded fraction of the web can be much larger than the capacity of the main memory of a computer, so the structure can only be stored in the external memory, which restricts the possible algorithmic solutions. Another aspect which needs consideration is the required response time, which is determined by the application. If the web algorithm aids a user in browsing the web, then it has no time to process the whole structure on-line. Some applications, however, preprocess the structure off-line and use the output of preprocession on-line.

In this survey we focus on web algorithms which use the web graph or a subgraph of it as an input, and extract some meaningful information. This article is divided into three sections discussing different applications. The first is about extracting the most densely connected communities of the web. The second application ranks some set of web pages according to the users interest. Finally, we summarize the problem of defining a similarity measure on pages, which can be evaluated effectively. Besides the heuristic solutions we take considerations about the underlying intuition, which makes us believe that the output is meaningful. Such an intuition is always related to some heuristic explanation of the evolution of the hyperlinks.

2 Tracing web communities

Intuitively a *web community* is a set of pages sharing a common interest on some topic, and the members frequently visit the most relevant pages about the topic. An example is depicted on Fig. 1. In the following section we will have a closer investigation on the structure of these communities yielding to an algorithm to enumerate them following the results of Kumar et al. [21].

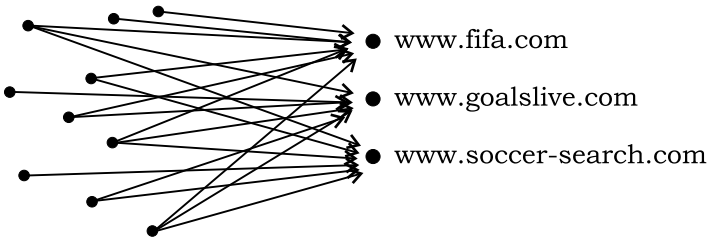

Figure 1: The web community is formed by pages related to soccer. The nodes on the right are good authority pages, while the others are fans.

Naturally, we can distinguish between two types of members of a community. A *fan* is a reader member not contributing any interesting information for others. Such a page is composed mainly for the user himself with some navigational tools according to his interest. The content of an *authority*, however, excites the other members of the community. Consequently, the fans are likely to link to authorities yielding to the following hypothesis.

Hypothesis 1 *A web community contains a large number of links directed from the set of fans to the authorities, thus forming a dense directed bipartite subgraph.*

One may view on this subgraph as an instance of a random bipartite graph, with m edges placed between the set F of fans and A of authorities randomly. Note that all the edges are directed towards A. It is a well-known fact from random graph theory that with high probability there exist $A' \subseteq A$ and $F' \subseteq F$ such that there is a complete bipartite graph between A' and F'. (The sizes of F' and A' are functions of m, $|F|$ and $|A|$.)

Hypothesis 2 *Each web community contains a complete bipartite graph with all the edges directed from one class to the other.*

Such a subgraph is referred to as a *core*, which acts as a mark of the community. For instance the community depicted on Fig. 1 contains a core with $|A'| = |F'| = 3$. Kumar et al. aim at finding empirical evidence of the above intuitive implications by implementing an algorithm, which traces the cores of the web. It is named *trawling* (by mixing the words crawling and tracing), and it enumerates a maximal set of pairwise disjoint cores from the web graph for fixed small constants $i = |A'|$ and $j = |F'|$ (in the range 3-10). As the input was almost the whole web graph, they coped with external memory implementation reducible to sorting and small batches of tasks. They conducted experiments on a database, which was crawled one year before the experiment. Some of the enumerated cores had grown largely during the one year period. So the trawling algorithm may indicate the presence of still small but developing communities as well.

3 Ranking the pages

In this section we describe the most successful and wide-spread application of algorithms based on the linkage information. Suppose we are looking up web pages containing a *query string* for

example "sushi restaurant". The phrase is submitted to a *query search engine* of a downloaded database containing a significant fraction of the web. We refer to the survey [4] on the anatomy of query search engines and the maintenance of collections of web pages. The search engine returns the *query set*, i.e., the set of pages in which the query string occurs. The size of such a set is likely to reach tens of thousands, and it can be even higher by magnitudes if the phrase is very common. Then, the computer enumerates the elements of the query set for the user. A typical user, however, does not check more than the first few elements, so the rest of the set remains unexplored. How can a machine distinguish between sushi restaurants, and decide which one should appear on the top of the list? In this section we focus on *ranking algorithms* which assign non-negative real values to the pages, such that the higher the rank of a page is, the more relevant information is available on the page. These values determine the order in which the elements of the query set will be enumerated on the screen.

