AN EFFECTIVE IMPLEMENTATION OF WEB CRAWLING TECHNOLOGY TO RETRIEVE DATA FROM THE WORLD WIDE WEB (WWW)

F. M. Javed Mehedi Shamrat, Zarrin Tasnim, A.K.M Sazzadur Rahman, Naimul Islam Nobel, Syed Akhter Hossain

Abstract: Internet (or just the web) is enormous, well off, best, easily accessible and proper wellspring of data and its clients are expanding quickly now daily. To rescue data from the web, web indexes are utilized which access pages according to the prerequisite of the clients. The size of the web is exceptionally wide and contains organized semi-organized and unstructured information. The greater part of the information present on the web is unmanaged so it is absurd to expect to get to the entire web without a moment's delay in a solitary endeavor, so web crawlers use web crawlers. A web crawler is a fundamental piece of the web search tool. Data Retrieval manages to look and recovering data inside the reports and it likewise looks through the online databases and the web. In this paper, discussed, developed and programmed a web crawler to fetch the information from the internet and filter data for useable and graphical purpose for users.

Keywords: Web Crawling, Web Technology, Data, Python, Data Extraction, Algorithm.

1. INTRODUCTION

THE World Wide Web (WWW) is a web customer server design. It is an incredible framework dependent on complete independence to the server for serving data accessible on the web. The data is masterminded as a huge, circulated, and non-direct content framework known as the Hypertext Document framework. These frameworks characterize some portion of a report as being hypertext-bits of content or pictures which are connected to different records by means of stay labels. HTTP and HTML present a standard method for recovering and introducing the hyperlinked records. Web programs, use web crawlers to investigate the servers for required pages of data. The pages sent by the servers are prepared at the customer side. Presently days it has turned into a significant piece of human life to utilize Internet to obtain entrance data from WWW. The present populace of the world is about 7.049 billion out of which 2.40 billion individuals (34.3%) use Internet [1] (see Figure 1). From .36 billion of every 2000, the measure of Internet clients has expanded to 2.40 billion out of 2012 i.e., an expansion of 566.4% from 2000 to 2012. In Asia out of 3.92 billion individuals, 1.076 billion (i.e.27.5%) use Internet, though in India out of 1.2 billion, .137 billion (11.4%) use Internet. The same development rate is normal in future as well and it isn't far away when one will begin reasoning that life is deficient without Internet. Figure 1: outlines Internet Users in the World by Geographic Regions.

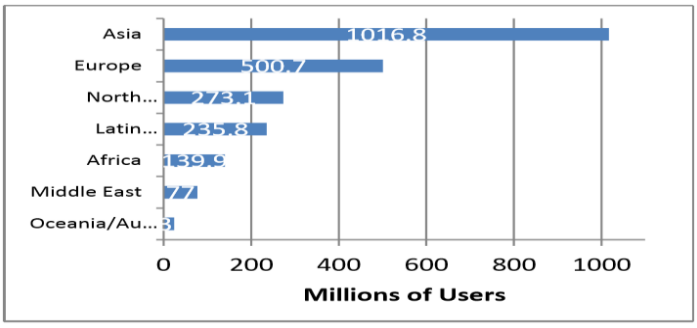

Fig. 1: *Internet Users in the World by Geographic Regions (Source: http://www.internetworldstats.com accessed on May 7, 2012).*

Starting in 1990, the World Wide Web has developed exponentially in size. Starting today, it is assessed that it contains around 55 billion openly list capable web reports [2] spread everywhere throughout the world on a huge number of servers. It is difficult to scan for data from such an immense accumulation of web reports accessible on WWW. Web crawler is a significant strategy for gathering information on and staying aware of, the quickly growing Internet. Web creeping can likewise be called a diagram search issue as web is viewed as a huge chart where hubs are the pages and edges are the hyperlinks. Web crawlers can be utilized in different regions, the most unmistakable one is to list an enormous arrangement of pages and enable other individuals to look through this record. A Web crawler doesn't really move around PCs associated with the Internet, as infections or shrewd operators do, rather it just sends demands for archives on web servers from a lot of as of now areas. The general procedure that a crawler takes is as per the following,

