



Design And Develop A Web Crawler For Web

Submitted by:

Mahzabin Ferdous Brinto

181-35-2399

Department of Software Engineering

Daffodil International University

Submitted to:

Nuruzzaman Faruqui

Lecturer (Senior Scale)

Department of Software Engineering

Daffodil International University

This Project report has been submitted in fulfillment of the requirements for the Degree of Bachelor of Science in
Software Engineering

APPROVAL

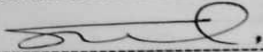
This thesis titled on "**Design and Develop a Web Crawler For Web**", submitted by **Mahzabin Ferdous Brinto** ID: **181-35-2399** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS



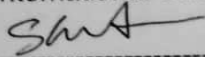
----- Chairman

Dr. Imran Mahmud
Head and Associate Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University



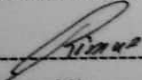
----- Internal Examiner 1

Md. Khaled sohel
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University



----- Internal Examiner 2

Md. Shohel Arman
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University



----- External Examiner

Rimaz Khan
Managing Director
Tecognize Solution Limited

DECLARATION

I, hereby, declare that this thesis "**Design And Develop A Web Crawler For Web**" has been done by me under the supervision of **Mr. Nuruzzaman Faruqui**, Lecturer (Senior Scale), Faculty of Science and Information Technology, Department of Software Engineering (SWE), Daffodil International University (DIU). It is additionally declared that neither this thesis nor any component has been submitted elsewhere for the award of any degree.

Mahzabin

Mahzabin Ferdous

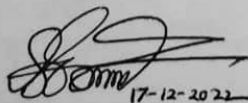
ID:181-35-2399

Department of Software Engineering

Faculty of Science & Information Technology

Daffodil International University

Supervised By :


17-12-20 22

Mr. Nuruzzaman Faruqui

Lecturer (Senior Scale)

Department of Software Engineerin

Faculty of Science & Information Technology

Daffodil International University

ACKNOWLEDGEMENT

This thesis is the consequence of an inspiring and exciting journey, where many passionate individuals have provided their heartiest support in many ways.

First, I want to express my gratitude towards the Almighty ALLAH for his divine blessings to let me complete my final year thesis successfully. I would like to express my humble respect to our honorable **Nuruzzaman Faruqi** sir, for his constant support, guidance and advice. I am fortunate to have him as my supervisor.

Besides, I would like to appreciate **Dr. Imran Mahmud, Associate Professor & Head (In Charge)** of the Department of Software Engineering, Daffodil International University for motivating us for quality research. I also want to thank my teacher who has been continuously supporting us throughout this undergraduation.

My heartfelt appreciation goes to **Mr. Maruf Hassan, Associate Professor & Director of Cyber Security Center** for sharing his knowledge & wisdom about cyberthreats & cybersecurity. He not only played the role of our teacher, but also played the role of a guardian.

I want to acknowledge our love to our parents and family members for their constant support & care they've always provided us.

Finally, I want to appreciate the entire department of Software Engineering of Daffodil International University.

TABLE OF CONTENTS

	Page No
APPROVAL	i
DECLARATION	ii
ACKNOWLEDGEMENT	iii
TABLE OF CONTENTS	iv-v
Abstract	vi
CHAPTER 1: INTRODUCTION	
1.1 Background	1
1.2 Motivation of the Research	1-2
1.3 Problem Statement	2
1.4 Research Scope	3
1.5 Research Questions	3
1.6 Research Objectives	3
1.7 Thesis Organization	3
CHAPTER 2: LITERATURE REVIEW	4-5
CHAPTER 3: METHODOLOGY.....	6
3.1 Seed URL.....	7
3.1.1 Transfer Current Seeds To a New Seed URL Collection.....	8
3.1.2 Deleting Or Deactivating Any URL From Seed URLs.....	8
3.2 URL Frontier:	8
3.2.1 URL Collection & Scanning Phase.....	9
3.3 Duplicate Detection.....	10

3.4 URL Extraction.....	11-12
3.5 Store pages.....	11
CHAPTER 4: IMPLEMENTATION AND EVALUATION.....	13
CHAPTER 5:Result & Discussion.....	14-15
CHAPTER 6:CONCLUSIONS	16
REFERENCES.....	17-19

Lists Of Figures

Figure 1 Block diagram of the methodology	7
Figure 2 Flow Chart of URL Frontier	9
Figure 3 Flow Chart of URL collection phase.....	10
Figure 4 Flow Chart of Duplicate Detection.....	11

