Journal of Computational Information Systems5:6(2009) 1757-1766
Available at http://www.JofCI.org

A Robust Approach of Automatic Web Data Record Extraction

Yongquan DONG[1,2], Qingzhong LI[1,†]

[1] *School of Computer Science and Technology, Shandong University, Jinan, China*

[2] *School of Computer Science and Technology, Xuzhou Normal University, Xuzhou, China*

Abstract

The automatic extraction of Web data record is a key problem for Web data integration. However, the bottleneck problem is the structure diversity of Web pages, which leads to the low precision of Web data record extraction. A robust approach is proposed to automatically extract data records from Web pages. Firstly, it utilizes the visual information to identify data region containing data records. Secondly, it uses clustering algorithm to divide similar nodes into one group. Finally, it uses repetitive regularity of text nodes of different records to generate the final data records. Comparing with many other existing works, the approach is applicable for pages on which any data record is in the same subtree and across different subtrees. Experiment results on real web pages show that our approach can achieve high extraction accuracy and outperform the existing techniques substantially.

Keywords: Data Record Extraction; Automatic Extraction; Information Extraction

1. Introduction

Structured data objects are a very important type of information on the Web. Such data objects are often records from underlying databases and displayed in Web pages with some fixed templates. In this paper, we also call them data records. Mining data records in Web pages is useful because they typically present their host pages' essential information, such as lists of products and services. Extracting these structured data objects enables one to integrate data/information from multiple Web pages to provide value-added services, e.g., comparative shopping, meta-querying and search.

Due to the huge amount of web pages, it is unreasonable to extract data records by hand or in semi-automatic ways. The paper proposes a robust automatic web data record extraction. Firstly, it utilizes the visual information to identify data region containing data records. Secondly, it uses clustering algorithm to divide the similar nodes into one group. Thirdly, it filters the noisy nodes. Finally, it uses some heuristic rules to generate data records.

The approach is applicable for pages on which data records are in the same subtree or across different subtrees. For example, in Figure 1(b), data record R_1,R_2,R_3 are individually under the subtrees rooted at t_1,t_2,t_3 and in Figure 2(b), data record R_1 is across two subtrees rooted at t_1,t_2;R_2 across three subtrees rooted at t_3,t_4,t_5; R_3 across three subtrees rooted at t_6,t_7,t_8. Most existing approaches have only discussed data

† Corresponding author.
Email addresses: dongyongquan@mail.sdu.edu.cn(Yongquan DONG), lqz@sdu.edu.cn (Qingzhong LI).

December, 2009

records in the same subtree and seldom researched on data records across different subtrees. As the situation of data records across the subtree covers the problem in the same subtree, this paper places emphasis on the problem of data records across subtrees. Experiment results on test beds of real web pages show that the approach can achieve high extraction accuracy and outperforms the existing techniques substantially.

The rest of the paper is organized as follows. An overview of related work is given next. Then, some basic definitions are given in Section 3. In Section 4 we present the observations for the characteristics of data records in web pages. The algorithm of web data record extraction is introduced in Section 5. Experimental results are described in Section 6 and conclusions are made in the last section.

Fig.1(a) Web Page with Data Records in the Same Subtree

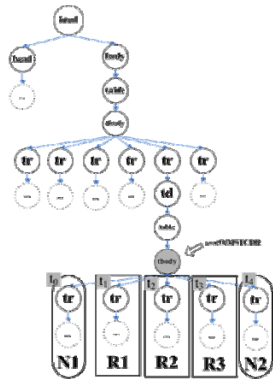

Fig.1(b) Fragment of DOM tree Corresponding to Figure 1(a)

Fig.2(a) Web Page with Data Records across Different Subtrees

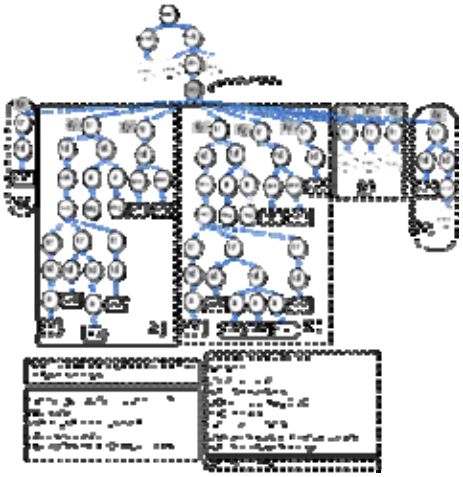

Fig.2(b) Fragment of DOM tree Corresponding to Figure 2(a)

2. Related Work

Web data record extraction has been an active research field for years. Many approaches have been proposed, but these approaches require some kind of human intervention to extract data records [1-7]. When the sources are not known in advance, these approaches are not feasible.

