International Journal of Safety and Security Engineering
Vol. 14, No. 2, April, 2024, pp. 467-476
Journal homepage: http://iieta.org/journals/ijsse

Performance Analysis of Regex-Based Processing for Dark Web Targeted Crawling

Muhammad Faris Ruriawan, Yudha Purwanto, Putri R. Yunelfi, Agus S. Popalia

School of Electrical Engineering, Telkom University, Bandung 40257, Indonesia

Corresponding Author Email: omyudha@telkomuniversity.ac.id

https://doi.org/10.18280/ijsse.140214

Received: 21 September 2023
Revised: 12 March 2024
Accepted: 25 March 2024
Available online: 26 April 2024

Keywords:
dark web, The Onion Router (TOR), focus crawling, in-depth crawling, regex

ABSTRACT

Data crawling in the dark web holds a critical significance in bolstering security intelligence efforts. Previous research has successfully developed fast crawlers for specific purposes such as digital investigations, abusive content, automated captcha breaking, etc. However, this research mostly focuses on faster download time and has not paid attention to the importance of assessing crawl accuracy. Due to the fast-changing dark web shape and content, accurate and complete crawled data is a vital part of security intelligence. This research has successfully developed a targeted dark web crawler by combining the focus and in-depth crawling for The Onion Router (TOR) network. Regex Text, Regex Wildcard, and Regex Optional are used to automatically filter the content by a specific keyword. The effectiveness of the crawler was tested in the five real-world dark website environments. From the testing with a depth of 3, the application achieved more than 98% accuracy. The Regex Optional processing performance was faster than the Regex Text and Regex Wildcard by over a second, due to the swift crawling attempt. In terms of accuracy, the Regex Optional achieved 99.14% which is 4.83% higher than Regex Text. The best keyword processing method in targeted crawling is Regex Optional, with an accuracy rate of over 99%.

1. INTRODUCTION

Security intelligence plays a pivotal role in safeguarding organizations and individuals from a rapidly evolving landscape of threats and vulnerabilities. By systematically gathering, analysing, and interpreting data from various sources, security intelligence provides a comprehensive understanding of potential risks and malicious activities. This proactive approach empowers decision-makers to make informed choices, allocate resources effectively, and implement targeted measures to mitigate potential breaches, cyberattacks, and physical threats.

As the dark web serves as a breeding ground for illicit activities, systematic data crawling becomes an essential tool for gaining insights into these hidden realms. By employing specialized algorithms and technologies, security experts can navigate this obscure landscape, collecting valuable data on emerging cyber threats from the extracted dark web content. This proactive approach enables organizations to stay ahead of malicious resources [1], identify vulnerabilities [2], and understand the tactics, techniques, and procedures they employ [3]. To harvest the dark web data, security intelligence developed a dark web crawler.

Focus crawling is a type of crawling that looks for specific subjects using keywords that most accurately describe the target topic. Previous researchers have investigated the focus crawling for surface web [4], hidden web [5], and dark networks [6]. In focused crawling, a regular expression (regex or regexp) is often used to define a pattern for locating specific text within a web page [7], exploring text semantics [8], and even DNA sequence mining [9]. By the use of regular expression, the surface crawlers were successfully developed with high accuracy and classified into certain classes [10, 11].

Over the past five years, there has been significant progress in the field of dark web crawlers, resulting in the development of crawlers that possess both speed and flexibility. The first research concern on dark web is about the content collection for specific topics. Most research in this area proposed a crawler that digs down the dark web to obtain a comprehensive database for topic profiling and classification. This kind of research can be seen in research of Pannu et al. [12] for creating a database of suspicious websites, Alkhatib et al. [6] for market structure and product summary, Yang et al. [13] for hidden threat intelligence, Lee at el. [14] to uncover types of cyber-criminal activities occurring in South Korea, Shiaeles et al. [15] for monitoring and analysis of attack trends in the IoT ecosystem, Shinde et al. [16] to uncover child and women abuse materials, Alharbi et al. [17] to analyze the Tor dark web graph's internal structure and connectivity. The second concern is about the technical aspect of the crawler to gain flexibility and overcome the website crawling limitations, such as crawler traps and captcha. In research of Shinde et al. [16], the relevance of the page and hyperlinks is calculated using natural language processing and ANN-based classifiers to produce a high harvest rate. David et al. [18] proposed the use of information theory to compare widely used distances and consider heterogeneous data to avoid crawler traps. The challenge of link harvesting was researched in the study of

Dalvi et al. [19] by the use of a dedicated parser. The CAPTCHA breaking method was proposed in research of Zhang et al. [20] using the generative method. David et al. [18] uses information theory to compare widely used distances and considers heterogeneous data to produce more accurate distances for classification and data mining. Dalvi et al. [21] uses a pre-trained NLP model to determine the relevance score used in content classification and gain a high harvest rate.

From previous research, the regular expression is widely used in surface and dark web crawling methods. Based on our latest research of Yunelfi et al. [22] the accuracy of Regex classified-based focus crawling can reach 20% higher than traditional filter crawling. However, for the advance of our knowledge, there still no research has analyzed the use of different Regex methods for the dark web crawler. Technically, the different Regex methods allow crawlers to have different speed, flexibility, and accuracy. This is essential to consistently uphold the principles of data integrity in alignment with forensic science standards, ensuring the potential admissibility of the acquired data as evidence in legal proceedings. This significance becomes particularly pronounced in cases involving the forensic acquisition of dark web content.

This research developed a Regex-based dark web-targeted crawler to elevate the accuracy and completeness without degrading the processing time. The crawling procedure was carried out by developing a system that details the dark web's crawling procedure from input to output in the form of URLs for content results categorized by a particular topic. The targeted crawling is developed based on the combination of focus and in-depth crawling. Regex Text, Regex Wildcards, and Regex Optional were used to process keywords in the focused crawling process to improve system accuracy and performance. The in-depth method can enhance the crawling depth by digging the dark web even deeper. White box validation was utilized to verify functions in this targeted crawling system, and the findings show that every function operates as it should. Additionally, in this study, accuracy and performance testing of the system findings were conducted by comparing the three Regex scenarios. Given the known that dark web URLs (Unified Resource Locator) are more complex than regular web URLs [23], it is important to develop a fast and accurate method to crawl the dark web.

This research contribution is stated as follows. First, the application of Regex-based targeted crawling has been successfully developed with more than 98% accuracy. Second, it was discovered that the Regex Optional processing was superior to the Regex Text and Regex Wildcard in terms of accuracy and performance. The combination of in-depth algorithm and focus crawling in the TOR network has been proven to provide high-accuracy crawling results.