Abstract

A web crawler is a computer program or piece of software that systematically and consistently browses the Internet and downloads web content. A web crawler automatically detects and organizes resources from the internet in an organized manner in accordance with user needs. The web crawler is a equipment that is essential to search engine optimization. The crawler of a search engine may browse webpages and gather important links from the internet, it will evaluate each web page's significance based on metrics such as the number of pages that link to it, Search engines would maintain records of the webpages their web crawlers had visited and crawled after indexing. Your website's pages won't appear in search results if they aren't indexed. In a short period of time, the crawler contacts millions of websites, consuming a sizable amount of network, storage, and memory in the process. Therefore, web crawlers are becoming more and more prominent over time. While scalability and robustness were previously the focus of research, Intersecting sub-problems, lesser scalability, increasing runtime and delayed network loading, low load balancing rate, lower rate of failure tolerance, etc. are some research gaps in this area. In this paper, an effort to cope with failure is the major focus is on deployment and tolerance between internal and external links. The crawling programmatic approach and its different phases are only explained briefly in a small number of papers. In this paper, we addressed the implementation of web crawling's underlying knowledge, to make it more easier for the audience to understand.

Keywords : Web crawler, Search Engines, Web pages, web spider.

CHAPTER 1

Introduction

Crawling the web is a strategic component for collecting data and keeping up with the constantly evolving Internet. A program known as a crawler automatically navigates the internet by downloading documents and browsing on links from one page to the next. It is a tool used by search engines . All search engines utilize internal web crawlers to maintain existing copies of the material. The search engine is divided into many components. The search engine depends most heavily on the crawler module out of all of these modules since it helps the search engine provide the best results. Getting into the specifics of programmatic implementation helps to better understand how search engines operate as each of them depend on web crawlers for indexing, processing and repository storing.

1.1 Background

Web crawling can also be referred to as graph traversal. Links and nodes make up the web. In essence, it is a sizable graph in which pages function as nodes and hyperlinks as edges. These nodes and edges are important because they help the crawling process get started. A crawler is a crucial part of a search engine that downloads webpages from the internet by clicking on links inside the pages (Kausar et al.,2013). By using the links, it moves on to the other seed URLs after starting with a few of them. Expanding a node in graph search is comparable to retrieving a page and extracting the links from it. The crawler contacts millions of websites in a short amount of time, using up enormous amounts of network, storage and memory in the process. These loads push the hardware to its limit, so this task needs to be carefully coordinated and divided among processes.

1.2 Motivation of Research

A search engine is created to retrieve data from the internet. It acts as a conduit between the user and the web. Crawler, Indexer, and Page Ranking are the three important components of a search engine. Web crawlers play a vital role in search engine

optimization . A search engine's crawler is capable of exploring webpages and extracting essential links from the internet (Deshmukh et al.,2021) .This highlights how heavily any search engine is depended on the crawlers. Before commencing to develop a smart crawler, it is therefore necessary to do a deep evaluation of all the crawlers previously in use in order to fully understand their limitations and gain insights into its working approach. (Udapure et al.,2014).Therefore, we became motivated to do this research on web crawler.

1.3 Problem Statement

In this paper, authors talks about the basic principles and the characteristics of web crawlers, talks about the new strategy (Yu et al., 2020) Though, only works for relevance topic doesn't work for detailed topics. In this paper, authors propose various optimization techniques that can minimize the coordination effort between crawling processes, so that they can operate more independently while maximizing their effectiveness. (Cho et al., 2002) but this paper can only reduce communication overhead by roughly 40%. Replicating more URLs does not significantly reduce the overhead. In this paper, The issue of overlap of downloads by client crawlers are addressed, it reduces the dependency of the system on a single client. The architecture is easily scalable. (Yadhav et al., 2008).But Potential work related to the parallel crawler that may be added on, this is the adaptation of this model for building a full-fledged search engine. This paper analyses the concepts of web crawler, the different types of crawlers and its working. (Udapure et al., 2014) . This paper implement the proposed framework. It focuses on evaluating the crawlers based on scalability and robustness on e-commerce websites . But it has more testing time and experimental groups should be considered, which can improve the accuracy of the experiment. (Yang & Thiengburanathum ,2021, March). While scalability and robustness were previously the focus of research. Intersecting sub-problems, lesser scalability, increasing runtime and delayed network loading, low load balancing rate, lower rate of failure tolerance etc. are some research gaps in this area. In this thesis paper , attemption to deal with failure tolerance and deployment between internal and external links are mainly focused.

1.4 Research Scope

In this paper, highlighted Scope are given below:

- i) Identify the factors that are causing the problem in existing models.
- ii) Give a model by solving existing problems.
- iii) Improve the accuracy from existing paper.

1.5 Research Question

- i) How can a web crawler perform efficiently by resolving its existing challenges?

1.6 Research Objective

In this thesis, we are trying to work with failure tolerance and deployment between internal and external links.

1.7 Thesis Organization

Giving a summary of this thesis was the major goal of this particular chapter. The rest of this paper is organized as follows. The study is composed of five chapters in total. In the first chapter, study background, motivation, objectives, problem statement, research questions and research scope are discussed. A brief summary of some prior research that is connected to this subject is included in Chapter 2. Research methodology is briefly described in Chapter 3. Chapter 4 is all about the implementation of the paper. Chapter 5 compares the outcomes with those of other trustworthy crawlers to summarize the findings. Finally, In the last chapter conclusion and recommendations are given.

