DESIGN AND IMPLEMENTATION OF A WEB-BASED CRAWLER USING BEE-SWARM ALGORITHM

[1]Mgbeafulike, I. J. ike.mgbeafuike@gmail.com

[2]Ojiako, F.N. lightmanfranklin1989@gmail.com

Phone: 08035971621

[1, 2] Chukwuemeka Odumegwu Ojukwu University, Uli Campus, Anambra State, Nigeria.

ABSTRACT

A web crawler (also known as a web spider or web robot) is a program or automated script which browses the worldwide web in a methodical, automated manner. The difficulties in implementing web crawlers state that bandwidth used in crawling is neither free nor infinite. So, it is essential to crawl the web not only in a scalable but in an efficient way. There are two major characteristics that generate issues in which web crawling is difficult: (a) large volume of web pages (b) rate of changes on the web. The purpose of this research is to design and implement a web-based crawler using bee-swarm algorithm that will be able to search the web based on specific topic in order to address the above stated problems. The methodology adopted was the Feature-Driven Development (FDD) methodology. FDD is a developer centric process which consists of modeling a feature into an overall shape and delivering the model as a build in short two weeks iterations. It focuses on specific units of work that go through a stringent process which proceeds through domain walkthrough, design, design inspection, code, code inspection and the promotion to build. This research findings show that the use of bee-swarm algorithm in implementing a web crawler shows a significant improvement in user's search experience as it uses collective intelligence in returning only relevant web pages based on a specific topic. This research project will provide invaluable material for researchers interested in the field of Web Crawling in any system. Inept study of the performance capabilities of various web crawling algorithms will give more insight into how to best apply them depending on the focus of the search. This will improve the overall overhead costs, increase the efficiency and work output of the system and reduce wastage of time in system resources management.

Keywords: Web crawler, bee-swarm, algorithm, web pages, internet

1.0 INTRODUCTION

A web crawler is a program that acts as an automated script which browses through the internet in a systematic way, Cabot (2017). The web crawler looks at the keywords in the pages, the kind of content each page has and the links, before returning the information to the search engine. This process is known as Web crawling. These web crawlers go by different names, like bots, automatic indexers and robots. Once you type a search query, these crawlers scan all the relevant pages that contain these words and turn it into a huge index, (Cabot, 2017).

Masanès (2007) explains that a Web crawler starts with a list of URLs to visit, called the seeds. As the crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit, called the crawl frontier. URLs from the frontier are recursively visited according to a set of policies. If the crawler is performing archiving of

websites, it copies and saves the information as it goes. The archives are usually stored in such a way they can be viewed, read and navigated as they were on the live web, but are preserved as 'snapshots'.

The difficulties in implementing efficient web crawlers clearly state that bandwidth for conducting crawls is neither infinite nor free. So, it is becoming essential to crawl the web in not only a scalable, but efficient way. There are two important characteristics of the web that generates scenario in which web crawling is very difficult.

vi. Large volume of web pages.

vii. Rate of change on web pages.

A large volume of web page implies that the web crawler can only download a fraction of the web pages and hence it is very essential that web crawler should be intelligent enough to prioritize download.

Another problem with today's dynamic world is that web pages on the internet changes very frequently, as a result, by the time the crawler is downloading the last page from a site, the page may change or a new page has been updated/placed to the site.

Web crawlers typically identify themselves to a Web server by using the User-agent field of a HTTP request. Web site administrators typically examine their Web servers' log and use the user agent field to determine which crawlers have visited the web server and how often. The user agent field may include a URL where the Web site administrator may find out more information about the crawler. Examining Web server log is tedious task, and therefore some administrators use tools to identify, track and verify Web crawlers. Spambots and other malicious Web crawlers are unlikely to place identifying information in the user agent field, or they may mask their identity as a browser or other well-known crawler (Cothey, 2004).

This work aims at addressing the problems of the web crawling through the implementation of a web based crawler using the Bee Swarm Algorithm that will be able to search the web based on a particular topic. Thus the primary objectives of this research is to develop the web based crawler to monitor the web page contents using the model based on the bee swarm algorithm

2.0 LITERATURE REVIEW

According to Shujaa and Bahaa (2013) Search engines are using web spiders to crawl the web in order to collect copies of the web sites for their databases, these spiders usually use the technique of breadth first search which is non-guided (blind) depends on visiting all links of any web site and one by one. Shujaa and Bahaa (2013) proposed a new algorithm for crawling web depending on swarm intelligence techniques, the adopted algorithm is bee swarm algorithm which takes the behavior of the bee for its work, the result in terms of speed and accuracy which means the relevancy of the collected sites.

2.1 WEB CRAWLERS

Web crawlers are programs that exploit the graph structure of the Web to move from page to page. In their infancy such programs were also called wanderers, robots, spiders, and worms, words that are quite evocative of Web imagery. It may be observed that the noun "crawler" is not indicative of the speed of these programs, as they can be considerably fast. From the beginning, a key motivation for designing Web crawlers has been to retrieve Web pages and add them or their representations to a local repository. Such a repository may then serve particular application needs such as those of a Web search engine.

2.2 WEB CRAWLING METHODOLOGIES

Given the current size of the Web, even large search engines cover only a portion of the publicly available part. A 2005 study showed that large-scale search engines index no more than 40-70% of the indexable Web a previous study by Steve Lawrence and Lee Giles showed that no search engine indexed more than 16% of the Web in 1999 (Gulli et al, 2005). As a crawler always downloads just a fraction of the Web pages, it is highly desirable that the downloaded fraction contains the most relevant pages and not just a random sample of the Web. This requires a metric of importance for prioritizing Web pages. Different methodologies and types of web crawler has been defined and used by different people. These include (i) Focused crawling: The importance of a page for a crawler can also be expressed as a function of the similarity of a page to a given query. Web crawlers that attempt to download pages that are similar to each other are called focused crawler or topical crawlers. The concepts of topical and focused crawling were first introduced by Menczer (1997) and by Chakrabarti *et al* (1999).

(ii) Restricting followed links: A crawler may only want to seek out HTML pages and avoid all other MIME types. In order to request only HTML resources, a crawler may make an HTTP HEAD request to determine a Web resource's MIME type before requesting the entire resource with a GET request. To avoid making numerous HEAD requests, a crawler may examine the URL and only request a resource if the URL ends with certain characters such as .html, .htm, .asp, .aspx, .php, .jsp, .jspx or a slash. This strategy may cause numerous HTML Web resources to be unintentionally skipped (Cho, 2001).
Some crawlers may also avoid requesting any resources that have a "?" in them (are dynamically produced) in order to avoid spider traps that may cause the crawler to download an infinite number of URLs from a Web site. This strategy is unreliable if the site uses a rewrite engine to simplify its URLs.

(iii) URL normalization: Crawlers usually perform some type of URL normalization in order to avoid crawling the same resource more than once. The term URL normalization, also called URL canonicalization, refers to the process of modifying and standardizing a URL in a consistent manner. There are several types of normalization that may be performed including conversion of URLs to lowercase, removal of "." and ".." segments, and adding trailing slashes to the non-empty path component (Pant et al, 2009).

(iv) Path-ascending crawling: Some crawlers intend to download as many resources as possible from a particular web site. So *path-ascending crawler* was introduced that would ascend to every path in each URL that it intends to crawl (Cothey, 2004) .

(v) Academic-focused crawler: An example of the focused crawlers are academic crawlers, which crawls free-access academic related documents, such as the *citeseerxbot*, which is the crawler of CiteSeer^{X} search engine. Other academic search engines are Google Scholar and Microsoft Academic Search etc. Because most academic papers are published in PDF formats, such kind of crawler is particularly interested in crawling PDF, postscript files, Microsoft Word including their zipped formats.

2.3 BEE SWARM ALGORITHM

According to Baris et al (2013), In nature, honey bees have several complicated behaviors such as mating, breeding and foraging. These behaviors have been mimicked for several honey bee based optimization algorithms.