Several works have addressed the problem of performing web data extraction tasks without requiring human input. Omini [8] parses web page into DOM tree and applies some heuristic rules to identify the boundaries of data records. The rules in Omini are so simple that they can not adapt to the complex web pages. IEPAD [9] uses the Patricia tree [10] and string alignment techniques to search for repetitive patterns in the HTML tag string of a page. The method used by IEPAD is very probable to generate incorrect patterns along with the correct ones, so human post-processing of the output is required and it is not completely automatic. RoadRunner [11] receives as input multiple pages conforming to the same template and uses them to induce a union-free regular expression which can be used to extract the data from the pages conforming to the template. The method can not deal with disjunctions in the input schema and it requires receiving as input multiple pages conforming to the same template. As well as RoadRunner, ExAlg [12] receives as input multiple pages conforming to the same template and uses them to induce the template and derive a set of data extraction rules. ExAlg makes some assumptions about web pages which do not hold in a significant number of cases: for instance, it is assumed that the template assigns a relatively large number of tokens to each type constructor. It is also assumed that a substantial subset of the data fields to be extracted have a unique path from the root in the DOM tree of the pages. And it also requires receiving as input multiple pages. MDR [13] mines multiple similar generalized nodes to identify data regions. Every generalized node corresponds to a data record which is composed of one or many nodes with the same parent node. But the experiments in [14] shows that it does not adapt to the third dataset and the approaches in MDR extract all possible data records which contain many irrelevance contents. DEPTA [15] uses the visual layout of information in the page and tree edit-distance techniques to detect lists of records in a page and to extract the structural data records that form it. DEPTA supposes that the same number of sub-trees must form all records and the visual gap between two data records in a list is bigger than the gap between any two data values from the same record. These assumptions do not hold in all web sources. ViNTs [14] uses the visual characteristics of pages from search engine to capture the content regularities and combines the structure of HTML tags to extract data records. In order to generate the wrapper, it requires some result pages with many records and one result page without records which show the limitations of the approach. [16] proposes a web data record automatic extraction approach based on tree structure. It uses the semantic blocks to model HTML page and differentiates different value roles in semantic block to induct data schema and extraction rules. The approach only finds data records in the same sub-tree and does not process the situation across different subtrees.

The approach proposed in the paper overcomes the above shortcomings. It only requires one web page to automatically extract data records and provide the solutions for the situation across different sub-trees.

3. Basic Definitions

In order to facilitate the following description, we first give some basic definitions.

Definition 1. A data region is a collection of two or more data records. For example, in Figure 2(a), the solid rectangle pointed by the arrow is data region of the page.

Definition 2. Let T=<V,E> be a DOM tree of a web document D where $V=V_I \cup V_L$, V_I is a finite set of tag nodes(internal nodes) and V_L is a finite set of content node(leaf nodes); $E \subseteq V \times V$, representing the directed edges.

Definition 3. Let T_u is a subtree anchored at node u where $u \in V_I$ and S_u is a token string corresponding to T_u. S_u is obtained through the next two steps:(1)replacing the text node in V_L of T_u with specific label "text"; (2)depth-first traversing the subtree of T_u and when passing a node, adding S_u with a token which is a label for the node. For instance, in Figure 2(b), S_{t1} of the t1 node is tr -> td -> table -> tbody -> tr -> td -> a -> text -> tr -> td -> text -> td -> a -> text -> tr -> td -> text -> td -> a -> img -> a -> img. In order to differentiate different tokens, we use the symbol "->" as the separator between labels. In fact, the token string is stored in a list.

Definition 4. A subtree T_u anchored at node u, $u \in V_I$, is a minimal subtree containing data region(MSTCDR) where u is the root of MSTCDR(rootOfMSTCDR). In Figure 2(b), the tbody node with gray shading is rootOfMSTCDR which is labeled by an arrow.