- It checks for the following page to download – the framework monitors pages to be downloaded in a line.
- Checks to check whether the page is permitted to be downloaded - checking a robot's prohibition document and furthermore perusing the header of the page to check whether any rejection directions were given do

- *F. M. Javed Mehedi Shamrat is currently pursuing Bachelor's degree program in Software Engineering at Daffodil International University, Bangladesh. E-mail: javedmehedicom@gmail.com*
- *Zarrin Tasnim is currently pursuing Bachelor's degree program in Software Engineering at Daffodil International University, Bangladesh. E-mail: zarrint25@gmail.com*
- *A. k. M. Sazzadur Rahman Rahman is currently pursuing master's degree program in Computer Science and Engineering at Daffodil International University, Bangladesh. E-mail: sohag933@gmail.com*
- *Naimul Islam Nobel is currently pursuing Bachelor's degree program in Software Engineering at Daffodil International University, Bangladesh. E-mail: nobel775#diu.edu.bd*
- *Syed Akhter Hossain, Professor and Head of Department of Computer Science and Engineering at Daffodil International University, Bangladesh. E-mail: aktarhossain@daffodilvarsity.edu.bd*

this. A few people don't need their pages filed via web indexes.

- Download the entire page.
- Extract all connections from the page (extra site and page locations) and add those to the line referenced above to be downloaded later.
- Extract all words and spare them to a database related to this page, and spare the request for the words so individuals can look for phrases, not simply catchphrases
- Optionally channel for things like grown-up content, language type for the page, and so forth.
- Save the outline of the page and refresh the last handled date for the page with the goal that the framework knows when it should re-check the page at a later stage.

2 LITERATURE REVIEW

Many researchers have used web crawlers to get web data for their research work. These research papers are helpful in analyzing the present work done and detecting the lacunas which remain unsolved in the current work. Web crawling can be used in web mining field to automatically discover and extract information from the WWW. The focus of the paper is to suggest a web crawler that uses a set of queries from a list of keywords to crawler through webpages rather than indexing all the webpages on the internet. For that, first, a set of seed URLs are set from where one URL is explored. Using Google API and the search interface of the URL, a query is made with the keyword list. Both the results are gathered, merged and a fitness value is calculated [3]. Finally this fitness value is used to index which webpage should be explored by the web crawler. In the paper, the authors suggested an architecture that scraps data from the provided addresses and retrieves the data. This data is classified in using a Naïve Bayes classification and saved in the database. When a job seeker makes a request, the scraper scrapes the data from the website, is categorized and stored in the database and response to the request is made [4]. The main focus of the paper is to propose a three-stage framework to extract web interfaces. First of all, using a search engine, the crawler searches for a webpage but avoiding search a large number of pages. It prioritizes the pages based on relevance. Secondly, the crawler ranking the links in the pages with a fast in-site search. Finally, an admin will collect the extracted data and process the top results [5]. The main objective of the paper is to extract data from websites using hyperlinks provided. The extracted data are mainly unstructured data. Finally, the authors show a comparison between TF-IDF algorithm and the BFS algorithm to show the accuracy rate suggesting TF-IDF algorithm gives more accurate results [6]. From the research gap, a web crawler is proposed that crawlers into the career section of a company's website. If a job circular is posted, it extracts the data from the post. This raw data is saved into the database. The web crawling is done automatically and the data gathered is processed and stored automatically as well.

