WEB DATA EXTRACTION SYSTEM USING PATH SEARCH ALGORITHM

**AMIT SING, ABHIJEET H. BHOLE,
VINAY SINGH, BUDDHABHUSHAN I. SANGOLE, JYOTI P. KSHIRSAGAR**

Abstract: Web is a huge reservoir of information. Data available is extremely diversified and abundant. To search for specific information, the user has to go through many pages of the Internet, filter the data and download related documents and files. This task of searching and downloading is time consuming. Web pages are in unstructured HTML format. There is a necessity to convert unstructured HTML format into a new structured format such as XML or XHTML. We propose an approach for implementing web data extraction and developing a Robust Web Data Extractor from HTML web pages. The various stages of building the system are Data Retrieval, Data Source Modeling, Data Cleaning/Filtering, Data Integration and Data Transformation. The data modeling stage renders Document Object Model (DOM) tree with the help of HTML Parser. Algorithms and rules are used to specifically analyze the HTML tags and extract the data. Furthermore, our system enables the user to perform his task without the need to write a script or program or even without any knowledge of computer programming. The system created will help in the decision making process, which is the prima facie requirement for success in corporate or social world.

Keywords: DOM tree, HTML, Web Data Extraction, XML.

Introduction: Today's web content is extremely diversified and abundant. Searching for the required information out of this ocean of data is a real difficult task. Internet is a powerful source of information. Most business applications depend on web to collect information that is crucial for decision making process. By analyzing web one can identify market trends, price details, product specification etc. Manual data extraction is time consuming and error prone. In this context automatic web data extraction plays an important role. Websites are usually designed for visualization not for data exchange. All pages of same website will be well designed. They may follow same template. Templates can be used to display objects of same type. Web page construction is the process of combining data to templates[1]. Web data extraction is the reverse process of page generation. If one page is given as input extraction target will be record level information The Document Object Modelling Tree is constructed using a HTML parser[2]. This tree is used for pattern and structure matching. Further, the information needs to be transformed. This is done using filtering, cleaning, transforming and integrating. The software uses scheduling techniques so that the web pages can be used repetitively. The result can be further transformed and can be send as SMS alert or emails.

Earlier Systems: In many application areas there has been a need to automatically extract relevant data from HTML sources and translate this data into a structured format, e.g., XML, or into a suitable relational database format. A first and obvious solution was the evolution from screen scrapers to web scrapers, which can navigate to web pages and extract textual content. However, web scrapers usually lack the logic necessary to define highly structured output data as opposed to merely text chunks or textual snippets. Moreover, they are usually unable to collect and integrate data from different related sources, to define extraction patterns that remain stable in case of minor layout changes, to transform the extracted data into desired output formats, and to deliver the refined data into different kinds of applications. For this reason, specific research on web data extraction was needed and several academic projects and some commercial research projects on web data extraction were initiated. While the academic projects such as XWRAP [10], Lixto [7], Wargo [12], and the commercial systems RoboMaker of Kapow technologies1 and WebQL of QL2 Software2 focused on methods of strongly supervised "semi-automatic" wrapper generation, providing a wrapper designer with visual and interactive support for declaring extraction and formatting patterns, other projects were based on machine learning techniques. For example, WIEN [8], Stalker [11], and DEByE [9] focused on automatic wrapper induction from annotated examples. Some other projects focused on specific issues such as automatic generation of structured data sets from web pages [6] and particular aspects of navigating and interacting with web pages [5].

Architectural Points of View, the system user gives the URL as input from where the data is to be extract. The respective web pages are downloaded and parsed through SAX and DOM[13] parser to generate a DOM tree. The data which is to be extracted is given by the user.

