International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 05 | May 2022 www.irjet.net p-ISSN: 2395-0072

Smart Crawler Automation with RMI

Kanishka Kannoujia[1], Satwik Verma[2], Mansi Ahlawat[3], Mr. Ashish Kumar[4], Dr. Rajesh Kumar Singh[5]

[1,2,3]*Student, Dept. of Computer Science & Engineering, Meerut Institute Of Engineering & Technology, Uttar Pradesh, India,*
[4]*Assistant Professor, Dept. of Computer Science & Engineering, Meerut Institute Of Engineering & Technology, Uttar Pradesh, India,*
[5]*Professor, Dept. of Computer Science & Engineering, Meerut Institute Of Engineering & Technology, Uttar Pradesh, India.*

Abstract - *There is plenty of information available on the web at the moment times. As per different researches, the foremost valuable and useful data out there's present within the web internet which we cannot access through standard browsers because they will only show the info present at surface level. So, the system is intended to fetch out all relevant information, which software bot is known as Crawler. This crawler could be a part of an exploration engine that fetches the foremost relevant and active links but there are a lot of challenges that acquire the image after we think of implementing a crawler.*

The key conclusion is that the system is to seek out active links from the internet with relevant data for the user and that links will be processed further by different machines by Remote Method Invocation. This will give fast service to the client by working distribution environment. On the collection the links, the most precise document will be download for the client as per there request.

Key Words: Crawler, Depth-First Search, Breadth-First Search, Active/Non-active hyperlinks, Natural Language Processing, Remote Method Invocation, JVM, Server.

1. INTRODUCTION

The deep web is a term used to describe the contents that are hidden beneath searchable web interfaces which therefore is undiscoverable by simple web browsers. The worldwide web is thought to be separated in: visible web, hidden web, and darknet. The data from the deep web cannot be extracted at the surface [1].

The visible web is freely accessible to everyone through simple gateways like Chrome, Firefox etc. Our visible web has sites such as news, social media, politics, and so on, but the deep web contains data such as login passwords [2]. Whereas, hidden web is that part of cyberspace whose content on search is not discoverable through normal browsers; the un-discoverability is due to sites being password-protected; a file named "robot.txt" is present on every web address that specifies if you are allowed to crawl over this address or not, if it permits then for how long you are permitted to crawl that particular web address ; Certain online pages have a time limit, for example, a page or site will only be accessible for a limited time, but the deep web typically contains legal content. [3][4]. The darknet is a key component of illegal activities like drug trafficking, arms trafficking, and child abuse, to name a few. Despite the fact that it is a concealed network with its own set of applications and protocols, it is a powerful tool. Crawling is the systematic visit of various web addresses in order to construct a data guide. Techniques such as parallel, focused, generic crawler can be used to implement crawlers [1].

Several characteristics of crawler

Distributed -- In distributed environment it can be synchronized across different machines [5].

Scalability – Crawling the huge amount of data is slow, but it is possible to improve it by scaling-in and scaling-out machines or can increase network [5].

Effectiveness and performance -- When it's a preliminary visit to a web address by the crawler then for the purpose of consuming system resources at maximum, it saves the files present on the site to the local.

Reliability – Crawlers intend to prioritize collecting high-quality websites that users require, increase page retrieval accuracy, and limit the addition of extraneous documents [5].

The system expressed in this article, a crawler is purposed which leads a collection of active and non-active links and it requires seed URL is both a starting point for the crawlers as well as an access point to archived pages. This system helps to how our crawler decides what exact content does or does not belong in your archives based upon these selections.

2. RELATED WORK

The Web is always evolving and growing. Web crawler bots start from a seed or a list of known URLs because it is impossible to determine how many complete web pages there are on the Internet. The documents are retrieved by the crawler from the URLs. They will find hyperlinks to other

URLs when they crawl those web pages and add those to the list of pages to crawl next.

Spiders are web search components that retrieve sites from the Internet and extract data [9].

