Improved Mobile and Web Accessibility of Unstructured Web Table

Pauli P. Y. Lai
Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong

Abstract - *Web table understanding is usually difficult for blind people. They depend on screen readers to convey the table information. However, if the table is large, the user may have long forgotten the heading before the last row is read. On the other hand, it is also difficult for users to view a large web table on a small-screen device. It usually requires extensive scrolling to view a large table with small-screen display. When the last row or last column is reached, the associated headers are probably out of screen and thus the meaning of data values is lost. These unstructured web tables arises a need for mobile and web accessibility improvements. In this regard, we propose a method to extract the structure from these tables and perform adaptation so that the mobile and blind users can perceive and retrieve information from the web table more easily and conveniently*.

Keywords: Web Table, Web Accessibility, Mobile Accessibility, Blind, Content Adaptation, Mobile Web Browser

1 Introduction

How do blind people read a web page? Since blind people cannot see, they can only perceive webpage by alternative sensory channels such as auditory channel or through tactile devices. The most common way for them to access the Internet is to use a Web browser and text-to-speech software [1]. However, screen readers present content linearly to users. This contrasts with the way in which sighted people use visual interfaces. The linear presentation of web content imposes difficulty on web table understanding by the blind. Therefore, we need to seek way to improve presentation of web tables to the blind.

On the other hand, it is also difficult for users to view a large web table into a small screen device. It usually requires extensive scrolling to view a large web table with small screen display. When the last row or last column is reached, the corresponding column headers or row headers are probably out of screen and thus the meaning of the data values is lost.

A web table is an HTML table that is enclosed between two HTML tags: <table> and </table>. There are two types of web table. A data table is that used for conveying the contents' information, and its contents' meaning depends on its structure. It is also called genuine table or meaningful table. A layout table is that used in constructing the layouts of HTML documents and its contents' meaning does not depend on its structure. It is also called non-genuine table or decorative table. A data table is usually equipped with column headers, row headers or both. In some cases, a data table may miss an explicit header but there is an implicit existence of eclipse abstractions for the table cells. In contrast, layout tables do not have logical headers that can be mapped to information within the table cells [2, 3].

There are some guidelines for web developers to create accessible web table [3]. However, this is not a compulsory requirement and thus there are lots of legacy web tables that are not designed with accessibility in mind. These tables are not well structured, i.e. they are not marked up properly. For example, the authors do not make of <THEAD> or <TH> tags to indicate column headers. Therefore, there is a need arisen to improve mobile and web accessibility for these unstructured web tables.

In our paper, we will describe how to extract table structure from unstructured data tables for better presentation to the blind and mobile users. The rest of the paper is organized as follows. Section 2 will present the process of analyzing table structure. Section 3 and 4 will discuss how to improve web accessibility and mobile accessibility of web tables respectively. Finally, section 5 will conclude our work.

2 Table Structure Analysis

2.1 Identifying Table Direction

In the area of table structure recognition research, many papers need to classify tables into genuine (also called data or meaningful table) or non-genuine (also called layout or decorative table) first [4, 5]. We do it by identifying Hparallel and Vparallel relationships among table cells. Hparallel relationship means that the table cells are horizontally similar whereas Vparallel relationship means that the table cells are vertically similar.

In [4], the authors also measure the cell similarity to determine whether a table is genuine because they assumed that value cells under the same attribute names demonstrate similar concepts. The metrics that they employ to measure the cell similarity includes string similarity, named entity similarity and number category similarity. They count how many neighboring cells are similar. If the percentage is above a threshold, the table tags are interpreted as a genuine table. This is similar to our approach but we use more metrics including the type of the node (e.g. text or anchor node), the font family, the font size, the font weight, the font style, the color, the background color, the content length, the hyperlink address, etc. to measure similarity. Also we would not assume that the value cells are number category which is not always true in fact.

To determine whether a web table is data table or layout table, we compare the cell similarity to identify Hparallel and Vparallel relationships. First we need to determine whether a row is Hparallel or whether a column is Vparallel. If the proportion of Hparallel cells to the total number of cells in a row is greater than some threshold (*th_cell*), then the row is regarded as Hparallel row. Similarly, if the proportion of Vparallel cells to the total number of cells in a column is greater than some threshold (*th_cell*), then the column is regarded as Vparallel column. After identifying the Hparallel rows and Vparallel columns, we can check whether a table is a data table or not. If the proportion of Hparallel rows to the total number of rows or that of Vparallel columns to the total number of columns is greater than some threshold (*th_rowcol*), then this is a data table. Otherwise, this is a layout table. The formulas for determining Hparallel rows, Vparallel column, row-wise data table and column-wise data table are shown in Figure 1. The details of the relationship identification process can be found in our previous work [6].