3.1 Page Rank algorithm

We introduce the results of Brin and Page [9] in the following section. They propose ranking algorithms, to assign a value for each page of the web off-line in advance. Thus, the same rank will be used for a given page in any query set.

Recall that the existence of a link (p_1, p_2) implies that page p_1 recommends p_2 to visit. So the more pages link to a node the more popular the corresponding node is. A simple idea is that the indegree of a node can be chosen as its rank. This solution, however, has the drawback of treating the opinions of pages in an equal manner. If `www.yahoo.com` links to a page p for example, then it should be present with higher weight in the rank of p.

Let $G(V, E)$ denote an arbitrary strongly connected directed graph, and for a node $v \in V$ the set of nodes linking to v is denoted by $d^-(v)$ and those linked from v by $d^+(v)$.

Definition 3 *The* simple Page Ranks $s_v \geq 0$, $v \in V$ *are the solution of the following linear system of equations with* $\sum_{v \in V} s_v = 1$

$$s_v = \frac{1}{|d^-(v)|} \sum_{w \in d^-(v)} s_w, \qquad v \in V.$$

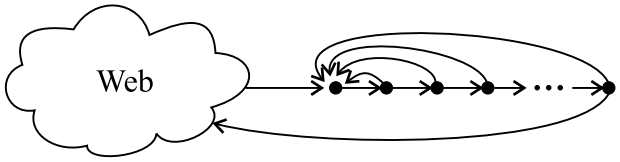

Figure 2: Let U denote the set of nodes of the chain. As there is only one page of $V \setminus U$ linking to any member of U, the total rank of nodes of U should be small. While, a random walk can be trapped in U as the probability of leaving U from its left most node without returning is $\frac{1}{2^{|U|-1}}$. Thus, the simple Page Rank of the left most node can be very high.

Notice that the solution is the stationary distribution of an arbitrary random walk on G [3]. The simple Page Rank still has a strong deficiency, if we apply it to the largest strongly connected component of the web graph. The problem is summarized through an example on Fig. 2. The following definition eliminates the difficulty:

Definition 4 (Brin, Page, 1998) *The* Page Rank $r_v \geq 0$ *of a node* $v \in V$ *is defined as the probability of* v *in the stationary distribution of the following random walk on the nodes of* G*. If the walk is currently on a node* $w \in V$*, then it jumps to aan arbitrary node of* V *with probability* $\frac{\epsilon}{|V|}$*, and to a node of* $d^+(w)$ *with probability* $\frac{1-\epsilon}{|d^+(w)|}$*, where* $0 < \epsilon < 1$ *is a fixed constant. Equivalently the* r_v *values solve the following linear system with* $\sum_{w \in V} r_w = 1$

$$r_v = (1-\epsilon) \sum_{w \in d^-(v)} \frac{r_w}{|d^+(w)|} + \frac{\epsilon}{|V|}, \qquad v \in V.$$

Let A denote the adjacency matrix of G, and P the stochastic matrix obtained from A by normalizing its rows. Furthermore J denotes the matrix with the same size as A containing only 1 entries. Then the Page Rank vector $\underline{r} \in R^{|V|}$ is the principal left singular vector of $(1-\epsilon)P + \epsilon J$, and it can be approximated by the power iteration method [16].

This procedure is successfully applied in practice to the largest strongly connected component of the web graph, as it is the heart of the search engine Google [28]. The value of $\epsilon \approx 0.1 - 0.2$ is used and the number of iterations is approximately $50 - 100$. Each iteration takes $O(|E|)$ time, so it is proportional to the number of hyperlinks of the whole web. Such an amount of data can only be stored in the external memory, so the above computation takes hours for the whole web.

3.2 Hub-authority algorithm

The algorithm discussed in this section ranks the set of pages sharing a query string q on-line. So the input is a subgraph $G_q(V_q, E_q)$ of the web graph induced by the nodes corresponding to the elements of the query set. Since the algorithm was published by Kleinberg [18], many variants appeared (see the survey [8].)