CHAPTER 2

Literature review

Although there is a finite quantity of information on the internet, it is difficult to manage since there are too many pages that contain that information. Fundamentally, this results from the fact that the web is a sizable, dynamic, and ever-growing media. According to Olston and Najork, this presents many significant problems that each crawler must overcome: A crawler must scale because there is a lot of information to parse; it must choose which information to download first and which information to refresh over time; it must not burden the websites that contain the content; and it must consider opponents, such as spider traps (Fetzer et al, 2015). In this paper, authors have studied various crawlers in detail to understand its working, merits and demerits is the objective of the research paper as this will help in formulating a concept for developing a search engine in future(Deshmukh et al.,2021). Although, this paper Could not Implement the concept for successfully developing a search engine. Whereas, author of this paper (Cording et al.,2011) have Investigated the potential of using approximate tree pattern matching based on the tree edit distance and constrained derivatives for web scraping. But, If the set of web pages to learn from is not sufficiently large, the web scraping phase will fail because algorithms for the top-down mapping are used. In this paper[(Uzun E., 2020) has used a novel approach, which provides time efficiency by employing the string methods and additional information obtained from web pages of a website during a crawling process, is introduced. But, they could not develop an algorithm that automatically obtains the desired element without creating a DOM tree.Authors of this paper (Udapure et al.,2014) discussed about various types of crawlers. Each crawler is specific to the specific application. But, this paper was unable to provide any new model architecture. Where as, In this paper(Sethi et al.,2021) provides a novel technique for computing the best data collection cycle for each web page that helps to render latest information to the user. Also, authors Throw some light on the web crawling previous work and discuss the various researches related to web crawler (Kausar et al.,2013).Authors of this paper(Kim et al.,2020) Proposes an efficient distributed crawler through stepwise crawling node allocation (EDC-SCNA), which can efficiently operate multiple crawling nodes. But, we always need to keep it mind that The number of nodes in the experiment cannot be increased unlimitedly because a host might be overloaded when numerous nodes are allocated.In this paper (Zhao et al.,2015) concern mainly about one mid-size high quality web crawler, which becomes perfect in every aspect like robustness, scalability and efficiency. But it should always be remembered that Large-scale search engines can only provide general users with common rather than custom search services,

impossible to consider diversified requirements of different users. Authors have illustrate the general architecture for a parallel crawler which include multiple crawling processes. It is designed to be scalable parallel crawler (Sharma et al, .2011). but ,this paper Cannot index database to increase work efficiency of the overall system. Inter-processes communication need to be added. Some mechanism for removing duplicate downloads need to be included in cases where different URLs point to same page. Authors presents an overview of search engines and its techniques. In order to improve retrieval accuracy of Web search, we studied its architecture and tools for web based retrieval. But that Cannot contribute for indexing a target Web page more accurately and allowing each user to understand, perform more fine-grained search (Sharma et al, .2015). Authors focus on the gathering part of the envisioned framework, and present a novel architecture that is able to transparently provide a crawling infrastructure for a variety of CTI sources in the clear, social, and dark web.(Koloveas et al,.2019). but this paper Cannot involve the extraction of cyber-threat intelligence from the relevant harvested content, by utilizing natural language understanding for named entity recognition/disambiguation. In this paper (Yang & Thiengburanathum, 2021, March) the proposed framework was implemented. It focuses on evaluating the crawlers based on scalability and robustness on e-commerce websites. But this paper has more testing time and experimental groups should be considered, which can improve the accuracy of the experiment.(Jain et al,.2013) Propose architecture for the webcrawling using focused crawler techniques. the method to convert unstructured data into structured data.(Shrivastava, 2018) gave a detailed study of web crawler, its architecture and types. They overviewed that It is necessary to design a crawler that can be a hybrid one and serve diversified purpose by applying threading and rapid text mining using AI tools.