$$\frac{\text{no. of Hparallel cells}}{\text{total number of cells}} \geq \text{th\_cell} \Rightarrow \text{Hparallel Row}$$

$$\frac{\text{no. of Vparallel cells}}{\text{total number of cells}} \geq \text{th\_cell} \Rightarrow \text{Vparallel Column}$$

$$\frac{\text{no. of Hparallel rows}}{\text{total number of rows}} \geq \text{th\_rowcol} \Rightarrow \text{Column - wise Data Table}$$

$$\frac{\text{no. of Vparallel columns}}{\text{total number of columns}} \geq \text{th\_rowcol} \Rightarrow \text{Row - wise Data Table}$$

Figure 1. The formulas for determining Hparallel rows, Vparallel column, row-wise data table and column-wise data table.

If two neighboring cells are similar, they are considered as parallel cells. If two neighboring cells are parallel in the same row ($C_{i,j}$ and $C_{i,j+1}$), they are said to be horizontally parallel (Hparallel). Similarly, if two neighboring cells are parallel in the same column ($C_{i,j}$ and $C_{i+1,j}$), they are said to be vertically parallel (Vparallel). A web table with table cells $C_{i,j}$ is shown in Figure 2.

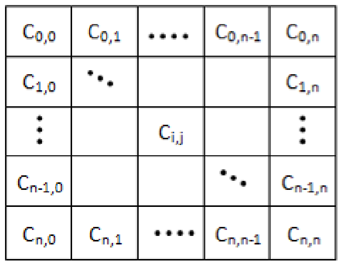


Figure 2. A web table with table cells $C_{i,j}$.

Since the values under an attribute are usually similar in terms of visual properties, we determine whether the table is column-wise or row-wise by the Hparallel or Vparallel realationships identified. If most of the columns are vertically parallel (Vparallel), then the table is row-wise. Similarly, if most of the rows are horizontally parallel (Hparallel), then the table is column-wise. In some cases, the table cells exhibit both Hparallel and Vparallel relationship, which means that the data records can be read in both directions. In these cases, attributes are on both rows and columns, and the table is therefore called row-column-wise or both-wise. The idea is illustrated in Figure 3. Figure 4 and Figure 5 show examples of a row-wise table and column-wise table respectively.

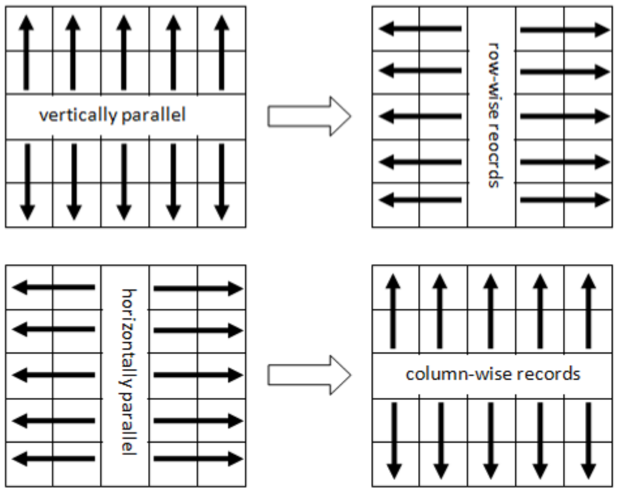
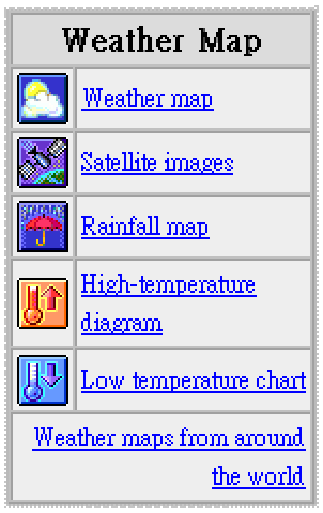
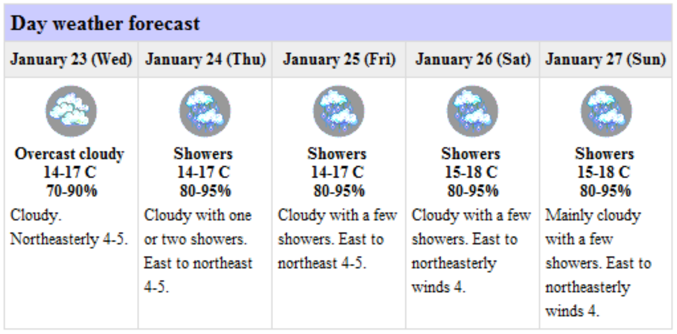


Figure 3. A table is row-wise if most columns are vertically parallel whereas it is column-wise if most rows are horizontally parallel.