3 METHODOLOGY

In this phase, a web crawler was implemented. A web crawler that is also called a web spider is a program that browses the web in a methodical manner to gather information. Web crawlers are used to gather data or copy the pages of any website it visits. But most importantly a web crawler is used to gather some specific data from a web site. Most of the website posts job circular for companies that are looking for recruitments. In order to post the circular on the website, the system has to extract the job circular post from the career section of each of the company's website. Using web crawling, the job circular information will be extracted from the websites which will be saved as parameters in the database of skill.jobs. In the web crawler that is implemented, a number of URLs of the career section of the companies' website were set in a queue. The crawler gets URLs from the queue and visits the webpages. From the web pages it extracts the posts of the job circulars published by the companies, then copy and save the information in the database of the website.

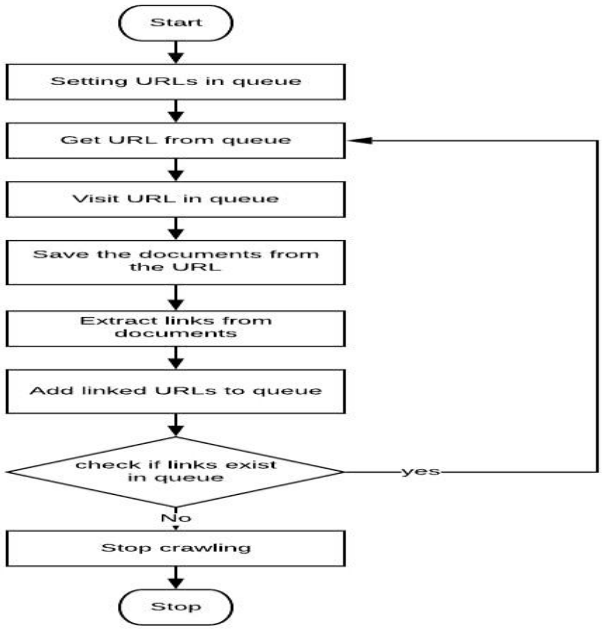

Fig. 2: *Sample of the process of Decision Trees.*

A web crawler system is designed, developed and implemented. To develop this system an algorithmic program is designed for implemented the developed module on the required system.

3.1. Algorithm 1: Web Crawler Working Procedure

1: Import Libraries;
2: Process_Function (max_pages){ → [Working Function]
3: Initializing Value;
4: While (page <= max_pages) → [Variable 'page' is less than 'max_page']
5: Set Url = (Web_Link); → [Webpage URL from where information is to be extracted]
6: Create Object; → [Create object for get all the information, returned values of the HTTP request]
7: Object = Url Request;
8: Convert Object to Plain_Text;
9: Create Object; → [Create object to pull out the information using 'BeautifulSoup']
10: For() { → [Used traverse through the information (HTML file) that is pulled out]
11: Declare Variable;

12: Variable = Concat_Text;
13: Display Variable;
14: }
15: Increment value;
16: Crawler_web (Argument) → [Function is called with argument '1']

3.2. Algorithm 2: Download and Store Data Using Hyperlink Procedure

1: Import Libraries;
2: Declare Variable;
3: Variable = Link Extracted; → [link extracted by the web crawler in saved in the variable]
4: Function_download_csv (csv_url); → [Function 'download_csv()' with parameter 'csv_url']
5: Create Object;
6: Object = Extract Information; [Object is created to extract the information from the link]
7: Declare Variable;
8: Variable = Read and Stored Data;
9: Declare Variable2;
10: Variable2 = String_format (Information); → [information is converted to string format and stored in variable]
11: Declare Variable3;
12: Variable3 = String_Spilit; → [String split in variables and saved in 'Variables3]
13: Create Object2;
14: Object2 = Create a CSV file; → [Object is created to create a csv file]
15: Open CSV File;
16: For (){ }→ [Using a 'for' loop each line variable is written on the csv file]
17: CSV_File = Closed and Stored;

