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.

Reference to this paper should be made as follows: Purnamasari, D., Simri Wicaksana, I.W., Harmanto, S. and Banowosari, L.Y. (2015) 'HTML table wrapper based on table components', *Int. J. Computer Applications in Technology*, Vol. 52, No. 4, pp.237–243.

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

interoperability, semantic web, and ontology. She is also a Member of Computer Professional Association (IPKIN – Ikatan Profesi Komputer Informatika Indonesia), and, Indonesia Cyber Psychology Association (APsiCI – Asosiasi Psikologi Cyber Indonesia).

This paper is a revised and expanded version of a paper entitled 'HTML extraction algorithm based on property and data cell' presented at *International Conference on Manufacturing, Optimization, Industrial and Material Engineering (MOIME 2013)*, Bandung-West Java, Indonesia, 10 March, 2013.

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.

Figure 1 Examples of table with property more than 1 row

Name		Phone Number	
First	Middle		Property
Lussiana	Lee	8972898	
Finisha	Noor	3891810	
Lyre	Widi	1290109	Instance
Susan	Melany	4029049	
Juwita	Okta	3920421	

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. Content of table have 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).

Algorithm1 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 <td> = count tag <td>...</td>  in first tag <tr>...</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

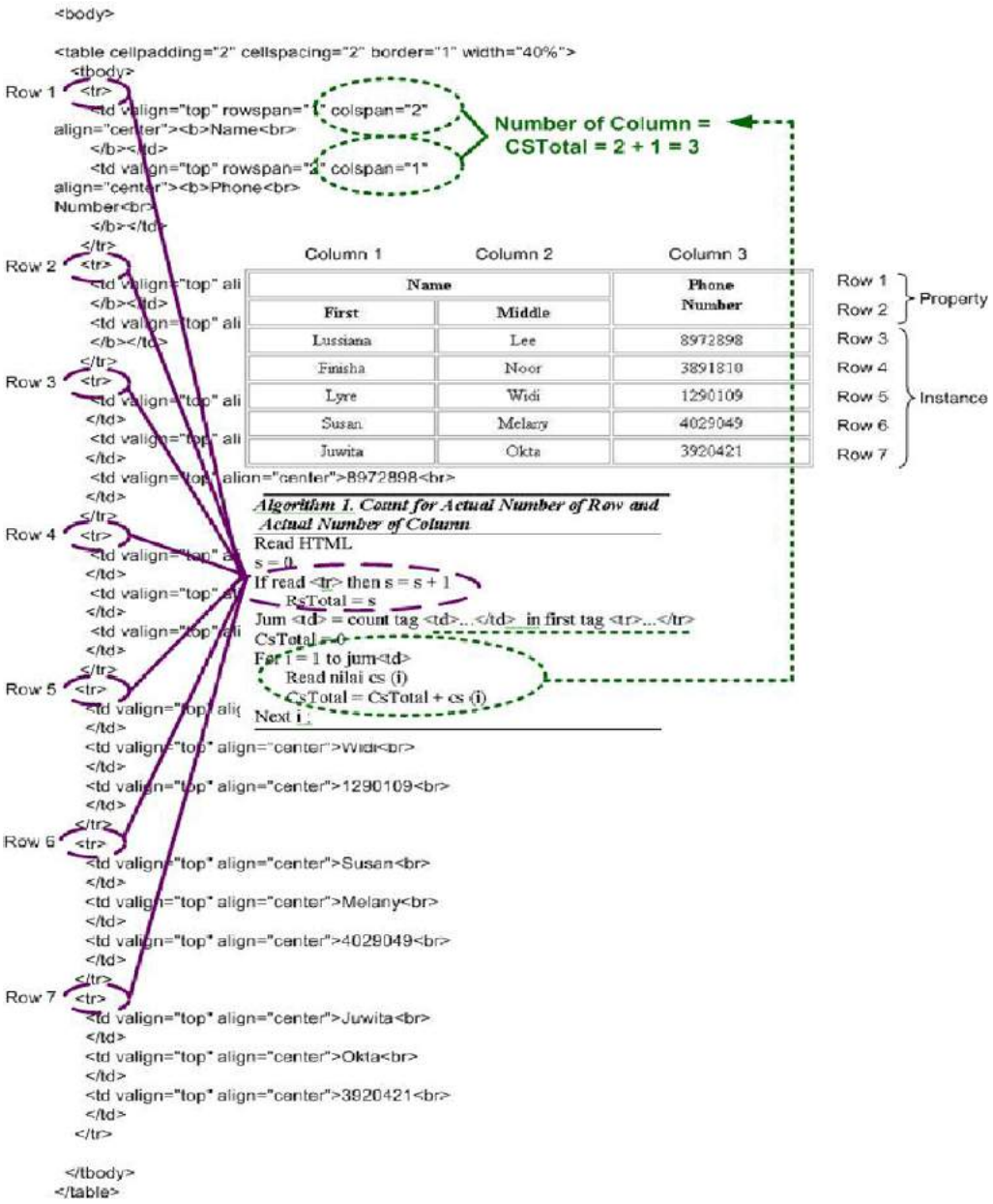
Jum<td>: the number of <td> tag

cs: *colspan* value; i: <td> tag; s: <tr> tag

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*.

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 s^{th} 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 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

```
mBatas (0) = 1
while s = 1 do
    rsMax (0) = 1
    Count jum<td>
    For i = 1 to jum<td>
        If rs (i) > 1 then
          If rs (i) > = rsMax (i-1) then rsMax (i) = rs (i)
          else
          If rs (i) < rsMax (i-1) then rsMax (i) = rsMax (i-1)
