Web Crawler: Extracting the Web Data

Mini Singh Ahuja
Research scholar
Punjab technical University
Jalandhar(India)

Dr Jatinder Singh Bal
Professor
KC Group of institutes
Nawashahr(India)

Varnica
Student (M.Tech)
GNDU, Regional Campus
Gurdaspur (India)

***Abstract*-- Internet usage has increased a lot in recent times. Users can find their resources by using different hypertext links. This usage of Internet has led to the invention of web crawlers. Web crawlers are full text search engines which assist users in navigating the web. These web crawlers can also be used in further research activities. For e.g. the crawled data can be used to find missing links, community detection in complex networks. In this paper we have reviewed web crawlers: their architecture, types and various challenges being faced when search engines use the web crawlers.**

Keywords—web crawler, blind traversal algorithms, best first heuristic algorithms etc.

I INTRODUCTION

The Web contains a large volume of information on different topics. In contrast to traditional collections such as libraries, the Web has no centrally organized content structure. This data can be downloaded using web crawler. So, Web crawler is software for downloading pages from the Web automatically. It is also called web spider or web robot. Web crawling is an important method for collecting data on, and keeping up with, the rapidly expanding Internet. Web crawling can also be called as a graph search problem as web is considered to be a large graph where nodes are the pages and edges are the hyperlinks. Web crawlers can be used in various areas, the most prominent one is to index a large set of pages and allow other people to search this index. A Web crawler does not actually move around computers connected to the Internet, as viruses or intelligent agents do, instead it only sends requests for documents on web servers from a set of already locations.
The general process that a crawler takes is as follows:-

- It checks for the next page to download – the system keeps track of pages to be downloaded in a queue.
- Checks to see if the page is allowed to be downloaded - checking a robots exclusion file and also reading the header of the page to see if any exclusion instructions were provided do this. Some people don't want their pages archived by search engines.
- Download the whole page.
- Extract all links from the page (additional web site and page addresses) and add those to the queue mentioned above to be downloaded later.
- Extract all words & save them to a database associated with this page, and save the order of the words so that people can search for phrases, not just keywords
- Optionally filter for things like adult content, language type for the page, etc.
- Save the summary of the page and update the last processed date for the page so that the system knows when it should re-check the page at a later stage.

II LITERATURE SURVEY

Many researchers have used web crawlers to get the web data for their research work. Web crawling can be used in web mining field to automatically discover and extract information from the WWW. Minas Gjoka [10], had used web data (extracted by web crawler) to measure the statistical properties of online social networks. Salvatore A. Catanese, Pasquale De Meo, Emilio Ferrara [11], has used crawlers on the social networking site i.e., face book. This data was later used to study the community structure of face book. Ari Pirkola [12], studied focused crawling to acquire biological data from the web. Priyanka-Saxena [13], introduced a web crawler called Mercator, which is a scalable web crawler written in java. Christopher Olston and Marc Najork [1] presented the basics of web crawling. In his paper he discussed the crawling architecture and also gave information about the future scope of crawling. Vladislav Shkapenyuk and Torsten Suel [14], gave the design and implementation of the distributed web crawler. Raja Iswary, Keshab Nath [15], discusses the different techniques to develop a crawler and how to build an efficient crawler.

III ARCHITECTURE OF WEB CRAWLER

A web crawler is one of the main components of the web search engines. The growth of web crawler is increasing in the same way as the web is growing. A list of URLs is available with the web crawler and each URL is called a seed. Each URL is visited by the web crawler. It identifies the different hyperlinks in the page and adds them to the list of URLs to visit. This list is termed as crawl frontier. Using a set of rules and policies the URLs in the frontier are visited individually. Different pages from the internet are downloaded by the parser and the generator and stored in the database system of the search engine. The URLs are then placed in the queue and later scheduled by the scheduler and can be accessed one by one by the search engine one by one whenever required. The links and related files which are being searched can be made available whenever required at later time according to the requirements. With the help of suitable algorithms web crawlers find the relevant links for the search engines and use them further. Databases are very big machines like DB2, used to store large amount of data [3].

