Int Journal of Social Sciences Management and Entrepreneurship 7(2): 028-037, 2023

ISSN 2411-7323

www.sagepublishers.com

AN ONTOLOGY BASED WEB CRAWLER WITH A NEAR-DUPLICATE DETECTION SYSTEM TO IMPROVE THE PERFORMANCE OF A WEB CRAWLER

[*] NGULAMU Daines Walowe

[*] Master of Science in Computer Systems at Jomo Kenyatta University of Agriculture and Technology

ABSTRACT

A web crawler is a program that searches the World Wide Web in an orderly manner in order to collect data based on a search query. Web crawling is therefore the process of finding web pages and downloading them automatically. Crawlers have a difficult time getting relevant and quality information according to the search query of the user from the web. This is due to the large volume of the World Wide Web. This characteristic of the web also challenges the web crawlers as they may download duplicate and near-duplicate web pages according to the search query. These web pages reduce the quality of search indexes as well as affect storage cost and page ranking. In order improve the performance of the web crawler, an ontology-based web crawler with a near duplicate detection system was designed. The experiment was carried out using secondary data from a sample web site which was used since crawling is an endless process. Using these two approaches, the ontology web crawler would search for relevant searches according to the search query of the user while the near-duplicate detection system would eliminate redundant data.

Key Words: Crawlers, Ontology-Based Web Crawler, Near-Duplicate Detection System

Background

A web crawler is a program that creates entries for a search engine by visiting web pages and extracting relevant information (Lawankar and Mangrulkar, 2016). The internet consists of various communication systems and networks, making it a vast and challenging environment to manage information effectively (Saini, 2016). The objective of a crawler is to gather as many useful pages as quickly and efficiently as possible (Pranav and Chauhan, 2015). However, the growth of the World Wide Web, with its billions of web pages constantly changing, presents difficulties for web crawlers (Suganiya and Haripriya, 2015). The sheer volume and dynamic nature of web content require robust crawling strategies.

Architecture of a web crawler

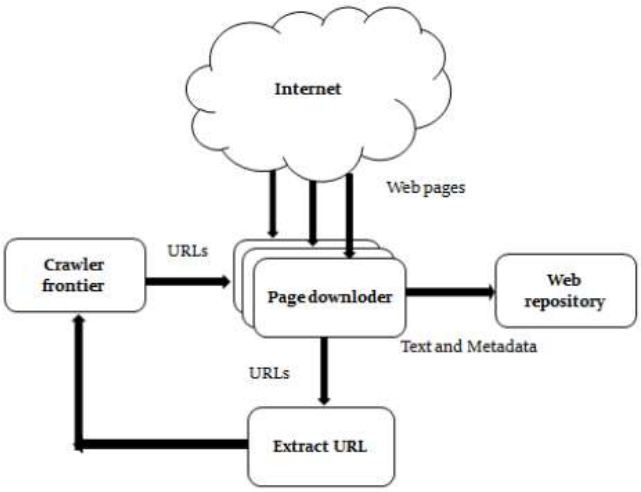

Figure 1: Architecture of a web crawler (Amudha,2017)

The architecture of a web crawler comprises several components (Amudha, 2017). The crawler frontier serves as a repository for URLs that are yet to be crawled. URLs can be provided by users or generated programmatically. The crawler starts by selecting a URL from the seed set stored in the frontier and continues fetching and extracting URLs until the frontier is empty or another condition is met. The page downloader retrieves web pages from the internet based on the URLs obtained from the crawler frontier. To accomplish this, an HTTP client sends requests and reads responses. The web repository is responsible for storing and managing the downloaded web pages, typically focusing on HTML pages and ignoring other media types. It stores both the original and updated versions of crawled web pages.

How a Web Crawler Works

A web crawler operates by following hyperlinks on web pages and collecting their content (Anuja and Nikhil, 2016). The crawler traverses the website's structure, treating the starting URL as the root and considering all links in that root page as its children. This process continues recursively to explore the website and avoid redundancy. The crawler checks each link for duplication, ensuring efficiency by avoiding unnecessary retrieval. However, web crawlers face challenges such as retrieving irrelevant information and encountering duplicate content, impacting the quality of search results (Agre and Mahanjan, 2015). Duplicate content consumes storage space, increases costs, and frustrates users (Nirmalrani et al., 2015).

