Random Web Crawls

Toufik Bennouas
Criteo R&D
6, boulevard Saint Denis
75010 Paris, France
t.bennouas@criteo.com

Fabien de Montgolfier
LIAFA - Université Paris 7
2, place Jussieu, Case 7014,
75251 Paris, France
fm@liafa.jussieu.fr

ABSTRACT

This paper proposes a random Web crawl model. A *Web crawl* is a (biased and partial) image of the Web. This paper deals with the hyperlink structure, i.e. a Web crawl is a graph, whose vertices are the pages and whose edges are the hypertextual links. Of course a Web crawl has a very special structure; we recall some known results about it. We then propose a model generating similar structures. Our model simply simulates a crawling, i.e. builds and crawls the graph at the same time. The graphs generated have lot of known properties of Web crawls. Our model is simpler than most *random Web graph* models, but captures the sames properties. Notice that it models the *crawling* process instead of the *page writing* process of Web graph models.

Categories and Subject Descriptors

I.6.m [**Simulation and Modeling**]: Miscellaneous

General Terms

Theory

Keywords

web graph, crawling, crawl order, model, hyperlink structure

1. INTRODUCTION

The Web is a fascinating object, very extensively studied since a few years. Among the many research problems it opens, an interesting one is the *topological issues*, i.e. describing the shape of the Web [11]. Understanding the hyperlink structure allowed the design of the most powerful search engines like Google, famous because it uses the PageRank algorithm from Brin and Page [21], and the design of other ranking methods like HITS from Kleinberg [15], and cyber-communities detection [19, 14], and many other applications. However, we only know parts of the Web. The *crawlers* are software that automatically browse the Web and cache the "most relevant" information, especially the documents URL and their hyperlinks. This is recursively performed, the analysis of crawled pages allowing to get new valid URLs. But bandwidth limitations, HTML errors, unreferenced pages, removed or modified pages, and the existence of dynamic pages (generated from requests in URL or from a session mechanism) make very hard, if not impossible, to output "the" Web: crawlers instead produce partial and biased images. This is not even a snapshot of the Web, since the crawling takes a long time while the pages are changing rapidly. From theses observations of the Web, one can try to infer the properties of the "real" Web, the underlying object, but it is hard since the biases are not well known. So we can not say that the properties of *The Web graph* are known, but only that some properties of *Web crawls* are known. The object we deal with in this paper are therefore the Web crawls, and not the Web itself.

In Section 2 we recall some of the most commonly admitted properties of Web crawls. In order to explain *why* the graphs have these properties, many hypotheses from sociology or computer sciences fields have been proposed. Many models describe random graphs and some of them (see Section 3) are specifically designed to model Web Graphs, i.e. the hyperlink structure of the Web. The authors usually compare measurements on their random graphs with existing crawls of the Web and conclude how accurate their model is [4, 6, 17, 18, 7, 2, 9, 10, 17, 18].

We also propose a model generating random Web crawls, and show that our random crawls share a lot of properties with real crawls. But our approach here is quite different from the *random Web graph* models. Indeed we do not try to model the *page writing* process, using sociological assumptions about how the people link their own pages to the existing ones. We try to model the *pages crawling* process itself instead. So we do not suppose that pages are linked preferentially to well-known pages, nor that the links of a page are likely a copy of the links of another pages, or such kind of things. We instead postulates only two things about a crawl:

- The in- and out-degree of the pages follow Zipf laws (aka power laws), and
- the graph is output by a crawler

We present our model in details in Section 4. In Section 5 we show that our random crawls have most of the main crawl properties presented in Section 2.

2. WEB CRAWL PROPERTIES

Web crawls can be quite large objects (for instance Google currently claims more than 8 billion pages in database) but are very sparse graphs, since the average degree is around 7 links per page [8]. Here crawls are directed graphs. They are not necessarily connected, since a connecting page may

WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.
ACM 978-1-59593-654-7/07/0005.

be removed from the Web and then deleted from the crawl. They have very few *sources* (pages with no incoming links, either submitted by peoples or unlinked after a while) and a lot of *sinks* (pages not crawled yet or with no external hyperlink).

2.1 Connectivity and Clustering

Small-World graphs, defined by Watts and Strogatz [24] and studied by many authors since, are graphs that fulfill the following two properties:

1. the *characteristic path length* (average distance between two vertices) is small: $O(\log n)$ or $O(\log \log n)$
2. the clustering coefficient (probability that two vertices sharing a common neighbor are linked) is high: $O(1)$.

This definition can be applied to directed graphs when omitting arc direction. The Web (in fact the crawls) characteristic path length seems small (about 16 clicks [8] or 19 [3]), consistant with the $O(\log n)$ axiom. Its clustering coefficient is high. Existing computations of its exact value differ, but it is admitted that it is greater than 0.1, while random graphs (Erdös-Rényi, see Section 3.1) with the same average degree have clustering coefficient $p \simeq 0$.

Crawls diameter (maximum distance between two pages) is potentially infinite because a dynamic page labeled by n in URL may refer to a dynamic page labeled by $n+1$, but since Web crawlers usually perform BFS (see Section 4.1) the diameter of crawls may be actually small.

2.2 Degree distribution

Zipf laws (a.k.a *power laws*) are probability laws such that

$$\log(Prob(X = d)) = \alpha - \lambda \log(d)$$

If the in-degree (respectively out-degree) distribution of a graph follows a Zipf law, $Prob(X = d)$ is the probability for a vertex to have in- (resp. out-) degree d. In other words, the number of vertices with degree d is $k.d^{-\lambda}$ (k depends on the number of vertices n). A graph class such that the degree of almost all graphs follow a Zipf law is called *scale-free* because some parameters like λ are scale invariant. Scale-free graphs have been extensively studied [4, 6, 17, 18]. Many graphs modeling social networks, interaction between objects (proteins, peoples, neurons...) or other network properties seem to have the scale-free property.

For Web crawls, a measure from Broder & al. [8] on a 200 000 000 pages crawl show that the in and out-degrees follow Zipf law. The exponents are $\lambda_{in} = 2.1$ for in-degree and $\lambda_{out} = 2.72$ for out-degree.

