Breadth-First Search Crawling Yields High-Quality Pages

Marc Najork
Compaq Systems Research Center
130 Lytton Avenue
Palo Alto, CA 94301, USA
marc.najork@compaq.com

Janet L. Wiener
Compaq Systems Research Center
130 Lytton Avenue
Palo Alto, CA 94301, USA
janet.wiener@compaq.com

ABSTRACT

This paper examines the average page quality over time of pages downloaded during a web crawl of 328 million unique pages. We use the connectivity-based metric PageRank to measure the quality of a page. We show that traversing the web graph in breadth-first search order is a good crawling strategy, as it tends to discover high-quality pages early on in the crawl.

Keywords

Crawling, crawl order, breadth-first search, page quality metric, PageRank

1. INTRODUCTION

According to a study released in October 2000, the directly accessible "surface web" consists of about 2.5 billion pages, while the "deep web" (dynamically generated web pages) consists of about 550 billion pages, 95% of which are publicly accessible [9].

By comparison, the Google index released in June 2000 contained 560 million full-text-indexed pages [5]. In other words, Google — which, according to a recent measurement [6], has the greatest coverage of all search engines — covers only about 0.1% of the publicly accessible web, and the other major search engines do even worse.

Increasing the coverage of existing search engines by three orders of magnitude would pose a number of technical challenges, both with respect to their ability to discover, download, and index web pages, as well as their ability to serve queries against an index of that size. (For query engines based on inverted lists, the cost of serving a query is linear to the size of the index.) Therefore, search engines should attempt to download the *best pages* and include (only) them in their index.

Cho, Garcia-Molina, and Page [4] suggested using connectivity-based document quality metrics to direct a crawler towards high-quality pages. They performed a series of crawls over 179,000 pages in the *stanford.edu* domain and used different ordering metrics — breadth-first, backlink count, PageRank [2], and random — to direct the different crawls. Under the breath-first ordering, pages are crawled in the order they are discovered. Under the backlink ordering, the pages with the highest number of known links to them are crawled first. Under the PageRank ordering, pages with the highest PageRank (a page quality metric described below) are crawled first. Under the random ordering, the crawler selects the next page to download at random from the set of uncrawled pages. (For repeatability, these crawls were "virtual"; that is, they were performed over a cached copy of these 179,000 pages.) Cho et al. evaluated the effectiveness of each ordering metric by examining how fast it led the crawler to all the "hot" pages. In this context, a "hot" page is a page with either a high number of links pointing to it, or a page with a high PageRank. They found that using the PageRank metric to direct a crawler works extremely well. However, they also discovered that performing the crawl in breadth-first order works almost as well, in particular if "hot" pages are defined to be pages with high PageRank.

This paper extends the results of Cho et al. regarding the effectiveness of crawling in breadth-first search order, using a much larger and more diverse data set. While Cho's work was based on a crawl of 179,000 pages from the *stanford.edu* domain, we performed a crawl of 328 million pages over the entire web, covering more than 7 million distinct hosts. We use connectivity-based page quality metrics, namely Brin and Page's PageRank and variations of it, to measure the quality of downloaded pages over the life of the crawl.

We find that not only does breadth-first search download the hot pages first, but also that the average quality of the pages decreased over the duration of the crawl. We also suggest that our crawler's modifications to strict breadth-first search — made to increase the overall download rate and to avoid overloading any given web server — enhance its likeliness of retrieving important pages first.

The remainder of this paper is structured as follows: Section 2 reviews the PageRank metric we used to evaluate the effectiveness of crawling in breadth-first search order. Section 3 describes the tools we used to conduct our experiments. Section 4 describes the experiments we performed, and the results we obtained. Finally, section 5 offers concluding remarks.