     Next i ;
mBatas (s) = rsMax (i) + s – 1
if mBatas (s) < mBatas (s-1) then mBatas (s) = mBatas (s-1)
s = s+1
until mBatas (s) ;
rowmaxpro = mBatas (s) ;
```

RsMax: the highest value of *rowspan*

mBatas: row value boundary as property

rowmax_pro: row boundary which named as property

i: <td> tag; s: <tr> tag

Figure 3 shows an <HTML> tag from the example of table in Figure 1; they are the tags for row 1 up to has 3. <td> tag existing on the first <tr> … </tr> 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 <td> tag that is inside the second <tr> … </tr> tags is no longer found any *rowspan.*

After obtaining the actual table size (row × 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 (fsor-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.

Figure 3 Illustration for Algorithm 2 (see online version for colours)

```
<body>

<table cellpadding="2" cellspacing="2" border="1" width="40%">
  <tbody>
Row 1 <tr>
    <td valign="top" rowspan="1" colspa
align="center"><b>Name<br>
    </b></td>
    <td valign="top" rowspan="2" colspa
align="center"><b>Phone<br>
Number<br>
    </b></td>
  </tr>
Row 2 <tr>
    <td valign="top" align="center"><b>First<br>
    </b></td>
    <td valign="top" align="center"><b>Middle<br>
    </b></td>
  </tr>
Row 3 <tr>
    <td valign="top" align="center">Lussiana<br>
```

(A) (B)

	Column 1	Column 2	Column 3	
Row 1	Name		Phone Number	Property
Row 2	First	Middle		
Row 3	Lussiana	Lee	8972898	Instance
Row 4	Finisha	Noor	3891810	
Row 5	Lyre	Widi	1290109	
Row 6	Susan	Melany	4029049	
Row 7	Juwita	Okta	3920421	

Algorithm 2. Finding the Largest Rowspan Value, and Number of Row to be The Property Boundary

```
mBatas (0) = 1
while s = 1 do
    rsMax (0) = 1
    Count jum<td>
    For i = 1 to jum<td>
        If rs (i) > 1 then
          If rs (i) > = rsMax (i-1) then rsMax (i) = rs (i)
          else
          If rs (i) < rsMax (i-1) then rsMax (i) = rsMax (i-1)
     Next i ;
mBatas (s) = rsMax (i) + s – 1
if mBatas (s) < mBatas (s-1) then mBatas (s) = mBatas (s-1)
until mBatas (s) ;
rowmax_pro = mBatas (s) ;
```