Algorithm: Web Crawler

Input: website link

Output: Active Link

Steps:

1. Start
2. Fetch URLs from the seed URL.
3. Determine the Host name's IP address.
4. Store the URLs in the linked list which have the same domain name.
5. The documents will be divided into RMI machines M1 and M2.
6. M1 and M2 start crawling from Step 2.
7. Check to see if the file has already been downloaded.
8. Check whether the fetched links are active or not.
9. Save active links.
10. End

In the above algorithm, in line 1 start the program, and the user can enter the seed url.

After that, the crawler starts fetching URLs from the seed URL first. And all the fetched URLs from the seed document must belong to the same domain or must have the same IP address.

All the document URLs must be unique otherwise we neglect the same. These URLs are stored in the linked list.

As soon as all the documents from Seed URLs are fetched out, the documents will be divided into RMI machines M1 and M2.

M1 and M2 start crawling from Step 1. And storing that URL in the same linked list.

If the link is active then we save the URL otherwise throw it away.

There are three techniques on which the author of web search worked upon: DFS (depth first search), BFS (breadth first search), and the optimal out of these two search strategies.

DFS (Depth First Search): For individuals with limited depth, a depth-first method is commonly used. From the start, the depth-first search method is used to visit the first web page which is seed URL and examine it before moving further, going across a path, and going across the next path. The depth-first web crawler system has the disadvantage that is predetermination of the terminating condition and how much crawling should be done because the links embedded in some web addresses are frequently of good worth, and the worth of the web address is progressively diminished as the crawler crawls over the web address. Furthermore, the structure of the cyberspace is so intricate and deep in which web spider keeps digging, which might cause issues [5].

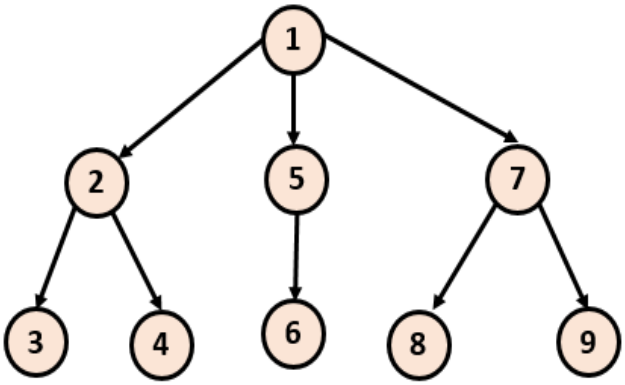

Fig. 1: DFS Graph

BFS (Breadth First Search): In more broad circumstances, the breadth-first search approach is applied. The worth of a web page is generally higher within a certain distance of the initial seed, so crawl the connection to the web from the beginning of the web page, pick one, and keep digging. The breadth-first search technique allows you to search by levels in the tree, and if the search at present level in the tree is incomplete, you won't be able to move a level ahead. In given aspect, the BFS strategy is unperceptive; it searches the entire knowledge domain, which reduces efficiency. A breadth-first search approach, on the other hand, is a suitable option if you want to focus on coverage. Figure 2 shows a network. [5]

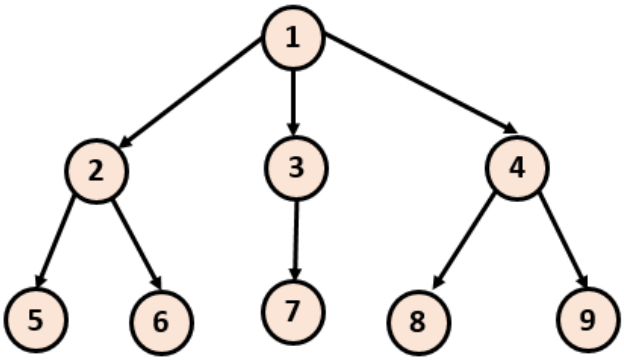

Fig 2: BFS Graph

We know our objective is near the bottom of the search tree, therefore BFS is a better option.

Visiting websites again: On the Internet, content is continually being updated, removed, or relocated. Web crawlers will periodically need to revisit pages to make sure the latest version of the content is indexed [8].

Selecting pertinent webpages (hyperlinks): Our focused web crawlers are meant to scan content that is semantic to domain and diminish the visits to extraneous links because the deep web has relatively little coverage for extracting databases, which limits data access.

