Using Space-Optimized Tree Visualization for Web Site-Mapping

Quang Vinh Nguyen
Faculty of Information Technology
University of Technology, Sydney
NSW 2007, Australia

Mao Lin Huang
Faculty of Information Technology
University of Technology, Sydney
NSW 2007, Australia

Abstract

This paper describes a new information visualization technique for web site mapping. It can be used especially for the display of very large web sites in a 2-dimensional space. Our strategy is to optimize the drawing of trees in a geometrical plane and maximize the utilization of display space by allowing more nodes and links to be displayed at a limit screen resolution. To enable the exploration of large web site-maps, we use a modified semantic zooming technique to view the detail of a particular part of the hierarchy at a time based on user's interest. Layout animation is also provided to preserve the mental map while the user is exploring the hierarchy by changing zoomed views.

Keywords: Web Site Mapping, Information Visualization, Graph Drawing, Space Optimized.

1. Introduction

As the amount of information on the Internet has grown rapidly for just a last few years, web sites have become much larger and more complex. Accompany with that, the lack of structure at both the inter-document and intra-document levels produces great challenges for information discovery. In order to overcome this problem, web designers apply a technique commonly known as *web site mapping* where user can gain an overview of the content and structure of a website to find particular information. There are many existing web site mapping techniques that can be summarized into three main categories: hierarchical mapping, graphical or mesh mapping and three-dimensional mapping.

- **Hierarchical mapping.** Hierarchical mapping [2] uses parent-child format to show the site structure. In this technique, the lower hierarchy displays further detail of the above hierarchies. Thank to its simplicity, naturalism and easy understanding, hierarchical mapping is used popularly in web sites of most of organization. *SiteTree [8]*, *MAPA [9]*, and *WebAnalyzer ring map [9]* are good examples of using hierarchical mapping for web site mapping.

- **Graphical or mesh mapping.** Graphical or Mesh Mapping [2] organizes the contents of a web site in a 2D space through nodes and edges. In this technique, a web page is represented as a node and relationship of pages is showed by edges. There are several good algorithms in graph visualization that are highly applicable in web site mapping such as *radial view [1, 4]*, *balloon view [1, 6]* and *hyperbolic browser [1, 5]* and *WebOFDAV [1, 10]*. However, only WebOFDAV and hyperbolic browser are highly applicable for visualizing large and very large hierarchies.

  WebOFDAV [1, 10] is a new approach for visualizing large sitemaps. This technique uses clustering approach to display a portion of information at a time. However, clustering property also prevents user from achieving entirely images of the sites (see Figure 1).

Hyperbolic browser is a new technique developed by Lamping and Rao [5]. This layout technique constructs trees in hyperbolic plane and then maps that structure into ordinary Euclidean plane. The algorithm produces nice tree visualization in side a disc and it is quite applicable for visualizing entirely large hierarchies. There are several implementations are available at Inxight-Xerox[1]. In spite of being a good layout technique, hyperbolic browser does not use entirely rectangular space to display information but only the area within the internal disc (see Figure 2). It also produces a large portion of the unused display space.

- **Three dimensional mapping.** Three-dimensional mapping [2] organizes the contents of a web site in a 3D space. The technique attempts to take advantages of 3 dimensional properties such as presenting better look and feel of the space, displaying more information in available space, etc. Cone trees [1, 6] is a good example of a graph visualization technique in 3D space (see Figure 3). However, three-dimensional mapping technique also requires much more expensive calculation compared to other techniques that is only applicable to powerful machines. As a result, this limits the popularity of this technique on the web.

1.2. Space-optimized tree visualization

Although many graphical or mess mapping visualization techniques have been developed for dealing with web site mapping, there are only few are good candidates considering the issue of optimizing display space in the layouts design.

