THE UNIVERSITY OF CALGARY

Evaluating History Mechanisms: An Empirical Study of Reuse Patterns in World Wide Web Navigation

by

Linda Marie Tauscher

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

JUNE, 1996

Approval Page

THE UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled “Evaluating History Mechanisms: An Empirical Study of Reuse Patterns in World Wide Web Navigation” submitted by Linda Marie Tauscher in partial fulfillment of the requirements for the degree of Master of Science.

_____

Supervisor, Dr. Saul Greenberg, Computer Science

_____

Dr. Brian Gaines, Computer Science

_____

Dr. Jeff Caird, Department of Psychology

_____

Date

Abstract

History mechanisms in user interfaces allow users to select and redo one of their previous activities, ostensibly reducing the cognitive and physical overhead that would have been required to specify them from scratch. Recently, history mechanisms have been incorporated into World Wide Web (WWW) browsers as navigation aids. Yet how effective are these Web-based history systems? Are they needed? Can they be improved?

The hypothesis of this research is that users revisit WWW pages, and that an examination of individual's WWW navigation patterns can provide insight into the design of history systems. Data was collected from 23 subjects who used an instrumented version of XMosaic 2.6 for 6 weeks. 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. They access only a few pages frequently, revisit recently visited pages, browse in very small clusters of related pages, and generate short sequences of repeated URL visits.

A further analysis of *conditioning methods* for history lists indicates that the stack-based method found in many commercial browsers shows only modest effectiveness, whereas a simpler approach that offers the ten or so recently visited URLs offers better predictiveness. Other approaches fare even better. Based on empirical evidence, nine design guidelines for WWW browser history mechanisms are then formulated. When used to evaluate existing history mechanisms, it is clear that today's browsers are not as effective as they could be.

Acknowledgements

I owe thanks to many individuals who offered support in various ways during my research and writing of this thesis.

First, I am deeply indebted to my fiancé, Tom Malaher, who was always available to answer my Unix questions, listen to my ideas, and offer encouragement. Tom also contributed in many other ways, and accepted the sacrifices that doing this work entailed.

My thesis supervisor, Saul Greenberg, worked closely with me during the construction of this thesis. His brilliant insights, professional editing skills, and knowledge of the domain enabled me to craft this solid bit of research. More importantly, Saul always saw the value of this work, and motivated me to continue during my periods of self-doubt.

Several of my colleagues read my thesis and offered advice during my mock defense. Thanks to: Andy, Mark, Suchitra, Carl, Doug, Shannon, Ruby, Husam, Darren, Tom, Lee, and Earle.

Twenty-eight individuals participated in my study; I thank them for their support of my research, and the benefits their data will bring to the broader research community.

For my graduate course work, I have been very fortunate to learn from outstanding researchers in the area of computer science, human-computer interaction, and/or human factors: Saul Greenberg, Jeff Caird, and Bruce MacDonald. There is definitely a small part of each of them in the way that I think about and carry out my work.

The women at the Alberta Research Council demonstrated to me that women do indeed excel in computer science and research careers, and that they can lead interesting and full lives at the same time. My gratitude goes to: Ruby, Marlene, Shelli, Julia, Lynn, Janet, Sheila, and Meg.

Finally, I must thank my parents, Anna and Frank Tauscher, who chose to immigrate to this great country—a place where I could afford to finance my nine years of post-secondary education on my own.

Dedication

For Anna

Table of Contents

List of Tables

List of Figures

1. Introduction

The World Wide Web (WWW) hypertext system is a large, distributed repository of information. People use graphical browsers to navigate through links and to view pages. Within these browsers, history mechanisms allow people to revisit pages they have viewed previously. However, the design of currently available history mechanisms tend toward ad-hoc approaches that do not appear to take advantage of previous research into history support within user interfaces. These mechanisms are not based upon actual studies of how people revisit Web pages, and their actual use has been examined only superficially.

The purpose of the research described in this thesis is to place the design of history systems on a more empirical footing. By doing so, we can perhaps identify shortcomings in current approaches, validate successful solutions, and suggest new approaches.