First, we intuitively explain how the structure of G_q evolved. Each page contains some navigational information and some textual information about the topic described by the query phrase. The one containing valuable navigational information is a good *hub*, the one containing the more relevant textual information is a good *authority* in G_q. We suppose the latent existence of two non-negative constants h_v *hub* and a_v *authority scores* assigned to each node $v \in V_q$ describing its value as a hub and as an authority. Let $\underline{a}$ and $\underline{h}$ denote the vectors of dimensions $|V_q|$ containing the hub and authority scores.

Hypothesis 5 *The probability that an edge* (v, w) *is present in* G_q *is a monotone function of* h_v *and* a_w*, so the edges tend to link from good hubs towards good authorities.*

Note that this hypothesis generalizes Hypothesis 1 by treating G_q as a web-community. It explains the existence of all the edges of G_q, not only the edges between fans (nodes with small a_w values) and authorities (nodes with large a_w values.) If the intuitions are statistically true, then $\underline{a}$ is an excellent ranking on the elements of $|V_q|$. On the other hand the nodes with high hub scores may also interest a user. For instance the `www.sushi.infogate.de/linx.htm` within the query set of "sushi" is a good hub, as it links to several authority pages about "sushi". Therefore it is worth enumerating both good hubs and authorities once the scores are available.

Kleinberg's algorithm computes hub and authority scores on the basis that the whole edge structure of G_q is statistically a trace of $\underline{h}$ and $\underline{a}$. The following iterative algorithm captures the mutual reinforcement relationship between hub and authority scores. The $|V_q|$ dimensional vectors $\underline{a}^{(i)}$ and $\underline{h}^{(i)}$ contains the *approximated hub* and *authority scores* $a_v^{(i)}$ and $h_v^{(i)}$ for all

$v \in V_q$, $i = 0, 1, 2, \ldots$ Furthermore, $d^-(v) \subseteq V_q$ and $d^+(v) \subseteq V_q$ denotes the subsets of nodes linking to v and linked by v respectively.

Definition 6 (Kleinberg, 1998) *Let $\underline{h}^{(0)} \neq \underline{0}$ denote an arbitrary non-negative vector of $R^{|V_q|}$. The* approximated hub *and* authority scores *are defined by the following iterative equations for all $v \in V_q$, where $\|\cdot\|$ denotes the 2-norm of a vector.*

$$x_v^{(i+1)} = \sum_{w \in d^-(v)} h_w^{(i)} \qquad \underline{a}^{(i+1)} = \frac{\underline{x}^{(i+1)}}{\left\|\underline{x}^{(i+1)}\right\|}$$

$$y_v^{(i+1)} = \sum_{w \in d^+(v)} a_w^{(i+1)} \qquad \underline{h}^{(i+1)} = \frac{\underline{y}^{(i+1)}}{\left\|\underline{y}^{(i+1)}\right\|}$$

By introducing the notation A_q for the adjacency matrix of G_q, it can be easily derived that $\underline{a}^{(i+1)}$ (and $\underline{h}^{(i+1)}$)is obtained from $A_q A_q^T \underline{a}^{(i)}$ (and $A_q^T A_q \underline{h}^{(i)}$) by normalization. Therefore the above algorithm is equivalent to the power iteration method [16]. Let λ_1 and λ_2 denote the largest and second largest eigenvalues of $A_q^T A_q$. Furthermore $\underline{u}$ and $\underline{v}$ denote the principal eigenvectors of $A_q A_q^T$ and $A_q^T A_q$ with $\|u\| = \|v\| = 1$. By the convergence properties of the power iteration method we obtain the following theorem.

Theorem 7 *If $\lambda_1 > \lambda_2$, then the vectors of approximated hub and authority scores converge to the principal eigenvectors $\underline{u}$ and $\underline{v}$:*

$$\lim_{i \to \infty} \underline{a}^{(i)} = \underline{u}, \qquad \lim_{i \to \infty} \underline{h}^{(i)} = \underline{v}.$$

In case of large subgraphs of the web graph the condition of this theorem is likely to hold. We mention that in practical applications the set V_q is not equal to the set of pages cantaining a string q. Instead, a *base set* of pages are used as V_q, which is a slightly perturbed version computed from the query set. For the details we refer to [18].

