Int. J. Human–Computer Studies (1997) **47,** 97–137

How people revisit web pages: empirical findings and implications for the design of history systems

LINDA TAUSCHER AND SAUL GREENBERG
Department of Computer Science, University of Calgary, Calgary, Alberta, Canada T2N 1N4. email: tauscher or saul@cpsc.ucalgary.ca

We report on users' revisitation patterns to World Wide Web (web) pages, and use the results to lay an empirical foundation for the design of history mechanisms in web browsers. Through history, a user can return quickly to a previously visited page, possibly reducing the cognitive and physical overhead required to navigate to it from scratch. We analysed 6 weeks of detailed usage data collected from 23 users of a well-known web browser. We found that 58% of an individual's pages are revisits, and that users continually add new web pages into their repertoire of visited pages. People tend to revisit pages just visited, access only a few pages frequently, browse in very small clusters of related pages and generate only short sequences of repeated URL paths. We compared different history mechanisms, and found that the stack-based prediction method prevalent in commercial browsers is inferior to the simpler approach of showing the last few recently visited URLs with duplicates removed. Other predictive approaches fare even better. Based on empirical evidence, eight design guidelines for web browser history mechanisms were then formulated. When used to evaluate the existing hypertext-based history mechanisms, they explain why some aspects of today's browsers seem to work well, and other's poorly. The guidelines also indicate how history mechanisms in the web can be made even more effective.† © 1997 Academic Press Limited

1. Introduction

The World Wide Web (web) hypertext system is a large, distributed repository of information. People use graphical browsers to navigate through links and to view pages. These browsers typically provide *history mechanisms* that allow people to select and revisit pages they have viewed previously. If people revisit pages often, such history mechanisms can mitigate three problems people face when navigating the web. First, they can help the user navigate through the vast amounts and poor structure of web information by providing easy access to information they had visited previously. Second, they can decrease resource use by supplanting search engines for finding old pages, and by eliminating navigation through intermediate pages en-route to the destination. Third, they can reduce a user's cognitive and physical navigation burdens: pages can be returned to with little effort, and they can show users where they have been.

However, today's design of history mechanisms tend toward *ad hoc* approaches that do not appear to take advantage of previous research into history support within user interfaces, e.g. Greenberg (1993) and Lee (1992). In particular, their designs are not based

† This article is a major expansion of a conference paper (Tauscher & Greenberg, 1997). The research reported in this article was performed as part of an M.Sc. project (Tauscher, 1996).

1071-5819/97/070097 + 41$25.00/0/hc970125

upon actual studies of how people revisit web pages, and their actual use has been examined only superficially.

Our goal is to place the design of history mechanisms within web browsers on an empirical foundation. We had three sub-goals.

1. We wanted to understand people's revisitation patterns when navigating the web, yet little empirical data is available. The proportion of web pages that are revisited by a particular user has not been quantified, and no research has examined patterns of page reuse. In Section 4, we will present quantitative results about revisits to web pages, and examine five possible patterns of reuse.
2. We wanted to evaluate current approaches in today's history systems, validate successful solutions and suggest better alternatives. Yet today's history mechanisms are rarely evaluated. From the research by Catledge and Pitkow (1995), we know that *Back* is heavily used to return to a page, but the history menu is not. Cockburn and Jones (1996) performed a usability study that illuminated user's difficulties with the current stack-based history mechanism. However, the goodness of predictions offered by this and other history schemes have not been evaluated, which we will do in Section 5.
3. We wanted to create a set of empirically based guidelines for the design of history mechanisms for web browsers. Although guidelines for history mechanisms do exist (Lee, 1992; Greenberg, 1993), these were developed from other domains. In Section 6.2, we revisit these guidelines and apply them to the existing browsers.

Before delving into our empirical findings and guidelines, Section 2 will first set the scene by summarizing how current browsers allow their users to return to previously visited web pages. Section 3 will then introduce the study, with results presented in Section 4 (sub-goal 1 above). Section 5 considers the goodness of prediction offered by various history schemes (sub-goal 2). Finally, Section 6 summarizes our findings and presents a few design guidelines that can be applied to the design and evaluation of history mechanisms (sub-goal 3).

2. History mechanisms in graphical web browsers

Many systems include some type of history mechanism that allows users to repeat their previous actions (in command- or menu-based systems), or to revisit previously viewed information (in on-line documents and hypertext systems) (Greenberg, 1993). Many web browsers now include history mechanisms similar to those found in earlier, small-scale hypertext systems. However, novel designs have also been developed to cater to the immense size and unique characteristics of the web. Because the goal of this paper is to place the design of web-based history mechanisms on an empirical foundation, this section will summarize the prevailing approaches found in today's commercial and research browsers. We do not provide a complete survey, for the sheer volume of browsers and add-on tools being developed make it nearly impossible to track all the approaches and their nuances.

Before we begin, a few definitions are in order. A *history list* is a list of previously visited pages maintained by the system. The actual ways that web page references are added to and removed from a history list may vary greatly between systems. Examples include strict sequential lists that place the most recently visited page on top of the list; stacks that push and pop pages onto and off the list; pruning strategies that remove unwanted pages; sessional vs. inter-sessional lists that record only the current browsing session or that track all sessions and so on. These and other approaches will be described in detail in Section 5. *Backtracking* allows a user to visit previously visited pages, and is usually accomplished through one of the two methods. First, *path-following* allows one to traverse in reverse order, their previously visited pages. This method may rely on the user remembering their navigation behaviour, either because they must recall the page visited and their sequence, or because they must realize that they can return to a page by retracing a particular pathway. A *direct jump* to a previous page allows a person to go directly to a previously visited page, without going through the other pages that are on the path to it.

2.1. STEPWISE PATH-FOLLOWING

Stepwise path-following is a method that allows people to retrace their path one page at a time. Many current browsers usually implement this through a *Back* and *Forward* button, menu option or shortcut key (e.g. Netscape's Navigator; see Figure 1). Selecting Back ostensibly traces the navigation path backwards in time; each time the button is pressed, the previous page relative to the current one is raised. Similarly, selecting Forward returns one to where they started from before they pressed Back, again one page at a time. An advantage of this approach is that retracing one's path also recreates their temporal context (the chronological order in which they visited the pages), which could be important in reducing disorientation.

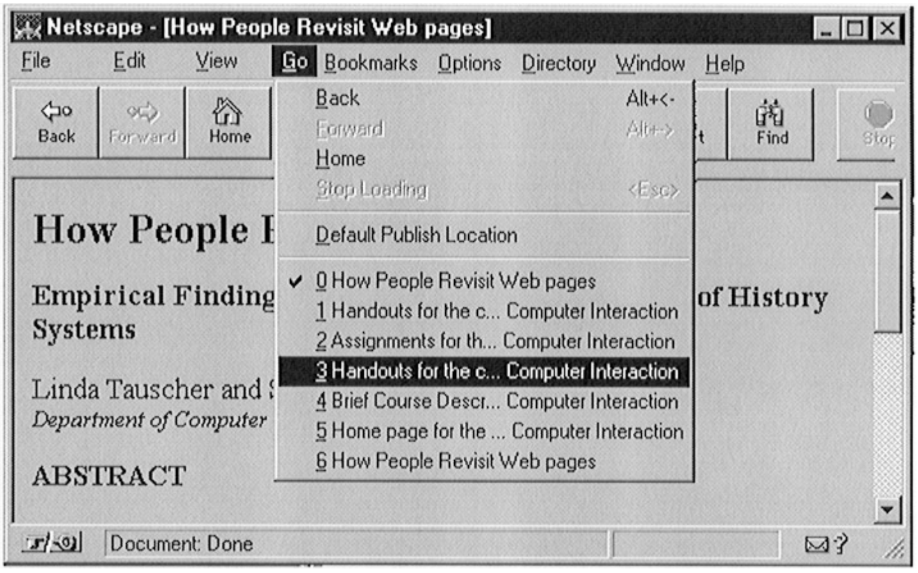

Netscape - [How People Revisit Web pages]
File Edit View Go Bookmarks Options Directory Window Help
Back
Forward
Home
Find
Stop
Back Alt+<-
Forward Alt+->
Home
Stop Loading <Esc>
Default Publish Location
✓ 0 How People Revisit Web pages
1 Handouts for the c... Computer Interaction
2 Assignments for th... Computer Interaction
3 Handouts for the c... Computer Interaction
4 Brief Course Descr... Computer Interaction
5 Home page for the ... Computer Interaction
6 How People Revisit Web pages
How People I
Empirical Finding
Systems
of History
Linda Tauscher and
Department of Computer
ABSTRACT
Document: Done

FIGURE 1. Netscape's Back, Forward and Home functions, accessible as buttons, menu items and shortcut keys. The pull-down Go menu shows the stack-based linear history display.

However, the Back and Forward buttons as currently implemented in most browsers do not do true path-following of a user's navigational trace. Rather, they are built on top of a stack model of visited pages. Depending upon how a page is loaded into a browser, it may or may not be added to the stack, and certain actions actually clear several pages off of the stack. Back and Forward actually move a pointer down and up the stack, and recall the page at the current position. This is problematic, for users have a conceptual model of moving through a literal navigational path, and are surprised when the path presented is different from what they have followed, or when previously visited pages seem to disappear (Cockburn & Jones, 1996). We will return to this issue in later sections.

In spite of the current implementations of stepwise path-following, both our study (see Section 4.1) and a study by Catledge and Pitkow (1995) show that Back is very heavily used, and accounts for between 30 and 40% of all navigation actions on average. In contrast, Forward does poorly, and accounts for only 1–2% of all actions.

2.2. DIRECTED JUMPS TO A HOME PAGE

Almost all browsers let users indicate what page should be presented by default as the first page seen in a session. Called a home page, these are usually created as a launching point to pages of personal interest. Browsers also provide a mechanism, usually a *Home* button or menu option, that allows a person to directly jump to this page (Figure 1). Although not implemented as a history mechanism, it has the effect of backtracking all the way to the beginning of the session. However, both our study and the study of Catledge and Pitkow (1995) indicate that Home is lightly used, accounting for $<1\%$ of all navigation actions.

2.3. DIRECTED JUMPS TO "ALREADY-VISITED" CUES

Browsers typically distinguish hyperlinks from normal text within a page by colouring or underlining them. When the page pointed to by that hyperlink has already been visited (even if it was accessed by some other means), most browsers will inform the users of this by changing the link's colour or style. This is the major "already-visited" cue in graphical web browsers, and users can even set preferences on how long the browser should remember and display already-visited pages. These cues serve as a history mechanism by indicating which links on the page let them jump directly to a previously visited page. Alternatively, they also serve to tell people they need not bother to visit a page because it has already been seen!

2.4. LINEAR HISTORY DISPLAY

A visual display of the linear history list is a common feature of graphical web browsers, and tends to be implemented as either a menu or dialog box containing the page titles that are recallable. A user typically scans the list for the desired page, and selects it to do a directed jump. Stepwise path-following could be done by selecting each page in turn, although this could be confusing if these pages were added to the history list after every visit. Some browsers avoid this problem by not adding pages accessed through a history mechanism.

Different browsers present history in different ways, each with their own scheme for deciding what items to include and in what order. One would expect, e.g. that the history display would be a literal trace of all pages visited. This is rarely the case. Most commercial systems implement the history list and display as a stack, usually including duplicate entries. Some may order the list so the most recent page appears at the top (e.g. Netscape), while others order it bottom-up (e.g. NCSA's Mosaic and TkWWW). These and other approaches for presenting history items will be revisited in Section 5.

An example linear history display, included as part of Netscape's Go menu, is illustrated in Figure 1. This list shows the current contents of the stack, with the most recent (current) page at the top (item 0). Duplicate pages are shown, e.g. items 0 and 6, and items 1 and 3. If the user were to select (say) item 3 from the list, Netscape would directly jump to that page. Netscape would not alter the actual list, but would just indicate the current position in the stack by placing the check mark next to item 3. If the user then chooses a link on that page, all items above that point (items 0, 1 and 2) would be popped off the stack and would no longer be accessible. This illustrates one way in which a stack implementation differs from a strict sequential history list.

2.5. DIRECTED SEARCHES OF THE HISTORY LIST

History displays allow people to recognize what pages they wish to return to. This can be problematic when the number of items on the list is long, or when people are looking for information that is not shown on the display. In this case, being able to perform a parameterized text search on the history list may prove useful.

Most commercial web browsers do not supply a search mechanism. However, they sometimes maintain a file that contains the global history list. Some sophisticated computer users use file manipulation commands (e.g. grep in Unix) to search this file for particular patterns. Recently, a few products have appeared that provide powerful search mechanisms of previously visited pages. For example, ISYS' HindSite, an add-on product built for Netscape's Navigator, indexes the full text of every page the user has visited, allowing one to search for previous pages by not only its title and http address, but by its content as well.

2.6. BRANCHING HISTORY DISPLAY

Hypertext navigation is rarely linear in practice. Some pages act as branching points, and people will follow several links indicated on that page. The linear history display shows this poorly, for the branching point will (perhaps) just be seen as a repeated entry. In contrast, a branching history display provides a two-dimensional representation of the web pages the user has visited. This provides more information about the structure of the web space the user has visited, where the branching points are, and what sub-paths have been followed. Two prototypes of branching history displays are described below.