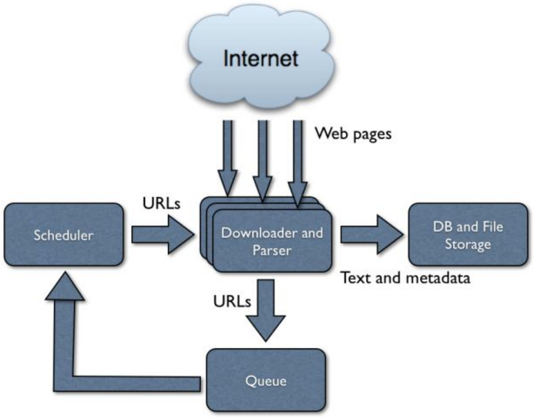

Fig 1: Architecture of Web Crawler

IV TYPES OF WEB CRAWLER

Different types of web crawlers are available depending upon how the web pages are crawled and how successive web pages are retrieved for accessing next pages. Some of which are following ones:-

A. Breadth First Crawler:

It starts with a small set of pages and then explores other pages by following links in the breadth-first [6] fashion. Actually web pages are not traversed strictly in breadth first fashion but may use a variety of policies. For example it may crawl most important pages first.

B .Incremental Web Crawlers:

An incremental crawler [5], is one, which updates an existing set of downloaded pages instead of restarting the crawl from scratch each time. This involves some way of determining whether a page has changed since the last time it was crawled. A crawler, which will continually crawl the entire web, based on some set of crawling cycles. An adaptive model is used, which uses data from previous cycles to decide which pages should be checked for updates, thus high freshness and results in low peak load is achieved.

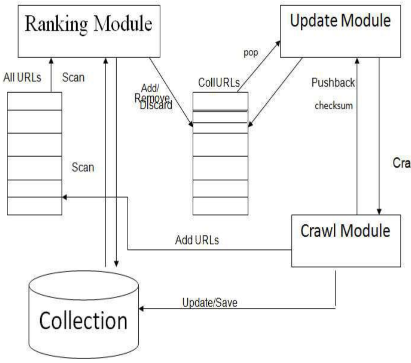

Fig.2 Architecture of Incremental Crawler

C.Form Focused Crawler:

Form Focused Cawler deals with sparse distribution of forms on the Web. Form Crawler [9] avoids crawling through unproductive paths by: limiting the search to a particular topic; learning features of links and paths that lead to pages that contain searchable forms; and employing appropriate stopping criteria. The architecture of the Form Crawler is depicted.

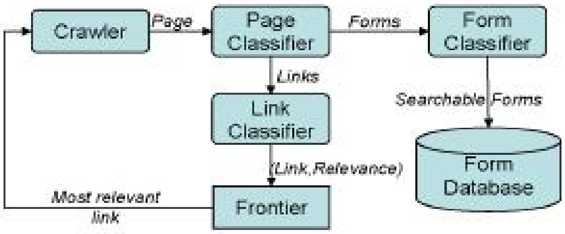

Fig.3 Architecture of Form Focused Crawler

The crawler uses two classifiers: the page and the link classifiers to guide its search. Later, a third classifier: the form classifier is used to filter out useless forms. The page classifier is trained to classify pages as belonging to topics in taxonomy. It uses the same strategy as the best first crawler.

D.Focused Crawler:

Focused crawler is based on the hypertext classifier which was developed by chakrabarti et al [17, 19]. Focused crawler has three main components: a classifier which makes relevance judgments on pages crawled to decide on link expansion, a distiller which determines a measure of centrality of crawled pages to determine visit priorities, and a crawler with dynamically reconfigurable priority controls which is governed by the classifier and distiller. The focused crawler aims at providing a simpler alternative for overcoming the issue that immediate pages that are lowly ranked related to the topic at hand. The idea is to recursively execute an exhaustive search up to a given depth, starting from the relatives' of a highly ranked page.

E. Hidden Web Crawlers:

A lot of data on the web actually resides in the database and it can only be retrieved by posting appropriate queries or by filling out forms on the web. Recently interest has been focused on access of this kind of data called "deep web" or "hidden web". Current day crawlers' crawl only publicly indexable web (PIW) i.e., set of pages which are accessible by following hyperlinks ignoring search pages and forms which require authorization or prior registration. In reality they may ignore huge amount of high quality data, which is hidden behind search forms.

