
Web content adaptation for mobile device: A fuzzy-based approach

Jeff J.S. Huang

Department of Computer Science and Information Engineering
Hwa Hsia Institute of Technology, Taiwan
E-mail: Jeff@cc.hwh.edu.tw

Stephen J.H. Yang*

Department of Computer Science & Information Engineering
National Central University, Taiwan
E-mail: jhyang@csie.ncu.edu.tw

Zac S.C. Chen

Department of Computer Science & Information Engineering
National Central University, Taiwan
E-mail: ggreaters@yahoo.com.tw

Frank C.C. Wu

Department of Computer Science & Information Engineering
National Central University, Taiwan
E-mail: 955202034@cc.ncu.edu.tw

*Corresponding author

Abstract: While HTML will continue to be used to develop Web content, how to effectively and efficiently transform HTML-based content automatically into formats suitable for mobile devices remains a challenge. In this paper, we introduce a concept of coherence set and propose an algorithm to automatically identify and detect coherence sets based on quantified similarity between adjacent presentation groups. Experimental results demonstrate that our method enhances Web content analysis and adaptation on the mobile Internet.

Keywords: Mobile content delivery; Content adaptation; Coherence set

Biographical notes: Jeff J.S. Huang received his PhD degree in Computer Science and Information Engineering from the National Central University at Taiwan in 2010. He is now an Assistant Professor of the Department of Computer Science and Information Engineering, Hwa Hsia Institute of technology, Taiwan. His research interests include e-Portfolio, e-learning, Web 2.0, CSCW, and CSCL.

Dr. Stephen J.H. Yang is the Distinguished Professor of Computer Science & Information Engineering, and the Associate Dean of Academic Affairs at the National Central University, Taiwan. Dr. Yang received his PhD degree in

Electrical Engineering & Computer Science from the University of Illinois at Chicago in 1995. Dr. Yang has published over 60 journal papers, and received the 2010 outstanding research award from National Science Council, Taiwan. His research interests include creative learning, 3D virtual worlds, App software, and cloud services. Dr. Yang is very active in academic services. He is currently the Editors-in-Chief of the International Journal of Knowledge Management & E-Learning, and the Associate Editor of the International Journal of Systems and Service-Oriented Engineering. Dr. Yang also served as the Program Co-Chair of APTEL 2011, ICCE 2010, TELearn 2009, ICCE 2009, IEEE SUTC2008, ICCE 2008, IEEE ISM2008, SDPS 2008, IEEE W2ME2007, IEEE CAUL2006, and IEEE MSE2003.

Zac S.C. Chen is a Ph.D. student in Computer Science and Information Engineering from the National Central University at Taiwan. His research interests include e-Learning, Web 2.0, CSCW, and CSCL.

Frank C.C. Wu is a Master student in Computer Science and Information Engineering from the National Central University at Taiwan. His research interests include e-Learning and web content adaption.

1. Introduction

Mobile devices such as PDAs and cell phones have been increasingly used for Internet (Huang, Yang, Huang, & Hsiao, 2010; Yang, Okamoto, & Tseng, 2008; Yang, 2006), for example by students to view online course contents regardless the places or time. Many content publishing tools provide content adaptation facilities that transform Web pages into proper formats before delivering them to different receiving devices (Yang, Zhang, Tsai, & Huang, 2010; Chen, Yang, & Zhang, 2010; Yang, Zhang, & Huang, 2008). This is because mobile devices have smaller screens, slower network connections, and less computing power. Therefore, we need to develop adaptable content to view and read easy on the mobile device. This requires that all web content be developed in a formalized way. However, a lot of Web contents have already existed and would continue to appear in HTML format. It is impractical to require all these HTML pages to be regenerated. Thus, how to make these large-screen-oriented HTML pages automatically and transparently adaptable and accessible to mobile devices is necessary yet highly challenging.

This research aims to address this problem by identifying atomic segments with tight semantic coherence in HTML contents and transforming them into appropriate formats based on device contexts. In contrast with other existing content adaptation techniques that focus on transforming of stored raw data content typically in XML format, our research is more efficient in content adaptation by parsing existing HTML pages and re-generating the original knowledge content.

The most challenging part is how to identify and detect atomic segments with tight semantic coherence from freely formatted HTML content. The semantic coherence is to indicate a group of semantically similar features or items of a collection in segments, which can be called as semantic coherence segments. These semantic coherence segments should be maintained as atomic units of the presentation content and should always be kept together on the same screen throughout any content adaptation process. Meanwhile, the associations between semantic coherence segments should be loosely coupled. Our previous study yielded an Object Structure Model (OSM)-based Unit-Of-Information (UOI) concept and technique, which can automatically decompose an HTML

page into a hierarchy of atomic UOIs that have to be displayed on the same screen (Yang, Zhang, Chen, & Shao, 2007). It presented an algorithm that can examine HTML tags and presentation layouts to group closely coupled presentation elements into UOIs. However, the experiments revealed that this syntax-oriented detection could not always lead to satisfactory results. In this paper, we introduce a concept of coherence set and propose an algorithm to automatically identify and detect coherence sets based on quantified similarity between adjacent presentation groups.

The remainder of the paper is organized as follows. First, we will discuss related work about content adaptation and decomposition methods. Second, we will talk about our coherence set concept and corresponding detection algorithm in more details. Third, we will present the adaptation module in our system. Finally, we will present our experiments and discussions to demonstrate the efficiency of the fuzzy based content adaptation algorithm.

2. Related work

The conventional approach to adapting Web contents for mobile devices is to provide specific versions (formats) of the same content for corresponding mobile devices. For example, a Web page may hold one HTML version supporting desktop devices and another Wireless Markup Language (WML) version supporting wireless devices. The approach is straightforward but labor-intensive and inflexible. Content providers have to prepare different layouts and formats for the same Web content, which results in tremendous overhead. Furthermore, any change in the content may result in consequent changes in every related version, which is highly inflexible and may easily cause inconsistency. Considering that Web contents often undergo frequent changes, the traditional approach is neither practical nor feasible for mobile content delivery.

To deal with the problem, many content adaptation prototypes have been built in the recent years. Among them, Yang, Zhang, and Huang (2008) proposed a middleware, called Segment Web Content Adaptation (SWCA), to perform content adaptation on any complex data types, in addition to text and graphic images. However, their assumption is that all Web contents are described in XML format and is available ahead of time. Burzagli, Emiliani, and Gabbanini (2009) discussed the issues related to Design for All (D4All), a developer-driven concept to build services of various device types. XML-based adaptation is the major example used to illustrate their concept. Berhe, Brunie, and Pierson (2004) presented a service-based content adaptation framework, in which an adaptation operator was introduced as an abstraction of various transformation operations such as compression, decompression, scaling, and conversion. Lemlouma and Layaida (2004) proposed an adaptation framework that defines an adaptation strategy as a set of description models, communication protocols, and negotiation and adaptation methods. However, the actual implementation of this approach is still in a primary phase. How to map from constraints to adaptation operators is still unsolved. The scalability issue is a bottleneck as well. Lee, Chandranmenon, and Miller (2003) developed a middleware-based content adaptation server providing transcoding utilities named GAMMAR, in which a table-driven architecture was adopted to manage transcoding services located across a cluster of network computers. However, its predefined table structure limited its extensibility for supporting new devices or transcoding methods.