Why history? We believe that improved history support within browsers can reduce the impact of three major problems in navigating the WWW: the quality and volume of Internet data, resource limitations, and the cognitive and physical burdens in using hypertext. For example, an effective history mechanism can assist the user in dealing with the vast amounts and poor structure of information by providing easy access to information previously visited. Because they automatically capture the user's page visits, history facilities can reduce the use of search engines. They can also eliminate navigation to intermediary pages that are navigated en-route to the page of interest. Finally, history mechanisms can show users where they have been, and help to situate them within the current context of the information space.

We begin this chapter by defining two key aspects of the problem domain: hypertext, and the World Wide Web. Next, we examine some of the characteristics of the WWW that motivate the study of history mechanisms within it. We then briefly review previous research into history within user interfaces, and state how our domain differs from those covered in previous studies. Finally, we state the problem this research addresses, and give an outline of the thesis.

1.1 Hypertext and the World Wide Web

Hypertext is a method of managing online information that uses a non-linear text model. Although there is no generally accepted definition for hypertext, most hypertext systems can be characterized by five main features (Akscyn, McCracken, and Yoder, 1988).

1. Information is divided into small units, often called *nodes*. These units can contain text, graphics, audio and video. One node is usually displayed per window.
2. Nodes are interconnected by *links*.
3. Users *navigate* in a hypertext database by selecting links in order to travel from node to node.
4. Users build information structures by creating nodes and links.
5. Hypertext databases may be *shared* and also *distributed*. Multiple users can access information located on different computer systems.

The largest distributed hypertext system is the World Wide Web, developed at the European Particle Physics Laboratory (CERN). In 1990, developers at CERN spread word of the Web's existence to the academic community, and it was introduced to the public in 1991. Use of the WWW by both academics and the general public increased dramatically in May 1993 after the National Center for Supercomputing Applications released a graphical interface (or browser) to the WWW called Mosaic. The Web has been growing explosively—its current growth rate is estimated at 20% per month—and it has become the so-called "killer application" of the decade.

The WWW supports two separate information discovery paradigms: hypertext links and indices. First, hypertext links allow users to access or *browse* WWW pages (nodes) without having to know how these pages are distributed (Berners-Lee, Cailliau, Groff, and Pollermann, 1992). Second, both hypertext links and the contents of Web pages are amenable to indexing. Various search engines or databases now exist on the WWW that index Web sites, and in response to a list of keywords, the search engine will return a Web page of hypertext links or indices. This search-and-browse paradigm that Berners-Lee, et.al. envisioned is a familiar method of navigating the WWW to all of its seasoned users.

Several features of the WWW make it a very powerful Internet exploration tool. First, information need only be stored once, since links can be made to its source and the page recalled without copying. Next, hypertext links allow the topology of the information to evolve, and support a structure that stretches from one's personal workstation to large databases on other continents. Third, indexes are interpreted as documents and can thus be found via searching. Finally, Web documents are not restricted to being static files; they can be "virtual", representing real-time views of changing data (Berners-Lee, Cailliau, Groff, and Pollermann, 1992).

1.2 Problems in using the World Wide Web

Though the WWW is a very powerful Internet exploration tool, there are several problems that reduce its effectiveness in providing people with easy access to information: the characteristics of Internet data, information overload, resource limitations, and the cognitive and physical burdens of hypertext navigation. This thesis will argue later that effective browser mechanisms for permitting users to easily return to pages can mitigate these problems.

1.2.1 Characteristics of Internet data

Users of the Internet experience difficulty in locating and accessing information due to five characteristics of Internet data.

1. The data is *diverse* in that it can exist in a variety of formats, be compressed with a variety of mechanisms that are platform dependent, and require special types of software to access. Fortunately, data formats are becoming transparent to the user due to the increased capabilities of WWW browsers, the evolving HTML standard, and the de-facto standards of typical WWW page formats.
2. Internet data is highly *decentralized* which can make locating information difficult. Centralized indexes provide a satisfactory solution to the problem but they do not scale well. For example, the popular Archie database was replicated to facilitate ease of access, and will soon be partitioned because the number of files it indexes is growing so large. WWW databases of page titles and contents are so large that the

software “robots” that traverse the WWW to update these databases cannot be run on a frequent basis.

