An Integration Approach for Data Extraction and Coalition

P Mohammad Fayaz Ahmed \(^1\), Gunthati Prathap \(^2\),
Sri Venkateswara Institute of Science & Technology
Kadapa-516003

Abstract:

Web database is nothing but online database which generates query result based on user's result. From the query result, extract the data from many application i.e., data integration which need cooperate with multiple web application. CTVS is the method for novel data extraction, which is used to extract the data automatically from query result pages (QRP). Firstly identifying and segmenting the query result records in the QRP. Secondly, align the segmented QRR into table. Due to presence of auxiliary information, in such cases QRRs are not contiguous. Specifically for this situation we proposed new techniques to handle. Designing new record alignment algorithm which is used to align the attribute in a record, first pair wise and then holistically, by joining the tag and data value likeness information. From this approach which achieves high precision and do better than.

Introduction:

Many web applications, such as meta-querying, data integration and similarity shopping, require the data from various web databases. For these applications to additional make use of the data embedded in HTML pages, automatic data extraction is needed. Only when the data are extracted and designed in a Structured manner, such as tables, can they be evaluated and combined. Hence, perfect data extraction is vital for these applications to perform correctly.

This paper focuses on the problem of automatically extracting data records that are determined in the query result pages generated by web databases. In broad-spectrum, a query result page contains not only the real data, but also other information, such as navigational panels, advertisements, remarks, information about hosting sites, and so on. The objective of web database data extraction is to take out any isolated information from the query result page, extract the query result records from the page, and align the extorted QRRs into a table such that the data values fit in to the same attribute are placed into the same table column.

We utilize the following two-step method, called Combining Tag and Value Similarity (CTVS)[1], to dig up the QRRs from a query result page.

QRR EXTRACTION:

Given a query result page, the Tag Tree Construction module first builds a tag tree for the page rooted in the <HTML> tag. Each node signifies a tag in the HTML page and its children are tags covered inside it. Each internal node n of the tag tree has a tag string tsn, which includes the tags of n and all tags of n’s descendants, and a tag path tpn, which contains the tags from the root to n. Next, the Data Region Identification module recognizes all possible data regions, which habitually contain dynamically, generated data, top down opening from the root node. The Record Segmentation module is used to pieces the recognized data regions into data records according to the tag patterns in the data regions. Given the segmented data records, the Data region merge module which combines the data regions containing like records. Finally, the Query Result Section Identification module selects one of the merged data regions as the one that holds the QRRs.

QRR Alignments:

QRR alignment is executed by a novel three-step data alignment method that combines tag and value similarity. 1. Pair wise QRR alignment aligns the data values in a pair of QRRs to offer the confirmation for how the data values should be aligned among all QRRs. 2. Holistic alignment aligns the data values in all the QRRs. 3. Nested structure processing identifies the nested structures that are present in the QRRs.

Related work:

Web database extraction has inward much attention from the Database and Information Extraction research areas in recent years due to the volume and quality of deep web data. As the arrived data for a query are embedded in HTML pages, the research has
purposeful on how to extract this data. Previous works pay attention on wrapper induction methods, which need human support to construct a wrapper. More newly, data extraction methods have been planned to automatically extract the records from the query result pages.

Table 1: Data Extraction Performance accomplished for the NESTED Data Set

<table>
<thead>
<tr>
<th>Method</th>
<th>CTVS</th>
<th>VnTs</th>
<th>DeLa</th>
<th>CTVS</th>
<th>VnTs</th>
<th>DeLa</th>
</tr>
</thead>
<tbody>
<tr>
<td># QRRs</td>
<td>420</td>
<td>420</td>
<td>414</td>
<td>420</td>
<td>414</td>
<td>414</td>
</tr>
<tr>
<td># extracted QRRs</td>
<td>421</td>
<td>420</td>
<td>414</td>
<td>420</td>
<td>414</td>
<td>414</td>
</tr>
<tr>
<td># correct extracted QRRs</td>
<td>411</td>
<td>409</td>
<td>398</td>
<td>420</td>
<td>388</td>
<td>417</td>
</tr>
<tr>
<td>record-level precision</td>
<td>97.6%</td>
<td>95.2%</td>
<td>91.1%</td>
<td>95.9%</td>
<td>86.4%</td>
<td>93.7%</td>
</tr>
<tr>
<td>record-level recall</td>
<td>97.9%</td>
<td>95.2%</td>
<td>91.7%</td>
<td>95.3%</td>
<td>87.8%</td>
<td>92.9%</td>
</tr>
<tr>
<td>page-level precision</td>
<td>95%</td>
<td>90%</td>
<td>90%</td>
<td>90%</td>
<td>75%</td>
<td>85%</td>
</tr>
</tbody>
</table>

While wrapper induction has the advantage that no irrelevant data are extracted, since the user can labeled only the items of interest, it needed labor intensive, time-consuming and manual tagging of data. Thus, it is not scalable to a huge number of web databases. Moreover, presented wrapper can execute weakly when the arrangement of a query result page altered, which may happen regularly on the web. Hence, the wrapper induction comes close to involves two further difficult problems: observing changes in a page’s format and maintaining a wrapper when a page’s format changes.

