ISSN No. 0976-5697

Volume 3, No. 5, Sept-Oct 2012

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.inf

Focused Web Crawling: An Architecture for Crawling of Country Based Financial Data

Debakar Shamanta*
Lecturer, Department of CSE
Shahjalal University of Science and Technology (SUST)
Sylhet, 3114, Bangladesh
debakar-cse@sust.edu

Marufa Rahmi
Lecturer, Department of CSE
Shahjalal University of Science and Technology (SUST)
Sylhet, 3114, Bangladesh
mrahmi-cse@sust.edu

Abu Awal Md. Shoeb
Assistant Professor, Department of CSE
Shahjalal University of Science and Technology (SUST)
Sylhet, 3114, Bangladesh
shoeb-cse@sust.edu

Abstract: Crawling the Web quickly and entirely is an expensive, unrealistic goal because of the required enormous amounts of hardware and network resources. But when only information about a predefined topic set is desired, a specialization of the aforementioned process called "focused crawling" is used. A focused crawler is an agent that targets a particular topic and visits and gathers only a relevant, narrow web segment while trying not to waste resources on irrelevant material. Compared to the standard web search engines, focused crawlers yield good recall as well as good precision by restricting themselves to a limited domain. In this paper we will introduce the technique of focused crawling of country based financial data.

Keywords: Crawler; Focused Crawler; Information retrieval; Financial data.

I. WEB CRAWLER

A web crawler is a program that collects web content from the World Wide Web automatically and stores this content into the storage. It starts with a list of URLs (seeds) to visit. [1] When it visits these pages, it parses all links from these pages. After collecting all these links the web crawler inserts them in the URL queue. Web crawler continuously visits the unseen links and also scans them for discovering more links and put only the unseen links in the URL queue. The web crawler is also called web robot, web scutter, web spider, ants, worms and bots. The basic algorithm [2] is:

a. Fetch a page
b. Parse it to extract all linked URLs
c. For all the URLs not seen before, repeat (a)–(c)

A good crawler for a large search engine has to address the following issues:

i. It needs to have a highly optimized system architecture that can download a large number of pages per second.
ii. It should have good memory management system to avoid the memory stack overflow.
iii. It has to decide which pages should be downloaded next.
iv. It must be strong against crashes.
v. It has to be manageable by the existing resources and web servers.

A. Basic Web Crawler Structure:

A crawler must have a good crawling strategy [3], but it also needs a highly optimized architecture. Our basic crawling steps are following:

a. Put all previous links in URL queue.
b. Start Crawling from top URL in URL queue or from seed URL.
c. Check the content type to categorize the page means doc, html, pdf etc. If valid page means HTML page then fetch module read the content of the page.
d. The URL checker module that uses to check the URL is alive or dead. That means check HTTP error, URL error etc. If valid then save the page in zip format.
e. The parsing module that extracts set of links from a fetched web page.
f. A duplicate elimination module that determines whether an extracted link is already in the URL frontier or has recently been fetched.
g. The duplicate elimination module that determines whether an extracted link is already in the URL queue or has recently been fetched.
h. If the link is new then DNS resolution module solve DNS problem and put the link in URL queue.
i. Then repeat step 2.
j. End

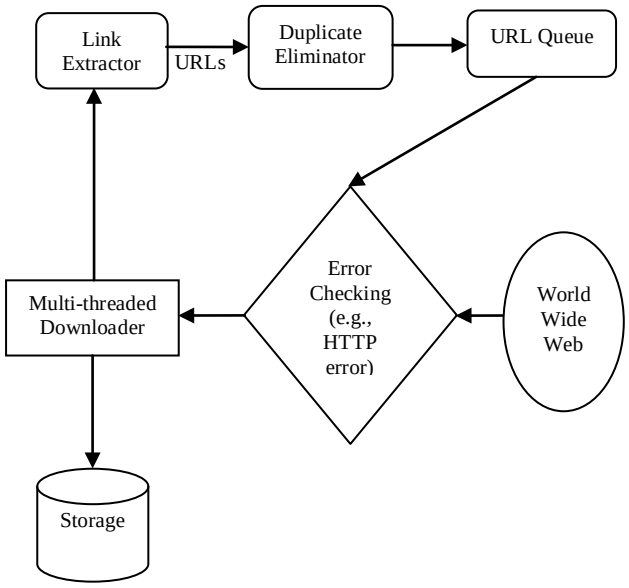

Figure 1: The basic crawler architecture

In figure 1 these threads may be running in a single process, or be distributed between multiple processes running at different nodes of a distributed system.