3. The data is largely *unstructured* and *fragmented* since standards do not exist for its form, identification, and classification.
4. Much of the data is provided and updated by many individuals in the Internet community, and its quality and currency varies.
5. Finally, the volume of data on the Internet is growing at a rapid rate, pushing the limits of both hyperlinks and of the usefulness of search results.

1.2.2 Information overload

The diversity, decentralization, unstructured nature, fragmentation, and rapid growth of the Internet has several consequences. The vast quantity of information leads to *information overload*, a situation where the user can no longer comprehend the information because of its volume (Keyes, Sykes, and Lewis, 1989). December (1994) divides information overload into two categories: information pollution, and information saturation. Information pollution arises due to the amount of data that is redundant, erroneous, and of generally poor quality. This information obscures the information of high value that the user is interested in. Information saturation occurs when the user cannot compare the value of available information sources on a particular topic because the information space has grown too large.

1.2.3 Resource limitations

Resource limitations related to the Internet take two forms: physical resource constraints, and scalability issues.

WWW browsing suffers due to many network and server related resource constraints. Bandwidth is a limited resource and more demands are being placed on it by the increasing number of users connecting to the Internet, and by the increasing amount of data (especially image, video and audio) that is being transferred. Bandwidth impacts response times which means users must wait longer for pages to download. The vast number of users means that servers may be inaccessible if their ceiling of users has been reached.

Fortunately, design changes in Internet software are alleviating some of these physical limitations. The latest Web browsers contain a variety of performance enhancing features including the ability to: cache Web pages, abort the transfer of a page while retaining the portion downloaded, activate a link before the page is fully downloaded, defer loading of graphics, and progressively render images (Berghel, 1996).

Scaleability issues are a major problem on the Internet and also manifest themselves in Web browser navigation aids. Hotlists and bookmarks do not scale well; beyond 50 to 100 items, the lists become unmanageable and awkward to use (Berghel, 1996). They are also arduous to create, organize, and maintain over time. The current solution is to allow the user to collect Uniform Resource Locators (URLs) into multiple hotlists or bookmark folders that can be nested in a hierarchical fashion, though this requires considerable effort on the part of the user.

1.2.4 Cognitive and physical burdens of hypertext navigation

The promise of hypertext is the freedom that it offers the reader in accessing information according to their needs and personal preferences. However, hypertext tends to lack a knowable structure, and this can place additional cognitive and physical demands on the reader as well as cause them to feel disoriented.

Cognitive overhead arises due to the additional effort and concentration necessary to maintain several tasks or trails through the network at one time especially in large, unfamiliar hypertexts.(Conklin, 1987). For example, a user reading the author's CHI96 *HCI and the Web* workshop position paper may notice a hyperlink to the home page of Andy Cockburn, a researcher whose work is referenced. The user must decide whether following the link is worth the distraction. If they activate the link, they must want for the Web page to download. While viewing the home page, the user may decide to explore Cockburn's page of recent publications. The user discovers a paper about navigational problems in the WWW, and begins to read that. Then they decide to follow a link from this paper to Catledge and Pitkow's (1995) study. This navigation sequence has generated four tasks or trails the user is currently engaged with: the CHI96 paper, Cockburn's list of papers (and possibly home page), Cockburn's paper, and Catledge and Pitkow's paper.

The user must recall the essence of each Web page and what they have read thus far as they divide their attention amongst the different *trails* they are currently pursuing.

Disorientation within hypertext arises from not knowing where you are in the network, and how to get to some other place (Conklin, 1987). This can also be a problem with linear text but is exacerbated in hypertext— its modular nature creates many more dimensions in which the user can move. As was discovered in the Intermedia system, information can easily become hard to find or forgotten altogether even in a moderate size network of 1000 nodes (Utting and Yankelovich, 1989). In the Web navigation example described above, the user has visited five different pages on three different Web sites. While reading the Catledge and Pitkow (C & P) paper, they may recall that the author's position paper also referenced the C & P paper and decide to return to position paper page. Doing so requires the user to reconstruct their initial navigation sequence, and how to backtrack to the first page. If they cannot recall the path they took, they will have lost their sense of location and direction.

