# Providing Interactive Site Maps for Web Navigation

Wei Lai
Department of Mathematics and Computing
University of Southern Queensland
Toowoomba, QLD 4350, Australia

Jiro Tanaka
Institute of Information Sciences and Electronics
University of Tsukuba
Tsukuba, Ibaraki 305-8573, Japan

**Abstract** *One-dimensional linear navigation method is commonly used in current Web browsers. That is, a Web page may include some highlighted text strings that are linked to other Web pages. This kind of navigation from one page to another page lacks the whole structure overview required for relationships between Web pages. This paper presents how to use tree structure diagrams, a 2-dimentional view, as site maps for Web navigation. According to the user's focus in the process of Web navigation, a tree diagram is formed as a site map for each Web page dynamically. The user can interact with the tree-based sitemap to navigate to another site map or looking at a detailed view (i.e. a Web page) for a node in the tree diagram. The user can also adjust the diagram for getting a preferred tree diagram layout if the user does not like the default layout form.*

*Keywords:* web navigation, tree structure diagrams, site maps, interaction

## 1. Introduction

The amount of information now available through the World Wide Web (WWW) has grown explosively. An increasing number of tools are available to assist the user to manage and access information on the WWW, such as Netscape and Internet Explorer. The key requirement for a Web browser is to show the details for the user's focused information and facilitate navigation within the whole information hyperspace. However, it is impossible to display this huge and growing hyperspace for the user to get its whole structure in helping navigation. The navigation approach used in most Web browsers is from one page to another page. Although current Web browsers can give bookmarks and history lists, which are at most a linear list, they cannot provide relationships between the URLs.

Some researchers have proposed "site mapping" methods [1, 2] to attempt to find an effective way of constructing a

structured geometrical map for one Web site (i.e. a local map). However, this can only guide the user through a very limited region of cyberspace, and does not help users in their overall journey through cyberspace.

Other researchers use a graph for WWW navigation. The whole cyberspace of the WWW is regarded as a Web graph [3, 4]. This approach more emphasizes on navigation and it does not pay more attention to getting better local view for site mapping. The graph layout in this approach is not hierarchical structure for hyperlinks and can not guarantee no node overlapping. This makes a site-mapping view sometimes unclear to the user.

This paper introduces our method of using hierarchical tree structures to represent subsets of the Web graph for Web navigation. A subset is formed based on the user's interaction. The subset should be changed dynamically and should follow the user's focus in navigation. This change from one subset to another should preserve the user's mental map [5, 6] for WWW navigation. We call these subset *local site maps*.

The main feature of our web graph interface is that we use tree diagrams as site maps for Web navigation, as the hyper documents' URLs embedded in a Web page's HTML file is a tree based hierarchical structure. We also focus on the local site map design. The user can interact with the tree diagram for navigating to another tree diagram or looking at a detailed view (i.e. a Web page) for a node in the tree diagram. The user can also adjust the diagram for getting a preferred tree diagram layout if the user does not like the default layout form.

In the following section, we show some examples of our site map displays. Section 3 introduces some design issues about our system. The techniques for tree diagram layouts are described in Section 4. We conclude and discuss our approach in Section 5.

## 2. Examples of Site Map Displays

The best way to show our site map displays is to use some examples. Figure 1 shows an on-line site map which is converted by parsing and filtering the Web page: www.sci.usq.edu.au. We provide three kinds of modes for the user's interaction: *Editing*, *Navigation*, *ShowPage*, which can be set up by clicking the left button, middle button, right button on the mouse respectively.

In the Editing mode, the user can edit and adjust tree diagram layout. For example, the user can change a site map from a horizontal tree (i.e. *h-tree*) in Figure 1 to a tip-over tree shown in Figure 2.