B. Pseudo Code for Basic Web Crawler:

Here's a pseudo code summary of the algorithm for a simple web crawler:

```
while not empty (the list of URLs to search)
{
        Take the first URL from the URLs queue
        If the URL protocol is not HTTP then
                break;
                go back to while
        If robots.txt file exist on site then
                If file includes “Disallow” statement then
                        break;
                        go back to while
        Fetch the URL
        If the opened URL is not HTML filethen
                break;
                Go back to while
        Save Page content in zip format
        Extract link from HTML file
        While not empty (all extract link)
        {
                If the link in URLs queue
                        break;
                        go back to while
                else
                        Push link in URLs queue
        }
}
```

The pseudo code mainly has two parts:

a. Specify the starting URL on web from which the crawler should start crawling.
b. Add the URL to the empty list of URLs or previous crawl URLs to search

C. Difficulties and Challenges:

It is fairly easy to build a slow crawler that downloads one or few pages per second for a short period of time, building a high-performance system that can download hundreds of millions of pages over several weeks presents a number of challenges in system design, I/O and network efficiency, and robustness and manageability [4]. Hence here arise some difficulties to solve the following challenges:

a. URL already Seen
b. Re-crawling pages for updates (freshness)
c. Efficient URL caching for web crawling
d. Detecting near duplicates for web crawling
e. Crawling Dynamic Pages and hidden web
f. Bandwidth management and DNS resolution
g. Politeness, Robustness, and Distribution Policy

D. Efficient URL caching for World Wide Web crawling:

If we represent web pages as nodes in a graph and hyperlinks as directed edges among these nodes, then crawling becomes a process known as graph traversal. Breadth-First is effective and efficient for crawler of broad search engine. However, crawling the web is not a trivial programming exercise but a serious algorithmic and system design challenge because of the following two reasons as described above:

a. The web is very large and the web has doubled every 9-12 months.
b. Web pages are changing rapidly. If we define “change” as “any change”, then a statistics we found that, about 40% of all web pages change weekly. [5] Even if we consider only pages that change by a third or more, about 7% of all web pages change weekly. [6] Caching is the idea of storing frequently used items from a slower memory in a faster memory. In the web crawler, since the number of visited URLs becomes too large to store in main memory, we store the collection of visited URLs on disk, and cache a small portion in main memory.

E. Detecting near duplicates for web crawling:

The quality of a web crawler increases if it can assess whether a newly crawled web page is a near-duplicate of a previously crawled web page or not.

a. Generic crawlers crawl documents and links belonging to a variety of topics
b. Focused crawlers use some specialized knowledge to limit the crawl to pages pertaining to specific topics. Documents that are exact duplicates of each other (due to mirroring and plagiarism) are easy to identify by standard check-summing techniques.

A system for detection of near-duplicate pages faces a number of challenges:

a. First is the issue of scale: search engines index billions of web-pages; this amounts to a multi-terabyte database.
b. Second, the crawl engine should be able to crawl billions of web-pages per day. So the decision to mark a newly-crawled page as a near-duplicate of an existing page should be made quickly.

c. Finally, the system should use as few machines as possible.

F. Bandwidth Management:

We managed the bandwidth by multithreaded system. We calculate every page download speed and total bandwidth. If the bandwidth is high then we increase the thread. By increasing and decreasing of thread we properly manage the full bandwidth.

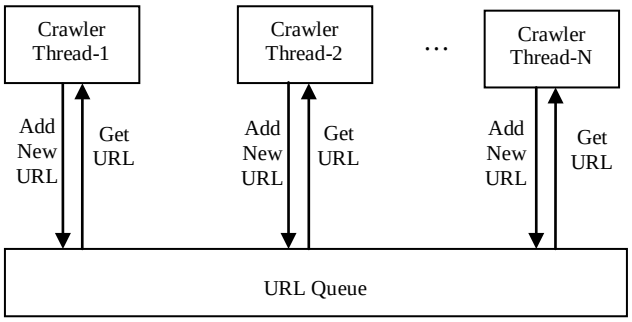

Figure 2: Crawler Thread.

II. FOCUSED CRAWLER FOR FINANCIAL DATA

Focused crawlers are programs designed to selectively retrieve web pages relevant to a specific domain for the use of domain-specific search engines and digital libraries, exploiting the graph structure of the web to move from page to page. Unlike the simple crawlers behind most general search engines which collect any reachable web pages in breadth-first order, focused crawlers try to "predict" whether or not a target URL is pointing to a relevant and high-quality web page before actually fetching the page.