Hypertext navigation can also place physical burdens on the user. The modular nature of hypertext may require additional physical effort to navigate via selecting links, and activating Back or Forward actions. Also, when disoriented, the user will likely perform additional navigation actions to locate their node of interest.

1.2.5 History and the WWW

We believe that the four problems in using the WWW described above can be mitigated through improved history mechanisms. For example, by capturing a user's navigation history, and providing access to it, the user will spend less time attempting to relocate that information in the future. This has three implications:

- the information of interest is easier to locate and access since the user can go to their personal history to find it, rather than venturing forth onto the broader Internet;
- information overload can be alleviated since the user will visit fewer pages overall, including fewer pages of low value;
- cognitive and physical overhead are reduced since the user will know where they have been, and does not need to expend additional effort locating a previously visited site.

In terms of resource limitations, more effective history mechanisms could improve response time by presenting the user with a list of best candidates—that is, URLs they are most likely to revisit. If the predictive goodness of the history list was high, the user could navigate directly to the site, rather than follow a path of links that they know will get them there, or query a search engine. Reducing the navigation to unnecessary pages will improve network utilization for all Internet users.

A second type of resource limitation discussed earlier is scalability. While managing hotlists requires considerable effort on the part of the user, history logs reduce this effort by automatically recording the Web pages visited. History in browsers currently takes two approaches to scalability: a recency-based sessional history list that operates like a stack, and an expiry date mechanism that changes the colour of a hyperlink to indicate a previous visit for a certain time period. However, these two approaches do not address all of the user's navigation needs.

1.3 History within User Interfaces

The problems described in Section 1.2 imply that additional interface support be provided to reduce users' cognitive and physical demands when navigating the WWW. History, and tools based on history, have been researched and applied to user support concerns in the past. This section provides an overview of two major taxonomies that characterize the uses of current history tools, and the types of history mechanisms that exist. In particular, our intent is to situate WWW navigation within these taxonomies.

1.3.1 Uses of information in a history

Repetition in user activities exists in both the computing and non-computing realms. For example, Greenberg (1993a) notes that people replay their favourite music, favour a subset of recipes and tools, and follow similar procedures to accomplish routine office tasks. Within computing environments, Hanson, Kraut, and Farber (1984), and Greenberg and Witten (1988) report that a few commands issued during a user session comprise a large percentage of the total commands used.

According to Lee (1992), recurrent activities arise for several reasons. Some tasks are routine and frequent or inherently repetitive. For example, by its very nature, problem solving through trial and error is a repetitious process. Also, a number of tasks may be performed using the same actions because less effort is required or poorly designed systems force the user to do so. Lastly, users are fallible and thus make errors due to formulating the wrong intention or executing an inappropriate action for an intention; hence, repeated attempts at specifying a task may be required.

Given that many computer-based tasks involve repetition, there is an opportunity to collect a record of user's interactions—called a *user history*—and make it available for future use (Lee, 1992). The appropriate use of user history can potentially enhance user-computer interactions.

Lee (1992) identifies seven basic uses of the information in a user's history: reuse, inter-referential input/output, error recovery, navigation, reminding, user modelling, and user interface adaptation. This thesis looks exclusively at the following three uses.

1. *History for reuse* allows a user to recall and optionally modify a history item to reduce cognitive and physical effort. A reuse facility must therefore capture or collect a history trace of user actions, and identify patterns in that trace to discern the history item the user is likely to select next. Most research into history has addressed history for reuse, particularly within the context of command line systems such as Unix.
2. *History for navigation* assists users in understanding where they are, where they just came from, and where they have been. History for navigation is receiving more attention due to the widespread research into and use of hypertext-based systems where navigation is a key activity.
3. *History for reminding* gives a person knowledge of past events. History for reminding is the focus of *memory aids*—devices designed to record and retrieve information about personal activities, etc. This use of history is implicit in the other two types of history we investigate for this thesis. That is, the visualization of the history data collected serves as a reminder to assist users in navigating back to previous Web pages.