Definition 5. Let $u \in V_I$, $v \in V_I$, the similarity of u and v is the edit-distance similarity between S_u and S_v denoted by edsim(u,v). The string edit distance between S_u and S_v(edsim(u,v)) is calculated using a variant of the Levenshtein algorithm [17] , which does not allow substitution operations(only insertions and deletions are permitted). To obtain a similarity score between 0 and 1, we normalize the result using the equation (1). In Figure 2(b), the similarity between t1 and t2 is 0.182 .

$$edsim(u,v) = 1 - ed(S_u, S_v) / \max(length(S_u),length(S_v)) \quad (1)$$

4. Basic Observations and Properties

We have detected and extracted lists of structured data records embedded in HTML pages to find the following properties:

(1) Data region always contains the center of the whole page;

(2) Data region always occupies comparatively large area of the whole page;

(3) Data records in data region are always composed of one or more contiguous subtrees in the page;

(4) Any two data records in data region do not overlap;

(5) Some non-record nodes sometimes occur in data region which are called noise nodes. They often occur at the beginning and end position of data records. The statistics information of record searching result often occurs at beginning position and the pagination information at end position. Noise nodes requires cutting.

(6) The data records in data region have strong similarity each other although there are some difference between them, for example, the existence of repetitive items and optional items. The similarity between data records embodies in tag path, word and url.

In Figure 2(a) the area pointed by the arrow is data region. It contains three data records (R_1,R_2,R_3) and the center of the whole page. It also occupies comparatively large area of the page. In Figure 2(b), we can

see that R_1, R_2, R_3 are adjacent in DOM tree and they have no intersection. At the beginning and end position, there are non-record subtrees which are respectively denoted by N_1 and N_2. In Figure 2(b), we can also see that R_2's Author is a repetitive item and R_2, R_3 has a more optional item "Other editions" than R_1. Nonetheless, they are very similar to each other. We can also find that every record has common word, such as "Author" etc and the tag path of the common word is same.

5. The Record Extraction Algorithm of Web Listing Page

In this section, we elaborate the process of record extraction in the following parts.

5.1. Finding Data Region

According to the properties 1,2, we can determine the region containing the center of the page and occupying the large area of the page, that is to say, we must find the rootOfMSTCDR in DOM tree. We use page rendering approach to obtain the coordinates of each node. As every node in DOM tree corresponds to a rectangle of page rendered in the explorer, the coordinates of every node is the information of the rectangle, denoted by [x,y,width,height], where x,y represents x-coordinate and y-coordinate of top left corner and width, height represents width and height of the rectangle. In order to effectively obtain rootOfMSTCDR, we need compute the ratio of the rectangle area of every node to the area of the whole page and judge whether the center of the page is in the rectangle. The two piece of information is easily computed, which are denoted by "ratio" and "isCenterLocInRec" and stored in every node. We compute the ratio using the equation (2). According to property 1, if isCenterLocInRec is true, the node contains data region, and conversely it does not contain data region. According to property 2, if ratio is greater than T_{region}, where T_{region} denotes the threshold of ratio, the node contains data region. Threshold Tregion can be acquired by training and we set it to 0.27 in the experiment.

$$ratio = area_{node}/area_{page}, \tag{2}$$

where $area_{node}$ represents the area of the node, $area_{page}$ represents the area of the whole page.

5.2. Clustering Child Nodes of RootOfMSTCDR

By property 3 and 4 every record is composed of one or several consecutive sibling subtrees which are direct descendants of the rootOfMSTCDR and does not overlap each other. We can leverage on this property to group the subtrees into different classes, and then combining them into different records.

For grouping the subtrees according to their similarity, we use a clustering-based process described in the following lines:

Let the set $\{T_1,T_2,\ldots,T_n\}$ is all the subtrees which are direct children of the node rootOfMSTCDR. Each Ti can be represented as a token string S_i using the definition 3;

Compute the similarity matrix. This is a n×n matrix where the (i,j) position(denoted m_{ij}) is obtained by $edsim(S_i,S_j)$, the edit-distance similarity between S_i and S_j.

