IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 1, Issue 1, March, 2013
ISSN: 2320 - 8791
www.ijreat.org

A Novel Approach for Automatic Data Extraction from XML Web Pages

Niranjana A[1] and Dr Vidhya V[2]

[1,2] Computer Science and Engineering, Sri Venkateswara College of Engineering, Sriperumbudur, Tamil Nadu, India

Abstract
Internet is far and away the foremost widespread supply of information. Web pages contain the common templates with contents which are populated across multiple pages to achieve high productivity of publishing. Readers can easily access the contents from templates which are guided by consistent structures. The template detection and extraction has become a more useful technique in the web documents, since unknown templates are harmful as they degrade the performance and accuracy of web applications due to irrelevant terms in the templates. In this paper, we cluster the web documents using Agglomerative Hierarchical Clustering method based on the similarity among template structures within the documents. Then, unknown number of XML web documents are handled using Minimum Description Length principle. We generate wrappers for the clustered web documents and most relevant data are extracted from the web pages. Hence, an experimental evaluation shows that this proposed work is effective and scalable for retrieving the most relevant and meaningful data from the web pages.

***Keywords*:** *XML Template Extraction, Agglomerative Hierarchical Clustering, Minimum Description Length technique, Wrapper Construction, Data Retrieval*

1. Introduction

World Wide Web is a way of accessing data over the medium of Internet. It is the most useful source of information which is growing at a rapid rate in number of sites. Website is a set of related web pages that contain contents such as text, images, audio and video. Web page is a document which might be accessed through a web browser. Thus, web pages contain a combination of distinctive content and template, which is presented across multiple pages. However the unknown templates are more harmful, since they contain a large number of irrelevant terms in the web documents and therefore templates degrade the accuracy of information extraction from web pages. Thus, template detection and extraction has become a more important technique to increase the performance of web applications. For instance, web oriented applications like community web sites such as DBLife and Rexa that tracks information about researchers, conferences, talks, projects and events relevant to a particular community.
Information Extraction (IE), is that the task of automatically extracting the relevant data from web documents. But, there are two basic challenges in automatically detecting the template. First, there is no way to differentiate between the text that may be a part of template and text that is additionally a part of information. Second, the schema of information in web pages is not usually a flat set of attributes, however it is more complicated and semi structured. Extraction of such information permits one to integrate the data from multiple web sites to provide value added services, e.g., comparative shopping, object search and data integration. Wrapper generation is an important problem with a wide range of web applications. Building wrappers around web sources will offer two advantages. First, the flexibility to obtain relevant data from an individual source is enhanced.

But information extraction from the web pages that follow syntactical regularities is not trivial. Scalability is an important challenge as there is large number of sites. Another challenge is flexibility as the format of web sources may change. Generation of wrapper can be done either by manually or by an automatic approach. Manually writing wrappers could be a time consuming and error prone task. In practice, templates change very frequently, requiring repeated human intervention to generate wrappers. Any XML web document can be represented as a Document Object Model (DOM) tree and many similarity measures for trees have been obtained. However clustering process is more expensive with tree related distance measures. Many of the previous works considered a small size labeled data for their training. Also, the templates have to be differentiated from the content for efficient data extraction. It is not possible to identify the templates simply by the frequencies of words.

In this work, in order to overcome those limitations in extracting the data from web documents, an approach for extracting the most meaningful data by detecting the templates is proposed. Here, a XML web document is represented as a set of paths in a DOM tree. By simply considering the paths, overhead in measuring the similarity between web pages is reduced. The essential paths for each XML document are identified from extracted XML paths. The web documents are then clustered using Agglomerative Hierarchical Clustering method. The wrappers are generated for each clustered web documents and the data encoded in the web documents are automatically extracted. After extracting the data from the web pages, the most meaningful data are identified by comparing the extracted data with the given keyword.

2. Related Work

The motivation of our work is to extract the most meaningful data from XML web documents. In this section we survey the previously proposed approaches for the accurate extraction of data from web pages. Any XML web document can be represented as a Document Object Model tree. Many similarity based measures are there for clustering of web documents based on the similarity between trees. But tree related distance measures are very expensive for clustering.

Arasu et al. [1] proposed Extraction of Structured Data from Web Pages which deals with extracting structured data from large collections of web pages with a common template, without any human intervention. The elements of a template are detected only by the frequencies of words. Structured data is any set of data values referring to common schema. Here, all pages are assumed to have same template.

