

Visual Based Content Understanding towards Web Adaptation

*Xiao-Dong Gu¹, Jinlin Chen², Wei-Ying Ma³, Guo-Liang Chen¹

¹ Dept. of Computer Science, Univ. of Sci. & Tech. of China,
230027, Hefei, China
qxd@mail.ustc.edu.cn

² School of Information Science, Univ. of Pittsburgh,
Pittsburgh, PA 15260 USA
jlchen@pitt.edu

³ Microsoft Research Asia
Beijing 100080, China
wyma@microsoft.com

Abstract. Web content structure is proposed to facilitate automatic web page adaptation in this paper. By identifying the logic relationship of web content based on layout information, web content structure effectively represents authors' presentation intention. An automatic top-down, tag-tree independent approach to detect web content structure is presented. It simulates how a user understands web layout structure based on his vision. Comparing to other content analysis techniques, our approach is independent to physical realization and works well even when the physical structure is far different from layout structure. Besides, our approach is an $O(n)$ -time process which is much more efficient comparing to other approaches with $O(n^2)$ -time complexity. Furthermore, our approach is tag tree independent, which means it can be applied to web contents of arbitrary physical realization formats. Experiments show satisfactory results.

1 Introduction

Web publishing is playing a more and more important role for information distribution today. When creating a new web page, the author first decides what to present, i.e., the semantic content. He then further decides how to present the information. Finally, a markup language is used to realize the presentation, which gives a physical structure to the content. Considering the whole process, we have XML to represent semantic structure and many markup languages such as HTML to represent physical structure. However, we still lack an effective way to represent presentation structure, which indicates authors' intention towards the presentation logic of web content.

* This work was performed when the author was an intern in Microsoft.

Furthermore, with the exponential growth of information and increasing diversity in terms of devices and networks in today's web, it becomes increasingly pressing to access desired contents accurately and conveniently. Various content adaptation technologies [1-4][13][14] have been developed for this purpose. One crucial issue for content adaptation is the need to effectively represent and understand presentation structure of web pages. Many web content analyzers have been proposed to extract structural information from web pages either manually [5-8] or automatically [9-11]. However, a big problem with these approaches is that they try to extract structural information from HTML tag tree directly, which often leads to unstable results because HTML tags were designed for both presentational and structural representation. Besides, most of them are bottom-up approaches which are time consuming. In addition, these approaches are only suitable for HTML documents.

To solve these problems, we first propose the web content structure, which attempts to represent author's presentation intention by identifying the logic relationship of web content based on visual layout information. An automatic top-down, tag-tree independent approach to detecting web content structure is then presented. It simulates how a user understands the web layout structure when he browses a page such as objects' size, position, color, background, etc. A projection-based algorithm is applied to segment a web page into blocks. Blocks are further divided into sub-blocks or merged if they are visually similar. In this way we avoid the breaking of logical chunks. Comparing to other existing approaches, our approach is independent of physical realization and performs even when the physical structure is different from visual layout structure. Besides, our approach is an $O(n)$ -time process which is much more efficient comparing to the $O(n^2)$ -time process in [11] (n is the number of basic objects). In addition, our approach is tag tree independent, which can be applied to contents with various physical realization formats.

This paper is organized as follows. In Section 2 we will first introduce the web content structure. It is followed by the automatic detection of web content structure in Section 3. Experimental results and conclusions are given in Section 4 and 5, respectively.

2 Web Content Structure

When designing a web page, the author first collects all the basic objects for the page. A basic object is the smallest unit of a web page, which cannot be further divided to perform some certain functions. He then groups related basic objects together to achieve a major function. This group of objects is called a composite object. Composite objects can be further grouped into a more complex one. This is a recursive process until all the objects in a web page are grouped together. During this process the author also needs to consider how to visually separate the composite objects.

Based on the analysis above, to fully express the presentation design of a web page, web content structure should represent the layout structure of a web page, which is the result of dividing and subdividing the content of a web page into increasingly smaller part, on the basis of presentation. Layout structure includes the structure of composite

objects and how they are separated from each other. Similar to the description of document representation in [12], the basic model of web content structure is described as below:

A web page Ω can be represented by a triple

$$\Omega = (O, \Phi, \delta) . \quad (1)$$

where $O = \{O_1, O_2, \dots, O_n\}$ is a finite set of objects, including basic objects and composite objects. Each composite object can be viewed as a sub-content structure. $\Phi = \{\Phi_1, \Phi_2, \dots, \Phi_t\}$ is a finite set of separators in visual representation, including horizontal and vertical separators. δ is the relationship of every two objects in O and can be expressed as:

$$\delta = O \times O \rightarrow \Phi \cup \{ \text{NULL} \} . \quad (2)$$

Suppose $O_i, O_j \in O$, $\delta(O_i, O_j) \neq \text{NULL}$ indicates that O_i and O_j are exactly separated by the separator $\delta(O_i, O_j)$, or we can say that the two objects are adjacent to some extent. Otherwise there exists other object between O_i and O_j .