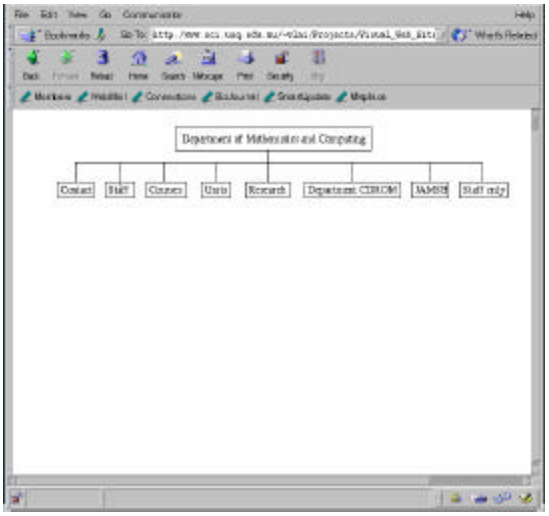

Figure 1: A tree-based site map

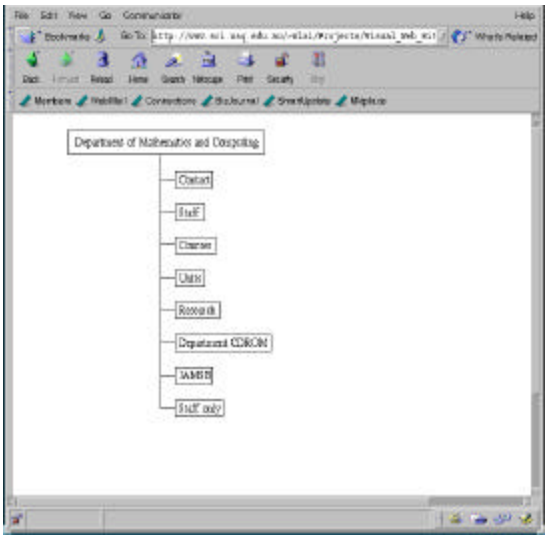

Figure 2: Another tree layout form for the same site map

In the Navigation model, the user can navigate the Web space by selecting the nodes in the tree diagram. For example, after the user clicks the node '*Courses'* and the node *'Bachelor Courses'* in Figure 2, another tree-based sit map (see the window on the left in Figure 3) is shown up based on the user's current focus node 'Bachelor Courses'.