Reis et al. [4] developed a system which automatically extract web news and relevant text contents from the pages of given website using Tree Edit Distance. This is a method in which a small number of sampled documents are clustered initially and other documents are grouped to the nearest clusters. But it is not easy to select proper training data of small size. Also it is difficult to decide how many clusters are to be generated from the web documents. Crescenzi et al. [3] proposed a method for Clustering Web Pages based on their structure. The structural similarity among pages defined with respect to their DOM trees. The resulting clustering can be used to build a model that describes the structure of the site in terms of classes of pages and links among them. Pages are compared only by their link information.

Zhao et al. [8] proposed a method for Fully Automatic Wrapper Generation for Web Pages. This paper presents a technique for automatically producing wrappers which can be used to extract the data from dynamically generated result pages which is returned by search engines. Automatic data extraction is very important for wide range of applications that need to interact with search engines such as meta search engines and web crawling.

Vieira et al. [7] proposed a Fast and Robust Method for Web Page Template Detection and Removal. This paper presents a new method for detection and removal of templates which is fast and accurate as well. The patterns are identified in the detection step which is used to remove the templates in the remaining web pages in the collection. This separation leads to an efficient process. Even though the detection task can be costly, it is applied to a small number of pages. On the other hand, the template removal can also be applied to a large number of pages, which can be done through an inexpensive procedure.

Zheng et al. [9] proposed a Joint Optimization Technique which combines Wrapper Generation and Template Detection. It includes template detection and wrapper generation in a single step. It utilizes similarity between pages or any other external features to detect templates. It separates pages with notable inner differences and then generates wrappers for the web pages. The approach is more stable as it does not rely on URLs to detect templates. Also, tree related similarity measures used for clustering are very expensive.

Layaida et al. [5] developed an article about an Impact of XML Schema Evolution. This aims to cover the most general issue of schema evolution by taking into account the impact on the validity of documents. It presents a framework for checking those criteria with the schemas by specifying the main standard document formats used on the web. It also provides a unifying framework that allows for the automatic verification of properties related to XML schema evolution and its impact on the validity of documents and queries.

3. Problem Formulation

As the templates degrade the accuracy of data extraction, this proposed work mainly focuses on the extraction of most meaningful and relevant data from XML web documents to enhance the performance of web applications. Semantic searching using XML can be achieved since it is used to get the most meaningful information compared to HTML. As Minimum Description Length is used for clustering, the scalability is also achieved.

4. System Model

The Fig. 1 shows the proposed system architecture. The proposed work involves five modules. The first module is XML parsing and DOM tree generation which is explained in section 4.1. In first module, the XML web documents are parsed using XML parser and DOM tree is generated for each XML document. XML tag paths are extracted from each generated DOM tree. The second module involves in Essential Path Identification which is described in section 4.2. In which essential paths for each XML document are identified.