Based on the definition above, authors can easily represent their presentation design in a formulas manner.

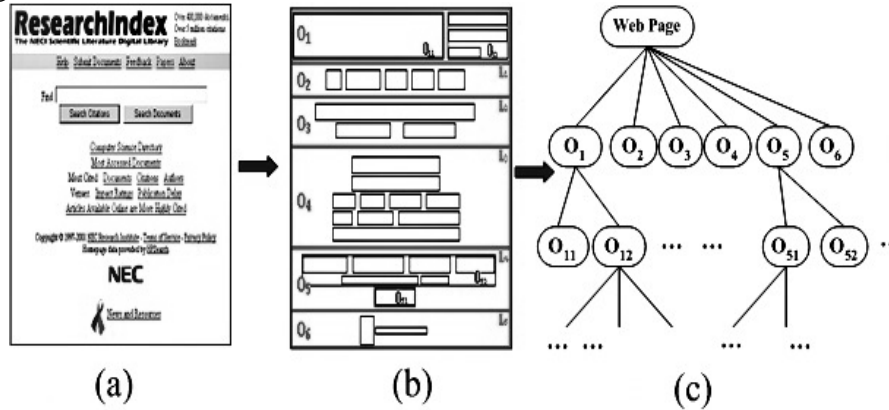


Fig. 1. An example of web content structure

Fig. 1 gives an example of web content structure, in which

$$O = \{O_1, O_2, O_3, O_4, O_5, O_6\}$$

$$\Phi = \{L_1, L_2, L_3, L_4, L_5\}$$

$$\delta = O \times O \rightarrow \Phi : \delta \begin{pmatrix} (O_1, O_2) \\ (O_2, O_3) \\ (O_3, O_4) \\ (O_4, O_5) \\ (O_5, O_6) \\ else \end{pmatrix} = \begin{pmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \\ L_5 \\ NULL \end{pmatrix}$$

In the page L_1-L_5 are all horizontal separators, O_1-O_6 are all Composite Objects, which can be further subdivided. For example, O_1 can be divided into two sub-blocks O_{11} and O_{12} by a vertical separator.

3 Automatic Web Content Structure Detection

Since web content structure reflects authors' intention directly, it plays a crucial role for content understanding. Therefore, detecting web content structure from existing web pages is very important for content adaptation. Although it is desirable that additional information be added for the generation of web content structure during the authoring stage, authors often decide not to do so. Thus, it is important that web content structure can be automatically detected.

In this section, we describe a top-down tag-tree independent approach to detecting web content structure based on page visual presentation. Our proposed approach simulates how a user understands the web layout structure when he browses a page. Below is the detailed description to the algorithm.

3.1 System structure

Web content structure detection is a reverse process to web authoring. We start from physical structure of web content to find out the presentation scheme. The detection of web content structure is to analyze and establish the quintuple of (1), i.e. O , Φ and δ defined in Section 2.

Fig. 2 illustrates the framework of our approach. The detection process is the division and repeated subdivision of a web page into increasingly smaller parts (objects). The detected structure can be visually represented by a geometric tree as in Fig. 1 (c). To construct web content structure, first basic objects are extracted from the physical structure (tag tree). We then preprocess the basic objects to find out decoration objects and group similar ones to reduce complexity. Then based on web visual representation, the whole page is divided into blocks through projection. Adjacent blocks are merged if they are visually similar. This dividing and merging process continues recursively until the layout structure of the whole page is constructed. Below we will first introduce the two major components in our system: dividing and merging. We then give a brief introduction to preprocessing module.