Web Crawler Features

Web crawlers require certain features for effective operation. Robustness refers to the ability to handle a large number of servers and restore content in case of interruptions (Wen, 2018). Politeness entails respecting the policies and limitations set by web servers to avoid overloading them and maintaining their value to users (Saini, 2016). Scalability is crucial, allowing crawlers to add machines or increase bandwidth as needed (Shirvastava, 2018). Efficiency is essential in utilizing processor, bandwidth, and memory resources effectively (Shirvastava, 2018).

Problem Statement

Web crawlers have faced several limitations regarding retrieval of data from the World Wide Web. One of the challenges of web crawlers is being unable to get relevant and quality information according to the search query of the user from the web. This is due to the large volume of the World Wide Web. Information is provided by millions of web content providers and web pages are constantly being added and content of web pages keeps changing (Suryawanshi and Patil, 2015).

Due to the large volume of the internet, users must either browse through a hierarchy of concepts to find the information they need or submit a query to a search engine to wade through thousands of results most of which are irrelevant (Suganiya and Haripriya, 2015).

There is an increase of duplicate web pages on the internet due to lack of a standard mechanism to guarantee the non-existence of a webpage before hosting them (Anun and Sumesh, 2015). Duplicate web pages and near-duplicate web pages affect the quality of the crawled content. They waste the user's time, impact on the crawler's storage affect page ranking and create additional overhead on search engines (Subramanyam et al. 2016).

Objective of the Project

The main objective of this thesis was to design an ontology based web crawler with a near-duplicate detection system to improve the performance of a web crawler. This was broken down into specific objectives:

Specific objectives:

i. To review ontology web crawlers with different approaches
ii. To review near duplicate detection with different approaches
iii. To design an ontology based web crawler with a near-duplicate detection system
iv. To test and evaluate the ontology based web crawler with a near-duplicate detection system

Literature review

Evolution of web crawlers

Web crawlers have undergone significant evolution over time, expanding their capabilities beyond data collection to include monitoring vulnerabilities and web application accessibility. The history of web crawlers dates back to when the web consisted of only around 100,000 web pages (Singla, 2015). The first web crawlers to emerge were the RBSE spider developed by NASA in 1994, followed by Brian Pinkerton's WebCrawler in the same year. Another notable crawler was Archive.org, also known as the Wayback Machine (Williams, 2015).

In the past decade, web crawlers have experienced further advancements, transitioning from simple spiders to more sophisticated bots capable of multitasking and serving multiple purposes. Distributed crawlers emerged in 2003, enabling the crawling of a vast number of web pages in a matter of seconds. The development of the Heritrix crawler in 2004 aimed to support focused and broad crawls efficiently. In 2007, the Ajax crawler was introduced, specifically designed to crawl through rich internet applications, utilizing a breadth-first algorithm for web page indexing. With

the advancement in internet technology and the widespread use of mobile phones, mobile-specific crawlers like Googlebot-mobile and Bingbot mobile were developed in 2014 (Singla, 2015).

Focused Crawler

According to (Sharma and Gupta 2015), the focused crawler is designed to search for web documents that are related to a specific topic or a set of topics. The topics are specified using standard documents. The focused web crawler searches for links that are likely to be related to the search of interest and the parts of the web that are irrelevant are avoided.

Architecture of a focused crawler

Bhatt et. al, 2015 stated that, the architecture of a focused crawler is composed of three components: namely seed detector, crawler manager, crawler, HTTP protocol module, link extractor and hypertext analyzer. The seed extractor retrieves the first URLs for the specific keyword. The crawler manager downloads documents from the internet. The crawler stores web pages downloaded from the web in a document repository. The HTTP protocol module requests for the documents received form the queue. The link extractor removes the link from the documents in the web repository. The hypertext analyzer finds the relevancy of the terms with the search keyword referring to the taxonomy hierarchy.