WebNet (Cockburn & Jones, 1996) displays in a separate window a scrollable graphical overview of the web sub-space visited in a session (Figure 2). Nodes appear as circles labelled with the page title; navigation is represented as a line connecting the source and destination nodes. An interesting feature of WebNet is the ability to see where one can go—the middle mouse button displays the titles of the links present on the

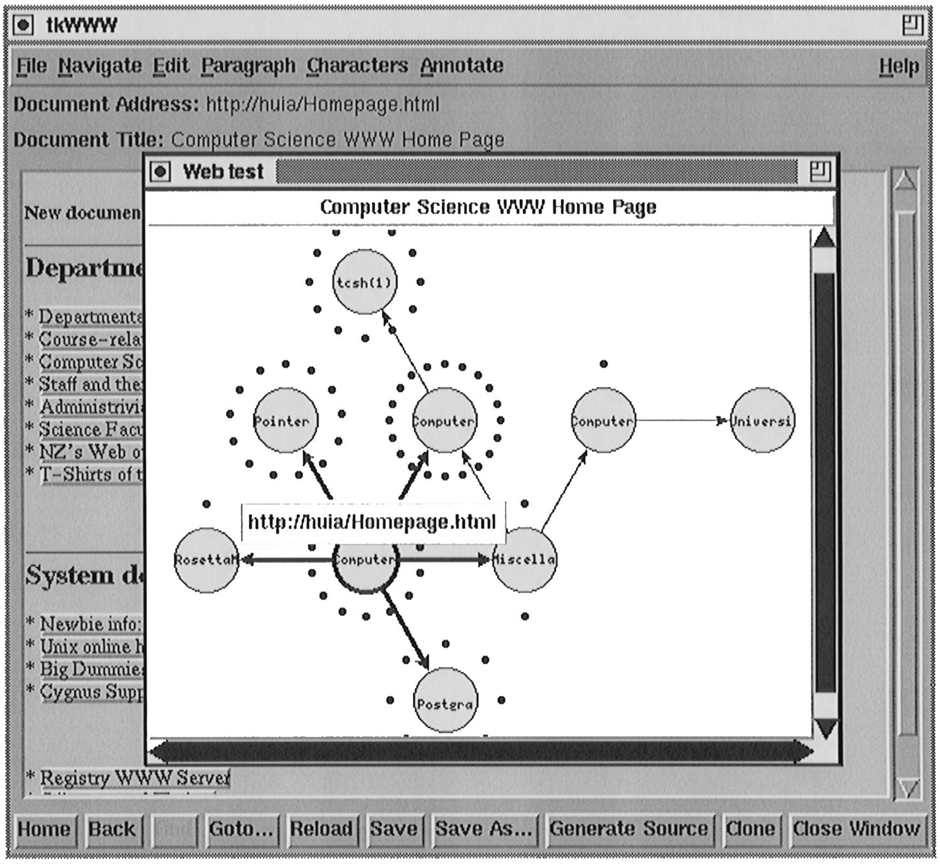
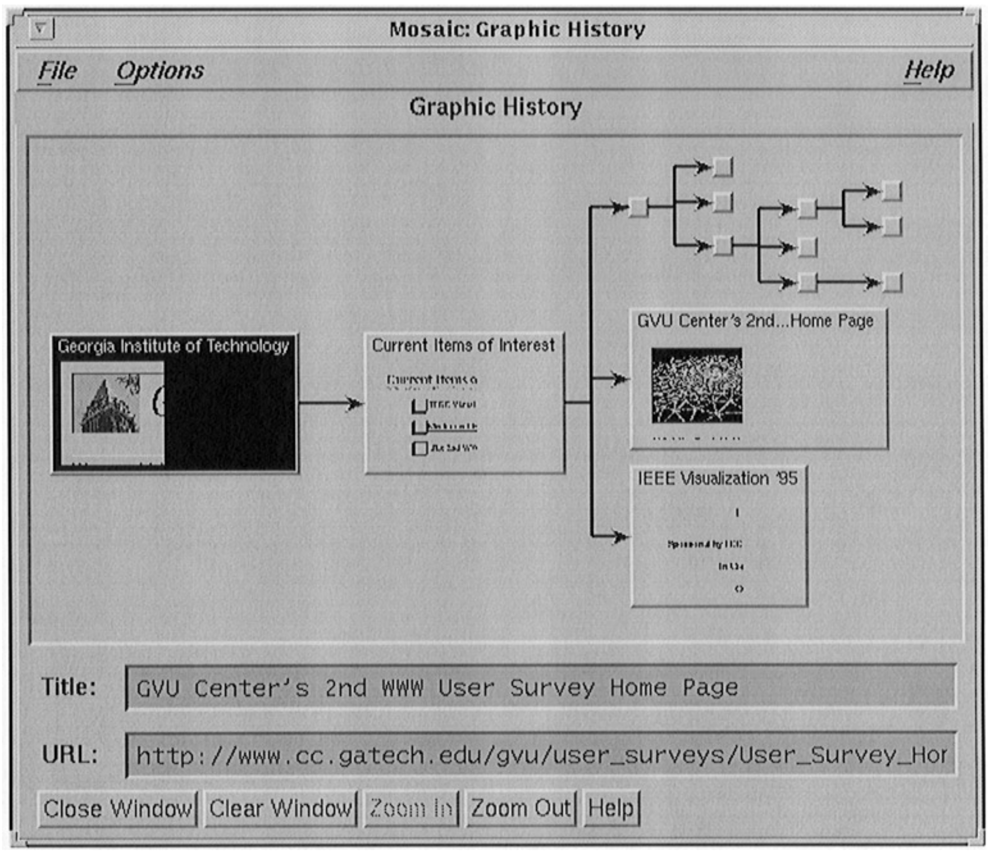

tkWWW
File Navigate Edit Paragraph Characters Annotate Help
Document Address: http://huia/Homepage.html
Document Title: Computer Science WWW Home Page
Web test
Computer Science WWW Home Page
tcsh(1)
Pointer
Computer
Computer
Universi
http://huia/Homepage.html
Rosetta
Computer
Miscella
Postgre
Home Back Goto... Reload Save Save As... Generate Source Clone Close Window

FIGURE 2. WebNet's branching history display (Cockburn & Jones, 1996).

corresponding page. The design of WebNet recognizes that these graphs can be quite complex, and that not all pages have equal status. It offers users a view filter that alters the size of the nodes proportionately with respect to a selected criterion. Current criterion include frequency of visits to pages, recency of visits to pages and distance of pages from the currently displayed page. For example, under the frequency view, the most frequently visited nodes are shown largest. The fewer the number of visits, the smaller the node appears (Cockburn & Jones, 1996).

MosaicG (Ayers & Stasko, 1995) modifies Mosaic version 2.5 to provide a two-dimensional view of the documents a user has visited in a session. The Graphic History View presents titles, uniform resource locators (URLs) and thumbnail images of the documents visited in a session, according to user preferences (Figure 3). The graphical layout is a two-dimensional tree built from left to right with visual cues, which should provide both spatial and temporal context important for reducing user disorientation. As the graph gets large, the user has the options of zooming out for a smaller representation of all documents in the tree, condensing branches of the tree that are no longer of interest



Title: GVU Center's 2nd WWW User Survey Home Page

URL: http://www.cc.gatech.edu/gvu/user_surveys/User_Survey_Hor

FIGURE 3. MosaicG's branching history display (Ayers & Stasko, 1995).

and manually controlling the amount of abbreviation of page titles. All of these solutions involve additional effort on the part of the user. Even so, users of MosaicG have expressed interest in even more power to manipulate the documents and tree structure, e.g. to reparent a node as the root of a tree, and to erase branches completely (Ayers & Stasko, 1995).

2.7. INTER-SESSIONAL HISTORY VIEWS

Commercial browser designs are sorely lacking in support for capturing, accessing and viewing inter-sessional history, i.e. navigations collected across browsing sessions. Only recently have inter-sessional history mechanisms been available, typically as add-on software. They all use various methods of indicating and saving session boundaries, and of pruning, organizing and representing the URLs.

Some systems are essentially sessional-based, but do allow users to save and recall their current history list. For example, the Graphic History View of Ayers and Stasko (1995) in MosaicG allows the user to save a browsing session as a text file; the user must explicitly invoke the *Save* command to do so. While the thumbnail images of each web page are not preserved, they are updated if the page is revisited. Similarly, WebNet (Cockburn & Jones, 1996) allows users to store inter-sessional views of particular navigation paths.

Some systems do save session files, but as individual entities. For example, Navinet Inc.'s Overdrive *Logger* keeps a record of all URLs visited, and permits the user to annotate each site and to turn logging on or off. A separate log file is created for each particular date that the user browses. The functions to filter, sort and combine history files are found within a separate Overdrive module, the *Organizer*. The current implementation of the Organizer is cumbersome, requiring considerable user effort to manipulate the history data.

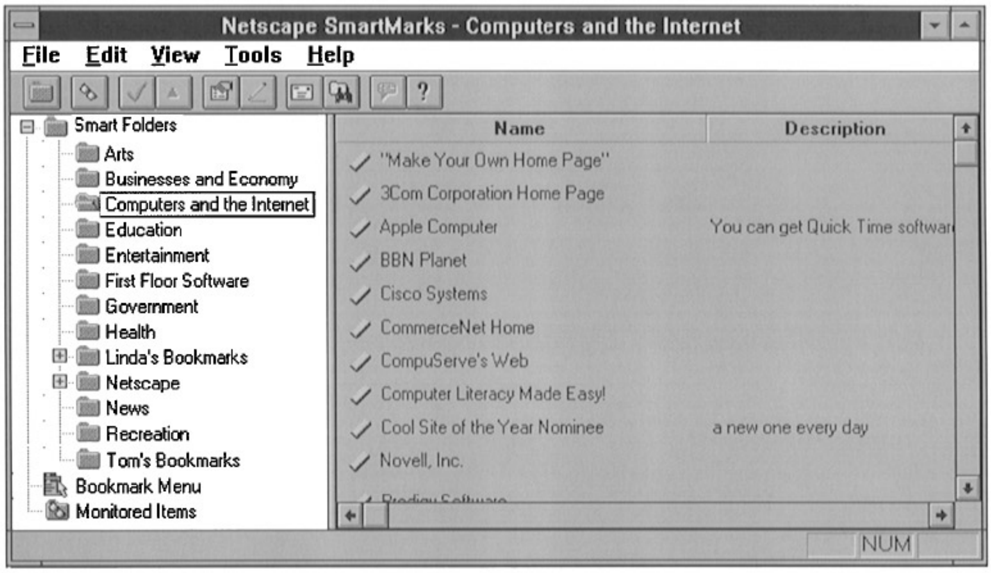
Only a handful of systems maintain a seamless inter-sessional history list. A good example is Apple's Internet Explorer, which records all of the Internet sites visited in the *Log* window. The log is persistent over multiple sessions, and the items in it may be sorted alphabetically, chronologically or hierarchically. One of the most powerful features of the Log is its tight integration with the overall desktop environment. Items in the log may be double-clicked (taking the user directly to that site), dragged into any of the user's personalized lists of URLs, or saved as Internet references on the desktop, allowing the user to share them with others.

Is it worth saving history across sessions? This theme is taken up again in Section 5, where some of our analyses will compare how likely it is that the next page a user would like to visit appears on the sessional and/or inter-sessional history lists.

2.8. PERSONALIZED LISTS OF URLS

All graphical web browsers contain some method for allowing the user to save interesting URLs to a list. This list is called a *Hotlist* in Mosaic, *Bookmarks* in Netscape Navigator and *Notebook* in Apple's Internet Explorer. All act as personalizable history lists of URLs, as they ease the burden of returning to sites in which one is interested. The drawback is that the user must explicitly add the URL to the list while viewing the page on the display or entering its URL into a dialog box.

Most browsers now support hierarchical hotlists, though few users seem to use them to any great extent (see Section 4.1). This may change as the hotlist interfaces are improved. Add-on products are also appearing. For example, SmartMarks is an advanced bookmark facility for Netscape Navigator (Figure 4). It was the first software system with the ability to organize bookmarks in a hierarchy of folders using drag-and-drop editing. Additional interesting features include the ability to save a search as a bookmark, the ability to search local folders, the ability to inform the user of changes to pages and a sample catalog with folders and links to popular topics and web sites. The importance of hotlists is indicated (at least in principle) by the huge efforts that people sometimes go through to author personal pages that do little more than organize personalized lists of URLs. The difference, of course, is that the list is now a hypertext document available from anywhere and by anyone. Managing these personal pages requires more skill and effort than managing bookmarks, as the user must be capable of authoring pages using HyperText Markup Language (HTML) or an HTML authoring tool.



Name	Description
✓ "Make Your Own Home Page"	
✓ 3Com Corporation Home Page	
✓ Apple Computer	You can get Quick Time softwar
✓ BBN Planet	
✓ Cisco Systems	
✓ CommerceNet Home	
✓ CompuServe's Web	
✓ Computer Literacy Made Easy!	
✓ Cool Site of the Year Nominee	a new one every day
✓ Novell, Inc.	

FIGURE 4. The SmartMarks hotlist product for Netscape.

2.9. NEW METAPHORS INCORPORATING HISTORY

A few researchers are considering new metaphors for browsing and collecting information on the web. Some include history mechanisms as an innate feature. One example is DeskScape, an experimental web browser based on a "deck" metaphor (Brown & Shillner, 1995). A deck consists of a collection of web pages, one on top of the other. The ability to collect pages as a deck integrates several components of current browsers, such as hotlists (a deck of personal pages) and the history list (an automatically created deck of pages just seen). It also allows new features, such as expanding all links on the current page (the pages pointed to by the links are returned as a deck). In terms of history, DeckScape retains all pages visited until the user explicitly discards them. Upon backtracking to a higher level in the tree, DeckScape places the next page accessed after its parent page in the deck. This allows the user to quickly switch back and forth between two or more pages that lie on different branches of the tree.

Another example is the WebBook (Card, Robertson & York, 1996), which allows users to collect HTML pages and preserve them as a book. Although somewhat similar in idea to DeckScape, its implementation differs considerably and has some quite powerful features. For example, a variety of novel viewing methods are available to scan the contents of a book, books can be generated as the result of a search, collections of books can be seen in a bookcase and so on.

2.10. DISCUSSION

History mechanisms are prevalent, in one form or another, in all serious web browsers. Obviously, designers feel that it is important to supply some kind of interface to allow users to return easily to revisited pages. Yet approaches are quite varied, and many *ad hoc* design decisions are evident. We really do not know how often people revisit pages (which would indicate the potential value of a history mechanism), or how they navigate back to them. Nor do we know how well the existing history systems are used [excepting the study of Catledge & Pitkow (1995)]. We do not know what items should appear on the history list, how they should be ordered, how long the list should be or how it should be displayed. Nor do we know how likely it is that a person could find their page on that list.

In order to understand the design of history mechanisms, and to place the approaches above in their proper context, we need to understand how people actually revisit web pages. In particular, we ask the following questions.

- Do users return to previously visited pages? At what rate?
- To what degree are current history facilities used?
- Are there particular patterns in how pages are revisited?
- Can browser history mechanisms be designed to accommodate different patterns of reuse, and thereby make it easier to navigate back to pages of interest?

