ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

Auto-Explore the Web – Web Crawler

Soumick Chatterjee[1], Asoke Nath[2]

M.Sc. Student, Dept. of Computer Science, St. Xavier's College (Autonomous), Kolkata, India[1]

Associate Professor, Dept. of Computer Science, St. Xavier's College (Autonomous), Kolkata, India[2]

ABSTRACT: World Wide Web is an ever-growing public library with hundreds of millions of books without any central management system. Finding a piece of information without a proper directory is like finding a middle in a haystack. Various search engines solve this problem by indexing an amount of the complete content that is available in the internet. For accomplishing this job, search engines use an automated program, known as a web crawler. The most vital job of the web is information retrieval, that too with proper efficiency. Web Crawler helps to accomplish that, by helping search indexing or by helping in making archives. Web Crawler automatically visits all the available links which is further indexed. But, usage of web crawler is not limited to only search engines, but they can also be used for web scrapping, spam filtering, identifying unauthorized use of copyrighted content, identifying illegal and harmful web activities etc. Web Crawler faces various challenges while crawling deep web content, multimedia content etc. Various crawling techniques and various web crawlers are available and discussed in this paper.

KEYWORDS: Web Crawler, Web Search, Web Indexing, Search Engines, WWW, Crawling techniques, Crawler algorithms

I. INTRODUCTION

The web is just like an ever-growing public library with hundreds of millions of books without a proper central management system. So, it is essential to index the web properly to retrieve required information efficiently. Currently World Wide Web provides a huge source of information, that can easily be accessed using any search engine, to extract valuable information out of this haystack. Searching all the web servers at real-time is not a realistic approach. So, all the pages need to be indexed properly beforehand. Google, Bing and other various search engines tries to index as much as possible. Search engines accomplish this in two main phases – Crawling and Indexing.

Everyday many new webpages are added and information present in existing websites gets changed. [1]. Due to the extremely large number of pages present on Web, the search engine depends upon crawlers for the collection of required pages [2]. For this reason and many other an automated program known as Web Crawler is used.

A Web Crawler, also known as a Web Spider or simply as a bot, is an internet based program that systematically browses the World Wide Web. A web crawler can identify all links in each page and then recursively continues. Web Crawlers also can extract content, that can be used for web scrapping. Web Crawler is used for many web systems starting from a simple program for just website backup to a major search engines like Google, Microsoft Bing etc. These search engines use this routinely to visit those enormous number of web pages, which are then indexed and made available upon user's search request. For that reason, characteristics of the crawler used such as coverage, refresh rate etc. directly effects the quality of search result returned. Apart from just using it just for search engines, web crawlers have a wide array of usage web data mining and extraction, social media analysis, detection of web spam and fraudulent web sites, finding unauthorized use of copyrighted content (music, videos, texts, etc.), identification of illegal and harmful web activities (e.g., terrorist chat rooms), etc [3].

II. WEB CRAWLER

A. *History of Search Engines & Web Crawlers*

The concept of hypertext and a memory extension really brought into life in July of 1945 by Vannevar Bush's As We May Think, that was published in The Atlantic Monthly where he urged scientists to work together to help build a body

DOI: 10.15680/IJIRCCE.2017. 0504006

of knowledge for all man-kind. He not only was a firm believer in storing data, but he also believed that if the data source was to be useful to the human mind we should have it represent how the mind works to the best of our abilities. He then pro-posed the idea of a virtually limitless, fast, reliable, extensible, associative memory storage and retrieval system. He named this device a memex [4].

Gerard Salton was the father of modern search technology developed at Harvard and Cornell, the SMART (Salton's Magic Automatic Retriever of Text) information retrieval system. Followed by Ted Nelson's project Xanadu and, he coined the term hypertext. His goal with Project Xanadu was to create a computer network with a simple user interface that solved many social problems like attribution [4].

Advanced Research Projects Agency Network or ARPANet which eventually lead to the Internet, was established way back in 1969. In 1990, the first search engine came into existence known as “Archie” - shortened from “Archives” which downloaded the directory listings from specified public anonymous FTP (File Transfer Protocol) sites into local repository, in a specified frequency. In 1991, “Gopher” protocol was created, that indexes plain text documents. “Jughead” & “Veronica” was two of the popular search engines created using this protocol. With the introduction of the World Wide Web by Sir Tim Berners-Lee in 1991, numerous of these Gopher based sites changed to web sites that were properly linked by HTML links [5].

In 1993, the first ever web crawler – “World Wide Web Wanderer” or simply Wanderer, was created by MIT to measure the size of the World Wide Web [6]. Later this was further developed to retrive URLs that were then stored in a database called Wandex, the first web search engine [7] followed by “Aliweb”. This index contains a list of URLs and user written keywords and descriptions.