CHAPTER 3

Methodology

Search engines do not have a mystical knowledge of which webpages are available online. Before delivering the appropriate sites for search queries, or the terms users use to discover a beneficial page, the programs must crawl and index them. Web crawler programs are used by search engines as their aides to scan the Internet for sites before saving that webpage data to be used in subsequent requests. Crawlers begin their crawling by downloading the robot.txt file from the website. Sitemaps that list the URLs that the search engine may crawl are included in the file. Web crawlers use links to find new pages after they begin crawling a page. In order to crawl the newly found URLs later, these crawlers add them to the crawl queue. These methods enable web spiders to index every page that has links to other pages. It's important to determine the period how search engines should scan pages because they change constantly. Multiple algorithms are used by search engine crawlers to decide how frequently a existing site should have been re-crawled and the number of pages on a website should be indexed. The crawler is implemented using Beautiful soup and a scrapy framework. An open-source project called the Scrapy Framework can quickly and effectively extract data from big web pages. The Beautiful Soup library is used to parse HTML components. Users must first enter the web application's base URL. Since the base URL serves as the process's starting point, Since this tool is totally automated, no human agent is required. The basic URL is first added to a queue. The First in First out (FIFO) method is used in this study to store URLs in the database. In order to prevent process overlap, this queue is used to track which URL is visited. After being visited, the URL is saved in a database. The downloaded material is then used to extract all possible URLs from HTML. after filtration important URLs are then added to a priority queue and marked as important. Inserting all of the retrieved URLs into a queue. And iteratively repeats each procedure. The objective of this method is to extract from the web application all possible URLs connected to direct object references. User personal information may contain URLs that are direct object reference candidates and contain some type of id.

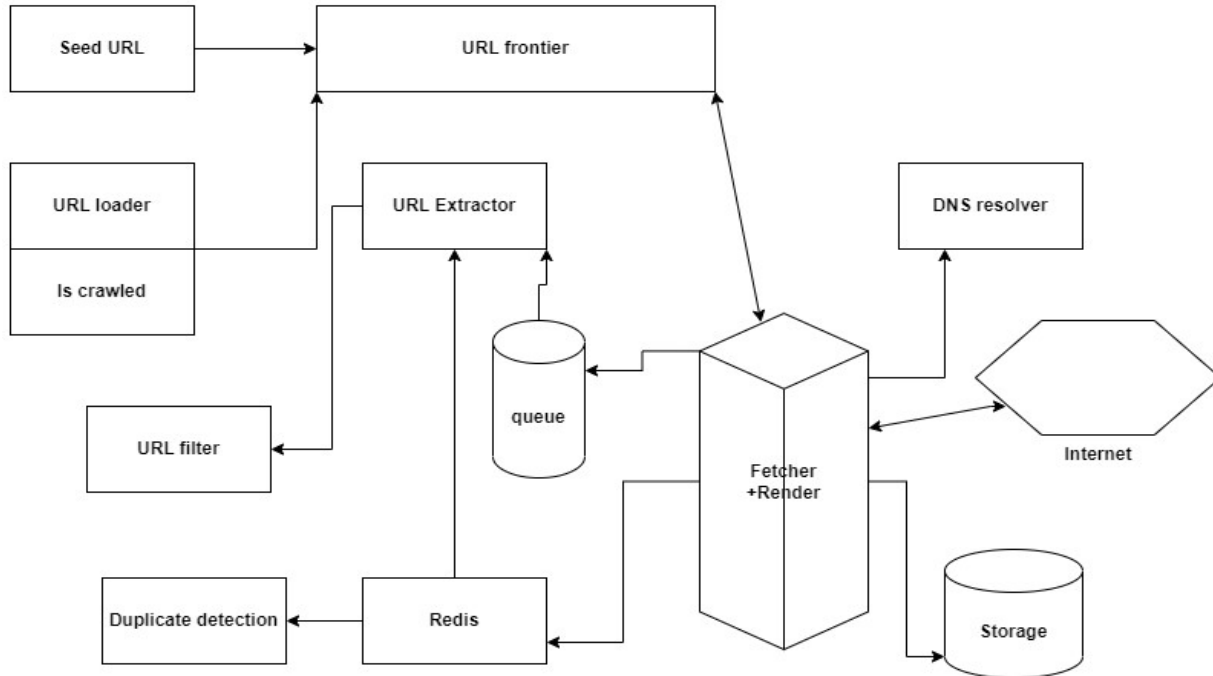


Figure 1 : Block Diagram of The Methodology

3.1 Seed URL:

The seed URL serves as the crawler's starting point or initiator URL. Any crawler must visit a site in order to continue obtaining URLs from that page and then continue crawling in a recursive manner. Therefore, it must begin by providing some seed URLs. The best approach is to gather URLs from a variety of domains, such as education, finance, sports, politics and more. Assemble the top 100-150 links from them. Make a list of everything. Thus, it can be used as seed URL. A crawler will collect extra information from a website and search including all links to other sites once it has reached a seed URL. The extent of the crawls as well as the substance of the collection will depend on how the seeds URLs were chosen and organized. If a crawler is configured to scan an entire domain, it will go through each link on each and every page and collect data from each subsequent page. Therefore, we can push it to URL frontier.

In this paper, few ways of handling seed URL were introduced.

3.1.1 Transfer Current Seeds To a New Seed URL Collection:

- i) Individually, add any seed to new collections.
- ii) Create new collections of seed URLs using set.

