Mapping Visualization of Web Page Structure

Hyung-Suk Kim
Electrical Engineering Department
Stanford University
Stanford, CA 94305 USA
hskim08@stanford.edu

Juho Kim
Computer Science Department
Stanford University
Stanford, CA 94305 USA
juhokim@stanford.edu

ABSTRACT
We present two novel tree layouts for representing the hierarchical tree structure of a segmented web page, the spread tree layout and the cascaded tree layout. The proposed layouts show the visual components as well as the structure of the tree. The spread tree layout emphasizes the structure of tree, while the cascaded tree focuses on preserving the position of the segmented webpage elements. In addition, we provide an interface for comparing two trees using brushing and linking techniques to show the underlying mapping between the two trees. A controlled experiment showed how keeping the position data helps the user anticipate the relation between the two trees.

Categories and Subject Descriptors
H.5.2 [Information Interfaces]: User Interfaces.

Keywords
Visualization, web design, tree mapping, focus+context

1. INTRODUCTION
"Good artists copy, great artists steal." - Pablo Picasso

Artist and designers often start by copying prevalent work, then modifying it to create one’s own style. This proves to be a difficult task for web page design since one must plunge into the code to apply the design into one’s own work. Recent projects try to solve this problem by retargeting the content of a webpage to the layout of other web pages [3]. This enables rapid prototyping of design layouts enabling the user to explore a wider design space.

The novel approach to web page design presents new problems in visualization. The first challenge was to display the tree hierarchy of a web page and the segmented content in a single visualization. The content of the nodes were not a group of disjoint data but parts of a layout that had relative positions between each other. Another challenge was in comparing two distinct trees and visualizing the underlying mapping between the two.

In this paper we propose two novel tree layouts for visualizing tree hierarchies of visual data, the Spread Tree and the Cascade Tree. The former emphasizes displaying the hierarchical tree structure of the visual nodes, while the latter focuses on preserving the position data. We use brushing and linking techniques to visualize the underlying mapping between the two trees. Focus+context techniques and other interactive methods are used in the Cascaded Tree to support the visualization of the tree structure.

Figure 1: Pipeline for retargeting webpage content.
(a) Segment pages into perceptual trees. (b) Compute mapping between content and layout trees. (c) Use mapping to retarget content onto the new layout.

A controlled experiment compares the two interfaces along with the previously used interface and analyzes the ease of use in anticipating the result of the synthesized page.
This paper is organized as follows. Section 2 provides previous work on tree comparison, multiple tree visualization, and 2D drawings. Section 3 illustrates the proposed method for visualizing the connection between two web pages. Section 4 describes experiment setup and user study procedure. Then the following sections discuss results and propose directions for future work.

2. RELATED WORK

2.1 Motivation

This paper is largely based on the research on re-targeting of the web page content [3]. It started from the idea of building a web design tool that can help people use any web page as the design template. For novice designers, it is often cumbersome to design their own web pages. Tweaking lines of HTML and CSS codes is required in many cases, making the learning curve of web design steeper. The simple idea of this research was to enable design re-targeting easier, so that existing content can utilize a rich gallery of existing designs that match the visual layout of the desired page.

Synthesis of two web pages is a three step process. Re-targeting system first analyzes and builds a perceptual tree of each web page. This decomposition of web pages uses a number of heuristics that take into account both visual cues within the web page and the conventional DOM (Document Object Model) structure. Visual cues include position, size, font, color, and HTML tags. Once the internal, perceptual tree for both the content and the layout page is built, then the problem of morphing two pages becomes a matter of finding an optimal node mapping between two trees. Re-targeting tool uses a machine learning algorithm to define a mapping between two web pages. In the final step, mapping definition is applied to synthesize two pages.

The common user task in the overall process is to identify which content page segment, or node, is mapped to which part in the layout page. Graphical interface to visualize mappings between two pages is essential in both verifying the mapping process and predicting the final synthesized page. There has been no prior work especially targeted for mapping visualization, making the design of such interface challenging.

