SMART CRAWLER-A TWO STAGE FRAME WORK FOR CRAWLING DEEP WEBSITES

[1]D.YASWANTH, [2]A.S.ANUHYA, [3]Mr.B.MANOJ,
[1,2]UG Student, [3]Assistant Professor, Department of Computer Science and Engineering,
Godavari Institute of Engineering & Technology, Rajahmundry, AP

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. Web Search is also known as spiders, robots, worms etc. In this proposed project the web search program acts as an smart 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.

The proposed smart application is used to measure the individual page traffic accurately and is mainly useful for Web–Masters to maintain the traffic of each and every web page in very sophisticated manner. As this application requires internet connection, the internet connection should be of enough bandwidth for processing of the Web pages URL's accurately. 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. In this project BFS algorithm also known as "Breadth First Search" is used, for crawling the web pages based on the individual page rank and there will be no back traversing of URL's, so this will give unique results. As an extension the monitoring of count of failed URL's and count of successful URL's crawled in this application has also been included, this gives more importance for the application of each and every URL.

Key words: Web search, Web pages, Crawled URL's, Breadth First Search, Traversing, etc.

1. INTRODUCTION

The economic and cultural importance of the web has guaranteed considerable academic interest in it, not only for affiliated technologies, but also for its content. Research into web pages themselves has been motivated by attempts to provide improved information retrieval tools such as search engines, but also by the desire to know what is available, how it is structured and to determine its relationship with other meaningful human activities. The advanced facilities available in search engines such as AltaVista and Info seek, but their use has raised questions of reliability that have led to the creation of a specialist web spider/analyzer to produce the raw data by direct Searching and analysis of the sites concerned [3]. Information scientists and others wishing to perform data mining on large numbers of web pages will require the services of a web Search or web-Search-based tool, either individually or collaboratively [5].

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 [5].

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

2. PROPOSED MODEL

In contrast, the multithreaded dynamic web Search actually searches the data at the time the URL is issued. While it doesn't not scale up, dynamic search guarantees valid results, and preferable to static search for discovering information in small and dynamic sub Webs. The benefits of the proposed model are:

1. The current system dynamically searches the web subnets for a given URL, so the downloaded links are the most recently updated links.
2. The user has the option of specifying from which website the search has to be started. The searching points are called seeds.
3. The client computer's speed is fully utilized.

3. ALGORITHM USED

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 Searches 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.
   - If the searched element is found in this node, quit the search and return a result.
   - 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.

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. RESULTS

Fig 1-Home Page

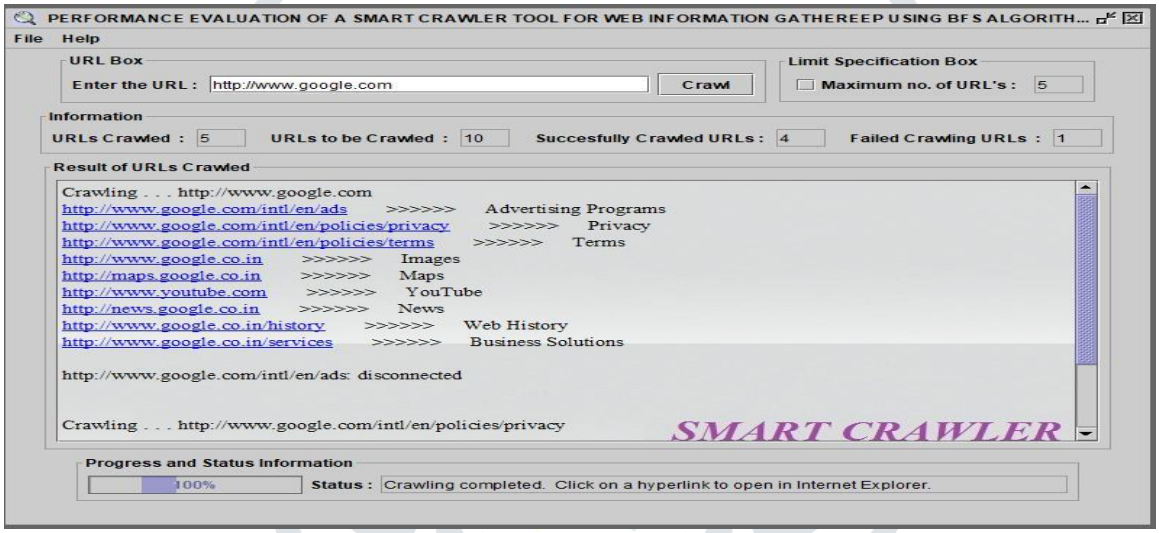

Fig 2-Processed Window with Complete Progress

5. CONCLUSIONS AND FUTURE SCOPE

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. It's 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.

We design this product for providing basic search features. The Web Search software can use a Search to look for links in a commercial Web site. We might also use a Search to find changes to a Web site. And also find out relative and absolute links. The web Search is designed in Breadth-first search and we can view all the contents present in those links.

A number of Searching algorithms have been proposed such as depth-first Searching, best-first Searching, fish search algorithm and shark search algorithm. The work can also extended by implementing any of those algorithms, keywords instead of URL, removing stop words, stemming, server redirection, Canonicalization, normalization and Robot exclusion principle.

REFERENCES

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

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

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

[4] Michael K. Bergman. White paper: The deep web: Surfacing hidden value. Journal of electronic publishing, 7(1), 2001.

[5] Yeye He, Dong Xin, Venkatesh Ganti, Sriram Rajaraman, and Nirav Shah. Crawling deep web entity pages. In Proceedingsof the sixth ACM international conference on Web search and data mining, pages 355–364. ACM, 2013.

[6] Martin Hilbert. How much information is there in the "information society"? Significance, 9(4):8–12 2012.

[7] Denis Shestakov and Tapio Salakoski. Host-ip clustering technique for deep web characterization. In Proceedings of the12th International Asia-Pacific Web Conference (APWEB), pages 378–380. IEEE, 2010.

[8] Eduard C. Dragut, Thomas Kabisch, Clement Yu, and UlfLeser. A hierarchical approach to model web query interfaces for web source integration. Proc. VLDB Endow., 2(1):325–336, August 2009.

[9] Jayant Madhavan, Shawn R. Jeffery, Shirley Cohen, Xin Dong,David Ko, Cong Yu, and Alon Halevy. Web-scale data integration: You can only afford to pay as you go. In Proceedingsof CIDR, pages 342–350 2007.

[10] Eduard C. Dragut, Weiyi Meng, and Clement Yu. Deep WebQuery Interface Understanding and Integration. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2012.

ACKNOWLEDGEMENT

We have great pleasure in expressing our gratitude to Sri K.V.V.Satyanarayana Raju, Founder & Chairman, Chaitanya Group of Institutions, Sri K. Sasi Kiran Varma, Vice Chairman, GIET Group of Institutions, Smt. Lakshmi Raju Executive Director, GIET, for their kind support in providing us an opportunity to do research in this college.