HYBRID FOCUSED CRAWLER - A FAST RETRIEVAL OF TOPIC RELATED WEB RESOURCE FOR DOMAIN SPECIFIC SEARCHING

Achintya Das¹ & Sudarshan Nandy²

The up-to-date web-world has become more complex in size and information relating to various aspects within its sphere. The human being is now in a cultural habit of searching the web for information. Search engine is also one of the techniques which helps the human empirical nature. Crawling is a procedure through which search engine crawls the web, and stores the necessary document and their corresponding URL in the back end. The work presented here is all about a development of a crawler which is hierarchical in nature even by maintaining parallelism during classification and analysis of the web page. This particular mechanism makes it faster than normal focused crawler used today. It is also based on the principle of the focused crawler which is also known as a topic driven crawler. With this embedded principle the hybrid crawler has the capability to search particular link related to the required query.

Keywords: Search Engine, Crawler, Focused Crawler, Hybrid-crawler, Information Retrieval, Relevance

1. INTRODUCTION

Search engine is a tool through which the user can retrieve various web services. The growth of the structure and the size of the web are enormous. The result of this has an impact on the average rate of information crawling of a normal user from the net. It is necessary for a crawler to crawl the web as fast as possible to serve the user more efficiently. Generally a search engine consists of: (a.) Query interface (b.) Crawler (c.) Repository (d.) Searcher (e.) Result interface. The query interface is a place where the user submits the information to retrieve their desired information from the web. The crawler part is basically the heart of the search engine. Crawler is responsible for classify, and retrieves the necessary information form the web. It actually separates the entire URL and store all the necessary document and URL associated with it, in a backend database. The repository is a place where the service providers publishes their services in a UDDI repository. This repository contains all the metadata appropriate for the service provider. This metadata generally consist of provider identity, category of provider and other technical details. The searcher finally finds the relevant result, and sends it to the result interface. The structure of this type of general search engine is depicted in Fig.1.

Fig. 1: Block Diagram of General Crawler

In general, the focused crawler leads to a domain specific search. The domain specific search is one of the ways to get relief from all the unusual link information in the search engine result interface. This type of crawler generally crawls all the relevant pages related to the topic. The focused one crawls only those pages which are relevant to the topic provided. The normal crawler looks for every pages and it generally take long time in-comparison with the focused crawler. The work presented here is developed on the principle of Focused crawler, and is also inherits the idea of parallel crawler. Pipeline strategy is used to run several processes and works in parallel. This type of working strategy brings a flavour of parallelism within the system. Work presented here is having the following target:

1. Implementing a hybrid system in focused crawling to reduce the load of crawler.
2. Large scale crawling.
3. Develop a domain specific searching using Hybrid-crawler.

The proposed crawling method uses some pipeline concept to maintain parallelism. These pipelines are within the control of scheduler which schedules the set of URLs generated from a node for downloading and crawling. This process also brings a hierarchical flavour to the system. So the ultimate design of the system maintains some properties...
of hierarchy and parallelism for which the work presented here actually is termed as a hybrid crawler. As the crawler works faster, the large scale crawling is possible for a particular web service. This property of the proposed hybrid-crawler is much more useful for the domain specific search. The focused crawling principle attached with this work helps for any domain specific search to concentrate on a particular topic searching. It is having a strong capability of searching topic and its related link. The search is based on the statistical analysis of finding topic in the web content.

2. Related Work

VPS or Vision based Page segmentation algorithm [2] is used to segment the page and then implying a meta search technique for finding the topic on various segments of that web page. The topic found from various segments of the web page is most relevant one. On the basis of this algorithm several other researchers like Yixue Sun [1] proposed a method of hybrid focused crawling. This paper contains a method which is used for relevance computation for web page, and is based on VIPS algorithm. The proposed hybrid focused crawling approach presented on that work is used to employ meta-search engines to maximize the crawling and semantic-structure-based web page analysis algorithm based on the VIPS algorithm. Chakraborti et al. was the first one to use a machine learning algorithm based classifier which detects the topic based on the content that belongs to that particular web page not by keyword matching [3, 6]. In that work Naïve Bayesian classifier was used to classify the web page, and hence it guides the topic driven crawler. The SVM technique was also used for the focused crawling for guiding the crawler in much more better way, and hence searches the relevant topic [7]. Many works also have been done on the sensitive filtering of data from the web while crawling which also depends on the highly sensitive keyword [8]. Some of the works are still going on for developing the focused crawling for reducing the number of irrelevant pages. Some of the research works include the BFS, sharksearch[5] or first search[4] algorithm technique for generating better result and large scale crawling. The aspect of all those works basically involves in developing various algorithmic procedures to improve the search, and to reduce the number of irrelevant search. The work presented in this paper concentrates on the speed of the crawler as well as its large scale of crawling with less number of irrelevant links.

