ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

# A Survey on Web Crawling Techniques by Emphasizing Path-mounting crawling

**Prerna Dewangan**

## Abstract

The World Wide Web (WWW) is being prolonged by an impulsive speed. As a result, search engines encounter many challenges such as yielding accurate and conversant results to the users, and responding them in an appropriate timely manner. A crawler is a program that downloads and stores Web pages, often for a Web search engine. Web crawler (also known as a Web spider or Web robot) is a program or automated script which browses the World Wide Web crawlers are mainly used to create a copy of all the visited pages for later processing by a search engine, which will index the downloaded pages to provide fast searches. We have concluded that the advantage with Path-mounting crawler is that they are very effective in finding isolated resources, or resources for which no inbound link would have been found in regular crawling.

**Keywords**: Web ceawling, Path-mounting crawling.

## Introduction

The Internet is a vast global network allowing people all over the world to communicate and share information. This is a system of interlinked hypertext documents stored on servers all over the world, accessible through a number of protocols built on top of the internet architecture. In order to harvest this enormous data repository, search engines download parts of the existing web and offer Internet users access to this database through keyword search. One of the main components of search engines is web crawler. Muskesh Kumar [1] says a crawler is a program used by search engine that retrieves Web pages by wandering around the Internet following one link to another. Web search engines such as Goggle, AtlaVista provides access to the Web documents. A search engine crawler collects web documents and periodically revisits the pages to update the index of the search engine. There are some reasons why we need a web crawler:

(i) To maintain mirror sites for popular Web sites.
(ii) To test web pages and links for valid syntax and structure.
(iii) To monitor sites to see when their structure or contents change.
(iv) To search for copyright infringements.
(v) To build a special-purpose index for example, one that has some understanding of the content stored in multimedia files on the Web.

The remainder of this paper is organized as follows: Section 2 provides a brief review of the Background. Section 3 describes the Behavior of web crawler. In Section 4, we drew the attention towards problems in existing systems. Finally, we concluded our study

## Background

There are different types of crawlers and the different techniques used make one to consider different issues while designing and implementing them[2] [4] [6] [7].

(i) General-Purpose Web Crawler: - General-purpose web crawlers collect and process the entire contents of the Web in a centralized location, so that it can be indexed in advance to be able to respond to many user queries. In the early stage when the Web is still not very large, simple or random crawling method was enough to index the whole web.
(ii) Topic-focused Web Crawling: - Topic-Focused Web Crawling initiation was motivated by the fact the Web is huge with an unprecedented scaling problem, but most people are only interested in a small fraction of the Web. The main objective is to only crawl on a small fraction of the Web to discover the set of pages covering a certain topic [2] [3] [4] [8] [9].

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

(iii) Path-ascending Crawling: - also known as Web harvesting software, because they're used to "harvest" or collect all the contents from a specific page or host. Some crawlers intend to download as many resources as possible from a particular web site.

(iv) Adaptive Crawler: - It is classified as an incremental type of crawler which will continually crawl the entire web, based on some set of crawling cycles. The adaptive model used would use data from previous cycles to decide which pages should be checked for updates. Adaptive Crawling can also be viewed as an extension of focused crawling technology.

When using the term "Web Crawler" most people would more than likely think of the most popular site on the web Google. However, there are also several other large-scale crawlers such as: Microsoft Bing, Internet Archive, Yahoo. There are also several open-source implementations for large-scale crawling such as: Apache Nutch1, ABot2 and Heritrix3. Majority of the larger--scale web crawlers are generally used as the background processing for search engines. Indexing and ranking pages based on their content quality and returning the correct information and results from search queries is a complicated and resource intensive task – hence why they're probably the most appreciated in terms of web crawling. However not all crawlers are design to cover the entire web in a "general" fashion. Crawlers such as the Heritrix are designed to crawl the entire web and mirror exactly what it discovers making it a crawler that is designed to download not only web-pages but other media types such as images and zip archives. Although there are open-source implementations regarding web crawling, majority of the large-scale web crawler solutions are "business secrets" making the competition to build an exceptional crawler all the more difficult. Since the web is growing an increasingly fast rate, there are billions of web pages to process. However, the number of URL's pointing to these billions of web pages greatly exceeds the number of web pages that exist, which is why it is important to design a structurally efficient web crawler.


***Figure 1. A general overview of a web crawler's architecture.***

As a general overview, a web crawler looks relatively simple. High-level components include:

- Fetcher & Parser – Downloads & parses downloaded content.
- Storage – Form of storage method for storing URLs and any specifically crawled content.
- Queue – a queue of URLs ready to be crawled by the crawler.
- Scheduler – a method of scheduling URLS (can either be based on content, timeout periods or other factors).

The functionalities of a Web crawler is given below:

1. **The crawler starts crawling with a set of URLs fed into it, known as seed URLs.**
2. **The crawler downloads the page.**
3. **It extracts the URLs from the downloaded page and inserts them into a queue. From the queue the crawler again retrieves the URL for downloading next pages.**
4. **The downloaded page is saved in the repository.**
5. **The process continues until the crawler stops.**

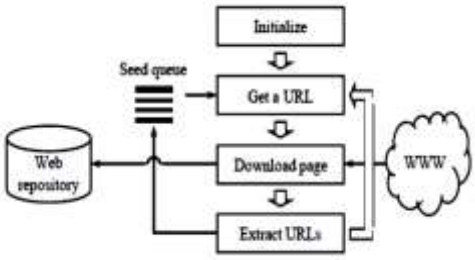


***Fig 2. : Flow of a crawling process.***

## Behavior of web crawler

The behavior of a Web crawler is the outcome of a combination of policies:

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

i. **Selection policy** that states which pages to download. Large search engines cover only a portion of the publicly available part. As a crawler always downloads just a fraction of the Web pages, it is highly desirable that the downloaded fraction contains the most relevant pages and not just a random sample of the Web. This requires a metric of importance for prioritizing Web pages. The importance of a page is a function of its intrinsic quality, its popularity in terms of links or visits, and even of its URL. Abitebou [10] designed a crawling strategy based on an algorithm called OPIC (On-line Page Importance Computation).1 In OPIC, each page is given an initial sum of "cash" that is distributed equally among the pages it points to. It is similar to a Pagerank computation, but it is faster and is only done in one step. An OPIC-driven crawler downloads first the pages in the crawling frontier with higher amounts of "cash". Experiments were carried in a 100,000-pages synthetic graph with a power-law distribution of in-links. However, there was no comparison with

ii. **Re-visit policy** that states when to check for changes to the pages, • a politeness policy that states how to avoid overloading Web sites. The Web has a very dynamic nature, and crawling a fraction of the Web can take weeks or months. By the time a Web crawler has finished its crawl, many events could have happened, including creations, updates and deletions. From the search engine's point of view, there is a cost associated with not detecting an event, and thus having an outdated copy of a resource. The most-used cost functions are freshness and age[11].

**Freshness**: This is a binary measure that indicates whether the local copy is accurate or not. The freshness of a page p in the repository at time t is defined as:

Fp(t)={ **1** if is equal to the local copy at time t and **0** otherwise

**Age**: This is a measure that indicates how outdated the local copy is. The age of a page p in the repository.

iii. **Parallelization** policy that states how to coordinate distributed Web crawlers. A Parallel crawler is a crawler that runs multiple processes in parallel. The goal is to maximize the download rate while minimizing the overhead from parallelization and to avoid repeated downloads of the same page. To avoid downloading the same page more than once, the crawling system requires a policy for assigning the new URLs discovered during the crawling process, as the same URL can be found by two different crawling processes.

## Problem Identification

Hence after studying various crawler architecture we have identified some difficulties.

1. There are two important characteristics of the Web that generate a scenario in which Web crawling is very difficult:
    a. Large volume of Web pages.
    b. Rate of change on web pages.
2. A large volume of web page implies that web crawler can only download a fraction of the web pages and hence it is very essential that web crawler should be intelligent enough to prioritize download.
3. Another problem with today dynamic world is that web pages on the internet change very frequently, as a result, by the time the crawler is downloading the last page from a site, the page may change or a new page has been placed/updated to the site.
4. Overloading websites: Crawlers can retrieve data much quicker and in greater depth than human searchers, so they can have a crippling impact on the performance of a site. Needless to say if a single crawler is performing multiple requests per second and/or downloading large files, a server would have a hard time keeping up with requests from multiple crawlers. The use of Web crawler is useful for a number of tasks, but comes with a price for the general community. The costs of using Web crawlers include:
    a. Network resources, as crawlers require considerable bandwidth and operate with a high degree of parallelism during a long period of time.

ISSN: 2277-9655
Scientific Journal Impact Factor: 3.449
(ISRA), Impact Factor: 1.852

b. Server overload, especially if the frequency of accesses to a given server is too high.
c. Poorly written crawlers, which can crash servers or routers, or which download pages they cannot handle.
d. Personal crawlers that, if deployed by too many users, can disrupt networks and Web servers.

## Conclusion

Edifice an effective web crawler to solve our purpose is not a difficult task, but choosing the right strategies and building an effective architecture will lead to implementation of highly intelligent web crawler application. After study we have concluded that the advantage with Path-ascending crawler is that they are very effective in finding isolated resources, or resources for which no inbound link would have been found in other crawling. On the other hand the main problem in focused crawling is that in the context of a Web crawler, we would like to be able to predict the similarity of the text of a given page to the query before actually downloading the page. A possible predictor is the anchor text of links; to resolve this problem proposed solution would be to use the complete content of the pages already visited to infer the similarity between the driving query and the pages that have not been visited yet.

## References

1. Mukesh Kumar, Renu Vig Learnable Focused Meta Crawling Through Web International Conference on Communication, Computing & Security [ICCCS-2012] ELSEVIER.
2. Bidoki, Yazdani et el, "FICA: A fast intelligent crawling algorithm", Web Intelligence, IEEE/ACM/WIC International conference on Intelligent agent technology, Pages 635-641, 2007.
3. Cui Xiaoqing Yan Chun," An evolutionary relevance calculation measure in topic crawler " CCCM 2009, ISECS International Colloquium on Computing, Communication, Control, and Management, 267 – 270, aug 2009.
4. Junghoo Cho, Hector Garcia-Molina, Lawrence Page, |Efficient crawling through URL ordering", 7th International WWW Conference , April 14-18, Brisbane, 1998.
5. Mukhopadhyay et al, "A New Approach to Design Domain Specific Ontology Based Web Crawler", ICIT 2007, 10th International Conference on Information Technology, 289 - 291, Dec. 2007 .
6. Peisu, Ke et el, "A Framework of deep web crawler", 27th Chinese Proceedings of the 27th Chinese Control Conference, Pages 582-586, July 16-18, 2008.
7. www.wikipedia.org/web_crawler , accessed last May 12, 2010.
8. Yadav, Sharma et el, "Architecture for parallel crawling and algorithm for change detection in web pages", 10th International Conference on Information Technology, Pages 258-264, ICIT 2007.
9. Yuan, Yin et el, "Improvement of pagerank for focused crawler", 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, Pages 797-802, SNPD 2007.
10. Abiteboul Serge, Mihai Preda, and Gregory Cobena (2003). "Adaptive On-line Pag