2. PAGERANK

There are many conceivable metrics for judging the quality of a web page: by analyzing its content, by measuring its popularity (that is, how often it is viewed), or by examining its connectivity (that is, by determining which other pages link to this page, and vice versa). Metrics based on connectivity have the advantages that they do not require information that is not easily accessible (such as page popularity data), and that they are easy to compute, so they scale well to even very large page collections. They also require retrieving only the links on each page, not the full page contents. Storing the full page contents requires several kilobytes per page, one to two orders of magnitude more than just storing the links.

PageRank is the connectivity-based page quality measure suggested by Brin and Page [2]. It is a static measure; it is designed to rank pages in the absence of any queries. That is, PageRank computes the "global worth" of each page. Intuitively, the PageRank measure of a page is similar to its in-degree, which is a possible measure of the importance of a page. The PageRank of a page is high if many pages with a high PageRank contain links to it, and a page containing few outgoing links contributes more weight to the pages it links to than a page containing many outgoing links. The PageRank of a page is expressed mathematically as follows. Suppose there are T total pages on the web. We choose a parameter d (explained below) such that $0 < d < 1$; a typical value of d might lie in the range $0.1 < d < 0.15$. Let pages $p_1, p_2, \ldots, p_k$ link to page p. Let $R(p_i)$ be the PageRank of p_i and $C(p_i)$ be the number of links out of p_i. Then the PageRank $R(p)$ of page p is defined to satisfy:

$$R(p) = \frac{d}{T} + (1-d)\sum_{i=1}^{k} \frac{R(p_i)}{C(p_i)}$$

This equation defines $R(p)$ uniquely, modulo a constant scaling factor. If we scale $R(p)$ so that the PageRanks of all pages sum to 1, $R(p)$ can be thought of as a probability distribution over pages.

The PageRank distribution has a simple interpretation in terms of a random walk. Imagine a web surfer who wanders the web. If the surfer visits page p, the random walk is in state p. At each step, the web surfer either jumps to a page on the web chosen uniformly at random, or the web surfer follows a link chosen uniformly at random from those on the current page. The former occurs with probability d, the latter with probability $1 - d$. The equilibrium probability that such a surfer is at page p is simply $R(p)$. An alternative way to say this is that the average fraction of the steps that a walk spends at page p is $R(p)$ over sufficiently long walks. This means that pages with high PageRank are more likely to be visited than pages with low PageRank.

In our experiments, we set $d = \frac{1}{7} = 0.14$. We also modified PageRank slightly so that pages with no outgoing links contribute their weight equally to all pages. That is, the random surfer is equally likely to jump to any page from a page with no outgoing links. We ran experiments using both the original PageRank algorithm, which does not distinguish between links to pages on the same versus different hosts, and a variant of PageRank which only considers links to different hosts.

3. TOOLS

We used two tools in conducting this research: Mercator and the Connectivity Server 2, both developed at our lab. We used Mercator to crawl the web, and the Connectivity Server 2 to provide fast access to the link information downloaded from the crawl.

Mercator is an extensible, multithreaded, high-performance web crawler [7, 10]. It is written in Java and is highly configurable. Its default download strategy is to perform a breadth-first search of the web, with the following three modifications:

1. It downloads multiple pages (typically 500) in parallel. This modification allows us to download about 10 million pages a day; without it, we would download well under 100,000 pages per day.

2. Only a single HTTP connection is opened to any given web server at any given time. This modification is necessary due to the prevalence of relative URLs on the web (about 80% of the links on an average web page refer to the same host), which leads to a high degree of host locality in the crawler's download queue. If we were to download many pages from the same host in parallel, we would overload or even crash that web server.

3. If it took t seconds to download a document from a given web server, then Mercator will wait for $10t$ seconds before contacting that web server again. This modification is not strictly necessary, but it further eases the load our crawler places on individual servers on the web. We found that this policy reduces the rate of complaints we receive while crawling.