There are two major visualization challenges in the re-targeting process. One lies in presenting two web pages and two trees at once. There are four different structures to be displayed on one screen, visual web page and internal perceptual tree for both content and layout pages. Efficient use of space and combined or modal approach to show different representations is desired. The next one is in intuitively displaying the mapping between two pages. Interface could benefit from brushing and linking technique, while some other issues are involved as well.

We present two novel visualization approaches to this problem. Cascaded tree and Spread tree. Cascaded tree places the next depth, preserving the position of each segment meanwhile displaying the tree structure in 3D-like way.

They both display internal nodes of the tree in actual, visual web page segments. External and internal representation of the page are combined in one visualization in both cases.

2.2 Related work

Related work falls into roughly three categories: space efficient tree visualizations, layered data visualization, and tree comparison.

DOI Tree [2] is a tree that efficiently shows data nodes using focus+context techniques under space constraints. Unlike the DOI Tree which shows text data with constant height as nodes the problem we face is different since the nodes are segmented images of a webpage. We propose Spread Tree where the nodes resize according to the depth and the state of the node so that the leaf node is emphasized.

Cone Tree [7] is a 3D approach to tree visualization. The additional dimension creates extra space for data at the cost of ease of navigation. The extra complexity and confusion of 3D display seemed irrelevant to the problem at hand. We use 3D depth cues to mimic the front view of a cone tree.

Recent works on treemaps [4] leverage cascading to delineate the structure of the treemap. For Cascaded Tree, we use depth cues by shadowing to show the depth of the node. The layout is not a treemap, but a tree where nodes are positioned at the actual position in the web page. This allows the user to see each node in the context of the whole web page.

Ramos et al. [6] proposed two novel interfaces for manipulating layered 2D drawings. Tumbler decomposes the layers in to a tilted 3D view. Splatter creates an exploded view to show all obscured objects. We add a visibility slider for showing and hiding layers of the tree. We also use alpha compositing to show underlying layers when a node is click to support focus+context.

Tree comparison is an active area of research. TreeJuxtaposer [5] is a system designed for representing and comparing large trees of several hundreds of nodes e.g. phylogenetic trees. It uses highlighting in areas of interest with “guaranteed visibility” to help users compare trees. Our approach uses a mirrored juxtapose layout to compare Spread Trees by symmetry. The problem of comparing similar structures was also addressed by Drucker et al. [1] proposed a comparison framework for comparing multiple versions of presentation slides.

3. METHOD

3.1 Cascaded tree

Cascaded tree attempts to maintain the original web page layout, and shows the internal tree structure in a cascaded way. Child nodes are placed on top of the parent node, adding a depth cue with a shadow. There is no change in tree size unless the user zooms in or out; expanding and collapsing nodes occurs within a tree. Node positions are fixed relative to the frame of the root, which is the entire page.

Cascaded tree has a depth control slider, which sets the deepest level it displays on the screen. As the depth value increases, it is possible to see the process of tree decomposition as child nodes at the next depth level are stacked on the parent depth. Zooming and panning is supported to enable customized navigation of pages. ‘View visual nodes’ button
shows or hides the content of each node. Hiding visual nodes is useful when boundaries for nodes are not clear and if the user wants to see the overall layout of the page. On loading, both trees are resized to occupy half of the screen each. 'Fit to screen' button dynamically measures the current tree size and enables returning back to the original size.

3.2 Spread tree
Spread tree focuses on revealing the internal tree structure of the web page. This interface uses a conventional tree layout, and displays two trees in a juxtaposed, mirrored view. It spreads out child nodes from a parent node, showing how the parent has been segmented by the tree decomposition process. Mapped nodes are positioned next to each other, since the leaf nodes from each tree are placed at the center of the screen.

Similar to Cascaded tree, Spread tree also has a depth control slider. It allows expanding and collapsing the tree to the desired level of depth. Clicking on a node also expands or collapses the tree, but only for the subtrees of the node clicked. In addition, it also supports zooming, panning, and 'Fit to screen' button. More design issues and proposed approaches for each issue is presented next.

