Web Search Engine: A New Crawling Tool for Mining Web Pages over Internet based on Page Usage

Gowthami.G [#1], S. A. Bhavani[*2]

M.Tech Scholar [#1], Assistant professor[*2]

Department of Computer Science & Engineering,
Anil Neerukonda Institute of Technology & Sciences
Bheemunipatnam (Municipality),Sangivalasa - 531162
Vishakapatnam (District), AP (INDIA).

Abstract

A web Search is a program, which automatically traverses the web by downloading documents and following links from page to page. They are mainly used by web search engines to gather data for indexing. Other possible applications include page validation, structural analysis and visualization; update notification, mirroring and personal web assistants/agents etc. Web Search are also known as spiders, robots, worms etc.In this project we are using the web search program as a crawler application where the crawling of the pages is done not by the overall page rank(I.e. Overall total page's rank count visited by users),but the pages are crawled based on individual page count of individual URL's.As our application is used to measure the individual page traffic accurately our application is mainly useful for Web –Masters for maintaining the traffic of each and every web page in an very sophisticated manner. As this application requires internet connection, the internet connection should be of enough bandwidth as in order for processing the Web pages URL's accurately and fastly.This application is limited for crawling up to non SSL protected pages but it is failed in crawling SSL protected pages as due to governing internet security policies. In this current application we can find out the count of successfully crawled URL's as well as failed URL's successfully based on the pages which were crawled by internet traffic.This work implements the "Breadth First Searching" algorithm, a refined version of one of the first dynamic Web search algorithm.

Keywords

BFS Algorithm, Web-Robots, Web agents, visualization, page validation

1. Introduction

The potential power of web mining is illustrated by one study that used a computationally expensive technique in order to extract patterns from the web and was powerful enough to find information in individual web pages that the authors would not have been aware of .A second illustration is given by the search engine Google, which uses mathematical calculations on a huge matrix in order to extract meaning from the link structure of the web. The development of an effective paradigm for a web-mining Search is, therefore, a task of some importance.

A web Search, robot or spider is a program or suite of programs that is capable of iteratively and automatically downloading web pages, extracting URLs from their HTML and

fetching them. A web Search, for example, could be fed with the home page of a site and left to download the rest of it.

Other possible applications include:-

- Page validation
- Structural analysis and visualization
- Update notification
- Mirroring and personal web assistants/agents etc.

For example, a user can navigate from the entry page to a thread page through the following paths as shown in Figure 1

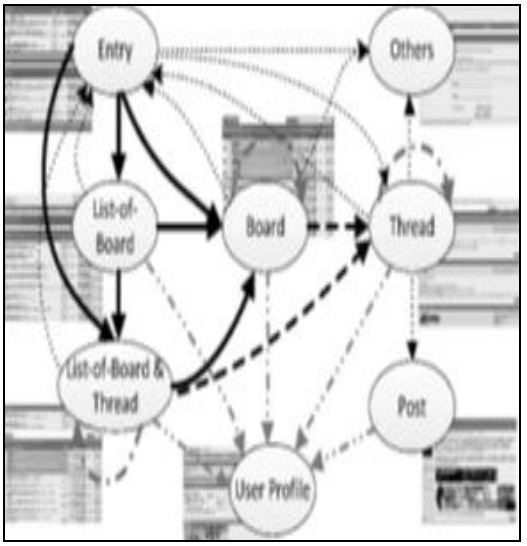

Figure. 1 Link relations

1.1 Main Fundamentals of a Web Search Program

Despite the numerous applications for Web Search, at the core they are all fundamentally the same. Following is the process by which Web Search work:

1. Download the Web page.

2. Parse through the downloaded page and retrieve all the links.

3. For each link retrieved, repeat the process.

Now let's look at each step of the process in more detail.

In the first step, a Web Search takes a URL and downloads the page from the Internet at the given URL. Oftentimes the downloaded page is saved to a file on disk or put in a database. Saving the page allows the Search or other software to go back later and manipulate the page, be it for indexing words (as in the case with a search engine) or for archiving the page for use by an automated archived.