3. System Architecture and Methodology

The proposed structure for this work is developed based on the principal of parallelism and with the help of a classifier it extracts the main documented part from the web page. There is scheduler section where proper hierarchical order is maintaining the index tree. So, in this architecture there is a blend of hierarchical as well as parallelism, and hence it can be called as a hybrid work. This hybridization is maintained all over the process, and thus its a Hybrid-crawler. The parallelism maintained inside of the crawling process is depicted in Fig. 2.

![Fig. 2: System Architecture of Parallel Pipeline](image)

In Fig.2 master process is the main program that controls all of its child process or customer. The customer here is of following type down-loader, parser and searcher and simultaneously they can communicate between them using pipe. The communication channel here is pipe and through which all read writing process are possible. The pipe is used here as a wrapper over the job channel. In this work the python multiprocessing library is used to maintain the parallelism of over all system.

In the Fig. 3. the system control flow structure is shown. The crawler is connected to the repository. The contain of the repository is a set of URL addresses. These addresses are very much relevant to the topic that the hybrid crawler needs to search for. First of all, the hybrid crawler fetches all URLs related to the specific topic from the repository. Then it inserts any one URL into the URLDB and initiate a new level. URLDB is a general URL database, storing the entire URL that are crawled by crawler from various web page content. A level may contain a number of URLs, but whatever the number of URLs are available for that particular level at a time one URL is send to the next step. Before proceeding further the URL is verified with the content of Visited DB where the name of visited URL is stored. If it is found on the Visited DB, then it discards that URL, and looks for next URL. On finding out next URL, that URL proceeds, otherwise it increases the level and fetches the first node(URL) of the previous level. After checking this URL, the next level of checking is proceeded for the response and other necessary information from the web server about that URL. If the URL passes through this check point, it passes to the download section which is connected directly to Web. The connection between download section and web should be supported by strong internet architecture. Then the content of that URL is passed to the Parser section. The parser is responsible for finding the whole documented part within the web page and their associated URL. Searcher part is responsible to find the
specific topic within the document part of Web page. The searcher looks for the specific topic all over the page. This section separates all the tags and URLs from the web page and classifies only the document part or any sentence of the web page, and then starts the search for particular topic. The downloading section, parser and searcher of Hybrid crawler maintain their inter process communication using Pipe and all the created process as maintained through the process synchronizer (Fig. 2.).

Fig. 3: System Architecture of Topic Driven Hybrid Crawler

3.1. Rules for Weight Table Construction

In order to make proper analysis of the proposed method, some of the parameters are needed to be observed and calculated during the running time of the Hybrid crawler. In this work the hybrid crawler generates a complete evaluation of the following parameters:

1. Numbers of words it uses to search for a topic: \( W_N \);
2. Numbers of matches found in document section: \( W_D \);
3. Numbers of matches found in Title tag of Web page: \( W_T \).

In parameter the number of matches found in the title as well as in the document part is important for this proposed work methodology. These two parameters along with number of ords found for searching form the key parameter to calculate the relevance of the page. From the experimental observation of various code structures for generating a web page, the title part is considered as important section for statistical analysis. The title part of any web page generally consists of gist assigned to the particular web page to recognize it within various web service. In case of document section same concept can be applied. In order to make a proper analysis, weights are assigned to parameters according to the number of times match found on the classified section (document, title) of that web content. The number matches found in the title tag of the web page is considered one important value of the page. Hence, for this proposed method, the weight considered for the title section is \( \eta = 2 \). The value of \( W_N \) or \( D \), \( W_D \) or \( T \) and \( W_T \) or \( N \) and can be expressed as follows:

\[
D \wedge T \equiv N:
\]

3.2. Relevance Calculation

The specified topic may appear in the different part of the web page. In this work the web content is classified into document and title section reflecting the content of the web page. In order to calculate the relevance of the page for a topic, the statistical parameter used is as presented below:

\[
\text{Relevance}(D, T) = \frac{\sum W_N \theta + \sum W_D \Phi + \sum W_T \eta}{\sum W_D \Phi + \sum W_T \eta}
\]

Here, \( \theta \), \( \Phi \), \( \eta \) are the weights assigned to \( W_N \), \( W_D \) and \( W_T \) respectively. If the relevance of a web page reaches specific level, that page can be considered for topic specific page. The crawler efficiency is also taken into account in this analysis. The efficiency can be measured through the number of successful URL visited per hour for the topic.