3.3 Design issues
3.3.1 Using Space Efficiently
Since we needed to show two trees at once we needed to use space efficiently. To fit multiple examples in a single page we had to look for a different way to visualize tree data. One way was to resize the nodes according to its depth and state. We implemented a resizing scheme for the Spread Tree, where the greater the depth the smaller the node. Also, if a node expanded it would reduce its size so the children nodes would be more visible.

The aforementioned approach visualized both the content and the tree structure of the page. But there were limitations. The parent nodes contained all the visual information of a child node, which was a redundancy of information. The position information of the nodes relative to the page was lost as a node was exploded into its sub-segments. We found that by collapsing the tree on to its parent nodes solved both problems.

3.3.2 Focus+Context
By collapsing the nodes on to the parent node the parent node became obscured. We solve this problem by using focus+context techniques. When a node is selected, it is pulled into the foreground and all other nodes become transparent to show the underlying structure. The fading out of other nodes highlights the selected node while maintaining the context.

3.3.3 Juxtaposed Mirrored View
3.3.4 Brushing and Linking

We use brushing and linking to show the underlying mapping in both Spread Tree and Cascaded Tree. On mouse over, mapped node on the other page is also selected. In the Cascaded tree, it might not be clear which node is selected since multiple nodes are likely to be packed in a limited region. To solve the problem, mouse click darkens all other parts of the tree, so that users can see the clicked part standing out visually. The same happens to the mapped node on the other tree.

4. EXPERIMENT

We conducted a comparative analysis of three different web page mapping visualizations: Cascaded tree, Spread tree, and HTML tree. The purpose of the experiment was to identify strength and weakness of each approach, so that we could list features that are essential in exploring different nodes and finding mappings within the visualization of web page structure.

HTML tree is the initial version of the mapping visualization designed in the prototyping process. Unlike other two approaches, this tree separates the visual segment of the web page with the internal tree representation. Brushing and linking is applied to the internal tree at the bottom, while the images at the top are only changed by the interaction at the bottom. We used this version as the conventional method for comparison purpose, since there were no existing methods available in this domain.

In the initial task design, we presented participants with page structure images for both the content and layout page on a paper. Nodes, or web page segments, in the content page were numbered and those in the layout page were not. The task was to use each visualization application to figure out and write down on the paper all the mappings that are present between two pages. After running two sessions of pilot study with Stanford graduate students, tasks were modified to find only five mappings per each visualization since it turned out that some tasks were taking too much time. We intentionally added nodes that had no mappings. This was to see how each interface dealt the issue of mapping absence, and to decide if additional interaction or feedback
Figure 10: The HTML tree was used in the experiment to see the effect of separation of internal and external representation of the page.

is necessary for nodes with no mappings. In the instruction, participants were informed to note absence of mapping if there were any.

Figure 11: The task given to the subject. 5 circles are nodes for which the subject was asked to find a mapping. The subject marked mapped nodes on the right with the corresponding node number on the left.

The final study design was 3x3 measure, composed of three different pairs of web pages and three visualization interfaces. Each participant had three tasks, each with different web page pair and visualization interface. In order to reduce learning effect and bias, we randomized both the tasks and the interfaces, so that the total of 36 combinations were generated. Due to the time constraints, only seven subjects participated, and they were all Stanford engineering students. Before starting the timed task to find five mappings, subjects had 2 to 3 minutes to get familiar with each interface and to ask any questions about using it. They were also told that their sessions will be timed and we will look at both task completion time and accuracy of the answers. Each session lasted between 30 minutes to 1 hour, including the post-study questionnaire. Survey questions are attached in the Appendix A.

Web pages used for the tasks were either self-designed or extracted from existing web pages. They all differed in layout and complexity, with 11 and 10 for the simplest pair, and 23 and 16, 23 and 30 for more complex ones. As seen from the difference in number of nodes, no pair had perfect mapping with its partner. The simple pair had almost perfect mapping, though. The page content ranged from pets and personal pages to hybrid cars and custom business card services.