In this report, the related issues are elaborated. In Section 2, the advanced research of dark web crawling was summarized. The novelty of this research is discussed in Section 3 by the targeted-crawling system design and implementation setup. The output of the crawling system is discussed and analyzed in section 4, in terms of time processing and crawled data accuracy. Based on the analysis from Section 4, the research conclusion is stated in Section 5.

2. RELATED WORKS

The dark web itself is the most obscure area of the deep web that is purposefully concealed, virtually unreachable via standard Web browsers, and only accessible with the use of specialized protocols, configuration, or authorization with the use of the TOR network [24]. The state-of-the-art dark web crawling techniques involve a multi-faceted approach that combines advanced technologies and innovative methodologies to efficiently gather and analyse information from the hidden corners of the internet.

The process begins with the development of a sophisticated web crawler equipped with intelligent algorithms that can dynamically adapt to the ever-changing structure of the dark web. An automated computer program that browses webpages based on hyperlinks is called a web crawler, often referred to as a web spider or web robot. Web crawling, often known as spiders, is the technique through which crawlers collect data from the Web [4]. This crawler's implementation intends to manage and maintain the index of online pages and enable quick, accurate, and useful searches [25].

It is well known that there will be several challenges when crawling on the dark web, including the fact that it is hard to penetrate because it is at the base of the internet network [6]. The difficulty of obtaining a dark web address from the regular web is due to its random and elongated shape, dark network logins, and web cycles brief and could end in days [26]. A few research studies have thoroughly scrutinized web crawling on the dark web such as I2P (Invisible Internet Project), IPFS (Interplanetary File System), and TOR. The crawling technique employs a combination of traditional URL-based crawling and content-based analysis to traverse through onion sites, forums, and other hidden platforms. The research focus on the crawling method is summarized in Table 1.

Table 1. Research on the technical aspects of dark web crawler

Reference	Summary	Tools
[27]	A novel crawling system using a human-assisted accessibility approach and an incremental crawler with a recall-improvement mechanism. The system improves access to dark web forums and outperforms standard periodic- and incremental-update approaches.	Not specified.
[28]	A novel crawling system to collect dark web forum content. The system uses a human-assisted accessibility approach, URL ordering features, and an incremental crawler with a recall-improvement mechanism. Experiments show that the system significantly improves access to dark web forums, outperforming standard periodic- and incremental-update approaches.	Not specified.
[24]	A conceptual crawling system to uncover suspicious and malicious websites from The Onion Router (TOR) network. The system creates a database of suspicious and malicious websites by scraping and linking attributes, updating automatically, and archiving previous versions.	Not specified.
[29]	A python-based scrapy spider for structure and content mining to explore the dark web. The toolkit includes preparing content and reconstructing the graph structure.	Python.

Reference	Summary	Tools
[30]	The study develops methods for classifying complaints and an interactive crawler to identify discrimination. Conservative exit policies are ineffective, but most Tor attacks generate high traffic volume, suggesting potential for detection and prevention without violating users' privacy.	Not specified.
[12]	A conceptual system for automatically creating a database of suspicious websites on the TOR network. The system would find these websites by looking for links within TOR webpages. The database would be special because it would keep track of old versions of itself, so law enforcement could search both current and past versions to find malicious sites.	Not specified.
[15]	Designs and implements Python scrapers for Crawling and scraping dark crypto markets marketplaces to investigate the possibility of extracting useful results regarding attack trends and predicting them in advance by crawling the Deep/Dark and Surface web. The result shows that proactive monitoring and analysis of attack trends can contribute to the protection of the IoT ecosystem.	Python.
[6]	A crawler that is able to simulate a user login to a dark market, crawling the whole website and fetching the required data from its pages. The market structure was uncovered and analyze the product summary.	
[13]	A framework for hidden threat intelligence by briefly summarize some of the most frequently occurring words on various dark websites. As the growing data set continues, the knowledge of the dark web will uncover.	Python.
[14]	A Selenium-based crawler was used to harvest data specific to South Korean dark websites. The focus was on investigating cyber-criminal activities by performing an in-depth profiling of the top 3 Korean dark web sites. The researchers were able to gain insights into the types of cyber-criminal activities occurring in the region.	Not specified.
[16]	This paper proposes a python-based focused crawling framework to uncover child and women abuse material in the Surface and Dark Net. The relevance of the page and hyperlinks is calculated using natural language processing and ANN-based classifiers from anchor text and local context of the hyperlinks. The result shows that the hyperlink selection method is an effective approach for web data mining and classification tasks.	Python.
[17, 29]	Analyze the Tor dark web graph's internal structure and connectivity, examining the bow-tie structure found in the World Wide Web. The web graph is generated from Python crawler data, with nodes representing individual Tor hidden services and edges representing hyperlinks. The graph is sparse, with few connected pairs, and can be decomposed into a bow-tie structure with small component sizes. The Tor network exhibits characteristics of small-world and scale-free networks.	Python.
[18]	This system uses information theory to compare widely used distances and considers heterogeneous data, resulting in more accurate distances than other tested methods.	Not specified.
[19]	This paper proposes a dedicated parser methodology for extracting URLs from the dark web, outperforming regular expressions and built-in parsers. It addresses challenges in link harvesting on the dark web and discusses factors that make it more efficient. However, this research only provides a URL validation for link harvesting.	Not specified.
[20]	A novel framework for automated CAPTCHA breaking, utilizing a generative method to recognize dark web text-based CAPTCHAs with noisy backgrounds and variable character length. The framework achieved over 94.4% success rate.	Not specified.
[21]	A SpyDark collects information from both the surface and the dark web. The crawler extracts text data, images, and hyperlinks, and uses a pre-trained NLP model to determine relevance. It can draw link tree of the content and mark the crawled page to relevan and irrelevant to specific criteria.	Not specified.
[31]	A Tor-based web crawling model was developed, which successfully scraped web content from both clear and dark web pages and dark marketplaces on the Tor network. This paper provides novel knowledge about ACN-based web crawlers and presents a model for crawling, scraping, and similarity analysis of clear and dark websites.	Python.

3. PROPOSED METHOD

Focus web crawlers are web crawlers that have been designed specifically for deep browsing [32]. By looking at similar terms, focus crawling searches online pages in a targeted manner [6]. There are three learning phases for the web crawling process of web exploration: content-based, link-based, and brother-based [33].