1.3.2 Greenberg's taxonomy of history mechanisms

Greenberg (1993a) distinguishes between three kinds of reuse facilities: adaptive systems, programming by example, and history mechanisms. *Adaptive systems* use dynamic models of previous inputs to predict subsequent ones, which are then made available to the user. *Programming by example* is concerned with the reuse and generalization of long input sequences. Finally, *history mechanisms* allow users to manipulate a temporally ordered list of their interactions. This kind of reuse facility is found in a variety of computer interfaces, and it is the type of reuse we investigate for this thesis. Thus, in this subsection, we briefly describe Greenberg's (1993a) taxonomy of history mechanisms, and discuss the interaction style that is relevant to this research: *history by navigational traces*.

Greenberg's taxonomy of history mechanisms is based upon four fundamentally different interaction styles that differ in the method they use for offering candidates for reselection, and the user interface they present for manipulating history data. The first three styles pertain to command-line interfaces. The final style, *history by navigational traces*, involves navigating some type of information structure such as the hypertext-based World Wide Web. These four mechanisms are listed below.

- *Glass teletypes* are traditional Video Display Units (VDUs) that present a fixed viewport into a virtual roll of paper; user interaction occurs through a command-line dialogue. History systems typically consist of time-ordered event lists that require substantial cognitive and physical effort to recall and manipulate previous commands.
- *Graphical systems* run on high resolution bitmapped workstations that allow text to be placed anywhere on the screen. They have reduced the need to remember what commands were submitted and in what order; typical history mechanisms present a menu of previous events, where items can be selected and manipulated with a pointing device.
- *History by editing transcripts* is common within window-based terminal emulators. This is not a command history mechanism per se, but similar capabilities are provided by allowing the user to select a text region from the display and paste it into the

command input area. The major benefit of this history method is that commands are retained in their original context; otherwise, considerable effort is still required to find the relevant item and create a customized list of items for reuse (Greenberg, 1993a).

- *History by navigational traces* involves the application of history where items must be retrieved through some navigational process e.g. traversing menu hierarchies, file directories, and hypertext documents. History can record the path taken, and the information finally selected. This is the type of interaction style that we will investigate for graphical WWW browsers.

1.4 WWW navigation vs. command-based systems: task and domain differences

Past research into command reuse and interface history mechanisms has addressed command-line systems such as Unix which have a glass teletype interaction style. However, navigating hypertext, particularly within the WWW, differs from command-line systems in several ways. These differences include the type of user interaction, the length of recurrences, and the cost differential in using history versus not using it. These differences motivate the study of reuse within the domain of hypertext and the WWW.

Type of user interaction. In command-line systems, the user's goal is to submit their next command to the operating system. If the next command was issued previously, the user has a choice of either selecting the command from a history list or reentering it. The goal in WWW browsing is more complex in that browsing may involve several activities. The user is interested in accessing a particular page: in the best case, this involves selecting a hyperlink on the current page or invoking a browser action such as *Back*. More often, satisfying the user's goal may require a series of page accesses (and references to other sources such as email, notes, friends) to locate the URL for the page the user desires.

Length of recurrences. The length of the recurrence (typing a Unix command line or typing a URL to visit) is also different. For example, most simple Unix recurrences are short (6 characters on average according to Greenberg, 1993a) whereas URLs can be very

long and unwieldy. Also, URLs do not exist within a limited *command* and *option* set such as Unix commands do. Thus, reentering a URL is very often not an option for users. Our study will show that few URLs are actually typed—most are accessed by clicking a hyperlink. The real cost of this method requires knowledge of the user's current hand position (mouse vs. keyboard), and the amount of scrolling required to access the hyperlink.