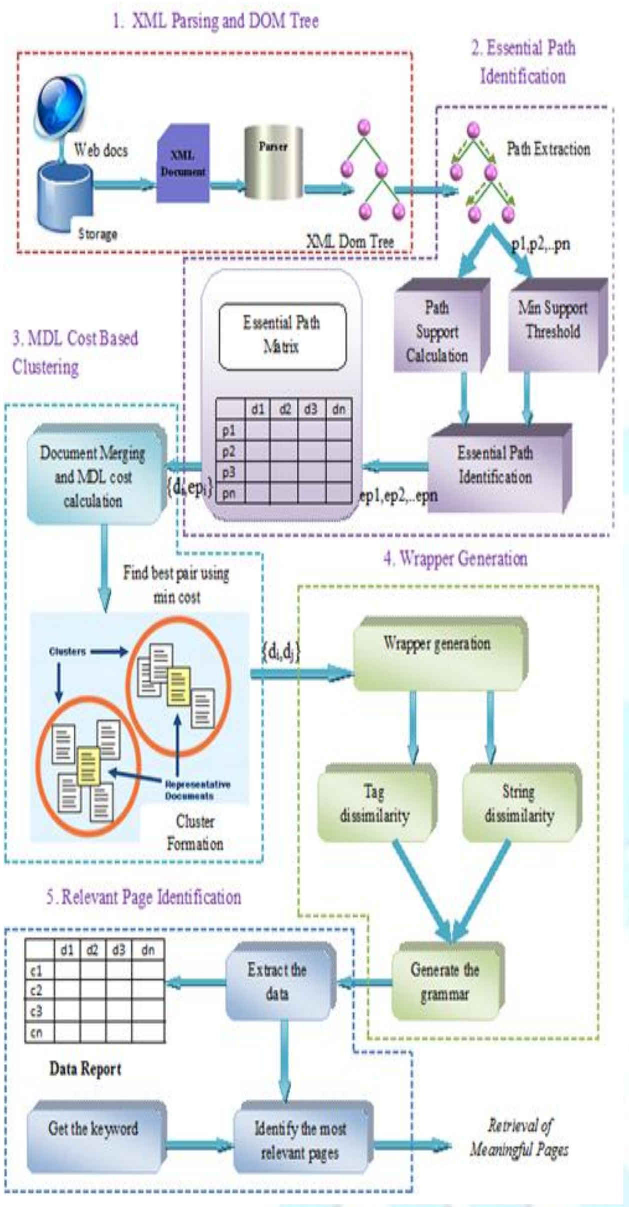

Fig. 1 Proposed System Architecture

The third module deals with MDL cost based clustering which is described in section 4.3. In which Rissanen's Minimum Description Length technique is used to cluster the similar structured XML web documents. The fourth module involves in Wrapper Generation which is explained in section 4.4. In Wrapper Generation, the wrappers are generated for each cluster. The final module is meaningful data retrieval which is explained in section 4.5. It deals with extraction of relevant data from generated wrappers.

4.1 XML Parsing and DOM Tree Generation

Document Object Model (DOM) Tree is a representation of the XML document. Also it defines the structure of each XML document. In which entire document is represented as a document node, every XML component is represented as an element node, the text in the XML elements are represented as text nodes, every attribute is represented as an attribute node. For example, consider a sample XML document in Fig. 2. DOM tree representation for the sample XML document d is given in Fig. 3.

```
<?xml version="1.0"?>

<Book_details>

<Title>Datastructures</Title>

<Author>Weiss</Author>

<Edition>II</Edition>

<Issue_date>02/08/2011</Issue_date>

</Book_details>
```

Fig. 2 Sample XML Document

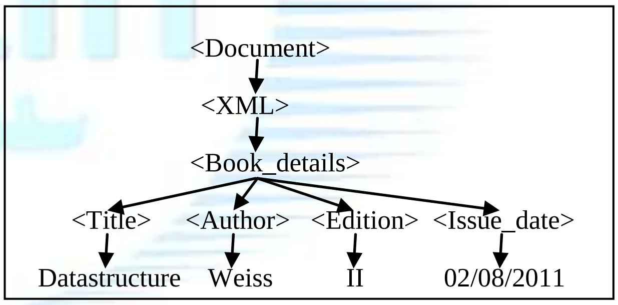

Fig. 3 DOM Tree Representation for XML Document d

The root node is the Document node. Path of each node in the DOM tree is represented by listing the nodes from root to the particular node. The symbol '|' is used as the delimiter between the nodes. For example, in the document d, path of the node *"Weiss"* is "*Document|XML|Bookdetails|Author|Weiss"*. The input XML web documents are parsed using the XML parser library and DOM tree is generated for each XML document. XML tag paths for each node in the document are extracted.

4.2 Essential Path Identification

Support Count Calculation: The algorithm 1 shows the support count calculation for each extracted XML path in each document. The support of a path is defined as the number of documents in document set D, which contains the path. If the path is already identified in the web document, the support count is increased. Otherwise, add path to the set of paths and set the support count as 1. Minimum support threshold *min _sup* is defined for each document d_i. If a path is contained by a document d_i and also the support of the path is at least the given minimum support threshold *min_sup*, the path is called as an essential path of document d_i.

