AUTOMATIC WEB TABLE TRANSCODING FOR MOBILE DEVICES BASED ON TABLE CLASSIFICATION

Chichang Jou
Department of Information Management, Tamkang University
151 Ying-Chuan Road, Tamsui, New Taipei City, Taiwan

ABSTRACT

Many techniques have been proposed to improve web browsing experiences on the mobile devices by transcoding the original web content. However, the original semantics of web tables tend to be broken in the transcoded results. We capture basic features of web tables from their DOM-tree (Document Object Model Tree) semantic information. We propose a new table feature called Cell Extension Direction (CED) to capture the extension direction of cell content as one-directional or bi-directional. CED is computed by checking the difference between the average composite object type (ACOT) of rows and that of columns. These features are used to classify web tables into data tables and layout tables. The classification results, along with CC/PP configurations of the mobile device, are then utilized to guide the applications of the following three transcoding strategies for tables: zooming, transposition, and one-column-view. We demonstrate that the table semantics could be preserved in the transcoding results.

KEYWORDS

Table Classification, Web Table Transcoding, Mobile Computing

1. INTRODUCTION

Along with the fast developments of wireless and computing technologies, more and more people use diversified mobile devices to receive emails, to browse web sites, and to handle business. These devices, such as tablets, smart phones and PDAs, have miscellaneous hardware and software configurations. Since most web pages are designed for large screens on desktop and notebook computers, most mobile web pages are either distorted or with broken images, and thus hinder their comprehension.

Web tables were originally designed to embed important static data, like time tables and exchange rates, in a two dimensional structure. Nowadays, they are also frequently used to control layout of arbitrary content, and to exhibit dynamic database content. Due to changes of trends in web page design and in embedding cell content, previously proposed table features could not correctly classify tables. Although W3C has provided guidelines[1] for designing web tables that transform gracefully, the transcoded tables on mobile devices either could not convey original semantics, or would hinder comprehension of semantics. In some cases, even after manipulating vertical and horizontal scroll bars back and forth, users still could not capture the table semantics.

We use the following example to explain the broken table semantics issue: Fig. 1 is a currency exchange table of four countries and exchange dates. Fig. 2 is the one-column-view of this table on a mobile device. With the loss of relative positions among cells, the correlation between the exchange dates and country names is lost.

[1] http://www.w3.org/TR/WAI-WEBCONTENT/wai-pageauth.html

		30 Nov.2010	2 Dec.2010	3 Dec.2010	6 Dec.2010	7 Dec.2010
Australian dollar	AUD	3.1614	3.1530	3.1819	3.2052	3.2035
Bulgarian lev	BGN	2.1905	2.1947	2.1984	2.2042	2.1989
Canadian dollar	CAD	3.2213	3.2092	3.2387	3.2257	3.2108
Swiss franc	CHF	3.2891	3.2577	3.2677	3.3094	3.2921

Figure 1. Original web table

Figure 2. One-column-view of the original table

From the above example, the binding degree of relationship among cell content in a table would greatly affect whether the table semantics could be preserved in the transcoded result. If tables could be classified to capture their functionality, then proper transcoding strategies could be selected, not only to fit the hardware and software configurations of each mobile device, but also to preserve the original table semantics.

We design and implement a “Web Table Transcoding based on Classification” system, abbreviated as WTTC, to classify web tables, and then transcode these tables based on the classification result and the CC/PP configurations of the mobile devices. In addition to extended table features extracted from the DOM-tree semantic information, we propose a new table feature called Cell Extension Direction (CED) that represents the extension direction of cell content as one-directional or bi-directional. CED is computed based on the difference of average composite object type (ACOT) of rows and ACOT of columns. Based on the above features, web tables are classified into data tables and layout tables. Along with the client’s CC/PP device configurations, the classification result would then be applied to guide the applications of zooming, transposition, and one-column-view transcoding strategies for tables. Our automatic transcoding results would thus be customized for each device, and preserve the structure and semantics of the original tables.