According to (Sain, 2016), the focused crawler can be further categorized into keyword based approach and exemplary document based approach. Keyword based approach can be further categorized into ontology based approach and link semantic based approach. Exemplary document based approached can be further categorized into data mining based approach. Keyword based approach extracts URLs based on the search criteria or keywords. They only retrieve web pages that contain the keywords and ignore other web pages as irrelevant searches. Ontology based approach uses the keywords and terms in the ontology to retrieve relevant documents by cross referencing the web documents with the keywords in the ontology. Link semantic based approach retrieves web documents through speculation that the web document is relevant based on the surrounding keywords and text on the links. Data mining approach is the process of examining data from different angles and summarizing it into useful information

Advantages of focused crawlers

It reduces network traffic by downloading only relevant pages. Before it downloads a page, it first checks weather the link is relevant to the topic or not and downloads only the web pages based on the query (Shrivastava, 2018). It can identify regions of the web that are dynamic as compared to those regions that are static (Bhatt et. al, 2016).

Challenges of focused web crawlers (Bhatt et. al, 2016)

Focused crawlers, in a bid to crawl pages that give instant benefit may miss relevant pages. Maintaining database freshness is an issue. Information contained in the HTML pages are constantly being updated either daily, weekly or monthly. These pages have to be downloaded by the crawler and updated in the database. This process becomes slow and affects the internet traffic. Focused crawlers depend on a particular context so that relevant pages are downloaded. In the absence of this particular context the crawler may download irrelevant web documents.

Web crawling techniques

Ontology based web crawler for specific domain

(Singhal and Kumar, 2016), stated that browsing the World Wide Web and the large amount of data it contains takes too much time. The search also contains irrelevant searches. They therefore proposed an ontology based web crawler for domain specific that would select and seek out web pages that fulfilled the user's requirements. They used the link analysis algorithm that would

prioritise URLs based on page ranking. This would ensure downloading of the most specific web pages.

Their proposed method would improve and retrieve more relevant pages. It consisted of calculating a total number of keywords which are stored in the database relating to the query which during crawling can calculate the relevance factor by retrieving the web page and multiplying the weight factor of the keyword.

Relevance factor =∑total count of keyword *weight factor

The proposed approach would input a web page "P" and the output would be the relevance score of the web page "P". The relevance score of the web page is initialized to "0.RELV_P=0". The first term "T" is selected together with the corresponding weight from the weight table. "T" is calculates according to the number of times it occurs in the web. The number of occurrence is multiplied with the weight "W". The weight is added to RELV_P. This is done until all terms in the weight table are visited.

They concluded that their approach would retrieve more relevant pages, reduce crawling time and save on space. Their approach included crawling through the web and adding web pages to the database relating to a specific domain and discard web pages not related to that domain. They stated that they developed an ontology that uses relevancy decision mechanism to fetches web pages.

Detection of near duplicate web pages using four stage algorithm

Nirmalarani et al (2015) stated that duplicate and near duplicate webpages decrease the effectiveness and efficiency of search results and also provide irrelevant searches to the user. This is due to the massive development of the web and the existence of billions of web pages. Near duplicate web pages are due to mirrored sites, replicas of the original site, plagiarized documents and representations of the same object. They therefore proposed a four stage algorithm that would detect near duplicate web pages. The algorithm is based on a term document weight (TDW) matrix that includes four stages namely pre-processing, minimum weighting, filtering and verification.

In their proposed method similarity verification is carried out from a large collection of web pages. It is based on the jacquard threshold value t, $0<t<1$, if t.t=0, it returns a full set of records. The set of records contains at least a single word or a single context similar to the user input record r. if t.t=1, it will return a similar record from the repository that is similar to the input record r.

In the first phase of the algorithm, web pages undergo preprocessing which includes, removal of stop words, stemming and HTML tag removal. In the second phase, features are weighed according to the number of appearance in different sections of the web document. The number of occurrences is multiplied by the weight. A weighting threshold is applied to reduce the fields in the TDW matrix. This is done by applying analytic combination criteria. The weights are then normalized with respect to the length of the section. A term document weight matrix is formed based on global ordering and standard normalization. In the third phase, the size of the TDW matrix is reduced by applying a filtering technique and positional technique. By doing this the number of comparisons is reduced. "T" is assumed as the threshold value as a jacquard similarity threshold. Filtering ensures that is any two records share any common words it will lead to similarity. In the fourth phase, verification is done by doing similarity checking to get an optimal number of duplicate records.