To overcome the problems of wrapper induction, some unconfirmed knowledge methods, such as Roadrunner, Omini, IEPAD [8], ExAlg [1], DeLa, PickUp, and TISP, have been proposed to automatically extract the data from the query result pages. These methods rely completely on the tag structure in the query result pages. Here, we talk about only DeLa since we evaluate its performance with CTVS.

<table>
<thead>
<tr>
<th>Nested Structure Processing</th>
<th>Single Result Page</th>
<th>Non-contiguous Data Regions</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTVS</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>DeLa</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>VnTs</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>ViPER</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

Table 2: Data Extraction Method for Summarization

Proposed methods:

The goal of web database data extraction is to remove any irrelevant information from the query result page, extract the query result records from the page, and aligns the extracted QRRs into a table such that the data values belonging to the same attribute are placed into the same table column. In proposed Search data is extracted from multiple pages and orderly in a structured manner.

We utilize the following two-step method, called Combining Tag and Value Similarity (CTVS), to extract the QRRs from a query result page p.

- Record extraction recognizes the QRRs in p and involves two sub steps: data region identification and the actual segmentation step.
- Record alignment supports the data values of the QRRs in p into a table so that data values for the same attribute are aligned into the same table column.

Architecture:

Given a query result page, the Tag Tree Construction module first builds a tag tree for the page entrenched in the <HTML>tag. Each node represents a tag in the HTML page and its children are tags covered inside it. Each internal node n of the tag tree has a tag string tsn, which comprises the tags of n and all tags of n’s descendants, and a tag path tpn, which consists of the tags from the root to n. Next, the Data Region Identification module identifies all possible data regions, which usually contain dynamically, produced the data, top down opening from the root node. The Record Segmentation module is segments the recognized data regions into data records according to the tag patterns in the data regions. Given the segmented data records and then the Data Region Merge module is used merges the data regions containing similar records. The Query result section identification module which selects one of the merged data regions has the one that contains the QRRs.
Algorithm:
Function Traverse(G)
Input: pair-wise alignment graph G
Output: a list C in which each element is a connected component in G with its corresponding breach flag
1. for each vertex u ∈ V(G)
2. color[u] := WHITE // unvisited
3. i = 0
4. for each vertex u ∈ V(G)
5. if color[u] = WHITE then
6. i = i + 1
7. C[i].vertices = {u} // start a new component
8. C[i].flag = Visit(u, C[j])
9. for each component C[i] in C
10. if C[i].flag = true then
11. BreakBreathPath(C[i])
12. update list C // since some C[i] will be broken into smaller sub-parts
13. return C

Function Visit(u, C[i])
Input: a vertex u in G and the connected component that u is in
Output: true or false value indicating whether the component contains breach paths
1. breach = false
2. color[u] = GREY // visited and under processing
3. for each vertex v ∈ Adj(u) // v is a neighbor of u
4. if color[v] = WHITE then
5. C[i].vertices = C[i].vertices + {v}
6. for each vertex w ∈ R(v), w • v // in the same record
7. if color[w] = GREY or BLACK then
8. breach = true
9. breach = breach OR Visit(v, C[i])
10. color[u] = BLACK // processing finished
11. return breach

Function BreakBreathPath(g)
Input: connected graph g
Output: updated g containing no breach paths
1. for i = 1, 2, ..., |V(g)| - 1
2. for j = i+1, i+2, ..., |V(g)|
3. if R(v) = R(v) then // in the same record
4. MinCut (v, v, g)

Function HolisticAlign(G, T)
Input: the pair-wise alignment graph G
Output: the holistically aligned table T
1. C_{left} = C_{right} = ∅
2. if C is not empty then
3. find the component C[i] in C which has the maximum number of edges
4. insert a globally aligned column in T by aligning all vertices in C[i] in the column
5. for each component C[k] in C, k ∈ i
6. if C[k] intersects with C[i] then
7. C_{left} = vertices of C[k] that are on the left side of C[i]
8. C_{right} = vertices of C[k] that are on the right side of C[i]
9. remove edges connecting C_{left} and C_{right}
10. C_{left} = C_{left} + C[k]
11. C_{right} = C_{right} + C[k]
12. else if C[k] is on the left side of C[i] then
13. C_{left} = C_{left} + C[k]
14. else
15. C_{right} = C_{right} + C[k]
16. HolisticAlign(C_{left}, T)
17. HolisticAlign(C_{right}, T)

Conclusion:
We obtain novel data extraction methods, CTVS, to automatically extract QRRs from a query result page. CTVS utilizes two steps for this job. The first step recognizes and segments the QRRs. We get better on existing techniques by allowing the QRRs in a data region, to be not contiguous. The second step aligns the data values among the QRRs. A novel alignment method is wished-for in which the alignment is performed in three following steps: pair-wise alignment, holistic alignment, and nested structure processing. Tests on five data sets show that CTVS[2] is usually more precise than present state-of-the-art methods.
References: