Automatic Generation of Diagrammatic Web Site Maps

Robert Inder, Jonathan Kilgour, John Lee
Human Communication Research Centre,
University of Edinburgh,
Email: robert@cogsci.ed.ac.uk, jonathan@cogsci.ed.ac.uk, john@cogsci.ed.ac.uk.

Keywords: Navigation, World Wide Web, Site Map.

ABSTRACT

In this paper we outline a site map application on the World Wide Web, whose aim is to improve navigation and orientation within the web site of the Human Communication Research Centre (HCRC). The map takes the form of a graphical fisheye view of the web site with the current node as the single focus. We discuss the semi-automatic process whereby the site map description is produced, describe how it is turned into a relevant map, and discuss reaction and usage patterns. We outline how the approach can be extended in three future directions: affording the user greater control of the graph drawing variables; covering more of the site, and indeed the web in the large; and using the user's interaction history to constrain the site map display further.

1:INTRODUCTION

At the Human Communication Research Centre (HCRC), we have implemented a graphical site map: Figure 1 shows the site map for the HCRC home page. This paper discusses the motivation and background for developing this map, the basis of the implementation, and where it might lead. Our long-term objective is to support users throughout extended interactions with large information structures, such as databases and hypertext documents.

Our starting point is the generally appreciated need to assist Web users with navigation. The problems of disorientation and becoming lost are all too familiar to many of us. Often, the only solution is to back up perhaps quite a long way to a known point (such as a site home page), which in typical browsers may lose the trace of some useful parts of one's session [2] [11]. It is clearly necessary for the user somehow to have more information, more clearly presented, about the nature of the information space being traversed and what kind of paths may be available through it.

We think graphical representations have much to offer in this context. Although we recognise there are fundamental limitations to their representational power [16], these seem mainly to apply to information of types not likely to be required in the present context — e.g. highly abstract or quantified information. For the presentation of information which is fairly determinate at the structural level, graphics should be relatively effective (cf. [18]). We are therefore interested in creating some kind of graphical presentation to help users access information that can be visualised as a spatial map. Of course, as soon as we propose graphical navigational aides, we have to address the problem of screen space: real estate in terms of pixels is precious and we have to choose carefully what to present and how to represent it. Once we have decided on a graphical representation, it is clear that there is more than one kind of information that we may be able to visualise in this way.

One of these information types is the user's interaction history. In circumstances where enough is known about the nature of the material being browsed, it's possible to exploit a notion of *discourse structure*, whereby the user's conversation with the system is seen to consist of various distinct elements, concerning e.g. particular topics and sub-topics. Inder and Oberlander [10] invoke this kind of discourse theory to describe hypertext navigation as a dialogue with the system. They propose that hypertext systems can aid the coherence of such a discourse by providing explicit links back to previous discourse elements (context resumption) and by inhibiting certain types of link when the user is digressing from the main dialogue. This approach makes no essential appeal to graphics - indeed the prototype system produces a dynamic reorganisation of the most immediately available navigational cues. But still it can be used to organise a visualisation of the user's session, showing the nature of the path shown and providing a range of direct options for moving back to various points on it, as demonstrated in [9]. However, the implemented system described relies on the fact that the underlying tool (*Emacs Info*) has typed links, and therein is its weakness: link typing mechanisms, when available, are not always used properly by authors, and are not available at all for some major information structures we wish to consider, in particular the WWW.

This paper, then, concerns the next step: to investigate dynamic graphical structures via the Web. It is our hope to combine elements of the discourse theory approach with elements of (semi-) automatically derived structure maps. We are still at an exploratory stage in this process, looking at options to satisfy our long term aim of providing a tool to run on the users machine and operate across all web sites. Working with the Web poses many problems: it is very large, and changing too rapidly for any kind of complete analysis to be practical; it is very diverse, in both content

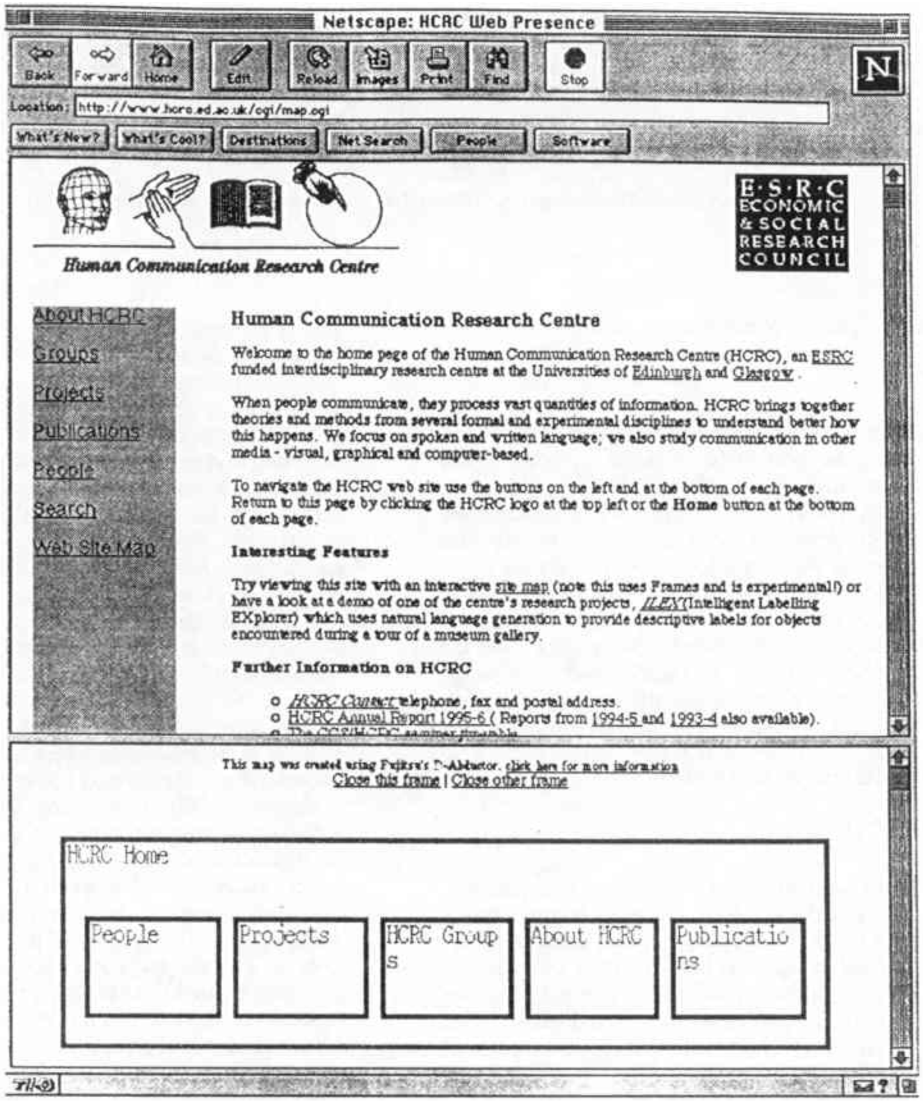

Figure 1: HCRC site map showing content frame (top) and map frame.

and form of information; and it is ill-structured. So although in the long term we aim to produce tools to help individual users navigate within and between sites anywhere on the Web, our goals for this work are more modest. We are experimenting with graphical presentation of information relating to a known portion of the Web, and in particular exploring what graphical presentations do, and do not, work within a web environment. Our initial exploration, therefore, is a site-specific tool. Anywhere in the area that is covered by the tool, you are offered a "web site map" button. Clicking it gives you a diagrammatic representation of the structure of the site, focused on the node you were viewing. Clicking on any node within the diagram refocuses the map on the node in question, and also displays the corresponding page. Figure 2 shows a close up of a map frame, focused on the *publications* node.