2.3 Strongly connected components and the Bow Tie structure

According to Broder, Kumar et al [8] the Web has a Bow Tie structure: a quarter of the page are in a Giant Strongly Connected Component (GSCC), a quarter are the "*in*" page, leading to the GSCC but not linked from there, another quarter are the "*out*" pages reachable from the GSCC but not linking to it, and the last quarter is not related to the GSCC. This famous assertion was reported even by Nature [23] but, since four years, an increasing number of people suspects it is a crawling artifact. According to the same survey, the distribution of the size of strongly connected components follows a Zipf law with exponent roughly 2.5.

2.4 Cores

Another well-known property of crawls is the existence of *cores*. A core is a dense directed bipartite subgraph, consisting in many *hubs* pages (or *fans*) pointing many *authorities*. It is supposed [19, 16] that such cores are the central structure of *cybercommunities*, set of pages about the same topics. The authorities are the most relevant pages, but they do not necessarily point one each other (because competition, for instance) but the hubs list most of them. Starting from this assumption, HITS algorithm [15] ranks the pages containing a given keyword according to a *hub factor* and an *authority factor*. Kumar et al. [19, 18] enumerate over 200,000 bipartite cores from a 200,000,000 pages crawl of the Web. Cores sizes (counting hubs, authorities, or both) follow Zipf laws of exponent between 1.09 and 1.4.

2.5 Spectral properties and PageRank factor

Another ranking method, the most popular since it does not depends on given keywords, is Google's PageRank factor [21]. It is an accessibility measure of the page. Briefly, the PageRank of a page is the probability for a random surfer to be present on this page after a very long surf. It can be computed by basic linear algebra algorithms. PageRank distribution also follows a Zipf law with the same exponent as the in-degree distribution [22]. Pages with high PageRank are very visible, since they are effectively popular on the Web and are linked from other pages with high PageRank. A crawler therefore easily finds them [5, 20] while it may miss low-ranked pages. This is indeed a useful bias for search engine crawlers!

3. RANDOM GRAPHS MODELS

3.1 Basic models: Erdös-Rényi

For a long time the most used random graph model was Erdös-Rényi model [13]. The random graph depends on two parameters, the number of vertices n and the probability p for two vertices to be linked. The existence of each edge is a random variable independent from others. For suitable values ($p = d/n$), E.-R. graphs indeed have characteristic path length of $O(\log n)$ but very small clustering ($p = o(1)$) and degree distribution following a Poisson law and not a Zipf law. Therefore they do not accurately describe the crawls. Other models have then be proposed where attachment is not independent.

3.2 Incremental generation models

Most random Web graph models [4, 6, 17, 18, 7] propose an incremental construction of the graph. When the existence of a link is probed, it depends on the existing links. That process models the creation of the Web across time. In some models all the link going from a page are inserted at once, and in other ones it is incremental.

3.3 Preferential Attachment models

The first evolving graph model (BA) was given by Barabasi and Albert [4]. The main idea is that new nodes are more likely to join to existing nodes with high degrees. This model is now referred to as an example of a *preferential attachment model*. They concluded that the model generates graphs whose in-degree distribution follows a Zipf law with exponent $\lambda = 3$.

Another preferential attachment model, called the *Linearized Chord Diagram* (LCD), was given in [6]. In this model a new vertex is created at each step, and connects to existing vertices with a constant number of edges. A vertex is selected as the end-point of the an edge with probability proportional to its in-degree, with an appropriate normalization factor. In-degrees follow a Zipf law with exponent roughly 2 when out-degrees are 7 (constant).

In the *ACL* [2] model, each vertex is associated a in-weight (respectively out-weight) dependent of in-degree (respectively out-degree). A vertex is selected as the end-point of the an edge with probability proportional to its weight. In these models edges are added but never deleted. The *CL-del model* [9] and *CFV model* [10] incorporate in their design both the addition and deletion of nodes and edges.

3.4 Copy models

A model was proposed by [17] to explain other relevant properties of the Web, especially the great number of cores, since the ACL model generates graphs which on average contain few cores.

The *linear growth coping model* from Kumar& al. [18] postulates that a Web page author shall *copy* an existing page when writing its own, including the hyperlinks. In this model, each new page has a *master* page from which it copies a given amount of links. The master page is chosen proportionally to in-degree. Other links from the new page are then added following uniform or preferential attachment. The result is a graph with all properties of previous models, plus the existence of many cores.

These models often use many parameters needing fine tune, and sociological assumptions on how the Web pages are written. We propose a model based on a computer science assumption: the Web graphs we know are produced by crawlers. This allow us to design a simpler (it depends only on two parameters get from experiments) and very accurate model of Web crawl.

4. A WEB CRAWL MODEL

In this section, we present the crawling strategies and derive our Web crawl model from them. It aims to mimic the crawling process itself, rather than the page writing process as web graph models do.

4.1 Web crawls strategies

Let us consider a theoretical crawler. We suppose the crawler visits each page only once. The benefit is to avoid modeling the disappearance of pages or links across time, because the law it follows is still debatable (is the pages lifetime related to their popularity, or to their degree properties?) When scanning a page, the crawler gets at once the set of its outgoing links. At any time the (potentially infinite) set of valid URL is divided into

1. *Crawled*: the corresponding pages were visited and their outgoing links are known
2. *Unvisited*: a link to this URL has been found but not probed yet
3. *Erroneous*: the URL was probed but points a non-existing or non-HTML file (some search engines index them, but they do not contain URL and are not interesting for our purposes)
4. *Unknown*: the URL was never encountered

The crawling algorithm basically choose and remove from its *Unvisited* set an URL to crawl, and then adds the outgoing unprobed links of the page, if any, to the *Unvisited* set. The *crawling strategy* is the way the *Unvisited* set is managed. It may be:

- DFS (depth-first search) The strategy is FIFO and the data structure is a stack
- BFS (breadth-first search) The strategy is LIFO and the data structure is a queue
- DEG (higher degree) The most pointed URL is chosen. The data structure is a priority queue (an heap)
- RND (random) An uniform random URL is chosen

We suppose the crawled pages are ordered by their discovery date. For discussing structural properties, the *crawled* pages only are to be considered. Notice that the first three strategies can only be correctly implemented with a single computed. The most powerful crawlers are distributed on many computers and their strategy is hard to define. It is usually something between BFS and Random.

4.2 Model description

Our model shall mimic a crawler strategy. It works in two steps: first constructing the set of pages, then adding the hyperlinks.

