Automatic Information Extraction from Web Pages

Budi Rahardjo, Roland H.C. Yap[1]
School of Computing
National University of Singapore
Republic of Singapore
{budiraha,ryap}@comp.nus.edu.sg

ABSTRACT
Many web pages have implicit structure. In this paper, we show the feasibility of automatically extracting data from web pages by using approximate matching techniques. This can be applied to generate automatic wrappers or to notify/display web page differences, web page change monitoring, etc.

Keywords
Wrappers, semi-structured data, Internet, tracking changes.

1. INTRODUCTION
The Web is now probably the largest sharable and searchable repository of information. Given the overwhelming amount of information contained on web pages and because a large portion of this content is not only dynamic but also changing regularly, it is useful to have effective ways of extracting just the relevant portions of information contained in that page of interest to a user. In a notification context, a user may want to be notified when some relevant portion of a webpage changes (perhaps due to an update), and furthermore either be able to see the relevant changes highlighted or to extract just those changes. A related scenario is that given a collection of webpages with a similar intrinsic structure, one may want to have a tool which can automatically extract particular portions of the webpage.

This problem is essentially one of how to extract data from a web page. The most common approach is to use some wrapper technology [1,5,7]. We focus on the data extraction aspect rather general semi-structured database context for wrappers. Although wrappers provide an effective way to data extraction from Web sources, they require high level of expertise and they are difficult to maintain and may not be robust. Another approach is based on Natural Language Processing (NLP) [9]. However, extensive corpus training and a rich grammatical knowledgebase is required.

In this paper, our goal is an approach which can be effective for the "naive" user, hence, as automatic as possible. It treats both the display of differences and extraction of data as a common problem. It works well with typical web pages of the order 10-100k.

2. EXTRACTION USING APPROXIMATE MATCHING
We make the basic assumption that web pages contain some implicit structure and are not just random HTML documents. Furthermore, where there are many textually similar pages, or when a web page changes, it is the data in the differences which may contain relevant information to be extracted. Thus automatic extraction is possible simply by finding a good method of identifying the portions of the web page which are different. Ultimately the user will still need to select which of those differences are relevant for extraction purposes. Those selected portions can then be highlighted for display, for example in the side-by-side display in Figure 1.

Our approach to finding the textual similarities and differences between web pages is to use minimum string edit distance to find the portions of the HTML which are inserted, deleted, substituted and conserved between two web pages. Instead of treating the whole web page as one string, we tokenize the HTML by treating the text between starting and ending tags as a single token, called here a *segment*. This recognizes that HTML tags are usually part of the structure of the document and form a skeleton where the rest of the text lies between. The initial segments give a gross characterization of the document. The advantage here is that it focuses on differences between HTML tags and makes it easier to deal with large web pages since tokens in string edit distance are segments. A finer grained analysis is performed after the initial segment differences are found. To deal with the details of the segments, we again use minimum string edit distance but here the words (space/punctuation delimited strings) in the segments are the tokens. However, it only makes sense to do the finer grained comparison when the segments are sufficiently similar. Segments where the edit distance is large are treated as being totally different, otherwise spurious results may occur.

The highlighted display in Figure 1 shows precisely the differences found in the initial comparison (we call this the *initial extraction*) between the two pages from CNNfn, i.e, the company name and stock statistics. Not all the different segments found are necessarily relevant to the end-user. Conversely, the initial extraction can be viewed as a training set for generating the wrapper but it may not be representative of all the data elements desired. In both cases, the user can either manually disable some of the highlighted segments from being extracted, or highlight additional segments to extract. It is also possible here to use a larger training set of more than two web pages, and define the data to be extracted with respect to a combination of the union and intersection of the different segments. After this process, we

[1] The Logistics Institute Asia-Pacific.

have enough information to automatically generate a custom wrapper.

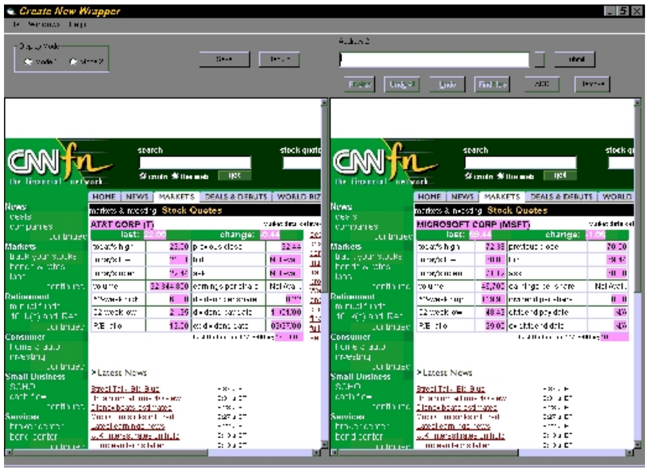

Figure 1: Initial Extraction Result

The custom wrapper produced is simply a segmented web page, up to two levels, where segments can be marked for extraction. The wrapper can then be executed against another web page, implicitly one with a similar structure. The data to extract is obtained from the text in the marked segments by recomputing the minimum segment edit distance with the new desired web page.