The size of the window was 1250x770 for both Cascaded tree and Spread tree. HTML tree had variable page size, since it does not resize thumbnail images for the web page segments and it also explicitly lists mappings vertically. The initial focus for Spread tree and HTML tree was in the root, which is the entire page. Cascaded tree does not have an initial focus. In all cases, trees were fully expanded in the initial setting. The dependent variables were task completion time, accuracy of answers (either right or wrong for five answers), and ratings for impressions and functionality of each interface on a 5-point Likert scale.

5. RESULTS

Having only seven participants for the user study due to the time constraints, we decided not to run in-depth statistical analysis such as ANOVA. In this paper, we only present average time, accuracy of answers, and survey results to discover issues in visualization design. We had a total of 9 participants, and two of them were pilot testers. Among seven participants, there were three females and four males, and their average age was 25.3 years old. They were all Stanford graduate students in engineering.

5.1 Mapping identification time

For identifying mappings between pages, we hypothesized that there would not be significant time difference between Cascaded tree and Spread tree. It was certain that Cascaded tree provides more intuitive interface for mapping detection, but the result states that the difference in task completion time is much more than we expected. While participants only spent 78.1 seconds on Cascaded tree for visually connecting five mappings, they spent almost 10 minutes on Spread tree. It was even more time than HTML tree, although statistical analysis is missing. Our observation on people using Spread tree found that they spend a majority of time figuring out the location of the current node in the whole page. The same applied to the HTML tree. One subject noted that "Spread tree was inconvenient mostly because I could not easily map the node id to the actual part of the webpage."

5.2 Accuracy of mappings identified

![Figure 12: Time taken for task completion](image-url)
It turned out that participants showed high accuracy in their answers for all three interfaces. The difference in values was almost negligible. It might be the case that the task was too easy for most subjects, since all mappings were supposed to be intuitively present in the visualization. Mapping identification time seems to be a more crucial factor that is manageable with different design approaches. We also figured out from the data analysis that people made almost no errors in picking nodes with no mappings.

Figure 13: Accuracy of answers: the percentage of correct answers.

5.3 Impressions of each interface
Survey results seem to reflect the task completion time. In the 5-scale ratings on impressions of each interface, Cascaded tree marked 3.87, while HTML tree marked 2.68 and Spread tree 2.38. Many people felt annoying and confusing when using both HTML tree and Spread tree.

Figure 14: Impressions of each interface in 5-scale

5.4 Functionality and interactivity
We had almost the same result with impressions on functionality and interactivity. Cascaded tree marked 3.88, while HTML tree marked 2.85 and Spread tree 2.68. Although Spread tree scored high on ‘visual appeal’, it scored low mostly on space efficiency and locating nodes and mappings. One participant mentioned "I’m surprised that the third version (Spread tree) which looks much more interactive than the first one (HTML tree) is actually not that much an improvement in practicality.”. Another participant pointed out that “targets are moving all over the screen. Animation is slow (in Spread tree).” He also suggested using a touch pad interface for this visualization.

Figure 15: Functionality rating of each interface in 5-scale

6. DISCUSSION
6.1 Tree navigation
Cascaded tree resulted in faster task completion time and high score on interactivity. We see this in two reasons: First, nodes were not decoupled from the images (compared to HTML tree). Second, the tree was space efficient (compared to Spread tree). The effect of combining the internal tree and external page layout in a single view reduced user confusion. Instead of four structures, there were only two structures to look at in Spread and Cascaded tree. As Spread tree expands as depth increases, it invariably contains more blank space inside for node separation. Users found it difficult to navigate through multiple layers especially when the page structure was complex. Cascaded tree solved this problem by eliminating the need for lengthy navigation.

Cascaded tree has a shortcoming in selecting a desired node. Since a point on the tree might belong to multiple nodes, possibly all connected as parent-child, it is sometimes difficult to select a desired node. In order to cope with this problem, we first tried to add a small offset for each cascaded layer. But this was misleading in position, and made pages look more complicated. Instead, we added a depth control slider so that you can only see a part of the whole tree.