The rest of the article is organized as follows: Section 2 is about related work. Section 3 is the system architecture of WTTC. Table classification and the classification results of WTTC are presented in Section 4. Section 5 illustrates the use of the classification results in applying proper transcoding strategies. Section 6 concludes the article by summarizing our achievements and identifying future research directions.

2. RELATED WORK

2.1 Transcoding of Web Tables

Bickmore et al. [2] employed a heuristic planning algorithm and a set of structural page transformations to produce the ‘best’ looking document for a given display size. For tables that could not be directly sent to the client, it output one sub-page per table cell. Their table transformation also determined ‘navigational sidebar columns’ and moved cells to the end of the list of sub-pages.

Chen et al. [3] developed a page-adaptation technique that split a page into smaller, logically related units that could fit onto a mobile device’s screen. The Web page then could be adapted to form a two-level hierarchy with a thumbnail representation at the top level for providing a global view and an index to a set of subpages at the bottom level for detailed information.

Hwang et al. [7] introduced two new heuristics, the generalized outlining transform and the selective elision transform, to preserve web page structures during transcoding. Both exploited layout characteristics of complex web pages. They attempted to preserve the original table structure: in addition to table attributes such as cell width, this transform used syntactic attributes such as font size to decide whether to elide a table cell.

Artail and Rayden [1] introduced a method that applied the device type and screen size to render web pages that fit the display area of the requesting device. It employed CSS elements to reduce the size of the web page’s building blocks. It used scripting for hiding and re-appearing parts of the page’s textual items and for converting tables to text. They applied the zooming strategy for the layout tables by adjusting the table width. For data tables, they rearranged cell contents so that the cell content would be presented as a pure text in a row.

He et al. [4] proposed a rule-based content adaptation system to facilitate extensible content adaptation for miscellaneous clients. They classified HTML objects into structure, content, and pointer objects. They used fuzzy logic to capture the measurement of distortion, caused by zooming and partitioning, and user satisfaction for each cell and each row, so that the adaptation quality could be used in guiding the adaptation decision.

Tajima and Ohnishi [9] introduced the concept of keys and developed a method of automatically discovering attributes and keys in tables. They then proposed three modes for browsing tables: normal mode, record mode, and cell mode. Each mode was provided with interaction features like hiding unnecessary rows and columns.

Most of the above researches tried to adjust the original tables so that users could use the vertical scroll bar to browse the result. However, this principle could not fit all cases, and may destroy table's semantics easily.

2.2 Use of Machine Learning in Table Classification

Hurst [6] modeled tables in terms of geometry, simple hierarchies of strings and database-like relational structures. He used simple two-dimensional geometry of tables to denote both the organization of their terms and the relations that hold between these terms. Geometric relationships were employed to express the structure (and by implication, the meaning) of the table.

Wang and Hu [10,11] applied machine learning techniques to classify web table into data tables and layout tables. To increase classification precision, Wang and Hu extracted a special feature, called content length consistency (CLC), to reflect whether there is a string length consistency among cells in a row or in a column. Similarly, they extracted a feature, called content type consistency (CTC), to reflect the average cumulative dominant content type for cells in a row or in a column.

Wang et al. [12] extracted extended visual and content features to classify web tables into data tables and layout tables. To preserve structures of data tables, the classification result was then applied to transform data tables into one-column view through zooming and rotation. Their features were mostly based on ratios of various objects, which are not suitable for describing diversified usage of tables.

Okata and Miura [8] proposed a classifier to check whether the pages include layout-purpose <table> tags using the ID3 technique. Their result showed that the tags could be classified with attribute values of border, number of rows, number of tags that appear ahead of the <table> tag, and the nest of <table> tags.