Initially, crawlers caused much controversy because it can crawl any link it gets, increases network overhead as well as crawls unwanted links, but this issue was later resolved in 1994 with the introduction of the Robots Exclusion Standard [8] which allowed web site administrators to block crawlers from retrieving part or all of their sites [5]. Also in the same year, first official “WebCrawler” was launched [9], which was the first “full text” crawler and search engine. The “WebCrawler” permitted the users to explore the web content of documents rather than the keywords and descriptions written by the web administrators, reducing the possibility of confusing results and allowing better search capabilities. During that period, commercial search engines such as Yahoo, Altavista started coming into existence. In the late 1997, Google was launched, capturing the majority of the market share, thanks to its simple user friendly interface and unbiased search results that were reasonably relevant, and a lower number of spam results [10]. These last two qualities were due to Google's use of the PageRank [11] algorithm and the use of anchor term weighing [12].

While early crawlers dealt with relatively small amounts of data, modern crawlers, such as the one used by Google, need to handle a substantially larger volume of data due to the dramatic enhance in the amount of the Web [5].

B. *Working Principle*

The structure of the World Wide Web can be viewed as a directed graph, where everything is present in a hierarchy. When a page is visited, it contains links to other pages. While viewing the Internet as a directed graph, web pages can be considered as nodes and the hyperlinks can be considered as edges. So, we can summarize the search operation as traversing a directed graph. Following this linked hierarchical structure, a web crawler can start with a given page and then visit to all those pages whose links are given in that page. For this way of traversing or crawling the graphical net like structure, they are also known as spiders, and be-cause this process is automated, these web crawlers are also known as robots. Web crawlers are designed to retrieve Web pages and insert them to local repository. Crawlers are basically used to create a replica of all the visited pages that are later processed by a search engine that will index the downloaded pages that help in quick searches [13].

The working of Web crawler stars with an initial set of URLs known as seed URLs. Then it downloads all the web pages for the seed URLs and extract new links present in those downloaded pages. The retrieved web pages are then stored and well indexed on the storage area so that by the help of these indexes they can later be retrieved. The extracted URLs from the downloaded page are then matched with the existing ones to know whether they are already indexed or downloaded or not. If they are not, the URLs are again assigned to web crawlers for further crawling. This process is continued recursively till there are no new links to be explored [3].

Figure 1 tries to depict how a web crawler works in a very simple manner. Digging a little deep, the scheduler and the queue together known as “Crawler Frontier” which stores the list of URLs to visit, a Multi-threaded downloader, also known as “Page Downloader” which downloads pages from the World Wide Web and a storage, known as “Web Repository” which receives the web pages received from the crawler and stores them in a database.

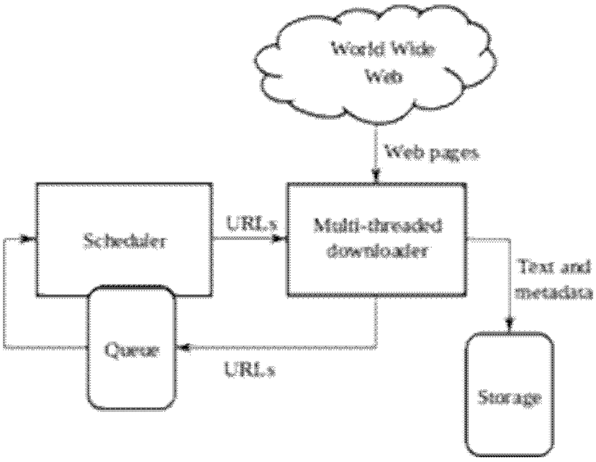


Fig. 1. Simple Working of a Web Crawler

Crawler frontier: It contains the list of unvisited URLs. The list is set with seed URLs which may be delivered by a user or another program [14]. Simply it’s just the collection of URLs. The working of the crawler starts with the seed URL. The crawler retrieves a URL from the frontier which contains the list of unvisited URLs. The page corresponding to the URL is fetched from the Web, and the unvisited URLs from the page are added to the frontier [15]. The cycle of fetching and extracting the URL continues until the frontier is empty or some other condition causes it to stop. The extracting of URLs from the frontier based on some prioritization scheme [11] [16].

Page downloader: The main work of the page down-loader is to download the page from the internet corresponding to the URLs which is retrieved from the crawler frontier. For that, the page downloader requires a HTTP client for sending the HTTP request and to read the response. There should be timeout period needs to set by the client to ensure that it will not take unnecessary time to read large files or wait for response from slow server. In the actual implementation, the HTTP client is restricted to only download the first 10KB of a page. [15] [11].

Web repository: It use to stores and manages a large pool of data "objects," [17] in case of crawler the object is web pages. The repository stores only standard HTML pages. All other media and document types are ignored by the crawler [18]. It is theoretically not that different from other systems that store data objects, such as file systems, database management systems, or information retrieval systems. However, a web repository does not need to provide a lot of the functionality like other systems, such as transactions, or a general directory naming structure [17]. It stores the crawled pages as distinct files. And the storage manager stores the up-to-date version of every page retrieved by the crawler [13].