The questions are the basis for our empirical study of web use. The data collection methodology, results, discussions and implications are the topics of the remaining sections of this paper.

3. The study

XMosaic 2.6 was modified to record a user's browsing activity. Each activity record included time, URL visited, page title, final action, method of invoking the action, user id and several other items. Volunteer participants then used the browser for approximately 6 weeks; all were practiced web users with at least 1 year of experience. At the end of the study, we analysed logs from 23 participants. This was followed by hour-long individual interviews, done to gather qualitative data about personal browsing methods and to help users understand why the patterns seen in the analysis arose.

What distinguishes this study from the many other web studies is that it collects data from web clients (browsers) rather than servers. This allows us to capture fine-grained events done by individuals on their personal browser, rather than pooled general statistics typically captured by servers. We are aware of only one other client-side study, by Catledge and Pitkow (1995), which had different objectives than our study (they analysed user navigation patterns on a site basis, and focused their recommendations on document design).

3.1. SUBJECTS

Subjects were 28 unpaid volunteers. All were experienced, having used a graphical web browser for at least 1 year. Nineteen subjects were associated with the University of Calgary as computer science students, researchers, professors or support staff. Nine subjects were employed by a local telecommunications company as programmers or software engineers. While this is a small and quite technically knowledgeable subject pool, we will show in Section 4.1 that the browsing behaviours of our subjects are quite similar to the browsing behaviours seen in users selected from larger and more diverse groups. Consequently, the analysis reported here is arguably representative of a more general population.

3.2. INSTRUCTIONS TO SUBJECTS

Subjects were given an orientation period, which included software installation and familiarization to the standard browser features through a training session. During this time, they were informed verbally of the following.

- Specific data about their browsing actions, including URLs visited, would be accessible only to the investigator.
- Their identity would be kept confidential.
- There would be no noticeable degradation of system performance.
- If the subject encountered a web page that was unreadable in the browser used for the study (Mosaic 2.6), they were instructed to view the page using their normal browser (Netscape Navigator).

The subjects neither required nor received any additional instructions during the actual study period. No subject asked to be withdrawn from the experiment and only one asked to see his personal data.

3.3. APPARATUS

Mosaic 2.6 was modified and compiled for the Sun OS 4.1.4 environment in the Department of Computer Science, and for the HPUX 9 environment at the local telecommunications company. Mosaic 2.6 was the latest supported release at the time the study was undertaken. It is written in the C language, and uses the Motif libraries for its graphical interface components. We instrumented Mosaic 2.6 to generate log files that recorded certain user actions. One log file was created for each subject. We used previously instrumented Mosaic 2.4 source code (graciously provided to us by Lara Catledge and James Pitkow from Georgia Institute of Technology) as a starting point for our own modifications to Mosaic 2.6. Because this study differs, we recorded a smaller and somewhat different subset of actions and data.

3.4. DATA COLLECTION

Table 1 lists the fields in each log entry, and the meanings of most items should be self-evident. Three fields will be explained further: final action; event/action path and same/new window. The final action field recorded the high-level user actions logged for this study, e.g. Exit, Back and Open_URL. The event/action path recorded the method used to invoke the action, e.g. Menu/File/Exit_Program indicates that the user accessed the File menu and selected the Exit Program menu item; some user actions can be accomplished through more than one method. The same/new window field recorded whether the action occurred within the current window or generated a new window. In all, we instrumented Mosaic 2.6 to capture 82 possible combinations of these three fields. The complete list can be found in Tauscher (1996).

TABLE 1
Fields contained by each log entry with examples

Field	Data type	Example
1	Time (Unix system format)	814679050
2	Machine name: process id	dp: 4800
3	User id	204
4	Window number	1
5	Event/action path	Menu/File/Open_URL
6	Same/new window	Same_Window
7	Final action	Open_URL
8	URL of page navigated to or URL modified or filename or email address	http://www.cgl.uwaterloo.ca/ ~ rhbartel/ GI96/info.html
9	Title of page navigated to	Graphics Interface '96

TABLE 2
Browser actions logged

Navigation actions	Non-navigation actions
Back	Clear_Global_History
Binary_Transfer	Close_Window
Clone_Window	Exit
External_Viewer	Hotlist_Add
Forward	Hotlist_Delete
Help	Hotlist_Edit
Home_Document	Hotlist_Insert
Mosaic_Comment	Hotlist_Load
New_Window	Hotlist_Save
Open_Local	Interrupt
Open_URL	Mail_To
Reload_Current	News_Index
StartUp_Document	News_List_Groups
Submit_Form	News_Next
Telnet_Window	News_Next_Thread
	News_Prev
	News_Prev_Thread

One further distinction in user actions was made for data analysis purposes. The 32 final actions were classified as either navigation or non-navigation actions (Table 2). Navigation actions are defined to be those actions that result in the display of a web page within the browser window. While we recorded all possible navigation actions, we only recorded non-naviagtion actions that we considered relevant to the objectives of the study.

Most of the final actions listed in Table 2 are self-explanatory in that they are either menu items and/or buttons on the Mosaic toolbar. The Open URL action, however, comprises the various ways in which a new web page may be displayed in the browser window, and includes the following.

- Anchor: clicking a hyperlink on a web page.
- Keyboard: typing a URL into the URL field at the top of the browser window.
- Hotlist: selecting a URL from the hotlist.
- Dialog: opening the Open URL dialog box and typing the URL.
- History: selecting a URL from the Window History dialog.
- Other: less frequent methods such as choosing a URL from the Documents or Navigate menus or causing a page to display with an external application.

3.5. METHOD

Subjects used Mosaic from 5 to 6 weeks (while all subjects ended their use of Mosaic on the same day, the system was introduced to the different subjects during individual orientation sessions over the first week). Logging began after the orientation sessions, from late October until early December 1995. From the subject's point of view, monitoring was unobtrusive—the modified Mosaic browser was identical in all visible respects to the standard Mosaic browser. However, no subjects used Mosaic as their normal browser, with all preferring Netscape Navigator. This is why the training session in the orientation included a review of the differences between the two browsers.

Six weeks after the study, the investigator analysed the collected data via specially designed analysis software and by manually viewing portions of the raw data. Each subject then participated in a 1 hour interview. During the interview, the investigator asked the subjects questions about their browsing activities and methods, based upon the statistics and plots generated from a subject's personal log files. Subjects were shown their top 15 most frequently accessed URLs and asked to describe their importance.

3.6. DATA SELECTION

A minimum amount of browsing activity is required for usage patterns to become evident and to stabilize. Of particular importance is recurrence rate (see Section 4.1). Cumulative curves from this study show that recurrence rate tends to stabilize after about 200 URLs have been visited. Therefore, if subjects did not generate at least 300 log events during the study period, their data were not considered. By these criteria, data from five of the 28 original participants was discarded, leaving 23 subjects.

3.7. DATA AVAILABILITY

Raw data that have been altered to strip subject's identities may be available to other researchers by special request. However, conditions do apply due to the confidential nature of the data.

4. Results: how people revisit web pages

Seven analyses pertaining to web page reuse are reported here. We begin by presenting some summary statistics, and then report the rate that web pages are revisited. The remaining analyses concern five different patterns that may suggest effective approaches to presenting revisited pages for future access. First, we examine how users visit both old and new web pages over time. Second, we look at the distance (in terms of URLs) between repeated visits to the same URL. Third, we assess the frequency of URL visits. Fourth, we determine the extent to which users repeatedly browse within locality sets (page clusters). Last, we identify repeated sequences of URLs as an estimate of path-following behaviour.

4.1. SUMMARY STATISTICS

An understanding of how users browse, in general, and how history facilities (such as Back) are currently used are fundamental to improving browser history mechanisms. In this subsection, we identify the primary methods people use to access an URL and the extent of their usage. We also consider the frequency of use of current browser history mechanisms—the back and forward buttons, hotlists and the history display. Finally, we reanalyse the Catledge and Pitkow (1995) data, and contrast the summary statistics of our subjects with theirs.

The 23 subjects in our study used the web at quite different rates. While a mean of 902 log statements were generated by each subject (median = 701), there is a wide standard deviation ($\sigma = 676$). Individual traces ranged from a low for one subject of 303 log statements to a high for another subject at 3299 statements. Regardless of the number of log statements a person generated, their most prevalent browsing action involved navigating to an URL; fully 90% ($\sigma = 4$) of the log statements were of this type. Of the remaining 10% of actions, about 3% involved: Hotlist manipulation (adding, deleting, modifying or inserting URLs); 2.5% were interruptions of the current page transfer; 3% were actions that terminated the session and a few other rare events (such as news reading and clearing the history list).

Because we are interested in navigation actions (90% of all recorded actions), they deserve further consideration. Frequencies of the individual events which comprise navigation actions (as detailed in the left column of Table 2) were analysed as a percentage of the total navigation events. Results are tabulated in Table 3. For data presentation purposes, some of these events have been collated within a single category.

- Home collects both Home_Document and StartUp_Document navigation actions.
- Helper_Application collects navigation to both an External_Viewer and Telnet_Window.
- New_Window collects Clone_Window, Help, Mosaic_Comment and New_Window.

As seen in Table 3 and as graphed in Figure 5(a), Open URL was the most frequently invoked action, making up 50% of all navigation events. Back was next at 30%. In contrast, Forward was infrequently used (1% of all navigation actions). (However, we should recall that Forward is only available after a user has gone "Back" at least one step.) Home, which includes both the automatic display of the StartUp document ($\sim 4\%$) and user selections of the *Home* document ($\sim 1\%$), occurred 5% of the time. Reload current comprised an average of 3% of the navigation events. However, it is worth mentioning that three subjects had very high use of Reload Current, at rates of 10, 13 and

TABLE 3
Frequency of navigation actions as a percentage of total navigation events

	Open URL	Open local	Back	Forward	Home	Reload	Binary transfer	Helper apps	New win	Submit form
Mean	50.07	0.16	30.24	0.84	5.04	3.07	1.11	4.56	0.75	4.16
S.D.	7.51	0.33	8.46	0.90	2.36	4.02	1.44	5.02	0.83	2.60
Mdn	51.33	0.00	31.33	0.60	4.38	1.15	0.29	3.32	0.44	4.33
Min	34.36	0.00	16.70	0.00	1.65	0.28	0.00	0.00	0.09	0.57
Max	65.34	1.18	44.54	3.49	9.86	14.54	4.72	18.09	3.86	8.48

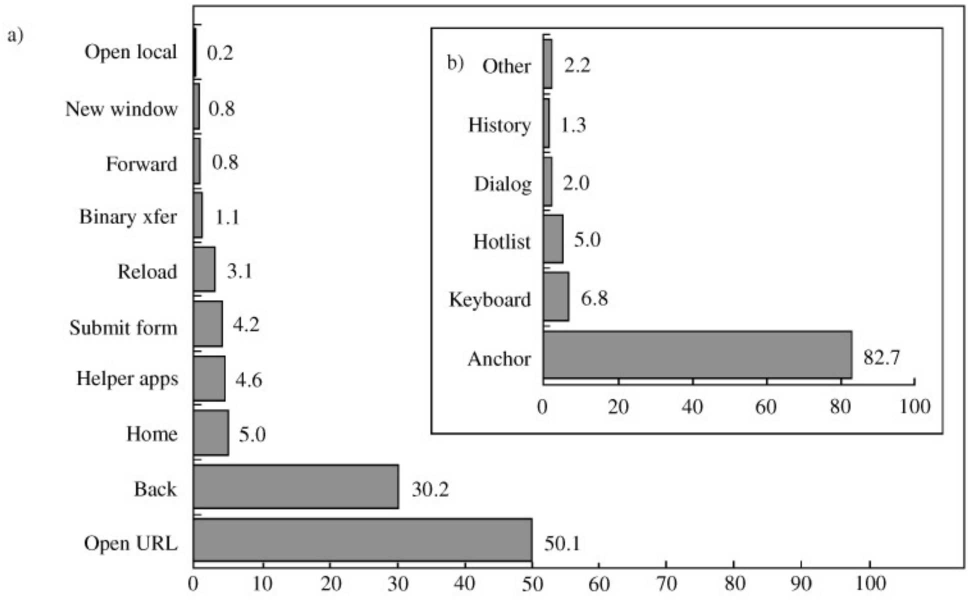


FIGURE 5. (a) Frequency of navigation actions as a percentage of total navigation events. (b) (Inset) frequency of Open URL types as a percentage of total Open URL actions.

15%. This heavy use is typically associated with page authoring, and is described further in Section 4.3. The remaining "special case" navigation actions included downloading of a file (binary transfer) at 1%; invoking Helper Applications such as telnet and image/document viewers at 5%; actions that created a new browser window (New Win) at <1% and submitting forms (used by search engines and interactive web applications) at 4%.

The Open URL category makes up 50% of all navigation actions. However, the Open URL category is itself comprised of individual events. Figure 5(b) shows the different types of Open URL events, and the proportion of each observed across all subjects. As seen in the figure, the most popular method of selecting an URL is via clicking a hyperlink or anchor (83% of all Open URL events). Typing an URL directly into Mosaic's URL field is done at a modest level (7%), as is accessing a page from the hotlist (5%). Users can also type URLs into a dialog box raised from a menu selection, although this is not done often (2%). Mosaic's Window History was used rarely to select a URL (1% of all Open URL actions).

We reanalysed data captured by Catledge and Pitkow (1995), and compared the above results with theirs (Tauscher, 1996). They also captured client-side user events from a larger pool of subjects using an older version of XMosaic during a 3 week period. Open URL dominated both groups (50% for our group, 54% for theirs), as did Back (30 vs. 37%). Forward was used rarely in both groups. As a percentage of Open URL actions, navigating via anchors prevailed in both groups (42 vs. 50%), while navigating via the hotlist or history list remained infrequent (2.7 vs. 2.3% and 0.7 vs. 0.1%, respectively). Although some of the observed differences between these groups are statistically significant, their existence can be explained by differences between browser capabilities, by differences in the actual data recorded and by differences in the web itself. We believe the results are similar for practical purposes. Consequently, we assume that the web browsing behaviour observed in our subjects is fairly typical. We can also assume that the other analyses we do (that were not done by Catledge and Pitkow) are probably generalizable to a larger population.

