Web data extraction using the approach of segmentation and parsing

P. Singam [1#], Prof. P. Pardhi [2*]

1# Student M. Tech. (Comp.Sci. & Engg), 2 Assistant Professor, Comp. Sci. & Engg. Deptt.*
R.C.O.E.M., Nagpur (India)

***Abstract*— Given the URL's, automatically extracting the data from these result pages is very important for many applications, such as data integration, which need to cooperate with multiple web databases. In this paper we present a method which can extract the data of our interest out of the identified data regions, filter out the unwanted data records and finally put the extracted data into the table or export to csv files. Extraction procedure includes segmentation of contiguous as well as non contiguous data region, filtration of noise, and applying parsers. The implication of this is improved efficiency and better control over the extraction procedure. Our experimental results confirmed this.**

***Keywords*— Data region, Data extraction, DOM structure, Harvesting, Web data.**

I. INTRODUCTION

In the last few years, several works in the literature have addressed the problem of data extraction from web pages. The importance of this problem derives from the fact that, once extracted, the data can be handled in a way similar to instances of a traditional database. With the explosion of the World Wide Web, a wealth of data on many different subjects has become available on line. This has opened the opportunity for users to benefit from the available data in many interesting ways.

Enormous amount of data is stored in open databases. Most databases retrieve web pages with structured data objects. The data is important and useful for many applications: i)Price comparison engines ii)Collecting individuals information etc..

There are roughly three knowledge discovery domains that pertain to web mining [8]: Web Content Mining, Web Structure Mining, and Web Usage Mining. Web Content Mining is the process of extracting knowledge from the content of documents or their descriptions. Web Structure Mining is the process of inferring knowledge from the World Wide Web organization and links between references and referents in the Web. Finally, Web Usage Mining, also known as Web Log Mining, is the process of extracting interesting patterns in Web access logs.

In this paper we have considered web content mining and addressed the problem of extracting data from a Web page that contains several structured data records. Web pages on many Web sites are produced dynamically as structural records. The Objective is to segment these data records, extract data items or fields from them and put the data in a database table.

There are two algorithms for the data extraction i.e. Top-down, bottom-up algorithm. On the basis of these two algorithms, there is a development of Hybrid algorithm called Bi-Direction Data Extraction. It can be able to extract and discriminate the relevance of different repetitive information contents with respect to the user's visual perception of the web page.

Another method to extract useful information from web pages is, first, extract URLs from web pages and then use these extracted URLs to retrieve next pages via the HTTP request. If all pages are accessed via URLs, such a data extraction model is called the URL-oriented data extraction model.

In this paper we are presenting an approach for automatic web data extraction from web pages for given URLs .

A Types of Web Pages-

With respect to page content, there are basically two kinds of pages: those containing semi structured data and those containing semi structured

text. For example consider the page presented in figure 1 and 2, which are example of pages containing semi structured data and semi structured text, respectively. While pages of first type feature data items (eg. Names, Price, Category, etc.) implicitly formatted to be recognize individually while pages of the second type being free text from which data item can only be inferred.

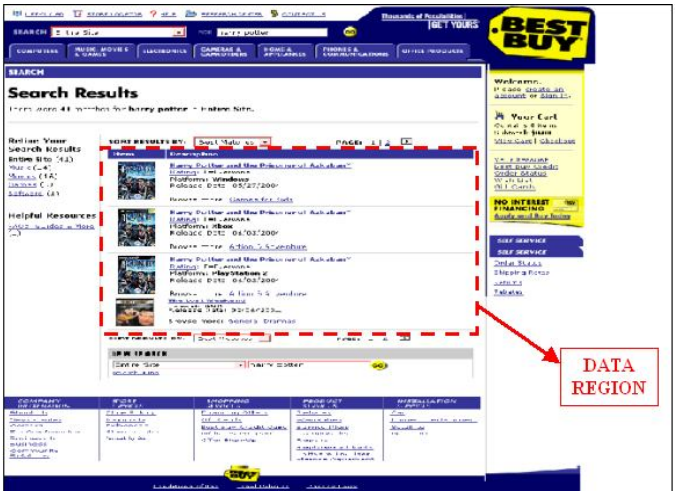

Figure 1: Pages containing Semistructured Data

Figure 2: Pages containing Semistructured Text

Regions of the HTML file that contain description of similar items (data records that needed to be extracted) are called data region. Each region doesn't necessary contain one data field and it may consists of several data fields.

