# A Review on Efficient Web Crawling

Anish Gupta[1]

*Research Scholar, Department of Information Technology, B. R. Ambedkar Bihar University, Muzaffarpur, Bihar (India)*
*E-mail: gupta.anish01@gmail.com*

K. B. Singh[2]

*Lecturer, Department of Physics, Government Polytechnic, Dharbhanga, Bihar (India)*

*Abstract-* **This Research paper reviews the challenges and issues faced in implementing an effective WebCrawler. A crawler is a program that retrieves and stores pages from the Web, commonly for a Web search engine. A crawler often has to download hundreds of millions of pages in a short period of time and has to constantly monitor and refresh the downloaded pages. In addition, the crawler should avoid putting too much pressure on the visited Web sites and the crawler's local network, because they are intrinsically shared resources. In this research paper we also discuss about the web indexing.**

*Key words* –**web crawling, web indexing.**

## I. INTRODUCTION

In crawler program collects automatically web pages to create a local index and / or a local collection of web pages. In the context of the World Wide Web, crawling refers to gathering web pages, by following hyperlinks, starting from a small set of web pages, for the purposes of further processing. For example, a Web search engine needs to gather as many pages as possible before it indexes and makes them available for searching. Although it seems pretty straightforward, writing a good web crawler is not very much so. There are a good number of challenges which vary subtly depending on whether it's a large-scale web crawler or a crawler for a handful of websites. These challenges include: ensuring politeness to the web servers (by observing the widely accepted robots exclusion protocol), URL normalization, duplicate detection, avoiding spider traps, maintaining a queue of un-fetched pages, maintaining a repository of crawled pages, re-crawling and a few more. For large-scale crawlers, one of the most important challenges is to increase the throughput by optimizing the resource utilization, because their coverage usually gets limited by this.

## II. CHALLENGES IN IMPLEMENTING A CRAWLER

Given this size and change rate of the Web, the crawler needs to address many important Challenges, including the following:

**A. What pages should the crawler download-** In most cases the crawler cannot download all pages on the Web. Even the most comprehensive search engine currently indexes a small fraction of the entire Web [1] [2]. Given this fact, it is important for the crawler to carefully select the pages and to visit "important" pages first, so that the fraction of the Web that is visited (and kept up-to-date) is more meaningful.

**B. How should the crawler refresh pages-** Once the crawler has downloaded a significant number of pages, it has to start revisiting the downloaded pages in order to detect changes and refresh the downloaded collection. Because Web pages are changing at very different rates [3] [4], the crawler needs to carefully decide which pages to revisit and which pages to skip in order to achieve high "freshness" of pages. For example, if a certain page rarely changes, the crawler may want to revisit the page less often, in order to visit more frequently changing ones.

**C. How should the load on the visited Web sites be minimized-** When the crawler collects pages from the Web, it consumes resources belonging to other organizations [5]. For example, when the crawler downloads page p on site S the site needs to retrieve page p from its file system, consuming disk and CPU resources. After

this retrieval the page then needs to be transferred through the network, which is another resource shared by multiple organizations. Therefore, the crawler should minimize its impact on these resources [Rob]. Otherwise, the administrators of a Web site or a particular network may complain and sometimes may completely block access by the crawler.

D. **How should the crawling process be parallelized**- Due to the enormous size of the Web, crawlers often run on multiple machines and download pages in parallel [6] [7]. This parallelization is often necessary in order to download a large number of pages in a reasonable amount of time. Clearly these parallel crawlers should be coordinated properly, so that different crawlers do not visit the same Web site multiple times. However, this coordination can incur significant communication overhead, limiting the number of simultaneous crawlers.

## III. A WEB CRAWLER DESIGN

The first crawler, Mathew Gray's Wanderer, was written in the spring of 1993, roughly coinciding with the first release of NCSA MOSAIC. [8]

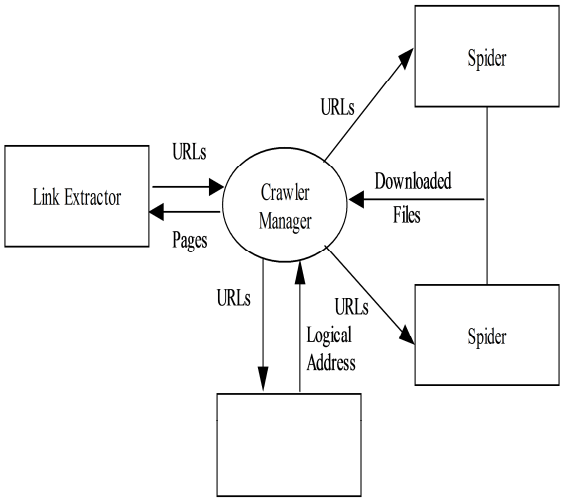


Fig. Web Crawler Architecture

A. **Crawler Manager:** Takes a set of URLs from Link Extractor and sends the Next URL to the DNS resolver to obtain its IP address. This saves a lot of time because spiders do not have to send requests to DNS every time they want to download a page.

B. **Robots.txt file:** are the means by which web authors express their wish as to which pages they want the crawlers to avoid. Crawlers must respect authors' wishes as well.

C. **Spider:** downloads robots.txt file and other pages that are requested by the crawler manager and permitted by web authors. The robots.txt files are sent to crawler manager for processing and extracting? the URLs. The other downloaded files are sent to a central indexer.

D. **Link Extractor:** look through the pages downloaded by the spiders, extracts URLs from the links in those pages and sends the URLs to the crawler manager for downloading afterwards.

Any crawler must fulfill following two issues: [9]

1) It must have a good crawling strategy

2) It has to have a highly optimized system architecture that can download a large number of pages per seconds.