In summary, the statistics indicate that half the navigation actions are Open URL events. The only history mechanism used extensively is the Back button (30%). Other mechanisms, including Home, Forward, the hotlist and history menus, are used infrequently. Finally, the browsing activities in both our study group and the Catledge and Pitkow study group are, for practical purposes, similar. Consequently, we assume that the way our subjects behave are typical of web users in general.

4.2. RECURRENCE OF WEB PAGE VISITS

History systems are only useful if users actually repeat their activities. While web browsers contain various features to support history (Section 2), we do not know how often people revisit their pages. This is important, as it gives us a bound for how useful a history system can be. In other domains, Greenberg (1993) has quantified this repetition of user actions. Examples include the frequency with which the telephone numbers are redialed (57%), how the same information is retrieved in a technical manual (50%) and how Unix command lines (including arguments) are repeatedly re-entered (75%). We analysed our own data and the Catledge and Pitkow (1995) data to derive the *recurrence rate* R: the probability that any URL visited is a repeat of a previous visit, expressed as a percentage. This is calculated simply as

$$R = \frac{\text{total URLs visited} - \text{different URLs visited}}{\text{total URLs visited}} \times 100\%.$$

Each user's data was analysed independently. The statistics below represent averages across all users, or representative individuals.

We found an overall recurrence rate of $R = 58\%$ ($\sigma = 9\%$) for our 23 subjects, and 61% ($\sigma = 9\%$) for 55 subjects from the Catledge and Pitkow study. These numbers clearly show that users revisit web pages to a significant degree. However, it also means that $\sim 40\%$ of all page navigations are to new pages. These recurrence rates qualify web browsing activity as a *recurrent system*. Greenberg (1993) coined this term to characterize systems where users predominately repeat activities they had invoked before, while still adding new actions to the repertoire from the many that are possible.

In post-study qualitative interviews, people gave us their major reasons for visiting new pages and revisiting old ones. People visit new pages due to the following reasons.

- Their information needs change.
- They wish to explore a particular site.
- A page is recommended by a colleague.
- They notice an interesting page while browsing for another item.

They revisit pages due to the following reasons.

- The information contained by them changes.
- They wish to explore the page further.
- The page has a special purpose (e.g. search engine, home page, index page).
- They are authoring a page.
- The page is on a path to another revisited page.

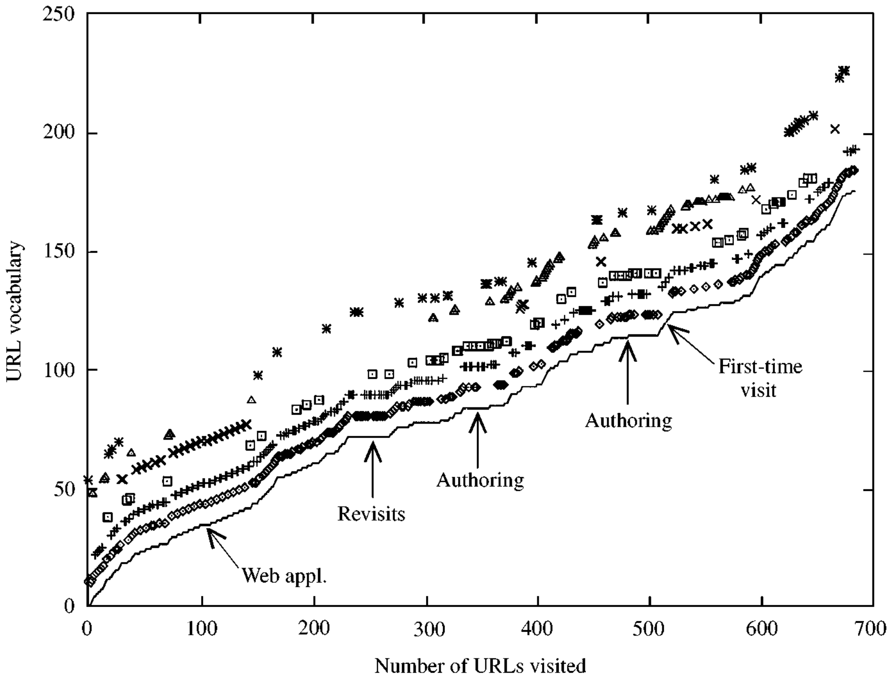
Since people's web browsing activity behaves as a recurrent system, we believe that browser interfaces should support page revisits by minimizing a user's effort to return to a page when compared to the effort of navigating there from scratch. The key is to give preferential treatment to the large number of repeated actions. This involves identifying patterns in history use, as discussed in subsequent sections.

4.3. GROWTH OF URL VOCABULARY

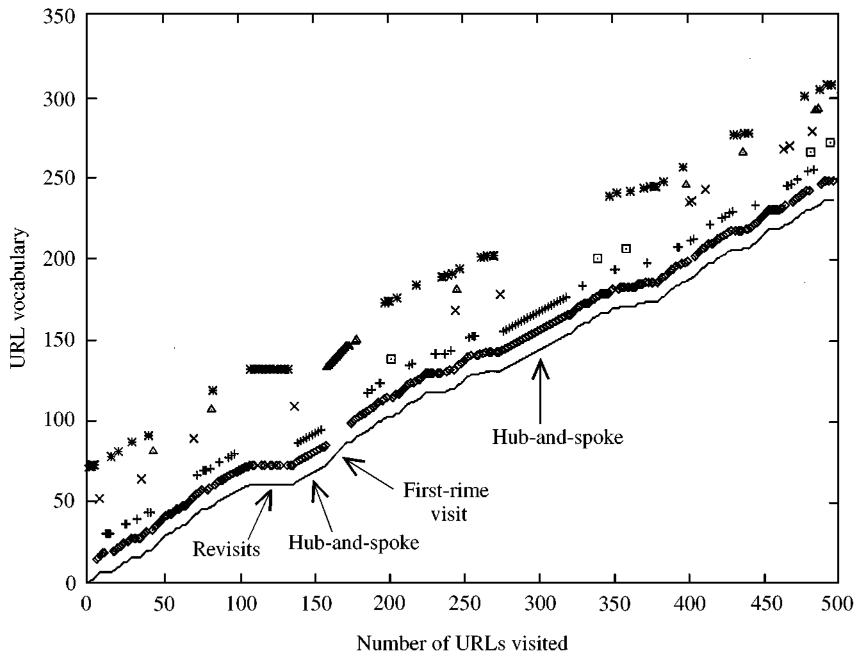
The first pattern considered shows the distribution of old and new page visits over time. We generated vocabulary graphs for each subject, where the URL *vocabulary*—the number of unique URLs seen so far—is plotted over the total number of URLs visited. These plots illustrate how users extend their vocabularies, how recurrences are distributed over time and when they do particular browsing actions.

For example, Figures 6(a) and (b) show the plots for participants 15 and 17. The curve *All* represents the overall URL vocabulary size at any moment in time. Major navigation actions are also plotted as separate curves shifted above the vocabulary line by a constant amount: *Open URL*, *Back*, *Reload*, *Forms* and *Helper Applications*. The *Other* category includes all remaining navigation actions. These curves show when the most common navigation actions were invoked and, taken together, comprise the *All* curve. URL vocabulary growth graphs for all subjects exhibit a linear slope, typified by the All lines in Figures 6(a) and (b). Both data and interviews indicate that users incorporate new URLs into their repertoire at a regular rate, and that revisits are fairly evenly distributed.

These plots are only roughly linear, and many local variations to the slope are also evident. We noticed that the nature and extent of these vary amongst individuals. We analysed these variations and their corresponding navigation actions, and asked participants questions about them during interviews. Consequently, we identified seven browsing patterns, five of which are illustrated in Figures 6(a) and (b).



a)



b)

FIGURE 6. URL vocabulary plots for (a) participant 15; (b) participant 17. ▬ All; ◇ Open URL; + Back; ⊡ Reload; ✕ Forms; △ Helper apps; ✱ Other.

1. *First-time visits* to a cluster of pages is evident as steeply sloped areas, e.g. between URLs 500 and 520 on the *x*-axis in Figure 6(a), and between 160 and 175 in Figure 6(b).
2. *Revisits to pages.* Plateau areas show revisits to pages. For example, we see a long plateau in combination with the Back or Open URL actions between URLs 240 and 280 in Figure 6(a), which occurred when this participant reviewed a set of on-line course notes they had seen before. Another example of plateaus occurred between URLs 110 and 140 in Figure 6(b).
3. *Authoring of pages.* These also manifest themselves as plateaus, where the subject uses Reload extensively to view the modified page, e.g. between URLs 480 and 510 in Figure 6(a). Note that this pattern did not appear in Figure 6(b), as the participant did not do any authoring.
4. Regular use of *web-based applications.* Between URLs 50 and 150 in Figure 6(a), there is a moderately sloped area with a combination of Open URL, Back and Forms activity. This was caused by the subject's revisits to a knowledge elicitation tool packaged as a web application.
5. *Hub-and-spoke.* People visit a central page (hub) and navigate the many links to a new page (spoke) and back again. We see this between URLs 140–160 and 280–320 in Figure 6(b), as an alternating occurrence of Open URL and Back. In both cases, these actions were consistently used to return to an index page to access another URL on the page. This is akin to a breadth-first search.

Two other browsing patterns were seen in other subjects, although they are harder to identify visually on these particular figures.

6. *Guided tour.* Some page sets include structured links (e.g. *next page*), and people can choose to follow these. These are typically manifested as a series of consecutive Open URL actions, where people choose Next and Previous hyperlinks embedded into the web page. This area will be steeply sloped if this was the user's first time following the guided tour, and they repeatedly chose the Next hyperlink.
7. *Depth-first search.* People follow links deeply before returning to a central page, if at all. This is visible as a series of consecutive Open URL actions followed by a series of *Back* actions, akin to the region around URL 200 in Figure 6(a). If this is a first time visit, the curve will have a step-like appearance.

Browsers and history mechanisms should support the many browsing patterns users exhibit. For example, stack-based history mechanisms and the Back button found in most browsers support both hub-and-spoke navigation and backtracking from depth-first search patterns. The Reload button is very convenient for authoring. Guided tours contain hyperlinks that encourage a linear pattern of navigation. Yet improvements in history designs are possible. Perhaps the excessive backtracking that results from depth-first navigation styles and the hub-and-spoke could be reduced by a branching history display as described in Section 2.6, or by retaining the index page within the browser window, as is often done with Netscape Navigator 2.0's *frames* feature.

In this section, we saw that both old and new web pages are visited regularly over time. The huge number of pages in the vocabulary implies that we cannot simply offer all previously visited pages within a visual history system, as this would be unmanageable.

Instead, we must offer the user only a handful of candidate pages that have a high probability of being revisited. Researchers in other domains have noticed that recently used actions are the ones most likely to be repeated. (Lee, 1992; Greenberg, 1993). This warrants an investigation into recency effects by considering URL visits as a function of distance.

4.4. URL VISIT FREQUENCY AS A FUNCTION OF DISTANCE

For any URL visited, the average probability that it has already been seen by the user is quite high ($R = 58\%$). But how do particular URLs contribute to this probability? Do all URLs visited have a uniform probability of recurring or do the most recently visited URLs skew the distribution? If a graphical history system displayed the previous p entries as a list, what is the probability that this includes the next entry (Greenberg, 1993)?

The recurrence distribution as a measure of *distance* was calculated for each subject. Distance is determined by counting the number of items between the current URL being visited from its first match on the history list. For example, a distance of 1 occurs when the user reloads the current page or successfully interrupts the transmission of a page. A distance of 2 occurs when the current page is a revisit to the one seen two pages ago. The mean recurrence rate R at a particular distance d over all S subjects is formally calculated as

$$R_d = \frac{1}{S} \sum_{s=1}^{S} R_{s,d},$$

and the algorithm to calculate $R_{s,d}$ is the following.

```
Given: a trace of URLs numbered from 1 to n, where n is the most recently visited URL
    an array of counters used to accumulate the number of recurrences at a particular
        distance
For each URL in the trace: find its nearest match on the history list of previously
    visited URLs
    calculate the distance (in terms of URL entries) between the current
        URL and its most recent match
Normalized proportion of the number of recurrences at a particular distance:
    for (distance := 1 to $n$)
        counter[distance] := (counter[distance]/$n$) * 100
```

Figure 7 plots this data up to a distance of 50, averaged across all subjects. The horizontal axis shows the distance of the repeated URL on the history list relative to the one being entered. The vertical axis represents R_d, the rate of URL recurrence at a particular distance. According to Figure 7, there is a $R_{d1} = 10\%$ probability that the current URL is a repeat of the previous URL (distance = 1), $R_{d2} = 19\%$ for a distance of 2, $R_{d3} = 2\%$, $R_{d4} = 5\%$, etc. Figure 8 explains how the use of Back contributes to the jaggedness of the line. First, a distance of 1 usually occurs in web navigation when the user reloads the current page or when they jump to a different part of the page (via internal anchors, which we counted as a navigation to the same page). Next, the spikes at the even page distances arise from users' navigating back to previous pages by the

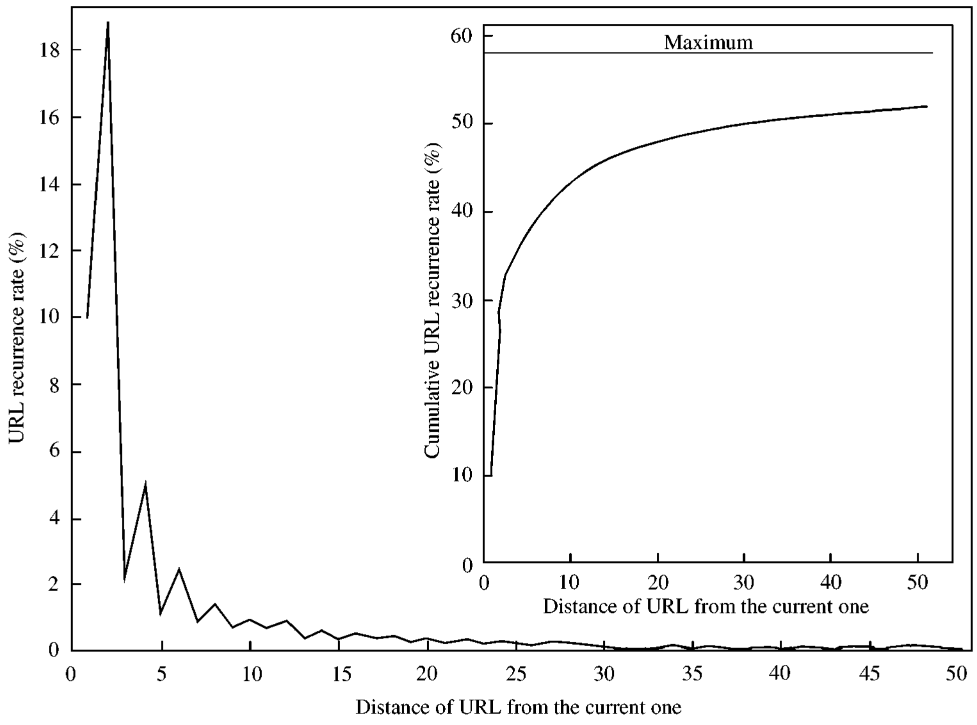


FIGURE 7. URL recurrence rate as a function of distance (all participants); inset plots recurrence rate as a running sum.