The paper has been organized as follows, section 2 related work section 3 discusses the approaches and techniques carried out for data extraction, section 4 gives the details of implementation, section 5 discusses the result obtained and its analysis, section 6 is conclusion and 7 is future implementation followed by refrences.

II RELATED WORK -

In [1] this paper proposes a novel approach to page segmentation, taking advantage of graph grammars to provide robust page segmentation the spatial graph grammar (SGG) is used in this approach to analyze Web interfaces. This approach interprets a Web page, or any interface page, directly from its image Image-processing techniques are used to divide an interface image into different regions and recognize and classify atomic interface objects, such as texts, buttons, etc., in each region..

In [2] this paper, the data extraction problem has formulated as the decoding process of page generation based on structured data and tree templates. Author propose an unsupervised, page-level data extraction approach to deduce the schema and templates for each individual Deep Website, which contains either singleton or multiple data records in one Webpage. Authors schema called FiVaTech, applies tree matching, tree alignment, and mining techniques to achieve the challenging task. FiVaTech contains two phases: phase I is merging input DOM trees to construct the fixed/variant pattern tree and phase II is schema and template detection based on the pattern tree.

According to the Author's [3] investigations development of a lightweight ontological technique using existing lexical database for English (WordNet) is able to check the similarity of data records and detect the correct data region with higher precision using the semantic properties of these data records, for aligning iterative and disjunctive data items. Tests also show that the wrapper is able to extract data records from

multilingual web pages and that it is domain independent.

A novel data extraction and alignment method called CTVS that combines both tag and value similarity is presented [5]. CTVS automatically extracts data from query result pages by first identifying and segmenting the query result records (QRRs) in the query result pages and then aligning the segmented QRRs into a table, in which the data values from the same attribute are put into the same column.

In [6] paper they introduce the webpage understanding problem which consists of three subtasks: webpage segmentation, webpage structure labeling, and webpage text segmentation and labeling. They segmented a webpage into semantic blocks and label the importance values of the blocks using a block importance model. Then the semantic blocks, along with their importance values, are used to build block-based Web search engines. These entities and their relationships are automatically mined from the text content on the Web.

III OVERVIEW OF PROPOSED WORK –

A web page usually contains various contents such as navigation, decoration, interaction and contact information, which are not related to the topic of

the web-page. Furthermore, a web page often contains multiple topics that are not necessarily relevant to each other. The problem of extracting data from a Web page that contains several structured data records. The Objective is to segment these data records, extract data items or fields from them and put the data in a database table.

We developed a method to extract data from a given web page. The algorithm first finds regions of the HTML file that contain description of similar items (data records that needed to be extracted). These regions are called data region record. The second phase of the algorithm is to identify the noisy data which is then filtered out by passing it through three filters. The next phase is to identify data fields in each extracted region. To be able to find regions of the HTML file that contain a data record, we first build a DOM tree from the input HTML file. Then similar adjacent nodes in the DOM tree are found. The similarity of two nodes is measured using the number of child and their structure. All the nodes that are classified as similar and are adjacent in the DOM tree (i.e. have the same parent) are considered as the same data region. The next step of algorithm is to find data fields in each extracted region. Each region doesn't necessary contain one data field and it may consists of several data fields. To be able to extract relevant field in each region we have designed parsers.

Before performing the extraction process, this tool turns the document into parse tree a representation that reflects its HTML tag hierarchy (DOM structure). Further extraction is done automatically by applying extraction rule to the DOM structure.