4. Experiment and Result Analysis

In the experimental set up for testing this method, a computer with Core 2 Duo (1.77GHz) processor, 2GB RAM, more than 500GB of hard drive space is used. The network infrastructure used for this work has a capacity in an average 20 – 25 Kb/Sec of data transfer rate. The result shown here is basically the relevance of a web page for the topic and the efficiency of the crawler. For the present experiment, following topics are studied:

A. HIV/AIDS    B. H1N1 or Swine Flu

4.1. Experiment 1

In the first experiment on the proposed method one repository is used. This repository contains number of URLs which are related to the topic. The list of relevant URLs found is:

http://www.unicef.org/aids/
http://www.undp.org/hiv/
http://www.unfpa.org/public/News/pid/1368
http://www.unfpa.org/public/News/pid/1038
http://www.unfpa.org/public/News/pid/1025
unesco-publications.asp
http://www.who.int/hiv/topics/mtct/en/index.html
http://www.who.int/hiv/en/
http://www.unfpa.org/hiv/
http://www.worldaidscampaign.org

http://www.unaids.org/UNGASS2010/

Some limited numbers of URLs are listed here due to space limitation. These relevant URLs are found from a set of 6000 numbers of visited nodes. The relevance of the analysis is depicted in the Fig. 4(a) and its relevance shown in Fig. 4(b). The graph plotted above with the help of ‘matplotlib’ library and the dots on the graph represents the relevance values calculated during the time of experimental analysis. In the average relevance graph the relevance value and average URL crawling is scaled over 100. The fluctuation in the relevance graph can be observed easily and it is caused due to crawling fluctuation rate. The crawling rate depends on the network speed and it is tested in an environment where the network speed is not stable. To understand this matter the indexing done by crawler during the experiment is depicted in the Fig. 5.

In Fig. 5 it is clearly observed that the rate of crawling fluctuates and it varies in between 0 to 54,217.

![Fig. 4: (a) The Relevance Calculation of Found URL (b) Average Relevance](image)

In Fig. 5 it is clearly observed that the rate of crawling fluctuates and it varies in between 0 to 54,217.

![Fig. 5: Index Rates of Crawler](image)

It is observed during the time of this experiment that when it touches the zero crawling rate in the 2nd running hour of crawler, the network speed is too low. Opposite thing happens with network speed of about 30-40 kb/sec, when crawling rate touches upper cut. In the search engine database URL DB table is used for the indexing purpose. The SQL query used for retrieving data from table is:

```
SELECT URL, URLTIME FROM 'search_engine10'. 'URLDB'
WHERE URLTIME BETWEEN 'time1' AND 'time2'.
```

In the query the ‘time1’ is different from ‘time2’. The query is manually passed to the MySQL query interface with 15 minutes time interval between ‘time1’ and ‘time2’, and the data are plotted in the fig. 5. Query generates a list of URL with its relevant crawling time and the list contains all the URL crawled within 15 minutes.

**4.2. Experiment 2**

This experiment is also done on the same environment and on same system. The topic is H1N1 or swine flu, and repository for the experiment is different from the previous
one. The relevant URLs found are:

- http://www.flu.gov/
- http://www.cdc.gov/h1n1flu/
- http://www.cdc.gov/h1n1flu/diagnostic_testing_public_qa.htm
- http://www.cdc.gov/h1n1flu/guidance_homecare.htm

These URLs are the sample from large content of swine flu relevant URL database. The contain capacity is about more than thousand and all of them are based on some concept of given topic. The crawler gathers more than 500 of URLs in an hour. The relevance value analysed is depicted in Fig. 6(a) and its average relevance over 100 scales is depicted in Fig. 6(b).

![Fig. 6: (a) The Relevance Calculation of H1N1 or SWINE FLU. (b) Average Relevance Over 100](image)

The experiment is manually stopped after some time and it is the reason behind that the line is stopped after 80 in the x-axis of fig. 6(a). In fig. 6(a) the average is taken over every 100 relevant value.

![Fig. 7: Index Rates of Crawler](image)

It can be easily observed from the relevance of the graph that it fluctuates highly. So it reflects the fact of crawling which is done at slower network speed of about 20 to 25 kb/sec. In this respect the indexing also fluctuates and it takes long time to crawl in the long range as it is done in the previous experiment. The indexing graph is generated with the help of same SQL query used in the experiment 1 and it is depicted in fig. 7. The index graph of two experiments shows that the crawling fluctuation rate affects due to the network speed. The first experiment shows that the upper limit (exact 54217) in the indexing rate done by the crawler is much better than second experiment (exact 29411). The experiment clearly reflects that with a standard network speed this methodology of crawling can perform a large scale crawling. It has been observed that with some stable network speed large scale crawling can be much faster than usual crawler.

5. CONCLUSION

The experimental analysis shows that it is possible for hybrid focused crawler to crawl more web resources in short time span. The experiment is done on low network speed, but the crawler still runs and accommodates URLs until it is stopped manually. The network speed throughout this experiment is fluctuating between 20 to 30 kb/sec, and it is observed that when some more band width is available, the crawler runs faster than its usual speed. So for this methodology of crawling, the crawl speed becomes proportionate with network speed. Hence, the efficacy of the network crawler in terms of fast resource retrieval becomes established.

REFERENCES