1st tag <tr> → s = 1 :
(A). i = 1; Rs (1) = 1; 1 >= 1; rsMax (1) = 1
(B). i = 2; Rs (2) = 2; 2 >= 1; rsMax (2) = 2
mBatas (1) : 2 + 1 -1 = 2
Rowmax_pro = 2

Algorithm 3. Get the Property

```
s = rowmaxpro
For i = 1 to jum<td>
  TdVal (s, i) = value in tag {<td>…</td>}i
Next i;
For s = rowmaxpro–1 down to 1
  PosisiCol (0) = 1
  jumAnggota (0) = 0
  c = 1
  For i = 1 to jum<td>
    If cs = 1 then
      PosisiCol (i) = (jumAnggota (i-1) + PosisiCol (i-1) – 1 ) + 1
      jumAnggota (i) = cs (i)
      TdVal (s, PosisiCol(i)) = value in tag <td>…</td>
    If cs > 1 then
      PosisiCol (i) = (jumAnggota (i-1) + PosisiCol (i-1) – 1 ) + 1
      jumAnggota (i) = cs (i)
      TdVal (s, i) = value in tag <td>…</td>
      For j = 0 to jumAnggota (i) - 1
        TdVal (s, PosisiCol(i) + j) =
        TdVal (s, i) concat TdVal (s + 1, c) ;
        c = c + 1
      Next j ;
  Next i;
Next s;
s = 1
For i = 1 to CsTotal
  TdVal (s, i) as property
Next i;
```

rowmax_pro: row border that as property

TdVal: variable that used to save value of cell content in tag <td>…</td>

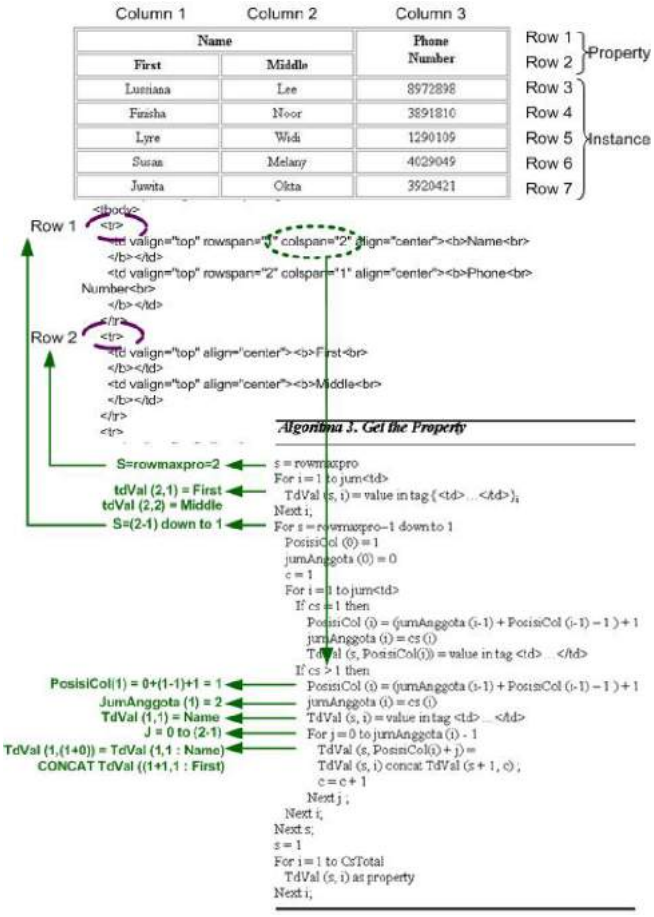
PosisiCol: column position for p i and row 2p

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 <td>…</td> in tag <tr>…</tr> as beginning value for rowmaxpro will be used; second repetition is executed to get initial value of s = rawmaxpro-1. At second repetition have two conditions, if cs = 1 and cs > 1. So value for tag <td>…</td> in tag <tr>…</tr> at (rowmaxpro-1) will be linked with value of tag <td>…</td>at tag tag <tr>…</tr> to –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.