One of the famous mating and breeding behavior of honey bees inspired algorithm is Marriage in Honey Bees Optimization (MBO). The algorithm starts from a single queen without family and passes on to the development of a colony with family having one or more queens. In the literature, several versions of MBO have been proposed such as Honey-Bees Mating Optimization (HBMO) Fast Marriage in Honey Bees Optimization (FMHBO) and The Honey-Bees Optimization (HBO)

The other type of bee-inspired algorithms mimics the foraging behavior of the honey bees. These algorithms use standard evolutionary or random explorative search to locate promising locations. Then the algorithms utilize the exploitative search on the most promising locations to find the global optimum. The following algorithms were inspired from foraging behavior of honey bees; Bee System (BS), Bee Colony Optimization (BCO), Artificial Bee Colony (ABC) and The Bees Algorithm (BA). Bee System is an improved version of the Genetic Algorithm (GA). The main purpose of the algorithm is to improve local search while keeping the global search ability of GA (Baris et al, 2013).

3.0 THE PROPOSED SYSTEM

The proposed system (bee crawling system) is built using VB.NET and a relational database management system to store web pages= Microsoft Access. The proposed system will be designed so as enable for the searching of the web via certain specific parameters so as to enhance and enrich the inter user's search experience. It will make use of the bee swarm algorithm to scan through any selected webpage to look for items or topics relating to the search query that the user seeks information on. The proposed web crawler application will be used for searching information related to any topic of interest from the Internet. It will identify the most promising links that lead to target documents, and avoid off topic searches. In addition, it does not need to collect all web pages, but selects and retrieves relevant pages only. It starts with a topic vector, and for each URL, the relevance is computed for the contribution of web page in the selected domain. If it is found to be important, it gets added to the URL list else, gets discarded.

3.1 Crawling algorithm

The bee swarm algorithm will be employed for the web crawler and it is as presented below:

Get initial web search space as n.
Get Fitness Value
Set i=0
While i<= Fittness Value
1. Increment i
2. Select the best closest links nodes
3. Recruit new "bee" nodes
4. Evaluate the fitness value of each new node
5. Sort the results based on their fitness value
6. Allocate the rest of the "bee" nodes for global search for other locations on the web page
7. Evaluate the fitness value for the other locations
8. Sort the overall results based on their fitness values

9. Rerun code until Fitness value is reached

3.2 System Architecture

The architecture of the proposed web crawler system is displayed in the figure 3.1.

