ARABELLA
A Directed Web Crawler

Pedro Lopes, Davide Pinto, David Campos and José Luís Oliveira
IEETA, Universidade de Aveiro, Campus Universitário de Santiago, 3810 – 193 Aveiro, Portugal

Keywords: Information retrieval, Web crawling, Crawler, Text processing, Multithread, Directed crawling.

Abstract: The Internet is becoming the primary source of knowledge. However, its disorganized evolution brought about an exponential increase in the amount of distributed, heterogeneous information. Web crawling engines were the first answer to ease the task of finding the desired information. Nevertheless, when one is searching for quality information related to a certain scientific domain, typical search engines like Google are not enough. This is the problem that directed crawlers try to solve. Arabella is a directed web crawler that navigates through a predefined set of domains searching for specific information. It includes text-processing capabilities that increase the system's flexibility and the number of documents that can be crawled: any structured document or REST web service can be processed. These complex processes do not harm overall system performance due to the multithreaded engine that was implemented, resulting in an efficient and scalable web crawler.

1 INTRODUCTION

Searching for knowledge and the desire to learn are ideals as old as mankind. With the advent of the World Wide Web, access to knowledge has become easier. The Internet is now one of the primary media regarding knowledge sharing. Online information is available everywhere, at any time and can be accessed by the entire world.

However, the ever growing amount of online information arises several issues. Some of the major problems are finding the desired information and sorting online content according to our needs. To overcome these problems we need some kind of information crawler that searches through web links and indexes the gathered information, allowing faster access to content. Web crawling history starts in 1994 with RBSE (Eichmann, 1994) and Pinkerton's work (Pinkerton, 1994) in the first two conferences on the World Wide Web. However, the most successful case of web crawling is, without a doubt, Google (Brin and Page, 1998): innovative page rankings algorithms and high performance crawling architecture turned Google into one of the most important companies on the World Wide Web.

However, 25 percent of web hyperlinks change weekly (Ntoulas et al., 2005) and web applications hide their content in dynamically generated addresses making crawling task more difficult and leveraging the need for novel web crawling techniques. Research areas include semantic developments (Berners-Lee et al., 2001), web ontology rules inclusion (Mukhopadhyay et al., 2007, Srinivasamurthy, 2004), text mining features (Lin et al., 2008) or directing crawling to specific content or domains (Menczer et al., 2001).

Our purpose with Arabella was to create a directed crawler that only navigates on a specific subset of hyperlinks, searching for information based on an initial subset of validation rules. The set of rules defines which links contain relevant and correct information and which therefore should be added to the final results. The rules also define a mechanism for creating dynamic URLs in real-time, according to information gathered in previously crawled pages, extending traditional web crawling operability. Arabella combines typical HTTP crawling features with a fast, easily extensible text processing engine that is capable of extracting nearly any structured information. This means that our tool is able to extract information from HTML pages, REST web services, CSV sheets or Excel books among others.

This paper is organized in 5 distinct sections. The following section provides a general overview of the web crawling subject and other relevant work done in the directed web crawling area. The third section presents the system architecture and we

Lopes P., Pinto D., Campos D. and Luís Oliveira J. (2009).
ARABELLA - A Directed Web Crawler.
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval*, pages 270-273
DOI: 10.5220/0002291602700273