History use costs. The costs in using a history mechanism also differ. In command-line systems, the trade-off occurs between reentering the command versus accessing and selecting it from the history list. Within WWW browsers, the effort in reentering a URL, selecting it from a hotlist, or navigating to the site are not the only costs: the user may not remember the URL, may not have placed it into their hotlist, or may not recall the path they took previously to arrive at the page. Thus, a major portion of the effort involved in accessing a page is in determining *how* to get to the page (by navigating to a jumping off point, using a search engine, etc.); a smaller portion concerns the actual navigation to the site once the user knows the route. So, history may be more important in WWW browsing because the cost of finding a page on one's own (versus reentering a Unix command) is much greater than selecting the item from a list or reentering it. Lee (1992) states that history tools can alleviate the physical effort of issuing a recurrent command; this is true for WWW browsing, though in this case history tools can also relieve a great deal of cognitive effort and significantly improve task performance times, not to mention the time to download the pages, and the possibility that intermediate sites may be inaccessible.

1.5 Problem Statement

The hypothesis of the research in this thesis is that users revisit WWW pages, and that an examination of the patterns of WWW navigation reuse can provide insight into more effective browser history mechanism design. Therefore, this research addresses two major problems.

1. There is little empirical data about how users repeat their access to previously visited pages. From research by Catledge and Pitkow (1995), we know that the *Back* action is heavily used, while the history list is seldom used to return to a page. However, the

proportion of Web pages that are revisited by a particular user has not been quantified, and no research has examined patterns of reuse in this domain. This thesis presents quantitative results about revisits to Web pages, and examines five possible patterns of reuse (e.g. browsing within the same subset of pages during multiple sessions).

2. Current history mechanisms have not been thoroughly evaluated. Cockburn and Jones (1996) performed a usability study that illuminated difficulties with the current browser history mechanism. However, the predictiveness of this mechanism has not been quantified, nor have other possible representations been explored. This thesis compares current designs and various alternatives. We will argue that today's WWW history mechanisms provide incomplete support for reaccessing pages, are piecemeal in their design, and have been developed in an ad hoc fashion.

The majority of this research discusses analyses performed upon navigational traces of WWW browsing. Where possible, we have replicated analyses performed in previous studies within the hypertext domain and from other domains. We then use our findings to evaluate current history mechanisms.

1.6 Thesis Outline

This thesis is divided into three parts. The first part, Chapter 2, characterizes the uses of history tools within hypertext systems. The second part, Chapters 3, 4, and 5, describes an empirical study and analysis of actual user WWW navigation behaviour. The third part, Chapters 6 and 7, examines various methods for presenting previously visited pages, and discusses the implications of these findings.

Chapter 2 characterizes the uses of history tools within non-distributed hypertext systems, and graphical WWW browsers. A taxonomy of the types of history mechanisms for these two broad classes of hypertext is proposed. Examples of systems that use these features are also discussed.

Chapter 3 describes the methodology for a WWW navigation usage study. This includes the objectives of the study, subjects, apparatus, data collection, method, data selection, and potential problems.

Chapter 4 presents summary statistics derived from the study data. The methods used to access a Web page provide insight into current history use while the rate that Web pages are revisited verifies the importance of this interface feature.

Chapter 5 examines five patterns of reuse that evolve over time: growth of the user's repertoire of pages, frequency of page visits, URL revisits as a function of distance, accessing a cluster of pages, and repeated sequences of URLs. These diverse patterns highlight the need for history mechanisms of various kinds to accommodate differences in user browsing strategies and task demands.

Chapter 6 evaluates eight methods of presenting a history of past URL visits to a user. It measures how well each predicts the user's next URL selection. The methods include the current stack-based model, as well as other approaches. Our results are also compared to a previous study that assessed methods for presenting Unix command line history.

Chapter 7 considers the implications of our research. Guidelines for history mechanism design are adapted from Greenberg (1993b), and applied to WWW browsers. Existing hypertext systems are evaluated against these guidelines.

Chapter 8 summarizes the thesis and our research contributions. A number of future research directions for reuse in WWW navigation are discussed.

2. History within Hypertext Systems

This chapter examines the use of history mechanisms within hypertext systems. For the purposes of this discussion, two classes of hypertext systems are considered: non-distributed hypertext, and graphical WWW browsers. Non-distributed hypertext systems are the standalone, small-scale, and platform-dependent systems that preceded the global World Wide Web. Examples of these systems include NoteCards (Xerox PARC), Symbolics Document Examiner, Intermedia (Brown University), and HyperCard (Apple). Five history mechanisms featured in several non-distributed systems are discussed in Section 2.1. Section 2.2 surveys history mechanisms within current graphical WWW browsers. While these contain most features from non-distributed hypertext, they also contain new features as well.