Figure 4 Illustration for Algorithm 3 (see online version for colours)



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:

- 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.

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

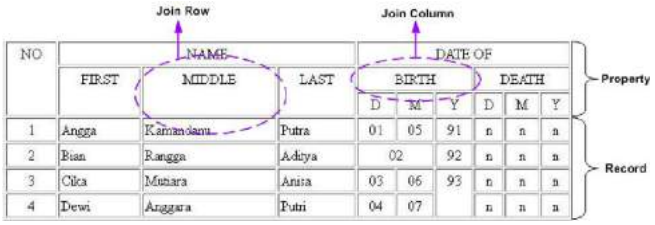


Table 1 Testing summary of Algorithm 1

Table model	*Precision* (%)	*Recall* (%)	*F-Measure* (%)
1	100.00	100.00	100.00
2	100.00	100.00	100.00
3	100.00	100.00	100.00
4	100.00	100.00	100.00
5	100.00	100.00	100.00
…	…	…	…
100	100.00	100.00	100.00
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 model	*Precision* (%)	*Recall* (%)	*F-Measure* (%)
1	100.00	100.00	100.00
2	100.00	100.00	100.00
3	100.00	100.00	100.00
…	…	…	…
40	67.00	100.00	80.00
…	…	…	…
99	100.00	100.00	100.00
100	100.00	100.00	100.00
Average (%)			97.67

Table 3 Testing summary of Algorithm 3

Table model	*Precision* (%)	*Recall* (%)	*F-Measure* (%)
1	100.00	100.00	100.00
2	100.00	100.00	100.00
3	100.00	100.00	100.00
4	100.00	100.00	100.00
5	100.00	100.00	100.00
…	…	…	…
33	28.57	33.33	30.77
…	…	…	…
99	77.77	77.77	77.77
100	100.00	100.00	100.00
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

Embley, D.W., Hurst, M., Lopresti, D. and Nagy, G. (2006) 'Table-processing paradigms: a research survey'. *International Journal of Document Analysis*, pp.66–86.

Gultom, R., Sari, R.F. and Budiardjo, B. (2011) 'Proposing the new algorithm and technique development for integrating web table extraction and building a mashup', *Journal of Computer Science*, Vol. 7, No. 2, pp.129–142.

Krupl, B., Herzog, M. and Gatterbauer, W. (2005) 'Using visual cues for extraction of tabular data from arbitrary HTML documents', *Proceeding WWW '05 Special Interest Tracks and Posters of the 14th International Conference on World Wide Web*, pp.1000–1001.

Lerman, K., Knoblock, C. and Steven, M. (2001) 'Automatic data extraction from lists and tables in web sources', *Proceedings of Automatic Text Extraction and Mining Workshop ({ATEM}-01)*.

Lin, J., Wong, J., Nichols, J., Cypher, A. and Lau, T.A. (2009) 'End-user programming of mashups with vegemite', *Proceedings of the 13th International Conference on Intelligent User Interfaces*, pp.97–106.

Liu, Y., Mitra, P. and Giles, C.L. (2008) 'A fast pre processing method for table boundary detection: narrowing down the sparse lines using solely coordinate information', *The Eighth IAPR International Workshop on Document Analysis Systems*, pp.431–438.

Tengli, A., Yang, Y. and Ma, N. L. (2004) 'Learning table extraction from examples', *Proceedings of The 20th International Conference on Computational Linguistics*, p.987.

Wong, W., Martinez, D. and Cavedon, L. (2009) 'Extraction of named entities from tables in gene mutation literature', *Proceedings of The Workshop on Current Trends in Biomedical Natural Language Processing, BioNLP '09*, pp.46–54.