4 RESULT AND IMPLEMENTATION

To develop and implement this system used python programming language and its default libraries. Data extraction, read-write and stored functions are fetched and imply by python raw code. Any web site can adopt this system to use web-crawling technology to collect data from World Wide Web (www).

```
temp.py*  web_crawler2.py
1 from urllib import request
2
3 notice_url= 'http://www.data.gov.bd/api/download/?id=02b25f92-8a24-489c-906e-3b5703650df4'
4
5 def download_csv(csv_url):
6     response = request.urlopen(csv_url)
7     csv = response.read()
8     csv_str = str(csv)
9     lines = csv_str.split("\\n")
10    dest_url=r'dataset.csv'#txt / csv
11    fx=open(dest_url,"w")
12    for line in lines:
13        fx.write(line+"\n")
14    fx.close()
15
16 download_csv(notice_url)
17
18
```

Fig. 3: *Code for the Web Crawler.*

In Figure 3, at first, I have imported the libraries' requests, BeautifulSoup, and request. Then a function named crawler_web() with the parameter max_pages is declared. In the function first a variable is declared and initialized with the value 1. A while loop is implemented with the condition that the loop is true as long as the variable 'page' is less than 'max_page'. In the loop an URL of a webpage is set from where information is to be extracted. We create an object 'source_code' for which we get all the information where we save the returned values of the HTTP request. The values are converted to plain text and save it in variable 'plain_text'. To pull out the information (HTML file) using 'BeautifulSoup' we create an object. Now a 'for loop' is used traverse through the information (HTML file) that is pulled out. If in the file there is a class 'btn bth-primary data-link' the link next to it will be extracted, then concat with the text 'http://www.data.gov.bd' and saved in the variable href. The variable title while saving the string of the link. The title and the link will be printed out in the output console. After the for loop ends, the variable 'page' will be incremented with 1 and the while loop will continue. The function 'crawler_web()' is called with argument '1'.

Fig. 4: *Hyperlinks of Files as Output of the Web Crawler.*

In Figure 4, we can see, in the HTML file, the link did not contain any text so there is no title and the variable contained nothing and 'None' is printed out. In the next line, we see an HTTP link that the web crawler extracted from the webpage. All the links on the page are extracted.

```
temp.py*  web_crawler2.py*
1  import requests
2  from bs4 import BeautifulSoup
3  from urllib import request
4
5  def crawler_web(max_pages):
6      page=1
7      while page <=max_pages:
8          url='http://www.data.gov.bd/export-and-import-riceddata-of-bangladesh'
9          source_code=requests.get(url)
10         plain_text=source_code.text
11         soup=BeautifulSoup(plain_text,features="html.parser")
12
13         for link in soup.find_all('a',{'class':"btn btn-primary data-link"}):
14             href="'http://www.data.gov.bd"+link.get('href')
15             title=link.string
16             print(title)
17             print(href)
18             print("\n")
19
20         page+=1
21
22 crawler_web(1)
23
24
```

Fig. 5: *Code to Download and Store Data Using Hyperlink.*

In Figure 5, the library 'request' is imported from 'urllib'. Then a link extracted by the web crawler in saved in the variable 'notice_url'. A the function 'download_csv()' with parameter 'csv_url'. The function is called with 'download_csv()' with the argument 'notice_url'. The function receives the link as its parameter. An object 'response' is created to extract the information from the link. The information is read and stored in variable 'csv'. The information is converted to string format and stored in variable 'csv_str'. The strings are then split into lines and saved in 'lines'. Now an object named 'dest_url' is created to create a CSV file named 'dataset.csv'. With the object fx, the 'dest_url' file is opened and written on. Using a 'for' loop each line from 'line' variable is written on the CSV file. Then the file is closed and stored.

Fig. 6: *Successful Downloading of File.*

In Figure 6, we can perceive, that the code runs successfully without any error.

