Issues Affecting the Extraction of Data from the Web

S.Butt and A.Phippen

Network Research Group, University of Plymouth, Plymout, United Kingdom
e-mail: info@network-research-group.org

Abstract

Much of the data found on the Web is of some value, especially data found on company websites. Therefore, the collection and storage of such data would provide a valuable resource. Due to the sheer volume of data it is impractical to expect a human being to be capable of accurately collecting it through browsing the Web. A solution to this problem is to automate the task of data extraction. Unfortunately, differing standards in the quality of documents on the Web restrict the amount of data retrieved and the accuracy of an automated process. This paper examines the types of data that may be required to be, and can be, extracted from a web page, as well as issues affecting the accuracy of data extraction. It is then suggested that the use of standard document syntax and structure, and the use of self descriptive elements, to aid the automated data extraction process may improve information flow between businesses on the Web.

Keywords

Data extraction, spidering, semantic web.

1. **Introduction**

The Web represents a vast repository of data. This data is presented to users of the Web in the form of web pages generated using HTML. How the data is represented within the source HTML file is generally of little or no concern to the author of the page as long as it can be interpreted correctly by a web browser and, in turn, understood by the end user. Whilst this approach to data presentation is adequate in terms of making the data human readable, it can present difficulties when attempting to perform automated data extraction involving little or no human interaction.

It can be assumed that much of the data found within a web page will be of interest to someone. In some cases this data will also have some value. Most companies nowadays have their own websites, which contain, at the very least, some information regarding the nature of their work, and contact details. Such information is of value as it relates to business – the company itself uses this data to acquire business, but it could equally be used by others (e.g. competitors, industry analysts) as it is in the public domain. The volume of data found within such sites can be great; therefore a human based approach to data extraction would prove time consuming (as well as being prone to simple human error). An automated method of efficiently extracting data from websites – providing information of value to its user – would have a value proportional to that of the data itself. Unfortunately, HTML is used to describe the layout and appearance of data for viewing through a browser and does not strictly enforce rules regarding syntax, which could make an HTML document more meaningful to a computer. This can seriously hinder attempts to automate the extraction of data.

This paper seeks to highlight the major issues associated with automating the data extraction process based upon the results of practical research into the area. It is largely concerned with the value of data extraction to businesses and the discussion presented here focuses on this specific subset of WWW users. Following on from this discussion it presents ideas for means with which problems in this area can be overcome, measures that can be taken by companies to facilitate business to business data sharing and the value of such measures.

2. The purpose of data extraction

The perceived value of data extracted from the Web will differ depending upon the perspective from which it is viewed. One company may be interested in acquiring contact details for potential customers, another company may be interested in profiling the types of technologies used within certain websites (e.g. CGI, applets, Flash, etc). It is also possible that certain information could be inferred from data extracted from the Web. For example, information regarding the type of server being used to host websites could be used to build an independent model of server usage. What data has value cannot be stated here, as it depends largely upon the aims of those who seek to exploit the data found on the Web. It will be assumed, therefore, that all data held within a website is potentially of value to someone. The following will centre on the extraction of data in general, with no specific focus on the usage of extracted data (although, examples may be used to illustrate the potential of the methods discussed herein).

Currently, there are a number of projects concerned with the topic of data extraction from the Web (Laender et al, 2001; Chen et al, 2001; Lage et al, 2004; Baumgartner et al, 2005; Myllimaki, 2002). Many of these focus on correcting the structure of Web documents and/or the retrieval of data from within the deep Web. The deep Web is a term used to refer to those parts of the Web that are not easily accessible using simple spidering (simply retrieving interlinked, static web pages – also referred to as crawling). Generally speaking these are dynamic web pages whose content is based upon user input, the contents of a back end data store or the results of server side processing. The contents of the deep Web are outside the scope of this discussion. However, the issues discussed within this paper may provide a basis for research into the value of data extraction from the deep Web.

All of the projects looking into the topic of automated data extraction on the Web – including that which was carried out as part of the research for this paper – encounter the same problems. These problems are a result of the inconsistency, across the Web, of the presentation of data within websites.

3. An automated process

A simple way of extracting data from the Web is for a human user to navigate websites page by page, viewing the source code for each, noting down each piece of valuable data that they come across. The obvious problems with this form of data extraction are the speed with which the process can be performed by a human and simple human error. It is safe to assume that a percentage of data within each website is likely to be either missed, or incorrectly recorded – resulting in the need for an additional process that validates the extracted data. This could also be a human process, or it could be automated, either way it increases the time and resources used to perform a single task. The value of the data extracted may be outweighed by the cost of actually performing the data extraction. Automating this process would lead to a significant improvement in performance and accuracy, and would allow for more efficient use of the human user's time.

3.1 The DataChecker system

For the purposes of this research a simple data extraction system was developed that was used to study the type of data that could be retrieved from the Web and the ease with which this could be achieved. This system, called DataChecker, was designed to accept a list of URLs pointing to websites from which data was to be extracted. The system navigates through each site (to a certain depth, so as to not get lost within large sites, or stuck in a loop of interlinked pages) collecting data as it goes. It is composed of two main components: the spider component is provided with URLs, which it follows, retrieving the HTML at each location. The extractor component takes the HTML returned by the spider and extracts data from it based on a file defining what should be extracted. Any URLs relative to the website currently being spidered are passed from extractor to spider, so perpetuating a retrieval/extraction cycle that runs as long as there are pages to retrieve within the site (or DataChecker reaches a predefined page limit).

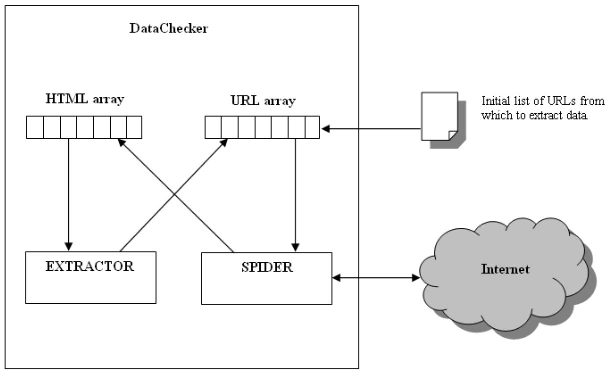

Figure 1: Simplified workings of the DataChecker system.

When provided with a list of URLs DataChecker splits them between a number of arrays. Each array is then associated with a spider and an extractor (as is an empty array that is eventually to be populated with the HTML returned by the spider). Each spider and extractor, operating on separate threads of execution, constantly checks the status of the two arrays (the spider checking the URL array and the extractor the HTML array) to see if they have any content. As soon as content is found it is then processed in the relevant fashion (URLs used to retrieve HTML, HTML used to extract data).

The retrieval of data by the spider is a simple task – the spider is essentially dumb, leaving the majority of the processing to the extractor. In the DataChecker system data is extracted from HTML using regular expressions. These are used to extract two types of data – that which is actually present within the HTML code (e.g. Hyperlinks and email addresses), and that which can be inferred from the code, such as whether or not a scripting language is being used (the extractor does not extract the data straight from the HTML, rather a Boolean value is generated based upon its existence). As the HTML retrieved by the spider can be represented as one long string of text, regular expressions are used to find matches for certain types of data within the HTML. A regular expression can be used to verify the existence of a certain string pattern – that is, a collection of specified characters arranged in a specific fashion – within a certain string. It can also record the matched substring(s) – allowing for future analysis. In this way the regular expression can be used to extract both types of data mentioned above. To extract a certain piece of data from a website simply requires a regular expression that will correctly match every occurrence of that data on every web page that is processed. This proves much more difficult than the previous sentence makes it appear.

```
<\s*a.*(href)\s*=\s*(?:\"(?<1>[^\"]*)\"|(?<1>\S+)).*>
```

Figure 2: An example regular expression - extracts links from within the anchor (a) tag.

When developing DataChecker an alternative method to regular expressions was considered. Due to the similarity of HTML and XML (both of which are derived from SGML) it was initially decided that the code retrieved by the spider would be treated as XML. In this way it would be possible to make use of XSLT to extract data from the HTML. Unfortunately, the same problem that would also affect extraction through the use of regular expressions – which will be discussed shortly – meant that this method was not viable (without significant extra processing of the HTML).