This paper presents a new technique of graphical or mesh mapping for web site mapping. Similar to tree-maps [1, 3], we also use area division to define the layout of subtrees. This

[1] http://www.inxight.com/

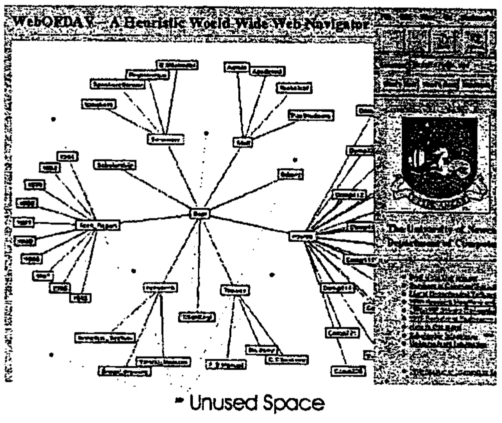

Figure 1. An example of WeboFDAV

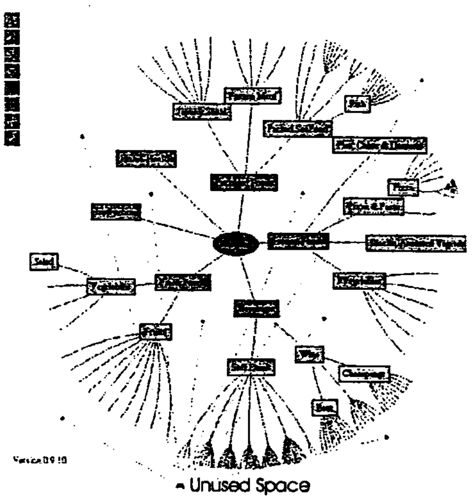

Figure 2. An example of Hyperbolic Browser

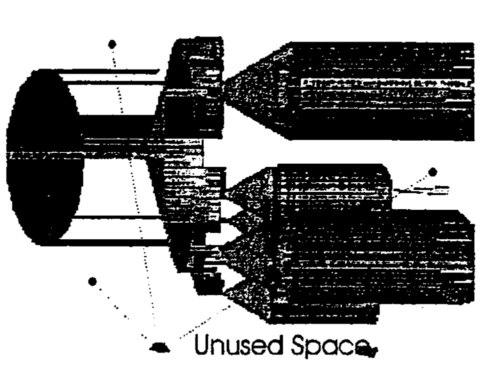

Figure 3. An example of Cone Trees

ensures the 100% efficiency of space utilization. On the other hand, we still use a node-link diagram (as graphical or mesh mapping techniques do) to show the hierarchy relationships of information. This improves dramatically the clarity of the data structure.

Our strategy is to optimize the drawing of trees in a geometrical plane and maximize the utilization of display space by allowing more nodes and links to be displayed at a limit screen resolution. In order words, we attempt to present entirely the web site structures at the screen resolution. We use either the *enclosure* to partition the display space into a collection of geometrical areas assigned to all nodes for the display of their subtrees, and *node-link diagrams* to show the relational structure. We take advantages of two approaches, the *enclosure* and the *connections*. To enable the exploration of large website hierarchies, we use a modified semantic zooming + filtering technique [1, 2, 7] to move a detailed view of a particular area of hierarchy around based on user's interest. Layout animation is also provided to preserve the mental map while the user is exploring the hierarchy by changing zoomed views. Figure 4 and 5 are respectively examples of our techniques for small and very large web site.

Section 2 describes technical specifications of our technique including the detail of space-optimized tree layout, navigation and interaction, and animation. Session 3 is our future work and conclusion.

2. Technical specification

We assumed that structure of a website is simplified as a rooted tree. We now review the terminologies that are used in our technique.

2.1. Terminology

A tree is a connected acyclic graph. A rooted tree consists of a tree T and a distinguished vertex r of T. The vertex r is called the root of T. In other words, T can be viewed as a directed acyclic graph with all edges oriented away from the root. If (μ, v) is a directed edge in T, we then say μ is the father of v and v is a child of μ. A leaf is a vertex with no children. If T contains a