6.2 The effect of overview
The major difference in design of Cascaded tree and Spread tree is the existence of overview. Cascaded tree itself is an overview, since the tree never shrinks or expands in frame size. Therefore it is undoubtedly easy to identify the position of the current node. Spread tree, on the other hand, does not show the position of the current node relative to the root, the entire page. This consumed much time for most participants. Identification of mapping was almost as easy as Cascaded tree, but figuring out where this node is positioned at was another problem for Spread tree users. In the future design, we could add an overview, just like a small Cascaded tree layout, to Spread tree layout.
In terms of mapping identification, visual nodes alone might not be enough to navigate all mappings that are present on the current pair of pages. It could reduce confusion and search time if we add a list of mappings in a sidebar. Or we could add a control to highlight only the nodes with or without mappings. These fixes will give users richer overview of two different pages at once.

6.3 Bias in test design
The result favorable for Cascaded tree was not surprising, not only because Cascaded tree was especially designed for mapping identification purpose, while Spread tree focused more on revealing the internal tree structure. The task we assigned to participants was not the right one to test the primary design goal of Spread tree. Whether if we really need to know the perceptual tree structure is questionable, but it is clear that it can complement what Cascaded tree cannot provide in visualization.

Although we spent much time designing the study and setting up the web interface, we ended up having not enough participants to derive statistically significant data. Some results were clearly standing either for or against our design decisions, but more data would have led to more advanced statistical analysis.

6.4 Future work
The Cascaded Tree and the Spread Tree have very different design goals. Implementing a transition between the two modes can help users leverage both visualizations to gain a better understanding of the given tree data. Also in dealing with more complex pages, such transition would save time-taking navigation through irrelevant nodes.

The work presented in this paper focused more on the visualization itself. More work can be done on the interacting with the visualization. We have started this with methods of interaction for understanding the structure.

The web page retargeting tool is for exploring multiple design alternatives. There is a need to display multiple mappings and allow the user to navigate through a gallery of design alternatives.

Showing the resulting synthesized page will help the user view the results of the mapping. Together with an editing interface this will allow users to customize the design while receiving direct feedback.

During implementation we realized that the tree visualizations may be extended to the visualization of general hierarchical 2D layouts. We feel a need to search for a more general use for the visualizations for representing position aware hierarchical data.

7. CONCLUSIONS
In this paper we have presented two novel visualizations for representing the hierarchical tree structure of a segmented web page, the Cascaded Tree and the Spread Tree. We have implemented interaction techniques with the visualizations to help users better analyze visual connection between two web pages. We performed user tests with mapping identification tasks. Results from user tests show the efficiency of the Cascaded Tree in such tasks, possibly due to image and node coupling, and less navigation. In future work, we intend to extend this work and integrate it into the web page retargeting tool.

8. REFERENCES

APPENDIX
A. SURVEY QUESTIONNAIRE
After completing all tasks, subjects were asked to fill a questionnaire presented below in a format of web survey. Each question had 5-point Likert scale of 'Strongly disagree', 'Disagree', 'Undecided', 'Agree', and 'Strongly agree'.

1. How well do the following words describe YOUR IMPRESSIONS OF HTML TREE during your tasks?

   - It was helpful.
   - It was useful.
   - It was confusing.
   - It was convenient.
   - It was relevant to the task at hand.
   - It was inappropriate.
   - It was obtrusive.
2. How did you find FUNCTIONALITY and INTERACTIVITY of the HTML TREE?

- It was visually appealing.
- It was using space efficiently.
- It was interactive.
- It was easy to find WHERE a specific node is located on the tree.
- It was easy to SELECT a specific node from the tree.
- It was easy to explore and navigate SIMPLE trees.
- It was easy to explore and navigate COMPLEX trees.
- It was easy to see the overall structure of the tree.
- It was easy to find the mappings between two trees.
- I would use it again.