As mentioned earlier, the rules governing the syntax of an HTML document are loose and not as strictly defined as XML. The W3 Consortium has defined exactly how to build a correctly formatted XML document (Bray et al, 2004). This standard way in which an XML document is built ensures that attempts to find a piece of data, defined by a certain tag within a well formed XML document will not be hampered by inconsistencies in the style or content of the code. As HTML does not have such strict restrictions placed upon it one website developer may present a piece of data in one way, whilst another developer presents it in a slightly different way. The difference between the two methods may not be great, but the fact that there is a difference makes extraction of that data more difficult. The less strict definition of how HTML documents are constructed can also lead to errors within the code that may be missed by the developer, but would lead to difficulties when attempting to extract data from it using a system such as DataChecker. The sheer number of web pages on the Internet means that the probability of errors within a selection of retrieved web page is going to be high. Such errors and differences in the presentation of data create difficulties for the extractor. If attempting to treat the HTML as XML, the majority of pages will be classed as incorrectly formatted and will not get processed correctly. This problem could be overcome by pre-processing the HTML to ensure that it adheres to the XML standard (turning it from HTML into XHTML), but this would increase the processing overhead for the system. Regular expressions can be used on every page returned by the spider, but to extract the correct data the expression needs to match all variations on how that data can be presented. It also needs to be robust enough to recognise errors in the code, but still extract the correct data. Such regular expressions would be extremely complex and hard to define. A certain percentage of data within websites would not be matched due to errors in the HTML.

Website developers creating sites that adhered to the XHTML standard (W3C HTML Working Group, 2002) would go some way towards solving the problems outlined above. However, it is not practical to assume that all developers would change their ways just to accommodate those wishing to perform automated data extraction. Many companies, though, could benefit from a standard method of presenting information about themselves and their products that would allow for easy extraction and processing.

3.2 Business to business data interchange

A great many companies take advantage of some form of EDI (Electronic Data Interchange) to do business. Vast amounts of business data are exchanged electronically, often with little or no human interaction. Such methods of doing business can increase productivity, decreased administrative overheads and lead to more efficient business models. Increasingly, the Internet is being employed to facilitate this exchange of information. Already, many businesses have their own websites – just another way in which information regarding the company and its business can be relayed via the Internet. This information, whilst delivered electronically, is largely consumed by the human computer user – prone to mistakes and slow to digest the data contained within the websites. By utilising automated data extraction processes and providing company data in a standardised format, this channel of data delivery could be used to provide more information than can be seen when simply viewing a web page in a browser.

Improving the consistency of company websites through the use of a mark-up language with strict rules on how the document is structured may help in attempting to allow computers to autonomously extract data. However, this only goes some way towards solving the problems experienced in the course of this research. Fewer errors in a web page's mark-up code will improve a data extractor’s ability to read data from a document, but, for some data there are a number of different ways in which they can be defined within a page. For example, a hyperlink can appear within a number of different tags (e.g. The 'A' tag, the 'AREA' tag, the 'IMG' tag).

Therefore, a data extractor needs to check for every different way in which a piece of data can be defined – adding to the processing overhead. Also, some data of value may have no standard way of being described. This data may be within the main body of text of the web page, or it may be within an image and therefore unable to be extracted by a text based system. An example of such a piece of data is the name of the website's developer. There is no standard way of including this information within a web page, but it could be of use to someone wishing to find out which websites were developed by which developer. The free flow of information between companies through their websites is hampered by problems such as this. To overcome these problems requires changes to the way in which websites are viewed (in terms of an information delivery media) and described at the level of the markup language.

4. Semantic issues

Berners-Lee et al, 2001 describes what is being hailed as the next major evolution of the Web. The Semantic Web is seen as a way for the vast amounts of data present on the Web to be made meaningful to computers, not just users. Currently, computers have very little idea about the nature of the data held on the Web, they simply retrieve data as instructed and present it to the user to interpret. In this way the search for specific data can be a frustratingly inefficient one, with the computers of the Web providing little in the way of assistance beyond acting as a data delivery service. What the Semantic Web proposes is a way in which the data held by the Web can be given meaning, which computers will understand. This should, theoretically, allow much of the work involved in the search for data to be carried out by computers rather than their users, with machines being able to make informed judgements about relevant data based upon a certain set of rules.