In their conclusion, they stated that their algorithm worked well in identifying near-duplicate web pages. This algorithm gives high levels of accuracy and helps to reduce the number of comparisons

Near-Duplicate Web Page Detection by Enhanced TDW and simHash Technique

Arun and Sumesh (2015), stated that the presence of near- duplicates on the web increased the cost on storage, reduced the performance of search engines, utilization of network bandwidth and increased load on the remote server. They also stated that duplicate content could exist either by accident or intentionally. They proposed a method that would detect duplicate and near-duplicate content. The proposed method included using an extended term document weighting (TDW) scheme, sentence level features and simHash technique. Their proposed approach included four phases namely preprocessing, word mapping, feature extraction and feature comparison.

Preprocessing includes removal of HTML tags and scripts from web pages. Weights were assigned on terms based on where the terms belonged on the page (Wt). Frequency (ti) was also assigned to terms based on how often they appeared on the web page.

Word mapping includes identifying words with synonyms and then extracting the row of its synonym and returning the minimum length word for stemming. A TDW list is computed based on the frequency of terms and tag weighting. Normalization is done based on total weights of feature terms that is

$(\sum fti*wi)$

Input web pages are compared with existing web pages by using a hybrid mechanism. Filtering uses the number of sentences on the web page. A sentence threshold Si is used to determine whether the web page passes the second stage of filtering. The difference in number of sentences should be less than or equal to the sentence threshold. Bit by bit comparison of simHash value is done to the input web pages and the subset of web pages from the sentence filtering. If the user specified bit threshold is satisfied then a web page is marked as a near-duplicate.

Arun and Sumesh (2015) concluded that their proposed algorithm was a four phase method for detecting near-duplicates and had an acceptable execution time as compared to existing methods.

Ontology based web crawling

Kumar and Kumar (2015) stated that, retrieval of equivalent web pages from the web is still a significant test to the retrieval system. They also stated that despite the use of keywords to retrieve information from the web, irrelevant web pages were also retrieved from the search. They therefore proposed a method that would trade on the event without trading off on ontology based web crawling which does not need feedback or training. This method would guide the crawling to the necessary data and it would also discover important pages first before crawling.

The proposed method according to Kumar and Kumar (2015) would first check the web pages for legitimacy that is which sort of markup language. The web pages on being of the characterized markup language, it is added to the line. The web document is the parsed. A confirmation is gotten from the server that the web document is okay. The contents of the web page are then compared with the terms in the ontology. The results are checked, the web page added to a list and reserved in an envelope.

Web page and SIMILAR-PAGE=0. A cutoff is established by the researcher to determine the relevance of the web document. Results are acquired by crawling the same website against the established breaking points. The first term (T) is taken from the ontology and assigned weight (W) according to the weight table. Compute how often the term (T) appears in a web page that is FREQUENCY. A result is produced by FREQUENCY * W. The result is added to SIMILAR-PAGE to get a new term SIMILAR_P that is SIMILAR_PAGE + RESULT = SIMILAR_P. The process is repeated until all the terms in the weight table are checked.

In the event that SIMILAR_P <RANGE (established cutoff) then the web page is discarded else the web page is downloaded.