We define holistic similarity between T_i and T_j, denoted $hs(t_i,t_j)$, as the inverse of the average absolute error between the columns corresponding to T_i and T_j in the similarity matrix. The holistic similarity is defined in equation (3). Therefore, to consider two subtrees as similar, the holistic similarity measure

requires their columns in the similarity matrix to be very similar. This means two subtrees must have roughly the same edit-distance similarity with respect to the rest of subtrees.

$$hs(T_i,T_j) = 1-\sum_{k=1..n}|m_{ik}-m_{jk}|/n \quad (3)$$

Now, we apply bottom-up clustering to group the subtrees. The basic idea behind this kind of clustering is described as follows. We regard every node as a cluster. For every node T_i, we perform the two steps. First, we compute the similarity between T_i and every current cluster C_j. The similarity is defined in equation (4). Then if there is a cluster C_k with the max similarity to T_i and the similarity value is greater than a threshold θ_{nc}, we add T_i into cluster C_k. Otherwise, we build a new cluster with only one node T_i. We repeat the two steps until all nodes have been processed completely. θ_{nc} represents the threshold between subtree and cluster and we set it to be 0.9 in the following experiment.

$$edsim(T_i, C_j) = \sum_{k=1,\ldots,p} hs(T_i,T_k)/p \quad ,\text{where } p=|C_j| \quad (4)$$

Figure 3 gives the clustering result of Figure 2(b). It generates five clusters. The categories of child nodes anchored at rootOfMSTCDR are denoted by cluster identifiers (CI). These cluster identifiers constitute a cluster identifier sequence (CIS) $C_0C_1C_2C_1C_2C_3C_1C_2C_3C_4$

Fig.3 The Clustering Result of Figure 2(b)

5.3. Filtering Noise Nodes

According to property 5, noise nodes often lie at the beginning or end position. We use the following heuristic rules to filter noise nodes. From the left node, if its CI only appears in the CIS once, we delete it until encountering a node whose CI appears several times in the CIS. Through these processing, we can filter the noise nodes at the beginning position. From the right node we use the same processing to filter the noise nodes at the end position. After filtered, CIS is denoted by CISAF (Cluster Identifier Sequence After Filtered). In Figure 3, from left to right, as C_0 occurs only once and C_1 three times, C_0 need to be filtered. From right to left, as C_4 occurs only once and C_3 occurs two times, C_4 need to be filtered. Finally, CISAF is represented as $C_1C_2C_1C_2C_3C_1C_2C_3$.

5.4. Generating Records

After obtaining CISAF, we should divide it into records. Nevertheless, the number of possible combinations would be two high: if the number of subtrees is n, the possible number of divisions verifying property 1 is 2n-1. In some sources, n can be low, but in others it may reach values in the hundreds (e.g. a source showing 25 data records, with an average of 4 subtrees for each record). Therefore, this exhaustive approach is not feasible. As many Web sources use templates to automatically generate list pages and fill them with results of a database query, data records in list page always have some common characteristics in the text node. We consider the following features of the text node: word, url and tag path.

Word feature. We first tokenize the text of each CI subtree and each numeric token is assigned to numeric schema, for instance, $12.95 will be converted to $99.99. Then we make statistics for the frequency of each word. Those words which occur the same times with highest word frequency are record

description words(RDW). For example, in Figure 2(a), we will get record description words, "Author", "Paperback", "List", "Our", "$99.99","99%","OFF","US","Funds",etc which all occur 3 times with highest word frequency.

Url feature. For the text with hyperlink, we convert the hyperlink to the specific form. For example, in Figure 2(a), for the text1 in R1, we get its url "http://www.allbooks4less.com/default.asp?Ntk1=Default&Ntt1=java&Ntx1=matchall&Nsl=3228&Ix=0&R=9780658011986B&Rt=5&Nty1=1" and convert it to an ordered string "http www allbooks4less com default asp Ntk1 Ntt1 Ntx1 Ns1 Ix R Rt Nty1". In the conversation process, we keep the variable name and delete its value. Then we make statistics for the frequency of each url variation string. Those strings which occur the same times with highest string frequency are record description url(RDU).