For the experiments described below, we configured Mercator to extract all the links from each downloaded page and save them to disk; for disk space reasons, we did not retain the pages themselves. We conducted a crawl that attempted to download 532 million pages over the course of 58 days (which we refer to as days 1 to 58 throughout the paper). Of all those download attempts, 328 million returned valid, unique HTML pages; the others resulted in TCP- and DNS-errors, non-200 HTTP return codes, non-HTML documents, or duplicates. Mercator's download rate decreased over the course of the crawl, due to increasing access times to one of its disk-based data structures that keeps track of which URLs have already been seen. The median download day was 22; the mean download day was 24.5.

The extracted links data was then loaded into the Connectivity Server 2 (CS2) [11], a database for URLs and links. A build of CS2 takes a web crawl as input and creates a database representation of the web graph induced by the pages in the crawl. A CS2 database consists of all URLs that were crawled, extended with all URLs referenced at least five times by the crawled pages. (Incorporating uncrawled URLs with multiple links pointing to them ensured that we did not ignore any popular URLs. Setting the threshold at five incoming links reduced the set of uncrawled URLs by over 90%, which enabled us to fit the database within the 16 GB of RAM available to us.) The CS2 database also contains all links among those URLs and host information for each URL. It maps each URL to all of its outgoing and its incoming links. It is possible to get all the incoming links for a given URL, or just the links from different hosts.

CS2 stores links in both directions in, on average, 2.4 bytes per link (as compared to 8 bytes per link in the original connectivity server (CS1) described in [1]). Like CS1,

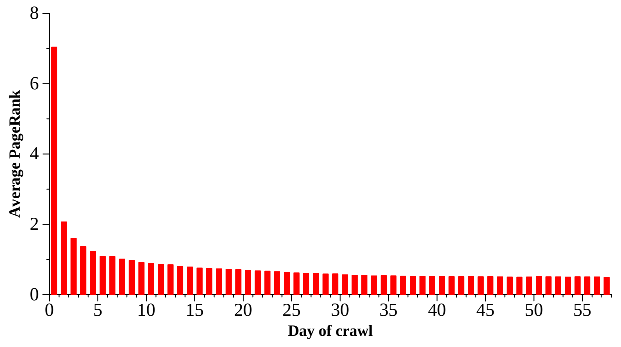

Figure 1: Average PageRank score by day of crawl

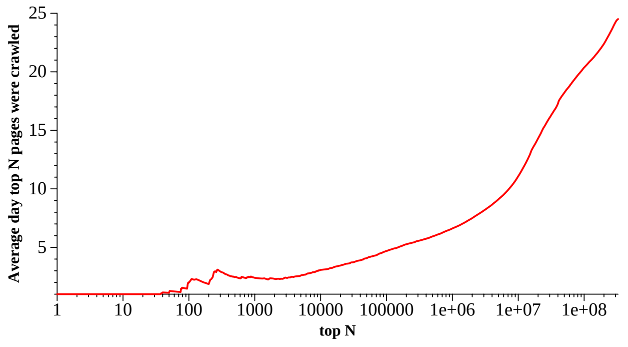

Figure 2: Average day on which the top N pages were crawled

CS2 is designed to give high-performance access when run on a machine with enough RAM to store the database in memory. On the 667 MHz Compaq AlphaServer ES40 with 16 GB of RAM used in our experiments, it takes 70-80 ms to convert a URL into an internal id or vice versa, and 0.1 ms/link to retrieve each incoming or outgoing link as an internal id. The database for our crawl of 328 million pages contained 351 million URLs and 6.1 billion links. Therefore, one iteration of PageRank ran in about 15 minutes.

4. AVERAGE PAGE QUALITY OVER A LONG CRAWL

In this section, we report on our experiments. We implemented PageRank and its variants over the CS2 interface, and ran each algorithm for 100 iterations on the 6.1 billion link database. (In all our experiments, the PageRank computation converged within less than 100 iterations.)

Although the PageRank scores are conventionally normalized to sum to 1 (making it easier to think of them as a probability distribution), we normalized them to sum to the number of nodes in the graph (351 million). This way, the *average* page has a PageRank of 1, independent of the number of pages.