Algorithm 1: Support Count Calculation

```
Input :  Document Set D
Output:  Support count value for each path in each
         document
1 begin
2     foreach document $d_i$ in D do
3         Call XML parser ($d_i$) to create DOM Tree for
          each XML document $d_i$
4         Visit all nodes of DOM Tree
5         if node visited is a Tag Node then
6             Append Tag Name to the path $p_j$
7             if path $p_j$ is already present in any
              document $d_i$ then
8                 Increment the support count for the
                  path $p_j$
9             end
10            else
11                Add path $p_j$ to the set of paths $P_D$
12                Set support count for the path $p_j$ as 1
13            end
14        end
15        else if node visited is a String Node then
16            Append String Name to the path $p_j$
17            if path $p_j$ is already present in any
              document $d_i$ then
18                Increment the support count for the
                  path $p_j$
19            end
20            else
21                Add path $p_j$ to the set of paths $P_D$
22                Set support count for the path $p_j$ as 1
23            end
24        end
25    end
26 end
```

Essential paths ignore the paths away in advance which cannot be a part of any template. The set of essential paths of an XML document d_i is denoted by E (d_i).

Essential Path Matrix Construction: The algorithm 2 shows how the essential paths for each XML document are identified and essential path matrix are constructed. Given document set D and path set P_D, an essential path matrix M_E is generated with 0/1 values. If path p_j is considered as an essential path of document, then the value of (i, j) in essential path matrix M_E is set to one. Otherwise, it sets to zero.

4.3 MDL Cost Based Clustering

The large number of XML web documents is clustered based on the similarity of templates employed in the documents. Thus the templates for every cluster are often known simultaneously.

Algorithm 2: Essential Path Matrix Calculation

```
Input :  Document Set D
Output:  Essential Path of $d_i$ in D and Essential Path
         Matrix $M_E$ for D
1 begin
2     foreach document $d_i$ in D do
3         foreach path $p_j$ in $d_i$ do
4             if sup ($p_j$) >= min_sup ($d_i$) then
5                 Set path $p_j$ as an essential path
6             end
7             else
8                 Simply ignore the paths
9             end
10        end
11    end
12    Declare matrix $M_E$ of $|P_D|$ X $|D|$
13    foreach document $d_i$ in D do
14        foreach path $p_j$ in $P_D$ do
15            if path $p_j \in$ essential path in $d_i$ then
16                Set $M_E$ [i][j] as 1
17            end
18            else
19                Set $M_E$ [i][j] as 0
20            end
21        end
22    end
23 end
```

Matrix Representation of Clustering: In a web document set D, n clusters are represented as c = (c1, c2 … c_n). A cluster c_i is represented by a pair (T_i, D_i) where T_i is a set of paths and D_i is a set of documents. A document is allowed to be in a single cluster only. To represent a cluster model C = (c1, c2 … c_n) for D, template matrix M_T and document matrix M_D are defined. Delta matrix M_Δ is constructed by adding -1/0/1 value to (M_T*M_D) which is defined in Eq. (1).

$$M_E = (M_T * M_D) + M_\Delta \qquad (1)$$

Where,
M_T - Template Matrix with its template paths
M_D - Document matrix with its documents
M_Δ - Difference Matrix with -1/0/1 value

Minimum Description Length Principle: The algorithm 3 shows a way to cluster the documents mostly based on their similarity and to decide on best cluster pair from minimized MDL cost. Rissanen's Minimum Description Length Principle [2] is used to manage the unknown number of clusters. It describes that the most effective model inferred from a given set of data is that the one which minimizes the sum of the length of the model and encoding of the data. It is referred as MDL cost of the cluster model 'C' whereas the MDL cost is that the number of bits needed to describe the data.

Algorithm 3: MDL Clustering Algorithm

```
  Input :   Document Set D, $MDLCost_{min} = \infty$
  Output:   Clustering of Documents in D
 1 begin
 2      Cluster Model C = {c1, c2 … $c_m$} with individual
        cluster $c_i = (d_i, E(d_i))$
 3      $(c_i, c_j, c_k)$ = FindBestPair (C)
 4      /* Let $c_k$ be the new cluster formed by merging
        clusters $c_i$ and $c_j$ as the best pair */
 5      while C'! = C do
 6           $(c_i, c_j, c_k)$ = FindBestPair (C')
 7           New Cluster Set C' = C $(c_i, c_j)$ U $(c_k)$
 8      end
 9      return C
10 end
11 Procedure FindBestPair (C)
12 begin
13      $MDLCost_{min} = \infty$
14      foreach pair $(c_i, c_j)$ in C do
15           MDLCost = FindMDLCost $(c_i, c_j)$
16           if $MDLCost < MDLCost_{min}$ then
17                $MDLCost_{min}$ = MDLCost
18                $c_k = (c_i, c_j)$
19           end
20      end
21      return $(c_i, c_j$, ck)
22 end
23 Procedure FindMDLCost $(c_i, c_j)$
24 begin
25      $D_k = D_i \cup D_j$
26      foreach path $p_k$ in $EP_k$ do
27           if sup $(p_k, D_k)$ >= min_sup $(D_k)$ then
28                Add path $p_k$ as an essential path $EP_k$
29                Set each $EP_k$ in $M_E$ as 1
30           end
31           else
32                Simply ignore paths
33                Set each $EP_k$ in $M_E$ as 0
34           end
35           $Ck = (D_k, EP_k)$
36           New Cluster Set C' = C - $(c_i, c_j)$ U $(c_k)$
37           L(C) = β/α (no of '1's in $M_T$ + no of '1's in
                $M_\Delta$+ no of '-1's in $M_\Delta$) + $L(M_D)$
38      end
39      return L(C)
40 end
```

The entropy H(X) of a random variable X is defined in Eq. (2). Pr(x) denotes the probability of a value x.

$$H(X) = \sum_{x \in \{-1,0,1\}} -Pr(x) \log_2 Pr(x) \qquad (2)$$

The MDL cost of a clustering C and Matrix M is denoted as L(C) and L(M). L(M) is given by $|M|*H(X)$ and $L(M_D)$ is given by $|D| \log_2 |D|$. The total MDL cost of clustering L(C) is calculated by the Eq. (3).

$$L(C) = L(M_T) + L(M_D) + L(M_\Delta) \qquad (3)$$

For a web document set D and its essential path matrix M_E, the best clustering model C can be identified from the minimized MDL cost L(C).

Agglomerative Hierarchical Clustering: The Hierarchical Clustering is a method used to build a hierarchy of clusters. The hierarchical clustering method used in the proposed work is Agglomerative Hierarchical Clustering, in which each input XML document is considered as an individual cluster initially. Whenever a pair of clusters is merged, first identify the template paths and the documents in the cluster. Then, the template matrix and document matrix are calculated. The template matrix, document matrix and the essential path matrix together determines the Delta matrix. After identifying the template matrix, document matrix and the delta matrix, the MDL Cost associated with the new cluster is calculated using the Eq. (4).

$$L(C) = \beta/\alpha \text{ (no of '1's in } M_T + \text{no of '1's in } M_\Delta + \text{no of '-1's in } M_\Delta) + L(M_D) \qquad (4)$$

Where α denotes the Pr(1) of M_E and β denotes the H(X) of M_E. The pair with maximum reduction in MDL cost is selected as the best pair to be merged. Consider an input XML document set with 5 sample web documents. The MDL Cost for best clustering model is calculated based on the Eq. (4) which is shown in Eq. (5).

$$L(C) = (0.8113/0.25)(9+0+0) + 11.61 = 40.82 \qquad (5)$$

Similarly, MDL cost can be calculated for 10, 15, 20 and 25 numbers of documents which are shown in Fig. 4. Hence, minimized MDL cost in this proposed work gives the best clustering model than the existing work.

4.4 Wrapper Generation

For each cluster, the wrappers are generated simultaneously. The wrapper is generated for a set of documents in the cluster inferring a grammar for the XML code. One input document is considered as the wrapper that is represented in the form of a regular expression. Another XML document is sample that can be parsed using the wrapper. The grammar that represents the wrapper is refined to find a common regular expression for the two documents. This is done by distinguishing the dissimilarities between the wrapper and the sample. Two kinds of dissimilarities can occur while comparing the documents. In String Dissimilarity, the text in the XML documents is dissimilar. This can be resolved by replacing the text within the wrapper by *"#PCDATA"*. In Tag Dissimilarity, the tags in the XML documents are dissimilar. Two types arise just in case of Tag Dissimilarity, Tags which are optional and Tags that appear iteratively. Whenever dissimilarity occurs, the dissimilarity is solved by generalizing the wrapper.

4.5 Meaningful Data Retrieval

The wrappers generated from every cluster are employed to extract the data present in the XML web documents. The data extracted from the web documents is compared with query value and also the web documents that contain those data are selected as the most informative pages. i.e., those web documents provide more details regarding the queried information.

5. Implementation Results

All the experiments done in the proposed work are performed on an Intel Core i3 machine with 2GB RAM, running on 32-bit Windows Operating System. All algorithms above mentioned were enforced in JAVA with JRE version 1.5 using Net beans. The input XML web documents are parsed using XMLParserv2 (http://www.java2s.com/Code/Jar/x/Downloadxmlparserv2jar.htm)

5.1 Real Life Dataset

The dataset includes XML web documents from various websites. The web documents are collected from five templates and the number of web pages from each template is 8 to 20. The total number of web documents is 54.

5.2 Performance

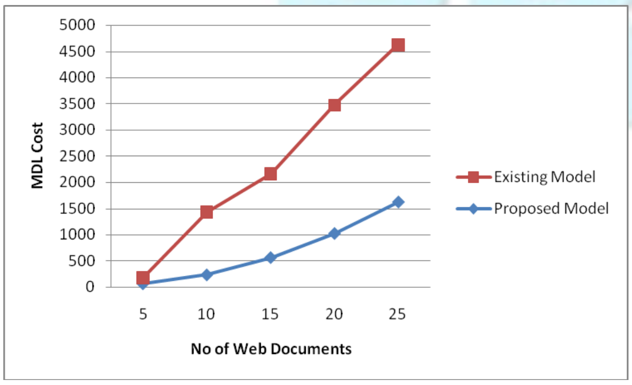

Fig. 4 Variation of MDL Cost with number of Web Documents

The variation of MDL cost with the varying number of web documents is shown in Fig. 4. The MDL Cost increases with the increase in number of documents. The approach for extracting the data and identifying the meaningful web documents is implemented with better effectiveness. The data extracted can be used for further classification of web documents and the relativity among web documents can also be measured. The web pages generated from similar templates are clustered and the data encoded in those web pages are extracted accurately. Thus the proposed work will improve the performance of search engines by retrieving the most relevant web documents.

6. Conclusion

Most of the websites contain massive set of web documents that are generated using common templates. The relevance in several web applications is affected since the templates are widely used on the web. Also, templates reduce the performance of web applications and results in wastage of resources. Therefore to avoid those issues, an approach for data extraction from heterogeneous web documents has been implemented in this proposed work.

References

[1] A. Arasu, and H. Garcia-Molina, "Extracting Structured Data from Web Pages", in Proceedings of the ACM SIGMOD International Conference on Management of Data, San Diego, USA, 2003, pp. 337-348.

[2] Chulyun kim, and kyuseok shim, "TEXT: Automatic Template Extraction from Heterogeneous Web Pages", IEEE Transactions on Knowledge and Data Engineering, Vol. 23, No. 4, 2011, pp. 612-626.

[3] V. Crescenzi, P. Merialdo, and P. Missier, "Clustering Web Pages Based on Their Structure", IEEE Transactions on Data and Knowledge Engineering, Vol. 54, No. 3, 2005, pp. 279-299.

[4] M. de Castro Reis, P. B. Golgher, A. S. da Silva, and A. H. F. Laender, "Automatic Web News Extraction Using Tree Edit Distance", in Proceedings of the 13th International Conference on World Wide Web, New York, USA, 2004, pp. 502-511.

[5] P. Geneves, N. Layaida, V. Quint, "Impact of XML Schema Evolution", ACM Transactions Internet Technology, Vol. 11, No. 1, Article 4, 2011, pp. 4.1-4.26.

[6] T. Teena Merin, and V. Vidhya, "A Novel Approach for Automatic Data Extraction from Heterogeneous Web Pages", IJCA Proceedings on Emerging Technological Trends in Advanced Engineering Research, 2012 ICETT(3):24-28, January 2013.

[7] K. Vieira, A. S. da Silva, N. Pinto, E. S. de Moura, J. M. B Cavalcanti, and J. Freire, "A Fast and Robust Method for Web Page Template Detection and Removal", in Proceedings of the 15th ACM International Conference on Information and Knowledge management, Virgina, USA, 2006, pp. 258-267.

[8] H. Zhao, W. Meng, Z. Wu, V. Raghavan, and C. Yu "Fully Automatic Wrapper Generation for Search Engines", in Proceedings of the 14th International Conference on World Wide Web, NewYork, USA, 2005, pp. 66-75.

[9] S. Zheng, D. Wu, R. Song, and J. Wen, "Joint Optimization of Wrapper Generation and Template Detection", in Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, California, USA, 2007, pp. 894-902.