In the second step, a Web Search parses through the downloaded page and retrieves the links to other pages. Each link in the page is defined with an HTML anchor tag similar to the one shown here:

```
<A
HREF="http://www.host.com/directory/file
.html">Link</A>
```

After the Search has retrieved the links from the page, each link is added to a list of links to be searched.

The third step of Web Searching repeats the process. All Search work in a recursive or loop fashion, but there are two different ways to handle it. In our project the Links can be Searched using breadth-first manner.

2. Related Work

In this section we will describe the research motivation and assumptions that are used in the proposed paper.

2.1 Research Motivation

The potential power of web mining is illustrated by one study that used a computationally expensive technique in order to extract patterns from the web and was powerful enough to find information in individual web pages that the authors would not have been aware of .A second illustration is given by the search engine Google, which uses mathematical calculations on a huge matrix in order to extract meaning from the link structure of the web. The development of an effective paradigm for a web-mining Search is, therefore, a task of some importance.

A web Search, robot or spider is a program or suite of programs that is capable of iteratively and automatically downloading web pages, extracting URLs from their HTML and fetching them. A web Search, for example, could be fed with the home page of a site and left to download the rest of it.

2.2 Research Mechanism

The proposed research model can be used as a web crawler, which is a program that automatically traverses the web by downloading documents and following links from page to page. They are mainly used by web search engines to gather data for indexing.

Other possible applications include:-

- Page validation
- Structural analysis and visualization
- Update notification
- Mirroring and personal web assistants/agents etc.

Web crawlers are also known as spiders, robots, worms etc.

"RANKING MODEL" which actually searches the data at the time when the URL is issued. While this "RANKING MODEL" does not scale up, it guarantees valid results for dynamic search. It is preferable to static search for discovering information in small and dynamic sub Webs. Finally, "*ranking adaptability"* measurement is proposed to quantitatively estimate if an existing ranking model can be adapted to a new domain with several experiments on various sites. By using our proposed ranking model using SVM, we get the following advantages:

- The proposed RA-SVM can better utilize by both the auxiliary models and target domain labeled queries to learn a more robust ranking model for the target domain data.
- Domain-specific features can steadily further boost the model adaptation, and RA- SVM-SR is comparatively more robust than RA-SVM- MR.
- Adaptability measurement is consistent to the utility of the auxiliary model, and it is deemed as an effective criterion for the auxiliary model selection

3. Proposed Algorithm and Methodology

In this paper we are going to implement BFS (Breadth First Search) Algorithm for implementing the search traversal technique and finally the sorted URLS will be placed in the order of BFS Hierarchy where there will be no chance of repetance of visited URL to be replaced once again in between the searched URL's.

3.1 Breadth First Search Algorithm

Breadth First Search (BFS) is an uninformed search method that aims to expand and examine all nodes of a graph systematically in search of a solution. In other words, it exhaustively searches the entire graph without considering the goal until it finds it. It does not use a heuristic.

From the standpoint of the algorithm, all child nodes obtained by expanding a node are added to a FIFO queue. In typical implementations, nodes

that have not yet been examined for their neighbors are placed in some container (such as a queue or linked list) called "open" and then once examined are placed in the container "closed".

In order to build a major search engine or a large repository such as the Internet Archive, high-performance Search start out at a small set of pages and then explore other pages by following links in a "breadth first-like" fashion. In reality, the web pages are often not traversed in a strict breadth-first fashion, but using a variety of policies, e.g., for pruning Search's inside a web site, or for Searching more important pages first.

Steps for BFS Algorithm are:

1) Put the starting node (the root node) in the queue.
2) Pull a node from the beginning of the queue and examine it.
   a) If the searched element is found in this node, quit the search and return a result.
   b) Otherwise push all the (so-far-unexamined) successors of this node into the end of the queue, if there are any.
3) If the queue is empty, every node on the graph has been examined -- quit the search and return "not found".
4) Repeat from step 2.