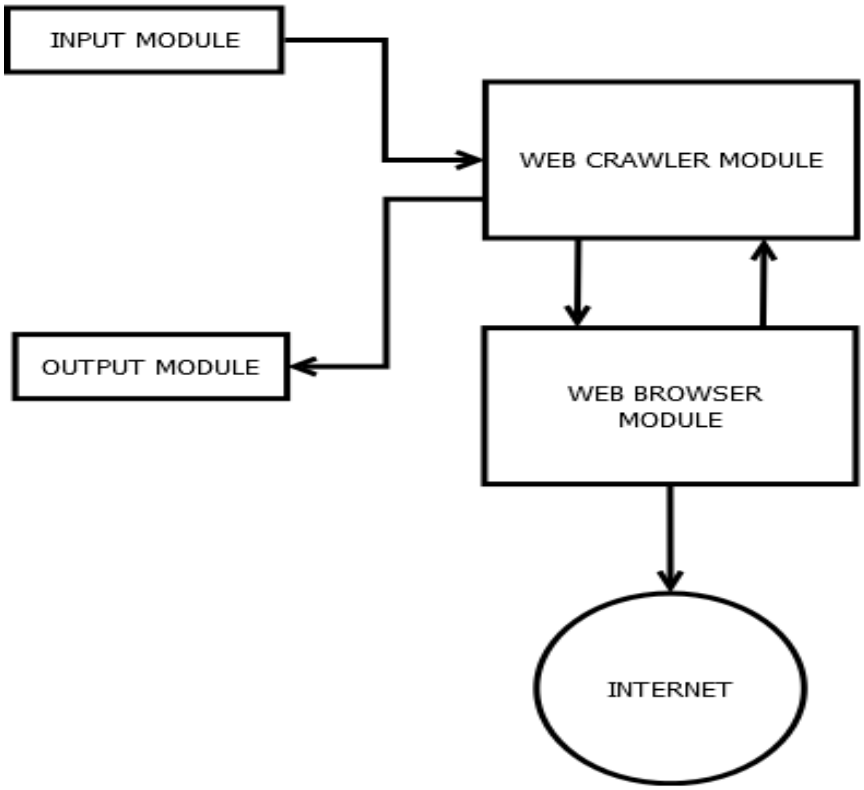

Figure 1: Architecture of the Proposed System

The following makes up the system:
Input module: This is the module that collects input from the user
Output module: this is the module that retrieves information and display to the user.
Web Crawler Module: Makes use of the bee swarm algorithm to search the web for data
Web Browser Module: Used to display the web page that is crawled
Internet: The access to the World Wide Web.

3.3 SYSTEM IMPLEMENTATION

The system was implemented using the following interfaces:
System Login Form: This form lets the system admin submits login details to gain access into the system.

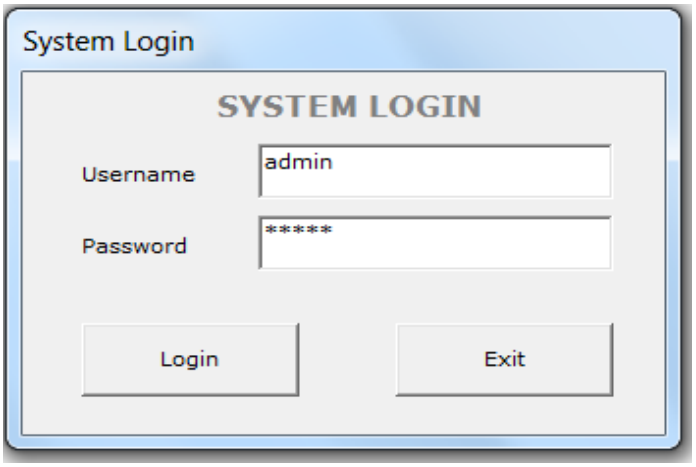

Figure 2: System Login

Web Crawler Input Form: This interface will be used to enter the desired parameters that will be used for crawling the web via the internet.