Some other researchers focused on content decomposition methods. Chen, Xie, Ma, Zhang, Zhou, and Feng (2002) proposed a block-based content decomposition method for quantifying content representation, in which an HTML page was factorized

into blocks, each assigned a score denoting its significance. This method enabled content layout to become adjustable according to the region of interest, attention value, and minimum perceptible size. Ramaswamy, Iyengar, Liu, and Douglis (2005) proposed an efficient fragment generation and caching method based on detection of three features: shared behaviour, lifetime, and personalization characteristic. However, the smallest adjustable element in these two approaches was a composite of objects, i.e., text, image, audio, and video. Its granularity of decomposition is too large for mobile device screens, therefore not suitable for mobile content adaptation. Another approach is CC/PP, which stands for Composite Capabilities/Preferences Profile, and is a system for expressing device capabilities and user preferences. Using CC/PP, creators of Web devices and user agents can easily define precise user or device profiles. Moreover, Zhang, Zhang, Quek, and Chung (2005) proposed some extension to CC/PP to enable transformation descriptions between various receiving devices. However, their work requested that the original content already has multiple presentation versions.

Moreover, MobiDNA is an adaptation algorithm to improve readability of Web content by using a caching strategy to reduce browsing latency (Hua, Xie, Liu, Lu, & Ma, 2006). Its adaptation process can adjust the size of Web content according to semantic blocks, defined as continuous content units that do not include two or more fragments within their content scopes. The semantic relationships between content units are limited to physical connections in this approach. Another approach, XAdapter is an extensible content adaptation system (He, Gao, Hao, Yen, & Bastani, 2007), where Web content is classified into objects (structure, content, and pointer objects) and adaptation techniques for structure objects (e.g., HTML tables) and objects cannot be further divided at content adaptation. While most approaches maintain the coherence of contents as far as possible, Xadapter is poor in coherence detection although it can prevent blurring caused by shrinking the texts or images. Nevertheless, the visual coherence between objects may be broken after content adaptation, because this approach does not consider whether some layouts cannot be rearranged.

In our previous research (Yang, Zhang, & Chen, 2008), we presented a JESS-enabled context elicitation system featuring an ontology-based context model to formally describe and acquire contextual information pertaining to service requesters and Web services. Additionally, a rule-based adaption strategy to enhance web content adaption based user's contextual requirements was proposed by Yang and Shao (2007). In Yang, Zhang, Chen, and shao (2007), we presented a UOI-based content adaptation method, which can automatically detect semantic relationships among comprising components in Web content, and then reorganize page layout to fit handheld devices based on identified UOIs. In Yang, & Chen, 2008; Su, Yang, Hwang, & Zhang, (2010), we presented a web page content adaptation to support interactive and collaborative learning in knowledge sharing by using mobile devices. However, our experiments revealed that this syntax-oriented detection may not always lead to satisfactory results. In this paper, we introduce a concept of coherence set and propose an algorithm to automatically identify and detect coherence sets based on quantified similarity between adjacent presentation groups.

3. Major coherence set identification and detection

3.1. Definitions

Definition 1. A presentation object, or an object *o*, refers to the minimum presentation unit of a Web page, containing semantic meanings and cannot be further divided in our

process. A group g refers to a collection of objects in a table row, which have high visual coherence and should always be kept in adjacent locations. A coherence set s implies that two or more groups have high visual coherence and the layout of them cannot be adapted. A coherence threshold is a boundary for deciding which groups should be included in a coherence set.

Our definitions imply two declarations. First, the size and location of an object can be adjusted. Second, adjacent groups shall be identified as a coherence set, if their similarity values exceed a predefined coherence threshold.

Example objects are shown in Fig. 1.(a). Object 1 and object 3 are two text areas; Object 2 is a picture. In its corresponding HTML code, a text area is delimited by an HTML tag `<TD>`; and a picture is delimited by an HTML tag ``. As shown in Fig. 1.(a), Object 1 represents the title of the picture and object 3 represents the caption of the picture. These three objects cannot be further divided without breaking their semantic meanings. However, their sizes can be adjusted.

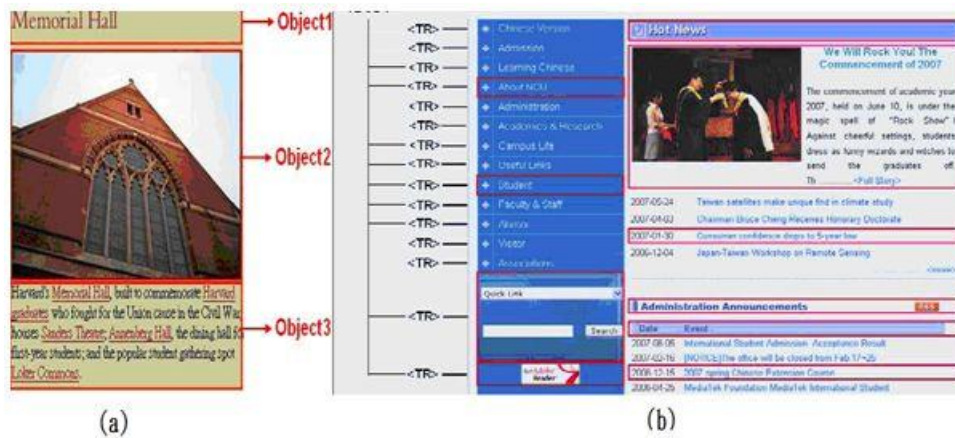


Fig. 1. Examples of objects

In its corresponding HTML code, a table row can be viewed as a group and be detected by catching the HTML tag `<TR>`. Fig. 1.(b) shows the individual groups of a Web page, together with their relationships with `<TR>` tags. Some groups are enclosed by red frames to help readers recognize them. The objects in one group can be moved together, but must be kept adjacent otherwise the connection between them may get lost.

The notation of group ensures that presentation objects with horizontal relationships be kept adjacent. However, the coherence breaking problem may still exist by splitting the adjacent objects in different groups. As shown in Fig. 2.(a), the adjacent objects “O1” and “O2” belong to the same group; same for “O3” and “O4.” If a simple single-column adaptation rule is applied, although O1 and O2 are still adjacent (so are O3 and O4), originally adjacent O1 and O3, O2 and O4 are separated. Therefore, it is necessary to identify the visual coherence of adjacent groups. Before a single-column adaptation process, we must confirm that the objects in different groups have no visual coherence.