Show the user active webpages: The proposed crawler can crawl the internet's web and distinguish between active and inactive hyperlinks, resulting in increased efficiency and accuracy.

3. METHODOLOGY

3.1 System Architecture

The use of web crawlers in the upcoming time is not going to decrease, perhaps it will increase exponentially because data is only increasing every second. So, the research on increasing the efficiency of crawlers will be continuing further. Users need to specify some parameters such as seed URL, filename (optional), levels to which users want to fetch hyperlinks, number of documents, and number of threads.

To begin, the graph data structure algorithm Breadth-First Search is utilized to look for links embedded in the seed URL as well as links embedded farther down in the fetched URLs. Need to ensure that any link should not contain a link to the previously fetched link i.e. duplicate links fetching should be avoided.

The crawling will start with the seed URL which is provided by the client, at the starting all the documents will be fetch out after that the documents of base URL will be distributed on the machines suppose M1 and M2. These machines will fetch all the documents one by one and will be store in the linked list. The fetching of document will never until or unless it reached to the maximum level specified by the client or the document gets empty.

After link fetching, use Bag of Words to extract features from the text, Term Frequency to get frequency of appearance of terms in text, Inverse Document Frequency to quantify the term importance in the text from Natural Language Processing to classify among large to find useful data [1].

Bag of Words: Natural Language Processing uses text modelling using a bag of words. To put it another way, it's a feature extraction method for text data. This method for obtaining data features is simple and customizable.

Term Frequency and Inverse Document Frequency: Last step left is indexing the results according to apriority of most useful to least useful. That is being done by performing data analysis on HTML tags extracted from these useful pages.

The formula to calculate Term Frequency is :

$$TF(k,l) = n(k,l) / \Sigma\, n(k,l)$$

Here, $n(k,l)$ = The number of times the nth word appeared in the document. $\Sigma\, n(k,l)$ = a document's total number of words

In mathematical terms, the TF-IDF score is calculated as follows:

$$IDF = 1 + \log(K/dK)$$

Were, K = the dataset's total number of documents

dK = total number of documents containing the nth word

Furthermore, the 1 in the preceding formula avoids the total suppression of phrases with 0 IDF. This method is known as IDF smoothing.

The TF-IDF is calculated as : TF - IDF = TF * IDF

3.2 Remote Method Innovation

To make the search process faster, we applied RMI (Remote Method Innovation). The RMI API is a Java library for creating distributed applications. An object can use the RMI to call methods on another JVM-based object. RMI make use of stub and skeleton objects to communicate with the remote object.

A remote object's method can be invoked from another Java virtual machine.

STUB:

A client-side gateway is an object called a stub. All outgoing queries are routed through it. Stub is an object at the client side that acts as gateway for a remote object. When invoking a method on the stub object, the caller has the following responsibilities.

It connects to a remote Virtual Machine (JVM), passes the parameters to the JVM, waits for the outcome, reads the return value or exception, and ultimately delivers the value to the caller.

SKELETON:

The skeleton is a gateway object which lives on the server and it also directs all inbound queries. After acquiring a request, skeleton follows these steps:

●Receives remote method's parameter and reads it.

●It uses the actual remote object to call the method, then writes and transmits the result to the caller.

●The Java 2 SDK includes a stub protocol that avoids the requirement for skeletons.