Focus crawlers discover web resources by navigating through link structures and selecting hyperlinks based on relevance. Focus crawling operates by precisely classifying information based on content relevance and preserving the maximum functionality of collected URLs and searchable forms [34]. This method of operation allows the use of focus crawling for platforms such as the World Wide Web [35], Twitter [36], and deep web data [34, 37]. This has led to the knowledge that focus crawling may be used by any search engine [38].

This research investigated sites on the dark web using a combination of in-depth algorithms and focus crawling techniques. A VPN (Virtual Private Network) and TOR browser are used as supporting software to directly check the website that is being browsed in addition to the targeted crawling technique. The focus crawling was combined with in-depth crawling to produce targeted crawling. The following block diagram shows the steps of the focus in-depth crawling method procedure. The basic model of our research is depicted as a level 0 data flow diagram in Figure 1. The diagram context shows a targeted crawler system that has three entities, namely the user, the TOR network, and the dark web site to be accessed.

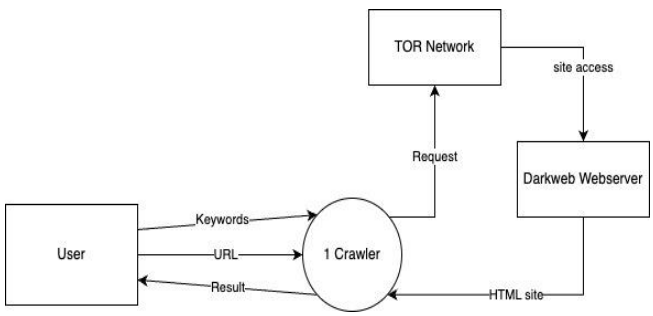

Figure 1. The context diagram of the proposed dark web crawling system

The targeted crawler has two main processes according to focus and in-depth crawling method. The focus crawling handled the keyword processing and content filtering. The in-

depth crawling was used to handle the URL processing in the focus crawling.

The system process begins with the user inputting keywords and the parent URL that the system will explore.

After processing the input, the system accesses the dark web with the assistance of the TOR network to reach the provided URL and further explore additional URLs based on the entered keywords. Following the exploration process, the discovered URLs, based on the input URL and keywords, are stored in a dataset under the name of the dark web input. The process is depicted in Figure 2.

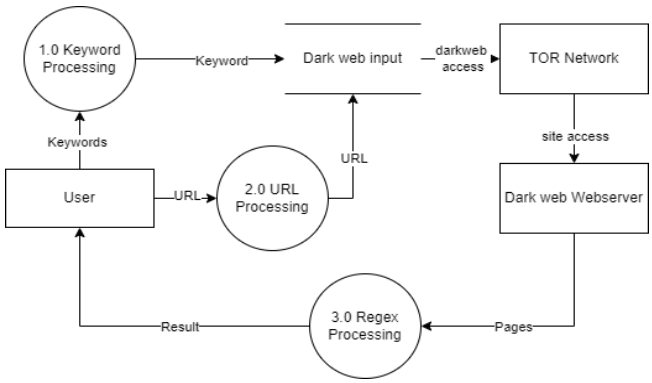

Figure 2. Data flow diagram level 1

The URL processing is then divided into three processes, namely the input verification process, the excluding process, and the fixing process. At DFD (Data Flow Diagram) level 2, there are already 2 data stores, namely the data store for raw URLs and the result URL data store. The raw URL data store contains a list of URL lists placed in the "a href" and "area href" tags. As the URLs contained in this tag are not completely perfect, it then enters the URL fixing process and then stored in a database.

The keyword processing and content filtering in the targeted crawling system is carried out using three processes: Regex Text, Regex Wildcard, and Regex Optional. Regex Text involves exploration based on the text of the keywords entered by the user such in RegexText pseudocode. Regex Wildcard explores keywords with the limitation symbols of a period (.) and/or a plus sign (+) to represent multiple letters in the keyword such as Regex Wildcard pseudocode. Meanwhile, the Regex Optional processing explores keywords with a question mark (?) symbol to indicate that the letters before the symbol are optional such as in Regex Optional pseudocode. The crawled data was then stored in a result database. Figure 3 shows the DFD level 2 of the URL processing system.

```
Program RegexText (keyword, inputkw, url)
1.  FOR url contains keyword
2.      F(i,j) → string inputkw[0…i] matches with string
            keyword[0…j]
3.      IF (inputkw[i]==keyword[j])
4.      PRINT (URL. Keyword)
5.      return TRUE
6.    ELSE
7.      Return FALSE
8.  End FOR
```

```
Program RegexWildcard (keyword, inputkw, url)
1.  FOR url contains keyword
2.     F(i,j) → string inputkw[0…i] matches with string
            keyword[0…j]
3.    IF (i<0 && j<0)
4.      return TRUE
5.    END IF
6.    IF (i<0 && j>=0)
7.      return FALSE
8.    END IF
9.    IF (i>=0 && j<0)
10.     return isAllStarts(keyword,j)
11.   End if
12. END FOR
```

```
Program RegexOptional (keyword, inputkw, url)
1.  FOR url contains keyword
2.      F(i,j) → string inputkw[0…i] matches with string
        keyword[0…j]
3.      IF (inputkw[i]&&keyword[j] not in konten)
4.        return FALSE
5.      ELSE
6.        PRINT (URL. Keyword)
7.        Return TRUE
8.      END IF
9.  END FOR
```

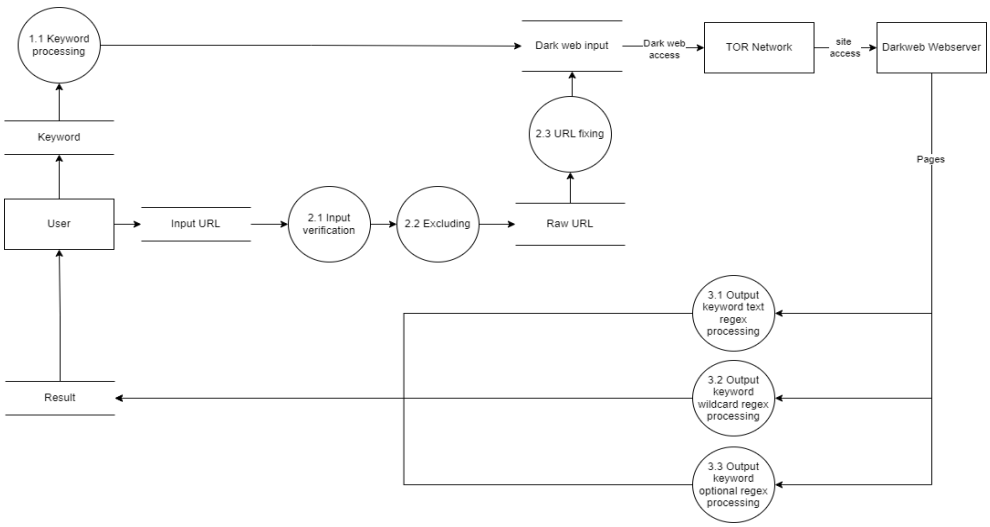

Figure 3. Data flow diagram level 2

The system was tested across various scenarios, including functional testing, accuracy testing, and performance testing. Functional testing involves assessing the system's functionality, with a focus on white-box validation to gauge its compliance with system requirements. During white-box validation, each process within the system was scrutinized individually to determine if it functioned correctly and as intended. This validation process was conducted for all code files, with the expectation of achieving a 100% success rate, thereby confirming the program's full functionality.

The testing and analysis were conducted in dark web TOR network. Due to the large scale of the dark web network, this research only crawls the specific URL which contains the specific keywords. First, the TOR network is activated, and the crawling parameter is set. The TOR connection was done on TOR network through TOR software which integrated with Virtual Private Network. In the experiment, the crawling parameters are keywords, in-depth value, pause time, and URLs. And then, the crawler crawls the dark web and keywords processing.

4. RESULT AND ANALYSIS

4.1 Simulation setup

The testing and analysis were conducted in dark web TOR network. Due to the large scale of the dark web network, this research only crawls the specific URL which contains the specific keywords. First, the TOR network is activated, and the crawling parameter is set. The TOR connection was done on TOR network through TOR software which integrated with Virtual Private Network. In the experiment, the crawling parameters are keywords, in-depth value, pause time, and URLs. And then, the crawler crawls the dark web and keywords processing. The flowchart of the crawling process is depicted in Figure 4.

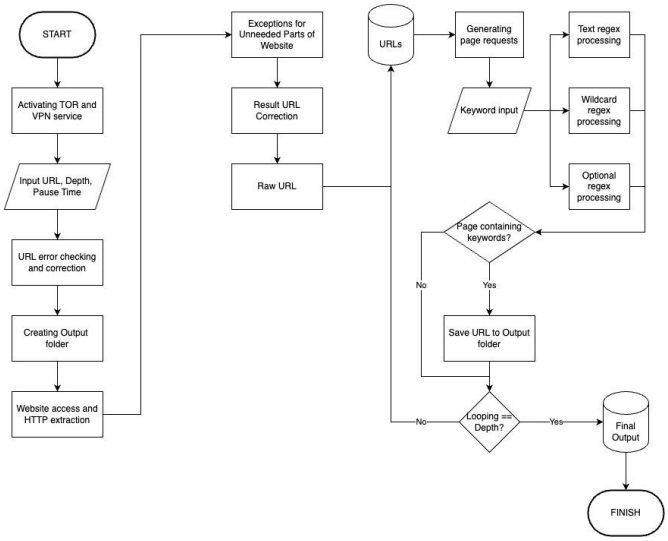

Figure 4. Crawling process

In this experiment, the input keywords are “cocain” and “hack”. The crawling parameters were set with the in-depth value set to three, and the pause time is set to 0. This limitation is in place because extending the depth further would necessitate introducing “pause intervals”. Thus, it would render performance measurements unreliable due to the impact of these pauses. Additionally, setting a limit on the crawling depth serves the purpose of simplifying accuracy assessment, as it was manually checked. It is important to note that no commercial dark web crawler is available for benchmarking accuracy, which necessitates manual verification of all website URLs during each trial. The wildcard and the optional character in the Regex Wildcard and Regex Optional are set to 1.

The rule-based focus crawling from [22] and targeted crawling were occupied to crawl the URLs to dig and scrap even deeper. Each URL was crawled ten times to statistically analyze the accuracy, completeness, and time processing. The list of the URLs is listed in Table 2.

Table 2. List of the tested URLs

No.	URL
1	http://rfyb5tlhiqtiavwhikdlvb3fumxgqwtg2naanxtiqibidqlox5v*****.onion/
2	http://xf2gry25d3tyxkiu2xlvczd3q7jl6yyhtpodevjugnxia2u665a*****.onion/
3	http://prjd5pmbug2cnfs67s3y65ods27vamswdaw2lnwf45ys3pjl55h*****.onion/
4	http://guzjgkpodzshso2nohspxijzk5jgoaxzqioa7vzy6qdmwpz3hq4*****.onion/
5	http://kq4okz5kf4xosbsnvdr45uukjhbm4oameb6k6agjjsydycvflce*****.onion/

The highest accuracy score was then calculated to select the best Regex-based crawling method. Given $X = \{x_1, x_2, x_3, \dots, x_i\}$; which x_i is the average harvest rate of Regex (i), and $Y = \{y_1, y_2, y_3, \dots, y_i\}$; which y_i is the number of crawled pages of Regex (i); i=3 which is Regex Text, Regex Wildcard, and Regex Optional, the accuracy score is calculated such in Eq. (1).

$$AS_i = x_i * \frac{y_i}{\max(Y)} \quad (1)$$

4.2 White box validation

The results of testing with this white box come in the form of software errors. All processes on each code file that are validated are shown in Table 3. All the functionalities employed in the system that was centered on crawling were found to be operating as expected. The system works properly according to the system requirements, which example of the crawled data can be seen in Figure 5 and Figure 6.

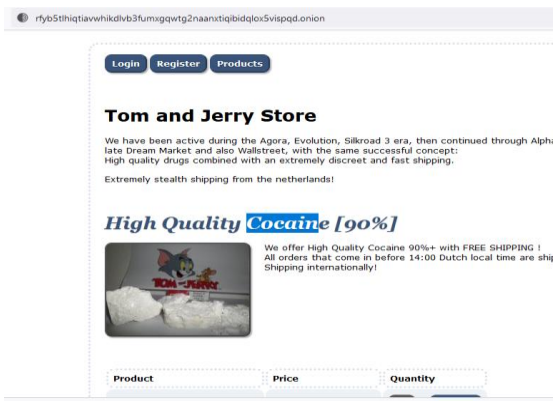

Figure 5. Example of crawled page data with keyword "cocaine"

Table 3. White box validation

Function	Expected Results	Results
Input	The URL and keyword input process is carried out to be crawled.	Users can input URL and keyword restrictions to complete the operation.
Checking TOR service	The checker module provides notification regarding the availability of TOR service.	The checker module provides notification regarding the availability of TOR service.
URL validation	The checker module can detect and repair any missing components of the URL.	The checker module can detect and repair any missing components of the URL.
Crawling when examining the Keyword Regex text	Retrieve URL results from dark web crawling and use a regular expression approach to find the entered keywords.	Retrieve URL results from dark web crawling and use a regular expression approach to find the entered keywords.
Checking Keyword Regex Wildcard while crawling	Using the URL of the returned results from dark web crawling to check the entered keywords using the Regular expression wildcard technique.	Applying the Regular Expression Wildcard technique to check the entered keywords, and use crawling to obtain URL results from the dark web.
Checking Keyword Regex Optional when Crawling	Retrieving the URL of the results from dark web crawling and using the optional Regular expression approach to check the entered keywords.	Can obtain URLs through dark web crawling and use the optional Regular expression approach to verify the entered keywords.
Create output file	Creating a text file with extension .txt that has a list of output URLs with filenames that correspond to the input keywords.	Can produce a text file with the extension .txt that contains a list of output URLs with filenames based on the keywords.
Crawling	Alpha crawl, checker, and core crawl can run the crawling process.	The system can crawl the dark web.
Crawl time notification	Shows information about the crawl time, which signifies that the URL procedure is finished.	Can show information on the length of the crawling process, indicating when the process for the URL has finished

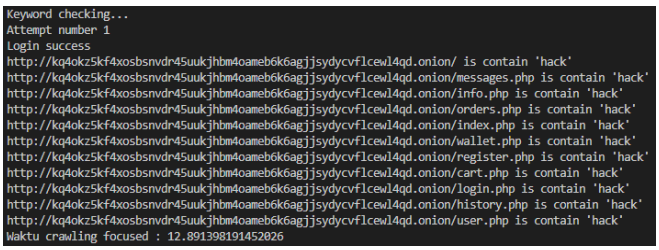

Figure 6. Example of crawling process using Regex Wildcard on keyword "hack"

4.3 Performance testing

The performance testing aimed to assess the application's overall performance. This involved collecting data on the average crawling times, enabling an analysis of crawling performance. Testing was done ten times for each URL with three different keyword processing methods, including Regex Text from research [22], Regex Wildcard, and Regex Optional. This system uses focus crawling on a parent URL to test the system's crawling performance for each keyword search procedure. The table displays the parent URL that crawled during the targeted crawling system test.

The dataset is the crawled dark web pages from the URLs in Table 2. The crawled dataset consists of two datasets which are dark web content and filtered content. The first is the dataset from the in-depth crawling, consisting of all pages on the web for related input URLs. The second is the filtered pages that match the targeted crawling keyword. Each epoch of the testing was then parameterized by crawling time, number of filtered pages, number of pages, attempt number, accuracy, and keywords. Each parameter was recorded for Regex Text, Regex Wildcard, and Regex Optional crawling in an epoch. The example of the crawling output is shown in Figure 7. The dataset structure can be seen in Figure 8. The log parameter result is a table of 300 data with seven features.

Based on the average per-page crawling time, it was determined that the crawler demonstrated a rapid crawling performance, with a testing speed of approximately 0.2 to 1.1 seconds per URL. It shows that the crawler can work properly with a fast per-page crawling time. The low per-page crawling time is not only influenced by the web server but also by the type of content. For example, the URL 2 is a static web that can be crawled faster. The average per page crawling in each URL is depicted in Figure 9.

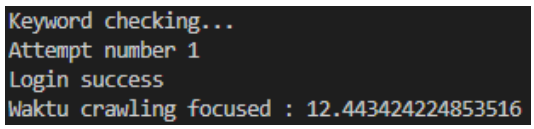

Figure 7. An example of the result of keyword checking and crawling

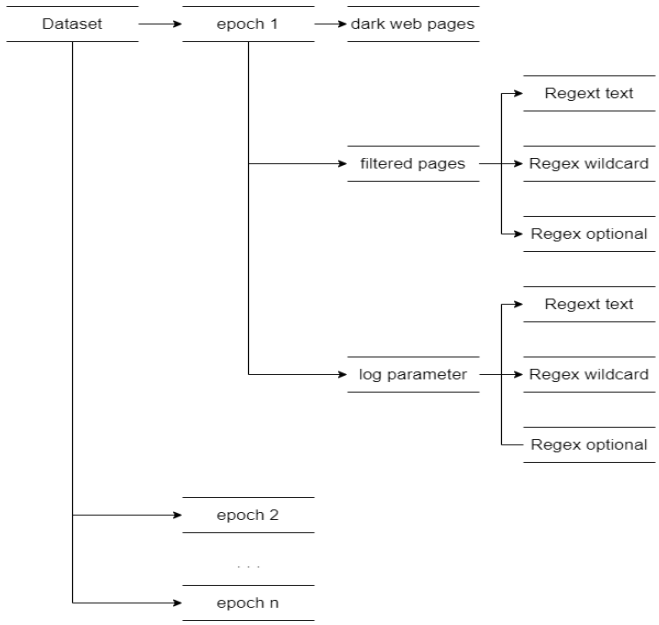

Figure 8. Dataset structure

From the ten times Regex Text crawling process from research [22] in five URLs, the average crawling time was 16.7 seconds. In comparison, the Regex Wildcard crawling process averaged 15.8 seconds, which was slower than the

Regex Optional process which averaged 14.4 seconds. The Regex Optional has the fastest crawling time compared to the others. The higher number of characters in keyword result in a higher crawling time. The statistical average crawling time result comparison can be seen in Figure 10.

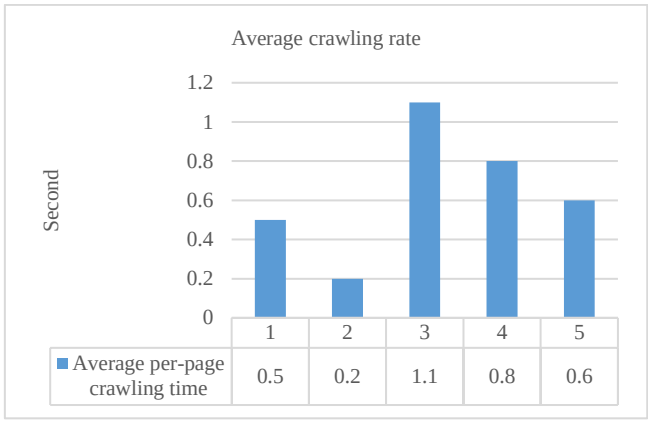

Figure 9. Average crawling rate

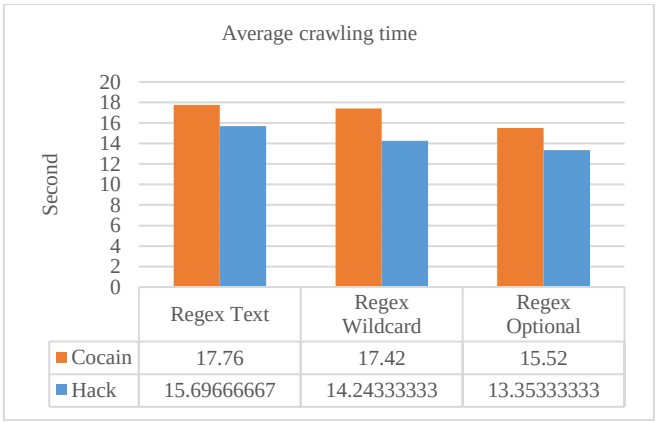

Figure 10. Average crawling time

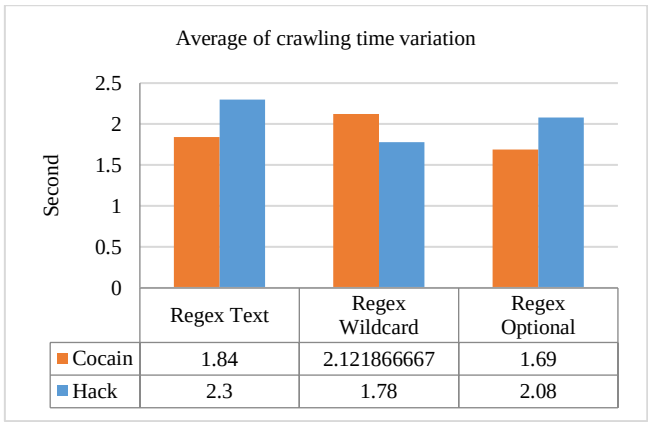

Figure 11. Average crawling time variation

The use of Regex-based crawling has shown high-speed average crawling times. Regex wildcard and Regex Optional provide a more stable crawling process. Based on testing the targeted crawling system on the keywords "cocain," and "hack", it was discovered that the crawling process was carried out with different variations in time in each experiment. Each URL variance value demonstrates that the crawling for each attempt may be affected by the network and web server performance and condition. Regex Optional achieves more stable average variation with 1.88 seconds which is better than 1.95 seconds in Regex Wildcard and 2.07 seconds in Regex Text. The results can be seen in Figure 11.

4.4 Completeness testing

The completeness is measured by the number of crawled and scraped data from the dark web. With the focused crawling, the data only contained the URLs and pages that contained the keywords. To measure the completeness, the number of crawled data was compared to the Regex Text in research [22], and analysed manually. For the depth = 3, the result shows that the Regex Optional has crawled and scraped a higher number of URLs than the Regex Text and Wildcard. The number of crawled pages can be seen in Table 4.

Table 4. The number of collected crawled pages

URL Number	Regex Text	Regex Wildcard	Regex Optional
1	2	2	3
2	12	10	12
3	7	6	7
4	11	11	14
5	11	10	11

4.5 Accuracy testing

Accuracy testing was employed to evaluate the URL content obtained from crawling in relation to the input keywords. This analysis also considered the impact of Regex Text from research [13], Regex Wildcard, and Regex Option on the collected data. The accuracy analysis was performed on the data obtained from the same scenario as the performance testing, where each URL was tested ten times using three distinct keyword processing methods. Each URL utilized in the earlier performance test was tested. By comparing the URL results returned by the system with manual search results on the dark web, it is possible to determine the correctness of the system. Two separate input keywords are used in this test to determine correctness. For the outcomes of employing two keywords to measure accuracy, Figure 12 shows the accuracy results.

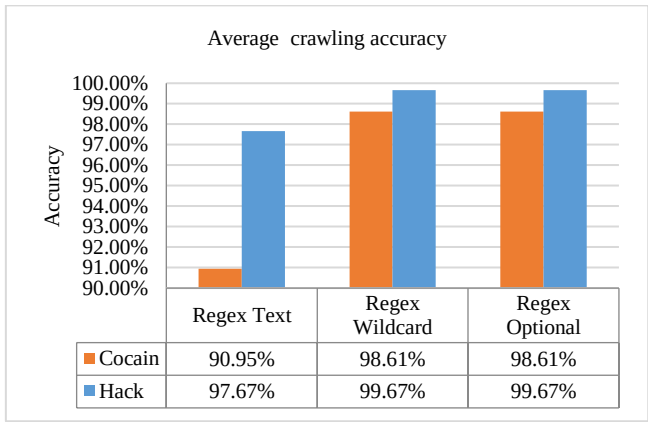

Figure 12. Average crawling accuracy

By comparing system results with manual search results on the dark web, the accuracy of the targeted crawling system is determined. The system divides its calculation of accuracy into three distinct keyword processing techniques. These techniques are employed with five different URLs and the keywords "cocain" and "hack" in the system.

The accuracy of each URL varies from the average system accuracy results according to the comparison of the system and manual results. In the targeted crawling process utilizing Regex Text such as in the study of Yunelfi et al. [22], the average accuracy is 94.31%. The Regex Wildcard and Regex Optional both achieved an average accuracy of 99.14% for the same two terms. However, as the number of crawled pages in Regex Optional is higher than in Regex Wildcard, Regex Optional achieved the best performance. The accuracy rate of the Regex Optional result achieved the highest score of 99,14% compared to Regex Text and Regex Wildcard which achieved 86% and 82%.

4.6 Discussion

According to the result of performance testing, the difference in crawling time is not influenced by the use of Regex Text, Regex Wildcard, and Regex Optional. From the Anova testing on the conjugated crawling time for each method, the result shows there are no influence of the Regex method on the crawling time. Compared to the significance value=0,05, the p-value=0,75 shows that the variance of the Regex-based crawling time is homogenous. However, the F-score shows that there is no difference in average crawling time using different Regex. It can be seen from the F-core=0.28 is lower than the F-critical which is 3,2. Thus, it shows that the use of Regex does not affect the crawling time. The Anova testing result can be seen in Table 5.