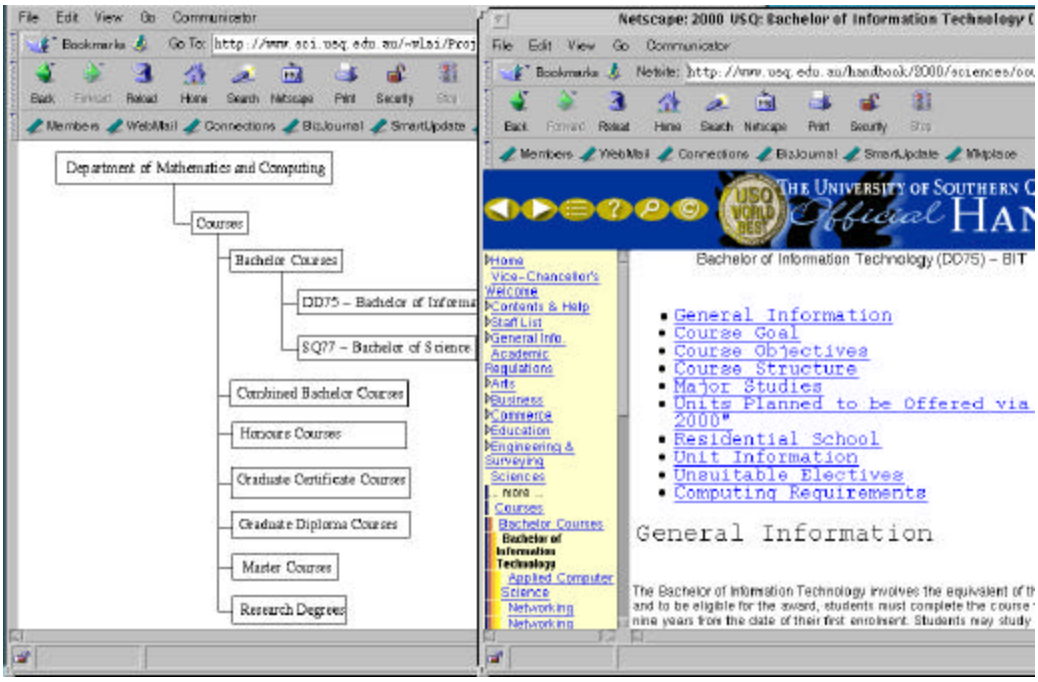


Figure 3: Navigation to another site map

Our system can support the display of a detailed Web page corresponding to a node in a tree-based site map (after the ShowPage mode is set up). For example, after the user selects the node '*DD75-Bachelor of Information Technology*' in the window on the left in Figure 3 in the ShowPage mode, the window on the right in Figure 3 is shown up to display the detailed view for this node.

## 3. System Design

Our system design integrates techniques on graphical user interfaces, automatic graph layout, distributed computing, Internet and Web programming, computer networks and data communications. This section introduces some design issues.

The architecture for our system includes two major components: a Web graph user interface component and a dialog component for the communication between the Web graph user interface and the WWW.

The Web graph user interface component displays Web sub-graphs (i.e. a tree diagram) and provides some editing functions for the user to adjust a graph layout, to navigate the Web graph by choosing a focused node, and so on. It provides three kinds of modes for the user's interaction with the tree-based site map as mentioned in Section 2.

The dialog component supports the construction of Web sub-graphs by communicating with Web sites over the Internet. It can quickly search the entire neighborhood of the focused node to form a Web sub-graph.

The dialog component has a Web site parser and an information filter. The Web site parser analyses the HTML file of the Web site corresponding to the focused node and extracts the hyper documents' URLs linking to this Web site to form nodes and edges in the Web graph. To reduce the complexity of Web graph, the information filter removes unnecessary information (edges and nodes) generated by the parser and only retains the essential part of the real Web graph. This simplified visualization is a tree structure. Then the Web graph user interface component maintains the user's orientation for Web exploration and it also reduces the cognitive effort required to recognize the change of views. This is done by connecting successive displays of the sub-set of the Web graph and by smoothly swapping the displays via animation.

We use the Java programming language as the major software development tool for the implementation of our system. A prototype of the Web graph user interface for WWW navigation has been developed. The techniques for drawing tree diagrams in our system are introduced in the following section.

## 4. Tree Diagram Layout

Each on-line Web page's HTML source file is downloaded and then is converted into a tree structure by our system. The reason to use tree diagrams is that the tree structure can be used to show hypertext hierarchical relations. For a tree diagram layout, we should consider the user's requirements. Different users may need different tree structure diagrams. In Figure 4, there are five layout forms of a tree. Which one is the "nice" layout? This should be decided by the user.

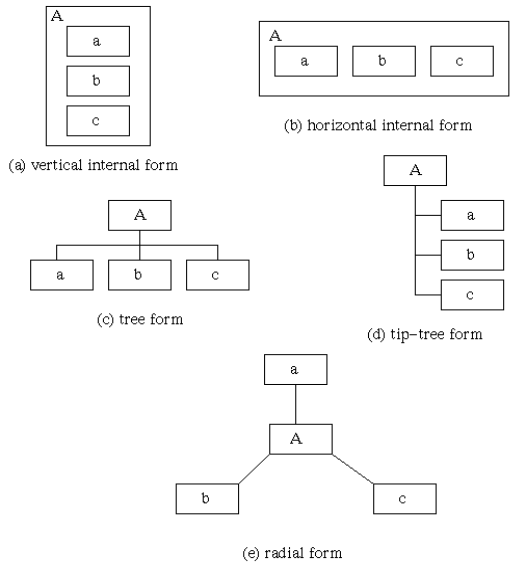


Figure 4: Five layout forms of a tree structure

Our system integrates some facilities for drawing various tree diagrams. Our Web

graph interface allows the user to select a tree diagram layout that he/she wants.