Weather Map	
	Weather map
	Satellite images
	Rainfall map
	High-temperature diagram
	Low temperature chart
Weather maps from around the world	

Figure 4. An example of a row-wise table with table caption and footer.



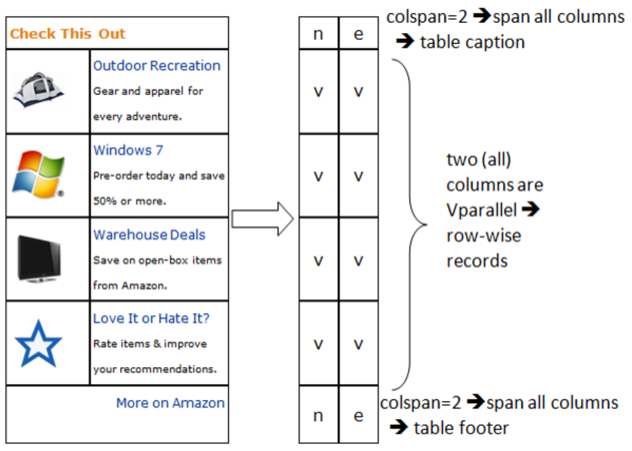
Day weather forecast				
January 23 (Wed)	January 24 (Thu)	January 25 (Fri)	January 26 (Sat)	January 27 (Sun)
Overcast cloudy 14-17 C 70-90% Cloudy. Northeasterly 4-5.	Showers 14-17 C 80-95% Cloudy with one or two showers. East to northeast 4-5.	Showers 14-17 C 80-95% Cloudy with a few showers. East to northeast 4-5.	Showers 15-18 C 80-95% Cloudy with a few showers. East to northeasterly winds 4.	Showers 15-18 C 80-95% Mainly cloudy with a few showers. East to northeasterly winds 4.

Figure 5. An example of a column-wise table with table caption.

2.2 Identifying Table Structure

A simple web table consists of row(s) and column(s) defined by <tr> and <td> tags respectively. A more complex web table may include caption, header, body, footer defined by <caption>, <col>, <colgroup>, <thead>, <tbody>, and <tfoot> elements. The web developers are encouraged to mark up the table properly so that the screen readers can read properly [7]. However, many authors may not design the web table with accessibility in mind and they simply use <tr> and <td> tags to construct a web table. Instead of using proper tags, they use visual properties to differentiate between header, body, and footer sections. This gives challenges to screen readers to interpret the web table properly. Moreover, to my best knowledge, there is no HTML tag available to signify row header which is usually found in both-wise table. Therefore, we propose a method to identify table caption, column header, row header, and footer sections.

We first create a character matrix to indicate the relationship exhibited by each table cell. We use 'v' to denote Vparallel relationship, 'h' to denote Hparallel relationship, 'b' to denote both relationships, 'n' to denote no relationship, and 'e' to denote empty cell due to column span or row span. If a table cell in the first row spanning all columns or a table cell in the first column spanning all rows, then it is regarded as table caption. If a table cell in other row (or other column) spanning all columns (or all rows), then it is regarded as a caption of a sub-table. If the table cell spanning all columns (or all rows) is in the last row (or last column), then it is regarded as table footer. An example is shown in Figure 6.



Check This Out	
	Outdoor Recreation Gear and apparel for every adventure.
	Windows 7 Pre-order today and save 50% or more.
	Warehouse Deals Save on open-box items from Amazon.
	Love It or Hate It? Rate items & improve your recommendations.
More on Amazon	

n e
v v
v v
v v
v v
n e
colspan=2 ➔span all columns
➔ table caption
two (all) columns are Vparallel ➔ row-wise records
colspan=2 ➔span all columns
➔ table footer

Figure 6. Conversion of a web table to a character matrix. The web table is adopted and adapted from www.amazon.com.