3.3 Stability of ranking algorithms

Both Page Rank and Kleinberg's algorithm were introduced in 1998, and the winner of the competition of the last five years is Page Rank implemented in Google. The difficulty of formal comparison comes from the nature of the problem: there is no objective function to measure the output of a ranking algorithm. Surprisingly Ng, Zheng and Jordan [25] found a formal way of analyzation and pointed out a significant difference between the two ranking algorithms. We summarize their results in the following section.

Suppose that we have a ranking algorithm $\Re$, which outputs the ranking $\underline{x} = \Re(G)$ on a graph G. Then G' is obtained from G by a perturbation, i.e., the link content is changed slightly. The algorithm $\Re$ is *stable*, if $\underline{x}' = \Re(G')$ and $\underline{x}$ are similar in the sense that the $\|\underline{x}' - \underline{x}\|$ value is small with some vector norm. The stability is required, since the web graph is treated as an instance of some random process, so any of the above hypothesises holds only statistically. Thus the output of an unstable algorithm would largely depend on the random noise.

The following perturbation result holds for any symmetric matrix S with $\lambda_1 > \lambda_2$ largest and second largest eigenvalues and normalized principal eigenvector $\underline{u}$. The eigengap of S is denoted by $\delta = \lambda_1 - \lambda_2$, and $\|X\|_F$ refers to the Frobenius-norm of the matrix X.

Theorem 8 *There exists a perturbed symmetric matrix $\tilde{S}$ with normalized principal eigenvector $\underline{\tilde{u}}$ such that*

$$\|S - \tilde{S}\|_F \leq 2\delta \qquad\qquad \|\underline{u} - \underline{\tilde{u}}\| = \sqrt{2}.$$

While omitting the details, we mention that the above statement can be derivated from the orthogonal diagonalization of S [25]. In case of Kleinberg's algorithm $S = A_q A_q^T$, where A_q is the adjacency matrix of the graph induced by a set of pages sharing some query phrase q, and it may have various structures depending on q. We depict one example on Fig. 3, when a small perturbation causes large variance in the hub and authority scores. Indeed the pages sharing a word in common may form more than one community, especially if the query string is some polysemantic word. For instance the phrase "windows gates" occurs either in architectural sites or in political issues about Microsoft. The structure of G_q will be similar to the one depicted on the left of Fig. 3, as it will contain two disjoint communities. Then the matrix $A_q A_q^T$ will have two blocks in the diagonal. The eigenvalues corresponding to the blocks may be similar yielding to a small eigengap δ. To sum it up, the Hub-Authority ranking is unstable for some query strings.

Figure 3: The hub (authority) scores are depicted next to the lower (upper) nodes. The scores are significantly different for the left and right graphs, though they differ only in the dashed edge.

Recall that Page Rank algorithm can be computed on the largest strongly connected component of the web graph. Now $G(V, E)$ denotes an arbitrary strongly connected directed graph. The Page Rank on G with $0 < \epsilon < 1$ is denoted by r_v, $v \in V$ and the vector with these values is $\underline{r}$.

Theorem 9 (Ng et al., 2001) *Suppose that the perturbed graph $\tilde{G}(V, \tilde{E})$ is obtained from G by adding or deleting arbitrary number of edges starting from any of the nodes $v_1, v_2, \ldots, v_k$, such that $\tilde{G}$ remains strongly connected. Then the following inequality holds for the Page Rank $\underline{\tilde{r}}$ computed on $\tilde{G}$ with the same ϵ:*

$$\|\underline{r} - \underline{\tilde{r}}\|_1 \leq \frac{2\sum_{i=1}^{k} r_{v_i}}{\epsilon}.$$

The above perturbation theorem meets the requirements of the WWW application. As todays' crawlers are not fast enough in downloading, the database may contain earlier versions of some web pages. Therefore the links starting from the nodes corresponding to the old web pages may have been changed. If the sum of the rank of these pages is small enough, the computed Page Rank is not biased much according the theorem. Another consequence is that a set of low-ranked users cannot force the Page Rank algorithm to raise their ranks by manipulating their own links.