C. *Basic Crawling Algorithm*

The basic steps a web crawler performs are as follows:

1. Insert seed URL or URLs to the frontier
2. Select a URL from the frontier based on the specified policies.
3. Fetch the corresponding web page
4. Parse the retrieved web page to extract the URLs.
5. Add all the unvisited links to the frontier.

Repeat from step 2 till the frontier is empty.

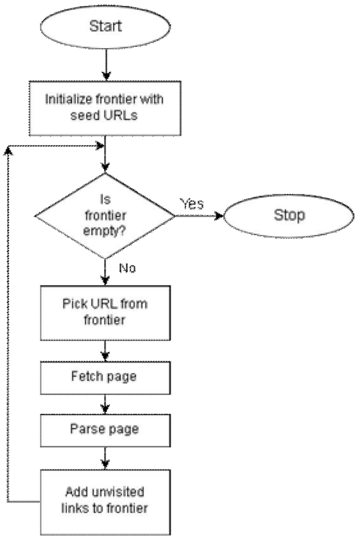


Fig. 2. Basic flow chart of web crawling process

D. *Web Crawling Algorithm*

Web crawlers use various search algorithms to perform the crawling process. Some of the search algorithms are as follows:

- Breadth First Search
- Best First Search
- Fish Search
- A* Search
- Adaptive A* Search

Breadth First Search: Breadth First Search is the simplest form of crawling algorithm. It starts with a link and keeps on traversing the connected links without taking into consideration any knowledge about the topic. Since it does not consider the relevancy of the path while traversing, it is also known as the Blind Search Algorithm. It is considered to give lower bound on efficiency for any intelligent traversal algorithm [19].

Best First Search: Best First Search is a heuristic based search algorithm. In this approach, relevancy calculation is done for each link and the most relevant link, such as one with the highest relevancy value, is fetched from the frontier. Thus, every time the best available link is opened and traversed [19].

Fish Search: Fish Search is a dynamic heuristic search algorithm. It works on the intuition that relevant links have relevant neighbours; hence it starts with a relevant link and goes deep under that link and stops searching under the links that are irrelevant. The key point of Fish Search algorithm lies in the maintenance of URL order [19].

A* Search: A* uses Best First Search. It calculates the relevancy of each link and the difference between expected relevancy of the goal web-page and the current link. The sums of these two values serve as the measure for selecting the best path [19].

Adaptive A* Search: Adaptive A* Search works on informed heuristics to focus its searches. With its each iteration, it updates the relevancy value of the page and uses it for the next traversal. The pages are updated for log(Graph Size) times, (after log(Graph Size) times the overhead of updating is much more than the improvement that can be achieved in getting more relevant pages) and then normal A* traversal is done [19].

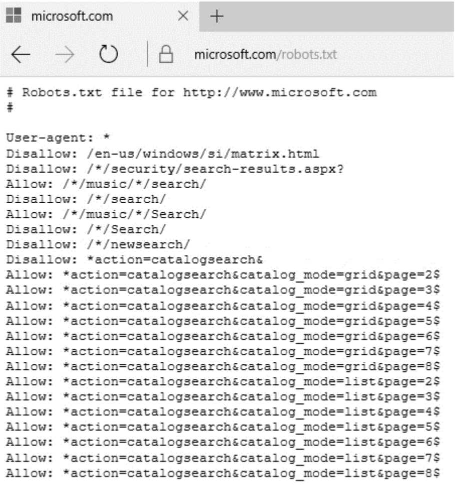
E. *Crawling Polices*

A web crawler is as good as the enforced policies. The behaviour of a web crawler is an outcome of combination of various policies. Such as:

- Selection Policy: States the web pages that to be downloaded
- Re-visit Policy: States when to check for changes to the pages, after what duration.
- Politeness Policy: States how to avoid overloading Web sites, enforcement of Robot Exclusion Standards.
- Parallelization policy: states how to coordinate distributed Web crawlers.

F. *Robots.txt*

This comes under the Politeness policies. To avoid over-loading servers and to stop crawler from crawling unintended places, this mechanism exists. Under this mechanism, public sites not wishing to be crawled or parts of the site not wishing to be crawled should make this known to the crawling agent. For instance, including a robots.txt file can request bots to index only parts of a website, or nothing at all.



```
# Robots.txt file for http://www.microsoft.com
#

User-agent: *
Disallow: /en-us/windows/si/matrix.html
Disallow: /*/security/search-results.aspx?
Allow: /*/music/*/search/
Disallow: /*/search/
Allow: /*/music/*/Search/
Disallow: /*/Search/
Disallow: /*/newsearch/
Disallow: *action=catalogsearch&
Allow: *action=catalogsearch&catalog_mode=grid&page=2$
Allow: *action=catalogsearch&catalog_mode=grid&page=3$
Allow: *action=catalogsearch&catalog_mode=grid&page=4$
Allow: *action=catalogsearch&catalog_mode=grid&page=5$
Allow: *action=catalogsearch&catalog_mode=grid&page=6$
Allow: *action=catalogsearch&catalog_mode=grid&page=7$
Allow: *action=catalogsearch&catalog_mode=grid&page=8$
Allow: *action=catalogsearch&catalog_mode=list&page=2$
Allow: *action=catalogsearch&catalog_mode=list&page=3$
Allow: *action=catalogsearch&catalog_mode=list&page=4$
Allow: *action=catalogsearch&catalog_mode=list&page=5$
Allow: *action=catalogsearch&catalog_mode=list&page=6$
Allow: *action=catalogsearch&catalog_mode=list&page=7$
Allow: *action=catalogsearch&catalog_mode=list&page=8$
```