To explore the boundary between headers (attributes) and data (values), we create two integer matrixes namely Hdiff and Vdiff to indicate the relationship difference between horizontal rows and between vertical columns respectively. This is because if the values demonstrate Vparallel relationship for a row-wise table, the corresponding attributes are usually Hparallel. The difference in relationship between table cells implies the boundary between headers and data. '1' in Hdiff means that the relationship of current cell is different from the neighboring cell in the next row while '1' in Vdiff means that the relationship of current cell is different from the neighboring cell in the next column. '0' means there is no difference. Since there is no next row (or next column) in the last row (or last column), we use '2' to indicate last row (or last column). Examples of converting character matrix into Hdiff and Vdiff matrix are shown in Figure 7.

To identify column headers, we find the first-appeared boundary of '1' and '0' in the Hdiff matrix for each column from top to bottom. Whenever a caption of sub-table is hit, the row(s) before would be set as the boundary. When the lowest boundary among all columns is found, the row(s) before which would be regarded as column headers. The row headers are identified in the similar approach with the row(s) of column headers excluded. An example is shown in Figure 8 and Figure 9. When searching boundary of '1' and '0' in the

Hdiff matrix for each column from top to bottom, a horizontal caption (Hcaption) is hit at the fourth row, and hence the first three rows are regarded as column headers. To search row header, the three rows are excluded since they are column-header rows. For the rest of the rows, the caption rows at 4th and 7th row are also skipped. As a result, only 5th, 6th, 8th and 9th rows are searched and from these rows, the lowest boundary of ‘1’ and ‘0’ happens to be before the 2nd column. Hence the first column is regarded as the row headers.

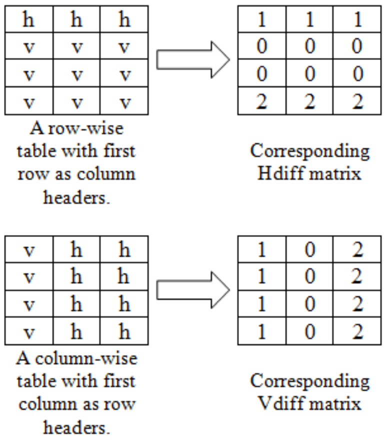
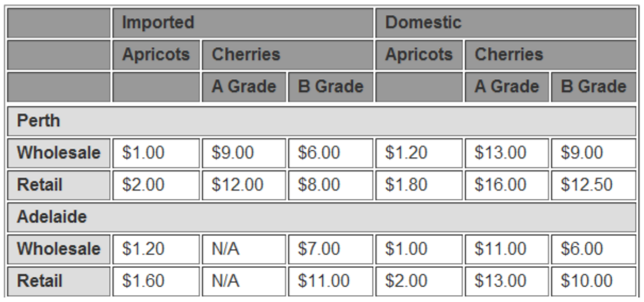
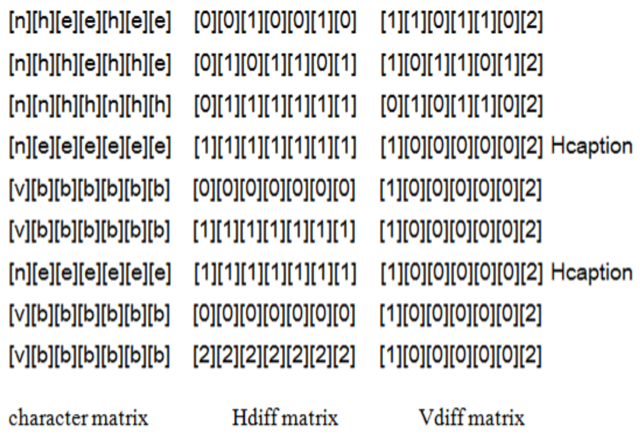

h h h
v v v
v v v
v v v
1 1 1
0 0 0
0 0 0
2 2 2
A row-wise table with first row as column headers.
Corresponding Hdiff matrix
v h h
v h h
v h h
v h h
1 0 2
1 0 2
1 0 2
1 0 2
A column-wise table with first column as row headers.
Corresponding Vdiff matrix

Figure 7. Conversion of a row-wise table to Hdiff matrix and conversion of a column-wise table to Vdiff matrix.