F.Parallel Crawlers:

As the size of the Web grows, it becomes more difficult to retrieve the whole or a significant portion of the Web using a single process. Therefore, many search engines often run multiple processes in parallel to perform the above task, so that download rate is maximized. This type of crawler is known as a parallel crawler.

G.Distributed Web Crawler:

This crawler runs on network of workstations. Indexing the web is a very challenging task due to growing and dynamic nature of the web. As the size of web is growing it becomes mandatory to parallelize the process of crawling to finish the crawling process in a reasonable amount of time. A single crawling process even with multithreading will be insufficient for the situation. In that case the process needs to be distributed to multiple processes to make the process scalable. It scales up to several hundred pages per second. The rate at which size of web is growing it is imperative to parallelize the process of crawling. In distributed web crawler a URL server distributes individual URLs to multiple crawlers, which download web pages in parallel. The crawlers then send the downloaded pages to a central indexer on which links are extracted and sent via the URL server to the crawlers. This distributed nature of crawling process reduces the hardware requirements and increases the overall download speed and reliability [2]. FAST Crawler [20] is a distributed crawler, used by Fast Search & Transfer.

V CRAWLING ALGORITHMS

There are many crawling algorithms available till date. The basic steps which are involved in the working of the crawler are:-

- Removing a URL from the URL list.
- Determining the IP address of its host name.
- Downloading of the related documents.
- Extracting any links available in the documents.

Features of a Good Crawling Algorithm

Speed

A simple crawler generally fetches 86,400 pages per day. In this way in order to fetch 20 billion pages 634 years are required. A large number of machines are required for the crawling purpose. An efficient mechanism should be followed in order to increase the crawling rate.

Politeness

Crawling algorithms should be designed in such a way that only one request is send to the server at a time. For this purpose, a politeness delay needs to be inserted between the requests. This will help reduce the risks.

Excluded Content

The site's robot.txt file needs to be fetched before fetching a page from the site so that it can be determined whether the web master has specified about how much file can be crawled [9].

Duplicate Content

Crawlers should be able to recognize and eliminate duplicate data available on different URLs. Methods like checksum, visitor counter, fingerprinting etc. are needed for this purpose.

Continuous Crawling

Carrying out full crawling after regular intervals is not a beneficial approach to follow. This results in low-value and static pages.

Spam Rejection

A crawler should be able to reject links to URLs on the current blacklist and can lower down the priority of pages that are linked to or from blacklisted sites.

Basic Crawling Approaches

A. Blind Traversing Approach

Firstly a seed URL is decided and the crawling process is applied. This process is called blind as there is particular method for selecting the next URL from the frontier. Breadth-First search is a very common example of this approach.

- Breadth-First Algorithm

It is the simplest crawling strategy. It was designed in 1994. It uses the frontier as the FIFO queue and crawls the links in the order in which they are encountered. It is basically used as a baseline crawler. Main drawback of this approach is that it traverses the URLs in the order in which they are entered into the frontier. It is good to implement this approach if the numbers of pages are less. In real life a lot of useless links are produced by useless pages which results in the wastage of time and memory of the frontier. Therefore a useful page should always be selected from the frontier.

B. Best First Heuristic Approach

This was developed in 1998 to overcome the problems of blind traversing approach. The links are selected from the frontier on the basis of some estimation, score or priority. Always the best available link is opened and traversed. Various mathematical formulas are also used.

- Naïve Best First Algorithm

It uses a relevance function to compute the lexical similarity between the desired keywords and each page and associate the value with the frontier. This approach always traverses the best available link. Priority queues are used as the data structure [9].

- Page Rank Algorithm

This was proposed by Brin and Page [8]. The page rank of a page represents the probability that a random surfer will be on that page at any given time. The page rank of a page is calculated as a sum of the page ranks of all pages linked to it, divided by the number of links on each of these pages.

- Fish Search Algorithm

Static search does not guarantees valid results but dynamic search do. An example of dynamic search is fish search algorithm. This algorithm considers that relative documents contain further relevant neighbouring documents. It treats the Internet as the directed graph in which the web pages are the nodes and the hyperlinks are the edges. Therefore, a directed graph needs to be traversed. A list of URLs to be searched is maintained in which URL with high priority will be at first place. It will be searched first and then its relative pages will be searched. The maintenance of the URLs is a key point of this algorithm [9].