Constructing the set of pages. Each page p has two fields: its in-degree $d_{in}(p)$ and its out-degree $d_{out}(p)$. In the first step of the crawl constructing process, we set a value to each of them. The in-degree and out-degree are set according to two independent Zipf laws. The exponent of each law is a parameter of the model, therefore our model depends on two parameters λ_{in} (for in-degree) and λ_{out} (for out-degree). These values are well known for real crawls: following [8], we have $\lambda_{in} = 2.1$ and $\lambda_{out} = 2.72$.

We shall have to chose the pages at random according to their in-degree. For solving the problem, n pages (the maximal size of the crawl) are generated and their in- and out-degrees are set. Then, a set L is created, where each page p is duplicated $d_{in}(p)$ times. The size of this set is the maximal number of hyperlinks. Each time we need to choose a page at random according to the in-degree law, we just have to remove one element from L.

Constructing the hyperlinks. Now the pages degrees are pre-set, but the graph topology is not yet defined. An algorithm, simulating a crawling, shall add the links. There are indeed four algorithms, depending on which crawling strategy shall be simulated. The generic algorithm is simply:

1. Remove a page p from the Unvisited Set and mark p as *crawled*
2. Remove the first $d_{out}(p)$ pages from L
3. Set these pages as the pages pointed by p
4. Add the unvisited ones to the Unvisited Set
5. Go to 1

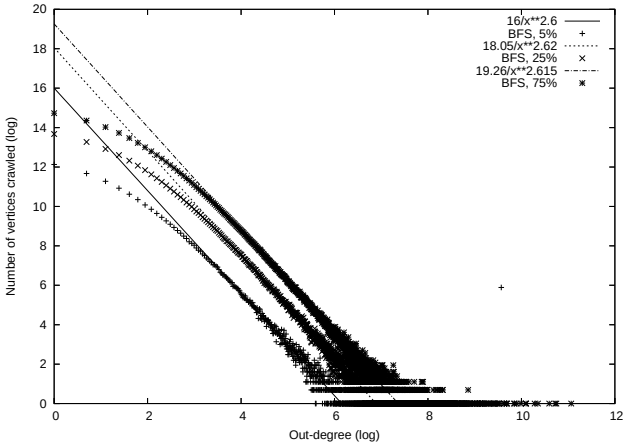

Figure 1: Out-degree distribution at three steps of a BFS

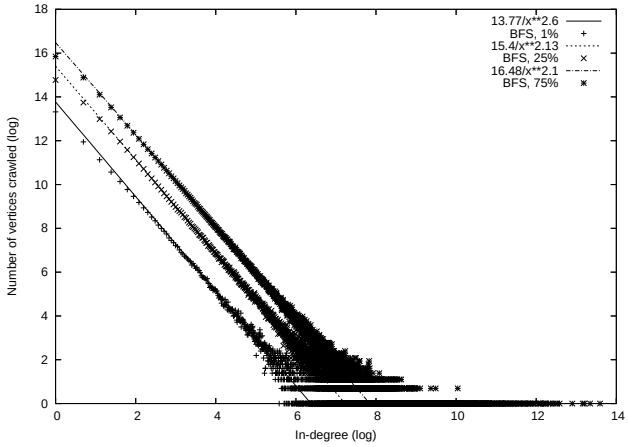

Figure 2: In-degree distribution at three steps of a BFS

The Unvisited Set is seeded with one or more pages. The way it is managed depends on which crawling strategy is simulated, i.e. which algorithm is chosen:

- For DFS algorithm, the Unvisited Set is a stack (FIFO)
- For BFS algorithm, it is a queue (LIFO)
- For DEG algorithm, it is a priority queue (heap)
- For RND algorithm, a random page is extracted from the Unvisited Set

Because the average out-degree of a page is large enough, the crawling process will not stop unless almost all pages have been crawled. The *progress* of the crawl (expressed in percent) is the fraction of crawled pages over n. As it approaches n, some weird things will occur as no more unknown pages are allowed. In our experiments (see the next section) we sometimes go up to 100% progress but results are more realistic before 30%; when the crawl can expand toward unkown pages.

Our model differs radically from preferential attachment or copy models because the neighborhood of a page is not set at writing time but at crawling time. So a page is allowed to point known or unknown pages as well.

5. RESULTS

We present here simulation results using the different strategies and showing how the measurements evolve across time. Thanks to the scale-free effect, the actual number of pages does not matter, since it is big enough. We have used several graphs of different sizes but with the same exponents $\lambda_{in} = 2.1$ and $\lambda_{out} = 2.72$ (experimental values from [8]). And unless otherwise specified, we present results from BFS, the most used crawling strategy, and simulations up to 20,000,000 crawled pages.

5.1 Degree distribution

At any step of the crawl, the actual degree distribution follows a Zipf law of the given parameters (2.1 and 2.72) with very small deviation (see Figures 1 and 2). This result is independent from the crawl strategy (BFS, etc.) It demonstrates that our generated crawls really are scale-free graphs.

5.2 Small-World properties

The distribution of path length (Figure 3) clearly follows a Gaussian law for BFS, DEG and RAND strategies. This distribution is plotted at progress 30% but it does not change a lot accros time, as shown in Figure 5. DFS produces far greater distances between vertices, and the distribution follows an unknown law (Figure 4). DFS crawls diameter is about 10% of the number of vertices! This is because DFS crawls are like long tight trees. It is why DFS is not used by real crawlers, and this paper focuses on the three other crawls strategies.

The clustering (Figure 6), computed on 500,000 pages simulation) is high and do not decrease too much as the crawl goes bigger. Our crawls definitely are small-world graphs.

5.3 Bow-tie structure?

The relative size of the four bow-tie components (SCC, IN, OUT and OTHER) are roughly the same for BFS, DEG and even RAND (but not DFS) strategies (Figure 7). When using only one seed, the size of the largest SCC converges toward two thirds of the size of the graph. These proportions thus differ from [8] crawl observations since the "in" and "others" parts are smaller. But with many seeds (it may be seen as many pages submitted to the crawler portal) the size of the "in" component is larger and can be up to one quarter of the pages. Our model replicates indeed very well genuine crawls bow-tie topology.

5.4 Cores

We used Agrawal practical algorithm [1] for cores enumeration (notice that the maximal core problem is NP-complete). Figure 10 gives the number of core of a given minimal size for a crawl up to 25 000 vertices. As shown, the number of cores is very dependent from exponents of Zipf laws, since high exponents mean sparser graphs. It means that our simulated crawls contain many core, as real crawls do.

Figure 9 shows that the number of $(4, 4)$-cores (at least four hubs and four authorities) is proportional to n and after a while stays between $n/100$ and $n/50$.

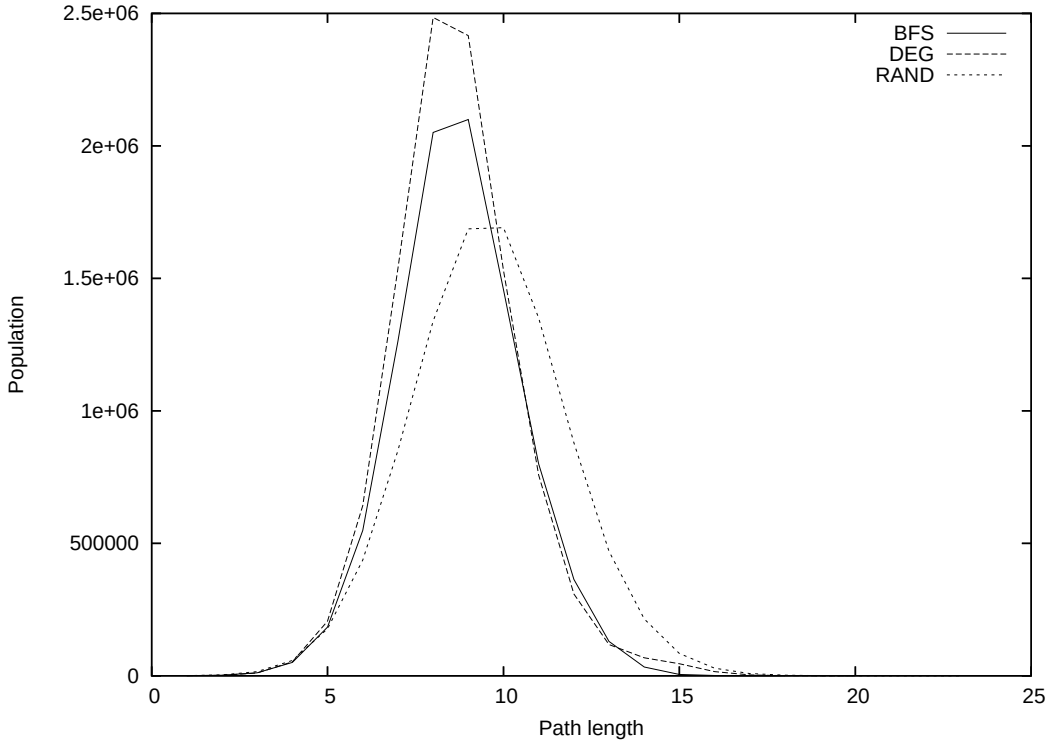

Figure 3: Distribution of path length for BFS and DEG and RAND

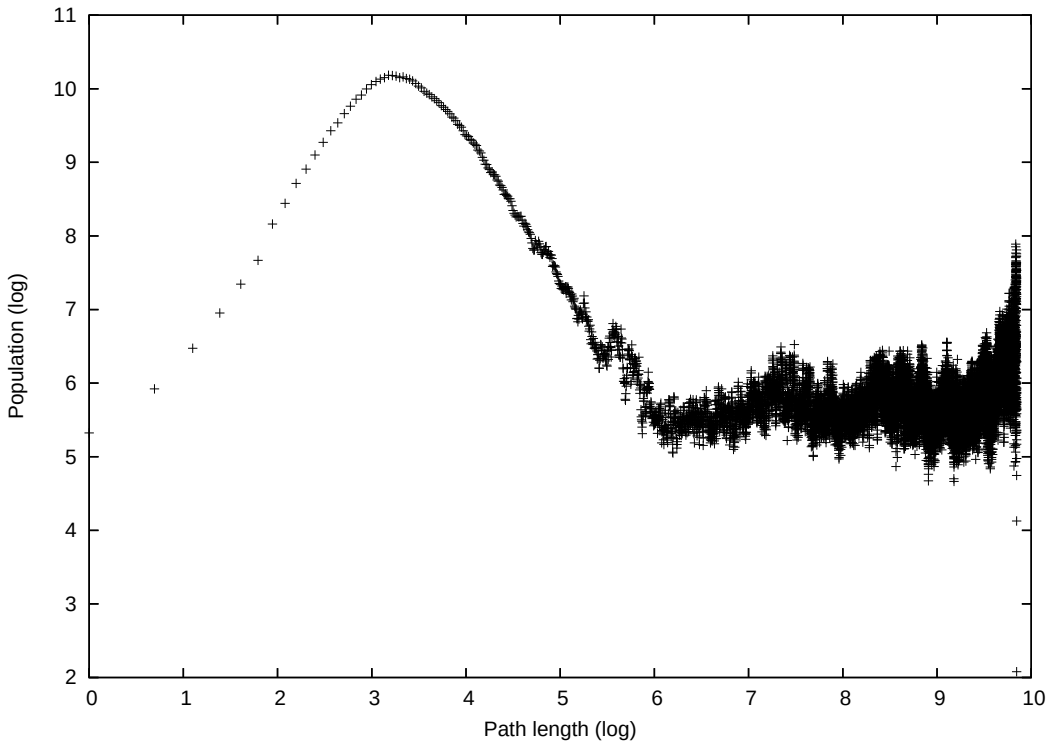

Figure 4: Distribution of path length for DFS (log/log scale)

