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 2, February 2017

A Survey on Web Crawler Approaches

Akshay Bharambe, Rishab Dey, Vishal Bahadurge, Bhavana Tanpure, Rashmi Ramteke

B. E Students, Dr. D. Y. Patil Institute of Engineering and Technology, Pimpri, Savitribai Phule Pune University, Pune Maharashtra, India

ABSTRACT: Data Mining has become more challenging in recent years due to the large size of data, and increased web security. Mining data from a web database differs from mining data from web sites because it is intended to collect specific data from a single web site. Collecting a very large data in a limited time is detected as a cyber-attack and will be banned from connecting into the web server [1]. To avoid this problem, the proposed crawling method is to mine web database faster and reliable than conventional web crawlers. The method used is to run multiple threads [2] from a single web crawler on a single computer and to distribute these threads into many publicly available proxy servers. This web crawler strategy highly increases the speed of mining and is more secure than using single thread of web crawler.

KEYWORDS: Focused web crawler;Proxy servers; Multi-threading

I. INTRODUCTION

Most publicly available digital data on the web is in an unstructured, or loosely structured, format. Thus, some processing is necessary in order to extract meaningful information. The very first step of this processing, that is constructing the data collection, is usually achieved using a web crawler. A web crawler accesses remote servers on the Internet in order to fetch, process and store web pages. The crawler starts from some initial seed URLs, store the corresponding web pages for later processing, and then extracts links to other web pages. Those links are themselves subsequently crawled.

In addition, data mining is becoming more challenging in recent years due to the large sizes of data, and increased web security. Mining data from a web database differs from mining data from web sites because it is intended to collect specific data from a single web site. Collecting a very large data in a limited time, is detected as a cyber- attack and will be banned from connecting into the web server. Different types of web crawlers are available. Also different strategies, approaches, architectures are there which can be used in these crawlers. This survey helps to summarize these crawlers and their architectures.

II. RELATED WORK

Harry T Yani Achsana, et.al. [1]. Focused web crawler is also called as specific web crawler. It is a web bot which mines specific data from web databases. Mined Data may be structured or unstructured because it is retrieved from database in web sites such as databases from social networks, forums, blogs, online libraries, online stores, or any web sites that use database to display their information. If we can collect the data from a web site, then we can retrieve the information and discover the knowledge contained in it.

Directly mining data from web site is fast process but there might be possibility of that web site monitor banned crawler. So to overcome this problem publicly available proxy server are used. To run Multiple crawler on single machine multithreading is used.

Speed of publicly available proxies is fluctuating from time to time. So in proposed system we maintain list of publicly available high speed proxies. Every time before a crawling website crawler fetches high-speed proxies. Crawler assigns single thread to each proxy. With the help of Proxies crawler starts crawling Seed URL and searches relevant hyperlinks of given keyword. Using Page Ranking algorithm crawler assigns rank to hyperlink according to given keyword. Previously Seed URL Are not crawl again and fro same keyword it returns hyperlinks from database.

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 2, February 2017

Crawled Hyperlinks are stored in database. Multi thread and distributed web crawler is faster in collection data than a direct (no proxy) single thread web crawler. Developing a multi thread web crawler distributed to publicly available proxy servers is easier.

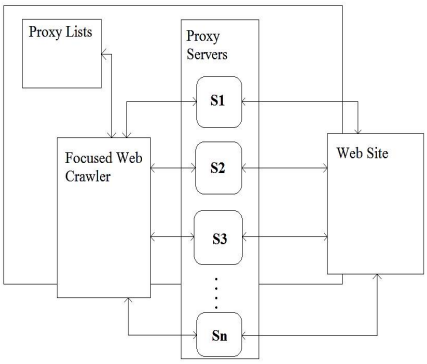

Figure 1. Focused Web Crawler Architecture

Do Le Quoc, et.al. [2] The simple web crawler uses sequence of algorithms and it is nothing but continuous execution of four phases. In the generate phase, the crawler extends the crawl frontier by determining the next set of URLs that it needs to fetch. These URLs are downloaded from web hosts in the fetch phase. The crawler analyses the content retrieved during the parse phase and refreshes appropriately the crawl database in the update phase. However, some of the design issues present in this architecture because of the size and rate of change of data.