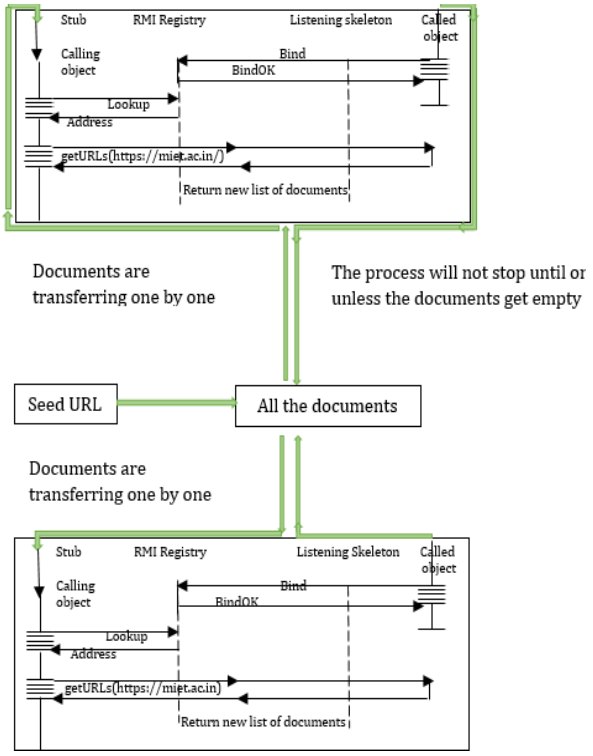

Fig. 3: Logical Operation of a Java RMI call

RMI EXAMPLE:

We followed all six steps in this example to build and launch the RMI application. The client application just requires two files: the remote interface and the client program. In the RMI application, both the client and the server use the remote interface. RMI passes the request to the remote JVM once the client program calls the proxy object's methods. The proxy object receives the return value, which is then passed on to the client application.

Step	Description
1.	Create the remote interface first.
2.	Assist with the remote interface implementation
3.	Using the rmic tool, create instances of the stub and the skeleton
4.	Now by utilizing the RMI registry utility, registry service should be started.
5.	Make the server application and run it.
6.	Make a client application and run it

Table 1: RMI working

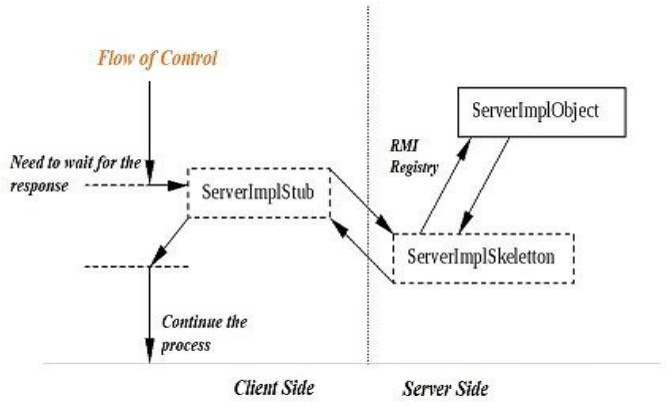

Fig. 4: Schematic overview when different actors pass the RMI request.

Optimal Solution:

Breadth-first Search technique is a fantastic place to start when it comes to obtaining URLs from one page before transferring to another. BFS is a superior choice because our goal appears at a shallow level in the search tree. BFS (breadth first search strategy) finds the shortest path and it is implemented using a doubly-ended queue. BFS is more effective at locating vertices near the specified source. As a result, BFS is taken into consideration.

On the one hand, we receive links, and on the other hand, each link is checked to see if it is active or not. All the active links will be collected then distributed to the machines suppose M1 and M2. These machines will start there crawling until or unless the documents get empty. Once all the documents are collected in the list then the content of each document is checked and compared with the seed

document to determine similarity. This will be accomplished with the help of a Natural Language Processing algorithm that performs large-scale analysis, i.e., processes massive volumes of data in seconds or minutes, whereas hand analysis would take days or weeks. NLP-powered tools can be taught to your company's language and criteria in a matter of steps, resulting in a more objective and accurate analysis.

The main advantage of RMI is that not only can RMI pass entire objects as inputs and return values, but it can also pass specified data types. Complex types, such as a standard Java hash table object, can now be supplied as a single argument. RMI uses built-in Java security measures to keep your system safe while users download implementations. To protect your systems and network from potentially hostile downloaded code, RMI leverages the security manager defined to defend systems from malicious applets. In extreme cases, a server may refuse to download any implementations.