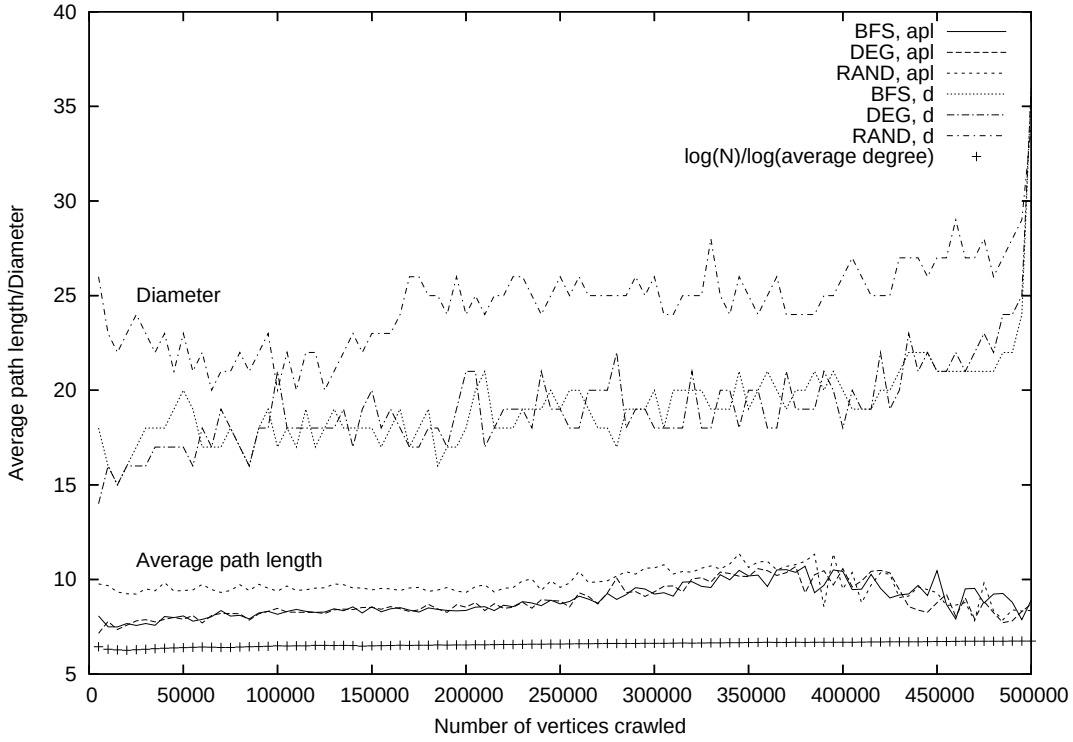

Figure 5: Evolution of diameter and average path length across time for BFS, DEG and RAND

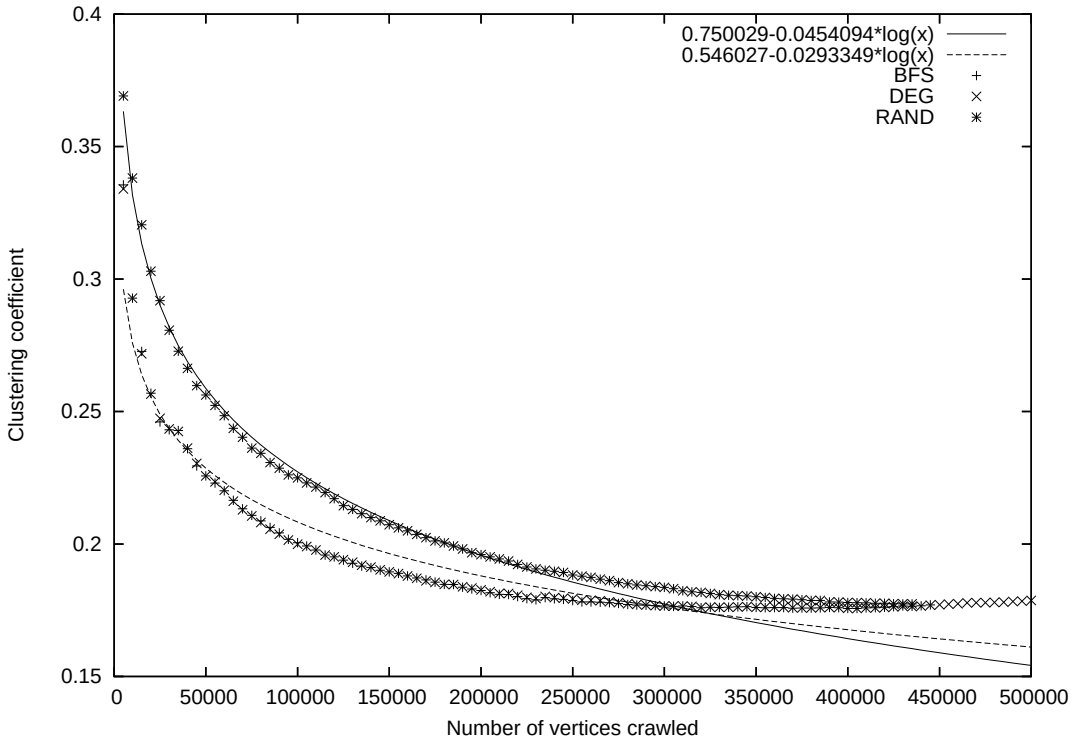

Figure 6: Evolution of clustering coefficient across time

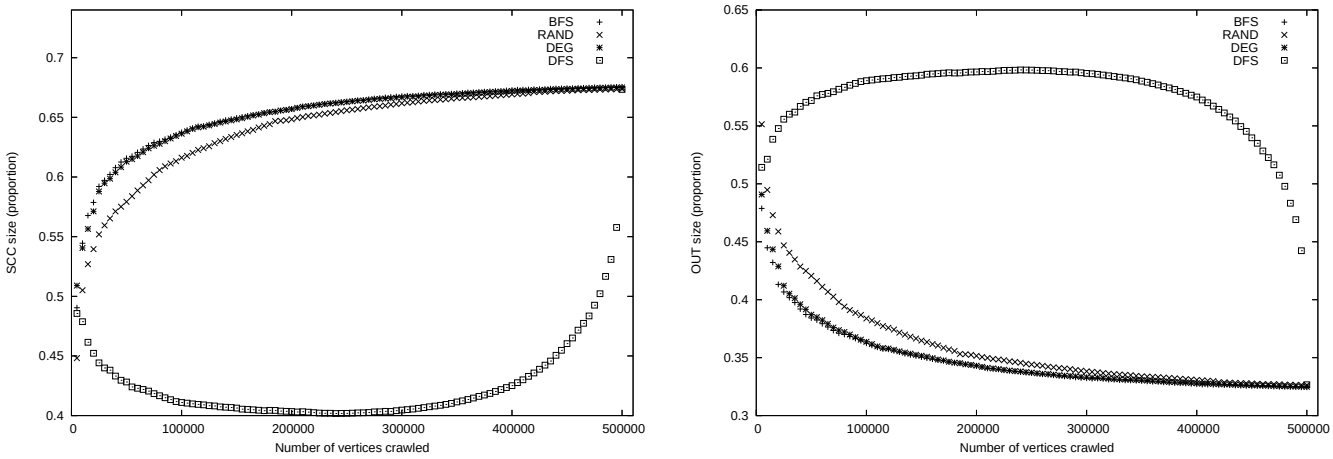

Figure 7: Evolution of the size of of the largest SCC (left) and of the OUT component (right) across time. One seed, up to 500,000 pages in the crawl

5.5 Pagerank distribution and "quality pages" crawling speed

Figure 10 shows the PageRank distribution (PageRank is normalized to 1 and logarithms are therefore negative). We have found result similar to Pandurangan *et al.* observations [22]: the distribution is a Zipf law with exponent 2.1. The crawl quickly converges to this value.

