Volume 4, Issue 4, April 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Web Crawler

Shalini Sharma
Department of Computer Science
Shoolini University, Solan (H.P.), India

Abstract*— *Web crawler is a computer program that browses the World Wide Web in a methodical, automated manner or in an orderly fashion. Web crawling is an important method for collecting data on, and keeping up with, the rapidly expanding Internet. A vast number of web pages are continually being added every day, and information is constantly changing. This Paper is an overview of Web Crawlers and the policies like selection, revisit, politeness, parallelization involved in it. The behavioral pattern of the Web crawler based on these policies is also taken for the study.

Keywords: Web Crawler, Behavior, Policies,WebPage,Optimization

I Introduction

The world wide web has grown from a few thousand pages in 1993 to more than two billion pages at present. The contributing factor to this explosive growth is the widespread use of microcomputer, increased case of use in computer packages and most importantly tremendous opportunities that the web offers to business .New tools and techniques are crucial for intelligently searching for useful information on the web. Web crawler are an essential component of all search engines and are increasingly becoming important in data mining and other indexing applications. Web crawlers are also known as spiders, robots, walkers and wanderers, are programs which browse the web in methodical, automated manner. They are mainly used to create a copy of all the visited pages other possible applications include page validation, stuructural analysis and visualization, update notification, mirroring and personal web assistants/agents etc.[1]

There are important characteristics of the Web that make crawling very difficult: • its large volume,
• its fast rate of change, and
• dynamic page generation.

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 have already been updated or even deleted.

II Working of Web Crawler:-

Crawling is the most fragile application since it involves interacting with hundreds of thousands of web servers and various name servers,which are all beyond the control of the system.Web crawling speed is governed not only by the speed of one's own Internet connection,but also by the speed of the sites that are to be crawled.Especially if one is a crawling site from multiple servers,the total crawling time can be significantly reduced,if many downloads are done in parallel.
Roughly, a crawler starts off by placing an initial set of URLs, in a queue, where all URLs to be retrieved are kept and prioritized. From this queue, the crawler gets a URL (in some order), downloads the page, extracts any URLs in the downloaded page, and puts the new URLs in the queue.
This process is repeated.
The web crawler can be used for crawling through a whole site on the inter/ intranet.you specify a start-URL and the crawler follows all links found in that HTML page.this usually leads to more links which will be followed again ,and so on .A site can be seen as a tree structure ,the root is the start URL,all links in that root-HTML page are direct sons of the root. Subsequent links are then sons of the previous sons.[4]
The different data structures involved in the web crawlers are followings: -
Repository: -Manages and stores a large collection of web pages. It contains full HTML of every web page. The web pages are compressed in the repository. Different compression techniques can be used for compressing web pages, which is a tradeoff between speed and compression ratio. In repository documents are stored one after the other.
Document index: - keeps information about each document. It is a fixed width ISAM (Index sequential access mode) index, ordered by doc ID including document status, a pointer into the repository, a document checksum, and various statistics.
Indexer: - It is a program that "reads" the pages, which are downloaded. Web indexing includes indexes to individual websites or web documents to provide a more useful vocabulary for Internet search engines. The indexer also examines

the HTML code, which makes up the web page and looks for words that are considered important. Words in bold, italics or header tags are given more priority. Different indexing methods are available.

Hit list: -Stores the list of occurrences of a particular word in a particular document including position, font, and capitalization information. There are two types of hits: fancy hits and plain hits. Fancy hits consider hits occurring in a URL, title, anchor text, or Meta tag. Plain hits include everything else.