3.1.2 Deleting Or Deactivating Any URL From Seed URLs

- i) No archived data will be lost when a seed is deleted. All relevant data, including crawl records and notes, will be deleted when a seed is deleted. The seed list for your collection won't contain any deleted seeds anymore.
- ii) A seed can be deactivated to eliminate it entirely from any spontaneous crawls. Non active seeds still continue to be on your collection's Seeds tab.

3.2 URL Frontier:

The crawl process at a node provides a URL to the URL frontier . When a crawler thread requests a URL, it keeps track of URLs throughout the frontier and references them in a specific manner. Frontier is essentially a queue that stores the unvisited URLs, it is typically used in conjunction with a frontier manager to specify and arrange the queue's operational procedures. To begin the crawling operation, the user sets up the frontier with seed URLs. All URLs are scanned to see if there are any repetitions. The frontier manager first downloads the appropriate web pages before extracting all of the sub URLs that were identified on the page after receiving the URLs for retrieval. The HyperText Transfer Protocol (HTTP) and, more recently, the HTTPS protected version are used to carry out the downloading procedure. The crawler first acts as a web client, sending HTTP queries to the server and reading the contents of the response. Once the connection has been established, the crawler sets a time out state. New URLs are added to the crawl history after filtering.

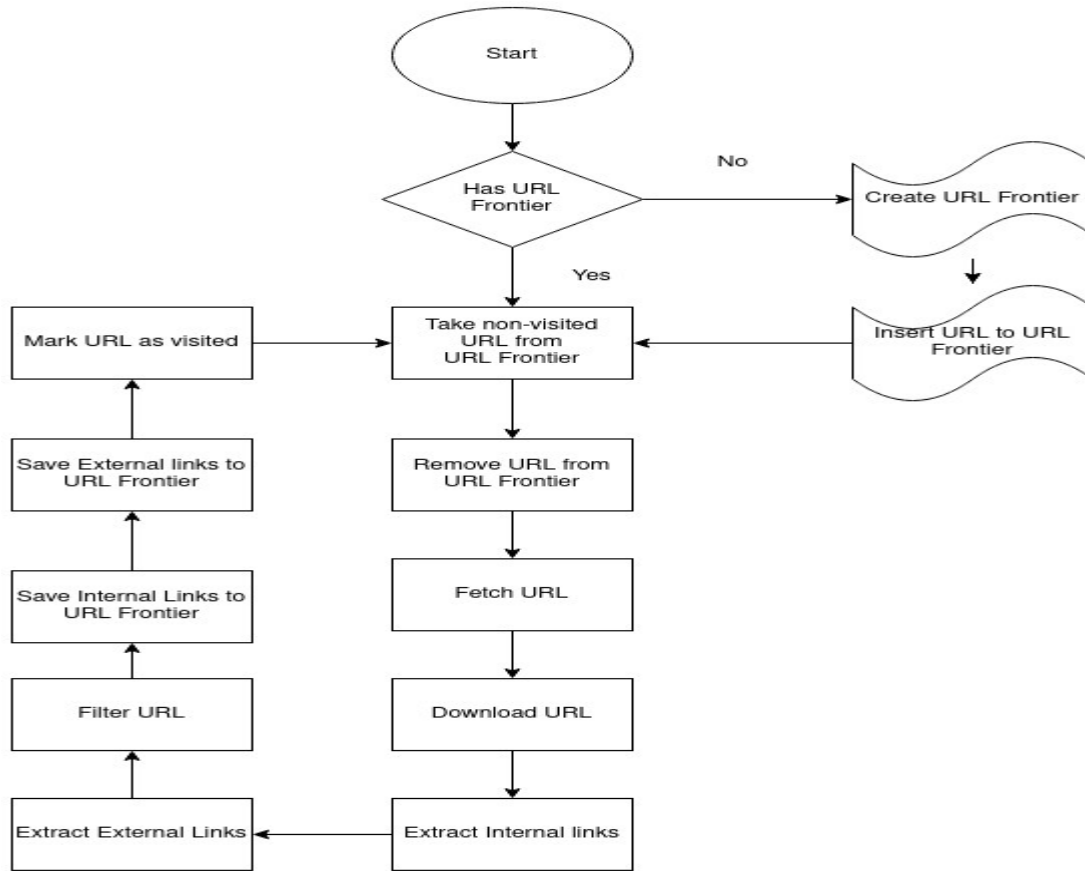


Figure 2: Flow Chart of URL Frontier

3.2.1 URL Collection & Scanning Phase

The crawler starts making HTTP requests towards other machines over the Internet for content when an user clicks a link in Web browser. Each URL is added to a queue. The program will go to an existing URL seed if users want to utilize it as their seed. If users enter a new URL, numerous programs called URL grabbers will copy it from www. After being fetched, each URL is kept in the crawl history. To check for duplicate URLs, the system examines every URL. After filtering them, it sends those URLs to the seeds.