This work addresses to those problems and it also tries to minimize the in site communication which decreases communication overhead on the system. It is divided into two sections, first for single site operations and second for multi-site operations.

For single site operations It uses map reduce paradigm and site storage at each mentioned phase of crawling. The map reduce operation works in three steps as map, shuffle and reduce.

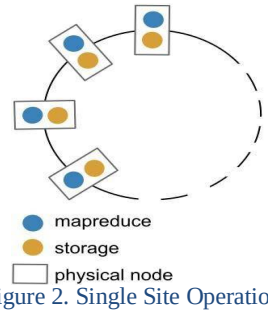

Figure 2. Single Site Operations

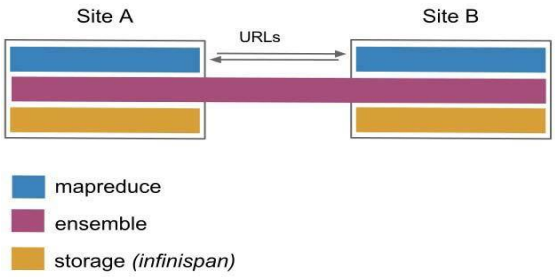

Figure 3. Multi-Site Operations

For site storage operation, in UniCrawl, the crawl database of a site is implemented as a single distributed map structure. This map contains for each page its URL, content, and out links (i.e., the URLs of pages linked from it).

Kartik Kumar Perisetla et.al. [3] The simple web crawlers use most of its data structures to hold frontier set in local address space instead of that space could be used to run more crawler threads for faster operation. The mutual exclusion principle provides access to the URL frontier for each crawler in a synchronized manner to prevent deadlock rather than allowing the threads to fetch URL from a centralized frontier. An efficient web crawler is an essential component of a search engine.

The proposed model has a thread generator program capable of generating numerous crawler threads in a specific time. Every crawler thread has capability to access the same centralised URL frontier. There are two models i) “Non-

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 2, February 2017

mutex" and ii) "Mutex". In "Non-mutex" multiple threads can access the URL frontier without synchronization whereas in "Mutex" multiple threads operate with synchronisation lock to access shared resource. Every thread fetches a page as a result of (Hyper Text Transfer Protocol) HTTP request and HTTP response actions. Every web server, as per robot exclusion protocol features a file named "robot.txt" that specifies which of the pages that are modified since robots last visited.

Mutual exclusion principle states that multiple processes or threads desiring to access identical resource will access it mutually exclusively, that is only one at a time.

With the use of binary semaphore as mutual exclusion lock (mutex), such principle could be achieved. Mutual exclusion for crawler threads applies in an exceedingly similar manner. Once a crawler thread needs to access the shared resource i.e. URL frontier, it checks for the availability of the mutex lock. If it's within the free state, then it locks it and accesses the frontier. By that point, if any other crawler threads need to access frontier it must wait till the lock is free by the thread that holds the lock. Only one thread can access the URL frontier at a time hence providing controlled access and avoiding deadlock. Every thread fetches the URL to be visited from the URL frontier and establishes the connection with the web server. Every thread that is created fetches a URL to be visited from the URL frontier and sends a HTTP request and waits for the response from the web server containing raw text of the page that is requested. All the other threads wait till the time this thread is fetching the URL from URL frontier. Other threads which were waiting for the lock acquired by the first thread releases, other threads acquire it.