3.2 Pseudo code for BFS Algorithm

The below pseudo code clearly represents the BFS algorithm procedure and its working principle.

Pseudo-Code for BFS Algorithm

```
function breadthFirstSearch (Start, Goal) {
    enqueue(Queue,Start)
    while notEmpty(Queue) {
      Node := dequeue(Queue)
      if Node = Goal {
        return Node  // the code below does not get executed
      }
      for each Child in Expand(Node) {
        if notVisited(Child) {
          setVisited(Child)
          enqueue(Queue, Child)
        }
      }
    } }
```

4. Implementation Roles

The following are the roles that are performed during the process of crawling the web pages based on individual page traffic**.**

4.1 Google API Role Flow Diagram

In this Google API's Role, it shows that operations performed by the Google API after accepting the URL from the user. The default actions are specified by the Ranking Model Web Crawler which is clearly shown in figure 2.

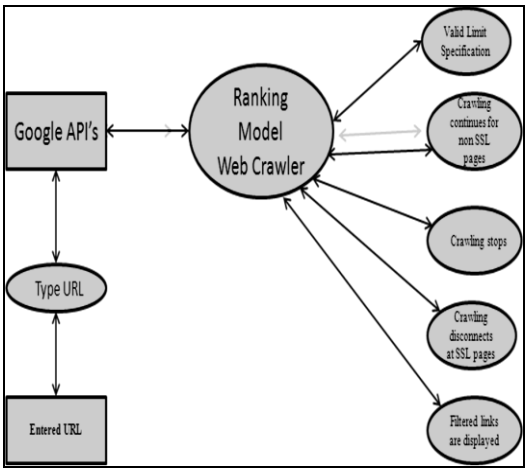

Figure 2. Role of Google API in our application

4.2 Role of User and his Flow of Events Diagram

In this User Role, it shows that the operations performed by the user after accepting the appropriate URL details from Google API's**.** The default actions which should be performed by the user is specified by the Ranking Model Web Crawler which is clearly shown in figure 3.

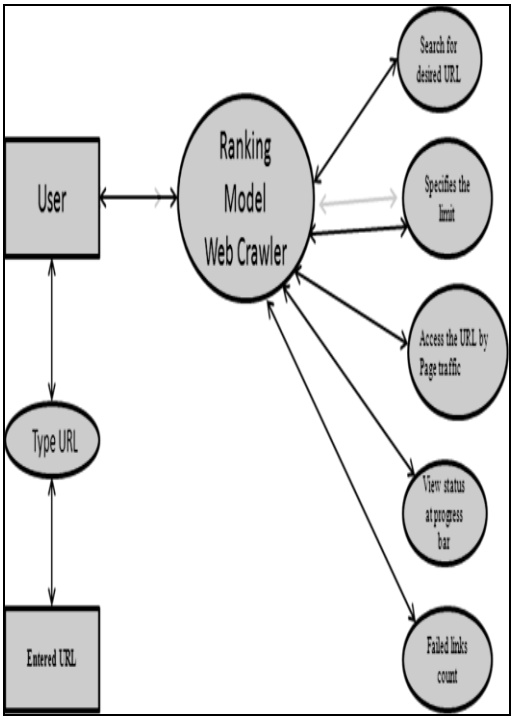

Figure 3. Role of User in our application

5. Implementation Modules

System Implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the user that it will work efficiently and effectively. The existing system was long time process. The proposed system was developed using Java Swing. The existing system caused long time transmission process but the system developed now has a very good user-friendly tool, which has a menu-based interface, graphical interface for the end user.

After coding and testing, the project is to be installed on the necessary system. The executable file is to be created and loaded in the system. Again the code is tested in the installed system. Installing the developed code in system in the form of executable file is implementation.