Fig. 3. Robots.txt file of Microsoft.com

In the figure 3, a screenshot of the current robots.txt file of Microsoft.com is given where rules for the crawlers are mentioned. Disallow rule notifies the crawler what are the locations that need not be crawled.

G. *Content awareness: Pull vs. Push model*

Content awareness (or "content collection") is usually either a push or pull model. In the push model, a source system is integrated with the search engine in such a way that it connects to it and pushes new content directly to its APIs. This model is used when real-time indexing is important. In the pull model, the software gathers content from sources using a connector such as a web crawler or a database connector. The connector typically polls the source with certain intervals to look for new, updated or deleted content [20].

Pull model is preferred over push model. In this model, site owners are known as Web Content Provider and crawler operators are known as Web Aggregators. Aggregator pulls content, it is not pushed to aggregators. Pull is just easier for both parties. No 'agreement' be-tween provider and aggregator [3].

Pull model also has certain disadvantages: avoiding redundant requests from crawlers, more control over the content from providers

III. CRAWLING TECHNIQUES

A. *General Purpose Crawling*

A general-purpose Web Crawler collects as many pages as it can from a set of URLs and their links. In this, the crawler can fetch many pages from different locations. General purpose crawling can slow down the speed and network bandwidth because it is fetching all the pages [3].

B. *Focused Crawling*

Focused Crawler is the Web crawler that tries to download pages that are related to each other [18] [21]. It collects documents which are specific and relevant to the given topic [5] [22]. It is also known as a Topic Crawler because of its way of working [21] [23]. The focused crawler determines the following – Relevancy, Way forward. It determines how far the given page is relevant to the particular topic and how to proceed for-ward. The benefits of focused web crawler is that it is economically feasible in terms of hardware and net-work resources, it can reduce the amount of network traffic and downloads [24]. The search exposure of focused web crawler is also huge [25] [26].

C. *Incremental Crawling*

A traditional crawler, in order to refresh its collection, periodically replaces the old documents with the newly downloaded documents. On the contrary, an incremental crawler incrementally refreshes the existing collection of pages by visiting them frequently; based upon the estimate as to how often pages change [18]. It also exchanges less important pages by new and more important pages. It resolves the problem of the freshness of the pages. The benefit of incremental crawler is that only the valuable data is provided to the user, thus net-work bandwidth is saved and data enrichment is achieved [27] [28].

D. *Distributed Crawling*

Distributed web crawling is a distributed computing technique. Many crawlers are working to distribute in the process of web crawling, in order to have the most coverage of the web. A central server manages the communication and synchronization of the nodes, as it is geographically distributed [25]. It basically uses Page rank algorithm for its increased efficiency and quality search. The benefit of distributed web crawler is that it is robust against system crashes and other events, and can be adapted to various crawling applications [29].

E. *Parallel Crawling*

Multiple crawlers are often run in parallel, which are referred as Parallel crawlers. A parallel crawler consists of multiple crawling Processes [30] called as C-procs which can run on network of workstations [31]. The Parallel crawlers depend on Page freshness and Page Selection [32]. A Parallel crawler can be on local network or be distributed at geographically distant locations [25]. Parallelization of crawling system is very vital from the point of view of downloading documents in a reasonable amount of time [31].

IV. APPLICATIONS

A. *Web Search Engines*

- One of the main application of crawlers is web search engines
- Search engines periodically visits websites to index new websites and to update changes in existing websites.
- Helps users to obtain information easily.

B. *Web Archiving*

- Typically done using batch crawlers
- Digital preservation
- The biggest: Internet Archive

C. *Vertical Search Engines*

- Data aggregating from many sources on certain topic
- E.g., apartment search, car search

D. *Web Data Mining*

- Typically done using Focused Crawlers.
- For example, opinion mining
- Or digests of current happenings on the Web (e.g., what music people listen now)

E. *Web Monitoring*

- Monitoring sites/pages for changes and up-dates

F. *Detection of malicious web sites*

- Typically, a part of anti-virus, firewall, search engine, etc. service
- Building a list of such web sites and inform a user about potential threat of visiting such

G. *Web site/application testing*

- Crawl a web site to check a navigation through it, validity the links, etc.
- Regression/security testing a rich internet application (RIA) via crawling
- Checking different application states by simulating possible user interaction events (e.g., mouse click, time-out)

H. *Copyright violation detection*

- Crawl to find (media) items under copyright or links to them
- Regular re-visiting 'suspicious' web sites, forums, etc.