Most of search engines use more than one crawler and manage them in a distributed method. This has following benefits:

- Increased resource utilization
- Effective distribution of crawling tasks with no bottle necks
- Configurability of the crawling tasks

## IV. WEB CRAWLING ISSUES

There are two important characteristics of the Web that generate a scenario in which Web crawling is very difficult: its large volume and its rate of change, as there is a huge amount of pages being added, changed and removed every day. Also, network speed has improved less than current processing speeds and storage capacities. The large volume implies that the crawler can only download a fraction of the Web pages within a given time, so it needs to prioritize its downloads. The high rate of change implies that by the time the crawler is downloading the last pages from a site, it is very likely that new pages have been added to the site, or that pages that have already been updated or even deleted. Crawling the Web, in a certain way, resembles watching the sky in a clear night: what we see reflects the state of the stars at different times, as the light travels different distances. What a Web crawler gets is not a “snapshot” of the Web, because it does not represents the Web at any given instant of time [10]. The last pages being crawled are probably very accurately represented, but the first pages that were downloaded have a high probability of have been changed. As

Edwards et al. note, "Given that the bandwidth for conducting crawls is neither infinite nor free it is becoming essential to crawl the Web in a not only scalable, but efficient way if some reasonable measure of quality or freshness is to be maintained." [11]. A crawler must carefully choose at each step which pages to visit next. The behavior of a Web crawler is the outcome of a combination of policies:

- A selection policy that states which pages to download.
- A re-visit policy that states when to check for changes to the pages.
- A politeness policy that states how to avoid overloading Web sites.
- A parallelization policy that states how to coordinate distributed Web crawlers

## V. WEB INDEXING vs WEB CRAWLING

The main work of web crawling program is web indexing. Web indexing also known as internet indexing refers to various methods for indexing the contents of a website or of the internet as a whole.

- Web Crawling: Finding information
- Web Indexing: Organizing information

We use software known as "web crawlers" to discover publicly available web pages. The most well-known crawler is called "Googlebot." Crawlers look at web pages and follow links on those pages, much like you would if you were browsing content on the web. They go from link to link and bring data about those web pages back to Google's servers. The crawl process begins with a list of web addresses from past crawls and sitemaps provided by website owners. As our crawlers visit these websites, they look for links for other pages to visit. The software pays special attention to new sites, changes to existing sites and dead links. Computer programs determine which sites to crawl, how often, and how many pages to fetch from each site. Google doesn't accept payment to crawl a site more frequently for our web search results. We care more about having the best possible results because in the long run that's what's best for users and, therefore, our business.

The web is like an ever-growing public library with billions of books and no central filing system. Google essentially gathers the pages during the crawl process and then creates an index, so we know exactly how to look things up. Much like the index in the back of a book, the Google index includes information about words and their locations. When you search, at the most basic level, our algorithms look up your search terms in the index to find the appropriate pages. The search process gets much more complex from there. When you search for "dogs" you don't want a page with the word "dogs" on it hundreds of times. You probably want pictures, videos or a list of breeds. Google's indexing systems note many different aspects of pages, such as when they were published, whether they contain pictures and videos, and much more. With the Knowledge Graph, we're continuing to go beyond keyword matching to better understand the people, places and things you care about.

## VI. CONCLUSION

In this review paper, we found that web crawling is an active research topic in the information retrieval community. The Web is very important today because it is the cornerstone of the information age, and is used by millions of persons every day, and it is natural that it provides opportunities for both business and research. Link analysis is, in a sense, the most important "new" component of the Web in relation to previous document collections and traditional information retrieval, and probably this explain why the field of web crawling has been so active.

## REFERENCES

1. Steve Lawrence and C. Lee Giles. Accessibility of information on the web Nature, 400(6740):107–109, July 1999.
2. Krishna Bharat and Andrei Broder. Mirror, mirror on the web: A study of host pairs with replicated content. In Proceedings of the Eighth International World-Wide Web Conference, Toronto, Canada, May 1999.
3. Junghoo Cho and Hector Garcia-Molina. The evolution of the web and implications for an incremental crawler. In Proceedings of the Twenty-sixth International Conference on Very Large Databases, Cairo, Egypt, September 2000.
4. Craig E. Wills and Mikhail Mikhailov. Towards a better understanding of web resources and server responses for improved caching. In Proceedings of the Eighth International World-Wide Web Conference, Toronto, Canada, May 1999.
5. Martijn Koster. Robots in the web: threat or treat? ConneXions, 4(4), April 1995.
6. Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In Proceedings of the Seventh International World-Wide Web Conference, Brisbane, Australia, April 1998.
7. Junghoo Cho and Hector Garcia-Molina. The evolution of the web and implications for an incremental crawler. In Proceedings of the Twenty-sixth International

Conference on Very Large Databases, Cairo, Egypt, September 2000.
8. A. Heydon, M. Najork, (1999). "Mercator: A Scalable, Extensible Web Crawler" in World Wide Web, 2(4): 219-229
9. T. Suel, V. Shkapenyuk, (2002). "Design and Implementation of a High-Performance Distributed Web Crawler" In Proceedings of the 18th International Conference on Data Engineering (ICDE'02), San Jose, CA Feb. 26--March 1, pages 357—368
10. Ricardo Baeza-Yates and Berthier Ribeiro-Neto. Modern Information Retrieval. ACM Press Addison-Wesley, 1999.
11. Jenny Edwards, Kevin S. McCurley, and John A. Tomlin. An adaptive model for optimizing performance of an incremental web crawler. In Proceedings of the Tenth Conference on World Wide Web, pages 106–113, Hong Kong, May 2001. Elsevier Science.