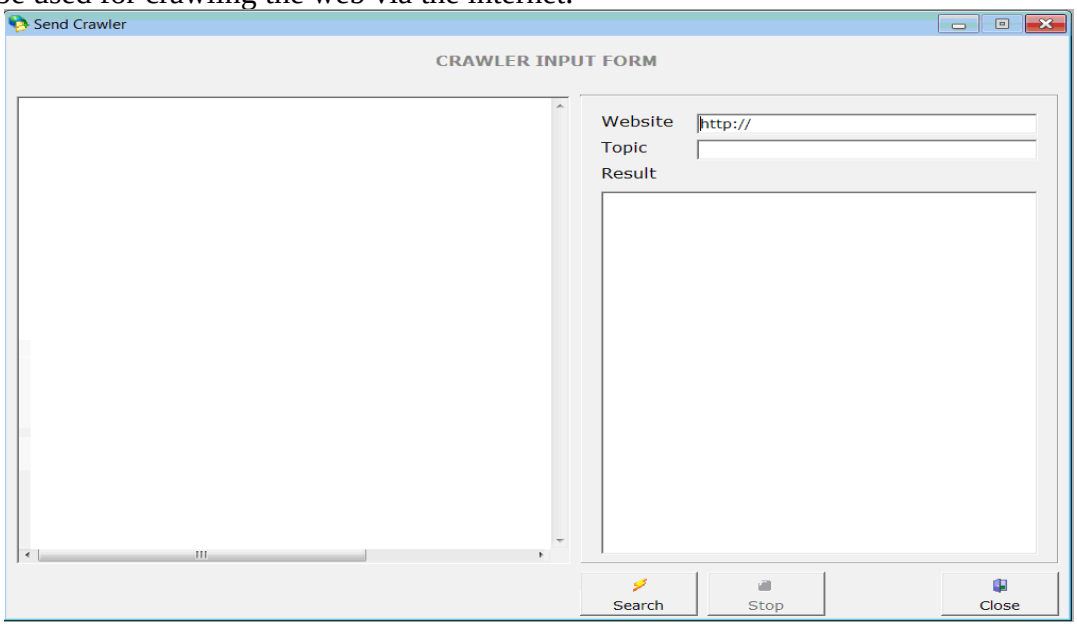

Figure 3: Web Crawler Input Form

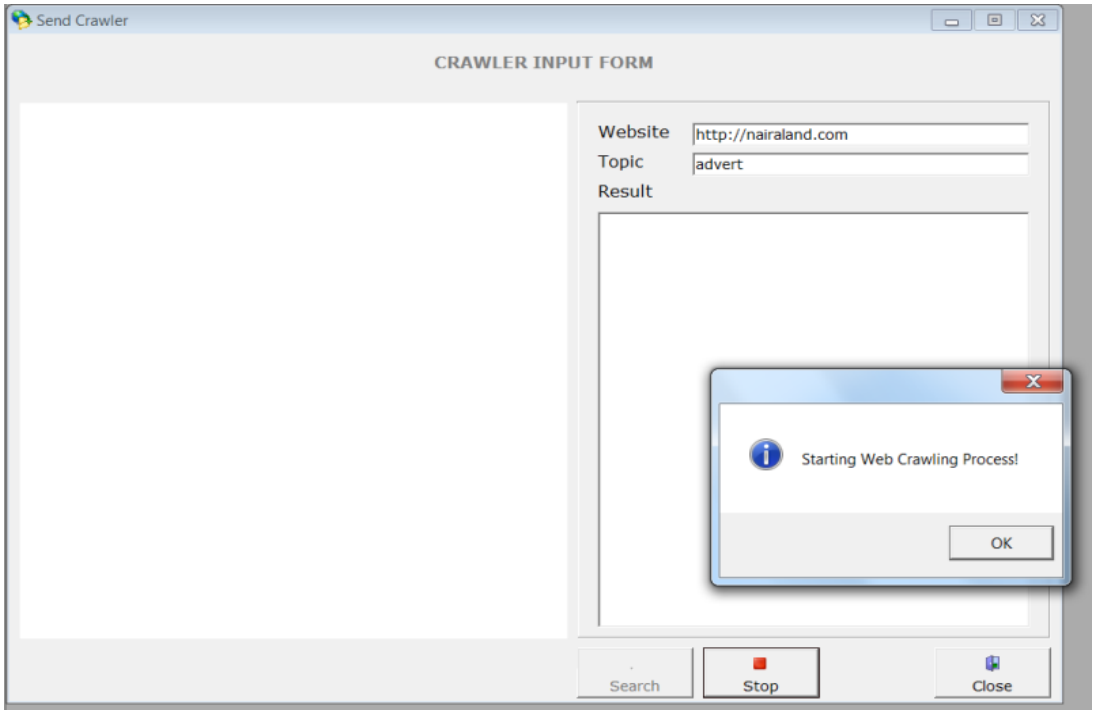

Figure 4: Web Crawling Commencement Process

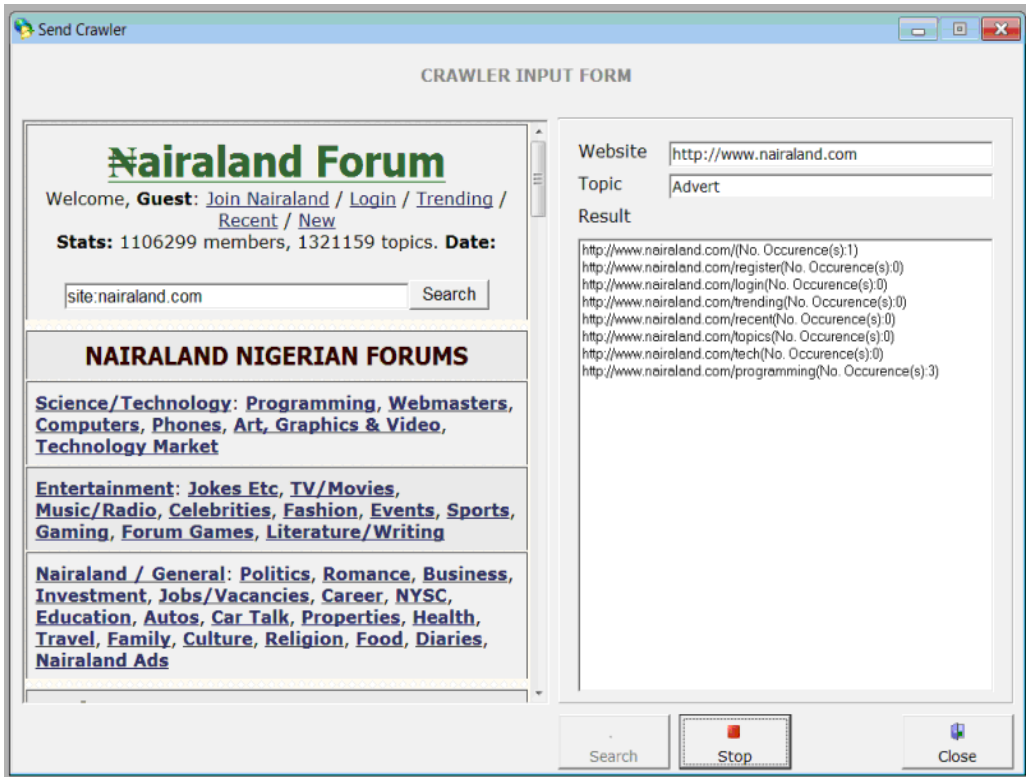

Figure 5: Web Crawling Process

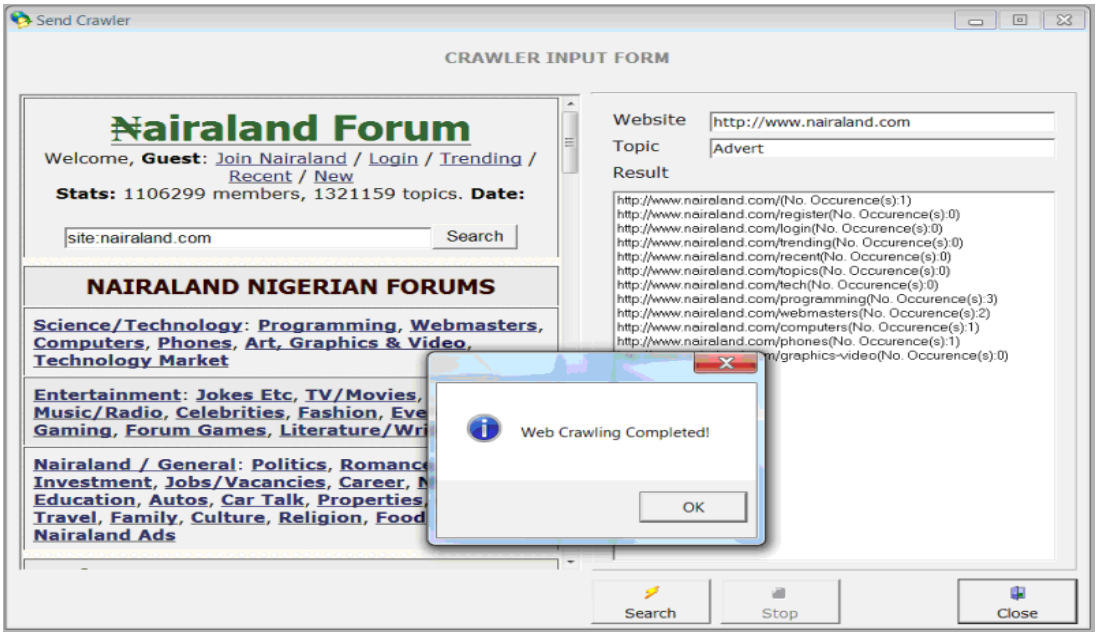

Figure 6: Web Crawling Termination

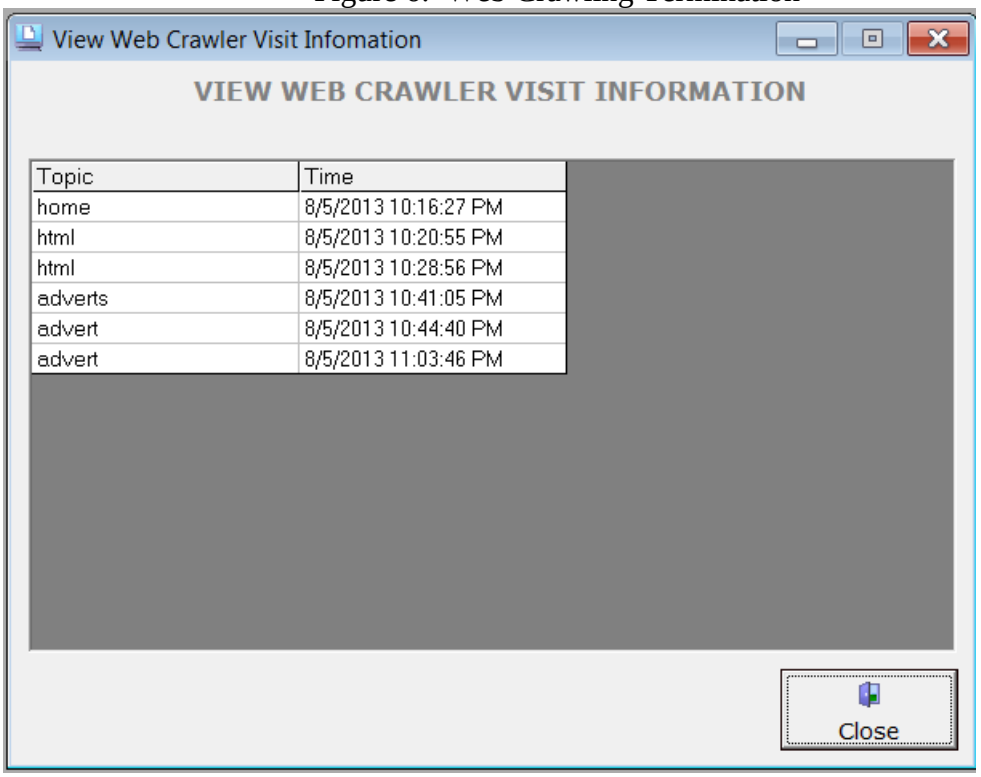

Figure 7: Crawler Visit Info Form

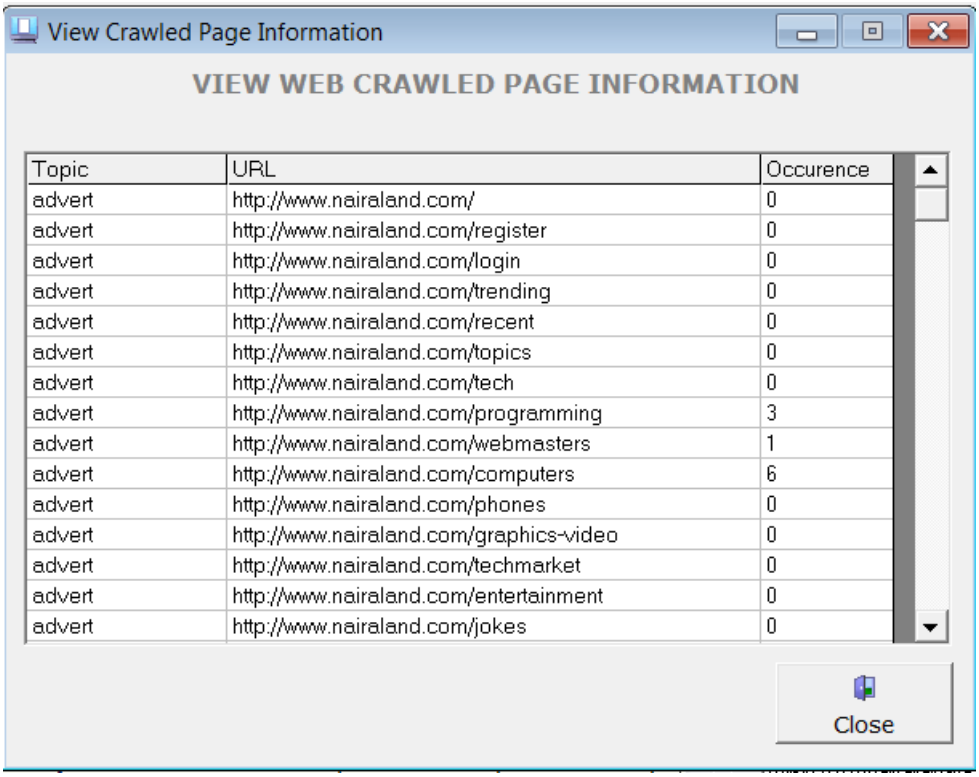

Figure 8: Crawled Page Info Form

4.0 CONCLUSION AND DISCUSSION

This research aims to address the problems of the web crawling stated in the previous section through the implementation of a web crawling application system that will be able to search the web based on a particular topic using the bee swarm algorithm. The work provides invaluable material