I. *Detection of illegal activities*

- Crawl to find terrorist chat rooms
- Finding web pages engaged in illegal activities such as selling controlled substance.

J. *Web Scraping*

- Extracting particular pieces of information from a group of typically similar pages
- When API to data is not available
- Interestingly, scraping might be more preferable even with API available as scraped data often more clean and up-to-date than data-via-AP

K. *Web Mirroring*

- Copying of web sites
- Often hosting copies on different servers to ensure constant accessibility

V. CHALLANGES

A. *Collaborative Crawling*

One of the major issues during this is that lots of redundant crawling. To get a data, often on a specific subject, need to crawl broadly and only a small subset is actually used. In a way, it is reconsidering pull/push model of content delivery on the Web [3].

B. *Deep Web & Dark Web Crawling*

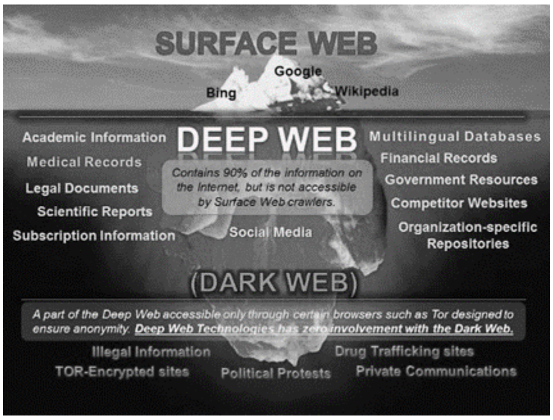


Fig. 4. Structure of the World Wide Web

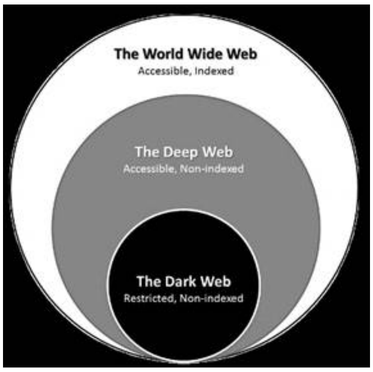


Fig. 5. Surface Web vs Deep Web vs Dark Web

The whole website can be categorized into three categories – Surface Web, Deep Web & Dark Web. Surface Web is that small portion of the WWW, which the search engines can index. It consists of around 10% of the complete Internet. Web Crawlers can easily crawl this space. Next comes the Deep Web. This is not necessarily the illegal part of the web. This consists of 90% of the Internet, includes all unindexed data. Certain portion of the deep web is known as dark web. This is also unindexed. Also, this portion has restricted access. For example, many sites are TOR-Encrypted and can only be accessed from the Tor Network. Web crawlers cannot crawl the deep or the dark web due to robot exclusion standard and other policies [33].

Bots, which doesn't honour the politeness policy, can still crawl the deep web & dark web. Dark Web is where user can operate without been tracked, maintaining total anonymity. The Dark Web is much smaller than the Deep Web and is made up of all different kinds of websites that sell drugs, weapons and even hire assassins. These are hidden networks avoiding their presence on the Surface Web, and its URLs are tailed up with .onion typically. 'The Onion Browser,' referred to as Tor can be used to access such web pages. There are various dark and deep web search engines in the Tor network that can be used to search in this segment of the Internet. So, eventually these search engines perform deep web crawling [34].

C. *Crawling Multimedia Content*

Typical web crawlers are text based. So, crawling multimedia content is not possible by them. Also, if a crawler performs multimedia crawling, bigger load is imposed on web sites since files are bigger. Also, resolving duplicates is also a big issue with multimedia. But multimedia crawling is important, as this includes more copyright issues.

DOI: 10.15680/IJIRCCE.2017. 0504006

While typical web crawlers use metadata of a multimedia content, which typically given in the “alt” tag, there are other approaches also exists. API-directed crawling is one of those methods. Using two separate crawlers, one for regular web content and another API-directed crawler can be used for better performance.

D. *Other Crawling Challenges*

Ordering policy:

- Resources are limited, while number of pages to visit essentially infinite
- Decision should be done based on URL itself
- PageRank-like metrics can be used to resolve this issue.
- More complicated in case of incremental crawls

Focused crawling:

- To resolve problems during focused crawling, avoid links leading to content out of the topic of interest
- Setting a good seed is a challenge

Re-visiting policy: It is difficult to set a perfect re-visiting policy as by the rate contents are updated of different web pages are different. Revisit very late may miss out latest content, on the other hand, revisiting very soon will increase unnecessary network traffic.