In their conclusion Kumar and Kumar (2015) stated that their proposed algorithm was everything a capable crawler would require as it discovers relevant web pages and discarded the irrelevant web pages

Strengths and Weaknesses of Various Web Crawling Techniques

Technique	Strengths	Weaknesses
Ontology based web crawler	• Retrieves more relevant web pages • Reduces crawling time • Saves space	• Ontology is static
Focused web crawler	• It gives a better performance than general web crawler • It leads to savings in hardware and network resources. • It would help keep the crawl up to date	• Missing relevant pages in a bid to give instant benefits • Issues with updating the database as the process in slow and affects the internet traffic • Downloading irrelevant web pages in the absence of a particular context.
Detection of near duplicate web pages using four stage algorithm	• It worked well in identifying near-duplicate web pages. • It gives high levels of accuracy • Helps to reduce the number of comparisons	• It doesn't retrieve relevant web pages.
A improved crawler based on efficient ranking algorithm.	• Reduced the load on the network • Ranked the web pages efficiently.	• Retrieved redundant data
An efficient approach for near-duplicate page detection in web crawling	• Reduced memory spaces for web repositories • Improved the quality of search engines.	• It doesn't retrieve relevant web pages.
Handling duplicate data detection of query result from multiple web databases using unsupervised duplicate detection with blocking algorithm	• It worked well in identifying near-duplicate web pages.	• A pre-trained approach is not suitable for web based retrieval as it is greatly query based.
Near-duplicate web page detection by enhanced TDW and simHash technique (2015)	• It removed redundant web pages from the user's search	• It doesn't retrieve relevant web pages.
Ontology based web crawling (2015)	• It retrieved relevant web pages • Retrieved fewer web pages	• Ontology is static

Conceptual web crawler

The proposed web crawler was a combination of the ontology based web crawler and a near-duplicate detection system. The ontology based web crawler would ensure that relevant web pages

are retrieved according to the search query of the user and a near-duplicate detection system would ensure retrieval of non-redundant information from the web reducing overhead on the search engine and saving the user's time in sorting of documents. Out of the relevant pages retrieved by the ontology crawler, the near-duplicate detection system would ensure that there are no duplicates or near-duplicate in the crawled web pages.

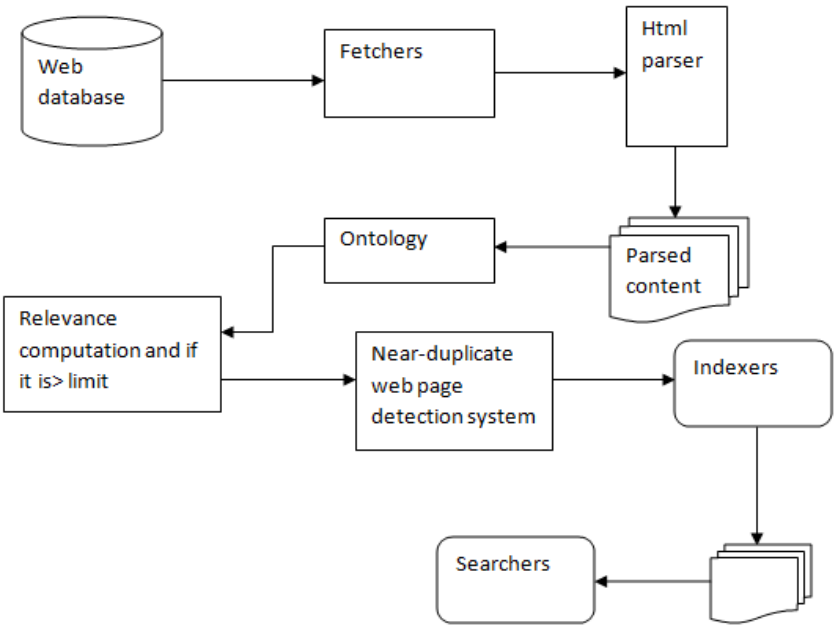

Figure 2.0: Conceptual web crawler (Kumar and Kumar, 2015)

Research methodology

A descriptive survey design was employed for this study to investigate possible relationships between variables. Descriptive research was used to gather, tabulate, organize, depict, and describe the data. Visual aids such as graphs and charts were utilized to enhance data understanding. The proposed approach was tested using various tests.

Several parameters were considered in the research. The relevance score was computed based on Kumar and Kumar (2015), using an algorithm that determined the relevance of a web page. The cutoff point set by the researcher was "3" to check the relevance of the web document. Weight was assigned to keywords and phrases in the ontology, with more specific terms receiving higher weights. Frequency and weight were used to determine the relevance of a document.