The details of the proof the theorem can be found in [25], but we give the intuition behind. The probability r_v can be defined in a slightly different way. Suppose that we choose a node $u \in V$ uniformly random, then perform a random walk starting from u with length $k \geq 0$, where k is an exponentially distributed random variable with parameter ϵ. The Page Rank r_v is equal to the probability of arriving at node v by this walk. Therefore, if we change the outgoing links of nodes which are unlikely to visit, then the resulting distribution $\underline{\tilde{r}}$ does not deviate much.

4 Finding similar pages

The last application we discuss in detail was not designed as a graph algorithm, but it can be easily adopted for the web graph. Soon after Kleinberg's iterative procedure was published, many researchers realized it is a special case of Latent Semantic Indexing (LSI) discussed in this section. For a more detailed survey containing the theoretical results about LSI see [27].

Suppose we are given a set of textual documents and we aim at defining a similarity measure, which can be evaluated for any pairs of documents effectively. The *word-document matrix* A with dimension $m \times n$ is assigned to the problem, where m is the number of words occurring in any of the documents and n is the number of documents. The entry $A_{i,j}$ is equal to the number of occurrences of the i^{th} word in the j^{th} document. (In practical applications $A_{i,j}$ is some monotone function of the number of occurrences.) According to the underlying concept, the similarity can be read from co-occurring words. Let a_s and a_t denote the columns of A corresponding to documents s and t, then the similarity is defined by

$$\frac{a_s^T a_t}{\|a_s\|_2 \|a_t\|_2},$$

which is equivalent to the cosine of the angle of the vectors a_s and a_t, so it is referred to as the *cosine measure*. The drawback of such a similarity is that for long documents the evaluation may be time-consuming and the polysemantic words can deviate the result.

The key idea of LSI [12] is to transform the columns of A to a low-dimensional *semantic space* in which the similarity of pages is more related to spatial closeness than in the original m-dimensional *term space*. LSI proposes the orthogonal projection of the vectors into the subspace spanned by the first k left singular vectors of A yielding to the matrix A_k. The constant k is the dimension of the semantic space and it is set in the range of $100 - 200$. Recall the linear algebra fact that A_k minimizes the error $\|A'_k - A\|_F$ with respect to $\text{rank}(A') \leq k$ [16]. Then the columns of A_k can be replaced by k dimensional vectors, which enables more efficient evaluation of the cosine measure.

Some early experimental results show that LSI copes with the problem of polysemantic words, and filters such co-occurrences effectively by applying the cosine measure on the columns of A_k. Papadimitriou et al. [26] analyzed LSI formally for special types of matrices, and their results was generalized by Azar et al. [5] by proving the LSI theorem. They assume that the similarity of a set of documents can be represented in a low-dimensional semantic space $\mathcal{S}$ and the word-document matrix A arises from the low-dimensional representational by a random perturbation. The LSI theorem essentially states that the similarity of documents in $\mathcal{S}$ can be approximated by the cosine measure applied on the columns of A_k rather than the original A, if the perturbation was small in compared with the singular gap $\sigma_k - \sigma_{k+1}$.

Back to the web algorithms one may replace the matrix A above by the transposed adjacency

matrix of a subgraph of the web graph induced by a set of web pages. Then LSI will define a similarity function based on the linkage information [**?**].

Finally, we mention that hub-authority ranking is special case of LSI. By applying the LSI projection with $r = 1$ to A_q, we obtain the matrix $A_{q,1}$ with rank one. From Theorem **??** and the properties of singular value decomposition, it is easy to derive that $A_{q,1} = {\lambda_1}^2 \underline{h}\underline{a}^T$ for the hub and authority vectors $\underline{h}$ and $\underline{a}$. To employ this fact Ng et al. [24] were able to modify hub authority algorithm by computing the hub and authority scores as a function of the first k singular vectors, where $k > 1$. Such a ranking can overcome the instability of Kleinberg's algorithm stated in 8. theorem as a consequence of LSI theorem. Another direction of current research is to combine the textual information and the linkage information for ranking algorithms by concatenating the adjacency and word-document matrices corresponding to a set of pages [1].