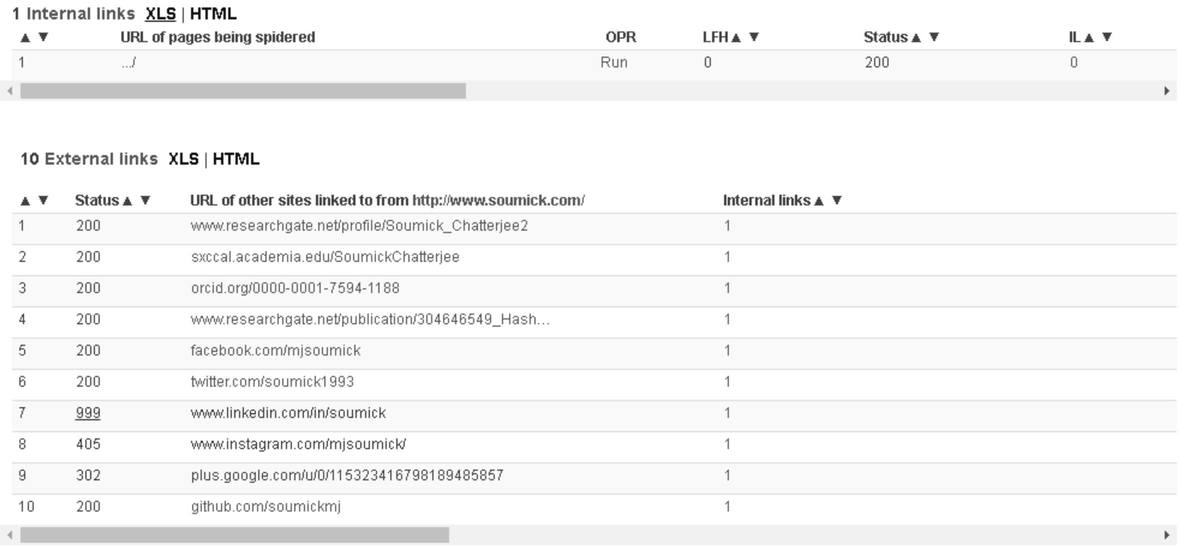
Generating good seed URLs: A crawler strats its work with seed URLs. If proper seed URLs are not provided to the crawler, then it won’t be able to retrieve required set of web pages. Generating good seed URLs is always a challenge as no general algorithm exists for this purpose. This issue is even severe during focused crawling.

Avoiding redundant content:

- Avoid visiting duplicate pages (different URLs leading to identical or near-identical content) - Near-duplicates might be very tricky (think of a news item propagation on the Web)
- Avoid crawler traps
- Avoid useless content (i.e., web spam)

VI. Simulation and Results

For simulation purposes, two web pages have been taken. The first webpage is the webpage of one of the authors: - http://www.soumick.com. This website is a single page website and contains all the subpages like intro, work etc. as divs in a single main page. So, there is no internal links that to be crawled.



1 Internal links **XLS** | **HTML**

▲ ▼	URL of pages being spidered	OPR	LFH ▲ ▼	Status ▲ ▼	IL ▲ ▼
1	.../	Run	0	200	0

10 External links **XLS** | **HTML**

▲ ▼	Status ▲ ▼	URL of other sites linked to from http://www.soumick.com/	Internal links ▲ ▼
1	200	www.researchgate.net/profile/Soumick_Chatterjee2	1
2	200	sxccal.academia.edu/SoumickChatterjee	1
3	200	orcid.org/0000-0001-7594-1188	1
4	200	www.researchgate.net/publication/304646549_Hash...	1
5	200	facebook.com/mjsoumick	1
6	200	twitter.com/soumick1993	1
7	999	www.linkedin.com/in/soumick	1
8	405	www.instagram.com/mjsoumick/	1
9	302	plus.google.com/u/0/115323416798189485857	1
10	200	github.com/soumickmj	1

Fig. 6. Internal and External links after scrawling soumick.com

As shown in figure 6, after soumick.com is crawled, it fetched only one internal link as this is a single page website and it fetched all the external links like research gate, academia, orcid, github etc. whatever was present in that website. But, as discussed earlier, the web crawler was unable to crawl multimedia contents such as photographs. This was a general-purpose crawling performed using the Breadth First Search algorithm following the pull model. During this crawling process, crawling was restricted to one level of external links. If this restriction would not have been placed, then the next step of crawling would have been performed by crawling those external links obtained. Here, the seed URL was soumick.com and all the URLs internal and external URLs fetched from it, was stored in the crawler frontier. Each unvisited links is fetched and then crawled. This example had only one internal link, and crawling external links were blocked, so the crawling process ended only after one iteration.

Second website taken for simulation was http://www.supernovatechlink.in. After crawling this website, 19 internal and 37 external links were found. Homepage of this website had certain links, such as company.html software.html contact.html those were stored in crawler frontier. Then again recursively those pages were crawled and further links like contact.aspx were found, that again stored in frontier for further crawling, which obtains link such as maps.google.co.in In this manner, a crawling starts from a set of seed URLs and then continues recursively until the terminating condition is reached – like in this example, terminating condition was if external link found, crawling should stop.