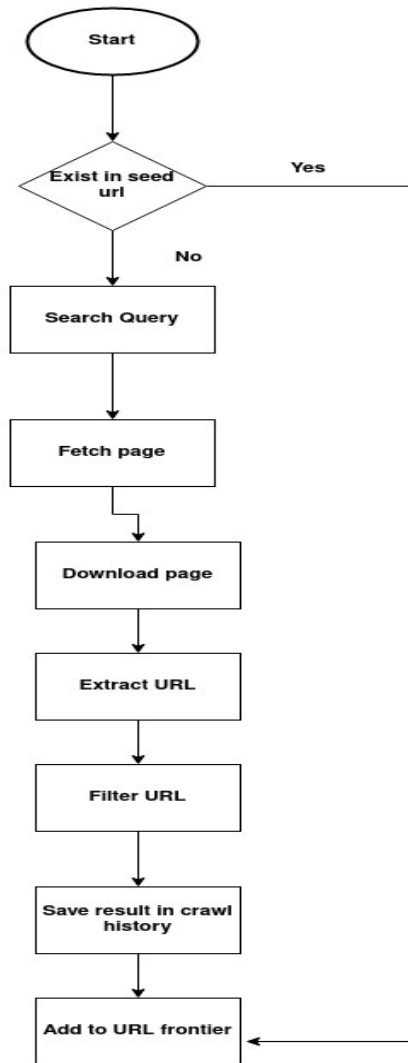


Figure 3 : URL collection & Scanning Phase

3.3 Duplicate Detection

Any URL that is processed must be checked to see if it is valid or not. It will be removed if it is invalid. If the URL is valid and not already included in the seed URL, it will be added to the frontier. Even if a URL is valid but contains a duplicate copy, that URL will also be removed. Any URL will only be added to the frontier if it is unique and is not already present in the seed URL.