The history mechanisms considered in this taxonomy are limited to system-supplied functions that permit a user to return to a previously visited node. Authors may also define navigation aids for this purpose in their content. For example, a WWW page may contain a *Return to Home Page* hyperlink to make it easier for the user to navigate back through a series of pages. However, these are static navigational aids that are intertwined with the content versus history mechanisms that change to reflect a user's browsing activities.

We conclude the chapter with a summary of the research questions addressed in this thesis.

2.1 History use in non-distributed hypertext systems

This section surveys history mechanisms used within non-distributed hypertext systems that include the following: backtracking, history list, "already-visited" cues, paths, and personalized lists of nodes. *Backtracking* allows the user to reverse their navigation sequence. *History lists* eliminate the need to backtrack through several nodes by providing a mechanism to quickly return to a previous node by selecting it from a list. *Already-visited cues* indicate that a node has been previously visited, and is useful for helping users avoid undesired repetition, and for guiding them to familiar territory from their current

position (Bernstein, 1988). The two remaining history mechanisms require explicit specification by the user. *Paths* create an association amongst a sequence of nodes, while *personalized lists of nodes* allow the user to record interesting places in the hypertext.

2.1.1 Backtracking

Backtracking allows the user to visit previously visited nodes. It can be accomplished using two methods: path-following, and arbitrary jumps to previous nodes. Path-following allows one to traverse in reverse order their previously visited nodes; this method relies on the user's memory of their navigation behaviour because they must recall the nodes visited and their sequence. Arbitrary jumps to previous nodes are usually implemented as a textual history list (see Section 2.1.2). Both methods provide temporal context which is important in reducing user disorientation.

Nielsen (1995) claims that backtracking is probably the most important hypertext navigation facility. According to Nielsen, it should always be available, always be activated in the same way, and it should be possible for the user to backtrack all of the way to their starting position. Problems can arise when the user backtracks more than once, and has visited a particular node more than once. In HyperCard, retracing one's steps does not necessarily allow the user to revisit locations in the order encountered because of the way its history list is ordered (see Section 2.1.2). Thus, backtracking often results in an endless loop that hops between two cards (Mylonas & Heath, 1990). Chronological backtracking is the simplest backtrack model though it is inefficient because the user will revisit certain nodes several times. It is important that an alternative backtrack model, if chosen, fit the user's conceptual model because most hypertext systems do not identify the node the user would be going back to. An exception is General Magic's Magic Cap interface which consistently lists the name of the node in the upper right hand corner of the screen (Nielsen, 1995).

2.1.2 History List

A second hypertext history mechanism is the history list. This mechanism tends to take one of two representations: a "cache" of graphical miniatures of the most recently visited nodes, or a sequential text-based list of the titles of nodes visited. Nielsen implemented the

