HISTREE - A HIERARCHICAL BACK MENU

Daniel Orner and I. Scott MacKenzie
Department of Computer Science and Engineering, York University
4700 Keele St., Toronto, ON Canada M3J 1P3
{orner, mack}@cs.yorku.ca

ABSTRACT

The majority of web pages visited by users are actually revisits. The most common method to revisit web pages is the Back button; however, due to its stack-based nature, pages frequently "pop off" the stack, resulting in those pages being inaccessible from the Back menu. We propose a solution, Histree, which utilizes low screen real estate, preserves the skills learned from use of the regular Back menu, and provides a full navigational history, allowing the user to revisit any page from the current session.

KEYWORDS

Browser history, back menu, hierarchical menus.

1. INTRODUCTION

Studies (Tauscher, 1997; Cockburn, 2001) have shown that the majority of web pages visited by users have been recently seen by them. Revisitation is therefore an important part of web browsing. There are several revisitation tools packaged with every browser: the Back and Forward buttons (and their attendant menus), the Go menu, and the History functionality. However, the majority of revisitation tasks are accomplished via the Back button. (Catledge, 1995; Cockburn, 2001) Studies (Cockburn, 2002) have shown that the Back menu is much more efficient than the Back button when used competently.

The main problem with both the Back button and its attached menu is its stack-based behavior. (Cockburn, 1996) To illustrate this problem, let us take a user who starts at page A and visits page B. He then clicks the Back button to return to page A, then visits page C. This navigation path can be shown as A → B ⇦ A → C. The contents of the Back menu will now be {A,C}. Page B has been pruned out of the menu entirely.

2. PREVIOUS WORK

Many solutions to this problem have been proposed. The major issue with nearly all of these solutions (Ayers, 1995; Doemel, 1995; Frecon, 1998; Hightower, 1998; Robertson, 1998; Gandhi, 2000; Milic-Frayling, 2003; Jhaveri, 2004) is that they require a separate visible window. This either significantly reduces the screen real estate, or requires the user to continually pop up a new window and close it. Both effects are a considerable detriment to the web browsing activity.

Some work exists on replacing the current stack-based behavior of the Back button with a recency-list implementation (possibly with some variations), which lists the most recently visited pages (Greenberg, 1999; Greenberg, 2000; Cockburn, 2002; Cockburn, 2003). However, several problems with this behavior have surfaced in these same studies. For example, hub-and-spoke behavior (where the user visits several pages from the same "hub" page) often causes problems for recency lists. Either the hub drifts further and further away from the last child page (by placing intervening children between the hub and the current page) or the intervening children begin to drift further and further away from the hub (by placing them *before* the hub). In addition, the studies mentioned concentrate on the Back button rather than the Back menu.

Two proposed solutions which integrate the navigational tool with the Back menu rather than opening a new window are WebView (Cockburn, 1999; Cockburn, 2003) and SmartBack (Milic-Frayling, 2004). The most recent study on WebView involves a version which takes up significant screen real estate via thumbnails, and is also not fully integrated into the browser, requiring the use of two separate methods of navigation. (Although thumbnails have been shown to be useful (Kaasten, 2002), we have found no study yet done on whether thumbnails *in addition* to titles/URLs make a marked improvement in accuracy or efficiency.) SmartBack is an add-on to the Back menu which marks important pages such as hubs for easy revisitation, but attempts to "predict" important pages are necessarily imperfect.

An intriguing study (Wen, 2003) suggests that providing users with navigational context – i.e., where they went and how they got there – could be a useful tool, perhaps replacing thumbnails in giving more modes of recognition. If the user is presented with titles or URLs which are not useful (as is so often the case in web browsing), simply providing cues pointing to the user's previous navigations could allow him to more easily find the desired page. Such information could be provided without taking up precious screen real estate, and could be easily integrated into the browser.

3. HISTREE

We proposed a solution, called Histree, to the problems in the Back menu. The solution involves replacing the static menu with a dynamic, hierarchical menu. In particular, when a page is "popped off" the stack, instead of disappearing, it is placed in a submenu which is emitted from the page it was clicked from. This submenu may have more children and submenus (indicating further paths). In fact, by going to the bottom of the Back menu, the user can recreate his entire browsing session!

An example of the behavior of Histree is shown in the following visitation track:

1 → 2 → 3 → 4 → 5 ⇦ 4 ⇦ 3 ⇦ 2 → 6 → 7 → 8

The Back menu as created by Histree would be as follows (the user is viewing Page 8):

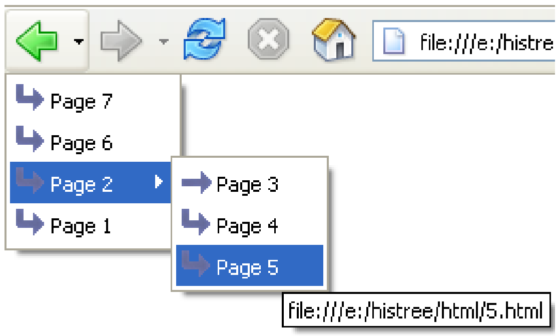

Figure 1. An example of Histree's generated Back menu

If the user had clicked to three different pages from page 2 rather than just to page 3, the remaining two pages would also be placed in a submenu. Histree can also collapse submenus if they would be too small, displaying siblings along with children. For example, the following navigational path:

1 → 2 → 3 →4 ⇦ 3 ⇦2 → 5 ⇦ 2 → 6 → 7

could be represented as follows (the user is viewing Page 7):

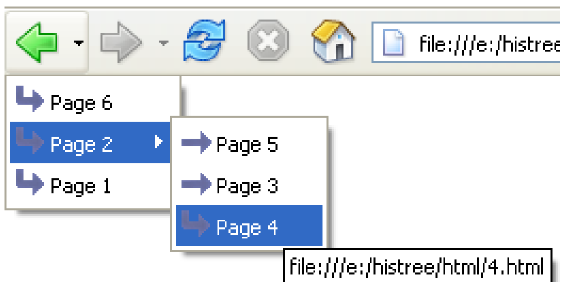

Figure 2. An example of siblings joining children in a submenu

2.1 FamilyTree Algorithm

Although Histree is useful in that it stores the navigation tree exactly, and provides the user a relatively stable viewpoint, its main disadvantage is that it remains a tree; if a user visits a page three times from different source pages, that page is listed three times in the tree. There is no way to see *how I got here* or *where I went from here* in a universal sense. To address this, a second algorithm, FamilyTree, was proposed.

FamilyTree is a graph rather than a tree, where each page (identified by the URL, frameset data, and HTTP POST data used to reach it) is listed only once. Each page has a list of parents and children. Other than this key point, FamilyTree acts much the same as Histree. It replaces the Back menu with a hierarchical menu; however, rather than directly paralleling the user's navigational path, it simply provides a list of the parents and children of the selected page. Each parent and child can then also be expanded to find *their* parents and children.

Although FamilyTree allows the user to more easily see the places he has been and will be from the current location, it may cause confusion due to his constantly changing viewpoint within the graph. In addition, the removal of duplicates also removes the advantage of presenting a complete navigational history like the Histree algorithm does.

We are convinced that offering the user navigational data will improve efficiency and ease of use in finding previous pages. These two algorithms provide two different ways of doing so while utilizing little screen real estate, and building on already-learned skills. At this point, however, we are unsure of which algorithm is more useful, or if some amalgamation of the two is better. A full user study is planned to address this point.

4. IMPLEMENTATION

Histree was implemented as an extension for Mozilla Firefox. Extensions are code segments written in Javascript and XUL (eXtensible User-Interface Language), an XML-based method of defining and overlaying the user interface of the browser. The advantage of using extensions is that any Firefox user can install only the extensions they want, without having to use a separate browser installation. Most previous studies were forced to overlay or replace the application, meaning that users were not using the browser they were accustomed to. This methodology allows users to keep all the preferences and behavior they are used to, while simply adding the Histree functionality. Extensions are largely cross-platform compatible, like Firefox itself.

The "Histree" extension actually consists of a choice of the two different algorithms mentioned above, via a provided preference panel. Users can also choose the length of the page title; titles which are too long have their middle characters truncated. Kaasten (2002) showed that this is an efficient way to keep page recognition high. If the Histree algorithm is used, users can also decide whether they want to have a separate tree per tab, or to have one "universal" tree. Under the universal algorithm, opening a link in a new tab or window will add a link from the previous page to the new one, just as if the user had clicked in the same tab.

The extension keeps track of the pages visited by using a generic graph structure, where each node on the graph contains a reference to the JavaScript history object (for immediate loading) and references to all parent and child nodes of the current node. In the Histree algorithm, this graph becomes a tree, as each node has only one parent. Its size is limited only by the computer's main memory. These references are added when each new page is clicked.

Drawing the hierarchical menu is not computationally complex, as each level is drawn only when the menu is expanded; the full tree need not be redrawn each time the menu is opened. In addition, drawing each level only requires directly following the stored references to parents or children; no node needs to be visited more than once.

Unfortunately, there is one large implementation issue: that of memory usage. Because the extension utilizes Firefox's built-in "history entry" objects, which may have references to many other objects that are normally garbage-collected, memory use will simply continue increasing as browsing continues. This makes the current implementation of Histree unfeasible for widespread usage. This problem could be ameliorated by storing the objects on disk rather than in memory, or by simply storing the URLs rather than full history objects. While eminently achievable, that is a non-trivial task, and it was decided that the user study would use the current implementation.

The Histree extension is available for download at http://www.cs.yorku.ca/~orner/histree.html.

4.1 Pilot Study

Histree was offered for download at the aforementioned website while development continued. "Beta testers" were taken from computer science students and friends of the first author. Feedback was largely positive, even during the bug-filled preliminary stages. No slowdown of browsing activity was reported. One user remarked that he had decided to switch from Internet Explorer to Firefox solely on the basis of Histree's functionality - this despite the memory issues mentioned above being explained to him!

A more in-depth and rigorous user study is planned, where the various algorithms will be compared against each other, the current stack-based Back menu, and a recency-list implementation.

5. CONCLUSION

We have presented Histree and FamilyTree, two algorithms which can solve the "pop-off" problem of stack-based Back menus, while keeping screen real estate to a minimum and allowing the user to retain his learned skills of the regular Back menu. We have implemented these solutions as an extension for Mozilla Firefox, and a full user study has been planned for the coming months.

REFERENCES

Ayers, E. Z., et al., 1995, Using Graphic History in Browsing the World Wide Web. *Proceedings of the Fourth International World Wide Web Conference.*, pp. 1-8.

Catledge, L. D., et al., 1995, Characterizing browsing strategies in the World-Wide Web. Proceedings of the Third International World-Wide Web conference on Technology, tools and applications. Darmstadt, Germany, pp. 1065-1073.

Cockburn, A., et al., 1996. Which way now? Analyzing and easing inadequacies in WWW navigation. *International Journal of Human-Computer Studies,* Vol. 45, No. 1, pp. 105-129.

Cockburn, A., et al., 1999, WebView: A Graphical Aid for Revisiting Web Pages. *OzCHI'99: Australian Conference on Computer-Human Interaction.*, pp. 7-14.

Cockburn, A., et al., 2001. What do web users do? An empirical analysis of web use. *Int. J. Hum.-Comput. Stud.,* Vol. 54, No. 6, pp. 903-922.

Cockburn, A., et al., 2002. Pushing Back: Evaluating a New Behaviour for the Back and Forward Buttons in Web Browsers. *International Journal of Human-Computer Studies,* Vol. 57, No. 5, pp. 397-414.

Cockburn, A., et al., 2003. Improving Web Page Revisitation: Analysis, Design and Evaluation. *IT&Society (www.itandsociety.org),* Vol. 1, No. 3, pp. 159-183.

Doemel, P., 1995. WebMap: a graphical hypertext navigation tool. *Comput. Netw. ISDN Syst.,* Vol. 28, No. 1-2, pp. 85-97.

Frecon, E., et al., 1998. WebPath - A Three-Dimensional Web History. *infovis,* Vol. 00, No., pp. 3.

Gandhi, R., et al., 2000, Domain Name Based Visualization of Web Histories in a Zoomable User Interface. Proceedings of the 11th International Workshop on Database and Expert Systems Applications., pp. 591.

Greenberg, S., et al., 1999, Getting Back to Back: Alternate Behaviors for a Web Browser's Back Button. *Proceedings of the 5th Annual Human Factors and the Web Conference.* NIST, Gaithersburg, Maryland, USA.

Greenberg, S., et al., 2000. *Contrasting Stack-Based and Recency-Based Back Buttons on Web Browsers*. Department of Computer Science, University of Calgary, Calgary, Report 2000-666-18.

Hightower, R. R., et al., 1998, Graphical multiscale Web histories: a study of padprints. *Proceedings of the ninth ACM conference on Hypertext and hypermedia.* Pittsburgh, Pennsylvania, United States, pp. 58-65.

Jhaveri, N., 2004. Intermediate and Post-Session Web Page Revisitation Techniques and Tools. In *Computer Sciences*, pp. 53.

JasonSmith, M., et al., 2003, Get a way back: evaluating retrieval from history lists. *Proceedings of the Fourth Australian user interface conference on User interfaces 2003 - Volume 18.* Adelaide, Australia, pp. 33-38.

Kaasten, S., et al., 2001, Integrating back, history and bookmarks in web browsers. *CHI '01 extended abstracts on Human factors in computing systems.* Seattle, Washington, pp. 379-380.

Kaasten, S., et al., 2002, How People Recognize Previously Seen Web Pages from Titles, URLs and Thumbnails. *People and Computers XVI (Proceedings of Human Computer Interaction 2002).*

Milic-Frayling, N., et al., 2003, WebScout: Support for Revisitation of Web Pages Within a Navigation Session. *Proceedings of the IEEE/WIC International Conference on Web Intelligence.*, pp. 689- 693.

Milic-Frayling, N., et al., 2004, SmartBack: Supporting Users in Back Navigation. *Proceedings of the 13th international conference on World Wide Web.* New York, NY, pp. 63-71.

Robertson, G., et al., 1998. Data mountain: using spatial memory for document management. In Proceedings of the 11th annual ACM symposium on User interface software and technology, pp. 153-162.

Tauscher, L., et al., 1997. How people revisit web pages: empirical findings and implications for the design of history systems. *International Journal of Human-Computer Studies,* Vol. 47, No. 1, pp. 97-137.

Wen, J., 2003. Post-Valued Recall Web Pages: User Disorientation Hits the Big Time. *IT&Society (www.itandsociety.org),* Vol. 1, No. 3, pp. 184-194.