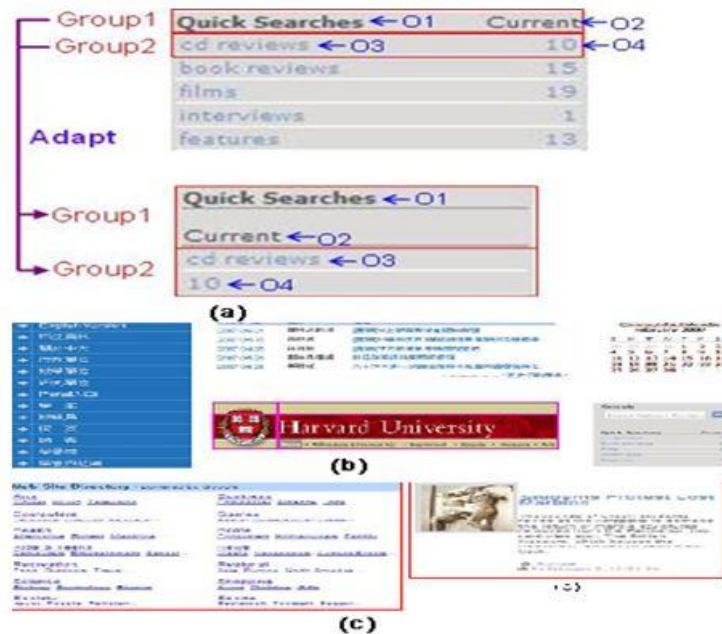


Fig. 2. Coherent set and semantic group

We thus introduce a concept of coherence set to specify that two or more groups have high visual coherence and their layout cannot be changed. Fig. 2.(b) shows some examples of coherence sets: a drop down menu, university icons, a calendar, etc. As shown in Fig. 2.(b), each coherence set comprises multiple groups (rows), and their relative positions have to be retained. To avoid from breaking the connection between them, it is the best that we do not move the objects in a coherence set.

A semantic block is a discrete chunk of information that conveys a specific type of information or serves a specific meaningful purpose within the overall structure of a topic. Literature and our previous work focus on semantic blocks (Yang et al., 2010; Hua et al., 2006), as shown in Fig. 2.(c). The core difference between a semantic block and a coherence set is that, the items in a semantic block may be flexible to be adapted (e.g., items in Yahoo! Can be adapted into one column) and items in a coherence set cannot be moved (e.g., dates in a calendar have to stay in the 7-column format). In other words, a coherence set may comprise multiple semantic blocks whose relative positions are fixed. In other words, a coherence set implies a stronger relationship between presentation objects.

The challenge now turns into how to identify coherence set with groups. After careful examinations, we found that groups with high visual coherence usually exhibit similar HTML attributes. For example, several groups form a list (e.g., in a calendar shown in Fig. 2.(b)); or they show similar functions such as hyperlinks (e.g., in a drop down menu shown in Fig. 2.(b)). Based on our observations, we hypothesize that similar adjacent groups may have presentation coherence, so that we can group them into a coherence set. Then the question is how to calculate quantified similarity between adjacent groups. We propose an algorithm that will be discussed in detail in the following sections. According to the obtained similarity value, we determine which groups have high visual coherence based on fuzzy inductive reasoning. Coherence threshold is a

predefined boundary that we introduce to enable automatic calculation and decision process.

3.2. Algorithm for coherence set detection

3.2.1. Similarity quantification algorithm

We examine and compare the HTML attributes of every pair of adjacent groups to quantify similarity between them. Our algorithm is derived from the Longest Common Subsequence (LCS) algorithm (Cormen, Leiserson, Rivest, & Stein, 2009), which is commonly used for finding the longest sequence that is a subsequence of all possible sequences. To evaluate the similarity value between two groups, the inputs of our algorithm are two sequences of their attributes. We define the similarity value as the proportion of LCS length of the two attribute sequences to the total length of them. Assume that “G1” and “G2” denote two adjacent groups and “S” denotes the similarity between them. The evaluation formula is shown as follows. The pseudo code of the similarity quantification process is summarized in Fig. 3.

$$\text{Similarity}(G_1, G_2) = \frac{L_l \times 2}{L_1 + L_2}$$

Where,

G_1 : group 1

G_2 : group 2

L_1 : the length of group 1

L_2 : the length of group 2

L_l : the length of the longest common subsequence of L_1 and L_2

The algorithm iteratively parses each group and catches and extracts its contained attributes and their values, and conducts the comparison. The method checkLevel() determines the level of the groups' similarity according to the predefined primary threshold and two secondary thresholds.

Here we use an example to explain how our above algorithm works. Fig. 4 shows 2 groups as input. For each group, every attribute name and its corresponding value is identified as an independent element and is assigned a capital letter. For example, in Group 1, attribute name “vAlign” is assigned letter “A;” its attribute value “top” is assigned letter “B.” As shown in Fig. 4, the same names or values in different groups are assigned the same letter. For example, Group 1 and Group 2 both comprise the same attribute name “vAlign,” which is assigned letter “A” in both groups. If two groups both contain the same attribute name and same attribute value, they share the same letters for their attribute name and attribute value. For example, Group 1 and Group 2 both have letter “A” and “B,” because they both have attribute name “vAlign” and corresponding value “top.” On the other hand, if two groups both contain the same attribute name but their corresponding attribute values are different, the two groups will share the same letter for their attribute name but different letters for their attribute values. For example, Group 1 and Group 2 both contain attribute name “alt,” so they both contain the letter

“G.” However, since the attribute values for the two groups are different (“Simmons” for Group 1 and “PantinG” for Group 2), letter H is assigned to Group 1 and letter K is assigned to Group 2. Note that the position of an attribute name or value in a group does not affect its letter assignment. For example, the attribute “vAlign” locates at different positions in Group 1 and Group 2; however, they are assigned the same letter “A.”

```

Analysis_Procedure( G1, G2 )

Length ← G1.length + G2.length
Similarity ← LCS( G1.right, G2.right ) × 2 / Length

LCS( Attr1, Attr2 )
  If ( Attr1 = null or Attr2 = null )
    Return 0
  If ( Attr1.value = Attr2.value )
    Return LCS( Attr1.Right, Attr2.Right ) + 1
  If ( Attr1.value != Attr2.value )
    Return Max( LCS( Attr1.Right, Attr2 ),
               LCS( Attr1, Attr2.Right ) )

Level ← checkLevel( Similarity ) // Check the level of groups similarity
    
```

Fig. 3. Pseudo-code of similarity quantification algorithm

As shown in Fig. 4, the parsing process results in a string of letters for each group, e.g., “ABCDEFGHJIJ” for Group 1 and “ABEFGKIJCD” for Group 2. The length of such a string is the number of elements (letters) contained in it. Both Group 1 and Group 2 have a length of 10. By finding the overlapping letters in the two strings from the same direction, we obtain an LCS length of 7. Running our similarity formula, we can conclude that similarity between the two groups is 70%.