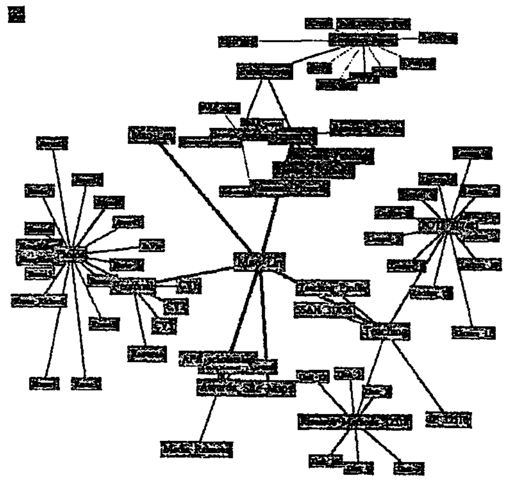

Figure 4. An example of Space-Optimized Tree Visualization for a small web site (approximately 80 pages)

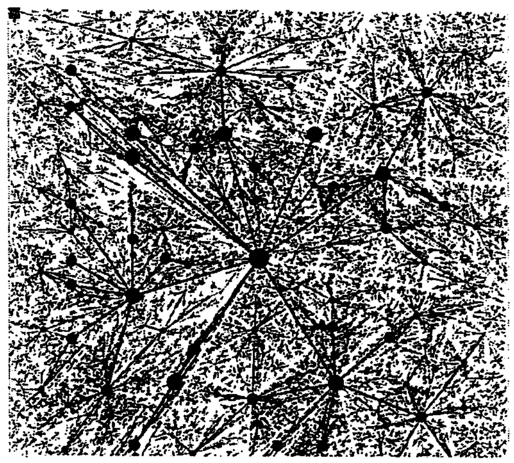

Figure 5. An example of Space-Optimized Tree Visualization for a very large web site (approximately 55000 pages)

vertex v, then the subtree $T(v)$ rooted at v is the subgraph induced by all vertices on paths originating from v. We also use a node to represent a vertex v with its displaying properties. This terminology is mainly mentioned in display session.

A geometrical local region of the vertex v in T is defined as a convex polygon $P(v)$, that contains the drawing of a subtree $T(v)$ and a directed edge (μ, v) where μ is the father of v. A wedge $wg(v)$, in our implementation, is defined by a vertex μ, a line l goes through that μ, and a clock-wise angle $\alpha(v)$; where μ is the father of v (see Figure 6). Thus we have $wg(v) = \{\mu, l, \alpha(v)\}$. A weight $w(v)$ is a integer value that associated with a vertex v in T.

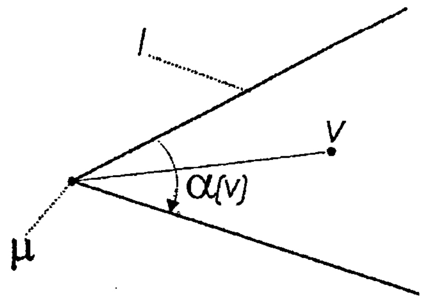

Figure 6. A wedge $wg(v)$

2.2. Geometrical layout

The geometrical drawing $D(T)$ of a tree T is responsible for the positioning of a set of vertexes $\{r, v_1, v_2, ..., v_n\}$ and the computation of local regions $P(v_1)$, $P(v_2)$, ..., $P(v_n)$ for the drawing of subtrees $T(v_1)$, $T(v_2)$, ..., $T(v_n)$ in a 2D plane.

Each vertex v_i is bounded by a polygon $P(v_i)$. The area of $P(v_i)$ is equal to the total areas of local regions of its children. The drawing of a subtree $T(v_i)$ is restricted within the area of $P(v_i)$. Figure 7 A drawing of subtree $T(v_i)$ rooted at v_i is calculated based on the properties of v_i and its boundary $P(v_i)$. We firstly define the local region $P(r)$ for root r as the entire display area, and the position of root r is at the center of $P(r)$. Then, all local regions $P(v_1)$, $P(v_2)$, ..., $P(v_n)$ are calculated recursively as described below:

Suppose that we want to calculate the region $P(v_i)$ of the subtree $T(v_i)$ rooted at v_i which has k children $\{v_i, v_{i+1}, ..., v_{i+k-1}\}$.