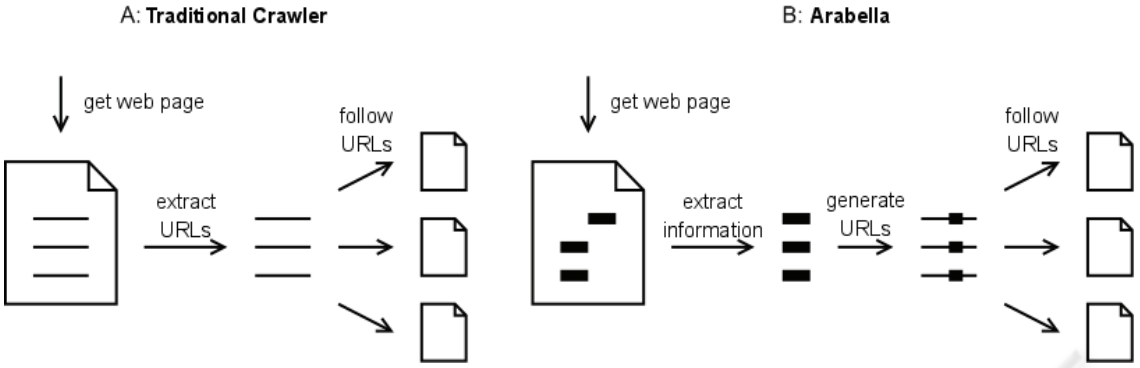

Figure 1: Comparison of traditional crawlers and Arabella.

include a usage scenario in section 4 where we improve DiseaseCard (Oliveira et al., 2004). Finally, there are conclusions and discussion about future developments.

2 WEB CRAWLING

Despite the fact that Google is, and will be, the king of search engines, web crawling has still evolved considerably in the last few years. The basic operation of a web crawler consists of a repeated cycle of solving website address using a DNS, connecting to the destination server and requesting the pages, receiving and processing the server response, advancing to the next link. DNS and web server requests are essential components and do not vary too much. Improvements are usually done on the page processing modules where several features can be added to extend the architecture.

Semantic web developments, brought by W3C, are one of the hottest research trends that web crawler developers have invested in. Structured and described content may ease the crawler tasks and will require changes in the basic crawler architectures. Swoogle (Ding et al., 2004) is a good example of research advances in this area.

Data mining is the subject of several information retrieval projects. Web crawling and the scope of data it can cover is the perfect ally to text mining tools. Whether research is focused on knowledge discovery (Chakrabarti, 2003) or on redesigning web crawling architectures to improve text mining results (Tripathy and Patra, 2008), the results show that this combination is successful.

The last mentionable topics are directed or oriented crawlers. These web crawlers (Miller and Bharat, 1998, Mukhopadhyay et al., 2007, Tsay et al., 2005) are related to a specific domain or research trend and are focused on result quality instead of result quantity. From Menczer's work (Menczer et al., 2001) we can conclude that this area can be enhanced and that there is space for further improvements. To solve the directed web crawler problem we need a tool to collect arbitrary text data from different websites, organized in a coherent structure. This tool should be easily configurable, fairly fast (Suel and Shkapenyuk, 2002) and should be presented in the form of a library, so that it can be easily integrated in any existing system. The problem with collecting arbitrary text data is that the same text can vary widely from site to site in terms of structure. Thus there was also a need for the tool to have basic text processing capabilities such as searching for regular expressions, cutting and splitting text, which could provide the necessary flexibility.

3 ARABELLA

Considering the features and flexibility we needed, we stated that none of the existing tools would satisfy our requirements. Despite their capabilities, these tools have the major drawback of being focused on a single task. Arabella closely integrates web crawling and text processing in a single Java library that allows full control over downloaded pages, saved content and the generation of new dynamic content. This feature enhances information retrieval tasks, improving the final result set and the overall quality of the application – Figure 1.

3.1 Motivation

With Arabella, we want to access content hidden in the deep web (Peisu et al., 2008), not limiting the crawlable space to web page referenced links. The majority of quality data is available online but is not directly referenced. To access this valuable information it is necessary to search specific databases, usually resorting to strict conditional

inputs. The URL to this information is dynamically generated in real-time, obstructing web crawlers' work – traditional web crawlers will not reach them - and increasing the complexity of the web crawling algorithms.