3. SYSTEM ARCHITECTURE AND DATA FLOW OF WTTC

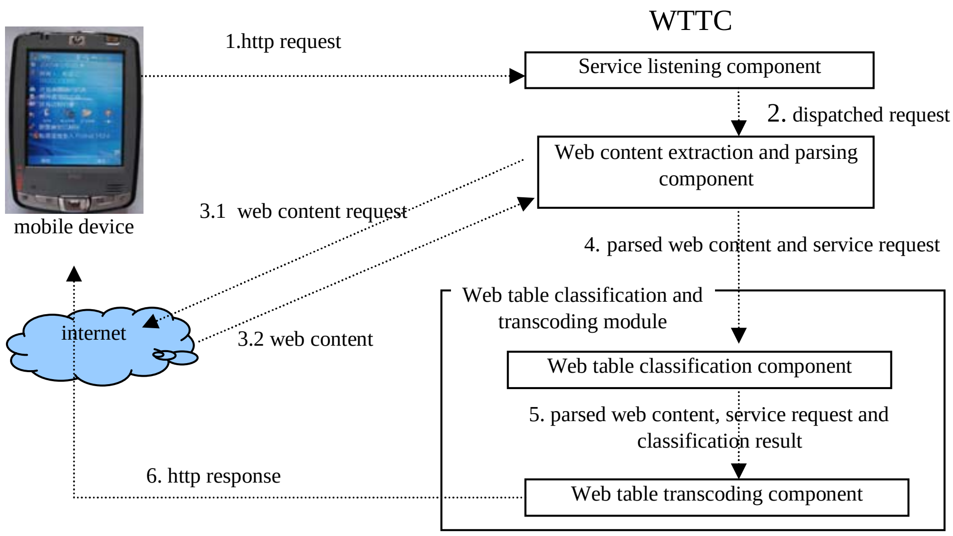


Figure 3. System architecture and data flow of WTTC

The design concept of WTTC is to preserve table semantics by keeping the relative positions of highly correlated cells. We extend an existing proxy server[2] to handle the web content extraction, web content transcoding, and transcoded web content delivery to the mobile devices. Fig. 3 displays the system architecture and data flow among the modules of WTTC. The numbers indicate the sequence of data flow or message transmissions. The operations of WTTC are explained as follows:

1) The mobile device sends out a web page request. The client's device information, including hardware platform, software platform, and browser user agent, will be embedded inside these requests through CC/PP diff, which is a modified version of predefined CC/PP profile from the hardware manufacturers. Many protocols have been proposed to enhance HTTP 1.1 protocol to include CC/PP profile diff. We adopt CC/PP-ex[3] in this framework.

2) The service listening component receives the request, and dispatches the request to web content extraction and parsing component.

3) The web content extraction and parsing component obtains the requested content from the internet, and use JTidy[4] to reformat the web page into the XHTML format and then build the DOM-tree for the page. Then the parsed web content and service request would be sent to the web table classification component.

4) The web table classification component would extract features of the leaf tables, meaning no nested table contained inside, from the DOM-tree to perform the classification. The classified result would be sent to the web table transcoding component.

5) The web table transcoding component would perform the transcoding according to the classification result.

6) The transcoded result is sent back to the mobile devices.

Since this article is focused on web table transcoding, we skip detailed descriptions regarding service listening component and web content extraction and parsing component.

4. WEB TABLE CLASSIFICATION

We adopt the following two table definitions from previous studies [6,11,12,13]:

1. Data table: The content in cells is highly correlated. Once the relative positions of the cells are changed, the correlation hierarchy is damaged and the information validity is lost. For this type of table, correlated cells' relative positions should not be changed.

2. Layout table: these tables are only considered as tables by their appearance rendered by a browser. The functionality of these tables is just to give a better layout. Even if the relative positions of their cells are changed, these tables are still readable and its meaning is not lost.

Fig. 1 is an example data table. Fig. 4 is an example layout table in a web page that provides Google search functionality for both WWW and its web site. Content of the four cells in the table are independent.