The tree structure used in our system is designed for supporting interactive layout operations. For a node's graphical appearance, each node in the tree structure is assigned some geometrical attributes such as its size, central position, shape and so on. Another feature is that each node is also assigned an attribute *layout form*. A layout form for a node specifies a layout function operating on this node's sub-tree. For example, Figure 4 shows five layout forms of a node *A* and its layout forms are *inclusive-v*, *inclusive-h*, *h-tree*, *tip-tree* and *radial* respectively.

We can get a tree diagram layout by defining layout forms for nodes. The default layout form for the root node is *h-tree*. The other nodes' default layout form is to inherit the layout form from its parent node in an object-oriented fashion. Figure 1 shows a tree diagram layout by default layout forms for it nodes.

A different layout can be obtained by changing the definition of layout forms for some nodes. For example, by changing the definition of layout forms for the node '*Department of Mathematics and Computer*' in Figure 1 to *tip-tree*, we can have another layout shown in Figure 2.

Our system provides a *layout toolbox* for the user to operate just on the graphical appearance of a node. The layout toolbox includes *S-toolbox* and *L-toolbox*. The S-toolbox contains a selection of shape forms. The L-toolbox contains a selection of layout forms. The user can choose any one of the shape (or layout) forms in the layout toolbox and select a node in the tree diagram to change that node's graphical appearance dynamically.

The layout toolbox gives the user a choice of aesthetics and drawing conventions. Figure 4 is an example of five different layout forms available with the layout toolbox for the same graph. The layout toolbox allows the user to experiment interactively in order to achieve a satisfactory result.

In our system, we draw a tree in *h-tree* layout form as shown in Figure 4 (c) by putting every sub-tree within a rectangle (see Figure 5 (a)). We use the same approach to tip-over tree drawing (see Figure 5 (b)). Although this approach cannot produce a tree drawing as compact as those tree drawing algorithms [7, 8, 9], it is more flexible for changing a sub-tree from one form to another. For example, we can change the sub-tree in Figure 5 (a) to Figure 5 (b). If a sub-tree overlaps another sub-tree, the layout adjustment technique, the force-scan algorithm [6], is applied.

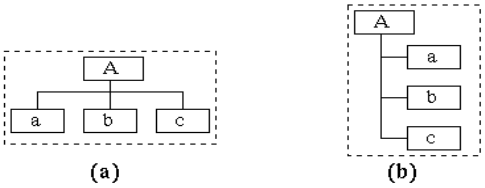


Figure 5: Drawing a sub tree in a rectangle

The method for radial tree drawings (for example, see Figure 6 (e)) is introduced in [10]. We have developed a method for drawing inclusion trees (as shown in Figure 4 (a) and (b) ). The method focuses on the creation of various inclusion trees by changing the gaps between a node and its inside nodes, and the gap amongst inside nodes.

The *inside_gaps* defines a set of gaps for a node. It includes: *g_gap, l_gap, r_gap, u_gap* and *b_gap*. The parameter *g_gap* defines the minimal gap between two inside nodes. The parameters *l_gap, r_gap, u_gap* and *b_gap* represent the minimal gaps between the inside nodes and left side, right side, up side, and bottom side of the node, respectively. Figure 6 illustrates these gaps.

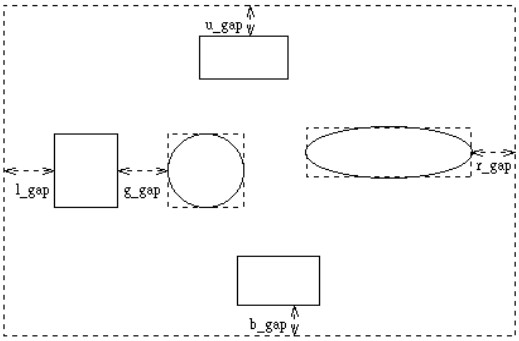


Figure 6: The inside gaps

For example, the diagram in Figure 7 (a) is the result of applying a layout function for drawing a tree in *inclusive-h* layout form. That is to draw the nodes in a horizontal arrangement with the definition of that: all gaps are a default value (a fixed size). Figure 7 (b) is the result after changing all gaps (except *u_gap*) to be zero for the *inclusive-h* tree in Figure 7 (a).