System Architecture:

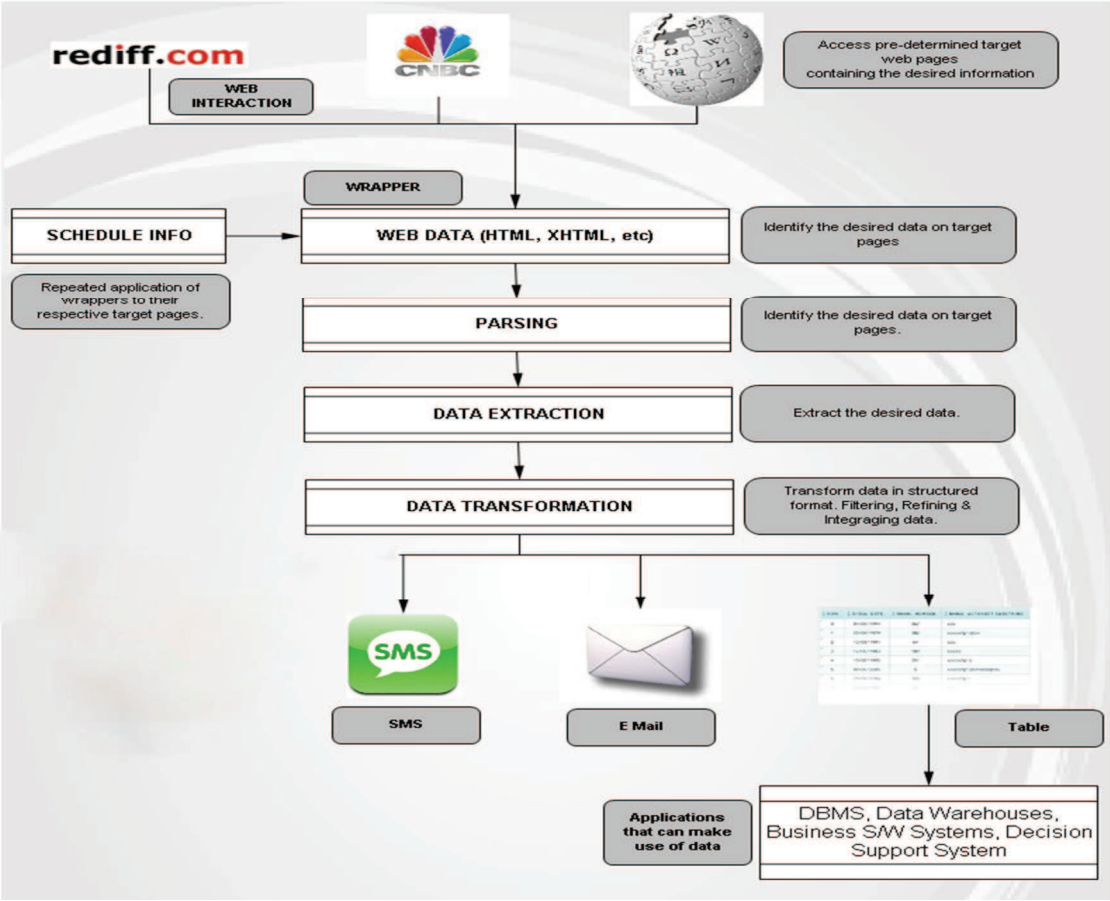

Figure 1.System Architecture

The data is searched in the DOM tree[13] using DFS to locate the path of the data. The URL, path and filter set is saved as an extractor set. The extractor set is scheduled to extract the data in accordance with user requirement. After loading the extractor the data extraction process get start and the required data is extracted. The extracted data is saved and transformed into the message, e-mail or table format according to user need.

Block Diagram: The proposed system is divided into following modules:

Module 1: Download and Parse

Module 2: Path based Web Data Extraction

Module 3: Tree Comparison Based Web Data Extraction

Module 4: Transform and Export

Module 1: Download and Parse

Input: URL of desired web page.

Output: DOM Tree.