The assignment of next thread is totally dependent on how the OS manages the priority for providing the lock to the next waiting thread. The raw data received from the HTTP response is actually added to the 'raw fetched data store'. The thread then repeats its action from fetching the URL. When there is no URL in the URL frontier all the threads are terminated. To extract metadata and links from pages the raw data fetched is processed. Further processing is done by the 'filter' process. The page extract title, outer text of the page, link text, are being read and then added to the metadata store. The links within the page are extracted and added them to the URL frontier.

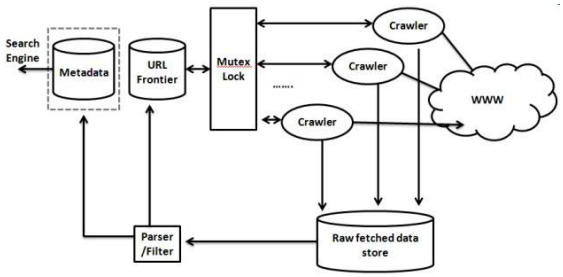

Figure 4. Multithreaded Crawlers using the Mutual exclusion lock

This proposed architecture has following limitations. The system takes a lot of time in acquiring the lock, writing data to database and releasing the lock. The synchronization model across crawler and parser threads are not that efficient in this model, thus makes the performance low. The approach to utilize the waiting time on mutual exclusion lock is not that efficient.

Seyed M. Mirtaheri, et.al. [4] In 1993 first web crawler introduced. World Wide Web Worm is first web crawler. This spider mainly collected information and statistic about the web using a set of seed URLs. Early web crawled iteratively downloaded URLs and updated their repository of URLs through the downloaded web pages.

WebCrawler appeared in 1994.It was the first parallel web crawler by downloading 15 links simultaneously. It also helps to increase the number of indexed pages increased from 110,000 to 2 million. World Wide Web was the first traditional web crawler. Figure 1 shows the architecture of a typical traditional web crawler. Frontier takes a set of seed URLs as input. The seed URLs are send to a module called Fetcher that retrieves the contents of the pages associated with the URLs from the web. Retrieved content is sent to Link Extractor. It parses the HTML pages and extract new

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 2, February 2017

links from them. Newly discovered links are passed to Page Filter and Store Processor. Discovered links are stored in Store Processor. The URLs are then passed to URL-Seen module. This module finds the new URLs that are not retrieved yet and passes them to Fetcher for retrieval. Above Procedure continue until all the reachable links are visited.

The challenges for this crawler are wastage of crawler Resources to crawl irrelevant Hyperlinks, execution time is more. And traditional web crawler does not have mechanism to stop them from bombing a server with many requests. Denial of service (DoS) attack occurs in traditional web crawler. It Slow down normal working of web server.

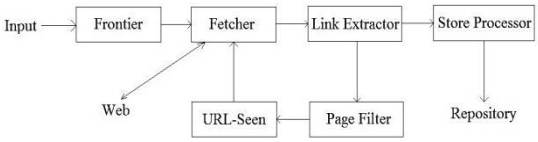

Figure 5. Architecture of traditional web crawler

Duygu Taylan, et.al. [5] A focused crawler is a web crawler that aims only web pages that are relevant to a pre-defined topic or set of topics. In order to determine a web page is about a particular topic, focused crawlers use *page scoring and link scoring* techniques. Page scoring determines whether a page is relevant or irrelevant. Link scoring determines the links to be crawl and prioritize the order of the links to be crawl. The page is downloaded first and then the classifier is used to decide whether the page is about the topic or not.

First, a simple URL optimization method is used. As a result, a few amount of links is crawled, but more accurate. Consequently, the system performance is also increased. The URL optimization method analyses link context, and tokenizes the terms that appears in both anchor text and in URL. The information obtained from link context is used for link scoring. For link scoring, several different methods are used. These methods determine links to be crawled. The link scoring method based on the Naïve Bayes (NB) classifier increases accuracy of the system considerably as compared to other methods. Based on these results, the NB link classifier is modified to incrementally update its training set during crawling to improve the system performance further.