VII. VARIOUS CRAWLER BOTS

- **Search Bots:** There are various crawlers available. Among various popular search bots, Googlebot, which is used by google is definitely the most famous one, where Bingbot by Microsoft is at the second position
  - Googlebot – Search bot used by Google Search
  - Bingbot – Search bot used by Bing search engine by Microsoft
  - Slurp Bot – Yahoo Search results come from the Yahoo web crawler Slurp and Bing's web crawler Bingbot, as a lot of Yahoo is now powered by Bing.
  - DuckDuckBot – DuckDuckBot is the Web crawler for DuckDuckGo, a search engine that has become quite popular lately as it is known for privacy and not tracking the user. It now handles over 12 million queries per day.
- **Non-search Web Crawlers:** The application of Web Crawlers are not limited to search engines. As discussed earlier, web crawlers can be used for various other purposes, such as – for web scrapping, web archiving, web page ranking etc.
  - Facebook External Hit – Facebook allows its users to send links to interesting web content to other Facebook users. This bot allows to crawl and collect information about these links.
  - Google Plus Share – This is the Google Plus alternative to the Facebook External Hit. Google plus users can share links using +1 button.
  - Google Feedfetcher – This bot is used by Google to grab RSS or Atom feeds when users choose to add them to their Google homepage or Google Reader.
  - Alexa Crawler – Alexa periodically crawls web pages to provide them rankings using this bot.

VIII. CONCLUSION

Web crawlers are one of the most important pillars of the modern World Wide Web. From web search to data analysis – web crawler is crawling with importance. Internet brings a lot of information to the user, but there should be proper indexing in place to obtain those in-formation efficiently. Web Crawlers help search engines to perform this operation. Web Crawlers are designed to download web pages and store them in local repository. Crawlers typically creates a replica of all the visited pages, which can further be used by search engines for indexing purposes or by other programs for various requirements – web data analysis, web scrapping etc. There are various search bots and non-search bots are available, discussed here. The major objective of this paper is to enlighten readers about various aspects of web crawling, different bots available and various challenges and various proposed solutions to overcome them.