Figure 4. Example layout table

4.1 Table Features for Classification

Many previously proposed features, like border width of the table, textual content ratio, width of column span, and length of row span, could no longer provide distinguishing power in classifying tables. For example, although texts are still used in many cells for displaying highly correlated information, many cells with correlated content contain composite multi-media objects. Thus, the textual content ratio itself could not provide the same distinguishing power as before. The table features extracted in this study are listed in Table 1.

2 http://www.cs.technion.ac.il/Labs/Lccn/projects/spring97/project1
3 http://www.w3.org/TR/NOTE-CCPPexchange
4 http://jtidy.sourceforge.net/

Hurst [6] and Wang et al. [12] observed that the textual content ratios for data tables normally were greater than a threshold, while the link content ratios and image content ratios below a threshold. Our cells-without-span ratio was to emphasize their difference of frequencies in cells with the colspan and rowspan attributes. It was normally lower than a threshold in data tables, so as to emphasize the correlation of cell content. It was normally higher than a threshold in layout tables, so as to improve the appearance.

Hurst [6] and Wang et al. [12] checked whether number of cells with colspan or rowspan is within an interval. We observed that due to change of web table usage, these two numbers have large deviations among tables. It is difficult to find an interval that is suitable for table classifications. Thus, we modified data types of these features to Boolean, to simply illustrate whether there exists at least one cell with colspan or rowspan.

Table 1. Table features used in WTTC

Feature name	Description	Sources
Link content ratio	Whether the ratio of number of cells containing links to total number of cells is greater than a threshold	Hurst[6], Wang et al. [12]
Textual content ratio	Whether the ratio of number of cells containing texts to total number of cells is greater than a threshold	
Image content ratio	Whether the ratio of number of cells containing images to total number of cells is greater than a threshold	
Cells-without-span ratio	Whether the ratio of number of cells without the span attribute to total number of cells is greater than a threshold	
Colspan existence	Is there at least one cell with the colspan attribute?	This research
Rowspan existence	Is there at least one cell with the rowspan attribute?	This research
CED (Cell Extension Direction)	“one-directional” or “bi-directional”	This research

Table 2. Composite cell object types and their values

Composite Cell Object Type	Type Value
Non-numeric string	1
Numeric string	2
Image	3
Link	4
String+image	5
String+link	6
Image+link	7
String+image+link	8
Form	9
other	10

We extend CTC [10, 11] to compute the Average Composite Object Type of rows and columns ($ACOT_{row}$ and $ACOT_{col}$) in a table, respectively. These two values are then used to determine a new table feature called Cell Extension Direction (CED). The steps for determining CED through computing $ACOT_{row}$ and $ACOT_{col}$ of a table are summarized as follows:

1) Record number of rows, number of columns, colspan of cells, and rowspan of cells in the table. Based on the above information, extract the Maximal Continuous Block without rowspan and colspan (MCB) of the table.

2) For the MCB, compute $ACOT_{row}$ and $ACOT_{col}$.

3) CED of the table is then determined based on the difference of $ACOT_{row}$ and $ACOT_{col}$.

Details about the above three steps are explained as follows: We define six basic object type of cell content: numeric string, non-numeric string, image, link, form, and other. Since objects in a cell may be of different object types, we extend the three basic object types, *i.e.* string, image, and link, to their combinations. Thus, totally we have ten composite cell object types, which are listed in Table 2.

Suppose there are *r* rows in the table. We use R_i ($1 \le i \le r$) to denote the *i*-th row, and RC_i the number of cells in R_i. For $1 \le j \le RC_i$, we use $RC_{i,j}$ to denote the *j*-th cell (from left to right) in *Ri*. $RC_{i,j,col}$ and $RC_{i,j,row}$ are used to denote the existence of colspan attribute and rowspan attribute for $RC_{i,j}$, respectively.