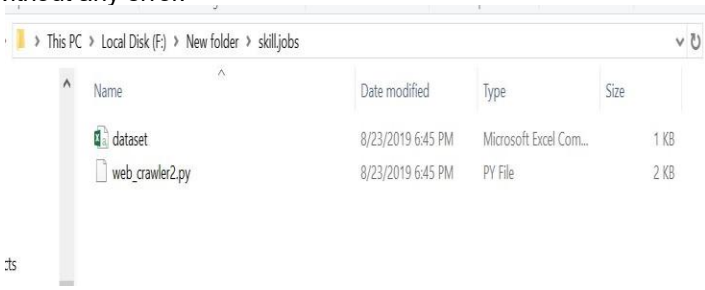

Fig. 7: *Downloaded File Created.*

In Figure 7, the CSV file name 'dataset.csv' is created in the same directory as the python file of the code is saved. If a file with the name does not exist, it will create a new file, and if the file with the name exists, it writes on the existing file.

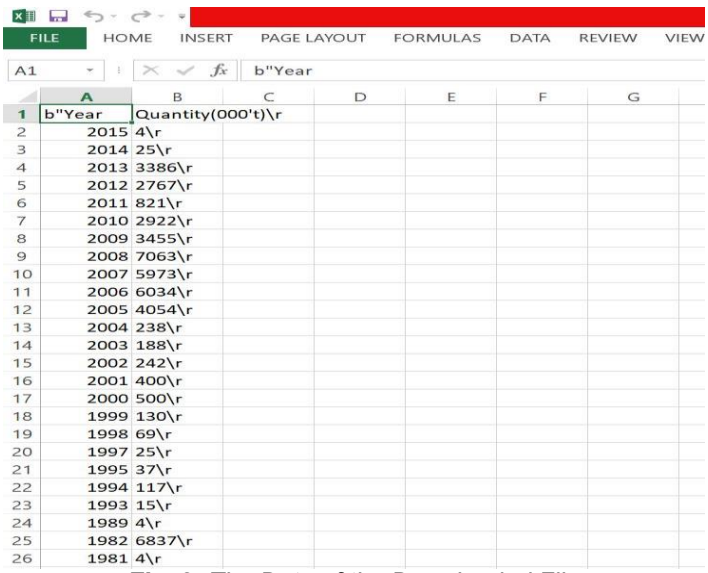

Fig. 8: *The Data of the Downloaded File.*

In Figure 8, inside the file 'dataset.csv' the dataset that was present in the webpage that was extracted using the web crawler is being copied and stored. Using this stored information, further processing will be done.

5 CONCLUSION

Web Crawlers are a significant part of the web crawlers. Web slithering procedure regarded elite are essential segments of different web administrations. It's anything but an insignificant issue to set up such frameworks: Data control by these crawlers spread a wide region. It is significant to save a decent harmony between irregular access memory and plate gets to. A web crawler is a way for the search engines and other users to regularly ensure that their databases are up to date. Web crawlers are a central part of search engines, and details on their algorithms and architecture are kept as business secrets. When crawler designs are published, there is often an important lack of detail that prevents others from reproducing the work. There are also emerging concerns about" Search Engine Spamming", which prevent major search engines from publishing their ranking algorithms. New Modification and extension of the techniques in Web crawling should be the next topics in this area of research.

ACKNOWLEDGMENT

The authors are grateful and pleased to all the researchers in this research study.

REFERENCES

[1] Internet Access All Over The World: http://www.internetworldstats.com accessed on May 7, 2012, Last Access: 20.11.2019.