	Imported			Domestic		
	Apricots	Cherries		Apricots	Cherries	
		A Grade	B Grade		A Grade	B Grade
Perth						
Wholesale	$1.00	$9.00	$6.00	$1.20	$13.00	$9.00
Retail	$2.00	$12.00	$8.00	$1.80	$16.00	$12.50
Adelaide						
Wholesale	$1.20	N/A	$7.00	$1.00	$11.00	$6.00
Retail	$1.60	N/A	$11.00	$2.00	$13.00	$10.00

Figure 8. An example of complex table adopted from http://www.usability.com.au/resources/tabletest.cfm.



```
[n][h][e][e][h][e][e]  [0][0][1][0][0][1][0]  [1][1][0][1][1][0][2]
[n][h][h][e][h][h][e]  [0][1][0][1][1][0][1]  [1][0][1][1][0][1][2]
[n][n][h][h][n][h][h]  [0][1][1][1][1][1][1]  [0][1][0][1][1][0][2]
[n][e][e][e][e][e][e]  [1][1][1][1][1][1][1]  [1][0][0][0][0][0][2] Hcaption
[v][b][b][b][b][b][b]  [0][0][0][0][0][0][0]  [1][0][0][0][0][0][2]
[v][b][b][b][b][b][b]  [1][1][1][1][1][1][1]  [1][0][0][0][0][0][2]
[n][e][e][e][e][e][e]  [1][1][1][1][1][1][1]  [1][0][0][0][0][0][2] Hcaption
[v][b][b][b][b][b][b]  [0][0][0][0][0][0][0]  [1][0][0][0][0][0][2]
[v][b][b][b][b][b][b]  [2][2][2][2][2][2][2]  [1][0][0][0][0][0][2]

character matrix         Hdiff matrix           Vdiff matrix
```

Figure 9. A character matrix, Hdiff matrix, Vdiff matrix converted from the web table in Figure 8.

3 Improved Web Accessibility of Unstructured Web Table

The above table structure analysis can be applied to improve Web accessibility of unstructured web table for the blind. In our previous work [8], we have described how to improve web accessibility by converting a webpage into an Interactive Voice Response Systems (IVRS) so that the blind people can access the webpage using mobile phone by listening to the index page and getting into details by pressing the corresponding number on the key pad. We would like to extend our previous work with web table adaptation included.

As discussed before, there are two types of web table namely layout table and data table. For layout table, since there are no attribute-value pairs, it could only be rendered in two ways: organizing by rows and organizing by columns. The table is read row by row if it is organized by rows and it is read column by column if it is organized by columns. We offer both options to user because in different cases the web table is organized differently.

For data table, the table structure is analyzed and it is organized by records. For row-wise table, it is read row by row; for column-wise table, it is read column by column. If there exists any header, it would be associated with the corresponding table cell and read together. For both-wise table, it would be organized by rows with row header being read first, and followed by the column header and its associated data.

For complex table like that in Figure 8, since there are multiple rows for column headers, the column header for each column is integrated first. For example, the integrated column header for 2nd column is “Imported-Apricots” and that for 3rd column is “Imported-Cherries-A Grade”. Also, there are two sub-tables which would be abstracted under the corresponding captions “Perth” and “Adelaide”. The structure of the web table is shown in Figure 10 and the levels of abstractions are shown in Figure 11. The blind users can then listen to the web table level by level through a mobile phone and select their interesting option by pressing the corresponding number on keypad. Hence, the web accessibility of the web table has been greatly improved.

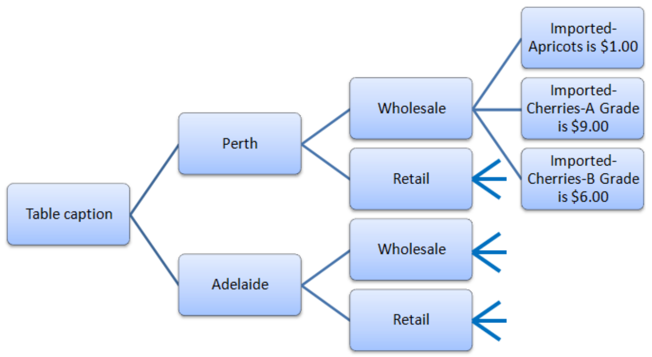
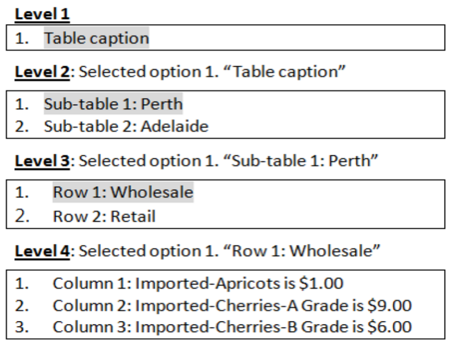


Table caption
Perth
Adelaide
Wholesale
Retail
Wholesale
Retail
Imported-Apricots is $1.00
Imported-Cherries-A Grade is $9.00
Imported-Cherries-B Grade is $6.00

Figure 10. The structure of the web table in Figure 6 is illustrated as tree level by level.