Our site map can be seen at *http://www.hcrc.ed.ac.uk/* by clicking the *Web Site Map* button. The site map tool is built out of two existing pieces of software: Fujitsu's D-Abductor [17] , for drawing graphs, and **gd** [6], a simple graphics library for C. The next section of this paper describes D-Abductor and how it was customised for our needs using the **gd** library. In Section 3, we go on to describe the mechanism whereby the HCRC web site is automatically derived, and how that helps us to automatically produce a graph description. We then go on to discuss the issues in producing an appropriate map from the graph description, and, on the basis of our live trial on the HCRC web site, we make observations on the problems and prospects of our approach. We provide some short analysis of access logs to the HCRC site and this, along with user comments on the site map, and readings in related areas, informs the section on future work.

2: LAYING OUT THE MAP

The kernel of the task facing us is going from a description of the structure of the site (in terms of the pages that exist

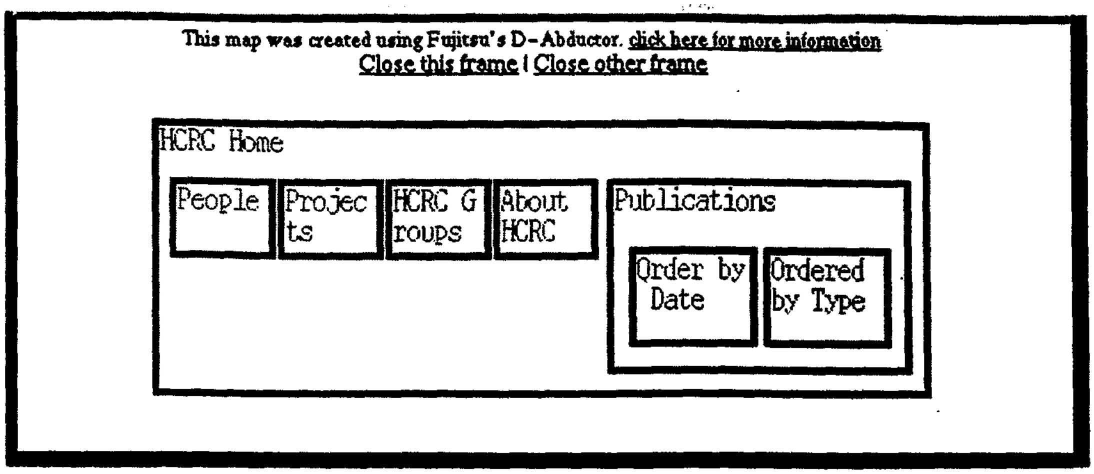

Figure 2: A close-up of the map frame with *Publications* as the focus node

and the relationships between them) to an acceptable arrangement of lines and text. Even for a person, producing a clear and aesthetically pleasing diagram takes skill and time. But since our interest is in dynamically modifying the map to suit the user, we're interested in doing it automatically. This is a problem of graph layout: there is a large literature on it [1], and it is hard.
The work described here uses the graph layout engine within Fujitsu's D-Abductor system [17] to generate diagrammatic maps.

D-Abductor was developed as a "thinking support tool", intended to help people thinking about complex issues using the KJ Method [12] which is well established and widely used, particularly in Japan. The later stages of the KJ method require users to present the structure of their ideas in diagrammatic form, and to support this process, D-Abductor contains an advanced graph layout system. It is based on the Sugiyama algorithm [17], and is able to represent both *inclusion* relations (depicted by nesting the nodes concerned) and *adjacency* relations (depicted by lines linking the nodes). It has a "focus" mechanism, which uses a variation of a fish-eye technique to make parts of the graph relatively more prominent, and can hide parts of the graph altogether [13].

Graphs created in D-Abductor can be saved (or exchanged with concurrent invocations of the tool) in D-Abductor's own "Simple" language. Simple files specify the nodes that exist, some of their properties, and the links between them, but *not* their locations. Node positions are determined by the graph layout algorithm each time the diagram is presented, and will reflect the user's current choice of layout parameters.

D-Abductor is available by ftp from *ftp://SunSITE.sut.ac.jp* as */pub/asia-info/japanese-src/packages/abd2.23.tar.gz*

D-Abductor itself is an interactive X-windows application for UNIX systems which supports incremental graph creation: the diagram is continually updated, with all changes being smoothly animated. However, for the current work, we modified D-Abductor to work with the "gd" library [6], a public domain library of subroutines which interpret graphical operations like those used in X-Windows in order to produce images in GIF format. We thus created a batch program which takes a graph description in Simple, together with a specification of a focus node and other formatting options, and produces a diagram that can be displayed in most web browsers. For the current application, the graph description will encode the structure of the (fragment of) the Web site to be mapped. It is discussed further in Section 3.

In addition to the diagram itself, the modified version of D-Abductor also produces a file of data which includes the coordinates assigned to the various objects in the image. This is then further processed into the form browsers need in order to use the image as an HTML "imagemap".

3: PRODUCING A GRAPH DESCRIPTION

This section describes the process of producing a description of (some part of) a Web site as a graph. We need to describe a given collection of web pages as a collection of objects (nodes), with links between them that reflect the underlying structure of the site.

One option is to use the original D-Abductor as a WYSIWYG graph editor, interactively creating a graph by hand and then saving it to a file in Simple format. In the process, a human designer can make sure that the intended structure of the site is clear and can optimise the graph, for instance, by excluding "unimportant" links that would otherwise clutter the presentation,

However, this is a difficult and time-consuming task, particularly where many pages are involved, and will typically have to be repeated every time the structure of the site changes. We are therefore keen to explore the automatic production of these graph descriptions.

HCRC's web site is in fact not authored as a collection of HTML documents. Instead, it is generated, using data from a number of central sources, and from user editable files within a known file structure. Instead of going directly from these sources to HTML, we go through an intermediate formalisation in SGML.

SGML (Standard Generalised Markup Language [7]) is an extremely powerful meta-language in which grammars can be written: We have written our own SGML grammar, or Document Type Definition (DTD), which describes in general terms the web site of an institution. We capture the important structural features of such an entity, and associate each feature with a set of attributes. For example, any institution will have associated people, who may fulfil one of a number of possible roles within the institution, and who may have telephone number, personal web space, be associated with a particular subgroup etc. In this way, our SGML captures the structure of the HCRC web site in a much more meaningful manner than HTML: we have all the advantages that generalised markup can give us: maintainability, transparency of information, and flexibility [4]. We can enforce a particular look and feel over the whole site (or meaningful subsets of it) without resorting to extensive editing and we can maintain links automatically, meaning that all references to any object will be updated when the central source is changed.

Given that we have the data for the HCRC site stored as SGML, it is a simple matter to produce a hierarchical tree structure in the *Simple Language* which can be used by D-Abductor (as described in Section 2). We also produce a mapping between the Simple Language IDs and the HTML file names produced by the translation.

For other Web sites, and for the Web in the large, producing a *Simple Language* description for D-Abductor to use is not so easy. Hand-produced descriptions are not only time consuming to produce but are susceptible to changes in site structure and content over time. The Web is not only large and uncharted but in many cases it is dynamic: the acts of navigation and search can cause new pages to be generated. If we are to expand our methodology to the Web as a whole we must address this important question. Clearly there is a certain amount one can do by analysing URLs. Pirolli et al [15] use URLs (at least partially) to derive usable structure from the Web. They combine this with other methods including content analysis. We have briefly experimented with the use of URL structure along with user interaction histories to form a graphical history tool We touch on this in Section 8.

We have found it natural to produce trees rather than more general graphs for our site descriptions. This is certainly in part due to the hierarchical division of the site imposed by the SGML site structure. However, trees are a familiar and intuitive construct for people to use and also lead to less cluttered graphical presentations. But trees are not the way of Hypertext: a hypertext network is generally a highly interconnected group of nodes, often with each node referring to a home node, a help node, or any number of other nodes which could be situated higher or lower on a hierarchy. Some applications which analyse web sites directly, first generate the graph of hypertext links in the site, then flatten that into a tree structure with repeating nodes [19]. As we mentioned in section 2, D-Abductor has the power to represent inclusion and adjacency edges, where the former are represented by boxes inside other boxes, and the latter are represented by arrows between boxes. We plan to experiment with the use of the second of these constructs to represent links across the hypertext structure.

4: TURNING THE DESCRIPTION INTO A MAP

Once we have a description of (part of) a web site, in terms of relations between nodes, and a mechanism for rendering that structure as a diagram, we are still not in a position to draw an effective map. The constraints on available screen space mean that presenting the entire structure is unlikely to be useful, and we must still decide which part of this information should be displayed at any point, and which should be hidden or omitted.

The initial approach taken for the HCRC web site was to produce a map of only the upper levels of the site. The contents of the current node, and of those nodes between it an the "top" of the site, is shown, but the structure within other sub-nodes is hidden. This avoids an explosion in the number of nodes to be displayed at the bottom level of the tree, although at the cost of limiting the ability to jump "across" the hierarchical structure.

However, there will inevitably still be cases where there are a large groups of pages which cannot necessarily be separated into any logical sub-groups.

For the HCRC web site, we have taken advantage of the rich representation of the site we have in SGML. So rather than having graphical nodes for each individual person inside the **People** box, we instead generate pages listing the staff grouped in a number of different ways, such as by Working Group membership, or by role. This small number of groupings are then presented graphically, in place of the individual members of staff. In fact, we see no advantage to having a graphic for each individual person, since a flat list of names is displayed more effectively in textual form as part of the site content.

However, this solution relies on having a richly structured representation of the site contents, together with its semantic interpretation. But these things are not generally available. Often, there is no representation richer than the HTML, and while there have been attempts to classify HTML pages, and impose certain semantic attributes on them [15] it is still an open question how effective this can be.

There are various ways of ameliorating the problem by increasing the number of nodes that can be represented in a given amount of screen space. For example, MAPA [8] opts for a 3D projection in which multiple boxes are partially obscured, the name of each box can be viewed by moving the mouse over it. A roughly similar effect can be obtained by using JavaScript to display the full names of nodes in the browser status line when the cursor moves over the node. Alternatively, AMIT [19] can display all subnodes in a scrolling list type display, but allows subsequent user interactions to collapse contiguous list elements within one level.

5: ISSUES ARISING

There are several issues which have arisen during the construction of the system, and which we have explored through variations on the basic behaviour described above.

One option is that of caching maps and hence reducing download times. On the live version of the site map, maps are named with unique URLs, so that caching is impossible on the server and the client side: a new map is generated for every click. We also experimented with server-side caching where names of maps come from their *Simple Language* IDs, and they are left on the server for the next request for a map with this ID. This has the advantages that we avoid the processing time to generate a map on every click (approximately 1 second on a Sun SparcStation IPX), and that client browsers can cache. However, our long term goal is to allow a great deal of user-customisation of the maps, and even the integration of information from the user's session, which will be unique for every user, and importantly will not be the same when a user re-visits the same page. Thus we lean toward the non-caching approach.

A second issue is synchronisation of maps and content pages. The current state of affairs on the live site is that synchronisation is *one-way* in that a map click causes a new page to be loaded in the content frame and a new map to be drawn, but navigation in the content frame does not cause the map to change accordingly. In a further experiment, we generate maps and content pages which include JavaScript calls to ensure *two-way* synchronisation. Now, navigation in the content frame as well as clicking in the map frame updates the map. The disadvantage here is the overhead in generating the new map when traversing a link in the content frame.

Thirdly, we have explored issues of customisation. It is possible to use special form pages to allow users to specify preferences for map display, colour and orientation. These can then be encoded as HTTP cookies [5], and affect the appearance of maps on a per-user basis. Our software allows user to select the orientation of the map (vertical or horizontal), one-way or two-way synchronisation as described above, and whether maps appear in a frame alongside the main document, or in a separate pop-up window. It also allows users to select colours for three different kinds of node: leaf, non-leaf and focus. These facilities are currently being evaluated for inclusion within the live site, and we discuss further plans for extending user-customisation in the Future Work section.

We have also conducted some related experiments in graphical web navigation. In the first experiment, we have explored the possibility of moving beyond static maps, using maps that "zoom in" to the node that the user's mouse is over, thus displaying more detail. In principle, such a mechanism would allow the user to browse the whole tree without recourse to the server, just moving the mouse, exposing more information as required. This could obviously be achieved, with as much sophistication as required, by downloading some kind of Java map-presentation applet to the user's browser, and then sending it the entire site map graph in some suitable format. This approach is closely related to that taken by MAPA (see below), which allows the user to zoom into a node by drawing a new diagram. Pilgrim and Leung [14] also use a Java presentation for their bifocal site maps.

Unfortunately, a major drawback to such techniques is that they can easily be rendered unusable by firewalls. However, we have managed to achieve reasonable, though not ideal, results, at least for a limited-degree (depth) of zooming, by pre-generating a number of alternative maps on the server, and using browser (JavaScript) functionality to display them.

In a second experiment, we have produced an extension to W3-mode for the Emacs browser called GW3-mode. W3-mode is a fully featured web browser which runs in an Emacs window. GW3-mode extends this to provide a graphical display of the session. D-Abductor is again used for the display, and we use both inclusion edges and adjacency edges. URLs are analysed to check whether the current focus has been served from the same site as the previous document. If so, this node is contained within the same parent box in the graphical representation. If not (and if the new site has not previously been visited) a new box is created for the new site which contains the current page as its only child. Link traversals, both within and between sites are represented by adjacency edges. We have produced some interesting results with these approaches, and plan to expand on these ideas in future implementations.

6: USAGE LOGS AND USER REACTION

While we did not attempt any formal experiments on the use of the site map, we have received comments from local users on the site map and we have done a small amount of analysis of Web access logs.

Most of the user's comments were supportive of the idea of providing a graphical map.

Perhaps not surprisingly, their criticisms were dominated by the aesthetics of the diagram. Unfortunately, this is something over which Abductor gives us only limited control. It is important not to under-estimate the difficulty of the graph layout task if adjacency links are to be allowed, and we currently need to work within the constraints of the tool we have.

Other comments indicated the value of screen space. It is clear that we must give users as much control as possible over screen organisation and also keep the maps themselves to a minimum size.

Some analysis of the server logs, which record which machines have fetched what information from HCRC's server, has been carried out. During the first three months of availability, from July to September 1997, about 20% of all accesses to HCRC's main "corporate" pages (i.e. the area of the web site covered by the site map) were to map pages. In other words, a significant minority of visitors to the web site have been interested in the map. More than half of the 159 people (machines) who accessed the site map at all accessed it more than once, and 45 machines, both inside and outside HCRC, have accessed map information more than 3 times, suggesting that these users, at least, found it useful.

Of course speed of access is also a factor in usability. The major delay seems to be download time rather than time taken for the server to draw the map, so all we can do is keep the diagrams as small as possible: none of the diagrams currently served exceeds 10K, which we deem to be reasonable.

7: RELATED WORK

AMIT (Animated Multiscale Interactive Tree-viewer) [19] is a tree based viewer developed at Bellcore. The tree is essentially a textual list where text can be different colours and sizes. One important contribution of this work is the integration of searching with navigation: inside an example domain which is a set of geographically disparate but thematically related web pages, search results appear as foci of a *table of contents* style textual list. The list can be manipulated in various ways by the user to display the sections he or she is interested in. In these ways, it can be made clear where abouts in a large structure, particular search results lie, and this, it is argued will make it easier to select and discard the appropriate pages.

Pirolli et al. [15] consider how to extract descriptions of structure from web sites. From some given page, links are recursively extracted that lead within the same site, and the abstract network of links between the pages is induced. Various techniques are also suggested for inducing certain kinds of typing on the links, e.g. by classifying the pages at their ends. Though not a great deal is said in [15] about visualisation, it is clear that such developments would potentially at least partly automate the complex task of preparing a site for mapping.

MAPA from Dynamic Diagrams [8] is a commercial graphical site map service, using Java for implementation. As we have already noted, the presentation of site maps in MAPA is via a 3D projection which effectively hides information until the user uncovers it by moving and clicking the mouse. Site map descriptions are strict hierarchies (trees are considerably easier to draw than graphs) where the initial description is produced manually. The system uses a link database to handle addition and deletion of web pages automatically, but structural changes are always manual. From a usability point of view, we feel the site map is rendered less useful than it could have been because the Java display does not link well (or flexibly) with the HTML display. An option for synchronisation between the map and content windows would improve the interaction greatly.

Pilgrim and Leung [14] also use a Java implementation, but come at the same problem from a slightly different angle. Using a bifocal display approach, the overall structure of the site is always visible, but with one area of the display being *in focus*, meaning that it is shown at a greater level of detail. The non-focused areas of the map are distorted so that links between nodes in the focus area and those outside remain intact. This approach has a somewhat similar end result to the fisheye approach used in our map presentations.

Bollen and Heylighen [3] have reported experiments in using link traversal to strengthen associative links between hypertext pages, in the manner of associative networks. They propose that this could possibly be extended to impose link structure on the web, for example by highlighting hypertext links that are commonly used.

8: CONCLUSION AND FUTURE WORK

It is clear that screen space is a major factor in providing site maps, or indeed any kind of support for navigation. There are tight limits on how much can be shown on a workstation screen: the computer "desktop" is as cramped as the folding tables in aircraft, and this small amount of space must be shared between the actual information itself and whatever tools are provided to help access it. For this reason, we allow users to choose to have maps presented either in a frame below the page itself, or in a separate window. The latter allows users who are comfortable with using multiple windows to use the full flexibility of their window manager to control how things are arranged on the screen. However, it certainly does not solve the problem of map size, and we must still consider how to maximise the amount of information in a display while minimising the screen real-estate used.

One approach is to exploit the transient nature of images on a computer screen and to either animate or dynamically augment the display in response to user activity. In effect, we can use the temporal dimension to expand the effective size of the screen. This is at the heart of the approach taken in MAPA: the 3D projection in its diagrams makes it natural for the representations of the various nodes to "overlap", thus reducing the amount of screen space used for each node. This forces almost all the information to be omitted from the diagram, but it remains usable because animation, in the form of diagram modification and "pop up" text, is then used to present the "hidden" information in response to the user's mouse movements.

However, while making diagrams dynamic can effectively create somewhat more space for us, on its own it will never give us enough. The key element is using the structure and abstraction in the information presented to determine what information should be hidden or removed. For example, most web sites have a structure - typically, a tree (here we are not referring to the structure of files and directories, or the URL structure, but rather we are considering the virtual link structure, as induced in [14]). There may also be other structure that would be useful, e.g. thematically related content, though this may be difficult to discover automatically. Such notions of structure are what make it possible to find suitable levels of abstraction for pruning the diagram by collapsing whole areas into a single black box.

Whatever techniques we consider for increasing the amount of information we can present, we must still consider carefully what information we should present. In the introduction, we suggested that the user's "itinerary" during a browsing session, or indeed across a number of sessions, is an important navigational asset which could conveniently be visualised as a graphical map. Yet our experiments in web history mapping (noted in Section 5) confirm that many of the issues in mapping history are closely related to those of mapping site structure, particularly in relation to focusing and pruning. Thus it will still be necessary to judge which part of a large interaction history should be displayed at any time. More importantly, it is not clear how history information can best be integrated with structural information (derived from links or by content analysis methods etc.) to produce a presentation which is both meaningful and useful. There are many unresolved issues in combining two informational structures that will sometimes align well, but at other times be orthogonal, particularly given that there will always be uncertainty in recognising the structures themselves.