- We firstly calculate local regions $P(v_i)$, $P(v_{i+1})$, ..., $P(v_{i+k-1})$ for the children $\{v_i, v_{i+1}, ..., v_{i+k-1}\}$.
- We then calculate positions of $\{v_i, v_{i+1}, ..., v_{i+k-1}\}$ that are inside their local regions $P(v_i)$, $P(v_{i+1})$, ..., $P(v_{i+k-1})$.
- We repeat the above calculation to all subtrees from the top to the bottom of the tree hierarchy and it stops when all leaves of the tree are reached.
- We technically ignore the layout calculations for those subtrees when local regions of these subtrees are too small to be displayed by the current screen resolution.
- We eventually get $P(v_i) = P(v_i) \cup P(v_{i+1})$, ..., $\cup P(v_{i+k-1})$

Weight calculation. We assign a weight $w(v_i)$ to each vertex v_i for the calculation of the local region $P(v_i)$ in relatively to its father. A weight is a integer value that is associated with a vertex in T. We have a set of weights $\{w(v_1), w(v_2), ..., w(v_n)\}$ associated with the vertex set $\{v_1, v_2, ..., v_n\}$ in T. This set can be calculated recursively from leaves in the following rules:

- If v_i is a leaf, its weight is $w(v_i) = 1$
- If v_i has k children $\{v_i, v_{i+1}, ..., v_{i+k-1}\}$, its weight is

$$w(v_i) = 1 + C\sum_{j=0}^{k-1} w(v_{i+j})$$

Where C is a constant (0 < C < 1), and $w(v_{i+j})$ *is the weight assigned to* l^{th} *child of* v_i.

Constant C is a vector that determines the difference between the weight of a vertex and their children. In other words, the larger the C's value is, the bigger the difference of local regions between vertexes with more descendants and vertexes with fewer descendants. We apply a constant $C = 0.6$ in our prototype system.

Wedge calculation. We use wedges to find positions of vertexes and calculate the local regions for these vertexes.

Suppose that a vertex v_i has k children $\{v_i, v_{i+1}, ..., v_{i+k-1}\}$ located in a local region $P(v_i)$. We want to divide $P(v_i)$ into sub-regions $P(v_i)$, $P(v_{i+1})$, ...,

$P(v_{l+k-1})$ for the drawings of subtrees $T(v_l)$, $T(v_{l+1})$, ..., $T(v_{l+k-1})$.

A sub-region $P(v_{l+m})$ of the m^{th} child of v_l consists of a wedge $wg(v_{l+m})$ and one (or more) cutting edges (boundaries) cut by other higher level local regions in the drawing.

A wedge is defined as $wg(v_{l+m}) = \{v_l, l, \alpha(v_{l+m})\}$, where v_l is the father of v_{l+m} and l is a straightline going through v_l that determines two boundaries of $P(v_{l+m})$.

The angle $\alpha(v_{l+m})$ of the wedge $wg(v_{l+m})$ of the m^{th} child is calculated by the formula below:

$$\alpha(v_{l+m}) \quad = \quad A\frac{w(v_{l+m})}{\sum_{j=0}^{k} w(v_{l+j})}$$

Where A is a constant (A = 360°) and $w(v_{l+j})$ is the weight associated with vertex v_{l+j}.

We repeat the above calculation to get a set of wedges $\{wg(v_1), w(v_2), ..., w(v_n)\}$ associated with every vertex in tree *T*. These wedges determine the area division of local regions.

Figure 7 shows an example of dividing v_l's local region into 4 sub-regions for its children $\{v_{l+1}, v_{l+2}, v_{l+3}, v_{l+4}\}$. Local region $P(v_l)$ of v_l is a pentagon and $\{\alpha(v_{l+1}), \alpha(v_{l+2}), \alpha(v_{l+3}), \alpha(v_{l+4})\}$ are respectively the angles of wedges $\{wg(v_{l+1}), wg(v_{l+2}), wg(v_{l+3}), wg(v_{l+4})\}$. Figure 8 is an example of area division of a data set.

Vertex position. The position of a vertax v_{l+m} is computed after the calculation of local region $P(v_{l+m})$. We need to find out a point *P* in the boundary of $P(v_{l+m})$ that the straightline connecting *P* and the father vertex v_l divides $P(v_{l+m})$ into two areas of the same size. We then choose the position of the v_{l+m} in the midpoint of the line (see Figure 9).