for researchers interested in the field of Web Crawling in any system. Inept study of the performance capabilities of various web crawling algorithms will give more insight into how to best apply them depending on the focus of the search. This will improve the overall overhead costs, increase the efficiency and work output of the system and reduce wastage of time in system resources management. Swarm intelligence is a good improvement in web crawling using keywords for a group of sites with speed and relevancy of the crawled sites to desired topics.

REFERENCES

i. Baris, Y., Michael S., Packianather, E., Mastrocinque, D., Truong P., and Alfredo L. (2013) Honey Bees Inspired Optimization Method: The Bees Algorithm. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4553508.

ii. Cabot, T. (2017). Web Crawlers, Everything You Need to Know. Medium. [online] *Available From <https://medium.com/@cabot_solutions/web-crawlers-everything-you-need-to-know-6dce26ee8ad8>*.

iii. Chakrabarti, S., VandenBerg, M., and Dom, B. (1999). Focused crawling: a new approach to topic-specific web resource discovery. *Computer Networks*, 31(11–16):1623–1640.

iv. Cho, J. (2001). Crawling the Web: Discovery and Maintenance of a Large-Scale Web Data, *Ph.D. dissertation, Department of Computer Science, Stanford University.*

v. Cothey, V.,(2004). Web-crawling reliability. *Journal of the American Society for Information Science and Technology*. 55 (14): 1228–1238.

vi. Gulli, A.; Signorini, A. (2005). The indexable web is more than 11.5 billion pages. *Special interest tracks and posters of the 14th international conference on World Wide Web.* ACM Press. pp. 902–903. doi:10.1145/1062745.1062789

vii. Masanès, J. (2007). Web Archiving. *Springer*. Pg. 1. ISBN 978-3-54046332-0. Retrieved April 24, 2014.

viii. Menczer, F. (1997). ARACHNID: Adaptive Retrieval Agents Choosing Heuristic Neighborhoods for Information Discovery.*In D. Fisher, ed., Machine Learning: Proceedings of the 14th International Conference (ICML97).* Morgan Kaufmann.

ix. Pant, G., Srinivasan, P., and Menczer, F.(2002). Exploration versus exploitation in topic driven crawlers.*In WWW02 Workshop on Web Dynamics.*

x. Shujaa, M. I., Bahaa, U. A. (2013). Building web crawler based on bee swarm intelligent algorithm.*International Journal of Computer Science Issues (IJCSI).*Vol. 10 issue 5, p.134.