Table 5. Anova test result

Source of Variation	SS	df	MS	F	P-Value	F crit
Between groups	17.1	2	8.5	0.28	0.75	3.2
Within groups	1251.8	42	29.8			

From further analysis, the difference in crawling time is primarily due to its swiftness of login attempt, with an average of 1,15 on Regex Text, compared to 1 on Regex Wildcard and 1 attempt on Regex Optional. The faster crawling time of the Regex Optional may be caused by the number of checking characters, which is a stable checking character in the keyword. It results in the complexity of O(m) which is the same as the Regex Text with m being the number of characters in the keyword string. From the pseudocode of Regex Wildcard, the complexity is O(m*n) with m being the number of characters in the input string and n being the number of characters in the wildcard pattern.

It is shown that the wildcard will search for all possible characters to replace the wildcard character in the keyword which makes it slower along with the rising number and complexity of character and wildcard. However, in our scenario, as the optional and wildcard characters are set to 1, the difference is not visible. The analysis of the influence of the depth and the number of wildcard characters will be reported in a separate report.

This study has limitations. The Regex processing is determined as greedy and lazy [39]. These behaviours come close to the previous finding that Regex has a slow processing time. In HTML files, the Regex needs to keep track of all HTML tags which is arbitrarily large. That is why in the next research, the HTML parser may be better used to crawl the dark web. For the alternative of the Regex processing in web content, the use of automata and formal language may be implemented in the future crawling method.

5. CONCLUSION

This research has successfully developed an application that utilizes a targeted crawler. Following white-box validation and testing, the crawler aligns with the system requirements. During performance testing, Regex Optional outperformed both Regex Wildcard and Regex Text, boasting an average crawling time that was over a second faster than Regex Wildcard and 2 seconds than Regex Text. Regex Optional proves superior to Regex Text and Regex Wildcard due to the swift access attempts on the initial access, thus reducing crawl duration.

In terms of accuracy, the accuracy of Regex Wildcard matches that of Regex Optional, standing at 99.14%, which is 4.83% higher than the accuracy achieved with Regex Text. This outcome arose from the observation that registering on Regex Wildcard and Regex Optional is a swifter process compared to Regex Text. Moreover, when using Wildcard Regex and Optional Regex, there have been no instances of access failures during dark web crawling, influencing the quantity of obtained URL results. The best keyword processing method in targeted crawling is Regex Optional, with an accuracy rate of over 99%.

However, the Regex-based processing may suffer a slower process with the rising number and complexity of characters and wildcards. The objective of future research is to improve this drawback by the use of better data processing. The use of an HTML parser for HTML files may increase the crawling time. The use of automata and formal language may improve the accuracy to gain better content analysis of the dark web.

ACKNOWLEDGMENT

This research is funded by Telkom University under the grant no. KWR4.073/PNLT3/PPM-LIT/2022.

REFERENCES

[1] Prayogi, A., Aji, R.F. (2023). Utilization of mobile network infrastructure to prevent financial mobile application account takeover. Jurnal Rekayasa Sistem dan Teknologi Informasi, 7(4): 797-808. https://doi.org/10.29207/resti.v7i4.5025

[2] Hafiz, N., Briliyant, O.C., Priambodo, D.F., Hasbi, M., Siswanti, S. (2023). Remote penetration testing with telegram bot. Jurnal Rekayasa Sistem dan Teknologi Informasi, 7(3): 705-714. https://doi.org/10.29207/resti.v7i3.4870

[3] Sianturi, T., Ramli, K. (2022). A security framework for secure host-to-host environments. Jurnal Rekayasa SIstem dan Teknologi Informasi, 6(3): 380-386. https://doi.org/10.29207/resti.v6i3.4018

[4] Sekhat, S.R.M., Siddesh, G.M., Manvi, S.S., Srinivasa, K.G. (2019). Optimized focused web crawler with natural language. Cybernetics and Information Technologies, 19(2): 146-158. https://doi.org/10.2478/cait-2019-0021

[5] Liakos, P., Ntoulas, A., Labrinidis, A., Delis, A. (2016). Focused crawling for the hidden web. World Wide Web, 19: 605-631. 19: 605-631. https://doi.org/10.1007/s11280-015-0349-x

[6] Alkhatib, B., Basheer, R. (2019). Crawling the dark web:

A conceptual perspective, challenges and implementation. Journal of Digital Information Management, 17(2): 51-60. https://doi.org/10.6025/jdim/2019/17/2/51-60

[7] Jiang, J.T., Song, X.Y., Yu, N.H., Lin, C.Y. (2013). FoCUS: Learning to crawl web forums. In Proceedings of the 21st International Conference on World Wide Web, pp. 33-42. https://doi.org/10.1145/2187980.2187985

[8] Idrees, A.M., Shaaban, E.M. (2020). Building a knowledge base shell based on exploring text semantic relations from arabic text. International Journal of Intelligent Engineering and Systems, 13(1): 324-333. https://doi.org/10.22266/ijies2020.0229.30

[9] Lakshmanna, K., Khare, N. (2016). FDSMO: Frequent DNA sequence mining using FBSB and optimization. International Journal of Intelligent Engineering and Systems, 9(4): 157-166. https://doi.org/10.22266/ijies2016.1231.17

[10] Sekhar, M., Siddesh, G.M., Manvi, S.S., Srinivasa, K.G. (2019). Optimized focused web crawler with natural language processng based relevance measure in bioinformatics web sources. Cybernetics and Information Technologies, 19(2): 146-158. https://doi.org/10.2478/cait-2019-0021

[11] Agun, H.V. (2023). Webcollectives: A light regular expression based web content extractor in Java. Social Science Research Network. https://doi.org/10.2139/ssrn.4480037

[12] Pannu, M., Kay, I., Harris, D. (2018). Using dark web crawler to uncover suspicious and malicious websites. In Ahram, T., Nicholson, D. (eds.) Advances in Human Factors in Cybersecurity, 108-115. https://doi.org/10.1007/978-3-319-94782-2_11

[13] Yang, Y., Zhu, G.C., Yang, L., Yu, H.H. (2020). Crawling and analysis of dark network data. In International Conference on Computing and Data Engineering, pp. 116-120. https://doi.org/10.1145/3379247.3379272

[14] Lee, J., Hong, Y., Kwon, H., Hur, J. (2020). Shedding light on dark Korea: An in-depth analysis and profiling of the dark web in Korea. International Workshop on Information Security Applications, Lecture Notes in Computer Science(), vol 11897. Springer, Cham. https://doi.org/10.1007/978-3-030-39303-8_27

[15] Shiaeles, S., Kolokotronis, N., Bellini, E. (2019). IoT vulnerability data crawling and analysis. In IEEE World Congress on Services (SERVICES), Milan, Italy. https://doi.org/10.1109/SERVICES.2019.00028

[16] Shinde, V., Dhotre, S., Gavde, V., Dalvi, A., Kazi F., Bhirud, S.G. (2021). CrawlBot: A domain-specific pseudonymous crawler. In ICCEDE 2020. Communications in Computer and Information Science, vol 1436. Springer, Cham. https://doi.org/10.1007/978-3-030-84842-2_7

[17] Alharbi, A., Faizan, M., Alosaimi, W., Alyamni, H., Agrawal, A., Kumar, R., Khan, R.A. (2021). Exploring the topological properties of the tor dark web. IEEE Access, 9: 21746-21758. https://doi.org/10.1109/ACCESS.2021.3055532

[18] David, B., Delong, M., Filiol, E. (2021). Detection of crawler traps: Formalization and implementation—defeating protection on internet and on the TOR network. Journal of Computer Virology and Hacking Techniques, 17: 185-198. https://doi.org/10.1007/s11416-021-00380-4

[19] Dalvi, A., Siddavatam, I., Thakkar, V., Jain, A., Kazi, F., Bhirud, S. (2021). Link harvesting on the dark web. In IEEE Bombay Section Signature Conference (IBSSC), Gwalior, India. https://doi.org/10.1109/IBSSC53889.2021.9673428

[20] Zhang, N., Ebrahimi, M., Li, W.F., Chen, H. (2021). Counteracting dark web text-based CAPTCHA with generative adversarial learning for proactive cyber threat intelligence. ACM Transactions on Management Information Systems, 13(2): 1-21. https://doi.org/10.1145/3505226

[21] Dalvi, A., Paranjpe, S., Amale, R., Kurumkar, S., Kazi, F., Bhirud, S.G. (2021). SpyDark: Surface and dark. In 2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC), Jalandhar, India. https://doi.org/10.1109/ICSCCC51823.2021.9478098

[22] Yunelfi, P.R., Popalia, A.S., Fahrani, F., Purwanto, Y., Ruriawan, M.F. (2022). DarkWeb crawling using focused and classified algorithm. Journal of Computer Engineering, Progress, Application & Technology, 1(2): 1-6. https://doi.org/10.25124/cepat.v1i02.4879

[23] Kumar, M., Bindal, A., Gautam, R., Bhatia, R. (2018). Keyword query based focused web crawler. Procedia Computer Science, 125: 584-590. https://doi.org/10.1016/j.procs.2017.12.075

[24] Iliou, C., Kalpakis, G., Tsikrika, T., Vrochidis, S., Kompatsiaris, I. (2017). Hybrid focused crawling on the Surface and the dark web. Eurasip Journal on Information Security, 11. https://doi.org/10.1186/s13635-017-0064-5

[25] Shrivastava, V. (2018). A methodical study of web crawler. Journal of Engineering Research and Application, 8(11): 1-8. https://doi.org/10.9790/9622-0811010108

[26] Monterrubio, S.M.M., Naranjo, J.E.A., Lopez, L.I.B., Caraguay, Á.L.V. (2021). Black widow crawler for TOR network to search for criminal patterns. In 2021 Second International Conference on Information Systems and Software Technologies (ICI2ST), Quito, Ecuador. https://doi.org/10.1109/ICI2ST51859.2021.00023

[27] Frana, P.L. (2004). Before the web there was gopher. IEEE Annals of the History of Computing, 26(1): 20-41. https://doi.org/10.1109/MAHC.2004.1278848

[28] Fu, T.J., Abbasi, A., Chen, H. (2010). A focused crawler for dark web forums. Journal of the American Society for Information Science and Technology, 61(6): 1213-1231. https://doi.org/10.1002/asi.21323

[29] Celestini, A., Guarino, S. (2017). Design, implementation and test of a flexible tor-oriented web mining toolkit. In International Conference on Web Intelligence, Mining, and Semantics, pp. 1-10. https://doi.org/10.1145/3102254.3102266

[30] Singh, R., Nithyanand, R., Afroz, S., Pearce, P., Tschantz, M.C., Gill, P., Paxson, V. (2017). Characterizing the Nature and Dynamics. In Proceedings of the 26th USENIX Security Symposium, BC, Canada, pp. 325-341.

[31] Bergman, J., Popov, O. (2023). Exploring dark web crawlers: A systematic literature review of dark web crawlers and their implementation. IEEE Access, 11: 35914-35933. https://doi.org/10.1109/ACCESS.2023.3255165

[32] Alfarisy, G.A.F., Bachtiar, F.A. (2017). Focused web

crawler for Indonesian recipes. In 2017 International Conference on Sustainable Information Engineering and Technology (SIET), Malang, Indonesia. https://doi.org/10.1109/SIET.2017.8304134

[33] Lu, H., Zhan, D., Zhou, L., He, D. (2016). An improved focused crawler: Using web page classification and link priority evaluation. Mathematical Problems in Engineering, 2016: 6406901. https://doi.org/10.1155/2016/6406901

[34] Mishra, P., Khurana, A. (2018). Accuracy crawler: An accurate crawler for deep web data extraction. In 2018 International Conference on Control, Power, Communication and Computing Technologies (ICCPCCT), Kannur, India. https://doi.org/10.1109/ICCPCCT.2018.8574286

[35] Fang, T., Han, T., Zhang, C., Yao, Y.J. (2020). Research and construction of the online pesticide information center and discovery platform based on web crawler. Procedia Computer Science, 166: 9-14. https://doi.org/10.1016/j.procs.2020.02.004

[36] Khazaie, A., Seghouani, N.B., Bugiotti, F. (2021). Smart crawling: A new approach toward focus crawling from Twitter. Information Retrieval. https://doi.org/10.48550/arXiv.2110.06022

[37] Patil, T.A., Chobe, S. (2017). Web crawler for searching deep web sites. In 2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA), Pune, India. https://doi.org/10.1109/ICCUBEA.2017.8463648

[38] Ghupta, A., Singh, K.B., Singh, R.K. (2018). Web crawling techniques and its implications. Globus An International Journal of Management & IT, 9(2): 7.

[39] Cox, R. (2019). Regular expression matching can be simple and fast. https://swtch.com/~rsc/regexp/regexp1.html, accessed on Feb. 12, 2024.