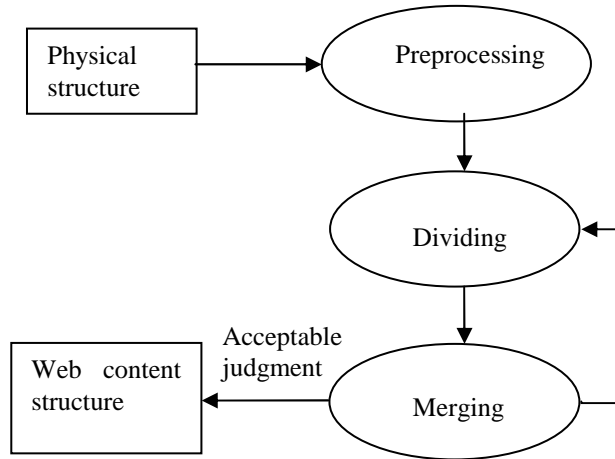


Fig. 2. Web content structure detection

3.2 Dividing Blocks via Projection

Projection is applied to divide a block into smaller sub-blocks. Projection refers to the mapping of a web page into a waveform whose value is the sums of the values of the object weights along projection axis. All objects in a web page are contained in rectangular blocks. Blanks are placed between these rectangles. Thus, the projection profile is a waveform whose deep valleys correspond to the blank areas of the web page. A deep valley with a width greater than an established threshold can be considered as separator between objects. The process of projection is performed recursively until all objects are located. Fig. 3 (a) gives an example of projection. Every object is projected along a projection axis. The sub-lines in the projection axis without any objects projecting into indicate the separators (see L_1 and L_2 in Fig. 3 (a)).

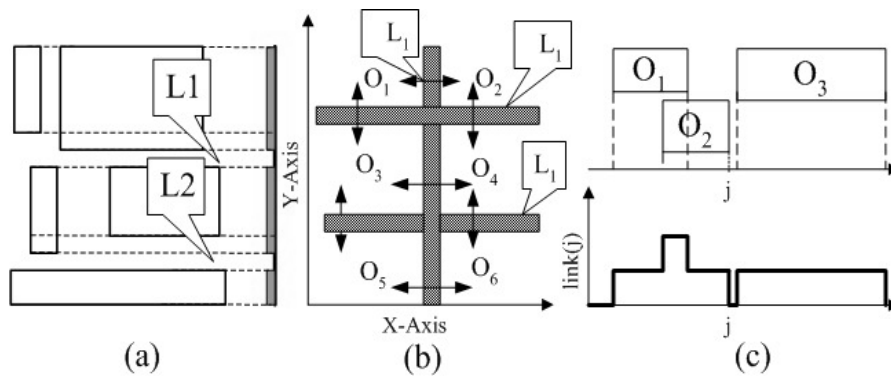


Fig. 3. Using projection to divide block

Fig. 3(c) explains the details of the projection process. Let $\{O_1, O_2, \dots, O_n\}$ be the n objects in a block, and s_i, e_i be the starting and ending point of O_i along a projection axis. Our goal is to find out all separators $\Phi = \{L_1, L_2, \dots, L_t\}$ along the projection axis. Let $\text{link}[j]$ be the judge value of point j at the projection axis. Then the weight value of O_i (1 by default) is added to $\text{link}[s_i]$. Similarly, the weight value of O_i is deducted from $\text{link}[e_i]$. Point j is considered a separator point if $\text{link}[j]$ is zero. Below is the detailed algorithm.

Step 1: Let C be the set of the starting and ending points of all the objects. $C = (\cup_i \{s_i\}) \cup (\cup_i \{e_i\})$, and $|C| = K$;

Step 2: Sort C into ascending order;

Step 3: Let $\text{link}[0] = 0$, we then project all the objects onto the projection axis:

for ($j = 1; j \leq K; j++$)

$$\text{link}[j] = \text{link}[j-1] + c - d;$$

where c is the number of objects with starting point at $C[j]$, and d is the number of objects with ending point at $C[j]$. This repetitive expression realizes the projection process in current block. If there are totally c objects starting at point j , the sum of the weights of these c objects should be added to $\text{link}[j-1]$ to get $\text{link}[j]$. Correspondingly, if there are totally d objects ending at point j , the sum of the weights of these d objects should be deducted from $\text{link}[j-1]$ to get $\text{link}[j]$.

Since default weight is equal to 1 for each object, the resulting values of the sum of weights are c and d , respectively. More fine-grained weight can be assigned to compute $\text{link}[j]$ for more accurate result.

Sub-line $C[j+1] \dots C[j+1]-1$ is a separator if $\text{link}[j]$ is 0. This means that there is no object in the area between Sub-line $C[j+1] \dots C[j+1]-1$.

Our algorithm requires $O(n)$ time because each step above costs $O(n)$ time. Comparing to the $O(n^2)$ -time required in [11], our approach significantly reduces the computational complexity.