Level 1

1. Table caption

Level 2: Selected option 1. “Table caption”

1. Sub-table 1: Perth
2. Sub-table 2: Adelaide

Level 3: Selected option 1. “Sub-table 1: Perth”

1. Row 1: Wholesale
2. Row 2: Retail

Level 4: Selected option 1. “Row 1: Wholesale”

1. Column 1: Imported-Apricots is $1.00
2. Column 2: Imported-Cherries-A Grade is $9.00
3. Column 3: Imported-Cherries-B Grade is $6.00

Figure 11. The levels of abstraction for the web table in Figure 8.

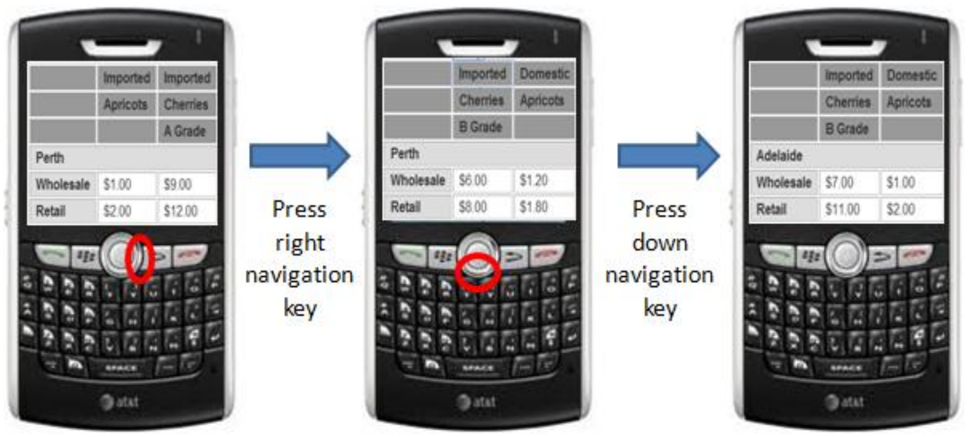
4 Improved Mobile Accessibility of Unstructured Web Table

As discussed before, there are two types of web table namely layout table and data table. Layout table is used to organize the presentation of information in grid format but there are no attribute-value pairs. Therefore, the table cells do not have any association with headers. Thus, the adaptation of layout table is to present each individual table cell from left to right and then from top to bottom. But since the screen estate for the mobile device is limited, whenever the accumulated width of the table cell to be printed exceeds the maximum screen width, a linebreak would be inserted so that the next table cell would be printed on a new line. Also, if the accumulated height of the table cell to be printed exceeds the maximum screen height, the next table cell would be printed on a new page so as to reduce the amount of vertical scrolling.

On the other hand, the above table structure analysis can also be applied to improve mobile accessibility of unstructured web table. Browsing a web table in a small screen device is a difficult task for users especially if the table is very large. Previous researches have various proposals for improving the ways of browsing large web table within limited screen estate. In [9], Tajima and Ohnishi propose three modes for browsing tables: normal mode, record mode, and cell mode. Normal mode renders tables in the ordinary way, but provides various useful functions for browsing large tables, such as hiding unnecessary rows and columns. Record mode regards each row (or column) as the basic information unit and displays it in a record-like format with column (or row) headers, while cell mode regards each cell as the basic unit and displays each cell together with its corresponding row and column headers. In [10], Potla, et al. adapt a HTML-based Web table into two adaptive styles: Single Narrow Layout and Multi page Layout . Single Narrow Layout offers one-dimensional browsing (either browsed by rows or by columns) within page by generating navigational hyperlinks based on row header and/or column header. In Multi page Layout, a hyperlink is generated for each <td> tag with the corresponding row header or column header or both (for multidimensional tables), which navigates to a new Web page. Except for the normal mode browsing proposed by Tajima and Ohnishi, all the above ways are limited to one-dimensional presentation even for multi-dimensional tables. Though users are guided with the column header and/or row header, it is easy for them to forget the navigational path before they reach the final data cell. Despite the normal mode browsing proposed by Tajima and Ohnishi offers two-dimensional browsing and provides function of folding unnecessary columns/rows, it is still difficult for users to manage and browse a large table with many columns and rows. In this regard, we propose a novel approach for browsing a large web table in small screen devices.