One of the key issues in the Semantic Web is the use of meta data – data used to describe a resource on the Web (i.e. a web page). The use of meta data to describe the contents of a web page would, in theory, allow the computer to understand, to a certain extent, those contents, as well as the user. The idea of a descriptive language used to facilitate some form of automated processing of web pages is of interest with regards to this research. A web page containing meta data, guiding the extraction of valuable data could improve the performance of automated data extractors and help to eliminate the problems encountered in the course of this research. An example of how this could be implemented would be by using additional mark-up tags to highlight data for extraction. Information regarding each individual item of data could be provided within the tags, either in the tag name or as an attribute. Such information could describe, in textual form, data that is presented in a form that is not extractable, e.g. an address presented as an image. The web page would then become a hybrid document containing both HTML specific information for describing the presentation of the document, and information provided specifically for use by applications seeking to extract data from the document. Such information would have no effect on the presentation of the web page, therefore its appearance would go unchanged and the human user would be unaware of any changes to the content of the document. Highlighting the data that can be extracted could also have the benefit of restricting the extraction of data – the data that the author of a website does not want extracting would not be highlighted in such a way (or would be highlighted as restricted). Whether or not an extractor ignores these restrictions is entirely down to the author of the extraction software – it is simply a matter of etiquette. This is not entirely unlike the situation with the current method of restricting access to resources on the Web using a robots.txt file (Koster, 1996).

A standard would be required to ensure that the description of extractable data was consistent across the Web and all data extractors would be capable of understanding what data was of value to it. The adoption of such a standard within the business world would provide greater integration between business processes and applications, and the publicly visibly data provided by company websites.

5. Conclusion

Given the vast amount of data available on the Web, automating the data extraction process is the only way to harvest large collections of it quickly and efficiently. Of the many software solutions already available to perform such a task, all must contend with the inconsistencies and errors inherent within Web documents. Many attempt to overcome these problems by pre-processing pages before extracting data – adding to the processing overhead.

Rather than the singular approach of trying to fit the application to the resources, a more collaborative effort to improve the contents of documents on the Web could lead to a greater flow of information. A standard already exists (XHTML) that, if adopted by website developers, would increase the consistency of the structure of web page data and, therefore, help to improve the performance of data extraction software. Coupling improved document structure with descriptive elements, that provide meaningful information for applications processing the document, would create an environment in which automated data extraction applications could thrive.

It is impractical to expect all web developers to alter their development practices to accommodate the changes outlined above. However, such changes could provide benefits to businesses. By providing an easy way in which to extract data relating to a company and its business from the company's website, the company could improve its visibility and draw the attention of interested parties. The flow of information between businesses, and between business and customer, would increase as the speed and efficiency of the automated data extraction process did. Accurate repositories of company data could grow, fed by data extractors, acting as directories which could be queried in order to find information that fulfils the customer's needs. Many uses could be found for the data extracted from company websites.

As stated earlier in this paper, any data found within a website may have some value. Improving the means by which data of value can be acquired could benefit the owner of the data as well as the seeker of the data. Further research into the areas identified within this paper could facilitate such improvements and lead to improvements in the description, and clarity, of documents on the Web.

6. References

Baumgartner R., Frölich O., Gotlob G. (2005). "Web Data Extraction for Business Intelligence: the Lixto Approach" [online]. Available http://www.dbai.tuwien.ac.at/proj/lixto/WebBI.pdf [Accessed 1st September 2005].

Berners-Lee T., Hendler J., Lassila O. (2001). "The Semantic Web". *Scientific American*. May 2001.

Bray T., Paoli J., Sperberg-McQueen C.M., Maler E., Yergeau F., Cowan J. (2004). "Extensible Markup Language (XML) 1.1". *W3C Recommendation* [online]. Available http://www.w3.org/TR/2004/REC-xml11-20040204/ [Accessed 22nd August 2005].

Chen H., Chau M. Zeng D. (2001). "CI Spider: a tool for competitive intelligence on the Web". *Decision Support Systems*. Vol. 34, No. 1, pp. 1-17.

Laender A.H.F., Ribeiro-Neto B., da Silva A.S. (2002). "DEByE – Data Extraction by Example". *Data and Knowledge Engineering*. Vol. 40, No. 2, pp. 121-154.

Lage J.P., da Silva A.S., Golgher P.B., Laender A.H.F. (2004). "Automatic generation of agents for collecting hidden Web pages for data extraction". *Data and Knowledge Engineering*. Vol. 49, No. 2, pp. 177-196.

Koster M. (1996). "Evaluation of the Standard for Robots Exclusion". *The Web Robots Pages* [online]. Available http://www.robotstxt.org/wc/eval.html [Accessed 22nd August 2005].

W3C HTML Working Group (2002). "XHTML 1.0 The Extensible HyperText Markup Language (Second Edition)". *W3C Recommendation* [online]. Available http://www.w3.org/TR/xhtml1/ [Accessed 12th September 2005].