To obtain MCB of the table, we find the smallest row number (i) and then the largest number (n_i) that satisfy the following four conditions:

1) $RC_i \ge 2$

2) $RC_i = RC_{i+1} = RC_{i+2} = \ldots = RC_{ni}$

3) for all $1 \le j \le RC_i$, $RC_{i,j,col} = 0$ and $RC_{i,j,row} = 0$

4) $i+1 \le n_i \le r$

Then, MCB is the block of rows from R_i to R_{n_i} . Condition 1 ensures that there is more than one cell in a row. Condition 2 ensures that numbers of cells for all rows in MCB are equal. Condition 3 ensures that there is no colspan or rowspan for all cells in MCB. Condition 4 ensures that rows in MCB are extended from top to down. Thus, number of rows in MCB is $n_i - i + 1$, and number of columns in MCB is RC_i. Note that some tables may not have a pair of i and n_i that satisfying all four conditions.

Suppose there are r_M rows in the MCB of a table. We use R_i ($1 \le i \le r_M$) to denote the i-th row of the MCB, and RC_i number of cells in R_i. Let T denote the set of the 10 composite cell object types. For all t in T, we use N_{it} to denote number of cells in Row R_i with type t. We then compute the maximal composite object type for row R_i, denoted as $MCOT_i$, as follows:

$$MCOT_i = \max_{t \in T}(N_{it}) \,/\, RC_i \qquad (1)$$

The average composite object type of rows for MCB is computed as follows:

$$ACOT_{row} = \frac{1}{r_M} \sum_{i=1}^{r_M} MCOT_i \qquad (2)$$

The same procedure could be applied to compute $ACOT_{col}$. For a small value δ as a threshold, CED is then determined based on the difference between $ACOT_{row}$ and $ACOT_{col}$ as follows:

$$\text{CED} = \begin{cases} \text{"bi-directional"}, & |ACOT_{Row} - ACOT_{Col}| \le \delta \quad or \quad if\ there\ is\ no\ MCB \\ \text{"one-directional"} & |ACOT_{Row} - ACOT_{Col}| > \delta \end{cases} \qquad (3)$$

4.2 Table Classification

We follow the method in [10,11,12] to train and test our proposed table classification in the business and news directory of the DMOZ open directory project. We use the following 12 keywords: “exchange rate”, “score”, “sport”, “finance”, “stock”, “weather”, “report”, “shopping”, “table”, “results”, and “value” to extract 1502 unique tables from 236 pages of 105 web sites. These 1502 tables are manually classified into 1198(79.76%) layout tables and 304 (20.24%) data tables. To decide the thresholds for the four table features with ratios, five-fold cross-validations are applied to obtain the thresholds in Table 3.

We use the following two classical classification techniques: Bayesian and ID3 decision tree. To illustrate the importance of the table feature CED, we also perform classifications without CED. By treating data tables as the desired classification, with δ in (3) set as 0, the precision, recall, and F-measure values of the classification results are listed in Table 4.

Table 3. Thresholds for table features with ratios

Feature name	Classification with CED	Classification without CED
Link content ratio	50%	50%
Textual content ratio	20%	80%
Image content ratio	80%	40%
Cells-without-span ratio	70%	80%

Table 4. Classification effectiveness

	Bayesian		ID3	
	Classification without CED	Classification with CED	Classification without CED	Classification with CED
Precision	0.4604	0.9801	0.7576	0.9114
Recall	0.8366	0.9673	0.1634	0.9412
F-measure	0.5939	0.9737	0.2688	0.9261

From Table 4, it is easy to see that the inclusion of CED boosts the precision and recall tremendously.

5. WEB TABLE TRANSCODING

In WTTC, the web table transcoding component calculates the table width based on font size information, number of characters in a string, image metadata, and CC/PP of the mobile device. It then performs transcoding based on the classification result. For layout tables, we would use one-column-view and zooming as main strategies, such that the whole table would be preserved together. For data tables, we would use zooming and transposition as main strategies.

When the calculated table width exceeds the screen width of the mobile device, the zooming strategy would be applied. Its procedures include: (1) decreasing the distances between cell content and their borders and (2) shrinking images inside cells.

The procedures for the one-column-view strategy are to adjust tables by adopting the sequential placement methods [6,16] used in placing the semantic blocks sequentially for web pages on the mobile devices. We adjust the <tr> and <td> positions in the <table> node of the DOM-tree, so that in the transcoded result, each row would have only one <th> or <td> tag. One-column-view strategy could make the transcoded result large enough to prevent unclearness caused by too much shrinking.

When the ratio of width and height of a data table exceeds a threshold (1.5 in WTTC), the transposition strategy would be applied. Its procedures are similar to the one-column-view strategy. By adjusting the <th> or <td> tags in a <tr> node of its DOM-tree, the rows in the original table would be transposed into their corresponding positions in the transcoded columns. For example, we would transpose the currency exchange table in Figure 5 (a) into Figure 5(b).

(a) Original table (b) Transposed table

Figure 5. Example of the transposition strategy

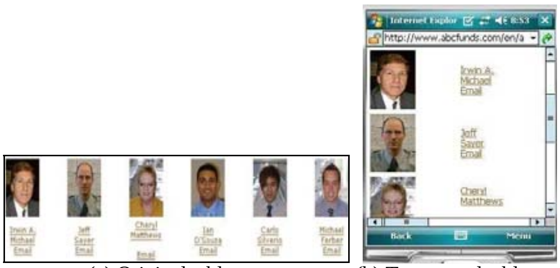
(a) Original table (b) Transposed table

Figure 6. Transcoding example for layout tables

5.1 Transcoding Layout Table

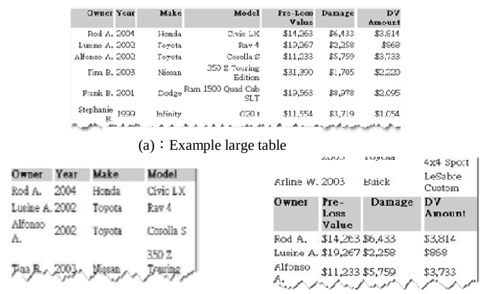
For layout tables, to keep a flexible zooming effect, one-column-view strategy would be applied first. Then the zooming strategy would be applied to every cell. Figure 6 is a transcoding example for layout tables. With the flexibility in shrinking ratio, this handling could prevent generating unreadable tiny transcoded results, especially for cases with large cell content in a row.

5.2 Transcoding Data Tables



(a) Original table (b) Transposed table

Figure 7. Transcoding example for data tables



(a)：Example large table

(b)：Upper Transcoded Result (c)：Lower Transcoded Result

Figure 8. Transcoding

For small data tables with $ACOT_{row} – ACOT_{col} > \delta$, we would transpose the table first. If the resulting table is too wide, then the zooming strategy would be applied. For small data tables with $ACOT_{col}$ - $ACOT_{row} > \delta$, since it is extended in the column direction, then the zooming strategy would be enough.

Figure 7 is a transcoding example for data tables with $ACOT_{row} – ACOT_{col} > \delta$. Since it is extended in the row direction, there is no attribute for its cells in the two rows. Since both the image and the name are accompanied with a link, its link content ratio is 100%. WTTC would perform the transposition strategy to o the one-column-view without losing semantics of the original table.

For large data tables with CED value "bi-directional" or $ACOT_{col}$ - $ACOT_{row} > \delta$, the above strategies could not achieve satisfying result. Normally number of columns in these large data tables is much larger than number of rows. In the transcoding, a splitting of columns would be performed first. First column of the original table would be treated as identifiers and be copied to all split fragments. Finally, a zooming strategy would be applied to each group so that the transcoded results could fit the screen width. For large data tables with $ACOT_{row} - ACOT_{col} > \delta$, general transcoding strategy would be appropriate. Figure 8 is a transcoding example of tables with 77 columns and 7 rows, in which a splitting procedure is applied before transcoding.

6. CONCLUSIONS AND FUTURE WORK

We designed and implemented a web table transcoding system based on table classification (WTTC) so that the semantics of the original table could be preserved. More specifically, we proposed a new table feature, Cell Extension Direction (CED), which represents the direction of correlated cell content. CED is determined by the difference between average composite object type for rows and that for columns.

We demonstrated the power of CED in classifying web tables into data and layout tables. The classification result and the CC/PP configurations of the mobile device were then used to apply the one-column-view, zooming, and transposition transcoding strategies customized for the mobile devices. For large data tables, we use transposition and zooming strategies. For layout tables, we use one-column-view and zooming strategies. The flexibility provided by WTTC in the zooming ratios helps improve users' browsing experience.

This methodology focuses on leaf tables, and nested tables are treated as layout tables. More investigation in verifying the above handling is ongoing. Besides the <table> tag, we would like to extend this methodology to other HTML elements, such as <DIV> and CSS files. <DIV> tags are becoming popular in rendering tabular data, and are used for many miscellaneous purposes. Without clear semantics from the cell content, the semantics of the <DIV> tag needs more investigation. Meanwhile, CSS files are commonly used in many web pages for presenting consistent styles. Features obtained from CSS files would improve the classification result. More interestingly, CSS files themselves may also need to be transcoded.

REFERENCES

[1] Artail, H. & Raydan. M. (2005). Device-aware Desktop Web Page Transformation for Rendering on Handhelds. Personal Ubiquitous Computing, 9(6), 368-380.

[2] Bickmore, T., Girgensohn, A., & Sullivan, J. (1999). Web Page Filtering and Re-authoring for Mobile Users. The Computer Journal, 42(6), 334-346.

[3] Chen, Y., Xie, X., Ma, W. & Zhang, H. (2005). Adapting Web Pages For Small-Screen Devices. IEEE Internet Computing Archive, 9(1), 50-56.

[4] He, J., Gao, T., Hao, W. & Yen, I.L. (2007). A Flexible Content Adaptation System Using a Rule-Based Approach. IEEE Transactions on Knowledge and Data Engineering, 19(1), 127-140.

[5] Hurst, M. (2002). Classifying Table Elements in HTML. Proceedings of the 11th International World Wide Web Conference, 7-11.

[6] Hurst, M. (2006). Towards a Theory of Tables. International Journal on Document Analysis and Recognition, 8(2-3), 123-131.

[7] Hwang, Y., Kim, J. & Seo, E. (2003). Structure-Aware Web Transcoding for Mobile Devices. IEEE Internet Computing, 7(5), 14-21.

[8] Okata, H. & Miura, T. (2007). Detection of Layout-Purpose TABLE Tags Based on Machine Learning. Lecture Notes in Computer Science 4556, Part 3, 116-123.

[9] Tajima, K. & Ohnishi, K. (2008). Browsing Large HTML Tables on Small Screens. Proceedings of the 21st annual ACM symposium on User Interface Software and Technology, 259-268.

[10] Wang, Y. & Hu, J. (2002). Detecting Tables in Html Documents. Proceedings of the 5th International Workshop on Document Analysis Systems, 249-260.

[11] Wang, Y. & Hu, J. (2002). A Machine Learning Based Approach for Table Detection on the Web. Proceedings of 11th International World Wide Web Conference, 242-250.

[12] Wang, C., Xie, X., Wang, W. & Ma, W.Y. (2004). Improving Web Browsing on Small Devices Based on Table Classification. Proceedings of the 12th International World Wide Web Conference, 20-24.

[13] Wu, J. J. & Yang, J. H. (2007). Using Content Analysis Technique to Enhance Content Adaptation System. Proceedings of the 9th IEEE International Symposium on Multimedia Workshops, 23-28.