HTML table wrapper based on table components

Detty Purnamasari*, I. Wayan Simri Wicaksana, Suryadi Harmanto and Lintang Yuniar Banowosari

Information System, Gunadarma University, Jl. Margonda Raya No. 100 Pondok Cina Depok, Indonesia
Email: detty@staff.gunadarma.ac.id
Email: iwayan@staff.gunadarma.ac.id
Email: suryadi@staff.gunadarma.ac.id
Email: lintang@staff.gunadarma.ac.id
*Corresponding author

Abstract: Tables are a model for data representation in the internet. Many approaches to harvesting table data are executed by doing the copy-paste. However, this method will be a problem if there is a huge amount of tables and they come from many internet sources. This paper presents an approach to prepare the table area and to wrap or extract table components in cells and property from HTML tables. This paper discusses how the approach works by testing Algorithms 1, 2, and 3. Algorithm 1 is used to determine the actual number of columns and rows of the table, and Algorithm 2 is used to determine the boundary line of the property. At the end of the process of extraction, Algorithm 3 is implemented to get content of the table. Tests were conducted at 100 tabular HTML format. The result of F-measure for Algorithm 1 is 100.00%, for Algorithm 2 97.67% and for Algorithm 3 94.91%.

Keywords: computer application; extraction algorithm; html; table wrapper; table property.


Biographical notes: Detty Purnamasari holds a Bachelor degree in Information System and Master’s degree in Accounting Information System from the Gunadarma University Jakarta, Indonesia. She is a staff of secretary at Gunadarma University and finished the PhD in Information Technology in the same university in 2013. Her research topic is about information system and the main focus is how to collect data in internet. She is a Member of PIKIN organisation, which is an organisation in Indonesia for computer users and a Member of APsiCI (organisation for Psychology and Cyber in Indonesia).

I. Wayan Simri Wicaksana obtained his Bachelor degree in Physic University of Indonesia at 1988, continued his Master program in Computer Integrated Manufacturing Swinburne University at Melbourne, Australia, and finished his Doctoral program in Informatics from the University of Dijon in France and Gunadarma University in Jakarta. He is a Professor in Information Technology and his main research is in the areas of database, semantic web, interoperability, and ontology. Currently, he is a Member of Computer Professional Association (IPKIN – Ikatan Profesi Komputer Informatika Indonesia), Indonesia Health Informatics Association (PIKIN – Perhimpunan Informatika Kesehatan Indonesia), Indonesia Cyber Psychology Association (APsiCI –Asosiasi Psikologi Cyber Indonesia) and Indonesia Informatics and Computer Higher Education Association (APTIKOM – Asosiasi Perguruan Tinggi Komputer Indonesia).

Suryadi Harmanto has obtained his Bachelor degree in Mathematics from the University of Indonesia and Master in Information System from Gunadarma University. He is a Professor in Computer Science. He is Founder of Indonesia Health Informatics Association (PIKIN – Perhimpunan Informatika Kesehatan Indonesia), and Indonesia Cyber Psychology Association (APsiCI –Asosiasi Psikologi Cyber Indonesia), and Indonesia Informatics.

Lintang Yuniar Banowosari holds a Bachelor degree in Information Management from Gunadarma University and Master degree in Computer Science from Asian Institute of Technology Bangkok, and Doctor in Computer Science/Information Technology from Gunadarma University. She is a Lecturer in Computer Science and Information Technology Faculty. Her research preferences are information system development, information
1 Introduction

Data can be represented in many forms; tables are a common model on the internet, and can be provided using copy-paste process. However, this process will be complicated owing to huge number of tables that come from many sources. This paper explains an approach to extract HTML tables from the internet. The first algorithm is to obtain the number of column and row, the second algorithm is to evaluate the border of row and property, and the third algorithm is to grab content of the table.

Extraction or wrapper is part of the application that makes the web resource become a resource that can be queried since it is in the form of a database, where the source is either semi-structured or unstructured (Lerman et al., 2001).

A table considered as a representation of structured data and related information in the form of two dimensions. Liu et al. (2008) state that the contents of the table are the data that are presented briefly. The table consists of cells, which can contain cell label and data (Tengli et al., 2004).

Research on the extraction table has been done by Tengli et al. (2004), which distinguishes the cell content by identifying the cell as property and the cell as an instance, and in this case the property is located in the first row of the table.

Biological data can be extracted from a table in two ways (Wong et al., 2009):

- table detection (table identification of documents)
- processing table (it extracts data from tables).

Detection of property is conducted by considering <hr> tag which is used to create a horizontal line in HTML.

Document object model (DOM) is also used by some researchers to perform the extraction of HTML tables (Lin et al., 2009; Gultom et al., 2011). DOM is a base or a stand-alone language used to represent and make the connection between objects of different documents into HTML or XML webpage, with the form of a tree structure as depiction (Krupl et al., 2005). Combine DOM with XY cut algorithm to find a table on the webpage.

Search the schema matching on the extraction of HTML tables made to merge the results of extraction, but the research on this area found several problems including the table location in webpages, merged attribute, and word synonyms in the process of data extracted merging (Embley et al., 2006).

Our approach starts with preparation for finding areas (arrays) on the extraction of HTML tables with respect to the position of property that could be more than one row in the table (Figure 1).

Figure 1 shows the table property in row positions, which are row 1 and row 2. Property could be in more than one row because of the merging of rows and columns. Figure 1 shows a merging of the first and second column ‘Name’ and the merging of rows 1 and 2 ‘Phone Number’. Then it shows the content/data table (instance) are from row 3 to row 7.

The paper is divided into four parts: the first part is an introduction that contains the definition of the problem and refers to previous similar research papers. The second part discusses the approach taken to perform the extraction of HTML table that contains three algorithms, and the third part contains testing of the algorithm, and the final part contains the conclusions and future work.

2 Table extraction approach

A property and data cell are components of a table. Property is title of column and data cell is content or instance or record of table. Relation in row-column as position pointer.

Referring to type of table, extraction process needs to identify the area of property and cell. The first step is to calculate number of rows and columns in a table. The second step is to consider which row or column as property cell. After 1st and 2nd algorithm are executed, 3rd algorithm can be implemented to get the content of cell.

1st algorithm is used to calculate the actual number of columns and rows. This is done since if the table has a property of more than one row, then there is a merging of rows and columns in the property cell of the tables, so we need an algorithm to calculate the actual number of columns and rows of the table. The calculation of the number of columns and rows is useful to know the size of the table (row × columns).
Algorithm 1 Count for Actual Number of Row and Actual Number of Column

Read HTML
s = 0
For a = 1st HTML line to endline do
If read <tr> then s = s + 1
RsTotal = s
Next a;
Jum = count tag <td>...<td> ...<td> in first tag <tr>...
CsTotal = 0
For i = 1 to jum:<td>
Read value cs(i)
CsTotal = CsTotal + cs(i)
Next i;

RsTotal: the total number of <tr> tag on tag <table>...<table>
CsTotal: the number of colspan value
Jum:<td>: the number of <td> tag
cs: colspan value; i: <td> tag; s: <tr> tag

Algorithm 2 is used to find the value of the row boundary as property (rowmax_pro) by finding the largest value of rowspan (RsMax) present in each ith tag <td>...<td> on the sth tag <tr>...<tr>. If it is not found rowspan value > 1 any longer, then the row boundary of the property is found. Figure 3 is an illustration of how to run the Algorithm 2.

Figure 2 is an illustration of how to run the Algorithm 1. In Figure 2 looks <HTML> tag of the example table in Figure 1 with the actual number of rows = 7, which is calculated by summing the <tr> tag inside the <table>...<table> tag. The actual number of columns = 3 is calculated by looking at the number of <td> tags inside the first <tr>...<tr> tag and if there is colspan in it, then it calculates the number of colspan.

Figure 2 Illustration for Algorithm 1 (see online version for colours)
Algorithm 2: Finding the Largest Rowspan Value, and Number of Row to be the Property Boundary

\[
\begin{align*}
\text{mBatas (0) = 1} \\
\text{while } s = 1 \text{ do} \\
\quad \text{rsMax (0) = 1} \\
\quad \text{Count jump\langle td\rangle} \\
\quad \text{For } i = 1 \text{ to jump\langle td\rangle} \\
\text{\quad If } rs (i) > 1 \text{ then} \\
\text{\quad \quad If } rs (i) > = \text{rsMax (i-1)} \text{ then } \text{rsMax (i) = rs (i)} \\
\text{\quad \quad else} \\
\text{\quad \quad If } rs (i) < \text{rsMax (i-1)} \text{ then } \text{rsMax (i) = rsMax (i-1)} \\
\text{\quad Next } i; \\
\text{mBatas (s) = rsMax (i) + s - 1} \\
\text{if mBatas (s) < mBatas (s-1) then mBatas (s) = mBatas (s-1)} \\
\text{\quad s = s+1} \\
\text{until mBatas (s);} \\
\text{rowmaxpro = mBatas (s);}
\end{align*}
\]

RsMax: the highest value of rowspan
mBatas: row value boundary as property
rowmax_pro: row boundary which named as property
i: \langle td\rangle tag; s: \langle tr\rangle tag

Figure 3 shows an \langle HTML\rangle tag from the example of table in Figure 1; they are the tags for row 1 up to has 3. \langle td\rangle tag existing on the first \langle tr\rangle ... \langle /tr\rangle tags have the largest rowspan value which is 2, so the row boundary of the table property exists in a table on the 2nd row, because once doing a reading for a \langle td\rangle tag that is inside the second \langle tr\rangle ... \langle /tr\rangle tags is no longer found any rowspan.

After obtaining the actual table size (row \times columns) and the boundary row of the table property, the next step is to get content of table that is property table (3rd algorithm).

After 2nd algorithm was executed, border for number of row for property table, in illustration is 2nd row, so rowmaxpro = 2 (see Figure 3). Value of rowmaxpro = 2 that is used in 3rd algorithm in s variable.

Algorithm 3 has decreased iteration (for-or-down to-do) that start from row border as property in 2nd algorithm (value of rowmaxpro) until first row. Value of colspan (cs) will consider in position of column of data that will be extracted.

\[
\begin{align*}
\text{Column 1} & \quad \text{Column 2} & \quad \text{Column 3} \\
\text{Name} & \quad \text{First} & \quad \text{Middle} & \quad \text{Phone} & \quad \text{Number} \\
\text{Luciana} & \quad \text{Lee} & \quad 3972896 \\
\text{Fuchsia} & \quad \text{Noor} & \quad 3891810 \\
\text{Lyra} & \quad \text{Wid} & \quad 1290169 \\
\text{Sarah} & \quad \text{Melany} & \quad 4025049 \\
\text{Juwita} & \quad \text{Osta} & \quad 3902421
\end{align*}
\]
Algorithm 3. Get the Property

\[
\begin{align*}
\text{\texttt{s = rowmaxpro}} \\
\text{\texttt{for i = 1 to jum<\texttt{td}>}} \\
\text{\texttt{TdVal(s, i) = value in tag \{\texttt{<td>...</td>\}};}} \\
\text{\texttt{Next i;}} \\
\text{\texttt{for s = rowmaxpro-1 down to 1}} \\
\text{\texttt{PosisiCol(0) = 1}} \\
\text{\texttt{jumAnggota(0) = 0}} \\
\text{\texttt{c = 1}} \\
\text{\texttt{for i = 1 to jum<\texttt{td}>}} \\
\text{\texttt{if cs = 1 then}} \\
\text{\texttt{PosisiCol(i) = (jumAnggota(i-1) + PosisiCol(i-1) - 1) + 1}} \\
\text{\texttt{jumAnggota(i) = cs(i)}} \\
\text{\texttt{TdVal(s, PosisiCol(i)) = value in tag \texttt{<td>...</td>}}} \\
\text{\texttt{if cs > 1 then}} \\
\text{\texttt{PosisiCol(i) = (jumAnggota(i-1) + PosisiCol(i-1) - 1) + 1}} \\
\text{\texttt{jumAnggota(i) = cs(i)}} \\
\text{\texttt{TdVal(s, PosisiCol(i)) = value in tag \texttt{<td>...</td>}}} \\
\text{\texttt{for j = 0 to jumAnggota(i) - 1}} \\
\text{\texttt{TdVal(s, PosisiCol(i) + j) =}} \\
\text{\texttt{TdVal(s, i) concat TdVal(s + 1, c);}} \\
\text{\texttt{c = c + 1}} \\
\text{\texttt{Next j;}} \\
\text{\texttt{Next i;}} \\
\text{\texttt{Next s;}} \\
\text{\texttt{s = 1}} \\
\text{\texttt{PosisiCol(1) as property}} \\
\text{\texttt{Next i;}} \\
\end{align*}
\]

rowmax_pro: row border that as property
TdTVal: variable that used to save value of cell content in tag \texttt{<td>...</td>}
PosisiCol: column position for \texttt{p} and row \texttt{2p}
JumAnggota: number of column in colspan.

Figure 4 is illustration for 3rd algorithm with example for table in Figure 1. Value of rowmaxpro from 2nd algorithm is used for 3rd algorithm. Value between tag \texttt{<td>...</td> in tag \texttt{<tr>}...\texttt{</tr>} as beginning value for rowmaxpro will be used; second repetition is executed to get initial value of \texttt{s = rowmaxpro-1}. At second repetition have two conditions, if \texttt{cs = 1} and \texttt{cs > 1}. So value for tag \texttt{<td>...</td> in tag \texttt{<tr>}...\texttt{</tr>} at (rowmaxpro-1) will be linked with value of tag \texttt{<td>...</td>at tag \texttt{<tr>}...\texttt{</tr>} to \texttt{–rowmaxpro}.

The approach can be implemented in many areas. For instance in manufacturing area, the main issue is to find a raw material. Currently many suppliers provide the information in internet by using table form. Automotive industry needs to have spare parts, such as tyre, engine, audio system etc. There are many products of audio system that inform by table in internet. To find the appropriate spare parts the manufacturer will copy many tables from many suppliers of audio system. The next step, the content of table will merge to become one single table. This effort is acceptable for limited amounts of data and sources. If the amounts of data and sources are huge, it can be hard effort and difficult to avoid an error. By implementing these two algorithms, the automatic process of data harvesting process in table of internet can be performed.

3 Experiment

Both algorithms in this paper are implemented using the Python programming language. The test preparation provides a table with various forms in HTML format totalling 100 tables. Tables used for evaluation in this case are in three models:

- table has property at the row and at first row
- table has property at right side of table
- table has no property.

The evaluation was implemented in that model.

Figure 5 shows one form of tables that are used as test inputs. The table in Figure 5 is a complex table which has property in the first until third row with some row consisting of span row.

The purpose of testing the Algorithm 1 is to determine the ability of the algorithm to count the number of rows and number of columns of the table, while the second algorithm testing aims to determine the ability of the algorithm to find the limits of the table properties ranging from row 1 to a particular row number in the table. Testing the third algorithm is to evaluate the ability to capture content of cell of the table. Here are the test scenarios:

- table has property at the row and at first row
- table has property at right side of table
- table has no property.

The evaluation was implemented in that model.
• Algorithm 1, Algorithm 2, and Algorithm 3 are written in the Python programming language.

• Input is HTML tables with different forms.

Output measured is accuracy of the actual number of rows and columns of the table that will be extracted (from Algorithm 1), and the accuracy of determining the row boundary of the table property (from Algorithm 2). Algorithm 3 considers value and position of cell.

Evaluation was conducted to find out the accuracy of each algorithm by using value of precision, recall, and F-measure. Precision is calculated by amount of relevant data divided by retrieved data. Recall is calculated by relevant data of algorithm divided by relevant manual process. F-measure is calculated from the average recall and precision value.

Table 1 shows a summary of the test results of Algorithm 1 to determine the actual number of rows and columns.

Table 2 shows the testing summary of Algorithm 2 to determine the accuracy of the table to determine the property boundaries on what row.

Table 3 shows the testing summary of Algorithm 3 to determine the accuracy of the table to determine the property boundaries on what row.

Figure 5 Examples of complex table form for trial (see online version for colours).

### Table 1 Testing summary of Algorithm 1

<table>
<thead>
<tr>
<th>Table model</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>F-Measure (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>2</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>3</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>4</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>5</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>100</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Average (%) 100.00

Table 1 is a summary of evaluation test by comparing the number of columns and rows in real table compared to the result of 1st algorithm. The result of F-measure is 100%.

Then Table 2 shows a summary test of Algorithm 2 to determine the accuracy of the table to determine the property boundaries on what row.

Table 2 shows the table which has F-measure <100.00%, the value is 80.00% (example from table model 40). The reason is that there are some blank cells. Result of 2nd algorithm with F-measure 97.67%. Summary test result for 3rd algorithm shown in Table 3.

### Table 2 Testing summary of Algorithm 2

<table>
<thead>
<tr>
<th>Table model</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>F-Measure (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>2</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>3</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>4</td>
<td>67.00</td>
<td>100.00</td>
<td>80.00</td>
</tr>
<tr>
<td>5</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>99</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>100</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Average (%) 97.67

### Table 3 Testing summary of Algorithm 3

<table>
<thead>
<tr>
<th>Table model</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>F-Measure (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>2</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>3</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>4</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>5</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>33</td>
<td>28.57</td>
<td>33.33</td>
<td>30.77</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>99</td>
<td>77.77</td>
<td>77.77</td>
<td>77.77</td>
</tr>
<tr>
<td>100</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
</tr>
</tbody>
</table>

Average (%) 94.91%

Table 3 shows the average for F-measure is 94.91%. From the evaluation, the algorithm has no optimal result in table with empty cell.

### 4 Conclusions

1st and 2nd algorithms are precondition algorithms that should be executed before 3rd algorithm (sequence process). The purpose is to be able extract content of table cell refer to property. Result of average F-measure 1st algorithm is 100.00%, 2nd algorithm is 97.67%, and 3rd algorithm is 94.91%.

Compared with Tengli et al. (2004) with similar main with result of F-measure is 91.42%. Referring to the result, the approach can provide better results compared to Tengli et al. (2004). For a subsequent study, we will use the Algorithm 1 and Algorithm 2 to conduct the cell contents an instance in the table extraction process.

### Acknowledgement

This work is partially supported by Gunadarma University and the Doctoral Program of Information Technology at Gunadarma University, Jakarta, Indonesia.
References