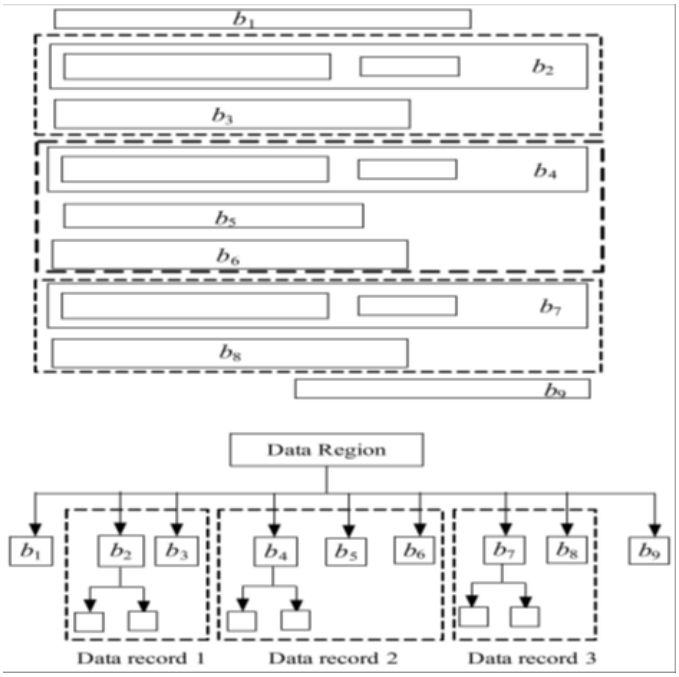

Figure 3: Segmented page and its equivalent DOM tree

The Document Object Model most often referred to as DOM is a cross-platform and language independent convention for representing and interacting with objects in HTML. The DOM tree defines the logical structure of documents and the

way a document is accessed and manipulated. It is constructed based on the organization of HTML structures (tags, elements, attributes). The HTML DOM views a HTML document as a tree-structure (node-tree). Every node can be accessed through the tree. Their contents can be modified or deleted. New elements can also be created. In this paper, the basic approach of web data extraction process is implemented through the Document Object Model (DOM) tree. Using a DOM tree is an effective way to identify a list or extract data from the web page. Anything found in an HTML document can be accessed, changed, deleted or added using the DOM tree. Fig 3. shows an Overview of the DOM Tree depicting the set of nodes that are connected to one another. The tree starts at the root node and branches out to the text nodes at the lowest level of the tree.

A Web data extraction-

By web data, we mean a content phrase in HTML that contains the information like phone no, E-mail address, price etc. when we search "Java" on www.amazon.com, we may get a simplified result page like:

<html><body><table>
<tr><td>Java 2: A Beginner's Guide</td></tr>
<tr><td>Head First Java</td></tr>
<tr><td>abcd@hotmil.com</td></tr>
</table></body></html>

The key phrases will be "Java 2: A Beginner's Guide", "Core Java", and "abcd@hotmail" We will extract a phrase list for each site we searched.

While extracting the key phrases we faced certain Issues. These are mainly:

1. Identify the data region.
2. Identify the boundary of data regions.
3. Non contiguous data region.
4. Noisy data regions.
5. Varying structure of web pages.

For handling each of these issues we have designed

Following Modules:

1. Data region processing.
2. Filtration of Unwanted data region.
3. Extract contents from data region.
4. Parsing the content.
5. Creating records.

Figure 4 below gives the architecture of proposed model showing the purpose of the each module.

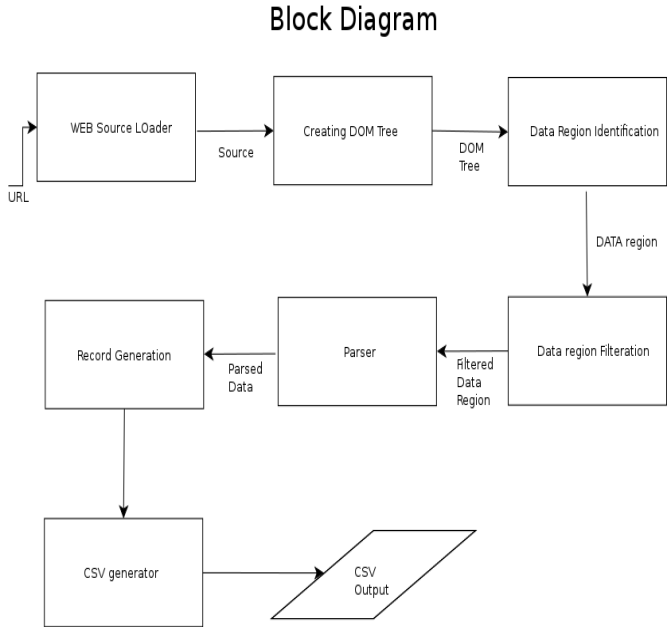

Fig4. Proposed Model

IV IMPLEMENTATION-

Now the algorithm is discussed here which we have designed for the process of web data extraction verifying with the experimental results. Our algorithm relies on the DOM tree representation of a web page, and traverses it in a bottom-up fashion in order to find the data-rich nodes

A Data Region Identification-

For data region identification, objects (node) of the DOM have considered. We first built the Document Object Model, which is constructed out of the body of the HTML page. While constructing this tree we ignore head tags of the page, since data is always arranged within the body tags. Each of these nodes is maintained in a list. For identifying Data Region

similar to [5] and [10], we compare tag strings of individual nodes including descendants and

combination of multiple adjacent nodes. Similar nodes are labeled as data region. Generalized node is introduced to denote each similar individual node and node combination. Adjacent generalized nodes form a data region. Gaps between data records are used to eliminate false node combinations. It has been observed that in many query result pages [5] some additional item that explains the data records, such as a recommendation or comment, often separates similar data records. Hence to handle noncontiguous regions we are maintaining a list of regions where we are storing the start node and end node exhibiting similar structure. We match each node with its sibling, if the mismatch occurs the next node is considered as the first node of the new data region and an entry is register in the data region list. Likewise all the complete page is traversed and data regions are identified. The data region identification algorithm discovers data regions in a top-down manner. Starting from the root node of resulting DOM tree of the query result page, the data region identification algorithm is applied to a node n and recursively to its children.

B Filtration of Unwanted data region.

After identifying the entire possible data region some of the regions may not content data of our interest, hence need to be filtered out. We have designed 3 types of filters 1) Minimum Filter 2) Blank Filter 3) Script Filters. After identifying all possible data regions, these data regions are passed through the filters which filters out unwanted/noisy data region

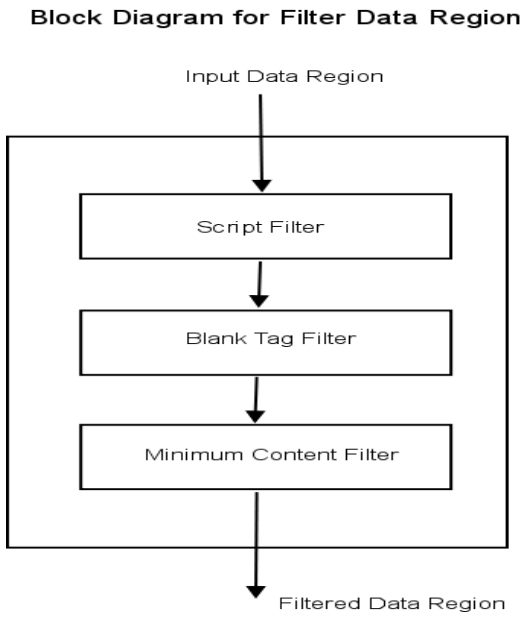

Fig:5 Block Diagram for Filter Data Region

C Extract contents from data region and Parsing-

The contents are then extracted from the remaining data regions. These extracted contents are then parsed to identify their labels and stored as record in csv files. For experimentation we have designed 3 parsers to identify 1)phone no 2)email id 3)price.

For creating these parsers we have written regular expressions which can automatically identify extracted text as phone number or email id or

price. Similarly we can write regular expression for identifying labels of other fields. We prefer to use the natural text segments [5] of a web page as atomic labeling units. The text features are very effective in web entity extraction and they are different for different entity types. For example, for price entity extraction, below are two example text features:

• the text fragment only contains RS/ $/INR[and digits;

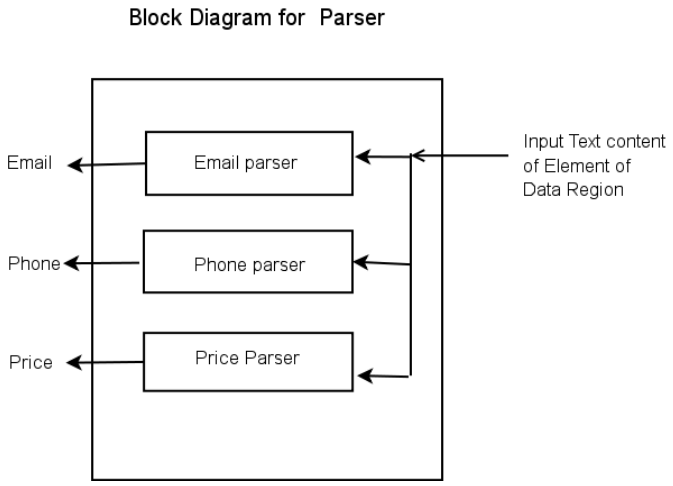

Fig:6 Block Diagram for Parser

D Algorithms

Web Data Extraction Algorithm
Step 1: Add URL in the list
Step 2: Select the URL from the List
Step 3: Find data regions
Algorithm to Find Data Region

```
Algorithm ProcessNode
    remainingNodeList = new ArrayList
        childNodes = node.ChildElements
    if childNodes.length > 1 then
            firstNode = childNodes[0]
                for index = 1 to  index < childNodes.length
                    nextNode = childNodes[index]
                        if  match(firstNode, nextNode) then
                          dataRegion = DataRegion(firstNode)
                          if dataRegion == null then
                            dataRegion = DataRegion(firstNode, nextNode)
                            dataRegionList=>add dataRegion
                dataRegionEndNode=>nextNode
                    else
                        remainingNodeList=>add(firstNode)
                        if index == childNodes.length - 1 then
                          remainingNodeList.add(nextNode)
                firstNode = nextNode
                index++
        for node1 to remainingNodeList
          process(node1)
      else
      if  childNodes.length == 1 then
          process(childNodes[0])
```

Step 4: Filter out unwanted data regions
Step 5: Extract the contents of the data region
Step 6: Parse the content and assign the labels to each field
Step 7: Store the extracted data

V EXPERIMENTATION-

This section describes the data we used in our experiments and reports results of the experiments. The algorithm has been used to conduct experiments on several sites. All experiments were conducted on an Samsung Laptopsn equipped with an Intel Pentium processor working at 2GHz, with 2GB RAM, running Linux and Java NetBeans IDE 7.2.

We have given 11 sets of experiments. The goal is

to examine time constraint of the web harvesting process for 11 different URL's with varying size f web pages. Our web harvesting algorithm identify the data regions and extract phone numbers, email id and price whichever is present. The experimental results obtained are given in the table below. Also we have shown charts for respective results.

Sr.	File Size (Size)	Data Region Regions	Data Region Time	Filtration Records Region	Filtration Records Time	Parsed Records Records	Parsed Records Time	Total time (seconds)
1	178658	129	0.692	82	0.003	14	0.052	0.747
2	209245	184	0.863	124	0.007	14	0.055	0.925
3	91386	28	0.187	21	0.003	3	0.03	0.22
4	90123	28	0.17	21	0.003	0	0.029	0.202
5	233703	111	0.813	75	0.003	14	0.033	0.849
6	214916	115	0.724	80	0.002	16	0.038	0.764
7	198625	58	0.466	43	0.001	15	0.02	0.487
8	349286	135	1.339	79	0.004	3	0.052	1.395
9	267721	70	0.89	58	0.003	24	0.045	0.938
10	215093	68	0.58	49	0.003	0	0.031	0.614

Table1:Analysis of performance of different processes with respect to time

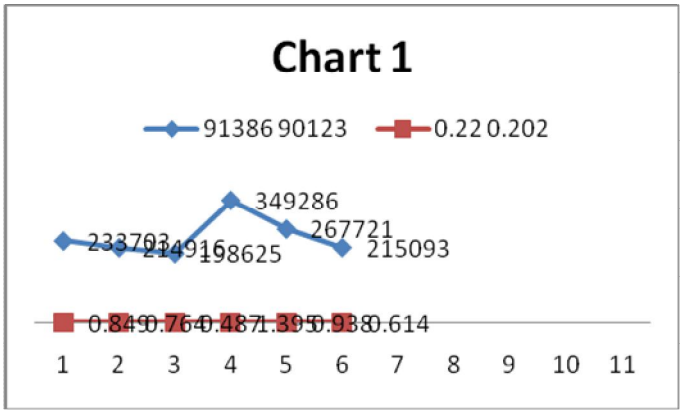

Chart1: Showing the results of total time required for extracting data against the file size.

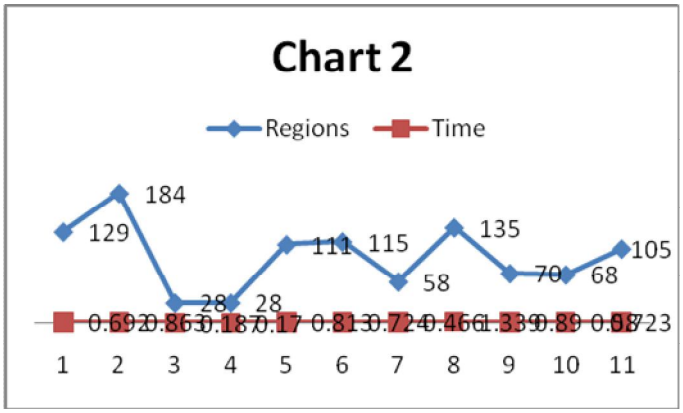

Chart2: showing the results of time required for extraction of data region.