VI USES OF WEB CRAWLING

There are various uses of web crawlers:

Crawlers can also be used for automating maintenance tasks on a Web site, such as checking links or validating HTML code.

Crawlers can be used to gather specific types of information from Web pages, such as harvesting e-mail addresses (usually for spam).

Search engines frequently use web crawlers to collect information that is available on public web pages. They collect data so that when Internet surfers enter a search term on their site, they can quickly provide the surfer with relevant web sites.

Linguists use web crawlers to perform a textual analysis. They comb the Internet to determine what words are commonly used today.

Crawlers have been used in biomedical applications such as finding the relevant literature on a gene [18].

Market researchers use a web crawler to determine and assess trends in a given market. After studying the trends they can make recommendation systems.

VII EXAMPLES OF WEB CRAWLER

A. RBSE

This was developed by Einchmann in 1994. It is generally based on two programs; spider and mite. Spider maintains a queue in a relational database and mite is a modified ASCII browser used to download pages from the web.

B. WebCrawler

This was developed by Pinkerton in 1994. It is the first publicly available full-text index of a subset of the web.

C.World Wide Web Worm

This was developed by McBryan in 1994. It is used for building simple index of document titles and URLs.

D.Google Crawler

This was developed by Brin and Page in 1998. It is used for full text indexing and for URL extraction.

E.Mercator

This was developed by Hayden and Najork in 1999. It is written in java and is a distributed crawler. It consists of protocol modules and process modules.

F.Web Foundation

This was developed by Edwards et al in 2001. It is similar to Mercator but written in C++. It consists of a controller machine and an ant machine. A non-linear programming method is used to solve the equation system.

G.WebRACE

This was developed by Zeinalipour-Yasti and Dikaiakos in 2002. It is a crawling and caching module developed in java.

UbiCrawler

This was developed by Boldi et al in 2004. It is written in java and has no central process.

I. Open Source Web Crawlers

- *Datapark Search*

It is a crawler and search engine released under the general public license.

- *GNU Wget*

It is a command line operated crawler written in C. It is used to mirror web and FTP sites.

- *Heritrix*

It is designed for archiving periodic snapshots of a large portion of the web. It is written in java.

- *WIRE- Web Information Retrieval Environment*

It is written in C++ and released under GPL.

- *YaCy,*

It is a free distributed search engine, built on principles of peer-to-peer networks.

- *GRUB*

It is an open source distributed search crawler that Wikia Search (http://wikiasearch.com) uses to crawl the web.

- *ICDL Crawler*

It is a cross-platform web crawler which is used to crawl Web sites based on Web-site Parse Templates using computer's free CPU resources only.

VIII CHALLENGES IN WEB CRAWLING

Scale

The web is growing at a very large scale day by day. For the crawlers to achieve to broad coverage and good performance, it needs to give very high throughput. This led to the creation of a large number of engineering based problems. To overcome these problems, the companies need to employ a large number of computers which may count to thousand and almost a dozen of high speed network links.

Content Selection Tradeoff

There are a number of crawlers which provide high throughput but are unable to crawl the whole web and cope up with the changes. The goal of crawling is to acquire content with higher value more quickly and gather information which contains all the reasonable content. Crawlers should ignore all the irrelevant, redundant and malicious content.

Social Obligations

Crawlers should follow safety mechanisms in order to avoid a denial-of-service attack. Crawlers should establish a good coordination with the different websites for which they work.

Adversaries

There are some content providers which try to inject useless content into the corpus of the crawler. Such types of activities are motivated by financial incentives like misdirecting Traffic to commercial websites [1].

Copyright

Crawlers ostensibly do something illegal: they make permanent copies of copyright material (web pages) without the owner's permission. Copyright is perhaps the most important legal issue for search engines. This is a particular problem for the Internet Archive (http://www.archive.org/), which has taken the role of storing and making freely available as many web pages as possible.

Privacy

For crawlers, the privacy issue appears clear-cut because everything on the web is in the public domain. Web information may still invade privacy if it is used in certain ways, principally when information is aggregated on a large scale over many web pages [3].

Cost

Web crawlers may incur costs for the owners of the web sites crawled by using up their bandwidth allocation. There are many different web hosts, providing different server facilities, and charging in different ways. Different hosts allow a wide variety of bandwidth. The consequences of exceeding the bandwidth results in excess of cost to be paid to get the website disabled [4].