It should be noted that our algorithm does not merely count the number of the same elements in two groups. Instead, we take into consideration the relative order and arrangement of attributes under investigation. In other words, the order of paired attribute names or attribute values has to be the same. For example, as shown in Fig. 4, letters “A” and “E” are considered paired elements in the two groups. In Group 1, element “A” is prior to element “E”; therefore, in Group 2, element “A” must be prior to element “E” and cannot be vice versa. Therefore, even if the numbers of the elements in two groups are the same, if their orders are different, then their similarity value may not be 100%. Fig. 5. shows such an example. Two groups each comprise two objects: a text area and an image. If we examine their HTML specifications, the attributes of the two groups are almost the same. If we just count the number of the same elements, the similarity value between these two groups will be close to 100%. Nevertheless, as shown in Fig. 5, the arrangements of the groups are different and the similarity value between the groups should be low. Our algorithm is designed to solve this kind of situation.

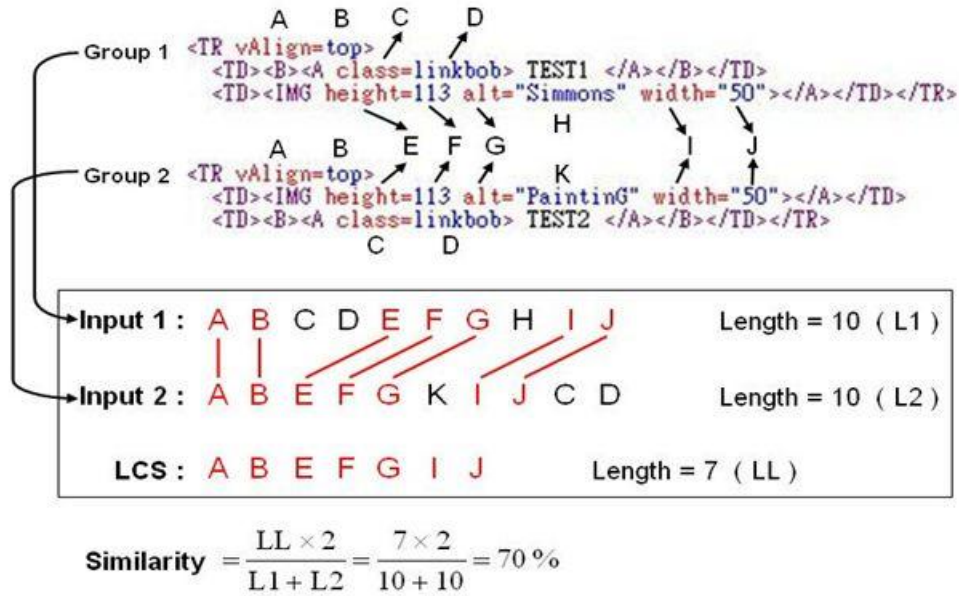


Fig. 4. An example of evaluating similarity of two groups



Fig. 5. Same objects in different order

The action of evaluating the similarity between two adjacent groups is a normalization-like process that quantifies the similarity value within the range of [0, 1]. We need to set a coherence threshold to help make decision for our adaptation strategy. If the similarity value between two groups exceeds a predefined threshold value, the groups will be grouped into a coherence set. Apparently, the setting of the threshold value may significantly affect the accuracy of content adaptation.

3.2.2. Determine the coherence threshold

We adopt the technique of fuzzy inductive reasoning (Chen, Yang, & Zhang, 2010; Reed, & Lim, 2002; Tsai, Cheng, and Chang, 2006) to help automatically identify coherence

sets. In more detail, we classify our statistical sample and explore a reasonable coherence threshold value. Our key idea is to evaluate the entropy that is a measure of the disorder in a sample, and then classify the sample while minimizing the entropy for an optimum partitioning. In other words, the threshold value with minimal entropy is the best value for classifying the sample.

To build such a sample, we first collect a number of HTML Web pages. Then we calculate the similarity value between each pair of adjacent groups using our algorithm introduced in the previous section, and manually decide whether the two groups belong to a high-coherence group or a low-coherence group based on their visual coherence. For each pair of evaluated groups, we use a record to track the calculated similarity value and our visual decision. Our built sample bed contains 1043 records. Fig. 6. illustrates a graph section that represents our sample bed. Each circle represents two adjacent groups; the location of a circle indicates its associated similarity value. If a circle locates right of another circle, it means that the former has higher associated similarity than the latter. Then we use visual coherence value to mark each circle. A black circle represents a high-coherence group pair that is not suitable for layout rearrangement; a white circle represents a low-coherence group pair that can be adapted and rearranged. As shown in Fig. 6, calculated similarity values and visual coherence values may not always match. For example, a black circle appears in the low-similarity region; and two white circles appear in the high-similarity region.

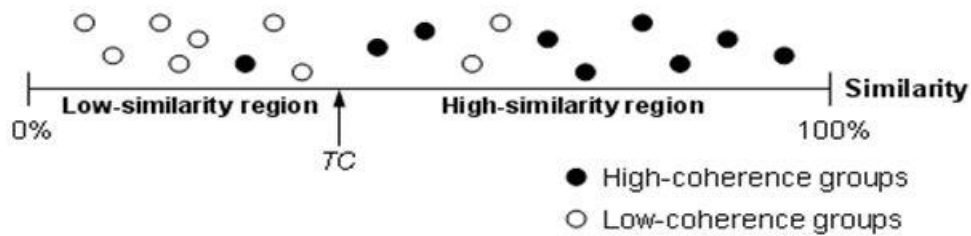


Fig. 6. A sample for fuzzy inductive reasoning

To seek the optimum threshold for the sample, we move an imaginary threshold candidate TC_i between 0% ~ 100%, and calculate the entropy for each TC_i to explore the minimal entropy. As shown in Fig. 6, the data are divided into two regions by TC_i . We calculate the entropy(S) over TC_i using the formula below:

$$S(TC_i) = L(TC_i)S_L(TC_i) + H(TC_i)S_H(TC_i) \quad (1)$$

$$S_L(TC_i) = -[L_1(TC_i) \ln L_1(TC_i) + L_2(TC_i) \ln L_2(TC_i)] \quad (2)$$

$$S_H(TC_i) = -[H_1(TC_i) \ln H_1(TC_i) + H_2(TC_i) \ln H_2(TC_i)] \quad (3)$$

Where i iterates from 1 to n and assume TC_i as the threshold,

$S(TC_i)$: overall entropy,

$S_L(TC_i)$: Entropy of the low similarity region,

$S_H(TC_i)$: entropy of the high similarity region,

$L_1(TC_i)$: probability that low coherence groups fall in the low similarity region,

$L_2(TC_i)$: probability that high coherence groups fall in the low similarity region,

$H_1(TC_i)$: probability that low coherence groups fall in the high similarity region,

$H_2(TC_i)$: probability that high coherence groups fall in the high similarity region,

$L(TC_i)$: probability that both kinds of groups fall in the low similarity region,

$H(TC_i)$: probability that both kinds of groups fall in the high similarity region

Formula (1) intends to calculate the total (global) entropy of the entire sample bed by the sum of the two regions' entropy with respective proportions. The two regions' individual entropy can be calculated using formula (2) and (3), respectively.