Figure 1 shows the average PageRank of all pages downloaded on each day of the crawl. The average score for pages crawled on the first day is 7.04, more than three times the average score of 2.07 for pages crawled on the second day. The average score tapers from there down to 1.08 after the first week, 0.84 after the second week, and 0.59 after the fourth week. Clearly, we downloaded more high quality pages, i.e., pages with high PageRank, early in the crawl than later on. We then decided to examine specifically when we had crawled the highest ranked pages.

We examined the pages with the top N PageRanks, for increasing values of N from 1 to 328 million (all of the pages downloaded). Figure 2 graphs the average day on which we crawled the pages with the highest N scores. Note that the horizontal axis shows the values of N on a log scale.

All of the top 10 and 91 of the top 100 pages were crawled on the first day. There are some anomalies in the graph between N equals 100 and 300, where the average day fluctuates between 2 and 3 (the second and third days of the crawl). These anomalies are caused by 24 pages in the top 300 (8%) that were downloaded after the first week. Most of those pages had a lot of local links (links from pages on the same host), but not many remote links. In other words, the pages on the same host "endorse" each other, but few other hosts endorse them. We address this phenomenon later in the last experiment, shown in Figure 4. After N equals 400, the curve steadily increases to day 24.5, the mean download day of the entire crawl.

Our next experiment checks that pages with high PageRank are not ranked high only because they were crawled early. For example, a page whose outgoing links all point to pages with links back to it might have an artificially high PageRank if all of its outgoing links have been crawled, but not too many other pages. For this experiment we ran the PageRank algorithm on the graph induced by only the first 28 days of the crawl. This graph contains 217 million URLs and 3.8 billion links between them. We then compared the top ranked pages between the two data sets. We found that of the top 1 million scoring pages, 96% were downloaded during the first 4 weeks, and 76% of them were ranked in the top 1 million pages in the 28 day data set. That is, it was clear that those pages were important even before the crawl had finished.

Figure 3 generalizes these statistics: for each value of N, we plot the percentage of overlap between the top N scoring pages in the 28 day crawl versus the 58 day crawl. Although the top few pages are different, by the top 20 ranked pages there is an 80% overlap. The overlap continues in the 60-80% range through the extent of the entire 28 day data set. This figure suggests that breadth-first search crawling is fairly immune to the type of self-endorsement described above: although the size of the graph induced by the full crawl is about 60% larger than the graph induced by the 28 day crawl, the longer crawl replaced only about 25% of the "hot" pages discovered during the first 28 days, irrespective of the size of the "hot" set.

Some connectivity-based metrics, such as Kleinberg's algorithm [8], consider only remote links, that is, links between pages on different hosts. We noticed that some anomalies in Figure 2 were due to a lot of local links, and decided to experiment with a variant of the PageRank algorithm that only propagates weights along remote links. This modification of PageRank counts only links from different hosts as proper endorsements of a page; links from the same host are viewed as improper self-endorsement and therefore not counted.

Figure 4 shows our results: the average PageRank for pages downloaded on the first day is even higher than when all links are considered. The average PageRank for the first day is 12.1, while it's 1.8 on the second day and 1.0 on the

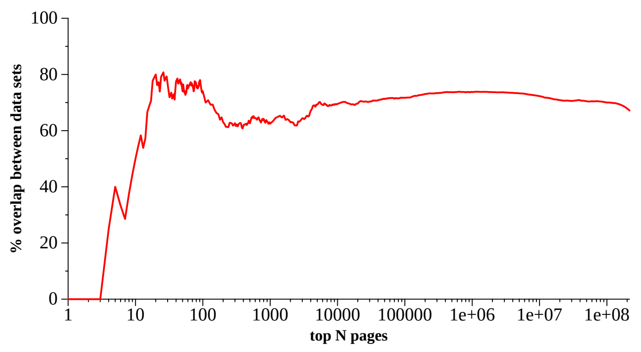

Figure 3: The percent overlap between the top N ranked pages in the first 28 vs all 58 days of the crawl