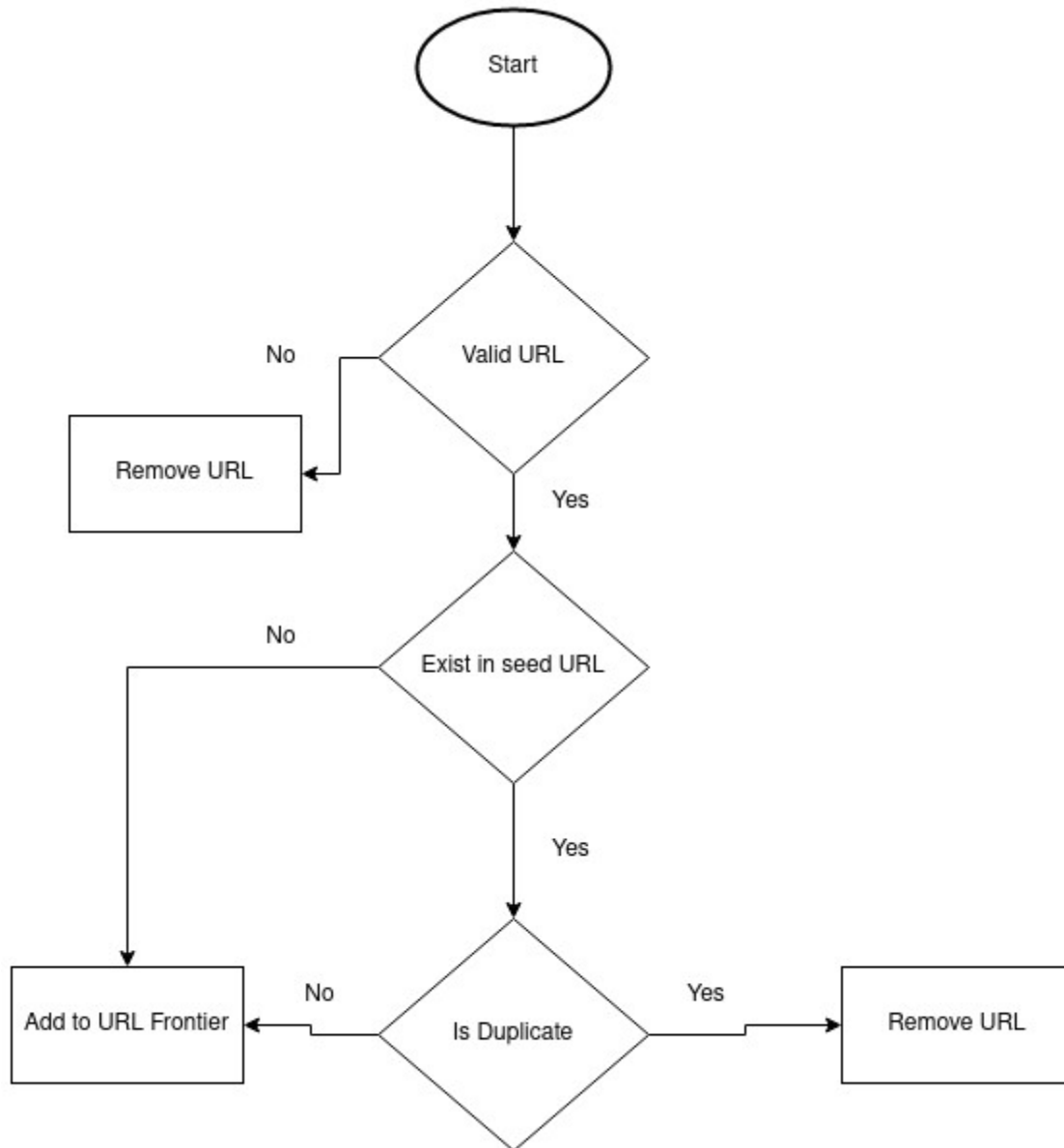


Figure 4 : Flow Chart of Duplicate Detection

3.4 URL Extraction:

Because of the huge number of web sites that are browsed per unit of time, the basic problem with URL frontiers is indeed the URL priority assignment, which has an impact on how quickly the frontier fills up. Therefore, in order to retrieve URL from frontier manager in an ideal way, the crawler must be provided with an effective extraction algorithm. The crawler begins parsing and reading the contents of the web pages after extracting them from the internet and

downloading them, which enables it to fetch the sub URLs and maintains it in the extraction loop. Parsing can be as easy as extracting links from the web pages that have been retrieved, or it can entail more complex methods like analyzing the HTML code to determine the next URL to browse. Before downloading pages, the system verifies that the URL is valid and checks for errors. After filtering errors, expected data is downloaded and generated.

3.5 Store pages:

Downloaded data is stored in local storage.

CHAPTER 4

Implementation

There are several libraries that make the implementation details of the web crawler more simpler, thus the development of the web crawler relies primarily on the programming language used. Python offer libraries and APIs for opening URL streams, reading buffers, building queues, performing input/output operations, and pattern recognition. As a result, designing and creating a unique web crawler heavily relies on implementation tools. In this paper , author has used scrapy framework , beautiful Soup library and argument parser from python built in libraries.

The implementation details are described below :

In this paper, authors have used argument parser to extract URL. if we had done it manually, that would have been hassle, we had to validate that. also there were possibility that extra data could have been added. Therefore, author has chosen built in library of python. The method crawl is implemented with one parameter (URL) , author has named that parameter as URL , the method crawles the given URL recursively, every internal and external links being visited in this method , recursively. Inside the crawl method authors have called another method called get_all_website_links, this method has one parameter named URL, main task of this method is to save URLs into a queue, thus we can use it as URL frontier. it also separates each and every internal link and external links as well as extracts them and saves them. The crawler will continue to iterate through the extraction process until the values of URLs are compliant with protocol, politeness, and type criteria. It also saves all the linked up website with a certain website. Then it passes all those links to be crawled . Inside the get_all_website_links method, we have used another method called is_valid, it has a boolean return type, again it has that same one parameter called URL, this method checks if a given URL is valid or not, if it has both protocol and domain inside it. It makes sure that only valid links are being visited and crawled.

CHAPTER 5

Result & Discussion

In this part, the author examine the crawler by two websites to crawl web pages. The author utilize the well-known web application <https://www.algoshot.com> for the first assessment. To crawl this website, we deploy 3 different types of crawlers.

TABLE 1

Test results of web crawlers

Web crawler	Processing time	activities
Parallel Web Crawler	21 sec	Visit page, check error, download page
Distributed web crawler	18 sec	Visit page, extract links, download page
Generic purpose crawler	23 sec	Specific page visit, download page

Environment 02

For this environment we crawled the website using different web crawlers by various authors.

Crawler by this author	15 seconds
Crawler by vikas_lalwani	21 seconds

Crawler by sb_t	25 seconds
-----------------	------------

In this paper, author were able to crawl the website in 15 seconds, where as crawler by vikash lalwani were running in 21 seconds and crawler by sb_t were in 25 seconds

Chapter 6

Conclusion

Web crawlers play an increasingly crucial role as the web expands and its content becomes more diverse. Web crawlers are thought to present fascinating problems as well as several benefits. A lot of difficulties must be overcome in order to establish a better web crawler because of how gigantic the web is and how challenging it is to give such extensive coverage. Different web crawling techniques have been examined in order to maximize the content of the Search Engine's repository. In this paper, web crawling techniques have been examined from a variety of crawler perspectives. Effective data collection will become increasingly important as more information becomes available. As the volume of data on the internet increases, it is necessary to become more efficient. As a consequence, new algorithms are needed to deliver more accurate search results faster. Therefore, the author will concentrate on improving the suggested model's accuracy more and more.

Reference

Deshmukh, S., & Vishwakarma, K. (2021, May). A Survey on Crawlers used in developing Search Engine. In *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)* (pp. 1446-1452). IEEE.