After calculating all the entropy values for each threshold candidate TC_i , the TC_i with the minimal entropy becomes the best threshold. Fig. 7 shows the distribution of entropy with different TC_i . The horizontal axis represents calculated similarity values; the vertical axis represents entropy. The blue curve shows the entropy changes for the low-similarity region and the purple curve shows the entropy changes for the high-similarity region. The two curves are shaped by formula (2) and (3), respectively. The total entropy (yellow curve) is calculated by formula (1). Fig. 7. shows that the total entropy becomes the lowest when the similarity value is about 0.5217. This value is thus the best threshold.

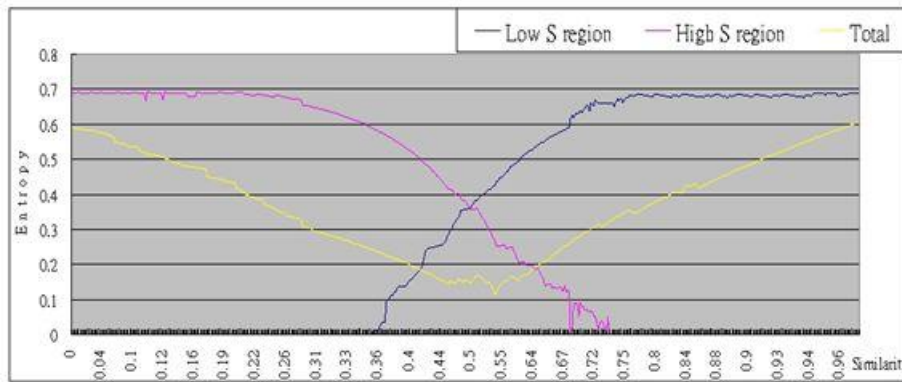


Fig. 7. A graph to represent the entropy of our statistics

3.3. Adaptation strategy

We further divide similarity values into four levels to represent different degrees of visual coherence. As shown in Fig. 8, we set three threshold values, including one primary threshold value (P) and two secondary threshold values (S1 and S2) in the horizontal similarity axis. The primary threshold is calculated using the method introduced in section 3.2; and the two secondary threshold values are set using the following algorithm:

S1: The goal is to ensure that the entropy of low-similarity region is low. Most groups in low-similarity region have low coherence. The point of S1 is set when the entropy of the blue curve just hits zero toward the left side.

S2: The goal is to ensure that the entropy of high-similarity region is low. Most groups in high-similarity region have high coherence. The point of S2 is set when the entropy of the purple curve just hits zero toward the right side.

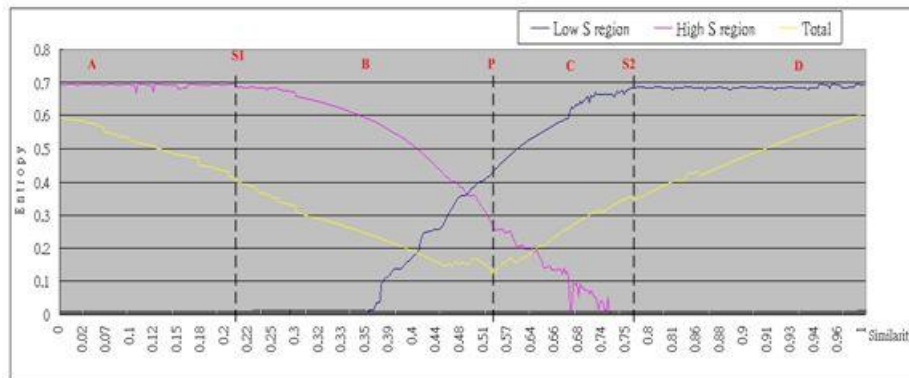


Fig. 8. Four levels of similarity values

Using the training results, S1 became about 0.2031 and S2 became about 0.7651. The similarity is thus classified into four regions by the threshold values: A, B, C, and D. For region A, we are almost certain that the contained groups are unrelated. For region B, most of the groups are unrelated but some exceptions may exist. For region C, most groups are related but some exceptions may exist. For region D, we are almost certain that the groups are related. In most cases, the similarity of low-coherence groups is lower than the primary threshold value (i.e., at region A or B). On the contrary, the similarity of high-coherence groups is higher than the primary threshold value (i.e., at region C or D).

According to our observations, most of exceptional cases in regions B and C belong to one of the following two situations: multi-title and exhibition. Fig. 9.(a) shows an example of multi-title situation. A table comprises two rows. The first row contains several titles; the second row contains the corresponding values according to the titles. Such an HTML specification makes it difficult to catch the column structure. The similarity of the rows (groups) is low, so that they may be considered unrelated. However, we can see that in each row, the elements are very similar to each other. As a result, such a structure should be considered as highly coherent. To detect such a situation, we can further examine whether the objects in each row have high similarity and whether each row has the same number of cells. Note that the titles may be images as well; as a result, we may have to check whether the media types between the cells are the same.

An exhibition situation means that each cell of a table has the same function or characteristic. Fig. 9.(b) shows an example of exhibition situation, where a website contains multiple pictures displayed in three rows. These pictures shall be put together, obviously. However, there is no need to force all pictures to be put in a particular order, because they all show the same meaning for this Web page. This is a special case where all objects are related to each other without limitation of presentation orders, so that their layout can be adapted. This situation is hard to detect because the attributes of images are always irrelevant to their meanings. However, we can still try to identify this situation according to some informal rules, to further analyze detailed structures and properties between adjacent groups. For example, if a table contains a significant set of pictures, we may further examine whether it shows an exhibition case. Although the exception handling further improve the precision of our content adaptation, it consume extra execution time. Therefore, we leave the exception handling component to be flexible, so that this function can be disabled if users consider time efficiency is critical. Additionally, users may have different browsing habits and favourite websites. Therefore, we also consider that users may want to change the degree of adaptation to better browse the Web

sites they constantly visit. To provide this flexibility, our system provides three levels of content adaptation: weak, normal, and strong. The weak level maintains most coherence of content using the main strategy of resizing. It has a high risk of blurring the content, though. The normal level is the default degree of our system, where the system decides the strategy based on the primary threshold. The strong level means that our system will adapt the layout using all predefined thresholds. It has a potentially higher probability of breaking the coherence of content.