Figure 11 shows the sum of the PageRank of the crawled pages across time (the PageRank computed at the end of the crawl, so that it must vary from 0 at beginning to 1 when crawl stops). In a very few steps, BFS and DEG strategies find the very small amount of pages that contains most of the total PageRank. This property of real BFS crawlers is known since Najork and Wiener [20]. Our results can be compared to Boldi *et al* crawling strategies experiments [5].

5.6 Discovery speed and Frontier

Figure 12 shows another dynamical property: the discovery rate. It is the probability for the extremity of a link of being already crawled. It converges toward 40% for all strategies. This is an interesting scale-free property: after a while, the probability for a URL to point a new page is very high, about 60%. This "expander" property is very usefull for true crawlers. This simulation shows it does not depends only on the dynamical nature of the web, but also from the crawling process itself.

hubs	auth	cores	hubs	auth	cores
2	2	220	3	6	5
2	3	83	3	7	2
2	4	37	4	4	40
2	5	14	4	5	14
2	6	14	4	6	5
2	7	14	4	7	2
3	3	84	5	5	12
3	4	37	5	6	3
3	5	14	6	6	3

Figure 8: Number of small cores (over 1000 pages)

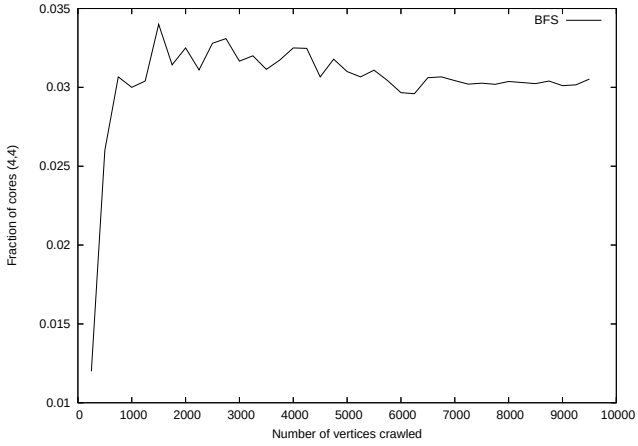

Figure 9: (4,4) Cores distribution

Figure 13 focuses on a well known topological property of crawls, that our simulations also produces, the very high number of sinks regardless of crawl size. Notice that their existence is a problem for practical PageRank computation [21]. In other words, the large "out" component of the bow-tie is very broad and short... Eiron *et al* survey the *Web frontier* ranking [12].

6. CONCLUSION

As said in Section 2, a *good* crawl model should output graphs with the following properties:

1. highly clustered
2. with a short characteristic path length
3. in- and out-degree distributions follow Zipf laws
4. with many sinks
5. such that high PageRank vertices (as computed in the final graph) are crawled early
6. with a bow tie structure