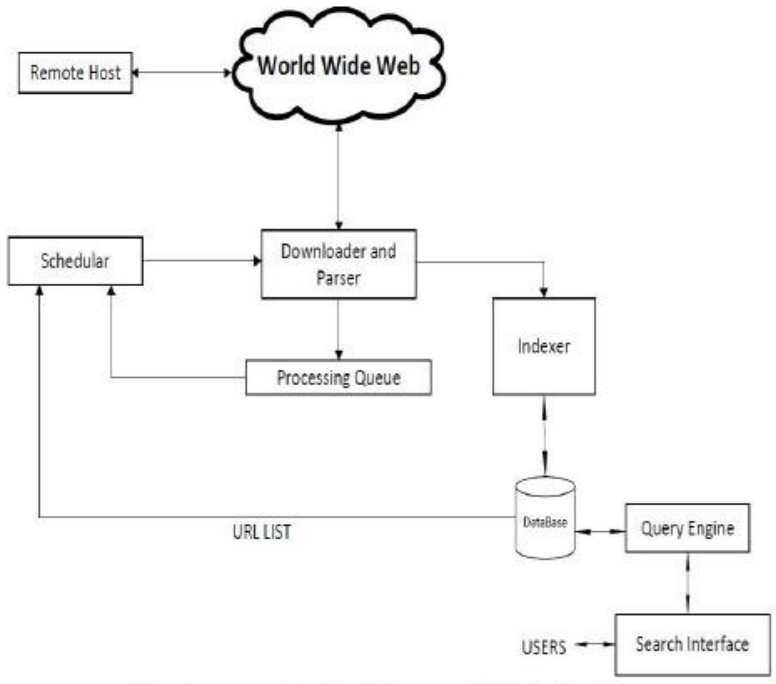

Fig. 1: General Architecture of Web Crawler

III BEHAVIOR OF WEB CRAWLER

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, and

• a parallelization policy that states how to coordinate distributed Web crawlers.

A. SELECTION POLICY

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.

Therefore good selection policy is very important .some of common selection policies are :

Restricting followed links

Path ascending crawling

Focused crawling

Crawling the deep web

B. *RE-VISIT POLICY*

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 age5.

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:

$$F_p(t) = \begin{cases} 1 & \text{if } p \text{ is equal to the local copy at time } t \\ 0 & \text{otherwise} \end{cases}$$

Age: This is a measure that indicates how outdated the local copy is. The age of a page p in the repository, at time t is defined as:

$$A_p(t) = \begin{cases} 0 & \text{if } p \text{ is not modified at time } t \\ t - \text{modification time of } p & \text{otherwise} \end{cases}$$

C. *POLITENESS POLICY*

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 crawlers is useful for a number of tasks, but comes with a price for the general community.The costs of using Web crawlers include:

• network resources, as crawlers require considerable bandwidth and operate with a high degree of
parallelism during a long period of time;
• server overload, especially if the frequency of accesses to a given server is too high;
• poorly-written crawlers, which can crash servers or routers, or which download pages they cannot handle; and
• personal crawlers that, if deployed by too many users, can disrupt networks and Web servers.

D. *PARALLELIZATION POLICY*

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.

IV. CONCLUSION

This paper describes about Different types of Web crawler and the policies used in the web crawlers. Different data structures used in web crawler and working is studied. A web crawler is a way for the search engines and other users to regularly ensure that their databases are up to date. Web crawlers are a central part of search engines, and details on their algorithms and architecture are kept as business secrets. When crawler designs are published, there is often an important lack of detail that prevents others from reproducing the work. There are also emerging concerns about" Search Engine Spamming", which prevent major search engines from publishing their ranking algorithms. New Modification and extension of the techniques in Web crawling should be next topics in this area of research.

REFERENCES

[1] Abiteboul Serge, Mihai Preda, and Gregory Cobena(2003). "Adaptive On-line Page Importance Computation". *Proceedings of the 12th International Conference on World Wide Web*.

[2] Shervin Daneshpajouh, Mojtaba Mohammadi Nasiri, Mohammad Ghodsi, "A Fast Community Based Algorithm for Generating Crawler Seeds Set". **[chakra 99]** Soumen Chakrabarti, and Martin Van Den Berg, and Byron Dom, " Focused Crawling: a New Approach to Topic-specific {Web} Resource Discovery*". Computer Networks*, 31 (11-16), pp. 1623-1640, **[michelan00]** Michelangelo Diligenti, Frans Coetzee, Steve Lawrence, C. Lee Giles, and Marco Gori

[3] Pant Gautam, Srinivasan Padmini, Menczer Filippo,(2004). "Crawling the Web" *In Levene, Mark; Poulovassilis, Alexandra. Web Dynamics: Adapting to Change in Content,Size, Topology and Use*. Springer. pp. 153-178.

[4] Cothey Viv, (2004). "Web-crawling Reliability". *Journal of the American Society for Information Science and Technology, 55 (14), 1228-1238.*

[5] Cho Junghoo, Hector Garcia-Molina, (2000). "Synchronizing a Database to Improve Freshness". Proceedings of the 2000 ACM SIGMOD *International Conference on Management of Data*. Dallas, Texas, United States:

[6] Jr. E.G. Coffman, Zhen Liu, Richard R. Weber, (1998). "Optimal Robot Scheduling for Web Search Engines". *Journal of Scheduling* **[edward00]** J. Edwards, K. McCurley, and J. Tomlin, "An Adaptive Model for Optimizing Performance of an Incremental Web Crawler", *Proceedings of the 10th International World Wide Web Conference*, Hong Kong, May 2001