Cording, P. H., & Lyngby, K. (2011). Algorithms for web scraping. *Lyngby: Technical University of Denmark.[Consulta: 14 Junio 2017]*.

Koloveas, P., Chantzios, T., Tryfonopoulos, C., & Skiadopoulos, S. (2019, July). A crawler architecture for harvesting the clear, social, and dark web for IoT-related cyber-threat intelligence. In *2019 IEEE World Congress on Services (SERVICES)* (Vol. 2642, pp. 3-8). IEEE.

Uzun, E. (2020). A novel web scraping approach using the additional information obtained from web pages. *IEEE Access*, 8, 61726-61740.

Udapure, T. V., Kale, R. D., & Dharmik, R. C. (2014). Study of web crawler and its different types. *IOSR Journal of Computer Engineering*, 16(1), 01-05.

Sethi, S. (2021). An optimized crawling technique for maintaining fresh repositories. *Multimedia Tools and Applications*, 80(7), 11049-11077.

Kausar, M. A., Dhaka, V. S., & Singh, S. K. (2013). Web crawler: a review. *International Journal of Computer Applications*, 63(2).

Kim, H., Byun, J., Na, Y., & Jung, Y. (2020). Implementation of Efficient Distributed Crawler through Stepwise Crawling Node Allocation. *Journal of Advanced Information Technology and Convergence*, 10(2), 15-31.

Zhao, H. (2015). Research on detection algorithm of WEB crawler. *International Journal of Security and Its Applications*, 9(10), 137-146.

Sharma, S., Sharma, A. K., & Gupta, J. P. (2011). A novel architecture of a parallel web crawler.

International Journal of Computer Applications, 14(4), 38-42.

Sharma, S., & Gupta, P. (2015, May). The anatomy of web crawlers. In *International Conference on Computing, Communication & Automation* (pp. 849-853). IEEE.

Yani Achsan, H. T., & Wibowo, W. C. (2013). A Fast Distributed Focused-Web Crawling. *Annals of DAAAM & Proceedings*, 24(1).

Ahmadi-Abkenari, F., & Selamat, A. (2010). A Clickstream-based Focused Trend Parallel Web Crawler. *International Journal of Computer Applications*, 9(5), 1-8.

Dong, S., Lu, X., Zhang, L., & He, K. (2003, December). An efficient parallel crawler in grid environment. In *International Conference on Grid and Cooperative Computing* (pp. 229-232). Springer, Berlin, Heidelberg.

Yang, D., & Thiengburanatham, P. (2021, March). Scalability and Robustness Testing for Open Source Web Crawlers. In *2021 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering* (pp. 197-201). IEEE.

Jain, N., Mangal, M. P., & Bhansali, A. (2013). An Approach to build a web crawler using Clustering based K-Means Algorithm. *Journal of Global Research in Computer Science*, 4(12), 14-22.

Shrivastava, V. (2018). A methodical study of web crawler. *Journal of Engineering Research and Application*, 8(11), 01-08.

Yu, L., Li, Y., Zeng, Q., Sun, Y., Bian, Y., & He, W. (2020). Summary of web crawler technology research. In *Journal of Physics: Conference Series* (Vol. 1449, No. 1, p. 012036). IOP Publishing.

Cho, J., & Garcia-Molina, H. (2002, May). Parallel crawlers. In *Proceedings of the 11th international conference on World Wide Web* (pp. 124-135).

Yadav, D., Sharma, A. K., & Gupta, J. P. (2008). Parallel crawler architecture and web page change

detection. *WSEAS Transactions on Computers*, 7(7), 929-940.

Udapure, T. V., Kale, R. D., & Dharmik, R. C. (2014). Study of web crawler and its different types. *IOSR Journal of Computer Engineering*, 16(1), 01-05.

Yang, D., & Thiengburanatham, P. (2021, March). Scalability and Robustness Testing for Open Source Web Crawlers. In *2021 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering* (pp. 197-201). IEEE.

Fetzer, C., Felber, P., Rivière, É., Schiavoni, V., & Sutra, P. (2015, June). Unicrawl: A practical geographically distributed web crawler. In *2015 IEEE 8th International Conference on Cloud Computing* (pp. 389-396). IEEE.

C. Olston and M. Najork. Web crawling. *Foundations and Trends in Information Retrieval*, 4(3):175–246, 2010. ISSN 1554-0669

Dhenakaran, S. S., & Sambanthan, K. T. (2011). Web crawler-an overview. *International Journal of Computer Science and Communication*, 2(1), 265-267.

<https://www.webfx.com/blog/internet/what-is-a-web-crawler/>

<https://blog.diffbot.com/knowledge-graph-glossary/seed-URL/>

<https://support.archive-it.org/hc/en-us/articles/208331753-Select-Seed-URLs>

<https://research.aimultiple.com/web-crawler/>