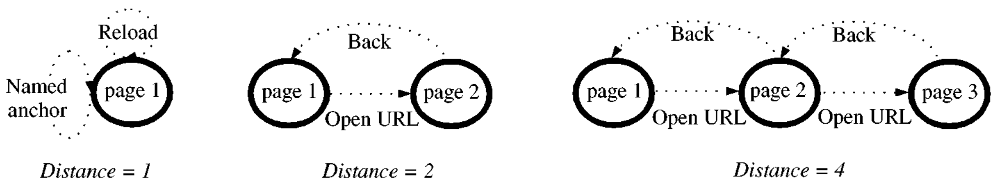


FIGURE 8. Distances generated by *Reload/Internal Anchor*, *Open URL* and *Back* actions.

"Back" navigation action (probably emerging from both hub-and-spoke browsing and from depth-first searches). The spike at a distance of 2 arises due to users' navigating back to the previous page from the current one. Performing a *Back* navigation action twice in a row accounts for the spike at a distance of 4. Similarly, performing three *Back* navigations consecutively likely accounts for the spike at a distance of 6. The most striking feature of this data is the extreme recency of the distribution. The 6 or so URLs just visited contain the majority of pages visited next, although the probability values of R_d rapidly decrease after the second item. Beyond a distance of 8, the low values ($< 1\%$ each) and low rate of decrease make these distant items equivalent for practical purposes.

This is illustrated further in the inset of Figure 7, which reports the same data for all subjects as a running sum of the recurrence probability up to a particular distance, denoted as R_D. The horizontal line at the top of the graph is the maximum mean recurrence rate for all subjects of 58%. The recency effect is clearly visible, where the most recently visited URLs are responsible for most of the cumulative probabilities. For example, there is an $R_{D6} = 39\%$ chance that the next URL visited will match a member of a set containing the six previous submissions. In comparison, all further contributions are slight (though their sum total is not). To illustrate, the probability values from $R_{D6} = 39\%$ increase by only 4 percentage points when the set includes the last 10 URLs ($R_{D10} = 43\%$), 9 points at 20 ($R_{D20} = 48\%$) and 13 points at 50 ($R_{D50} = 52\%$).

4.5. FREQUENCY OF URL ACCESSES

Frequency is a popular method for ranking items of interest. We examined this pattern in two ways. First, we generated a frequency graph for each subject. Second, we developed a taxonomy of conceptual page types for frequently visited pages from our post-study interviews.

All subjects produced a similar frequency distribution, where only a small number of URLs are highly visited, and a very large number of URLs have very low usage frequencies. Each subject, on average, visited 60% of their pages only once, 19% were visited twice, 8% were visited 3 times and 4% were visited 4 times (Figure 9). However, a handful of URLs were visited frequently. Subjects were shown summary reports listing their top 15 URLs by frequency. They were asked to explain why they were frequently accessed. The list below indicates how people categorized their frequently visited page types. Users tended to revisit the first five page types repeatedly over a long time, whereas they revisited the remaining types in intense bursts and did not revisit them again.

1. *Personal page*(*s*). These pages have been authored by the user typically for their own use. They may include personal information and hyperlinks to web sites of high interest. If the user owns several personal pages, a main page linking them together is referred to as their personal home page. For nine subjects, the personal home page was the most frequently accessed page. For seven subjects, their home page appeared either second, third, fourth or sixth on the list. For two subjects, this page did not appear on the top 15 list.
2. *Start-up page*. For 15 of the 23 subjects, the start-up document was the most frequently accessed page. This overlaps somewhat with personal pages, for users often set their personal home page as the start-up document. For the eight corporate subjects, the start-up page was set by the system administrator to the local corporate home page, though two subjects did choose to change this to their personal home page. The 15 subjects from University of Calgary were permitted to set their start-up document to whatever page they wished. Twelve subjects used their personal home page; one used his department's home page; another used a group research home page he had authored; the final subject showed a variety of start-up documents because he used an "URL Launcher" application that invoked Mosaic with particular URLs he had typed into the Launcher.

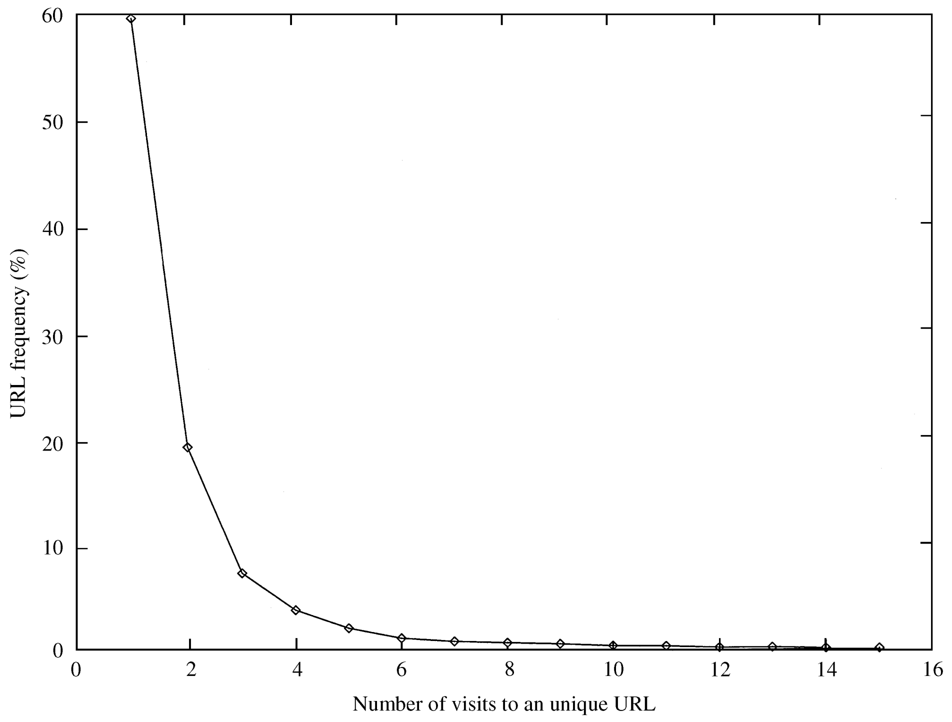


FIGURE 9. Frequency of URL visits for all subjects.

3. *Organization/other user's home page.* This is the main page for an organization or individual making information available via the web. It is revisited frequently because it acts as a gateway to pages for that organization/individual.
4. *Index pages.* This page contains links pertaining to a particular topic, and is a good starting point when the user is searching for information about that topic. It also acts as a gateway to other pages.
5. *Search engines.* This page type includes web-based applications that specifically assist the user in locating a web page or site. Examples are WebCrawler, OpenText, Lycos and site-specific search engines.
6. *Web applications.* Some pages are revisited because they are closer to applications than static information. Custom web applications used by subjects during this study include a knowledge elicitation tool, a print queue query, a corporate phone book query, a HyperNews discussion group for a project, a French to English dictionary and a documentation status interface.
7. *Navigation page.* This type of page is used to access a page of interest; it has no other use to the user. It will appear as frequently accessed if the user often follows a particular path that includes the navigation page.
8. *Authored page.* This type of page will be visited often as it is being developed during an authoring phase.

What is clear from these findings is that only a few pages are visited frequently (Figure 9), those that typically belong to one of several page types. Some are visited on a regular basis, while others may be revisited several times during the same browsing session but are no longer of interest after that.

4.6. LOCALITY

While recency characterizes recurrences in terms of distance, *locality* considers recurrences in terms of periods of time where repeated references are made solely to a small and related group of items, called a locality set. While this concept originates in computer memory research (Madison & Batson, 1976), several researchers have applied this analysis to user interactions (Henderson Jr & Card, 1986; Lee, 1992) on the premise that we can associate portions of a person's task with locality. This assumes, of course, that the user will select actions from a particular subset of activities every time they perform the task. We reconsider locality here in terms of page access. For example, consider a user who goes to a site to read on a topic, where her next 10 URL navigations just switch between four unique pages. The locality set would be of size 4, with a duration of 10. If she revisited this site and the same set of 4 pages more than once, we would say that the locality set has recurred.

We applied the locality detection algorithm, which originated in computer memory research (Madison & Batson, 1976; Lee, 1992), to the web data to determine whether users generate locality sets, i.e. whether they browse within clusters of pages. We analysed the results in four ways. First, we calculated the mean number of locality sets of a particular *size* for each participant and all participants. Second, we calculated the average *duration* of sets of a particular size. Third, we assessed the extent of locality set *recurrences* by reporting the total and unique occurrences for each set size, and by plotting locality sets generated over the user's browsing timeline. Fourth, we attempted to identify the activities that comprise locality sets by reviewing reports listing these sets with each participant during post-study interviews. Further details on the analysis and results are reported in Tauscher (1996). In this paper, we present only a summary of the main results.

While locality sets were found, they do not appear to offer much value in terms of predicting the user's next activity within web browsing. There are several reasons for this claim.

1. Most locality sets were very small, consisting of only one or two unique URLs; Figure 10 illustrates this. The x-axis is the actual size of the locality set, while the y-axis plots the number of times that set size occurred. The *total sets* plot in the figure shows the dramatic drop in numbers of sets as the set size increases. A locality set of size 1 is already captured by a simple history list as the last item entered. Sets of size 2 are of limited usefulness, for it is likely that the other URL in the set is a hyperlink on a current page and, thus, very easy to access in this manner.
2. These sets lasted for only a short time. Their minimum duration must equal their set size before they can be counted, and observed locality sets did not go much beyond this. For example, the weighted average duration over all subjects for sets of size 1, 2 and 3 was 2.9, 4.4 and 5.7 URLs, respectively.

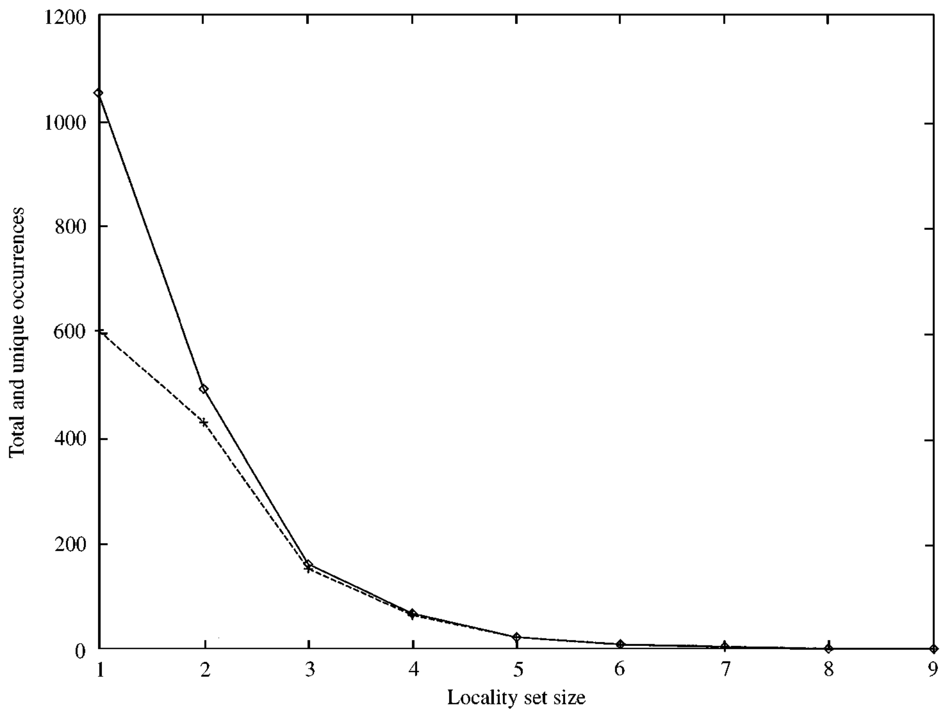


FIGURE 10. Total vs. unique locality sets per set size. -◇- Total sets; ·+· Unique sets.

3. Few locality sets were repeated. 79% of sets of size 1 were unique while 87% of sets of size 2 were unique. All sets of size 5 and greater were never repeated by the user. The *unique sets* plot in Figure 10 illustrates this phenomena, where repeatability drops dramatically as set sizes increase.
4. Only 15% of URLs visited were part of a locality set. This is known as *locality rate*. Our result means that locality sets captured for future use will only contain 26% (15% of $R = 58\%$) of the URLs the user is likely to revisit.

Post-study interviews revealed several browsing behaviours that influence the nature and extent of locality set occurrences.

1. In general, participants who were more directed and focused in their use of the web generated more locality sets, had sets of longer duration and had higher locality rates. For example, 42% of URLs browsed by one participant were part of a locality set. However, this was an extreme case.
2. Some simple sequences of actions generated locality sets. For example, locality sets of size 1 appear with repeated use of Reload, because the same URL is reaccessed repeatedly. Thus, individuals performing authoring tasks showed higher than average locality rates. Similarly, participants who navigated within a single page using internal anchors generated locality sets because we counted these as a revisits to the same page.