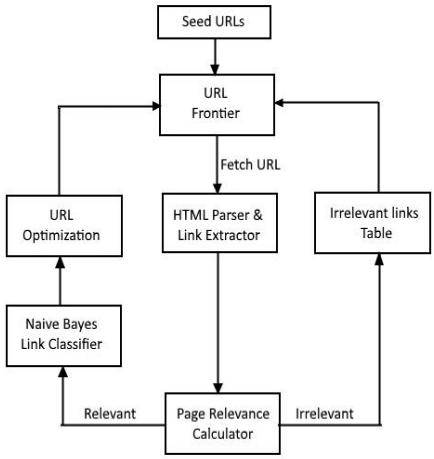

Figure 6. Architecture of Focused Web Crawler

The focused crawler starts with a set of seed URLs and topic words. The topic words are recorded in topic words weight table. First, a query is submitted to a well-known web search engine such as Google in order to find related first n pages. Words that appear in these pages are included in the topic words weight table. The weights are calculated by using the frequency of occurrence of terms in pages. The URLs of these first n pages also form the initial set of seed URLs that is added to URL frontier. The URL frontier component is a queue that stores links waiting to be crawled. The system works until URL frontier queue is empty. The URL frontier retrieves a page if a visited site allows visiting

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 2, February 2017

that is specified in robot.txt file. If a site includes a sitemap.xml file that lists the links for that site, the crawler retrieves this sitemap file in addition to the requested page. Links specified in sitemap file are parsed using an XML parser and sent to the frontier's URL queue. Each retrieved page is parsed with an HTML parser and its content is tokenized into terms. We apply pre- processing methods including stemming and stop word filtering. The page is represented in the vector space model using term frequencies.The page relevance calculator calculates a page score by computing the similarity between the retrieved page and the topic words weight table. The pages with scores above a specific threshold are accepted as similar. After the page relevance calculation, the links included in each page are analyzed. Then, the crawler determines the links to be added to the URL frontier queue for further crawling.

Relevance Calculation

The weight of topic words in different positions calculated as follows:

$$wk = \begin{cases} C_t \ fk & Title - Meta\ Text \\ C_b \ fk & Body\ Text \end{cases}$$

In the equation above, C_t and C_b are meta and body text coefficients respectively. Wk is the weight of the topic word k and fk is the frequency of k in related text. This weight variable is used to calculate the page relevance with cosine similarity measure as follows:

$$Similarity(t,p) = \frac{\Sigma_{i=1}^{n} w_{t_i} * w_{p_i}}{\sqrt{\Sigma_{i=1}^{n}(w_{t_i})^2 + \Sigma_{k=1}^{n}(w_{p_i})^2}}$$

In the above equation, t is the topic words weight table, p is the web page fetched. Wti is weight of words in the topic words weight table, whereas Wpi is the weight words in the page. Relevance score obtained from equation is between 0 and 1.

Link scoring for relevant pages

In a crawling process, the effectiveness of the focused crawler does not just only rely on the maximum amount of relevant pages to be fetched, but it also depends on the speed of the crawling process. The speed of a crawler depend the number of relevant URLs inserted into the Frontier's URL queue. Therefore, a mechanism, called URL optimization, is required for frontier to select links; those are more likely to be relevant.

Naïve Bayes (NB) is one of the most widespread algorithms in text classification. There are two commonly used event models used with NB in text classification: Multi-variate Bernoulli and multinomial models. In the first model, a document is described as a vector of 1's and 0's representing existence of a word. On the other hand, in multinomial model, the vector consists of word frequencies.