In the majority of situations, these dynamic URLs can be parameterized with sets of keys whose domains are either well known or easily generated. To achieve this important goal, we take advantage of Arabella's text processing capabilities. Using the Arabella engine we are able to define sets of keys that will be searched inside the crawled pages, enabling the conversion of these keys to other similar keys, and using them to generate new addresses, in real time, that will be added to the crawlable URL list – Figure 1-B. This is the opposite of traditional web crawler architectures - Figure 1-A – where the crawler reads a configuration file containing the URLs that will be processed and where new URLs will be found and added to the processing list.

3.2 Execution

Initially, the spider parses the configuration file and builds an internal data structure representing the web map. Task dependency is calculated - if task B depends on content produced by task A, B should not be executed until A finishes – and a thread manager is created that will ensure execution takes place as planned.

When the Spider starts, the thread manager initially launches all tasks that do not depend on any of the other tasks. When each task finishes, it signals termination to the thread manager, and the thread manager is then responsible for launching any ready-to-run tasks (those whose dependencies are already met). Each task is divided in two steps, retrieving the raw text, and processing the raw text. Text retrieval can be done in several distinct ways. Textual URLs are simple HTTP URL that can contain a placeholder for previously obtained keys – stored in a named container. Collected links are entire strings stored previously in a named container and will be added to the crawling cycle. Collected content is simple text that is gathered according to the rules defined in the configuration file and will also be stored in a named container.

When retrieving text from the web, requests to web servers are made asynchronously, in order to achieve a higher speed. However, crawling at high speeds can be considered impolite, so we have added customized control over interval between requests.

Each stop has only one location but may have more than one parallel action. This is extremely useful when categorizing information, since there can be more than one processing flow acting over the same piece of text, thus, collecting different pieces of information.

4 LOVD INTEGRATION

In order to provide an overview of the actual system functionality, we describe a real-world usage scenario involving the integration of several locus specific databases. This can be a cumbersome task in the field of bioinformatics due to the heterogeneity of adopted formats and the existence of a large number of distinct databases. The Leiden Open (source) Variation Database – LOVD (Fokkema et al., 2005) – is a successful attempt at creating a standard platform that can be easily deployed and customized by any research lab.

DiseaseCard's main objective is to offer centralized access to a vast amount of information sources focusing on rare diseases. With the main goal of improving DiseaseCard's quality, we have decided to increase its range by including access to several LOVD installations distributed through several distinct web locations.

Arabella comes into play as the tool that discovers which LOVD installation contains information regarding the genes that are associated with a given rare disease. We use it to select the index entries that contain the wanted gene, discard irrelevant information, and expand the URLs with the gene name in order to form the URLs for the variation pages.

We will use the PEX10 (human peroxisome biogenesis factor 10) gene for this example. We instantiate the Spider with the appropriate configuration file, put the gene name in the `gene` container and start the crawl. After the crawl we print the contents of the containers on the screen.

```
Container 'databaseRowsContainingGene':
   "2009-04-14" "2.0-16""13""870" "868"
        "268"   "dbPEX, PEX Gene Database"
      "http://www.medgen.mcgill.ca/dbpex/""P
      EX1,PEX10,PEX12,PEX13,PEX14,PEX16,PEX1
      9,PEX2,PEX26,PEX3,PEX5,PEX6,PEX7"
   "2009-04-01" "1.1.0-11"   "13" "804"
    "226"   "dbPEX, PEX Gene Database"
    "http://www.dbpex.org/"
    "PEX1,PEX2,PEX3,PEX5,PEX6,PEX7,PEX10,PEX
   12,PEX13,PEX14,PEX16,PEX19,PEX26"
```

5 CONCLUSIONS

Knowledge discovery and information retrieval are scientific areas greatly promoted by the evolution of

the Internet. With this evolution, online content quantity and quality has increased exponentially. Along with it, so has the difficulty in gathering the most suitable information related to a certain subject. Web crawling engines provide the basis for indexing information that can be accessed with a search engine. However, this approach is generic – the goal is to find content in the entire web. This means that finding valid information with certified quality for a certain topic can be a complex task.