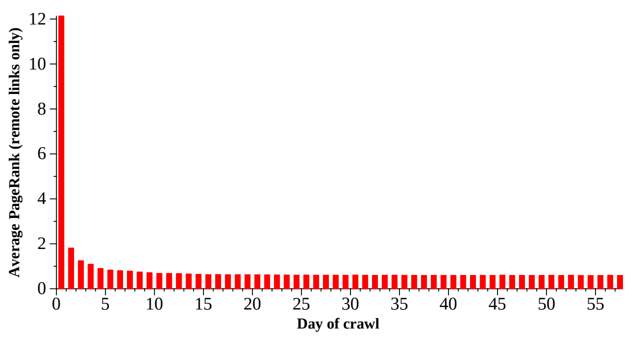

Figure 4: Average PageRank when only remote links are considered

fourth day. The average PageRank then declines gradually down to 0.6 on the last day. Notice that the average PageRank on the first day of crawling is higher than in Figure 1, and that the curve falls more sharply. This drop indicates that our crawling strategy is not biased toward self-endorsing hosts, as a crawler using the standard version of PageRank would be. We believe that this lack of bias is due in part to our crawler's politeness policies, which impose a rate limit on its accesses to any particular host.

There are some flaws with a metric based *only* on remote links. For example, *http://www.yahoo.com/* has a very high PageRank score. However, it only has local outlinks, so its weight gets evenly distributed over all pages in the graph, rather than just to the other pages in Yahoo! to which it points. Transitively, the pages on other hosts to which Yahoo! links do not benefit from the high score of *http://www.yahoo.com/*. In the future work section below, we outline some ideas for remedying this problem.

5. CONCLUSIONS

The experiments described in this paper demonstrate that a crawler that downloads pages in breadth-first search order discovers the highest quality pages during the early stages of the crawl. As the crawl progresses, the quality of the downloaded pages deteriorates. We speculate that breadth-first search is a good crawling strategy because the most important pages have many links to them from numerous hosts, and those links will be found early, regardless of on which host or page the crawl originates.

Discovering high-quality pages early on in a crawl is desirable for public web search engines such as AltaVista or Google, given that none of these search engines is able to crawl and index more than a fraction of the web.

Our results have practical implications to search engine companies. Although breadth-first search crawling seems to be a very natural crawling strategy, not all of the crawlers we are familiar with employ it. For example, the Internet Archive crawler described in [3] does not perform a breadth-first search of the entire web; instead, it picks 64 hosts at a time and crawls these hosts in parallel. Each host is crawled exhaustively; links that point to other hosts are saved to seed subsequent crawls of those hosts. This crawling strategy has no bias towards high-quality pages; if the hosts to be crawled are picked in random order, the quality of downloaded pages will be uniform throughout the crawl.

Similarly, the Scooter web crawler used until recently by AltaVista downloaded pages in essentially random order. (At this point, AltaVista is using Mercator.) This approach made it easier to provide politeness guarantees — essentially, it spread the load imposed by the crawler evenly over all web servers — but as a result, the quality of the discovered pages is uniform over the life of the crawl.

We cannot make any statements about other large-scale web crawlers. Most search engine companies treat their crawling strategy as a trade secret, and have not described it in the literature.

Cho et al. [4] showed that using a connectivity-based ordering metric for downloads, such as PageRank, will steer the crawler towards even higher-quality pages than using breadth-first search. However, computing PageRank values for several hundred million or more pages is an extremely expensive computation. It took us over a day to compute the PageRanks of our graph of 351 million pages, despite the fact that we had the hardware resources to hold the entire graph in memory! Using PageRank to steer a crawler would require multiple such computations over larger and larger graphs, in order to take newly discovered pages into account, and is essentially infeasible in real time. On the other hand, crawling in breadth-first search order provides a fairly good bias towards high quality pages without the computational cost. We believe that crawling in breadth-first search order provides the better tradeoff.