For a large web table that is not small enough to fit for a small screen display, the table would be partitioned into various sub-tables which are then distributed to multiple sub-pages such that each of which approximately occupies the whole screen of the device, and thus no scrolling is required. Each sub-table contains different data values with their associated column/row headers replicated. The idea is borrowed from a function which is already supported in some spreadsheet programs—freeze pane. After identifying column headers and row headers in section 2, we can adopt freeze pane approach to lock the column headers and/or row headers and allow navigation only for data values. Then the users can freely navigate any data cell with its associated column/row headers displayed together.

Figure 12 demonstrates the idea of browsing the web table from Figure 8 using the freeze pane approach. Since the table contains some column headers that span several columns such as " Imported", "Domestic" and "Cherries", the spanning cell needs to be split into some pseudo-cells according to the number of cells it spans. Each pseudo-cell is duplicated with the name of the column header so that when the table is partitioned in the middle of the spanning cell, the name of column header would still be present. When the user wants to see the next few columns, he/she can press the right navigation key, the data values with their associated column headers would be displayed while the row headers would be fixed. When the user wants to see the next few rows, he/she can press the down navigation key, the data values with their associated row headers would be displayed while the column headers would be fixed.



	Imported	Imported
	Apricots	Cherries
		A Grade
Perth		
Wholesale	$1.00	$9.00
Retail	$2.00	$12.00

	Imported	Domestic
	Cherries	Apricots
	B Grade	
Perth		
Wholesale	$6.00	$1.20
Retail	$8.00	$1.80

	Imported	Domestic
	Cherries	Apricots
	B Grade	
Adelaide		
Wholesale	$7.00	$1.00
Retail	$11.00	$2.00

Figure 12. Browsing the web table from Figure 8 using the freeze pane approach.

To print the sub-table pages, first we need to compute the number of rows and columns for each sub-table page. Given the specified font family and font size, the widths and heights in pixel for all table cells can be computed. Then, the maximum width required by each column and the maximum height required by each row can be obtained. With this information, we can then calculate the available width for displaying data cells by deducting the maximum width occupied by the row headers from the window width. The available width is used to compute the maximum number of columns to be displayed in each sub-table page. Since the maximum width for each column may be different, the maximum number of columns for each sub-table page may hence be different. The maximum number of rows for each sub-table page can be found in a similar manner.

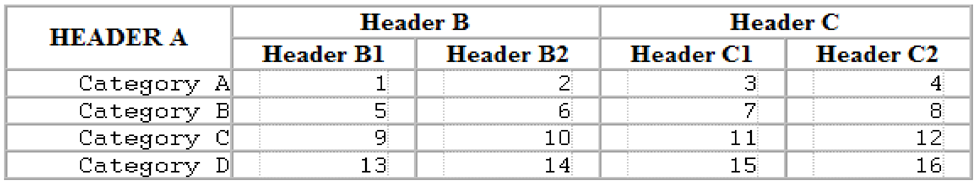
From the above algorithm, we have made the following assumptions:

1. It is assumed that at least two rows should be displayed in each sub-table page even though the accumulated height exceeds the available height.

2. It is assumed that at least two columns should be displayed in each sub-table page even though the accumulated width exceeds the available width.

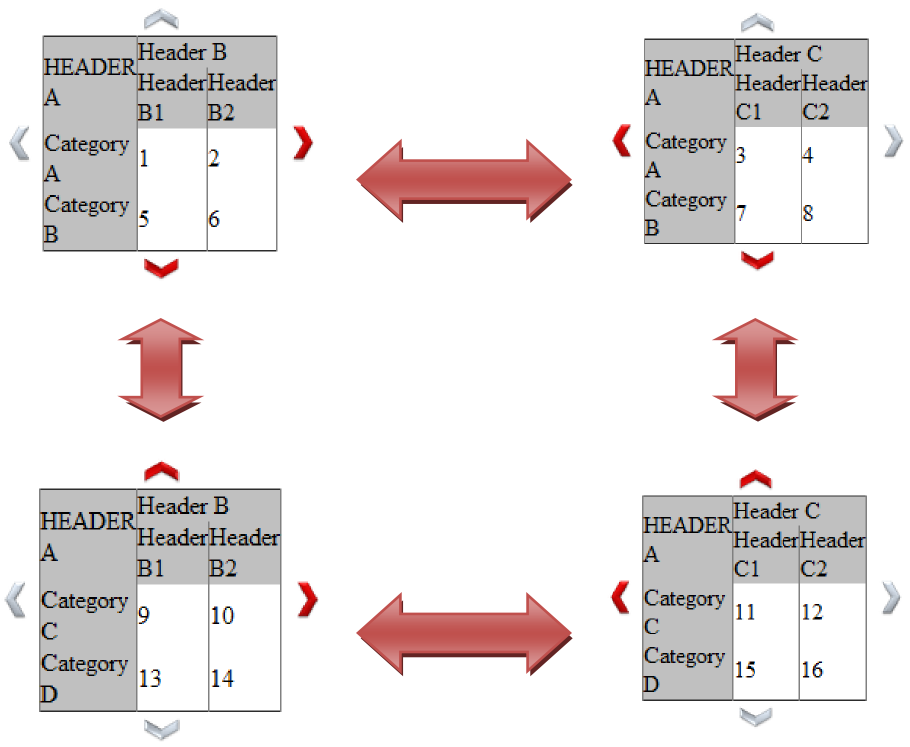
3. For the tables without row headers, it is assumed that the first column is the key to locate record and thus the first column would be regarded as row header so that the first column would also be replicated in each sub-table page.