In terms of near-duplicate detection, an approach from Arun and Sumesh (2015) was employed. A threshold of "80" was set for comparing web documents, and a hybrid mechanism was used to filter and calculate differences in documents. If the calculated difference was below the threshold, the document was considered a duplicate and discarded.

Pseudo code for relevance and duplicate detection was developed to outline the steps for determining the relevance of a web page and detecting duplicates.

The experiment was conducted on a Windows machine with specific hardware specifications. Xampp software was installed and used for developing the crawler. A sample website consisting of 102 pages was used for simulation purposes, with the URL http://localhost/web-test/

Results, analysis and discussions

To begin the experimentation, the researcher utilized a sample website with 102 web pages and entered its URL into the crawler's interface for indexing. Initially, the relevance and duplicate detection aspects were not taken into consideration. As a result, all 102 web pages were indexed.

Next, the crawler incorporated the use of ontology to determine the relevance of the web pages. The ontology, centered around the domain of "web design," guided the crawler in indexing pages that matched the ontology's content. In this phase, 76 web pages were deemed relevant, while 26 were identified as irrelevant. For web that does not align with the ontology, the issue of duplicate or near-duplicate web pages, the crawler performed a similarity comparison.

Performance measures were also assessed during the experiments. The results, as presented in Table , demonstrated the impact of incorporating ontology and near-duplicate detection on the crawler's performance. With the inclusion of ontology, the number of web pages retrieved decreased to 76, while the time elapsed for crawling reduced to 0.031 seconds. When near-duplicate detection was added, the number of web pages further reduced to 42, with a time elapsed of 0.009 seconds. The crawler with both ontology and near-duplicate detection retrieved 38 web pages in only 0.005 seconds.

Table 1: A table showing experimental results

	No. of web pages retrieved	**Time elapsed**
Normal crawler	102 web pages	0.079 seconds
Crawler with ontology	76 web pages	0.031 seconds
Crawler with near-duplicate detection	42 web pages	0.009 seconds
Crawler with ontology and near-duplicate detection	38 web pages	0.005 seconds

The analysis of the experimental results revealed that the integration of ontology and near-duplicate detection improved the relevance and efficiency of the web crawler. The utilization of ontology facilitated the filtering of irrelevant web pages, ensuring a more targeted and accurate indexing process. Moreover, the near-duplicate detection system successfully identified and discarded duplicate or near-duplicate web pages, enhancing the overall quality of the indexed content.

Summary

An ontology web crawler with a near-duplicate detection system was proposed in this research, combining features from existing frameworks. The ontology-based web crawler utilized a domain-specific ontology, while the near-duplicate detection system employed TDW and simHash techniques.

The experimental data consisted of 102 web pages from a test website, categorized as relevant, irrelevant, and duplicate pages, with a focus on the "web design" domain. Performance evaluation was based on the time elapsed and the number of web pages retrieved.

Compared to other approaches, the proposed framework demonstrated advantages in retrieving relevant web pages while removing duplicate and near-duplicate content. It addressed the limitations of approaches that only retrieved relevant pages or removed redundant data separately. By integrating ontology and near-duplicate detection, the proposed approach achieved relevance and redundancy filtering simultaneously.

The results showed that the proposed approach outperformed a normal crawler. When using ontology, the number of retrieved web pages decreased to 76, and with near-duplicate detection, it further reduced to 42. The combined use of ontology and near-duplicate detection resulted in

retrieving 38 web pages. The elapsed time for crawling also decreased progressively, highlighting the efficiency of the proposed approach.

Conclusion

Due to the every growing World Wide Web, web pages are constantly updated, deleted or changed. This poses a challenge on web crawlers as they are only able to download a fraction of the web pages on the World Wide Web. This means that even when web crawlers download pages they may be irrelevant and do not match the search query. The crawled results also include mirrored or near-duplicate web pages. The proposed web crawler will crawl relevant web pages that match the search query at the same time get rid of redundant data. This approach will reduce computational time, manage storage space, improve page ranking and reduce user's time in searching for relevant information.