Description:

- User provides the URL of the web page where desired web data is available.
- The source code of these web pages is downloaded.
- It is then parsed with SAX parser and Filtered.
- After that it is parsed with DOM parser to generate the DOM tree.

Module 2: Path based Web Data Extraction

Input: DOM Tree and Sample Data.

Output: User specific data.

Description:

- User provides the sample data of current instance.
- This Data is searched within the DOM tree using DFS algorithm.
- When match is found that path is saved in extractor set.
- Now the extractor set contains URL, Filter set, Path.
- This extractor is loaded and passed to module 1.
- Module 1 generates DOM tree of current instance.
- In DOM tree the saved path is followed and required data is extracted.

Module 3: Tree Comparison Based Web Data Extraction

Input: DOM Tree.

Output: Changed Data

Description:
- Extractor set contains only the URL.
- Module 1 is called and DOM tree of current instance is generated.
- The DOM tree is compared with previously generated DOM tree.
- The data is extracted wherever there is change in data or structure of DOM tree.

Module 4: Transform and Export
Input: Extracted Data.
Output: SMS, E-Mail, Table, Log
Description:
- The extracted data is again filtered.
- The filtered data is transformed into tables by serialization.
- Also the extracted data exported to user via SMS or E-Mail as per user choice.

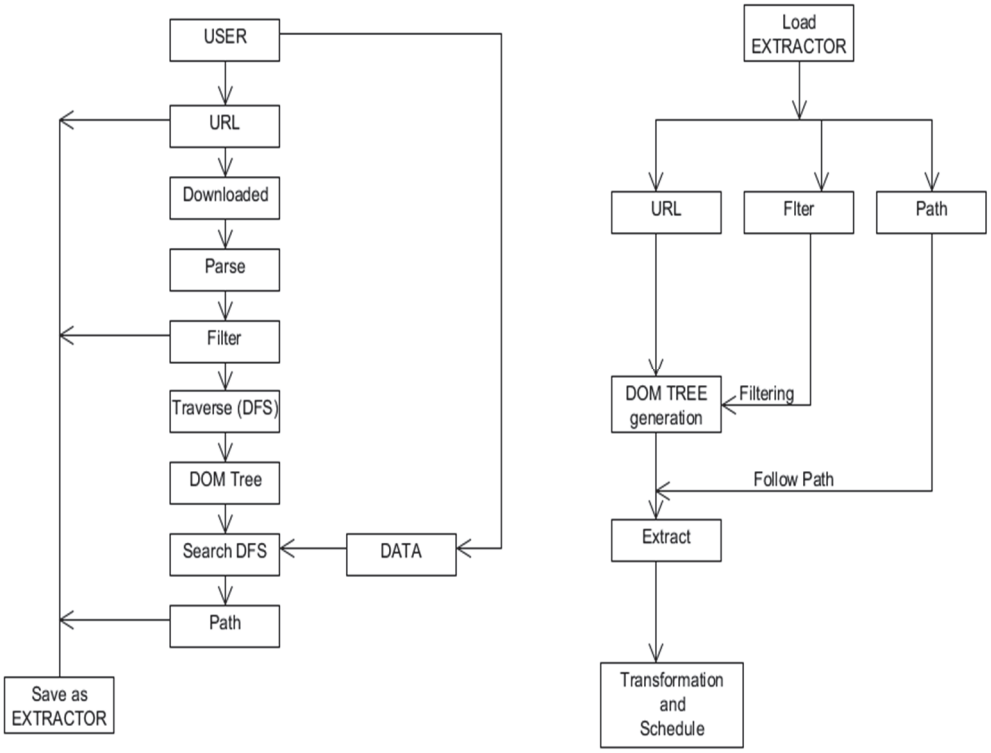

Figure 2.Block Diagram

Algorithm: The proposed algorithm works in iterative manner. The extractor set is generated which contains input URL and path to the required data in DOM tree and filter set. This extractor set is loaded and scheduled. The extracted data is then transformed.