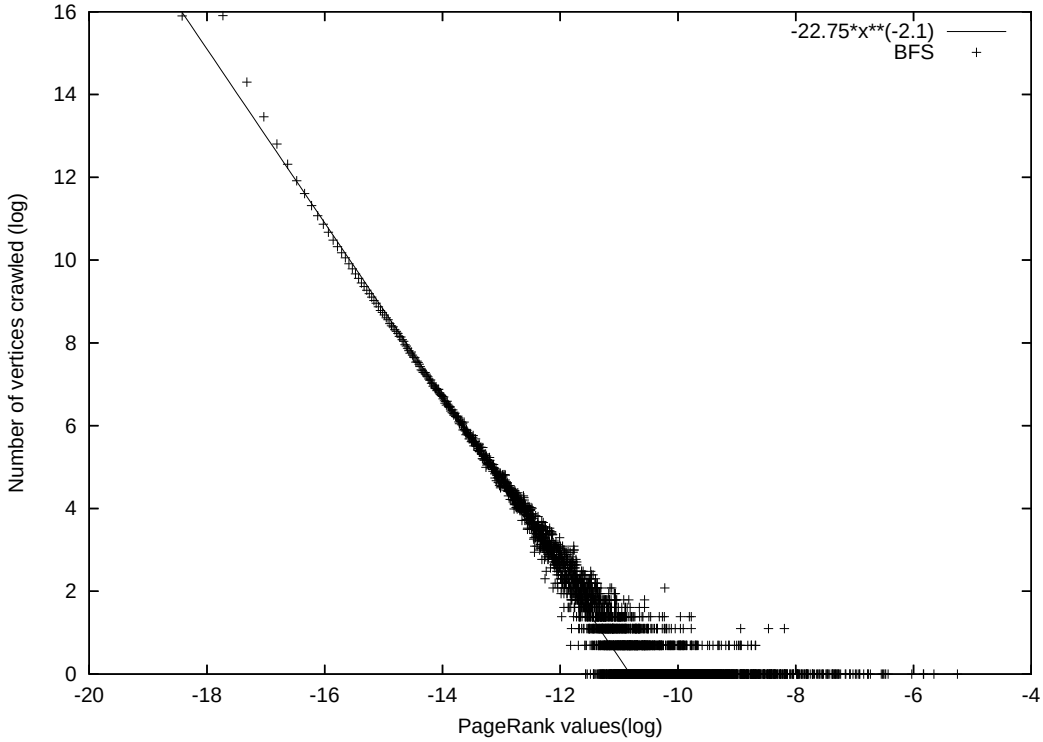

Figure 10: PageRank distribution

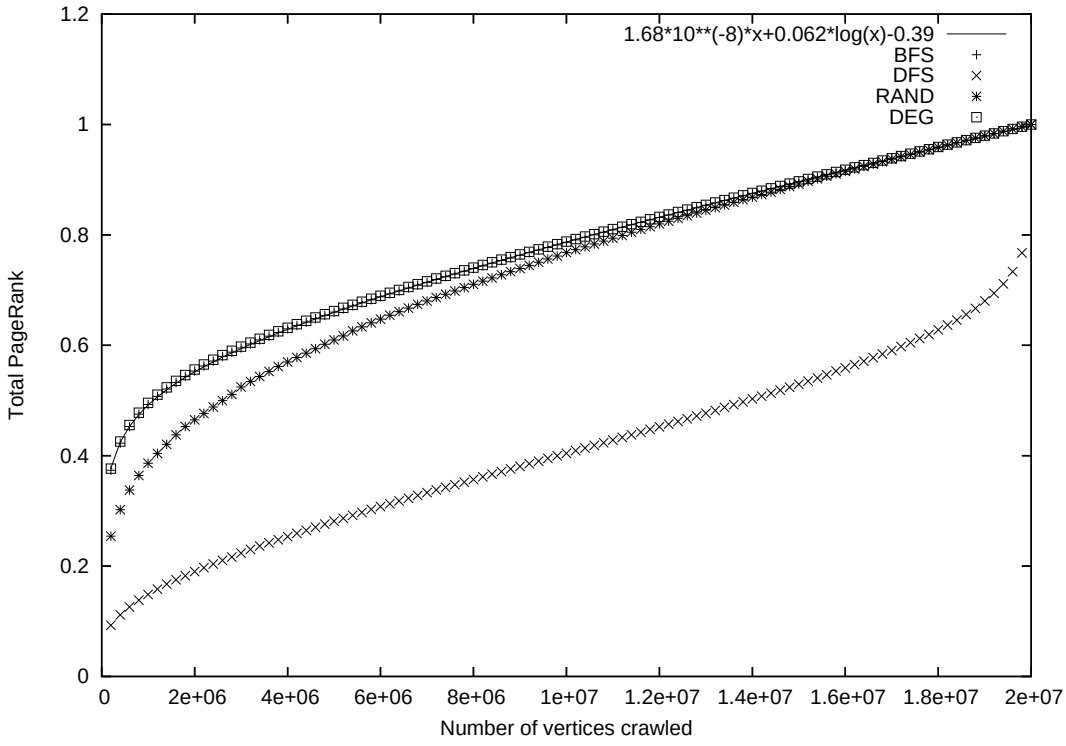

Figure 11: PageRank capture

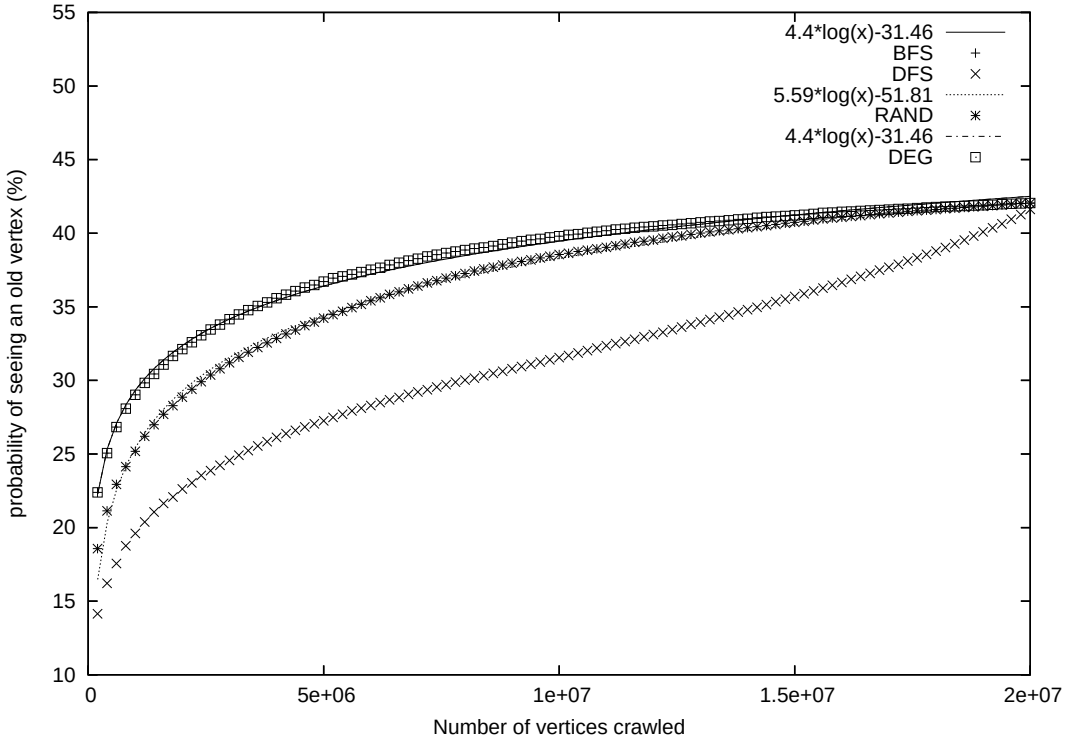

Figure 12: Evolution of the discovery rate

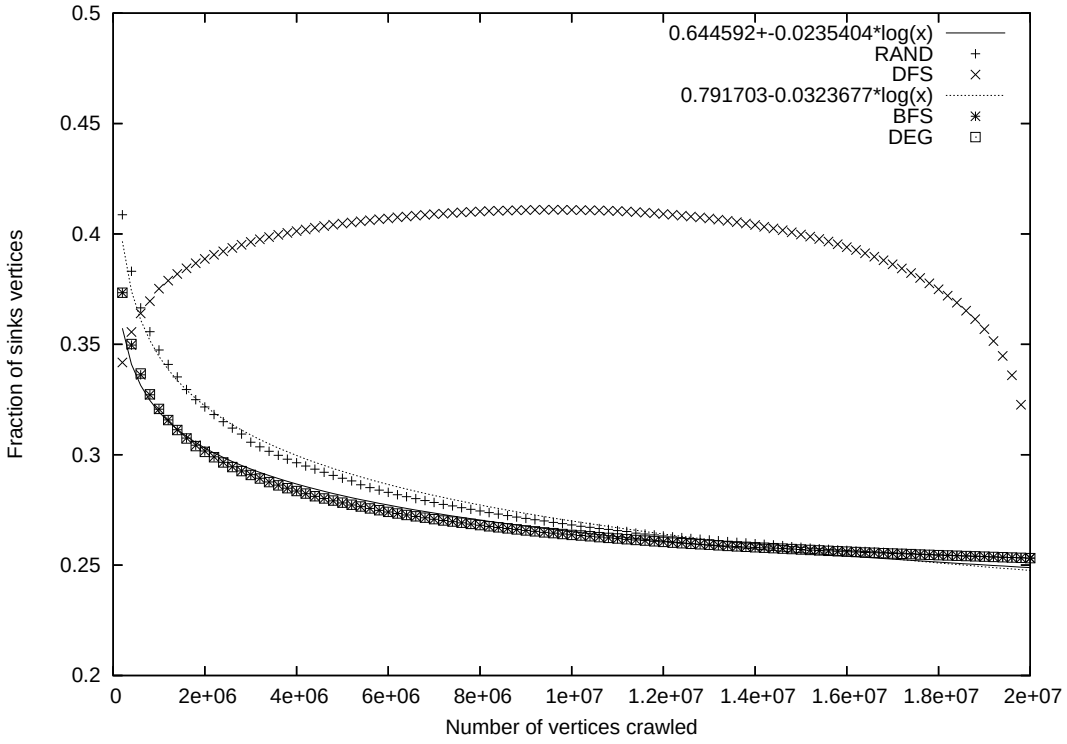

Figure 13: Evolution of the proportion of sinks (pages with no crawled successors) among crawled pages

As shown in Section 5, our model meets all these objectives. Property 3 of course is ensured by the model, but the other ones are results of the generating process. The basic assumption of degree distribution, together with the crawling strategy, is enough to mimic the properties observed in large real crawls. This is conceptually simpler than other model that also have the same properties like the Copy model [17].

The Bow Tie structure we observe differs from [8] since the largest strongly connected component is larger. But together with the other topological properties measured, it proves that we reproduce quite well the topology of real crawls with our very simple model. It is nice, because we have fewer assumption than [6] or [18]. Our approach is different from the Web graph models, that mimic the page writing strategy instead of the page crawling, but give similar result. It points out that we need more numerical or other measures on graph in order to analyze their structure.

BFS, RAND and DEG strategies are the most used in simple crawlers. We show that they produce very similar results for topological aspects. For dynamical aspects (PageRank capture for instance) BFS and DEG seems better, but are harder to implement in a real crawler. DFS is definitely bad, and for this reason is not used by crawlers. Parallel crawler use, however, more sophisticated strategies that were not modeled here.

So our *random Web crawls* model can be compared with the existing *random Web graph* models [4, 6, 17, 18, 7, 2, 9, 10, 17, 18]. But unlike them, it is not based on sociological assumptions about how the pages are written, but on an assumption on the law followed by the pages degrees and, for the structural properties, on only one assumption that the graph is output by a crawler. The design is then quite different from the design of the random Web graph models, but the results are the same.

We can interpret this conclusion in a pessimistic way: it is hard to tell what are the biases of the crawling. Indeed we have not supposed that the Web graph has any other specific property than degrees following a Zipf law, and yet our random crawls have all properties of real crawls. This means that one can crawl anything following a Zipf law, not only the Web, and output crawls with the specific properties of the Web crawls. So the comparison of the result of a Web graph model with real crawls could be not enough to assert that the model captures properties of the Web.

7. REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB '94*, pages 487–499. Morgan Kaufmann Publishers Inc., 1994.