2.2 Navigation and interaction

To be able to explore very large web sites with detailed viewing, we use a modified semantic zooming technique [1, 2, 7] to display the detail of a particular part of the hierarchy based on user's interest at a time. When selected by a mouse click, a node in visualization moves forward to the position of the root. All its

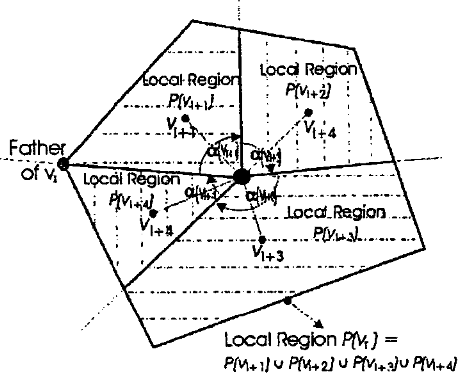

Figure 7. An example of dividing v_l's local region into children's local region of 4

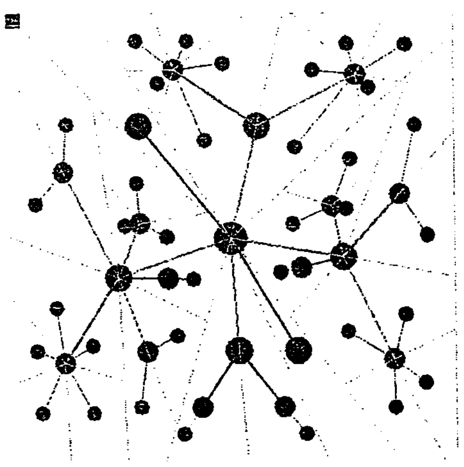

Figure 8. An example of area division of a data set – each node is bounded by a polygon

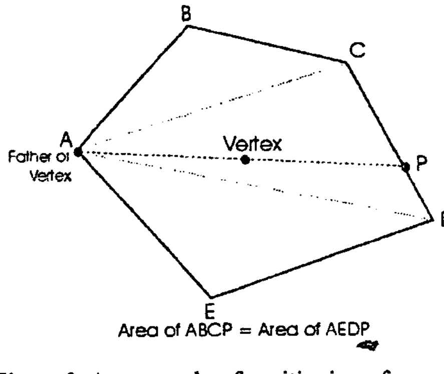

Figure 9. An example of positioning of a vertex

ancestors and siblings are ignored except a few ancestors are kept as a history path for the tracking of navigation. The local region of a selected node now expands to the entire display area. In other words, during the navigation, we only visualize the subtree of selected node and a history path (including the root). This viewing technique requires the recalculation of positions of all vertexes in a subtree at a time in corresponding to a mouse click (see Figure 10).

As same as the area division where a child's area is always smaller than its father, we also apply this rule to our viewing technique. The size of nodes and the width of edges we choose are proportional to their levels in the hierarchy. In order words, the closer to the root, the larger of nodes and the wider of edges are. This rule improves the clarity of the presentation of tree hierarchies.

As mentioned above, a history path is created for going backward purpose. The path displays orderly all direct-ancestors of the selected node. We assign different graphic properties to these ancestors to distinguish them from the normal nodes for the enhancement of clarity.

2.3 Animation

The animation is involved in order to preserve mental map when view changes. This is a real challenge because our technique focuses on visualizing very large hierarchies. The expensive computation of big data set (more than 10000 nodes) might disrupt smooth animation.

Therefore, we only apply animation to those nodes that are visible in the screen resolution. As a result, the number of nodes need to be involved in animation is reduced dramatically to from a few tens up to a few hundreds. In addition, during the animation period, edges are drawn and calculated with a minimal width. This also decreases remarkably the animation cost.

When a node is selected, all siblings and ancestors of this node are blurring away gradually while the selected node moves to its father's position. During that movement, all descendants of the selected node will smoothly move to and occupy the space of selected node's father. The animation stops when the selected node reaches the root position (the center of the display area).

3. Conclusion and future work

Space-optimized tree visualization is an effective and efficient web site mapping technique for visualizing large and very large web sites. This layout algorithm can draw the entire structure of web sites at the limit screen resolution. The property of attempting to display maximal numbers of web pages helps user to easily understand the entire structure of the site. This web site mapping technique also allows the viewer to navigate and zoom any particular session of the large web site structure in cooperating with our modified semantic zooming technique.