4. RESULT

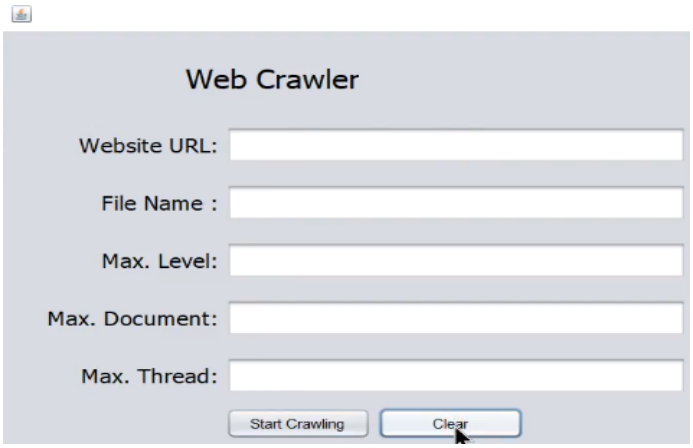

Fig. 5: Running the Web crawler for website to crawl

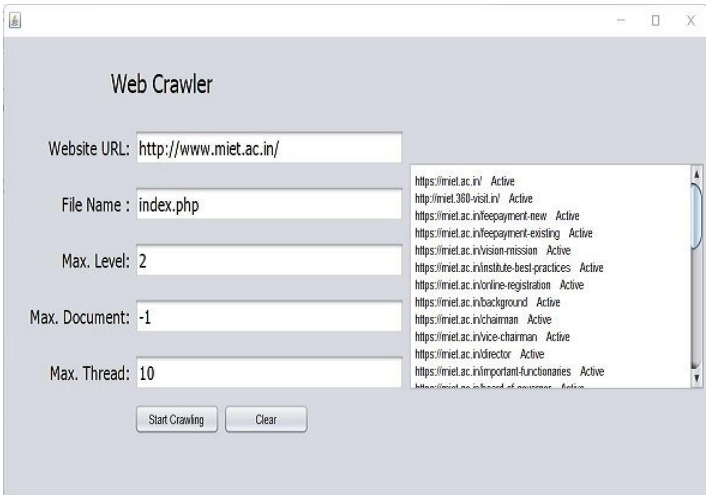

Fig. 6: Fetching active link

5. CONCLUSION

We presented a smart crawler in this research which deftly explores hidden web while minimizing resource & time dissipation. A perceptive spiderbot begins the crawl from center webpage using the seed URL, continues until the depth indicated (maximum levels specified) or until the last link is available, or until the quantity of documents is reached. The crawler can also distinguish between links and find whether they are active or not by invoking the site's webserver, and it includes a text-based site classifier that uses natural language. Technique for machine learning processing. The RMI optimizes the entire process by providing a distributed environment in which to handle the client's request. Natural Language will be used to show the client the most relevant and exact document.

REFERENCES

[1] Ajay Khare, Ashwini Dalvi, and Faruk Kazi, "Smart Crawler for Harvesting Deep web with Multi-Classification", July 1-3, 2020.

[2] Patrick Dave P. Woogue, Gabriel Andrew A. Pineda, and Christian V. Maderazo, "Automatic Web Page Categorization Using Machine Learning and Educational-Based Corpus", In proceedings International Journal of Computer Theory and Engineering,Vol.9, No 6, December 2017.

[3] BasselAlkhatip, Randa Basheer, "Crawling the Dark Web: A Conceptual Perspective Challenges and Implementation", In proceedings journal of Digital Information Management Vol.17, No. 2, April 2019.

[4] Tejaswani Patil, Santosh Chobe, "Web Crawler for searching Deep web sites", In Proceedings Third International Conference on Computing, Communication, Control And Automation, 2017.

[5] Linxuan Yu et al 2020 J. Phys.: Conf. Ser. 1449 012036,In proceedingsJournal of Physics:ConferenceSeries.

[6] G. Pavai1, T. V. Geetha. (2016) Improving the freshness of the search engines by a probabilistic approach based incremental crawler. Springer Science+Business Media New York.19:1013- 1028.

[7] Deka, GC, (2018) NoSQL Web Crawler Application. Advances in Computers.109:77-100.

[8] LEWIS E.ERSKINEDAVID R.N.CARTER-TODJOHN K.BURTON Education Technology Laboratory, College of Human Resources and Education, 220 War Memorial Hall, Blacksburg, VA, 24061, USA.