Using the algorithm above, we detect separators $L_1[0 \dots t_1-1]$ in X-axis and $L_2[0 \dots t_2-1]$ in Y-axis. Then the division in current level is:

$$O = \{O_1, O_2, \dots, O_{(t_1+1)(t_2+1)-1}\}$$

Sub-blocks are indexed from left to right as shown in Fig. 3 (b);

$$\Phi = L_1[0 \dots t_1 - 1] \cup L_2[0 \dots t_2 - 1]$$

$$\delta(O_{i(t_1+1)+j}, O_{i(t_1+1)+j+1}) = L_2[j] \quad (0 \leq i \leq t_2, 0 \leq j < t_1)$$

$$\delta(O_{i(t_1+1)+j}, O_{(i+1)(t_1+1)+j}) = L_1[i] \quad (0 \leq i < t_2, 0 \leq j \leq t_1)$$

As can be seen in Fig.3(b), projection in Y-axis reveals two horizontal separators L_1 and L_2 while in X-axis reveals one vertical separator L_3 . These three separators divide current block into six sub-blocks: $O_1 \sim O_6$. The seven bidirectional arrows represent the relationship between sub-blocks and separators. Thus O, Φ, δ are all detected.

3.3 Merging sub-blocks via layout similarity

Since the division method above is only related to the position of objects, the separators detected may break a holistic object. Therefore the merging of some adjacent sub-blocks is necessary.

Simulating human's decision on whether two objects are similar, we use visual similarity to decide whether two objects are holistic and should be merged. For basic objects, if two objects are of different media type, their similarity $x=0$. Otherwise, x is related to the media type. Below is an example of basic text objects. In a similar way we can compute the similarity for basic objects of other media types.

- Starting from $x=1.0$, we first compare key HTML attributes (like $\langle H1 \rangle \dots \langle H6 \rangle$, $\langle A \rangle$). If not equal, $x = x * \text{Modifier_key}$.
- Compare alignment and other common attributes. If not equal, $x = x * \text{Modifier_Common}$
- Compare font size attribute. If not equal, $x = x * \text{Modifier_Size}$.
- Compare styles (bold, italic, underline...). If not equal, $x = x * \text{Modifier_Style}$.
- Compare font face. If not equal, $x = x * \text{Modifier_Face}$.
- Compare text length, we have the following modification method:

$$x = x * \left(\frac{\min(\text{length1}, \text{length2})}{\max(\text{length1}, \text{length2})} \right)^{\text{factor Adjust}}$$

To calculate the similarity of two composite objects, we use an approximate similarity measurement to compare two element strings that enables weighted mismatches and omissions (skips). Weight of skipping may differ from element to element, because some of the objects (such as those with attributes like $\langle H1 \rangle \dots \langle H6 \rangle$) could be more important than others, and thus skipping of them would be costly (a small weight) or not allowed (zero weight).

Some additional rules are used to modify visual similarity between objects for our application based on the definition of visual similarity above:

1) Distance is an important factor for human to decide visual similarity. Let d be the distance between two objects. The larger d is, the less their visual similarity is. Let x be the visual similarity obtained above, a monotonic degressive function $\text{Dist_Modifer}(d)$ is applied to embody the impact of distance to similarity:

$$x = x \cdot \text{Dist_Modifer}(d)$$

2) Color has great influence for human to decide visual similarity. Two objects with the same distinguished color and background color (different from those of surrounding objects) are considered as a whole. Thus, $x=1$ if adjacent objects have the same color and background color. Otherwise x remains its original value.

Based on the visual similarity between two adjacent objects, we can then decide whether to merge them or not.

3.4 Preprocessing

Preprocessing is necessary to prevent decoration objects from enveloping separators and prevent unexpected separators (e.g., shadowed separator in Fig. 4 is unexpected if the dotted rectangle is a tightly-united object) from breaking object integrity.

By preprocessing we can not only decrease the opportunity of detecting unexpected separators, but also reduce the problem scale thus speedup the whole process.

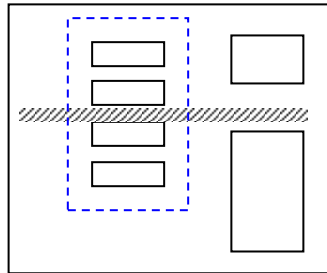


Fig. 4. Unexpected separator

4 Experimental Results

To evaluate our proposed approach, we randomly selected 50 web pages from popular sites listed in <http://www.yahoo.com>. We run our web content structure detection over these pages and the results are listed in Table 1.