3. Participants using web-based applications (i.e. an application based on forms) generated more locality sets because their browsing was more confined to a set of URLs. However, web-based applications were not used extensively by most participants. This may change as web-based applications increase in number and usefulness.

In conclusion, though locality is an interesting concept, it does not appear to offer much in terms of predicting the user's next activity within web browsing. Locality set sizes tend to be very small, and sets themselves rarely repeat. It is also unclear how locality could be applied practically within a working history system. In general, locality is too constrained or limited a concept for web browsing because it requires that references be made solely to a particular set of URLs; one URL reference outside of this set breaks the phase. This does not mean we should give up, for there could be other (better) ways of clustering pages that identifies the particular tasks that people are working on, e.g. the explicit grouping of pages into sets as found in Rooms (Henderson Jr & Card, 1986) and DeckScape (Brown & Shillner, 1995).

4.7. LONGEST REPEATED SEQUENCES

The concept of *paths*, an ordered traversal of hypertext links, has been associated with hypertext ever since Vannevar Bush envisioned hypertext in 1945. If paths exist, it may be useful to capture and offer them via history, thus simplifying people's efforts to retrace a path. Also, if users follow paths solely as a route to a destination, shortcuts could allow a user to go directly there.

We applied the pattern detection module (PDM) algorithm (Crow & Smith, 1992) to the data to identify longest repeated sub-sequences (LRSs) of page visitations. This algorithm finds the longest repeated sub-sequences (LRSs) in a log file, where the following are defined.

- *Repeated* is defined as a sub-sequence occurring at least twice; thus, the minimum frequency for a LRS will be two.
- *Sub-sequence* is a set of consecutive symbols.
- *Longest* means that, although a sub-sequence may be part of another repeated sub-sequence found by the algorithm, there is at least one occurrence of this sub-sequence in the log file where it is the longest repeat. For example, the algorithm may find both "a b c" and "a b c d e" as LRSs, if on at least one occasion "a b c" is not followed by "d e". As another example, the LRSs found may sometimes overlap, e.g. the LRS "a b c d" and the LRS "c d e f" may be represented on one occasion by "a b c d e f", as long as they also occur separately (Crow & Smith, 1992).

In the domain of web browsing, the LRS members are URLs that the user has visited. We used three methods to analyse the nature and extent of LRSs found in our subjects' URL traces at individual and aggregate levels. First, we calculated the mean number of LRSs and the average frequency of LRSs of a particular length. Second, we examined participants' plots to qualitatively assess recurrences. Third, we attempted to identify activities that comprise LRSs through post-study interviews. Further details on the analysis and results are reported in Tauscher (1996), and only a summary of the main results are reported here.

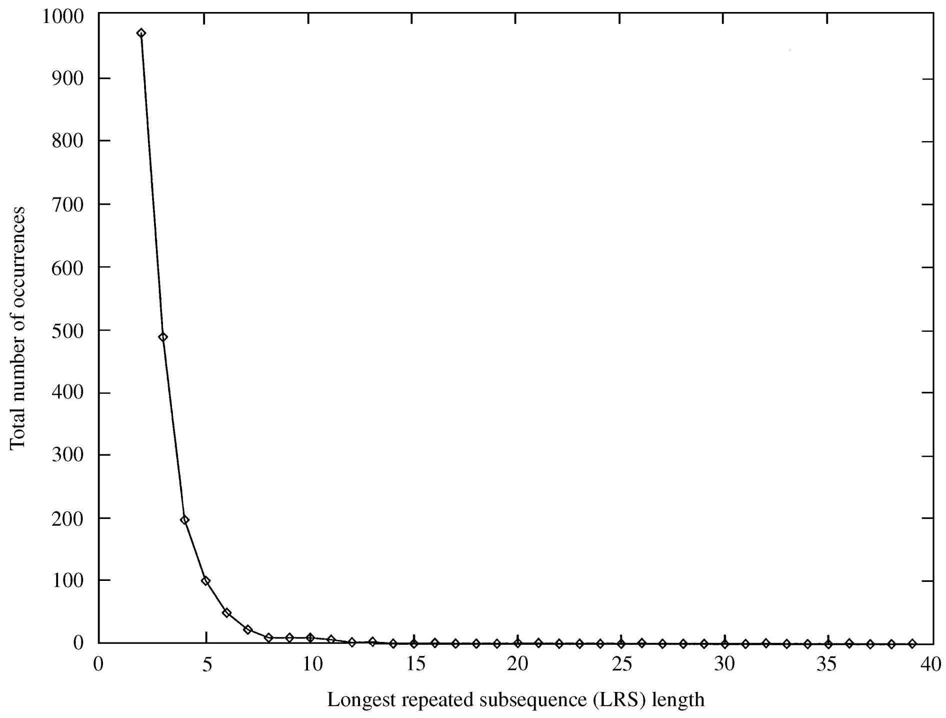


FIGURE 11. Total number of occurrences for longest repeated sub-sequences of a given length.

As with locality, we discovered that LRSs are not particularly useful for predicting web browsing for a number of reasons.

1. While LRSs do exist, they tend to be short. Figure 11 illustrates this phenomena. The x-axis indicates the length of the LRS, while the y-axis describes how often the LRS of that length occurs. The plot shows the dramatic drop in LRS occurrences as the sequence length increases. While 971 LRSs of length 2 were found, only a small number of LRSs of length 7 or greater exist (Figure 11).
2. The few longer URLs usually reference only one or two pages. As with locality sets, we noticed that this occurred when participants reloaded a page several times during an authoring session, navigated within a page using internal anchors or submitted a form repeatedly.
3. In terms of repetition, the average frequency for LRSs of all lengths hovered around 2 which is the minimum requirement to be considered as a LRS.
4. Many paths show a recency effect. That is, the next LRS often follows the previous one very closely. This suggests that the user was likely browsing the same site during the same session, took a slight diversion and then reaccessed the same sequence of pages.
5. Some LRSs were generated by following *Next* and *Previous* hyperlinks that were embedded within a related set of web pages. These sequences show more unique

URLs as compared to many other types of sequences that involve revisiting some URLs. However, capturing such sequences in a history mechanism is also of little use since the user probably wants to visit each page, and an easy method to follow the path already exists through these embedded links.

These results suggest that history mechanisms that present LRSs are of limited use. Like the locality algorithm, the criteria for LRSs were applied according to their strict definition. However, we found that there were many LRSs that have URLs in common but are considered distinct because they do not share exactly the same sequence. Perhaps relaxing the PDM algorithm to count similar as well as exact sequences would increase this value. In other words, the algorithm could allow the occasional navigation to a side trail if the user then continued along a well-travelled path; hypertext, by nature, encourages such explorations but the current algorithm would consider these deviations as distinct.

5. Conditioning the distribution

The recurrence statistics and distributions in the previous section were derived by considering all page visits for a user as one long input stream with no barriers placed between sessions. We saw that 58% of URL navigations are revisits to previous pages. We also saw that a small set of recently visited pages was a good predictor of what a person would revisit next. In contrast, frequency, locality sets and longest repeated sequences are poorer predictors. However, we have seen in Section 4.4 that although this small set of recently visited URLs accounts for a high proportion of revisits, others lie outside. Consider a set of the 10 previous URLs on the history list. From the inset in Figure 7, there is a 42% chance that the next URL the user would like to visit has not appeared before, an $R_{D10} = 43\%$ chance that it has occurred within the last 10 visits, and a 15% chance that it appeared at distances greater than 10. This section explores the possibility that the order and distribution of URLs can be conditioned, first to increase the recurrence probabilities over a set of a given size, and second to evaluate methods for presenting history lists that are currently in use. Eight conditioning methods are presented within four major categories: recency, frequency, stack and hierarchically structured lists. A later results section will consider how effective each method is.

We will illustrate these methods by using the small sample trace in Table 4, which shows the last 16 pages visited by a user. Pages are numbered by order of visit, with #16 being the most recently visited page. The user's actions to navigate to those pages are shown on the right. Italicized pages are revisits. Each conditioning method is then applied to this trace, and the ordering of items that will be shown to the user in the conditioned history list is given in Table 5.

5.1. RECENCY ORDERED HISTORY LISTS

Three types of recency ordered history lists were evaluated. The first is *sequential ordering*, the time-ordered list of all URLs visited by the user, including revisits to the same URL. Thus, the history list as shown in Table 5(a) is a literal trace, and is an exact match to the trace in Table 4.

TABLE 4
A trace of the last 16 pages visited, and the user actions to get them. The top page (16) was just visited

Visit no.	URL	Action
16	acsl.cs.uicu.edu/kaplan/applets	Open URL
15	acsl.cs.uicu.edu/kaplan/worlds-environ	Open URL
14	acsl.cs.uicu.edu/kaplan/worlds	Open hotlist
13	www.acm.org/sigchi/chi96/forms/Proc	Open URL
12	www.acm.org/sigchi/chi96/call/index	Open URL
11	*www.acm.org/sigchi/chi96/*	Open URL
10	*www.acm.org/sigchi/homepage*	StartUp Doc.
9	*www.acm.org/sigchi/homepage*	Back
8	*www.acm.org/sigchi/cscw96/*	Back
7	www.acm.org/sigchi/cscw96/dates	Open URL
6	www.acm.org/sigchi/cscw96/	Open URL
5	*www.acm.org/sigchi/homepage*	Back
4	*www.acm.org/sigchi/chi96/*	Back
3	www.acm.org/sigchi/chi96/Deadlines	Open URL
2	www.acm.org/sigchi/chi96/	Open URL
1	www.acm.org/sigchi/homepage	StartUp Doc.

TABLE 5
Examples of history lists conditioned by different methods. Numbers represent the URLs in Table 4

(a) Sequential ordering by recency
16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1

(b) Recency, duplicates in latest position
16, 15, 14, 13, 12, 11, 10, 8, 7, 3

(c) Recency, duplicates in original position
16, 15, 14, 13, 12, 7, 6, 3, 2, 1

(d) Frequency, second key recency
10, 11, 8, 16, 15, 14, 13, 12, 7, 3

(e) Stack, sessional
16, 15, 14, 13, 12, 11, 10

(f) Stack, persistent
16, 15, 14, 13, 12, 11, 10, 1

(g) Context-sensitive web sub-space
14 (16, 15)
10 (13, 12, 11, 8, 7, 3)

(h) Hyperlink sub-list (session 1 only)

16	12 (13)	7
15 (16)	11 (12, 3)	3
14 (15)	10 (11, 6)	
13	8 (7)	

Greenberg (1993) claims two benefits of recency-based history. First, the URLs presented are the ones the user has just visited. Thus, the user will remember and can effectively predict which URLs will appear on the list. Second, recency does not suffer from the initial start-up instability that other methods do when there are only a few URLs available to present to the user.

The problem is that repeated items have multiple entries, which occupy valuable space on a history list of a limited length. Hence, two strategies for pruning redundant URLs were applied: saving the URL only in its *original position* on the history list and saving the URL only in its *latest position*. Tables 5(b) and (c) provide an example of both approaches to pruning duplicates. Note that there are fewer URLs on these lists as compared to the strict sequential version which retains all URLs. Also, note that the user's StartUp document (a heavily accessed page) occupies the bottom position on the list (#1) when URLs are saved in their original position, while it is propagated up the list when they are saved in the latest position (#10). Thus, we expect the "latest position" approach to perform better, because just revisited URLs will stay at the top of the list (whereas they migrate to the bottom on "original position") and because local context is maintained (Greenberg, 1993).

5.2. FREQUENCY ORDERED HISTORY LISTS

Frequency ordering, where the most revisited page appears at the top of list and the least visited page appears at the bottom, is an obvious way of ranking URLs. However, frequency ordering could be problematic. While user's needs and interests change quickly, the newer URLs need to be revisited frequently before they can migrate to the top of the list. Similarly, older frequently used items that are no longer of interest remain near the top. Still, there are certain types of pages that users tend to frequent regularly (Section 4.5), and perhaps we can expect useful offerings of frequency ordering after periods of extended browsing (which stabilizes the frequency distribution).

An issue associated with frequency ordering is how to break ties with URLs that have the same frequency. Greenberg (1993) evaluated two schemes for *secondary sorting* within frequency-ordered lists: recency and reverse recency. Recency was found to perform better so that is the method of secondary sorting that we have applied. Table 5(d) shows the effect of frequency ordering with secondary sorting by recency upon the navigation session in Table 4.

5.3. STACK-BASED APPROACHES

Current web browsers maintain a history list that operates as a stack. The most recently visited page is usually pushed onto the top of the stack, so older pages appear underneath. Unlike recency, pages can be popped off the stack and lost. The way browsers push and pop pages from the stack depends upon three techniques the user employs for displaying the page: loading, recalling and revisiting (Cockburn & Jones, 1996). In their terminology, *recalling* a page changes the pointer to the currently displayed page in the stack. *Loading* a page causes it to be added to the top of the current position in the stack, possibly resulting in all other pages above the current position to be lost. *Revisiting* a page occurs when the user reloads the page and has no effect upon the stack.

We expect that the stack method will perform reasonably well for very short recurrence distances, as it will appear similar to recency. It will lag at modest distances because some recent URLs are popped off the stack when the user loads a page while at some point other than the stack's top. It will do poorly for long distances because current browsers clear the stack between sessions. Table 5(e) shows the history list based on this *sessional stack* at the end of the example navigation session. Note that the trace in Table 4 shows that a new session started on page #10 (indicated by the start-up document action), so earlier pages are lost.

Because we may want to revisit pages from previous sessions, we constructed a *persistent stack* that retains the stack from the prior sessions. While the persistent stack will perform similar to the sessional stack for short distances, it should do better for long distances because some URLs are retained between sessions. Still, we do not expect this method to perform better than recency ordering with no duplicates due to the absence of some URLs. Also, the persistent stack will be longer than necessary due to the presence of duplicates. Table 5(f) shows this persistent stack; it now includes a pointer to a page from the first session.

5.4. HIERARCHICALLY STRUCTURED HISTORY LISTS

Two methods that employ hierarchical structuring were examined: *recency ordered hyperlink sub-lists* and *context-sensitive web sub-space* history lists.