Fig. 9. Two exceptional cases

Finally, we discuss the complete procedure of our adaptation system. It is common that the structure of a Web page has multiple layers, e.g., a table contains several smaller tables embedded. Our adaptation procedure starts from the outmost table and recursively goes into inner layers. The reasons are two-fold. First, the tables in an outer layer always have lower visual coherence. Even if we rearrange these tables, the risk of coherence breaking is low. Second, an outer layer contains fewer tables and the table cells are typically larger, so that it may take less time to analyze them thus to reduce more content widths when we transform these tables into single columns. For each iteration, we target on one table, adapting or maintaining the table cells contained. The adaptation procedure will repeat and extend to the inner layers until the content width is smaller than the receiving screen size. Pseudo code of our procedure for analyzing one table is summarized in Fig. 10. It represents a round of adaptation procedure aiming at one table. Repeat the above three steps for other tables until the width of the entire content fits the screen size.

```

Adaptation_Procedure()

  Insert_Groups(); //create the data structure

  idx ← Table.left
  While ( idx.left != null )
    If ( idx.Width > screenSize )
      Analysis_Procedure( idx, idx.left ) // analyze two groups
    idx ← idx.left

  idx ← Table.left
  While ( idx != null )
    If ( ! idx.isFixed )
      Partition( idx ) // layout adaptation
    If ( ! idx.Width > screenSize )
      Objects_Resizing( idx ) // Shrink the objects
    Idx ← idx.left

```

Fig. 10. Pseudo code of adaptation procedure

The parameter “idx” is a pointer that indicates a location in the data structure. First, a data structure including group nodes and attribute nodes is created. Second, the procedure checks the contained groups’ widths. Afterwards, the procedure focuses on the groups, whose widths exceed the screen’s width, and performs Analysis_Procedure() method that is discussed in Section 3.2. The value of the parameter “isFixed” in each group is assigned after the analysis procedure is performed. The procedure will check the parameter “isFixed” to determine which groups can be rearranged by applying single-column adaptation. If “isFixed” is a false, the Partition() method will be executed to adapt the groups’ layout. After layout adaptation, if the width of a group is still greater than the screen size, the objects in the group will be shrunk through Objects_Resizing() method. If the group has no objects, it means that the group is not in the most inner layer and the resizing process will not be executed. Fig. 11 shows detailed adaptation rules. The parameter “Degree” represents the degree of layout adaptation upon users’ preferences. The parameter “isFixed” may be changed according to the degree of layout adaptation. If the condition meets, the “isFixed” of both groups will be changed to “true.”

Sometimes, it is hard to calculate the width of a table, because the author of the content may not assign the “width” attribute. In this situation, our system will take a default guess and consider that the width of the table exceeds the screen range thus requires adaptation.

Our observations reveal that the coherence breaking problem is always caused by moving objects, not tables. Thus, our system mainly analyzes the most inner layer because the objects are included in the most inner tables. For other layers, the tables will be moved directly to save some time of content analysis. As a matter of fact, the visual coherence between the tables is always low, and our single-column process will keep the groups adjacent with each other. Therefore, we may ignore most analysis for outer tables.

```

If ( Degree == "Weak" ) // Weak degree of layout adaptation
  If ( Level != A )
    G1.isFixed ← True
    G2.isFixed ← True
  If ( excHandle == True && Level == A ) // Exception handling
    If ( isMulti-title( idx, idx.Left ) ) // Detect multi-title structure
      G1.isFixed ← True
      G2.isFixed ← True
  If ( excHandle == True && Level == C or D )
    If ( isExhibition( idx, idx.Left ) ) // Detect exhibition structure
      G1.isFixed ← False
      G2.isFixed ← False

If ( Degree == "Weak" ) // Weak degree of layout adaptation
  If ( Level != A )
    G1.isFixed ← True
    G2.isFixed ← True
  If ( excHandle == True && Level == A ) // Exception handling
    If ( isMulti-title( idx, idx.Left ) ) // Detect multi-title structure
      G1.isFixed ← True
      G2.isFixed ← True
  If ( excHandle == True && Level == C or D )
    If ( isExhibition( idx, idx.Left ) ) // Detect exhibition structure
      G1.isFixed ← False
      G2.isFixed ← False

Else // Strong degree of layout adaptation
  If ( Level == D )
    G1.isFixed ← True
    G2.isFixed ← True
  If ( excHandle == True && Level == A ) // Exception handling
    If ( isMulti-title( idx, idx.Left ) ) // Detect multi-title structure
      G1.isFixed ← True
      G2.isFixed ← True
  If ( excHandle == True && Level == C or D )
    If ( isExhibition( idx, idx.Left ) ) // Detect exhibition structure
      G1.isFixed ← False
      G2.isFixed ← False

```

Fig. 11. Adaptation rules

4. System implementation

4.1. System architecture

A proxy “VCAProxy” was implemented to perform content analysis and adaptation. The operating system (Microsoft Windows XP) was installed with software platforms including: proxy server, Java JVM and MySQL. The client devices: HP iPAQ Pocket PCs h5500 with Microsoft Windows C.E. 3.0 operating system, Sony Ericsson P900 smart phones with Symbian operating system. We chose Java as our programming language to develop our system, mainly due to its cross-platform compatibility. The architecture of our system is based on proxy. After examining several open-source Java proxy candidates, we selected Muffin (n.d.) as our proxy server. Muffin is a relatively

light-weight proxy supporting PHP and AJAX. It can be executed on Unix, Windows 95/NT, and Macintosh. It also supports several network protocols such as HTTP/1.0, HTTP/1.1, and SSL. The architecture of our system (Fig. 12.) and executing flow was described as follow:

1. A CAgent is embedded at client side, which goal is to extract users' contexts (e.g., the resolution of screen and users' preference settings) and send them to the Context Management module via the Message Controller that is in charge of data delivery.
2. The Context Management module is to analyze users' contexts and check if adapted pages already exist in the Cache. If they exist, the cached pages will be sent to users' devices directly.
3. Otherwise, the Message Controller will fetch the required Web content from the server and send the content to the Adaptation Module, which comprises an analysis module.
4. After the adaptation process, a suitable version of the content is generated and stored in the Cache.
5. Finally, the adapted content will be sent to users' browser via the Message Controller.

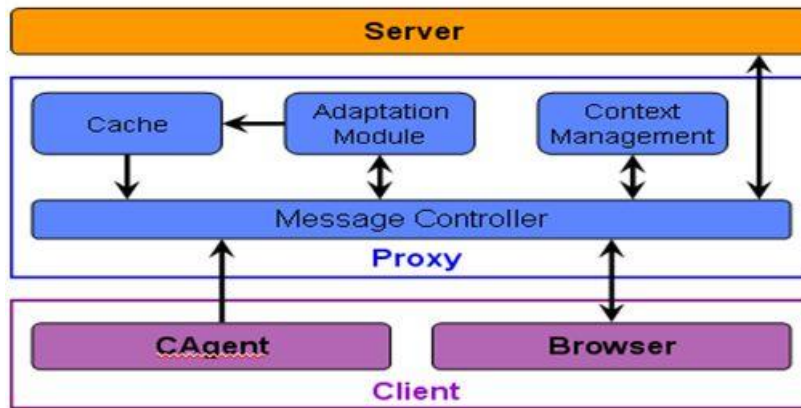


Fig. 12. System architecture

Even if the CAgent is not installed on clients' devices, the Context Management module still can analyze incoming requests and detect the type of users' devices and the versions of their browsers, then roughly determine the devices' screen size. As shown in Fig. 12, our system is set up as a proxy server. If a user inputs our proxy address, our proxy will automatically provide adapted pages to the user in real time.