[2] W. Aiello, F. Chung, and L. Lu. A random graph model for massive graphs. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 171–180, 2000.

[3] R. Albert, H. Jeong, and A.-L. Barabasi. Diameter of the world-wide web. *Nature*, 401:130–131, September 1999.

[4] A.-L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[5] P. Boldi, M. Santini, and S. Vigna. Do your worst to make the best: Paradoxical effects in pagerank incremental computations. In *Workshop on Algorithms and Models for the Web-Graph*, 2004.

[6] B. Bollobás, O. Riordan, J. Spencer, and G. Tusnády. The degree sequence of a scale-free random graph process. *Random Structures and Algorithms*, 18(3):279–290, May 2001.

[7] A. Bonato. A survey of models of the web graph. In *The Proceedings of Combinatorial and Algorithmic Aspects of Networking*, August 2004.

[8] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, J. Wiener, A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Graph structure of the web. In *Proceedings of the ninth international conference on World Wide Web*, pages 309–320. Foretec Seminars, Inc., 2000.

[9] F. Chung and L. Lu. Coupling on-line and off-line analyses for random power graphs. *Internet Mathematics*, 1(4):409–461, 2003.

[10] A. F. Colin Cooper and J. Vera. Random deletion in a scale free random graph process. *Internet Mathematics*, 1(4):463–483, 2003.

[11] K. Efe, V. Raghavan, C. H. Chu, A. L. Broadwater, L. Bolelli, and S. Ertekin. The shape of the Web and its implications for searching the Web, 31 –6 2000.

[12] N. Eiron, K. S. McCurley, and J. A. Tomlin. Ranking the web frontier. In *WWW conference*, 2004.

[13] P. Erdös and A. Rényi. On random graphs. In *Publicationes of the Mathematicae*, volume 6, 1959.

[14] H. Ino, M. Kudo, and A. Nakamura. Partitioning of web graphs by community topology. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 661–669, 2005.

[15] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[16] J. M. Kleinberg. Hubs, authorities, and communities. *ACM Computing Surveys (CSUR)*, 31(4es):5, 1999.

[17] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal. Stochastic models for the web graph. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 57. IEEE Computer Society, 2000.

[18] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Extracting large-scale knowledge bases from the web. In *VLDB*, pages 639–650, 1999.

[19] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. In *WWW conference*, 1999.

[20] M. Najork and J. L. Wiener. Breadth-first crawling yields high-quality pages. In *Proceedings International WWW Conference(10), Hong-Kong*, 2001.

[21] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Computer Science Department, Stanford University, 1998.

[22] G. Pandurangan, P. Raghavan, and E. Upfal. Using pagerank to characterize web structure. In *8th Annual International Conference on Computing and Combinatorics*, pages 330–339, 2002.

[23] Nature. 405:113, 11 May 2000.

[24] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(1–7):440–442, 1998.