Recency-ordered hyperlink sub-lists is similar to the recency ordered history list with duplicates saved only in their latest accessed position. The difference is that for each URL on the normal list, a secondary recency-based history list can be raised containing only those pages that the user had visited by selecting a hyperlink from that page. The user first scans down i entries in the normal list for an exact match that terminates the search, or for a page that contains the desired hyperlink. In the latter case, the sub-list of hyperlinks is displayed (perhaps as a cascading menu) and the search continues until an exact match is found j entries later. The distance of a matching recurrence is simply $i + j$. Table 5(h) shows the hyperlink sub-list. The hierarchy in this example is not very full because it is generated from a short trace; we expect that longer traces would fill the slots in the secondary lists.

We expect recency-ordered sub-lists to perform better than recency with duplicates saved in their latest position. First, more URLs are accessible through a hierarchy. Second, if the user needed to visit a page only because it contained a hyperlink to the desired page, that page can now be selected directly from a sub-list. Third, because sub-lists contain only URLs that the user has already accessed from a particular page, the user may find it easier to find a specific URL on the sub-list, especially if the jump-off page is long and contains many irrelevant links.

The *context-sensitive web sub-space* is based upon a graphical history display designed by Cockburn and Jones (1996), illustrated in Figure 12. The display creates a new web "sub-space" each time the user *directly accesses* a page. This page is added as an item in a *webs* menu. Any pages accessed until the next directly accessed page (the sub-space) are added to a secondary menu that cascades from the original *webs* menu entry. For our analysis, we considered the following actions as a *direct access* to a URL (vs. accessing the URL with a hyperlink selection): typing a URL, selecting a Hotlist item, cloning or opening a new window and accessing a URL via client-dependent hard-wired buttons or menus. Within the main and secondary menu, we sort the URLs based on recency with duplicates saved only in their latest position. A URL can thus occur only once in the main menu or a particular secondary menu, but it may be found within several sub-spaces if the user navigated to it in different ways. This is appropriate since

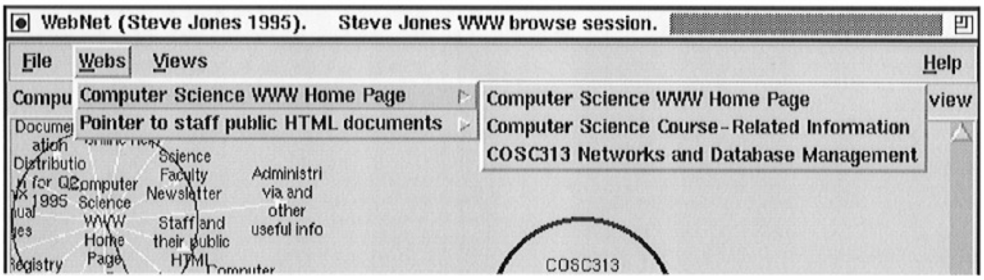

WebNet (Steve Jones 1995). Steve Jones WWW browse session.
File
Webs
Views
Help
Computer Science WWW Home Page
Pointer to staff public HTML documents
Computer Science WWW Home Page
Computer Science Course-Related Information
COSC313 Networks and Database Management
COSC313

FIGURE 12. WebNet's context-sensitive web sub-space, displayed in a menu (Cockburn & Jones, 1996).

sub-spaces seem to be a reasonable method for inferring a user's context when browsing. That is, when the user follows a series of hyperlinks, many of the pages visited will tend to be related in some way.

Table 5(g) shows the web sub-space history list at the end of the navigation session. There are two URLs in the main menu indicating that two different URLs were direct accessed (the *StartUp Document* and the *Open Hotlist*). The last sub-space the user browsed is located at the top of the list. The sub-lists show the contents of the web sub-space sorted in recency order.

5.5. EVALUATION METHODS

We evaluated all methods described above by implementing them as algorithms, and used our subjects' traces to simulate their performance in practice. Our evaluation accounts for the following factors.

1. *Theoretical performance*. We know that a perfect predictor of revisits on average, could not better the recurrence rate $R = 58\%$, which is reached only if the user reuses previous URL visits at every opportunity. Thus, we can see how close our algorithms are to this upper bound.
2. *Presentation*. We asssume predictions will be displayed to users as some kind of visible list, perhaps including a hierarchical sub-list (e.g. cascading menu). However, our calculations work just as well if keyboard and/or graphical equivalents (e.g. the Back and Forward buttons) are used to move through this list. Calculations are made on data structures that mimic these lists.
3. *Number of items*. While showing a history list of the complete trace would give optimum performance, this is not realistic given the large number of items that users would have to scan through. Pragmatically, each list should contain only a small set of previous submissions and offer them to the user as predictions. We consider how well a method performs with different list sizes by calculating predictability as a running sum over distance, where a chosen distance now indicates the maximum number of items scanned in the history list. Figure 13 was created in this way. We have already introduced R_D to indicate the cumulative recurrence rate R at a particular distance D. We will also use R_{D10} (R at a distance of 10) as a comparative benchmark. This is reasonable: as seen in Figure 13, the first

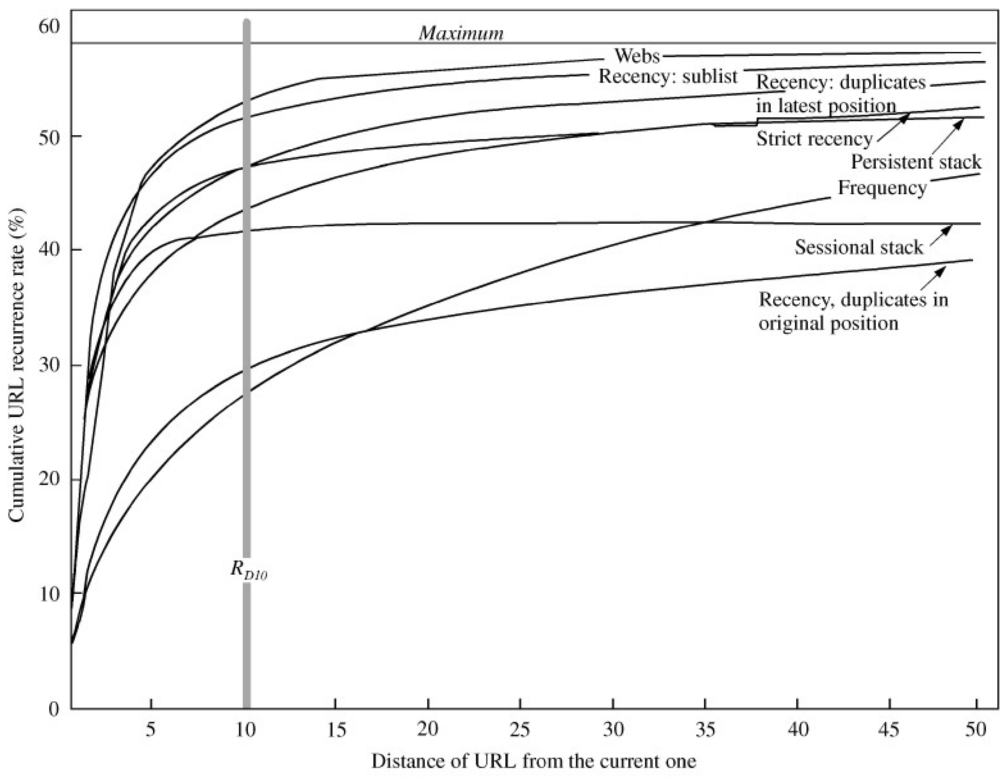


FIGURE 13. Cumulative probabilities of recurrences over distances up to 50.

few items ($D < 10$) of all methods contribute the most to the running sum, and there are diminishing returns for showing lists with $D > 10$.

4. This analysis does *not* account for the conceptual model that the history list presents. Yet this is important, for a person's ability to predict whether the item they seek is present on the list is generated from their conceptual model. Consequently, we will discuss (but not measure) the simplicity of the method's conceptual model. A possible approach for qualitatively assessing the cognitive load of the conceptual models required by different history facilities is cognitive modelling (May, Barnard & Blandford, 1993). However, such work goes beyond the scope of this paper.
5. Additionally, our analysis does *not* compare a user's cognitive and physical effort involved in choosing items from each history method. For example, selecting items from a hierarchical sub-list would need more effort than choosing from a sequential list, because a user must decide what branches to follow, raise cascading menus and so on. Recognizing item names in the list also has a cost. If something akin to a Back button is used, the cost would now include waiting for each page to appear and recognizing whether it is the one they want. There is probably a tradeoff between predictability (especially with complex methods) and effort, and future research should measure this as well.

5.6. RESULTS

All results are plotted in Figure 13. A strict sequential list of URLs ordered by recency performs reasonably well when a small set of URLs are listed, e.g. $R_{D10} = 43\%$. A benefit of this method is that its conceptual model is simple and familiar. That is, a person knows what they have just done and can thus predict if an item will be on the history list. We will use this value as a benchmark to contrast other methods.

Pruning duplicates is a simple way of improving the performance of a recency-ordered list when duplicates are saved only in their latest position ($R_{D10} = 47$ vs. 43% for strict recency). While this type of list does not show the exact sequence of URLs visited by the user, it still presents a clear conceptual model, and we expect that the user can easily understand how the list contents are ordered. Saving duplicates in only their original position is very poor ($R_{D10} = 29\%$), as frequently used items remain at the bottom. The striking differences between these three recency orders are illustrated by the plots in Figure 13.

Frequency ordering is the worst predictor of the eight evaluated for short lists, with $R_{D10} = 27$ vs. 43%. While it does improve as distance is increased, it does not catch up to strict recency (Figure 13). Frequency has other problems, as already mentioned. Users may find it more difficult to predict which pages would appear on a frequency-ordered list beyond the two or three that they visit the most. Frequency ordering suffers instability as well, when few items are on the history list, and excessive inertia when the list is long. Still, frequency could be applied to a few key URLs, possibly used as an auxiliary method in conjunction with another history mechanism that gives better overall performance.

The sessional stack method found in most web browsers is slightly better than strict recency at very short distances ($R_{D5} = 40$ vs. 37%). and slightly worse at R_{D10} (42 vs. 43%). As seen in Figure 13, it is much worse as the list gets long. The persistent stack is an improvement over the stack and strict recency methods in terms of its recurrence probabilities over distance ($R_{D10} = 47$ vs. 43%). Both approaches suffer problems as users typically form an incorrect conceptual model with this method—Cockburn and Jones (1996) discovered in their usability study that users were often surprised at the behaviour of their history list, and they could not predict how it worked. Because other methods are equal or better than even the persistent stack in terms of predictability (such as recency with duplicates removed), they are better choices.

Recency-ordered hyperlink sub-lists have the highest recurrence probability over all methods for very short distances (2–4), and are second best at modest distances ($R_{D10} = 51$ vs. 43%). The catch is that this result is optimistic, since a person requires greater cognitive and physical effort to select items from the hyperlink sub-lists, e.g. to make an accurate selection from a hyperlink sub-list, the user must recall which main list item contains the desired URL. Also, note that hyperlink sub-lists can provide access to 55 URLs for a R_{D10} (the main list of 10 items + 9 items on sub-list 1 + 8 items on sub-list 2 + 7 items on sub-list 3, etc.).

The best method we evaluated—context-sensitive web sub-spaces—showed that 53% of all URL selections can be successfully predicted with a set of 10 items (Figure 13). Given that $R = 58\%$, on average, which is the best a perfect reuse facility could achieve, this method is $\sim 91\%$ effective. The caveat is similar to hyperlink sub-lists, as users of context-sensitive web sub-spaces require greater physical effort to select a sub-list item, and greater cognitive effort to recall which sub-list might contain the URL. In addition, users may have more difficulty in understanding how this method works, as they need to know what a "direct access URL" is to grasp the way the history list is organized.

In conclusion, our analysis of conditioning methods shows that several methods improve upon the effectiveness of current stack-based history mechanisms. As seen in Figure 13, recency is a simple yet reasonable predictor, especially when duplicates are saved only in their latest position. A further appeal of recency is that it is conceptually easy for a user to understand. While the two hierarchical methods are better predictors, we suspect that they may not work as well due to the extra physical and cognitive overheads mentioned earlier. Further research is required to evaluate how useful these methods are in practice.

6. Discussion

6.1. HIGHLIGHTS AND GENERALIZATIONS

Based on these results we formulate some empirically based generalizations of how users revisit pages using features found in a typical web browser such as Mosaic.

1. Users revisit a considerable number of web pages. Our analysis of recurrence rate shows that there is a 58% probability that the next page visited was previously seen. This qualifies web browsing as a recurrent system.
2. While many pages are revisited, users continually incorporate new pages into their repertoire at a regular rate. There are also local variations in the vocabulary growth rate and use of navigation activities across users and across their browsing timeline. These variations indicate the presence of different browsing activities.
3. Users visit very few web pages frequently. Consequently, many web pages are only visited once (60%) or twice (19%). The few frequently accessed pages tend to fall into certain definable categories.
4. Users exhibit considerable recency of revisits. The major contributions to the recurrence distribution are provided by the last few pages visited, which also explains why "Back" is frequently used (30% of all navigation actions).
5. Users revisit pages that have not been accessed recently. For example, 15% of recurrences are not covered by a list of the last 10 URLs visited. Still, doubling or tripling the size of the list does not increase its coverage much.
6. Users do not have strongly repeatable linear patterns when browsing selected clusters of pages. Both locality sets and longest repeated sequences are small, rarely repeated and also exhibit recency.
7. Methods to present a history list of previously visited pages are available that are more predictive and usable than the current stack-based approach. Presenting the last 10 or so recent URLs, with duplicates saved only in the latest position, surpasses current stack-based approaches and are likely much more usable. Other methods fare even better, although their usability must be determined.
8. The Back button is very effective and simple to use. Perhaps it could be improved further by basing it on a recency rather than a stack model.

9. The poor use of history facilities is likely due to several interface problems. First, hotlists in Mosaic and many other contemporary browsers require considerable effort to manage. Consequently, users may not bother to add an URL to the list, may forget that it is there or only record URLs that are convenient starting points. Second, The "Window History" in Mosaic 2.6 is not visible and requires several actions to access. (While Netscape 3.0 does allow a history window to remain in view, it must be raised through an explicit menu action for every session.) Third, history in Mosaic, Netscape and other browsers are based on the stack model, which means that the desired URL may have been popped off the list even though it was entered a short time ago. We expect these problems will be mitigated in future systems by both interface redesign (perhaps by making the last few items always visible on the display) and by taking advantage of new browser features (e.g. using frames to include hotlists as a fundamental part of the display). However, browser redesign will require careful consideration and evaluation, which is beyond the scope of this paper.