5. Experiments results

In this section, we first describe our experiment design with a scenario, and then discuss our system performance through quantitative.

5.1. Experiment design and setup

We selected 30 volunteers as testers who usually access the Internet via mobile devices. The participants are 30 college school students in Tainan County. We provided to testers mobile devices, including iPAQ Pocket PCs h5500 and Sony Eriksson P900 smart phones. We required that the testers access the Internet via our content adaptation system and provide some comments and feedback to us. In order to eliminate bias, we surveyed the testers for their often-time accessed websites. We then classified these websites into five categories: school, business, entertainment, news, and portal.

Based on our survey, we identified a list of experimental websites that each tester was required to access constantly and perform some tasks, including browsing the Web pages and locating specific information on the mobile devices. The list is summarized in Table 1. We also designed quantitative evaluations by automatically testing these experimental websites.

Table 1
The statistics of detection accuracy

Category	No.	Website name	High-coherence groups		Low-coherence groups		Accuracy
			Actual No.	Detection result	Actual No.	Detection result	
School	1	NCU	32	32	16	16	100%
	2	NKFUST	7	7	25	25	100%
	3	NCTU	15	15	11	11	100%
	4	Harvard	4	4	14	8	66.66%
	5	Berkeley	0	0	8	8	100%
Business	6	ASUS	14	14	15	8	75.86%
	7	IBM	0	0	9	9	100%
	8	Notiemail	8	8	31	31	100%
	9	Amazone	21	21	25	15	78.26%
	10	Ebay	0	0	16	14	87.5%
Entertainment	11	Flowgo	7	5	21	21	92.85%
	12	Flickr	14	14	10	10	100%
News	13	BBC	19	19	23	15	80.95%
	14	Google News	9	9	47	47	100%
Portal	15	Sify	18	16	26	26	95.45%
	16	My Way	3	3	19	19	100%
	17	Rediff	3	3	32	29	91.42%
	18	DIMUSEUM	0	0	8	8	100%
	19	Blackboard	24	20	36	36	93.33%
	20	Love to Know	18	16	20	17	86.84%

5.2. Quantitative evaluation

5.2.1. Precision

We conducted experiments to examine how accurately our system detects and identifies coherence sets in HTML Web pages. Our experimental websites and the testing data are summarized in Table 1: total number of high-coherence groups and low-coherence groups of the websites, and the number of groups detected by our system. We define the system accuracy as follows:

The actual number of high-coherence groups and low-coherence groups are determined by a manual visual process. We identified three dedicated volunteers to determine the coherence between groups. The rules they used are as follows: if the groups form a list or have a high degree of visual coherence, the groups will be grouped into a coherence set.

According to the statistics, accuracy of detection of our system is 92.45% in average. The number is significantly better than accuracy of our previous approaches UOI (78.49%). Note that although the UOI approach intends to detect semantic blocks that are different from coherence sets, they all share the same goal to detect semantic units with high coherence connections. As a matter of fact, the concept of a coherence set is stricter than that of a semantic block.

5.2.2. Performance comparison

In Fig. 13 shows some comparisons with and without content analysis. In both (a) and (b), the right screen shows the display after our content adaptation process; the left shows without.

In Fig. 14 shows comparisons of content adaptation performance between our approach and two related approaches, UOI and MobiDNA, based on the same testing platform and environment. The top of Fig. 14 shows the original Web content shown on desktops, where a search area in a portal Web page is associated with some dynamic functions. A user can select different ways of searching by clicking the buttons and the search area will be modified upon the user's selection. The adaptation results of our approach, UOI, and MobiDNA are shown in Fig. 14.(b), (c), and (d), respectively. In this example, MobiDNA approach ruins the appearance of the search area; UOI approach cannot maintain the function of modifying the search area. This may be because that wrong tags are inserted or the tags are put in a wrong way. In contrast, our approach maintains the original display and functions of the search area, by adapting layouts for a suitable presentation. Fig. 14 proves that our system maintains more features and functions in Web pages.

According to the statistics, accuracy of detection of our system is 92.45% in average. The number is significantly better than accuracy of our previous approaches UOI (78.49%). Note that although the UOI approach intends to detect semantic blocks that are different from coherence sets, they all share the same goal to detect semantic units with high coherence connections. As a matter of fact, the concept of a coherence set is stricter than that of a semantic block.

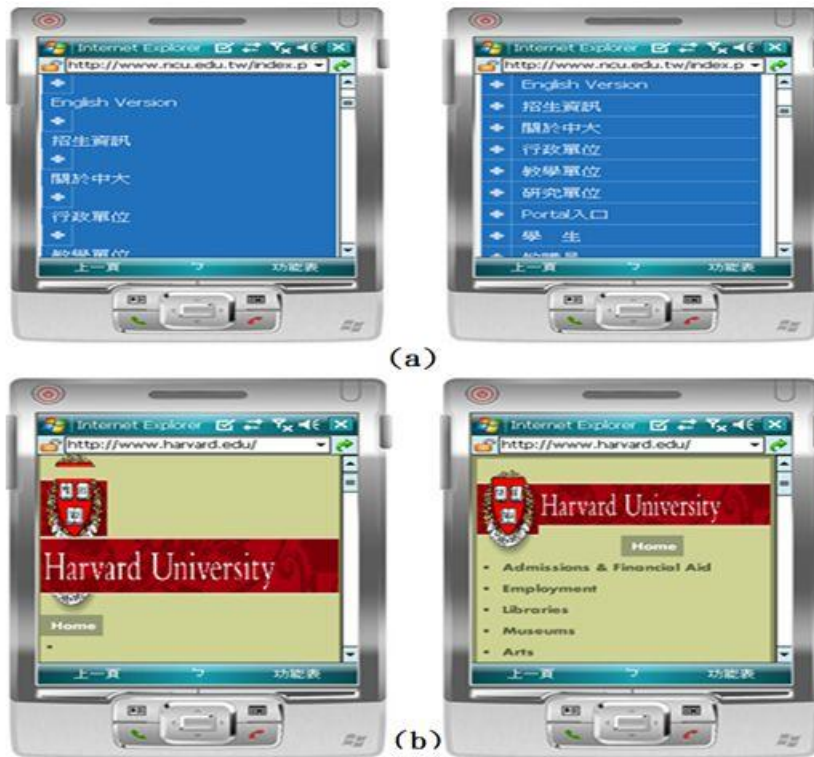


Fig. 13. Comparison with/without content analysis



(a)



Fig. 14. Comparison for dynamic function maintaining

6. Conclusions and future work

Experimental results demonstrate that our method enhances Web content analysis and adaptation on the mobile Internet. Specifically, our content adaptation technique can improve mobile Internet navigation. Without updating any previous Web content designed for desktop computers, our solution moves one step further to the ultimate goal of “One-for-All”. The myth of “Write Once, Show Everywhere” allows content providers to prepare a content page only once in HTML oriented to desktop computers; and it can be presented onto various devices with the help of our underlying content adaptation technique.