In this context, the HCRC site map has let us experiment with ways of presenting small amounts of known information within a Web context. Usage patterns suggest that the resulting diagrams are of some use, and there is scope for scaling up the techniques used to generate them to handle larger bodies of information. However, the real challenge will come when we start to combine site structure with browsing history.

ACKNOWLEDGEMENTS

The work described in this paper was carried out in the Human Communication Research Centre (HCRC), in part under a Research Collaboration agreement with Fujitsu Laboratories Ltd. HCRC is funded by the UK Economic and Social Research Council (ESRC). The first author was supported by an award under the NEDO Industrial Technology Fellowship Program, and the assistance of Dr. Toshikazu Kato and the members of the Intelligent Systems Division at ETL in Tsukuba is also gratefully acknowledged.

REFERENCES

[1] G. D. Battista, P. Eades, R. Tammasia, and I. G. Tollis. 1993. "Algorithms for annotated graph drawing: an annotated bibliography," Dept of Computer Science, Brown University, Providence, Technical Report.

[2] M. Bieber and J. Wan. 1994. "Backtracking in a Multiple-window Hypertext Environment," presented at ACM European Conference on Hypermedia Technology.

[3] J. Bollen and F. Heylighen. 1996. "Algorithms for the self-organisation of distributed multi-user networks.," presented at Cybernetics and Systems 96.

[4] M. Bryan. 1988. *SGML: An Authors Guide to the Standard Generalized Markup Language*: Addison Wesley, 1988.

[5] Cookies : Preliminary specification 1997 "http://home.netscape.com/newsref/std/cookie_spec.html" .

[6] GD 1.2: A graphics library for fast GIF creation . 1996. "http://www.boutell.com/gd".

[7] C. F. Goldfarb. 1994. *The SGML Handbook*. Oxford: Clarendon Press.

[8] MAPA: Java implemented Web Site Mapper and site analysis service, 1997. Dynamic Diagrams. Inc. "http://www.dynamicDiagrams.com".

[9] R. Inder. 1997. "Graphical presentation of structure in interaction with hypertext," Human Communication Research Centre. Technical Report HCRC/TR-94.

[10] R. Inder and J. Oberlander. 1995. "Using Discourse to aid Hypertext Navigation," presented at HCI International. Yokohama.

[11] S. Jones and A. Cockburn. 1996. "A Study of Navigational Support Provided by Two World Wide Web Browsing Applications," presented at Hypertext, Washington DC.

[12] J. Kawakita. 1975. *The KJ Method - a Scientific Approach to Problem Solving*. Kawakita Research Institute, Tokyo.

[13] K. Misue and K. Sugiyama. 1991. "Multi-viewpoint perspective display methods: Formation and application to compound graphs.," *Advances in Human Factors/Ergonomics*, vol. 18A, pp. 834 - 838.

[14] C. Pilgrim and Y. Leung. 1996. "Applying bifocal displays to enhance WWW Navigation," presented at The Second Australian World Wide Web Conference, Surfers Paradise, Australia.

[15] P. Pirolli, J. Pitkow, and R. Rao. 1996. "Silk from a Sow's Ear: Extracting Usable Sources from the Web.," presented at CHI 96: Human Factors in Computing Systems.

[16] K. Stenning and R. Inder. 1995. "Applying semantic concepts to the media assignment problem in multi-media communication," in *Diagrammatic Reasoning: Cognitive and Computational Perspectives*, J. Glasgow, N. H. Narayanan, and B. Chandrasekaran, Eds. Cambridge, MA.: MIT Press, pp. 303 - 338.

[17] K. Sugiyama and K. Misue. 1991. "Visualization of Structural information: Automatic Drawing of Compound Digraphs," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21.

[18] E. R. Tufte. 1991. *Envisioning Information.*, 2nd edn. ed. Cheshire, Connecticut: Graphics Press.

[19] K. Wittenburg and E. Sigman. 1997. "Visual Focusing and Transition Techniques in a Treeviewer for Web Information Access," presented at IEEE VL 97, Capri, Italy.