For every RDW or RDU, we keep the tag path of its text node. If the tag path is same, we can judge that the RDW or RDU belongs to an independent record and its CI also belongs to an independent record. As in general a record does not has two or more RDWs or RDUs and data records are always contiguous in the DOM tree, different RDW or RDU can be used to generate the final records. For example, in Figure 2(b), "Author" is the RDW of text2 and text8 and "List" is the RDW of text5 and text13, so the CI(t1) containing text2 and the CI(t2) containing text5 should belong to the same record. The same is for the CI(t3) containing text8 and the CI(t4) containing text13. Due to the CI(t5) does not have RDWs or RDUs, it is combined with the immediately previous CI to formulate a record. So t3,t4,t5 group into a record and we can get the final data records which is showed in Figure 4.

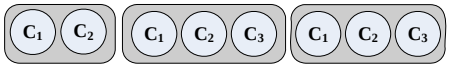

Fig.4 The Final Data Records

6. Experiment

This section describes the empirical evaluation of our techniques. First, we introduce the datasets used in Section 6.1. Then, the experimental results are presented in Section 6.2.

6.1. Datasets

We first used a set of 20 pages from 10 different web sites. The tests performed with these pages were used to adjust the algorithm and to choose suitable values for the used thresholds. Through the experiments, we set T_{region} to 0.27 and set θ_{nc} to 0.9.

Then, we chose 300 new websites in different application domains (book, music, movies, house, airfare, etc). We performed one query in each website and collected the first page containing the list of result. Some queries returned only 2-3 results while other returned hundreds of results.

6.2. Experiment Results

(1)The result of two stages

We measured the results at two stages of the process. The first stage is determining the accuracy of rootOfMSTCDR and the second stage is determining the precision and recall of generating data records. Table 1 shows the results obtained in the empirical evaluation. In the first stage, we use ICEBrowser [18] to render the web page and obtain the coordinate of each node. As is shown is Table 1, we can see that the

data region is correctly detected in all pages but three. In those cases, the area of the sidebar is larger than the area of data region which does not conform to property 2. In the second stage, the approach has a 98.9% precision and 97.8% recall. On the whole, our approach is remarkable.

Table 1 The Experiment Result of Two Stage

Stage 1	#Correct	#Incorrect	Accuracy
	297	3	99%
			Precision
Stage 2	#Records to Extract	4680	98.9%
	#Extracted Records	4577	Recall
	#Correct Extracted Records	4526	97.8%

(2)Compared with MDR

We compare our approach with the MDR system [13] . We separately apply our approach and MDR to the same dataset described in 6.1. The experimental results are given in Table 2.

In Table 2, Column 1 lists the site of each test page. Due to space limitations, we could not list all of them here. The number of pages that we used in our experiments is 300. Column 2 gives the number of data records contained in each page. These are those obvious data records of the page and do not include navigation areas, which have regular patterns as well. Column 3 shows the number of data records by our approach. It gives perfect results for all pages except one which lost four data records. Column 4,5 and 6 describe the result of MDR. Column 4 shows the number of correct data records found by MDR. Column 5 gives the total number of data records (which may not be correct) found by MDR. Column 7 gives some remarks about the problems with MDR.

The last three rows of the table give the total number of data records in each column, the recall and the precision of our approach and MDR. The precision and the recall are computed based on the total number of data records found in all the pages by each system and the total number of data records that exist in all the pages.

Before further discussing the experimental results, we first explain the problem description used in the table:

all-miss-error(n-miss-error): This means that all (or n) data records found by the system with some parts missing and some parts inaccurate.

non-found: None of the correct data records is found.

n-err: It means n incorrect data records are found.

miss n records: n correct data records are not identified.

The following summarizes the experimental results in Table 2.

(1) Our approach is able to give perfect results for web pages. From the last two rows, we can see that our approach has a 98.9% precision and 97.8% recall, which are remarkable. However, MDR only has a 57.3% precision and 42.1% recall.

(2)In columns 4 and 6, those cells that do not contain remarks show that the system finds all data records correctly. We can see that MDR only gives perfect results of few pages.

(3)As MDR determines data region according to tag "Table", we can not use MDR to the data records in many current popular web pages whose data region is under tag "Div".

(4)MDR extracts the data regions which contain repetitive items and does not consider page's topic, so it extracts many non-genuine data records.

(5)MDR deals with the situation across subtrees poorly and can not decide data record boundaries. In contract, our approach does not have this problem. It identifies the boundaries of the data records very well.