There are mainly seven modules, they were divided based on two categories:-

1) User category
2) Server category

User Based Modules

There are totally 3 modules or tasks which come under user category. They are as follows:-

- Provide input to the system
- Request for Search
- Receive the links of his given URL

Provide input to the system:

Here in this module, the user needs to provide the valid information for the crawling of data. Here the input is a valid web address that starts with Http and ends with any of the domains like .com /.org/.eddo/.in or any domain name. If the Http Address consists of any invalid characters or wrong mail id, the application is not able to crawl the required user search data**.**

Request for Search:

Here in this module, the user after entering valid http web site address, he needs to request the server for search the web site name with the URL what we have provided to the application. For executing this module we need to have internet connection, if the connection fails the server can't able to accept the URL request which is given by the user.

Receive the links of his given URL:

In this module, if the internet connection is available perfectly for the user request, the application is able to take the request to the Google server. Google server process the links that was asked by the user privilege and finally send the traffic links of searched Ural to the requested User.

Server Side Modules

They are totally four modules or tasks which come under Server side. They are as follows:

- The system takes input (URL, Limit) from the user.
- Transmits the user request to the server
- Receives the Web pages from the Internet.
- Processes the Web documents to extract the links**.**

The system takes input (URL, Limit) from the user:

This is the module where the server takes the user input in the form of (URL, Limit) for processing the required Web pages traffic links. By default the minimum limit for URL's is 5 and maximum is we can give any required number of user wish. The system takes the request for Search from the user**.**

Transmits the user request to the server:

In this module the user request what that is asked by user is taken to the server and finally processed for retrieving output links**.**

Receives the Web pages from the Internet

In this module, the server will able to give the retrieved web pages to the user based on the limit, All the links may not be successfully crawled but some may failed while crawling so the failed URLS count is also indicated to the user in the final result.

Processes the Web documents to extract the links:

In this module the server gives the privilege to the user to process the web URL's that was generated by the server**.**

6. Conclusion

We have described the architecture and implementation details of our Searching system. Searching forms the backbone of applications that facilitate Web information retrieval. The web Search is designed using Breadth-first Searching, which provides the highest page rankings. Itself capable of searching a large collection of web sites by using idle processing power and disk space. The testing of the system has shown that it cannot operate fully automatically for tasks that involve searching entire web sites without an effective heuristic for identifying duplicate pages.

7. References

[1]. Herbert Schildt Java Complete Reference. Fifth Edition.

[2]. Core Java 2: Volume I & II - Fundamentals By Cay S. Horstmann, Gary Cornell

[3]. Java™ Tutorial, Third Edition: A Short Course on the Basics by Mary Campione, Kathy Walrath, Alison Huml.

[4] .Data Mining Techniques By Arun K Pujari.

[5]. Flippo Menczer, Gutam Pant, Padmini Srinivasan, Searching the Web.

[6]. Pankaj Jalote, An Integrate Approach to Software Engineering, 3rd Edition.

[7]. Searching the Web. Gautam Pant, Padmini Srinivasan and Filippo Menczer3.

[8]. S. Chakrabarti. Mining the Web. Morgan Kaufmann.

[9]. F. Menczer and R. K. Belew. Adaptive retrieval agents: Internalizing local context and scaling up to the Web. Machine Learning

[10]. J. Rennie and A. K. McCallum. Using reinforcement learning to spider the Web efficiently.Morgan Kaufmann, San Francisco, CA, 1999.

[11]. G. Salton and M.J. McGill. Introduction to Modern Information Retrieval. McGraw-Hill, 1983.

[12]. "Efficient Searching Through URL Ordering", Junghoo Cho, Hector Garcia-Molina, Lawrence Page. 7th International Web Conference (WWW 98).

8. About the Authors

Gowthami.G is currently pursuing her 2 Years M.Tech (CSE) in Department of Computer Science and Engineering at Anil Neerukonda Institute of Technology & Sciences, Bheemunipatnam (Municipality), Sangivalasa, Visakhapatnam. Her area of interests includes Data Mining

S. A. Bhavani is currently working as Assistant Professor, in Department of Computer Science and Engineering at Anil Neerukonda Institute of Technology & Sciences, Bheemunipatnam (Municipality), Sangivalasa, Visakhapatnam. Her research interests include Networks Security & Image Processing.