6.2. DESIGN GUIDELINES

This section proposes eight guidelines for the design of history mechanisms in web browsers. They are adapted from those previously formulated by the second author (Greenberg, 1993) for the design of reuse facilities within other domains, and from our own observations and empirical results described in the previous sections.

Maintain records of URLs visited and allow users to recall previous URLs from those records. Our study shows that though users incorporate new URLs into their repertoire at a regular rate, 58% of web pages are revisited. Web navigation is thus classified as a recurrent system. Hence, a history mechanism has value, and as a first requirement, it must record the URLs that users visit. To obtain the maximum benefit from this data, users must be able to access the URLs during their current as well as later sessions. However, web browsers fail to maintain adequate records of URLs visited, as most only supply users with sessional-based history. While they do maintain a persistent history list for internal purposes, browsers do not give users access to the history data; it is only used to indicate which hyperlinks have been recently accessed. Even within sessions, the history list may discard some of the URLs visited because of its stack-based model. Some add-on softwares attempt to remedy these problems. For example, the Overdrive Logger records all URLs visited for 1 year in individual files for each day. While an improvement over Netscape Navigator 2.0's capabilities, it is not easy for the user to integrate history data from the multiple files.

It should be cheaper, in terms of physical and cognitive activity, for users to recall URLs from a history mechanism than to navigate to them via other methods. The prime motivation for providing a history system is to reduce the physical and/or cognitive effort of returning to a particular web page. The heavy use of the Back button (30%) indicates that it is an effective way for the user to reach the last few items, as long as page redisplay is quick. In contrast, users select URLs from the history dialog less than 1% of the time, likely due to various physical and cognitive overheads involved. We believe that several factors in history mechanism design must be met in order to reduce the overhead in returning to a web page.

- The history system should attempt to predict the user's next URL selection. If it does so effectively, the user is more likely to access the history system to retrieve the URL vs. navigating to it via other methods.
- The best predictions should be clearly distinguishable so that they are the first ones that the user sees. For example, the most likely history items could be placed at the top of the list or they could be presented as top-level buttons. Back works reasonably well because of this. In contrast, history lists in most browsers are usually invisible until explicit action is taken and are often ignored or forgotten by users.
- The order and inclusions of predictions should be understandable, so that users will have some degree of confidence that the item they want is on the list. In contrast, the stack method is difficult to comprehend, especially when pages disappear mysteriously (Cockburn & Jones, 1996).
- A minimum number of physical actions should be required to access and retrieve an item from the history system. Many current systems require fairly heavyweight access mechanisms, such as menu navigation and explicit actions to raise the history list.
- The history mechanism should provide some clues as to the structure of the web space and the pages previously visited to help the user regain context and orient themselves.
- Accessing the history mechanism should be minimally disruptive to the user's current task.

A selectable history list of the previous 6–10 *URLs visited provides a reasonable set of candidates for reuse.* Greenberg (1993) concludes that a lengthy history list is unlikely to be worthwhile considering the high cost of real estate on even large screens and the user's cognitive overhead of scanning the possibilities. Similarly, a list that is too short can exclude potentially valuable predictions. We believe that a reasonable list length is about 6–10 items. Our results show that a menu of the previous 10 URLs visited covers, on average, 43% of all inputs. This is fairly good, considering that the best a history list could do is 58%, the recurrence rate. Doubling the length of the menu to 20 items only increases the probability by 5%. However, the list could be even shorter than 10 URLs since the items that contribute most to the probability of a recurrence are at a distance of 1, 2, 4, 6 and 3 (10, 19, 5, 2 and 2%, respectively). Another benefit of presenting the most recent URLs is that the user will likely be able to predict if the URL they seek will appear on the list; if they recently entered it, it should be there. Yet current browsers have *ad hoc* history display lengths. Depending on the browser, the history list grows indefinitely, has an arbitrary cut-off or offers a customizable length. Further, the stack model used by web browsers does not fit into either of these categories. Items are removed when the user loads a page while at some point other than the top of the stack. Depending on the type of browsing the user is engaged in, the length of the stack varies, and it may grow to be very lengthy.

Other strategies for presenting the history list, particularly pruning duplicates and hierarchical structuring, increase the probability of it containing the next URL. A significant number of URLs to be revisited are not covered by the last 10 pages visited (26% of the recurring total). We have already mentioned that doubling or tripling the size of the list does not increase its coverage much (see Figure 13). But it is these missed URLs that could help the user most since they occurred long ago and are, thus, more difficult to recall and/or locate. This is why we explored alternative methods for conditioning the history list in Section 5. Pruning duplicates from a recency-ordered list is a simple improvement, as it makes room for additional URLs in a list of a fixed size (47 vs. 43% for strict recency for a 10-item list).

Also, hierarchical structuring is a way of bringing distant nodes closer by offering branching points. We have already presented two menu-based methods to do so: context-sensitive sub-lists and web sub-spaces (51 and 53%, respectively, vs. 43% for strict recency for a 10-item list). Graphical representations of branching structures also incorporate hierarchical structuring, as seen in the WebNet of Cockburn and Jones' (1996), the Graphic History View in MosaicG (Section 2.6) of Ayers and Stasko (1995) and the DeckScape (Section 2.9) of Brown and Shillner (1995). All these systems incorporate features to manage the hierarchy/network when the number of nodes become large: fisheye views, pruning and branch collapsing, deck metaphors. Of course, these are just prototypes, and we would expect further interface refinements before their full value can be reached.

History based on recency is not effective for all possible recalls because it lists only a few previous events. Alternative strategies must be supported. Recency was a strong reuse pattern but we found that other patterns exist. For example, a few key pages are accessed with a high frequency. One of these, the user's home page, is easily accessible by the option of it being both the start-up document and reachable through the Home button on the browser toolbar. Other frequently accessed pages could be made available on a toolbar for easy access. A drawback of frequency ordering is that it has a certain degree of non-intuitiveness. That is, during post-study interviews, subjects were sometimes surprised to see certain URLs on their 15 most frequent URLs list. Greenberg (1993) suggests that combining a recency-based short-term memory with a frequency-based long-term memory could generate better predictions. For example, the browser could show the most recent URLs, as well as the top three most frequently visited pages. Two other alternative strategies are worth mentioning. First, identifying and presenting paths to the user may be useful, though additional research is required to improve path detection within the web domain. Second, for infrequently accessed URLs that have not been visited recently, the ability to perform a parameterized text search on one's history could be beneficial.

Consider current browsers within this context. Most of the alternative strategies for recalling distant pages are provided (if at all) through helper applications or add-on software that are not well-integrated with the browser. At the simplest, we have to go beyond session boundaries (Section 2.7). For example, Navinet Inc.'s Overdrive Logger provides access to inter-sessional history through browsable lists of URLs. Its flaw is that the user must recall which day they visited the site to access the appropriate history file. Apple's Internet Explorer does much better with its collection of all visited items in its Log window. Searching the list with text patterns (Section 2.5) is possible in ISYS HindSite. While Netscape Navigator 2.0 also incorporates a search feature, it only searches the user's bookmarks list and not the internal global history list.

History items should have a meaningful representation. When pages are presented as items in a menu, they must be identified by a name or symbol. Yet finding appropriate symbols for pages is an issue. URLs are usually poor identifiers as their names are rarely meaningful. Page titles, identified by a HTML tag within a page, may not exist, be too long to display, or may be a poor match of the user's understanding of the page content. Current systems take several approaches to this problem. Cockburn and Jones (1996) are now investigating different methods of displaying page titles in WebNet. Their major issue concerns displaying titles that can be read when there are a large number to display. Ayers and Stasko's Graphic History View in MosaicG (Figure 3) allows users to display web pages by their URL, page title or a thumbnail, where the user can control the abbreviation of page titles. However, we do not know how many pages are distinct enough to be easily recognized from a small thumbnail, nor if users are willing to sacrifice their screen space for displaying many of them. Of course, the best representation could be the page itself. The Back button, e.g. redisplays previous pages within the browser at full size. Because recently visited pages are cached locally, they appear almost instantaneously.

Support grouping of URLs into high-level web tasks, and switching between tasks. Locality did not prove to be prevalent enough to identify common tasks. Still, we believe that it would be useful to allow items on the history list to be grouped in a way that reflects a user's task. Several experimental web systems meet this guideline to various degrees. First, WebNet's web sub-spaces, illustrated in Figure 12, appears to be an intuitive way of partitioning a user's navigation history (Cockburn & Jones, 1996). While this technique proved to be the best conditioning method of the eight examined, its usability remains to be assessed. Second, DeckScape's deck metaphor (Section 2.9) and its impressive implementation make it a very powerful tool for allowing a user to organize and switch between groups of web pages (Brown & Shillner, 1995). Finally, the WebBook (Section 2.9) offers a very appealing book metaphor to gather, collect, store and switch between sets of pages (Card, Robertson & York, 1996).

Allow end-user customization of history data. The seam between history and hotlists should be eliminated. The idea is that the history list could offer sets of candidates, and that users can decide at any time which of those deserve greater emphasis and saving for posterity. For example, the Graphic History View of Ayers and Stasko allows the user to zoom into certain portions of the hierarchy and condense branches of the tree. WebNet allows both manual and automatic creation of web sub-spaces that can be redisplayed independently. Overdrive's Organizer module permits the user to filter, sort, organize and save portions of their browsing history. The DeckScape browser allows the user to pull pages away with a click-and-drag operation, permitting easy creation and modification of various decks according to the user's information needs. DeckScape also contains a special *HotList* deck to which the user can add a page by selecting the "copy to hotlist" button present on each document (Brown & Shillner, 1995). Finally, decks can be moved, iconified or resized by the user.

7. Conclusions

This paper provides empirical data that justifies the need for suitable history mechanisms in graphical web browsers. Our analysis of different designs strongly suggests that the predictability of URLs presented by current stack-based models of history can be improved upon. Using the methodology and design guidelines herein, designers can validate and refine current history mechanisms and investigate new approaches.

There are still many unanswered questions. We need to evaluate the physical and cognitive effort for reviewing a particular conditioned set of history list predictions. We need to validate the design guidelines that we have proposed. We also need to assess how usage patterns change along with future browser interfaces (such as re-designed history mechanisms) and HTML extensions (e.g. *frames* and *Java*). We do suspect that some of the numbers reported here would not change dramatically, such as the high recurrence rate and the recency effect. In contrast, the numbers reflecting low usage of history mechanisms should change (hopefully) as the browser interface is improved.

Many thanks to L. Catledge and J. Pitkow, who graciously provided us with their data logs for comparative purposes, and who gave us clues on how to instrument Mosaic by providing us with their version of it. The 28 people who volunteered to participate in the study made this research possible. Discussions with Andy Cockburn helped expand our research. This research was partially funded by Canada's National Sciences and Engineering Research Council.

References

AYERS, E. & STASKO, J. (1995). Using graphic history in browsing the World Wide Web. In *Proceedings of the 4th International World Wide Web Conference*, Boston, MA. http://www.w3.org/pub/Conferences/WWW4/Papers2/270/.

BROWN, M. & SHILLNER, R. (1995). DeckScape: an experimental web browser. In *Proceedings of the 3rd International World Wide Web Conference*, Darmstadt, Germany. http://www.igd.fhg.de/www/www95/papers/90/deckscape-final-v1/paper.html.

BUSH, V. (1945). As we may think. *Atlantic Monthly*, **176,** 101–108.

CARD, S., ROBERTSON, G. & YORK, W. (1996). The WebBook and the Web Forager: an information workspace for the World-Wide Web. In *Proceedings of the ACM CHI '96 Conference on Human Factors in Computing Systems*, Vancouver, British Columbia, Canada.

CATLEDGE, L. & PITKOW, J. (1995). Characterizing browsing strategies in the World Wide Web. In *Proceedings of the 3rd International World Wide Web Conference*, Darmstadt, Germany. http://www.igd.fhg.de/www/www95/papers.

COCKBURN, A. & JONES, S. (1996). Which way now? Analysing and easing inadequacies in Web navigation. *International Journal of Human Computer Studies*, **45**(1), 105–129.

CROW, D. & SMITH, B. (1992). DB_Habits: comparing minimal knowledge and knowledge-based approaches to pattern recognition in the domain of user-computer interactions. In R. BEALE & J. FINLAY, Eds. *Neural Networks and Pattern Recognition in Human–Computer Interaction*, pp. 39–61. Chichester: Ellis Horwood.

GREENBERG, S. (1993). *The Computer User as Toolsmith: The Use, Reuse, and Organization of Computer-based Tools*. Cambridge, MA: Cambridge University Press.

HENDERSON Jr, D. A. & CARD, S. (1986). Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interfaces. *ACM Transactions on Graphics*, **5,** 211–241.

ISYS HindSite for Netscape Navigator. *http://www.isysdev.com/hindsite.html.*

LEE, A. (1992). *Investigations into history tools for user support*. Ph.D. Thesis, Department of Computer Science, University of Toronto, Ontario, Canada.

MADISON, A. & BATSON, A. (1976). Characteristics of program localities. *Communications of the ACM*, **19,** 285–294.

MAY, J., BARNARD, P. J. & BLANDFORD, A. (1993). Using structural descriptions of interfaces to automate the modelling of user cognition. *User Modelling and User-Adapted Interaction*, **3,** 27–64.

Navinet, Inc. (1996). Overdrive™. *http://www.nvnt.com.*

Netscape Communications Inc. (1996). Netscape navigator. *http://home.netscape.com/comprod/products/navigator/index.html.*

TAUSCHER, L. (1996). *Evaluating history mechanisms: an empirical study of reuse patterns in World Wide Web navigation.* MSc Thesis, Department of Computer Science, University of Calgary, Alberta, Canada. http://www.cpsc.ucalgary.ca/grouplab/papers/.

TAUSCHER, L. & GREENBERG, S. (1997). Revisitation patterns in World Wide Web navigation. *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems*, Atlanta, GA, pp. 399–406. New York: ACM Press.