From our experiments, we can conclude that our approach is very accurate and is dramatically better than MDR.

Table 2 Results of Experiment Two

URL	Rec.	Ours	MDR		
			corr.	found	remark
http://www.allbooks4less.com/	5	5	0	5	all-miss-error
http://www.bookbrothers.net/	20	20	0	0	none-found
http://www.meijer.com/	15	15	0	0	none-found
http://www.textbookx.com/	15	15	15	15	
http://www.taobao.com	20	20	0	0	none-found
http://search.ap.dell.com/	20	20	0	0	none-found
…	…	…	…	…	…
http://www.tigerdirect.com/	20	20	20	24	4 err
http://www.jr.com/	12	8	10	20	10 err
http://www.giftbrand.com	10	10	10	13	3 err
http://www.samsclub.com/	15	15	12	12	miss 3 records
Precision		98.9%	57.3%		
Recall		97.8%	42.1%		

7. Conclusion

In this paper, we proposed a robust approach to automatically extract data records in a Web page. Our approach only requires a web page with three or more data records. It is automatic and thus requires no manual effort. It not only adapt to the situation in the same subtree, but also the one across subtrees. Experimental results show that our new method is extremely accurate. It outperforms the existing state-of-art system dramatically. In our future work, we plan to research on data alignment and data labeling, so that we can extract the fields of a data record and store them into database to provide better value-added service.

Acknowledgement

The authors thank the anonymous reviews for the invaluable comments. This work is supported by the National Natural Science Foundation of China under Grant No.90818001, the Natural Science Foundation of Shandong Province of China under Grant No.Y2007G24 and Y2007G38.

References

[1] Hammer, M. Hugh, G. Molina and H. Garcia-molina, "Semistructured Data: The TSIMMIS Experience", in Proceedings of the First East-European Workshop on Advances in Databases and Information Systems 1997, pp. 1--8

[2] A. Sahuguet and F. Azavant, "Building intelligent web applications using lightweight wrappers", Data Knowl. Eng., 2001, 36 (3): 283--316

[3] A. Hogue and D. Karger, "Thresher: automating the unwrapping of semantic content from the World Wide Web",2005, pp. 86--95

[4] V. Kovalev, S. S. Bhowmick and S. Madria, "HW-STALKER: a machine learning-based system for transforming QURE-Pagelets to XML", Data Knowl. Eng., 2005, 54 (2): 241--276

[5] I. Muslea, S. Minton and C. A. Knoblock, "Hierarchical Wrapper Induction for Semistructured Information Sources", Autonomous Agents and Multi-Agent Systems, 2001, 4 (1-2): 93--114

[6] Y. Zhai and B. Liu, "Extracting Web Data Using Instance-Based Learning", World Wide Web, 2007, 10 (2): 113--132

[7] R. Baumgartner, S. Flesca and G. Gottlob, "Visual Web Information Extraction with Lixto",2001, pp. 119--128

[8] D. Buttler, L. Liu and C. Pu, "A Fully Automated Object Extraction System for the World Wide Web",2001, pp. 361

[9] C. H. Chang and S. C. Lui, "IEPAD: information extraction based on pattern discovery",2001, pp. 681--688

[10] G. H. Gonnet, R. A. Baeza-Yates and T. Snider, "New indices for text: PAT Trees and PAT arrays", 1992,: 66--82

[11] V. Crescenzi, G. Mecca and P. Merialdo, "RoadRunner: automatic data extraction from data-intensive web sites",2002, pp. 624--624

[12] A. Arasu and H. Garcia-Molina, "Extracting structured data from Web pages",2003, pp. 337--348

[13] B. Liu, R. Grossman and Y. Zhai, "Mining data records in Web pages",2003, pp. 601--606

[14] H. Zhao, W. Meng, Z. Wu, V. Raghavan and C. Yu, "Fully automatic wrapper generation for search engines",2005, pp. 66--75

[15] Y. Zhai and B. Liu, "Web data extraction based on partial tree alignment",2005, pp. 76--85

[16] D. Hu and X. Meng, "Automatically extracting Web data using tree structure", Journal of Computer Research and Development, 2004, 41 (10): 1607-1613

[17] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals",1966, pp. 707--710.

[18] "ICEBrowser. http://www.icesoft.com/