Based on the experimental results, we conclude that our coherence set-based content analysis and adaptation in Mobile computing is effective. We believe that when portable devices become more popular in Web browsing, our content adaptation technique will have more impacts on the enhancement of mobile Internet navigation. In general, our technique can be implemented as a standalone service as what we did in our experiments, or can be embedded as a plug-in component into existing Web content proxy servers.

Some future research includes the following. First, we only extracted and analyzed the “class”-level attributes in HTML source code to determine whether the groups have the same style sheet in this study. We plan to continue to assess the effect of analyzing CSS files and catch detailed information in CSS file during the analysis procedure. Second, the current system only analyzed the most inner layer of Web content with low the coherence between tables. To build more test cases and examine the precision of our system, the analysis of outer layers of Web content should also be considered. (3) Some functions may be broken while we immediately insert some other tags to adapt the page with Javascript functions and <DIV> tags. We will continue our exploration to solve the problem in our future research.

Acknowledgement

This work is supported by National Science Council, Taiwan under grants NSC98-2511-S-008-006-MY3 and NSC98-2511-S-008-007-MY3 and NSC 100-2511-S-146 -001.

References

- Berhe, G., Brunie, L., & Pierson, J. M. (2004). Modeling service-based multimedia content adaptation in pervasive computing. In *Proceedings of the First Conference on Computing Frontiers on Computing Frontiers* (pp. 60–69).
- Burzagli, L., Emiliani, P. L., & Gabbanini, F. (2009). Design for all in action: An example of analysis and implementation. *Expert System with Applications*, 36(2), 985–994.
- Chen, R. C. S., Yang, S. J. H., & Zhang, J. (2010). Enhancing the precision of content analysis in content adaptation using entropy-based fuzzy reasoning. *Expert Systems with Applications*, 37(8), 5706–5719.
- Chen, L. Q., Xie, X., Ma, W. Y., Zhang, H. J., Zhou, H. Q., & Feng, H. Q. (2002). *DRESS: A slicing tree based web representation for various display sizes*. Technical report MSR-TR-2002-126, Microsoft Research.
- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to algorithms* (3rd ed.). Cambridge, MA, USA: The MIT Press.
- He, J., Gao, T., Hao, W., Yen, I. L., & Bastani, F. (2007). A flexible content adaptation system using a rule-based approach. *IEEE Transactions on Knowledge and Data*

- Engineering*, 19(1), 127–140.
- Hua, Z., Xie, X., Liu, H., H. Lu, & Ma, W. Y. (2006). Design and performance studies of an adaptive scheme for serving dynamic web content in a mobile computing environment. *IEEE Transactions on Mobile Computing*, 5(12), 1650–1662.
- Huang, J. J. S., Yang, S. J. H., Huang, Y. M., & Hsiao, I. Y. T. (2010). Social learning networks: Build mobile learning networks based on collaborative services. *Educational Technology & Society*, 13(3), 78–92.
- Lemlouma, T., & Layaida, N. (2004). Context-aware adaptation for mobile devices, In *Proceedings of 2004 IEEE International Conference on Mobile Data Management* (pp. 106–111).
- Lee, Y. W., Chandranmenon, G., & Miller, S. C. (2003). *GAMMA: A content adaptation server for wireless multimedia applications*. Bell-Labs, Technical Report.
- Muffin. (n.d.). Muffin. Retrieved from <http://muffin.doit.org>.
- Reed, E. C., & Lim, J. S. (2002). Optimal multidimensional bit-rate control for video communications. *IEEE Trans. Image Processing*, 11(8), 873–885.
- Ramaswamy, L., Iyengar, A., Liu, L., & Douglis, F. (2005). Automatic fragment detection in dynamic web pages and its impact on caching. *IEEE Transactions on Knowledge and Data Engineering*, 17(6), 859–874.
- Su, A. Y. S., Yang, S. J. H., Hwang, W. Y., & Zhang, J. (2010). A Web 2.0-based collaborative annotation system for enhancing knowledge sharing in collaborative learning environments. *Computers & Education*, 55(2), 752–766.
- Tsai, Y. C., Cheng, C. H., & Chang, J. R. (2006). Entropy-based fuzzy tough classification approach for extracting classification rules. *Expert Systems with Applications*, 31(2), 436–443.
- Yang, S. J. H., Okamoto, T., & Tseng, S. S. (2008). Guest editorial: Context-aware ubiquitous learning. *Educational Technology and Society*, 11(2), 1–2.
- Yang, S. J. H. (2006). Context aware ubiquitous learning environments for peer-to-peer collaborative learning. *Educational Technology & Society*, 9(1), 188–201.
- Yang, S. J. H., Zhang, J., Tsai, S. T. C., & Huang, J. J. S. (2010). Applying semantic segment detection to web page presentation on the mobile Internet. *Journal of Information Science and Engineering*, 1(1), 96–129.
- Yang, S. J. H., Zhang, J., & Huang, A. F. M. (2008). Applying web page adaptation technique to the augmentation of mobile learning. *Journal of Research and Practice in Technology Enhanced Learning*, 3(3), 253–273.
- Yang, S. J. H., Zhang, J., Chen, R. C. S., & Shao, N. W. Y. (2007). A UOI-based content adaptation method for improving web content accessibility in the mobile internet. *ETRI Journal*, 29(6), 794–807.
- Yang, S. J. H., Zhang, J., & Chen, I. Y. L. (2008). A JESS enabled context elicitation system for providing context-aware web services. *Expert Systems with Applications*, 34(4), 2254–2266.
- Yang, S. J. H., Shao, N. W. Y. (2007). Enhancing pervasive web accessibility with rule-based adaptation strategy. *Expert Systems with Applications*, 32(4), 1154–1167.
- Yang, S. J. H., & Chen, I. Y. L. (2008). A social network-based system for supporting interactive collaboration in knowledge sharing over peer-to-peer network. *International Journal of Human-Computer Studies*, 66(1), 36–50.
- Zhang, J., Zhang, L. J., Quek, F., & Chung, J. Y. (2005). A service-oriented multimedia componentization model. *International Journal of Web Services Research (JWSR)*, 2(1), 54–76.