Further Research

Despite the fact that the proposed model is efficient, it would need improvement in some areas. The ontology remains static; it can be improved to be dynamic by adding new relations that is the crawler should search for web pages related to the search even if they don't contain the keywords searched. Domains and concepts should be added when visiting new web pages. Standardization of weights needs to be done because as of now experts assign weights to terms according to the area of expertise and knowledge.

References

Anisha Gupta. (2013). A review on efficient web crawling.

Arun Pr, Sumesh Ms. (2015). Near-duplicate web page detection by enhanced TDW and simHash technique. *International Conference on Computing and Network Communications.*

Ayar Pranav, Sandip Chauhan. (2015). Efficient Focused Web Crawling Approach for Search Engine. , *International Journal of Computer Science and Mobile Computing, Vol.4 Issue.5, May- 2015, pg. 545-551.*

Bo Wen. (2018). Application of distributed web crawler in information management systems. *School Of Computer Science And Technology.*

Brian Pinkerton. (2000) Web crawler: finding what people want.

Deepak Kumar, Aditya Kumar. (2013).Design Issues for Search Engines and Web Crawlers: A Review . *IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, p-ISSN: 2278-8727Volume 15, Issue 6 (Nov. - Dec. 2013), PP 34-37*

Dr. Naresh Kumar, Shirank Awasthi, Devvrat Tyagi. (2016). Web crawler challenges and their solutions. *International Journal Of Scientific And Engineering Research*.

Dvijesh Bhatt, Daiwat Amit Ayas, Sharnil Pandya. (2015). Focused web crawler. *Advances In Computer Science And Information Technology.*

Eldhose P sim. (2015). Classification of detection of near duplicate web pages using five stage algorithm.

Gujan .H. Agre, Nikita .V. Mahajan. (2015). Keywords focused crawler. *International Conference On Electronics And Communication Systems.*

J. Prasanna Kumar & P. Govindarajulu .(2013). Near-Duplicate Web Page Detection: An Efficient Approach Using Clustering, Sentence Feature and Fingerprinting, International Journal of Computational Intelligence Systems, 6:1, 1-13, DOI: 10.1080/18756891.2013.752657

Janet williams. (2015). Evolution of web crawling: how crawling the web emerged as a maintenance discipline.

Jyoti Mar, Naresh Kumar, Dinesh Rai. (2019). An improved crawler based on efficient ranking page algorithm. *Internation Journal Of Advanced Trends In Computer Science And Engineering.*

K. Subramanyam Sharma, Dr. K. SrujanRaju, P.Yadagiri. (2016). An Efficient Approach for Near-duplicate page detection in web crawling

K. Subramanyamm Sharma, Dr.K Srujan Raju. (2016). An effiecient approach for near-duplicate page detetction in web crawling. *Imperial Journal of Interdisciplinary Research.*

Md Abu Kausar, V.S.Dhaka, Sanjeer Kumar Singh. (2013). A web crawler: A review. *International Research Journal Of Computer Applications, Vol 63 No 2*

Mridul B sahu, Samiksha Bharne. (2016). A survey on various kinds of web crawlers and intelligent crawler. *International Journal Of Scientific Engineering And Applied Science.*

Ms Girija. M. (2016). Handling duplicate data detection of query result from multiple web databases using unsupervised duplicate detection with blocking algorithm. *International Research Journal Of Engineering And Technology.*

Parigha Suryawanshi , D.V.Patil. (2015). An Overview of Approaches Used In Focused Crawlers. *International Research Journal of Engineering and Technology (IRJET) Volume: 02 Issue: 09 | Dec-2015*

S. Amudha. (2017). Web crawler for mining web data. *International Research Journal Of Engineering And Technology.*

Satinder Bal Gupta . (2012). The Issues and Challenges with the Web Crawlers . *International Journal of Information Technology & Systems, Vol. 1, No. 1*

Shailesh Singh, Syed Imtiyaz Hassan. (2017). Detecting duplicates and near duplicate records in large databases. *International Journal On Computer Science and Engineering*.

Vardana Shrivastava. (2018). A methodolical study of a web crawler. *Journal Of Engineering Research And Application.*

Vishaka. (2018). Issues and challenges wth web crawlers. *International Journal Of Science And Research.*