Algorithm_ Pgath _Search:
Input: URL and Data to be Extracted
1. Download the webpage.
2. Generate the SAX tree using SAX parser.
3. The SAX tree is traversed by DFS to generate a DOM tree.
4. The data is searched in the DOM tree using DFS and the path to the data is stored as a extractor.
5. This extractor set is scheduled, and loaded. Current instance of URL is downloaded and parsed to generate the DOM tree.
6. This data is extracted from traversing the DOM tree using the path in a extractor set.
7. The extracted data is then transformed into SMS or e-mail.
8. Steps 5,6 and 7 are repeated after every interval stated in schedule.

Proposed System: In this paper we proposed a system to extract the data from the web and saving

the extract data accordance to the user need. We are using the DFS and DOM parser algorithm for extracting the data. The URL, from the data is to be extracted and the data, which is to be extracted is given by the user to the system. Using the extractor set the data is extracted.

Conclusion: We introduced a novel approach of web data extraction using path search algorithm. This paper offers new algorithm in implementing web data extraction using DOM parser and DFS. User can conduct their job without the need to write a script or program or even without any knowledge of computer programming technique.

References:

1. Chulyun Kim and Kyuseok Shim, "TEXT: Automatic Template Extraction from Heterogeneous Web Page", *IEEE Transaction on knowledge and data engineering, VOL. 23, No. 4, APRIL 2011*
2. Robert Baumgartner, Wolfgang Gatterbauer, Georg Gottlob,"WEB DATA EXTRACTION SYSTEM"
3. Rudy AG. Gultom Riri Fitri Sari Bagio Budiardjo, "Implementing Web Data Extraction and Making Mashup with Xtractorz", *2010 IEEE 2nd International Advance Computing Conference.*
4. Emilio Ferraraa, Pasquale De Meob, Giacomo Fiumarac, Robert Baumgartnerd, "Web Data Extraction, Applications and Techniques: A Survey", *Preprint submitted to Computer Science Review February 24, 2013.*
5. V. Anupam, J. Freire, B. Kumar and D.Lieuwen. Automating web navigation with the WebVCR. *Computer Networks*, 33(1-6): 503-517, 2000.
6. V. Crescenzi, G. Mecca and P. Merialdo. RoadRunner: Towards automatic data extraction from large Web sites. *VLDB*, 2001.
7. R. Baumgartner, S. Flesca and G. Gottlob. Visual web information extraction with Lixto. *VLDB*, 2001.
8. N. Kushmerick, D. S. Weld and R.B. Doorenbos. Wrapper induction for information extraction. *IJCAI*, 1997.
9. A.H.F. Laender, B.A. Ribeiro-Neto and A.S.da Silva. DEByE – Data extraction by example. *Data and Knowledge Engineering*, 40(2): 121-154,2000.
10. L. Liu, C. Pu and W. Han. XWRAP: An XML-enabled wrapper construction system for web information sources. *ICDE*, 2000.
11. I. Muslea, S. Minton, and C. Knoblock. A hierarchical approach to wrapper induction. *Autonomous Agents and Multi-Agent Systems*, 4(1/2): 93-114, 2001.
12. A. Pan, J. Raposo, M. Álvarez, P. Montoto, V. Orjales, J. Hidalgo, L. Ardao, A. Molano and Á. Viña. The Denodo data integration platform. *VLDB*, 2002.
13. W3Schools.com, The HTML DOM node Tree, http://http://www.w3schools.com/htmldom/ dom_ nodetree.asp
14. Wikipedia, Web Scrapping (Web Data Extraction),http://en.wikipedia.org/wiki/Web_data_extraction.

* * *

JSPM's Rajarshi Shahu College of Engineering, Pune-411033
amit_singh7@live.com,pluto.abhi@gmail.com,vinaysingh391992@gmail.com
c.sangole@live.com,jpkshirsagar@gmail.com