[2] World Wide Web Timeline: https://www.pewresearch.org/internet/2014/03/11/world-wide-web-timeline/, Last Access:20.11.2019

[3] Manish Kumar, Ankit Bindal, Robin Gautam and Rajiesh Bhatia, "Key word query based focused Web crawler", 6th Internation conference of smart computing and communications, ICSCC 2017

[4] C Slamet, R Andrian, D S Maylawati, Suhendar, W Darmalaksana and M A Ramdhani “Web Scraping and Naïve Bayes Classification for Job Search Engine”, The 2nd Annual Applied Science and Engineering Conference (AASEC 2017)
[5] Jeny Thankachan and Mr. S. Nagaraj,“Intelligent Web Crawler: A Three-Stage Crawler for Effective Deep Web Mining”, International Journal of Recent Trends in Engineering & Research (IJRTER) Volume 02, Issue 04; April - 2016 [ISSN: 2455-1457]
[6] Ahmed, Tanvir & Chung, Mokdong “Design and application of intelligent dynamic crawler for web data mining ”, Korea Multimedia Society, Spring Conference 2019.
[7] S. Saranya, B.S.E. Zoraida, and P.V. Paul, "A Study on Competent Crawling Algorithm (CCA) for Web Search to Enhance Efficiency of Information Retrieval," Proceeding of Arti ficial Intelli gence and Evolutionary Algor ithms in Eng ineering Systems, Springer, New Delhi , pp. 9-16, 2015.
[8] K.S. Kim, K.Y. Kim, K.H. Lee, T.K. Kim, and W.S. Cho, "Design and implementation of web crawler based on dynamic web collection cycle," Proceeding of The International Confer ence on Information Network, IEEE , pp. 562566, 2012.
[9] Y. Kim, H. Hong, and M. Chung, "Application of Cohesion Devices for Improvement of Distributional Representation," Proceeding of The 14th International Conference on Multimedia Information Technology and Applications (MITA), pp. 84-87, 2018.
[10] M.Y. Ivory and M.A. Hearst, "Improving web site design," Proceeding of IEEE Internet Computing 2, Vol. 6, No. 2, pp. 56-63, 2002.
[11] D. Debraj and P. Das, "Study of deep web and a new form based crawling technique," International Journal of Computer Engineer ing and Technology (IJCET), Vol. 7, No. 1, pp. 36-44, 2016.
[12] Z. Guojun, J. Wenchao, S. Jihui, S. Fan, Z. Hao, L. Jiang, et al., "Design and application of intelligent dynamic crawler for web data mining," Proceeding of 2017 32nd Youth Aca demic Annual Conference of Chinese Associ ation of Automation (YAC) IEEE , pp. 1098-1105, 2017.
[13] K.A. Pakojwar, R.S. Mangrulkar, and V.G. Bhujade, "Web data extraction and alignment using tag and value similarity," Proceeding of 2015 International Conference on Innova tions in Information, Embedded and Commu nication Systems (ICIIECS), pp. 1-4, 2015.
[14] S. Kolhatkar, M.M. Pati, M.S. Kolhatkar, and M.S. Paranjape, "Emergence of Unstructured Data and Scope of Big Data in Indian Education," International Journal of Advanced Computer Science and Applica tions (IJACSA) , Vol. 8, No. 1, pp. 150-157, 2017.
[15] M. Afsharizadeh, H. Ebrahimpour-Komleh, and A. Bagheri, "Query-oriented text summarization using sentence extraction technique," Proceeding of 4th International Conference on Web Research (ICWR) , pp. 128-132, 2018.
[16] S. Ringe, N. Francis, and A.H.S.A. Palanawala, "Ontology Based Web Crawler," International Journal of Computer Applications in Engin eering Sciences, Vol. 2, No. 3, pp. 194-197, 2012.
[17] L. Jiang, Z. Wu, Q. Feng, J. Liu, and Q. Zheng, “Efficient deep web crawling using reinforcement learning,” Proceeding of Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, Berlin, Heidelberg, pp. 428-439, 2010.
[18] Y. Kim, B. Kim, and M. Chung, “Unstructured data analysis and multi-pattern storage technique for traffic information inference,” The Journal of Multimedia Information System, Vol. 21, No. 2, pp. 211-223, 2018.
[19] R. Jason and A. McCallum, "Using reinforcement learning to spider the web efficiently," Proceeding of International Conference on Machine Learning (ICML) , Vol. 99, 1999.