3. PRELIMINARY RESULTS

A prototype system shown in Figure 1 incorporating these ideas has been implemented in Visual Basic 6.0. Visual Basic is convenient as it has a web browser control that is integrated with other components, which is convenient for implementing the direct side-by-side user interface illustrated in Figure 1. We have experimented with many semi-structured web sources such as Amazon, CNNfn, AltaVista, online news etc. Most of these pages are quite large and complex in structure with quite a lot of variation across pages. We have found that the simple approach here still works quite well in automatically identifying most, if not all of the information, which one would typically want to extract from such web sites. Of course, a completely automatic procedure is not possible. We show however that most of it can be done by the initial extraction and the GUI makes it trivial to customize the additional relevant portions. A table of typical extraction/wrapper execution timings is given in Table 1. For most pages, the average execution time is on the order of 1-2 seconds on a Pentium II/350. The execution time is quite acceptable given that the prototype is running in Visual Basic. It will be substantially faster in C.

	CNNfn	Amazon	AltaVista	MSN	Infoseek
Size(Kb)	37	64	29	49	35
Time(Sec)	0.9	2.5	1.5	1.3	1.0

Table 1: Wrapper execution time for various web sources

4. DISCUSSION AND CONCLUSION

The motivation of this work was to investigate the question how well can string edit distance based wrappers work on typical web pages. The goal is that the method should be as automatic as possible while being usable by naive users, i.e. users should not have to write wrapper descriptions or Perl scripts. Furthermore, the wrappers obtained should work well on the targeted page and that the execution speed be reasonable. We believe that our approach here succeeds on all these counts. While one can have more sophisticated segment processing, the heuristics used here already work well on typical web pages.

The most relevant work is HTMLDiff [2] and TopBlend [4] both of which are tools for visualizing the differences between webpages. HTMLDiff compares pages using longest common subsequence gives a merged view of two webpages with the differences highlighted. TopBlend which uses heaviest common subsequence also provides a side-by-side view similar to Figure 1. The main difference from the work here is that their emphasis is on visualizing differences, while we are focused on automatic extraction of information and creating wrappers. A different approach is to learn wrappers from examples, such as wrapper induction [5]. Here we show that a string matching process works well in practice without needing more assumptions.

Many applications which utilize the simple wrappers created here can be built. One is an information gathering application where given a set of URLs in a web page, the system extracts and offers a unified view of extracted information from relevant webpages. This is useful for example with news and report web sites. Another application is a toolkit for web monitoring, which is more effective than more simplistic notification services such as NetMind [6] which only informs user that the web page has changed and requires that the user specify at least 8 words to monitor on the page of interest. Here we can focus on the specific segments of interest which is not only more precise but also easier to use.

5. REFERENCES

[1] N. Ashish and C.A. Knoblock. Semi-automatic wrapper generation for internet information sources. In *Proc. Second IFCIS Conf. on Cooperative Information Systems*, 1997.

[2] T. Ball and F. Douglis. Tracking and viewing changes on the Web. In *Proc. 1996 USENIX Technical Conf.*

[3] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. D. Ullman, and J. Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *IPSJ Conf.*, pp. 7-18, 1994.

[4] Yih-Farn Chen, et al. TopBlend: An Efficient Implementation of HtmlDiff in Java, AT&T Labs - Research Tech. Report 00.5.2., 2000.

[5] I. Muslea, S. Minton and C.A. Knoblock. Hierarchical Wrapper Induction for Semistructured Information. In *Proc. of Intl. Conf. on Autonomous Agents*, 1999.

[6] NetMind. http://www.netmind.com.

[7] A. Sahuguet and F. Azavant. WysiWyg Web Wrapper Factory (W4F). *Proc. of WWW Conference*, 1999.

[8] D. Smith and M. Lopez. Information extraction for semi-structured documents. In *Proc. of 1st Workshop on Management of Semistructured Data*, 1997.

[9] S. Soderland. Learning to Extract Text-based Information from the WWW. In *Proc. of the Third Intl Conf. on Knowledge Discovery and Data Mining* - KDD-97, 1997.