5 Conclusions

So far we have discussed the most well known applications of web algorithms for enumerating the web communities, ranking the pages and defining similarity function. Each algorithm was based on some explanation of the evolution of the web graph. The Page Rank algorithm simply assumes that a short random walk will find popular pages. For web communities, hub-authority algorithm and LSI the intuition was very similar. It is still an open question if there exists some non-linear model, with effectively computable parameters, which may be used in ranking or clustering. Another open problem comes from the application point of view. The trawling algorithm and some other clustering heuristics are able to find closely related groups of pages. However, the result can only be used, if the millions of clusters are in a hierarchical order. How can we employ the link structure of the web to automatically build a hierarchical order on the clusters?

Acknowledgement

I wish to thank András Benczúr, Katalin Friedl and Eszter Friedman for their valuable comments on this paper.

References

[1] Dimitris Achlioptas, Amos Fiat, Anna R. Karlin, and Frank McSherry. Web search via hub synthesis. In *IEEE Symposium on Foundations of Computer Science*, pages 500–509, 2001.

[2] William Aiello, Fan Chung, and Linyuan Lu. A random graph model for massive graphs. In *ACM Symposium on Theory of Computing*, pages 171–180, 2000.

[3] D. Aldous. Random walks on finite groups and rapidly mixing markov chains. In *Seminaire de Probabilites XVII, 1981/82, Springer Lecture Notes in Mathematics 986, pp. 243–297*, 1981.

[4] Arvind Arasu, Junghoo Cho, Hector Garcia-Molina, Andreas Paepcke, and Sriram Raghavan. Searching the web. *ACM Transactions on Internet Technology (TOIT)*, 1(1):2–43, August 2001.

[5] Yossi Azar, Amos Fiat, Anna R. Karlin, Frank McSherry, and Jared Saia. Spectral analysis of data. In *ACM Symposium on Theory of Computing*, pages 619–626, 2001.

[6] Albert-László Barabási, Réka Albert, and Hawoong Jeong. Scale-free characteristics of random networks: the topology of the word-wide web. *Physica A*, 281:69–77, 2000.

[7] Alan Borodin, Gareth O. Roberts, Jeffrey S. Rosenthal, and Panayiotis Tsaparas. Finding authorities and hubs from link structures on the world wide web. In *Proceedings of the 10th World Wide Web Conference (WWW)*, pages 415–429, 2001.

[8] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107–117, 1998.

[9] Junghoo Cho and Hector Garcia-Molina. Synchronizing a database to improve freshness. In *Proceedings of the International Conference on Management of Data*, pages 117–128, 2000.

[10] E. G. Coffman, Zhen Liu, and Richard R. Weber. Optimal robot scheduling for web search engines. Technical Report RR-3317, INRIA, 1997.

[11] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[12] Stephen Dill, S. Ravi Kumar, Kevin S. McCurley, Sridhar Rajagopalan, D. Sivakumar, and Andrew Tomkins. Self-similarity in the web. In *The VLDB Journal*, pages 69–78, 2001.

[13] P. Erdős and A. Rényi. On the evolution of random graph. *Math. Inst.*, 1960.

[14] G.H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1983.

[15] Jon Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[16] Jon Kleinberg. The Small-World Phenomenon: An Algorithmic Perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, 2000.

[17] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, D. Sivakumar, Andrew Tomkins, and Eli Upfal. The web as a graph. In *Proceedings of the nineteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–10. ACM Press, 2000.

[18] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. Trawling the Web for emerging cyber-communities. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(11–16):1481–1493, 1999.

[19] Ravi Kumar, Prabharkar Raghavan, Sridhar Rajagopalan, D. Sivakumar, Tomkins, and Eli Upfal. Stochastic models for the web graph. In *FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)*, 2000.

[20] Andrew Y. Ng, Alice X. Zheng, and M. Jordan. Stable algorithms for link analysis. In *Proc. 24th Annual Intl. ACM SIGIR Conference*, 2001.

[21] Andrew Y. Ng, Alice X. Zheng, and Michael I. Jordan. Link analysis, eigenvectors and stability. In *Proc. Int. Joint Conf. Artificial Intelligence, Seattle, WA*, August 2001.

[22] Christos H. Papadimitriou, Hisao Tamaki, Prabhakar Raghavan, and Santosh Vempala. Latent semantic indexing: A probabilistic analysis. pages 159–168, 1998.

[23] Jared Saia. Spectral analysis for information retrieval and data mining, 2000.

[24] The Google search engine. Commercial search engine founded by the originators of pagerank. located at `http://www.google.com`.

[25] D. J. Watts and S. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.