Table 1. Experiment results with a test set of 50 pages

Detecting results	Number of document
Correct	45
Acceptable	4
Failed	1

Totally the layout structures of 45 (90%) pages are correctly detected. Some apparent chunks are missed for 4(8%) pages. In most of these cases, one logical chunk is broken into two or more segments. These segments are usually well separated by a big white space from surrounding segments. Human consider these segments as a whole chunk mostly according to their semantic meaning. Therefore, this is not strictly the fault of our algorithm. Our detector fails to provide proper analysis to the page be-

cause the page is confused in visual representation that even human eyes cannot partition it correctly.

One benefit with our approach is that it is independent of how the web page is realized in HTML tag-tree. Fig. 5 gives an example. The left side bar (indicated in dotted rectangle and was actually detected as a navigation list), which should be one logical chunk, is actually realized with two HTML tables, which leads to mis-detection in tag-tree based approach [11]. But with our approach the side bar is detected as a whole.



Fig. 5. Our approach successfully detects the side bar as a whole while tag-tree based approaches fail in this case because the tags are spread across different part of HTML file.

5 Conclusions

In this paper the web content structure was proposed for web authoring, adaptation and information retrieval. By identifying the logic relationship of web content based on visual layout information, web content structure can effectively represent authors' presentation intention. An automatic top-down, tag-tree independent algorithm to detect web content structure was presented. It simulates how a user understands the

web layout structure based on its visual representation. Comparing to other approaches, our method is independent of physical realization and works well even when the physical structure is far different from visual structure. Besides, our approach is an $O(n)$ -time process which is much more efficient than other approaches with $O(n^2)$ -time complexity. Experiments show satisfactory results.

References

1. Ma, W.Y., Bedner, I., Chang, G., Kuchinsky, A., and H.J.Zhang. A Framework for Adaptive Content Delivery in Heterogeneous Network Environments. in Proc. MMCN2000 (SPIE Vol.3969), San Jose, USA (2000) 86-100.
2. Chen, J.L., Yang, Y.D., and Zhang, H.J.: An Adaptive Web Content Delivery System, in Proc. AH2000, Springer (2000) 284-288.
3. Smith, J.R., Mohan, R., and Li, C.S.: Scalable Multimedia Delivery for Pervasive Computing, in Proc. of the 7th ACM International Conference on Multimedia (1999) 131-140.
4. Bickmore, T.W., and Schilit, B.N.: Digestor: Device-independent access to the World Wide Web, in Proc. WWW6 (1997) 655-663.
5. Hammer, J., Garcia-Monlina, H., Cho, J., Aranha, R., and A. Crespo: Extracting semistructured information from the web, in Proc. PODS/SIGMOD'97 (1997) 18-25.
6. Ashish, N., and Knoblock, C.: Wrapper generation for semi-structured Internet sources, in Proc. PODS/SIGMOD'97 (1997) 8-15.
7. Simth, D., and Lopez, M.: Information extraction for semi-structured documents, in Proc. PODS/SIGMOD'97 (1997) 117-121.
8. Nestorov, S., Abiteboul, S., and Motwani, R.: Inferring Structure in Semistructured Data, in Proc. PODS/SIGMOD'97 (1997) 39-43.
9. Embley, D.W., Jiang, Y., and Ng, Y.K.: Record-Boundary Discovery in Web Documents, in Proc. SIGMOD'99, Philadelphia PA (1999) 467-478.
10. Lim, S.J., and Ng, Y.K.: An Automated Approach for Retrieving Hierarchical Data from HTML Table, in Proc. CIKM'99, Kansas City, MO (1999) 466-474.
11. Yang, Y.D., and Zhang, H.J.: HTML Page Analysis Based on Visual Cues, in Proc. of the 6th International Conference on Document Analysis and Recognition, Seattle, USA (2001)
12. Tang, Y.Y., Cheriet, M., Liu, J., Said, J.N., and Suen, C.Y.: Document Analysis and Recognition by Computers, Handbook of Pattern Recognition and Computer Vision, World Scientific Publishing Company (1999)
13. Chen, J.L., Zhou, B.Y., Shi, J. Zhang, H.J., and Wu, Q.F.: Function-based Object Model Towards Website Adaptation, Proc. of the 10th International World Wide Web Conference, Hong Kong, China (2001) 587-596.
14. Yang, Y.D., Chen, J.L., and Zhang, H.J.: Adaptive Delivery of HTML Contents, in WWW9 Poster Proceedings (2000) 24-25.