To overcome this problem, there is a new wave of web crawling engines that are directed to a specific scientific domain and can reach content hidden in the deep web. Arabella is a directed web crawler that focuses on scalability, extensibility and flexibility. Considering that the crawling engine is based on an XML-defined navigational map, the engine is scalable as the map can contain any number of stops and rules. Rule and stop dependencies can be added, as well as multiple rules for the same address. Arabella is a multithreaded environment that is prepared to deal with these situations. Extensibility is also a key feature of this application. Its modular approach and small code base allow the quick addition of new features or the reconfiguration of existing ones.

However, the most important feature is flexibility. Interleaving web requests with text processing allows the execution of a set of important functionalities. Arabella is a web crawler that allows dynamic generation of new URL addresses in real time and storage of gathered information for further use in the execution cycle. With proper configuration, Arabella can also crawl through any type of online accessible document whether is HTML, CSV, XML, REST web service, tab delimited or simple text.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nº 200754 - the GEN2PHEN project.

REFERENCES

Berners-Lee, T., Hendler, J. & Lassila, O. (2001) The Semantic Web. Sci Am, 284, 34-43.

Brin, S. & Page, L. (1998) The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems, 30, 107-117.

Chakrabarti, S. (2003) Mining the Web: Discovering Knowledge from Hypertext Data, Morgan Kauffman.

Ding, L., Finin, T., et al. (2004) Swoogle: A Search and Metadata Engine for the Semantic Web. Language, 652-659.

Eichmann, D. (1994) The RBSE Spider -Balancing Effective Search Against Web Load. Proceedings of the First International World Wide Web Conference. Geneva, Switzerland.

Fokkema, I. F., Den Dunnen, J. T. & Taschner, P. E. (2005) LOVD: easy creation of a locus-specific sequence variation database using an "LSDB-in-a-box" approach. Human Mutation, 26, 63-68.

Lin, S., Li, Y.-M. & Li, Q.-C. (2008) Information Mining System Design and Implementation Based on Web Crawler. Science, 1-5.

Menczer, F., Pant, G., et al. (2001) Evaluating Topic-Driven Web Crawlers. Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New York, NY, USA, ACM.

Miller, R. C. & Bharat, K. (1998) SPHINX: a framework for creating personal, site-specific Web crawlers. Computer Networks and ISDN Systems, 30, 119-130.

Mukhopadhyay, D., Biswas, A. & Sinha, S. (2007) A new approach to design domain specific ontology based web crawler. 10th International Conference on Information Technology (ICIT 2007), 289-291.

Ntoulas, A., Zerfos, P. & Cho, J. (2005) Downloading textual hidden web content through keyword queries. JCDL '05: Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries. New York, NY, USA.

Oliveira, J. L., Dias, G. M. S., et al. (2004) DiseaseCard: A Web-based Tool for the Collaborative Integration of Genetic and Medical Information. Proceedings of the 5th International Symposium on Biological and Medical Data Analysis, ISBMDA 2004. Barcelona, Spain, Springer.

Peisu, X., Ke, T. & Qinzhen, H. (2008) A Framework of Deep Web Crawler. 27th Chinese Control Conference. China, IEEE.

Pinkerton, B. (1994) Finding what people want: Experiences with the WebCrawler. Proceedings of the Second International World Wide Web Conference.

Srinivasamurthy, K. (2004) Ontology-based Web Crawler. Computing, 4-8.

Suel, T. & Shkapenyuk, V. (2002) Design and Implementation of a High-Performance Distributed Web Crawler. World Wide Web Internet And Web Information Systems.

Tripathy, A. & Patra, P. K. (2008) A web mining architectural model of distributed crawler for internet searches using PageRank algorithm. 2008 IEEE Asia-Pacific Services Computing Conference, 513-518.

Tsay, J.-J., Shih, C.-Y. & Wu, B.-L. (2005) AuToCrawler:
An Integrated System for Automatic Topical Crawler. Machine Learning.