According to the maximum number of columns and the maximum number of rows for each sub-table page obtained, the data cells would be printed to each sub-table page together with their associated row headers and column headers which are highlighted in gray. An example of adaptation is shown in Figure 13 and Figure 14 below with specified window dimension of 200x200. Note that in this example, the first column is not row header originally. It is set as row header due to assumption 3 above.



HEADER A	Header B		Header C	
	Header B1	Header B2	Header C1	Header C2
Category A	1	2	3	4
Category B	5	6	7	8
Category C	9	10	11	12
Category D	13	14	15	16

Figure 13. Original web table for adaptation.



HEADER A	Header B	
	Header B1	Header B2
Category A	1	2
Category B	5	6

HEADER A	Header C	
	Header C1	Header C2
Category A	3	4
Category B	7	8

HEADER A	Header B	
	Header B1	Header B2
Category C	9	10
Category D	13	14

HEADER A	Header C	
	Header C1	Header C2
Category C	11	12
Category D	15	16

Figure 14. Adapted sub-table pages for web table in Figure 13.

5 Conclusion

In conclusion, we have proposed a method to analyze unstructured web table by identifying table direction and extracting headers from data table. With the table structure identified, the web table is then re-organized into multiple levels of abstraction so that the blind users can access the table level by level by pressing the corresponding number on keypad. It has enhanced the table content understanding for the blind and the users can even access the web table when they are in mobile. Also, with the column headers and row headers identified, the web table can be adapted to fit into small screen display by adopting the freeze pane approach. This has greatly improved both the mobile and web accessibility of unstructured web table and the users are able to perceive and retrieve information from web table more easily and conveniently.

6 References

[1] J. Herrman. (2010), *Giz Explains: How Blind People See the Internet*. Available: http://gizmodo.com/5620079/giz-explains-how-blind-people-see-the-internet

[2] S. W. Jung and H. C. Kwon, "A machine learning based approach for separating head from body in web-tables," presented at the Proceedings of the 7th international conference on Computational Linguistics and Intelligent Text Processing, Mexico City, Mexico, 2006.

[3] (4/2/2012). *Creating Accessible Tables*. Available: http://webaim.org/techniques/tables/

[4] H. H. Chen, *et al.*, "Mining tables from large scale HTML texts," presented at the Proceedings of the 18th conference on Computational linguistics - Volume 1, Saarbr\&\#252;cken, Germany, 2000.

[5] Y. Wang and J. Hu, "A machine learning based approach for table detection on the web," presented at the Proceedings of the 11th international conference on World Wide Web, Honolulu, Hawaii, USA, 2002.

[6] P. P. Y. Lai, "Reverse Engineering of Web Pages for Logical Section Discovery," in *IET Younger Members Exhibition and Conference 2010*, Hong Kong, 2010.

[7] H. Ahmadi and J. Kong, "Efficient web browsing on small screens," presented at the Proceedings of the working conference on Advanced visual interfaces, Napoli, Italy, 2008.

[8] P. P. Y. Lai, "Application of content adaptation in web accessibility for the blind," presented at the Proceedings of the International Cross-Disciplinary Conference on Web Accessibility, Hyderabad, Andhra Pradesh, India, 2011.

[9] K. Tajima and K. Ohnishi, "Browsing large HTML tables on small screens," presented at the Proceedings of the 21st annual ACM symposium on User interface software and technology, Monterey, CA, USA, 2008.

[10] Y. Potla, *et al.*, "Adapting Web Page Tables on Mobile Devices," ed: IGI Global, 2012, pp. 1-22.