IX FUTURE SCOPE OF WEB CRAWLING

Already a lot of research is going on in the field of web data extraction techniques. In future work can be done to improve the efficiency of algorithms. Also, the accuracy and timeliness of the search engines can also be improved. The work of the different crawling algorithms can be extended further in order to increase the speed and accuracy of web crawling [9].

A major open issue for future work about the scalability of the system and the behavior of its components. This could probably be best done by setting up a simulation test bed, consisting of several workstations, that simulates the web using either artificially generated pages or a stored partial snapshot of the web [14].

X CONCLUSION

Web crawlers are an important aspect of all the search engines. They are the basic component of all the web services so they need to provide high performance. Data manipulation by the web crawlers covers a wide area. Building an effective web crawler to solve different purposes 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. A number of crawling algorithms are used by the search engines. A good crawling algorithm should be implemented for better results and high performance.

REFERENCES

[1] Christopher Olston and Marc Najork (2010), "Web Crawling", now the essence of knowledge, Vol. 4, No. 3 (2010) 175–246.

[2] DhirajKhurana, Satish Kumar (2012), "Web Crawler: A Review", International Journal of Computer Science & Management Studies, Vol. 12, Issue 01, ISSN: 2231 –5268.

[3] Mohit Malhotra (2013), "Web Crawler And It's Concepts".

[4] Subhendukumarpani, Deepak Mohapatra, BikramKeshariRatha (2010), "Integration of Web mining and web crawler: Relevance and State of Art", International Journal on Computer Science and Engineering Vol. 02, No. 03, 772-776.

[5] Nemeslaki, András; Pocsarovszky, Károly (2011), "Web crawler research methodology", 22nd European Regional Conference of the International Telecommunications Society.

[6] Shkapenyuk V. and Suel T. (2002), "Design and Implementation of a high-performance distributed web crawler", In Proc. 18th International Conference on Data Engineering, pp. 357–368.

[7] Gautam Pant, Padmini Srinivasan and FilippoMenczer, "Crawling the Web", The University of Iowa, Iowa City IA 52242, USA.

[8] Sergey Brin and Lawrence Page, "The Anatomy of a Large Scale Hypertexual Web Search Engine", Computer Science Department, Stanford University, Stanford.

[9] Sandeep Sharma (2008), "Web-Crawling Approaches in Search Engines", Thapar University, Patiala.

[10] Minas gjoka (2010), "Measurement of Online Social Networks", university of california, Irvine.

[11] Salvatore A. Catanese, Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, Alessandro Provetti (2011), "Crawling Facebook for Social Network Analysis Purposes", WIMS'11, May 25-27, 2011 Sogndal, Norway, ACM 978-1-4503-0148-0/11/05.

[12] Ari Pirkola (2007), "Focused Crawling: A Means to Acquire Biological Data from the Web", University of Tampere Finland, ACM 978-1-59593-649-3/07/09.

[13] Priyanka-Saxena (2012), "Mercator as a web crawler", International Journal of Computer Science Issues, Vol. 9, Issue 1, No 1, ISSN: 1694-0814.

[14] Vladislav Shkapenyuk, Torsten Suel, "Design and Implementation of a High-Performance Distributed Web Crawler", NSF CAREER Award, CCR-0093400.

[15] Raja Iswary, Keshab Nath (2013), "Web Crawler", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 10, ISSN: 2278-1021.

[16] Allan Heydon and Marc Najork, "Mercator: A Scalable, Extensible Web Crawler", Compaq Systems Research Center

[17] S. Chakrabarti. *Mining the Web*. Morgan Kaufmann, 2003.

[18] P. Srinivasan, J. Mitchell, O. Bodenreider, G. Pant, and F. Menczer. Web crawling agents for retrieving biomedical nformation. In *ETTAB: Agents in Bioinformatics*, Bologna, Italy, 2002

[19] S. Chakrabarti, M. van den Berg, and B. Dom. Focused Crawling:Anewapproach to topicspecific Web resource discovery. *Computer Networks*, 31(11–16):1623–1640, 1999.

[20] K. M. and Michelsen, R. (2002). Search Engines and Web Dynamics. Computer Networks, vol. 39, pp. 289–302, June 2002. .