Adocumentdescribedbywordsandtheirfrequencies.N_{an}isthefrequencyofword$W_n$indocumentd_a.Thenthemultinomialdistribution would be:

$$P(d_a|c_b;\theta) = P(|d_a|)|d_a|!\,\Pi_{n=1}^{|v|}\frac{P(w_n|c_b;\theta)N^{an}}{N_{an}!}$$

$$\theta_{w_a|c_b} = P(W_a|c_b;\theta_b) = \frac{1 + \Sigma_{n=1}^{|D|} N_{na} P(c_b|d_n)}{|V| + \Sigma_{k=1}^{|V|}\Sigma_{n=1}^{|D|} N_{nk} P(c_b|d_n)}$$

To classify a new document, the following formula is used:

$$P(c_a|d_a;\theta) = \frac{P(c_a|\theta)P(d_b|c_a;\theta_a)}{P(d_b|\theta)}$$

The NB classifier uses terms (words) that appear in both anchor text and in URL as feature vectors. Anchor texts of links and links themselves are tokenized using regular expressions and those tokenized words are used as features in NB. For instance, characters such as "_,.,-,/,+,?,=" are eliminated and the link is spitted into words.

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 2, February 2017

III. CONCLUSION AND FUTURE WORK

With the increase in web data, retrieval of useful information from this huge data is very difficult task. To deal with this, Web crawler extracts useful information. In addition, security is one of the major constraints.

In all the studied papers, different architectures are available. Among these, the focused crawler is best to retrieve topic relevant data from the web. As mentioned in "A Fast Distributed Focused-Web Crawling", the use of proxies improves system performance and anonymity over simple distributed or simple focused crawlers.

REFERENCES

1. Achsan, Harry T. Yani, and Wahyu Catur Wibowo. "A fast distributed focused-web crawling." *Procedia Engineering* 69 (2014): 492-499.
2. Fetzer, Christof, Pascal Felber, Etienne Riviere, Valerio Schiavoni, and Pierre Sutra. "Unicrawl: A practical geographically distributed web crawler." In *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, pp. 389-396. IEEE, 2015.
3. Perisetla, Kartik Kumar. "Mutual Exclusion Principle for Multithreaded Web Crawlers." *Editorial Preface* 3, no. 9 (2012).
4. Mirtaheri, Seyed M., Mustafa Emre Dinçktürk, Salman Hooshmand, Gregor V. Bochmann, Guy-Vincent Jourdan, and Iosif Viorel Onut. "A Brief History of Web Crawlers." *arXiv preprint arXiv:1405.0749* (2014).
5. Taylan, Duygu, Mitat Poyraz, Selim Akyokuş, and Murat Can Ganiz. "Intelligent focused crawler: Learning which links to crawl." In *Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on*, pp. 504-508. IEEE, 2011.
6. Xiao, Zhefeng, et al. "Design and implementation of facebook crawler based on interaction simulation." *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*. IEEE, 2012.
7. Gao, Qing, et al. "A high-precision forum crawler based on vertical crawling." *Network Infrastructure and Digital Content, 2009. IC-NIDC 2009. IEEE International Conference on*. IEEE, 2009.
8. Tadapak, Punnawat, Thanaphon Suebchua, and Arnon Rungsawang. "A machine learning based language specific web site crawler." *Network-Based Information Systems (NBiS), 2010 13th International Conference on*. IEEE, 2010.
9. Chakrabarti, Soumen, Martin Van den Berg, and Byron Dom. "Focused crawling: a new approach to topic-specific Web resource discovery." *Computer networks* 31.11 (1999): 1623-1640.
10. Pirkola, Ari, and Tuomas Talvensaari. "Effects of Start URLs in Focused Web Crawling."
11. Yang, Sheng-Yuan, and Chun-Liang Hsu. "An ontology-supported web focused-crawler for Java programs." *Ubi-media Computing (U-Media), 2010 3rd IEEE International Conference on*. IEEE, 2010.
12. Micarelli, Alessandro, and Fabio Gasparetti. "Adaptive focused crawling." *The adaptive web*. Springer Berlin Heidelberg, 2007. 231-262.