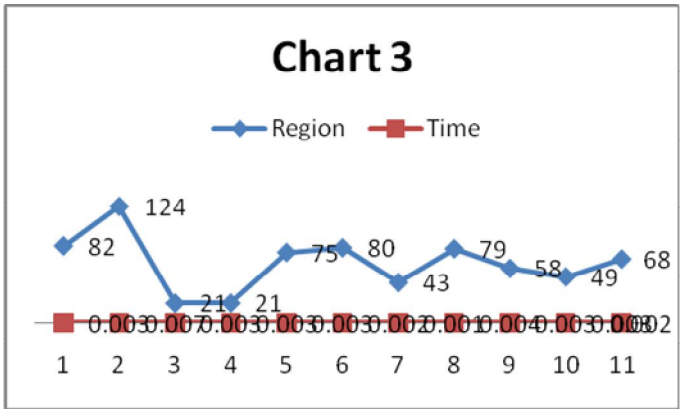

Chart3: showing the results of region filtration against required time

VI CONCLUSION –

- We propose a Extractor system which is able to extract data from various web sources continually by automating the entire web data extraction process.
- Our approach includes the DOM tree generation, each time the web page is traversed its object get created and is stored in the list.
- Since it traverse the entire web page and stores only the start node and next node entry, it considerably reduces the required storage space.
- Extractor system allow the users to efficiently and effectively perform the task of web data extraction through an user interactive GUI.
- Web harvester is working efficiently with any varying structure of web pages.
- Experimental results on real-life data-intensive Web sites confirm the feasibility of the approach.

VII FUTURE WORK-

- The work can be extended for extracting more fields from web pages.
- In present approach we didn't consider the image data we have addressed only text data hence it can also be included as a part.
- The extracted data can be used to populate big databases.

REFERENCES -

[1] Jun Kong, Omer Barkol, et al., "Web Interface Interpretation Using Graph Grammars", IEEE transactions on systems, man, and cybernetics—part c: applications and reviews, vol. 42, no. 4, july 2012

[2] Mohammed Kayed and Chia-Hui Chang, " FiVaTech: Page-Level Web Data Extraction from Template Pages", IEEE transactions on knowledge and data engineering, vol. 22, no. 2, february 2010

[3] Jer Lang Hong, "Data Extraction for Deep Web Using WordNet", IEEE transactions on systems, man, and cybernetics—part c: applications and reviews, vol. 41, no. 6, november 2011

[4]Weifeng Su, Jiying Wang, Frederick H. Lochovsky , "Combining Tag and Value Similarity for Data Extraction and Alignment" IEEE transactions on knowledge and data engineering, vol. 24, no. 7, july 2012

[5] Zaiqing Nie, Ji-Rong Wen, and Wei-Ying Ma, "Statistical Entity Extraction From Web"

[6] Luis Tari, Phan Huy Tu, Jo¨ rg Hakenberg, Yi Chen, Tran Cao Son, Graciela Gonzalez, and Chitta Baral "Incremental Information Extraction Using Relational Databases", IEEE transactions on knowledge and data engineering, vol. 24, no. 1, january 2012

[7] Hassan A. Sleiman and Rafael Corchuelo, "A Survey on Region Extractors From Web Documents", IEEE transactions on knowledge and data engineering

[8] Dave King "Introduction to the Web Mining Minitrack", 2012 45th Hawaii International Conference on System Sciences

[9] Alberto H. F. Laender, et.al. "A Brief Survey of Web Data Extraction Tools", Department of Computer ScienceFederal University of Minas Gerais 31270901n Belo Horizonte MG Brazil

[10] Y. Zhai and B. Liu, "Structured Data Extraction from the Web Based on Partial Tree Alignment," IEEE Trans. Knowledge and Data Eng., vol. 18, no. 12, pp. 1614-1628, Dec. 2006.

Books:

[11] ERCIM NEWS 34 89 April 2012 "Special theme:Big Data"

[12] A Comparison of Leading Data Mining Tools (ARTICAL) John F. Elder IV & Dean W. Abbott Elder Research