REFERENCES

1. Bharat Bhushan, Narender Kumar," Intelligent Crawling On Open Web for Business Prospects", IJCSNS International Journal of Computer Science and Network Security, VOL.12 No.6, June 2012
2. Keerthi S. Shetty, Swaraj Bhat and Sanjay Singh, "Symbolic Verification of Web Crawler Functionality and Its Properties", International Conference on Computer Communication and Informatics (ICCCI -2012), Coimbatore, INDIA, IEEE Conference Publications,2012
3. Current Challenges in Web Crawling - Denis Shestakov, in Proc. ICWE 2013, 2013, pp. 518-521
4. http://www.searchenginehistory.com/ Last accessed: 2:25 AM 4/14/2017
5. Web Crawler: A Review - Md. Abu Kausar, V. S. Dhaka, Sanjeev Kumar Singh - International Journal of Computer Applications - Volume 63– No.2, February 2013
6. M. K. Gray. World Wide Web Wanderer, 1996b. URL http://www.mit.edu/people/mkgray/net/ Last accessed: 2:25 AM 4/14/2017
7. W. Sonnenreich and T. Macinta. Web Developer.com, Guide to Search Engines. John Wiley & Sons, New York, New York, USA, 1998.
8. M. Koster. A Standard for Robot Exclusion, 1994b. URL http://www.robotstxt.org/wc/exclusion.html Last accessed: 2:25 AM 4/14/2017
9. B. Pinkerton. Finding What People Want: Experiences with the WebCrawler. In Proceedings of the Second International World Wide Web Conference, Chicago, Illinois, USA, Oct. 1994
10. Google. Google's New GoogleScout Feature Expands Scope of Search on the Internet, Sept. 1999. URL http://www.google.com/press/pressrel/pressrelease4.html Last accessed: 2:25 AM 4/14/2017
11. L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.
12. S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. In P. H. Enslow Jr. and A. Ellis, editors, WWW7: Proceedings of the Seventh International Conference on World Wide Web, pp. 107–117, Brisbane, Australia, Apr. 14–18 1998. Elsevier Science Publishers B. V., Amsterdam, The Netherlands. doi: http://dx.doi.org/10.1016/S01697552(98)00110-X.
13. Study of Web Crawler and its Different Types - Trupti V. Udapure, Ravindra D. Kale, Rajesh C. Dharmik - IOSR Journal of Computer Engineering (IOSR-JCE) - Volume 16, Issue 1, Ver. VI (Feb. 2014), PP 01-05 2011
14. Pant Gautam, Srinivasan Padmini, Menczer Filippo, "Crawling the Web" In Levene, Mark; Poulovassilis, Alexandra. Web Dynamics: Adapting to Change in Content, Size, Topology and Use. Springer. pp. 153-178. 2004
15. Gautam Pant, Padmini Srinivasan, "Learning to Crawl: Comparing Classification Schemes", ACM Transactions on Information Systems, Vol. 23, No. 4, October 2005, Pages 430–462.
16. Ioannis Avraam, Ioannis Anagnostopoulos, "A Comparison over Focused Web Crawling Strategies" 2011 Panhellenic Conference on Informatics, IEEE Conference Publications
17. Jun Hirai Sriram Raghavan Hector Garcia-Molina Andreas Paepcke, "WebBase : A repository of web pages"
18. Junghoo Cho and Hector Garcia-Molina. 2000a. "The evolution of the web and implications for an incremental crawler", In Proceedings of the 26th International Conference on Very Large Databases
19. Aviral Nigam. Web Crawling Algorithms. International Journal of Computer Science and Artificial Intelligence. Sept. 2014, Vol. 4 Iss. 3, PP. 63-67
20. https://www.information-management.com/news/understanding-content-collection-and-indexing Last accessed: 2:30 AM 4/14/2017
21. S.S. Dhenakaran1 and K. Thirugnana Sambanthan2, "WEB CRAWLER - AN OVERVIEW", International Journal of Computer Science and Communication Vol. 2, No. 1, January-June 2011, pp. 265-267
22. Shashi Shekhar, Rohit Agrawal and Karm Veer Arya, "An Architectural Framework of a Crawler for Retrieving Highly Relevant Web Documents by Filtering Replicated We Collections", 2010 International Conference on Advances in Computer Engineering, IEEE Conference Publications 2010
23. Gautam Pant, Padmini Srinivasan, "Learning to Crawl: Comparing Classification Schemes", ACM Transactions on Information Systems, Vol. 23, No. 4, October 2005, Pages 430–462
24. Debashis Hati, Biswajit Sahoo, Amritesh Kumar, "Adaptive Focused Crawling Based on Link Analysis", 2nd International Conference on Education Technology and Computer (ICETC),2010
25. Pavalam S. M., S. V. Kasmir Raja, Jawahar M., and Felix K. Akorli, "Web Crawler in Mobile Systems", International Journal of Machine Learning and Computing, Vol. 2, No. 4, August 2012
26. Manas Kanti Dey, Debakar Shamanta, Hasan Md Suhag Chowdhury, Khandakar Entenam Unayes Ahmed, "Focused Web Crawling: A Framework for Crawling of Country Based Financial Data", Information and Financial Engineering (ICIFE), IEEE Conference Publications, 2010
27. A. K. Sharma and Ashutosh Dixit, "Self Adjusting Refresh Time Based Architecture for Incremental Web Crawler" International Journal of Computer Science and Network Security, vol.8 no.12, 2008, pp. 349-354
28. Niraj Singhal, Ashutosh Dixit, and Dr. A. K. Sharma, "Design of a Priority Based Frequency Regulated Incremental Crawler", International Journal of Computer Applications (0975 – 8887) vol.1, no. 1, 2010, pp. 42-47.
29. Vladislav Shkapenyuk Torsten Suel "Design and Implementation of a High-Performance Distributed Web Crawler", CIS Department Polytechnic University Brooklyn, NY 11201
30. Junghoo Cho, Hector Garcia-Molina, "Parallel Crawlers", WWW2002, May 7–11, 2002, Honolulu, Hawaii, USA, ACM 1-58113449-5/02/0005
31. Shruti Sharma, A.K.Sharma, J.P.Gupta, "A Novel Architecture of a Parallel Web Crawler", International Journal of Computer Applications (0975 – 8887) Volume 14– No.4, January 2011
32. AH Chung Tsol, Daniele Forsali, Marco Gori, Markus Hagenbuchner, Franco Scarselli, "A Simple Focused Crawler" Proceeding 12th International WWW Conference 2003(poster), pp. 1.
33. https://danielmiessler.com/study/internet-deep-dark-web/#gs.DQC1_t0 Last accessed: 2:30 AM 4/14/2017
34. http://thehackernews.com/2016/02/deep-web-search-engine.html Last accessed: 2:30 AM 4/14/2017

BIOGRAPHY

Soumick Chatterjee is a M.Sc. Computer Science student from Department of Computer Science, St. Xavier's College (Autonomous), Kolkata, is currently in 4th Semester; interested in Web Technologies, Cross-platform development, Cryptography, Steganography, Image Processing with successful experience in Tech Entrepreneurship.

Dr. Asoke Nath is Associate Professor in the Department of Computer Science, St. Xavier's College (Autonomous), Kolkata. Dr. Nath is involved in research work in Cryptography and Network Security, Steganography, Green Computing, Mathematical modelling of social networks, Big data analytics, Cognitive Radio, Data Science, e-learning, MOOCs etc. He has published more than 185 papers in Journals and conference proceedings.