A dialogue box (see Figure 8) is provided for the user to define the gaps for an inclusion tree diagram (General means *g_gap*, Left *l_gap*, Right *r_gap*, Top *u_gap*, and Bottom *b_gap*).

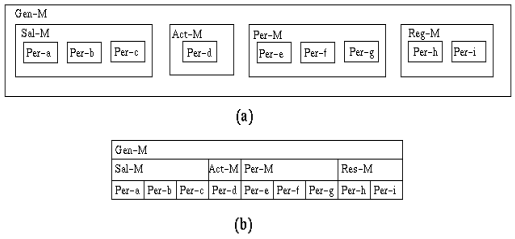


Figure 7: A tree based form drawing

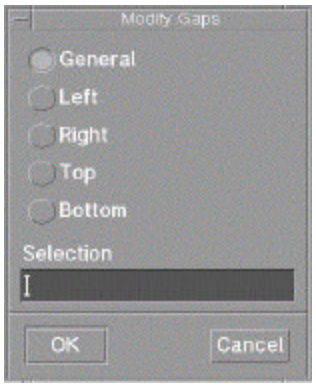


Figure 8: The dialogue box for defining gaps

## 5. Conclusion

This paper introduces a new Web navigation system that provides visible subsets of the Web graph using tree diagrams. Our system creates an automatic tree diagram layout that does not overlap or exceed the viewing area. A tree diagram can be displayed automatically for each Web page during Web navigation. Also, the user is allowed to interact with the tree diagram to get a preferred layout if the user does not like the default layout form.

Recent feedback from users is that they would like to combine our Web graph interface and a current Web browser (such as Netscape) together for Web navigation. It seems that they do not like to use the Web graph interface alone for navigation.

Further work is needed to improve the Web graph user interface. In particular, we will continue to investigate layout techniques that will enhance potential usability of the system. We feel that a new Web browser should integrate the features in current Web browsers (such as Netscape and Internet Explorer) and our tree-based site mapping approach.

## References

[1] Pilgrim, C. and Leung, Y. 1996, 'Applying Bifocal Displays to Enhance WWW Navigation', *Proceedings of the Second Australian World Wide Web Conference*.

[2] Maarek Y. S. and Shaul, I. Z. B. 1997, 'WebCutter: A System for Dynamic and Tailorable site mapping', *Proceedings of the Sixth International World Wide Web Conference*, pp. 713-722.

[3] Huang, M., Lai, W. and Zhang, Y. 1999, 'Mapping and Browsing the Web in a 2D Space', *Proceedings of the 10th International Workshop on Database and Expert Systems Applications*, pp. 248-252, IEEE Computer Society Press, at Florence, Italy, September.

[4] Lai, W., Huang, M., Zhang, Y. and Toleman, M. 1999, 'Web Graph Displays by Defining Visible and Invisible Subsets', *Proceedings of the Fifth*

*Australian World Wide Web Conference*, pp. 207-218.

[5] Eades, P., Lai, W., Misue, K. and Sugiyama, K. 1991, 'Preserving the Mental Map of a Diagram', *Proceedings of COMPUGRAPHICS 91*, pp. 34-43.

[6] Misue, K., Eades, P., Lai, W. and Sugiyama, K. 1995, 'Layout Adjustment and the Mental Map', Journal of Visual Languages and Computing, No. 6, pp. 183- 210.

[7] Reingold, E. and Tilford, T. 1981, 'Tidier Drawings of Trees', *IEEE Transactions on Software Engineering*, Vol. 7, No. 2, pp. 223-228.

[8] Moen, S. 1990, 'Drawing Dynamic Trees', *IEEE Software*, pp 21-28, July.

[9] Bliesch, S. 1993, 'Aesthetic Layout of Generalized Trees', *Software Practice and Experience*, Vol. 23, No. 8, pp. 817-827.

[10] Eades, P. 1991, 'Drawing Free Trees', Technical Report IIAS-RR-91-17E, Fujitsu Limited, Japan.