We are currently investigating on new animation algorithms to find out the one that is most suitable for the space-optimized tree visualization. Another improvement over the original layout is also being implemented. This algorithm uses area approach to define a node boundary based on its weight instead of wedge angle. We also investigate on new viewing techniques that can keep the global view of the entire tree structures while a detailed view of a part of the tree can be displayed during the navigation.

We have presented our space-optimized tree technique for visualizing web site mapping large hierarchies. Although our work is not completed yet, we believe that this technique will become a valuable web site mapping tool for visualizing large web sites.

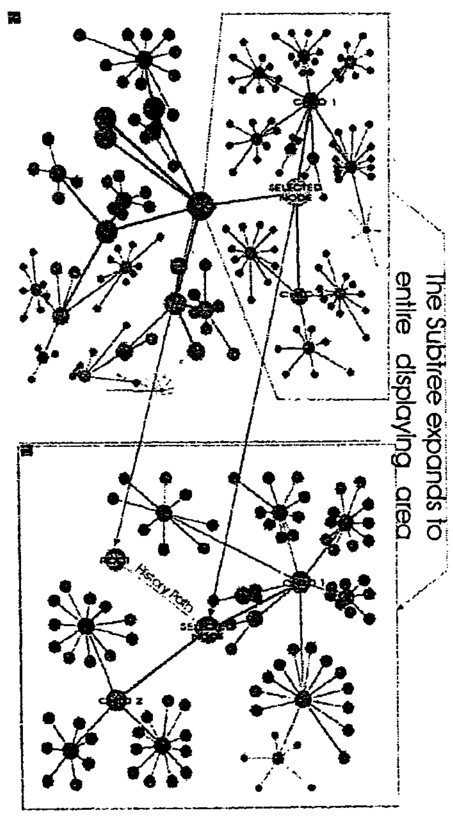

Figure 10. An example when a node is selected

References:

[1] Herman, G. Melançon, M.S. Marshall. Graph Visualization in Information Visualization: a Survey. In: IEEE Transactions on Visualization and Computer Graphics, pp. 24-44, 2000.

[2] Mao L. Huang. Information Visualization in Web Site Mapping: a Survey. In Proceeding of SCI2000/ISAS2000 Conference, pp. 390-395, 2000.

[3] Johnson, Brian, and Shneiderman, Ben. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In Proceedings of the 1991 IEEE Visualization, IEEE, Piscataway, NJ, pp. 284-291, 1991.

[4]. P. Eades. Drawing Free Trees. Bulleting of the Institute fro Combinatorics and its Applications, pp. 10-36, 1992.

[5] J. Lamping and R. Rao. The Hyperbolic Browser: A Focus + Context Technique for Visualizing Large Hierarchies. Journal of Visual Languages and Computing, vol. 7, no. 1, pp. 33-55, 1995.

[6] G. G. Robertson, J. D. Mackinlay, and S. K. Card. Cone Trees: Animated 3D Visualizations of Hierarchical Information. Human Factors in Computing Systems, CHI '91 Conference Proceedings, ACM Press, pp. 189-194, 1991.

[7] G. W. Furnas, X. Zhang. MuSE: A Multi-Scale Editor. In Proceeding of the UIST'98 Symposium, ACE Press, 1998.

[8] C. I. Pilgrim and Y. Leung. Design WWW Site Map Systems. IEEE Internet Computing, vol 31, pp. 32-35, Jan-Feb 1999.

[9] M. Walter. MAPA Product Overview – Full strength mapping for the World Wide Webb. Seybold Report on Internet Publishing, January 1997.

[10] M. L. Huang, P. Eades and R. F. Cohen. WebOFDAV – Navigating and visualizing the Web on-line with animated context swapping. In Proceeding of the 7th International World Wide Web Conference, Brisbane, Australia, April 1998.

[11] Mark Feldman. Basic 2D Math. The Win95 Game Programmer's Encyclopedia. Copyright 1997,http://www.geocities.com/siliconvalley/2151/math2d.html.