6. FUTURE WORK

There are two directions in which we would like to extend this work. One direction is to try a variant of PageRank which weighs links to pages on remote hosts differently than links to other pages on the same host. From the experiment that generated Figure 4 above, we learned that remote links should count more than local links, but that weights should be propagated along local links as well (e.g., to distribute the weight of *http://www.yahoo.com/* to the pages that Yahoo! recommends). We suspect that some search engines already use different weights for links, but there has been no formal study of how to divide the weights among the links or even whether the division should be static (e.g., remote links get 80% of the total weight) or proportional to the number of total links (e.g., each remote link gets four times the weight of each local link).

The other direction is to try different connectivity-based metrics. While PageRank is the only connectivity measure we know aimed at ranking all of the pages on the world wide web, Kleinberg's algorithm [8] is another well-known connectivity analysis algorithm targeted towards computing quality scores for pages. The algorithm computes two scores for each document: a *hub score* and an *authority score*. Pages with high authority scores are expected to have high-quality content; the authority scores are similar in intent to PageRanks. Kleinberg's algorithm is designed to rank the results of a query to a search engine, and only considers a small set of pages when it computes authority scores. However, we believe that we can extend the algorithm to consider the entire graph of the web.

7. REFERENCES

[1] K. Bharat, A. Broder, M. Henzinger, P. Kumar, and S. Venkatasubramanian. The connectivity server: Fast access to linkage information on the web. In *Proceedings of the 7th International World Wide Web Conference*, pages 469–477, Brisbane, Australia, April 1998. Elsevier Science.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the 7th International World Wide Web Conference*, pages 107–117, Brisbane, Australia, April 1998. Elsevier Science.

[3] M. Burner. Crawling towards eternity: Building an archive of the world wide web. *Web Techniques Magazine*, 2(5):37–40, May 1997.

[4] J. Cho, H. Garcia-Molina, and L. Page. Efficient crawling through URL ordering. In *Proceedings of the 7th International World Wide Web Conference*, pages 161–172, Brisbane, Australia, April 1998. Elsevier Science.

[5] Google Inc. Press release: "Google launches world's largest search engine." June 26, 2000. Available at *http://www.google.com/press/pressrel/pressrelease26.html*

[6] M. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork. On near-uniform URL sampling. In *Proceedings of the 9th International World Wide Web Conference*, pages 295–308, Amsterdam, Netherlands, May 2000. Elsevier Science.

[7] A. Heydon and M. Najork. Mercator: A scalable, extensible web crawler. *World Wide Web*, 2(4):219–229, Dec. 1999.

[8] J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, pages 668–677, San Francisco, CA, Jan. 1998.

[9] P. Lyman, H. Varian, J. Dunn, A. Strygin, and K. Swearingen. How much information? School of Information Management and Systems, Univ. of California at Berkeley, 2000. Available at *http://www.sims.berkeley.edu/how-much-info*

[10] Mercator Home Page. *http://www.research.digital.com/SRC/mercator*

[11] J. L. Wiener, R. Wickremesinghe, M. Burrows, K. Randall, and R. Stata. Better link compression. Manuscript in progress. Compaq Systems Research Center, 2001.

VITAE

Marc Najork is a senior member of the research staff at Compaq Computer Corporation's Systems Research Center. His current research focuses on high-performance web crawling and web characterization. He was a principal contributor to Mercator, the web crawler used by AltaVista. In the past, he has worked on tools for web surfing, 3D animation, information visualization, algorithm animation, and visual programming languages. He received a Ph.D. in Computer Science from the University of Illinois at Urbana-Champaign in 1994.

Janet L. Wiener is a member of the research staff at Compaq Computer Corporation's Systems Research Center. She currently focuses on developing algorithms to characterize the web and tools (such as the Connectivity Server) to support those algorithms. Prior to joining Compaq in 1998, she was a research scientist at Stanford University working on data warehousing, heterogeneous data integration, and semi-structured data. She received a Ph.D. from the University of Wisconsin-Madison in 1995, and a B.A. from Williams College in 1989.