There has been much research on algorithms designed to determine the quality of web pages. Our efficient focused crawler is made for collecting the financial data for a specific country.

The Technique to get the financial data for a specific country:

a. We have to determine for which country we are developing the system. Now in current world every country has a unique domain extension. For Bangladesh the domain name ends with '.bd', and for Canada '.ca' etc. From the domain name we can easily detect the country specific origin of the website.
b. Local web site may be ended up with '.com', '.org', '.net' etc. But from the domain name we can easily find the IP address and it is then geo-mapped to find out the country name.
c. If the website contains any finance related key word then we can easily categorize it as a financial website of any specific country. We have also used some specific keyword for understand the specific country and financial information.
d. Stats for 7th of September 2012 from http://www.domainworldwide.com there are near about 46,061,532 website. Our web crawler crawls a web page, gets the domain name, and then address is stored in a shared memory. This run-time shared memory helps us to check very quickly whether a domain link has already visited or not. If the shared memory overflows, we will have to use disk space for both storing and searching links for duplicate checking.

III. STRUCTURE OF FOCUSED CRAWLER

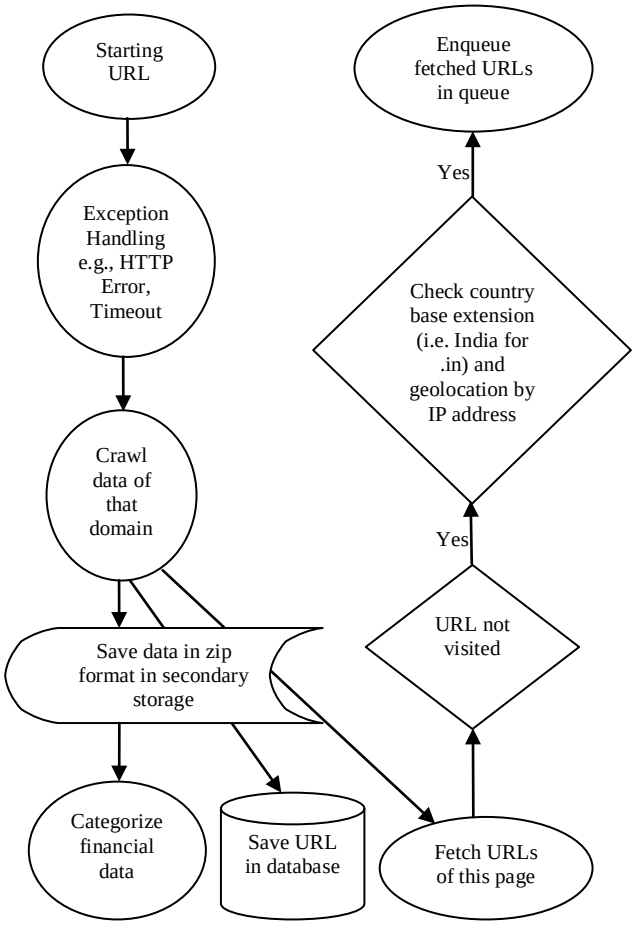

Figure 3: Focused crawler architecture

IV. CONCLUSIONS

We are working for long time to determine the trends in the industry, the needs of the customer, and the best way to address the customer satisfaction. We have crawled a very little portion of data from WWW because of low bandwidth of internet connection, and low configuration hardware. There are lots of challenges remained and we need to solve them completely for developing a complete large-scale focused web crawler.

V. REFERENCES

[1] http://en.wikipedia.org/wiki/Web_crawler

[2] C. Castillo, 2004, 'Effective Web Crawling'.PhD thesis, University of Chile,http://www.chato.cl/papers/crawling_thesis/effective_web_crawling.pdf

[3] Junghoo Cho, Hector Garcia-Molina andLawrence Page , 1998. 'Efficient crawling through URL ordering', Computer

Networks and ISDN Systems, Volume 30, Issue 1-7, 1998 pp. 161 - 172

[4] VladislavShkapenyuk and TorstenSuel, 2002, 'Design and Implementation of a High-Performance Distributed Web Crawler', http://cis.poly.edu/suel/papers/crawl.pdf

[5] J. Cho, H. Garcia-Molina, 2000,'The evolution of the web and implications for an incremental crawler', http://rose.cs.ucla.edu/~cho/papers/cho-evol.pdf

[6] D. Fetterly, M. Manasse, M. Najork, J. L. Wiener, 2003, 'A large-scale study of the evolution of web pages',Proceeding WWW '03 Proceedings of the 12th international conference on World Wide Web, pp.669 – 678