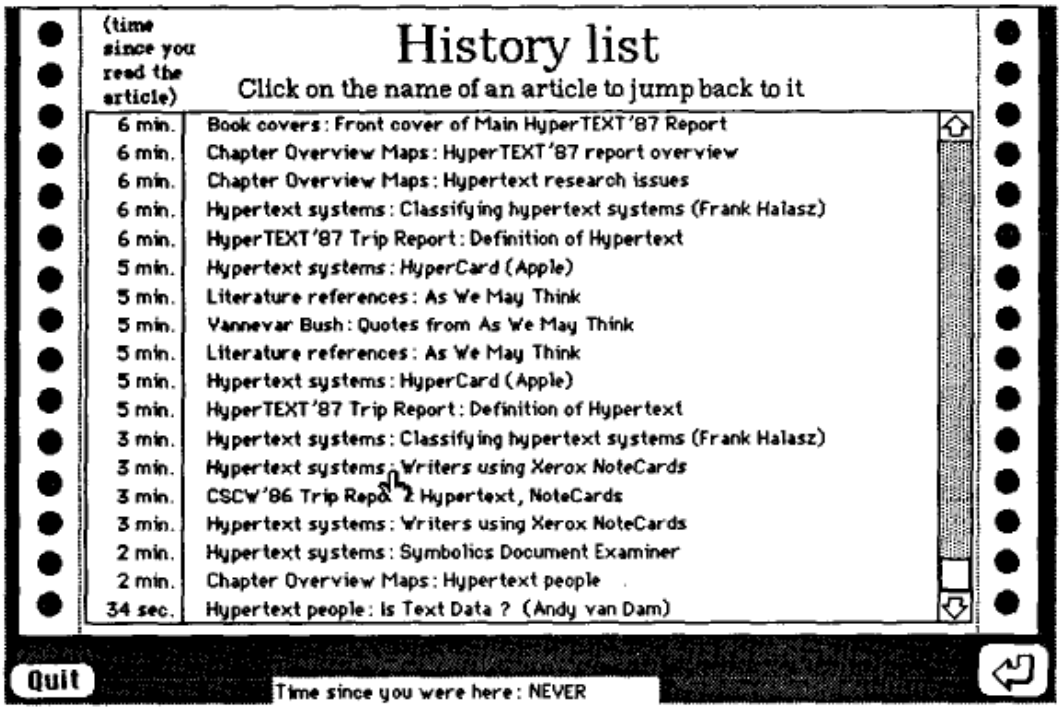

Figure 2.1. History list from Hypertext '87 Trip Report (Nielsen, 1995, p. 30)

latter method in a HyperCard-based hypertext system that contains a trip report to the Hypertext '87 conference.

A history list can be accessed by clicking the history list icon located in the bottom right hand corner of the screen. A new card appears containing a scrollable list of previously visited cards (articles) in sequential time order (Figure 2.1). Each item consists of the time since the article was read, and the name of the article. The most recent article appears at the bottom of the list which is counter to the normal top-down scanning of lists, and thus may be less effective. The history list is not restricted to the current session; the user can scroll through articles accessed days or perhaps years ago. Also, it is a truly linear mapping of the user's path through the article, for the system adds a history item every time the article is visited—that is, duplicates are not removed (Nielsen, 1995).

Bernstein's (1988) Hypergate system also offers the user a menu of recently visited pages from which they may jump directly to any page in the menu. The length of the menu is limited to thirty pages.

The Symbolics Lisp environment contains a window-based hypertext system called the Document Examiner, designed for its extensive online manual. Document Examiner provides two types of historical lists, a command history and a history of *topics* examined (Utting and Yankelovich, 1989). For the latter, the main window contains a bookmarks area that displays a history list of previously viewed topics, and topics explicitly added by the user.

A visual "cache" approach was used by the National Museum of Denmark in their museum information system (Nielsen, 1995). Each screen in the information system includes a graphic of a museum artifact. A miniature of this graphic is displayed horizontally along the bottom of the screen to provide the user with a quick method for returning to the eight most recently visited nodes.

Macintosh HyperCard contains a history facility called *Recent* that also presents a list of pictorial miniatures. When accessed, *Recent* replaces the current card with one displaying the last 42 unique cards visited. The miniatures are placed in order of first appearance and a large border distinguishes the last card visited. When more than 42 cards have been visited, the first ones are replaced by newly visited cards. Any card on the display can be revisited by selecting it (Greenberg, 1993b).

A variation on the history list called the *History Tree* shows NoteCards users how they traversed a set of linked nodes, including digressions, and multiple visits (Figure 2.2). The display is hierarchical rather than linear. The history tree can be annotated with text and graphics (Utting and Yankelovich, 1989).

2.1.3 "Already-visited" cues

Visual indicators such as checkmarks, asterisks, or the plus sign serve as "footprints" on overview diagrams and help users to avoid returning to nodes that have been recently visited (Balasubramanian, 1994). Bernstein's Hypergate system displays a small marker called a "breadcrumb" next to hyperlinks that will lead to material the reader has already seen. In NoteCards, Foss's extensions to the graphical History List place a plus sign next to nodes already visited, one for each time visited (Utting and Yankelovich, 1989).

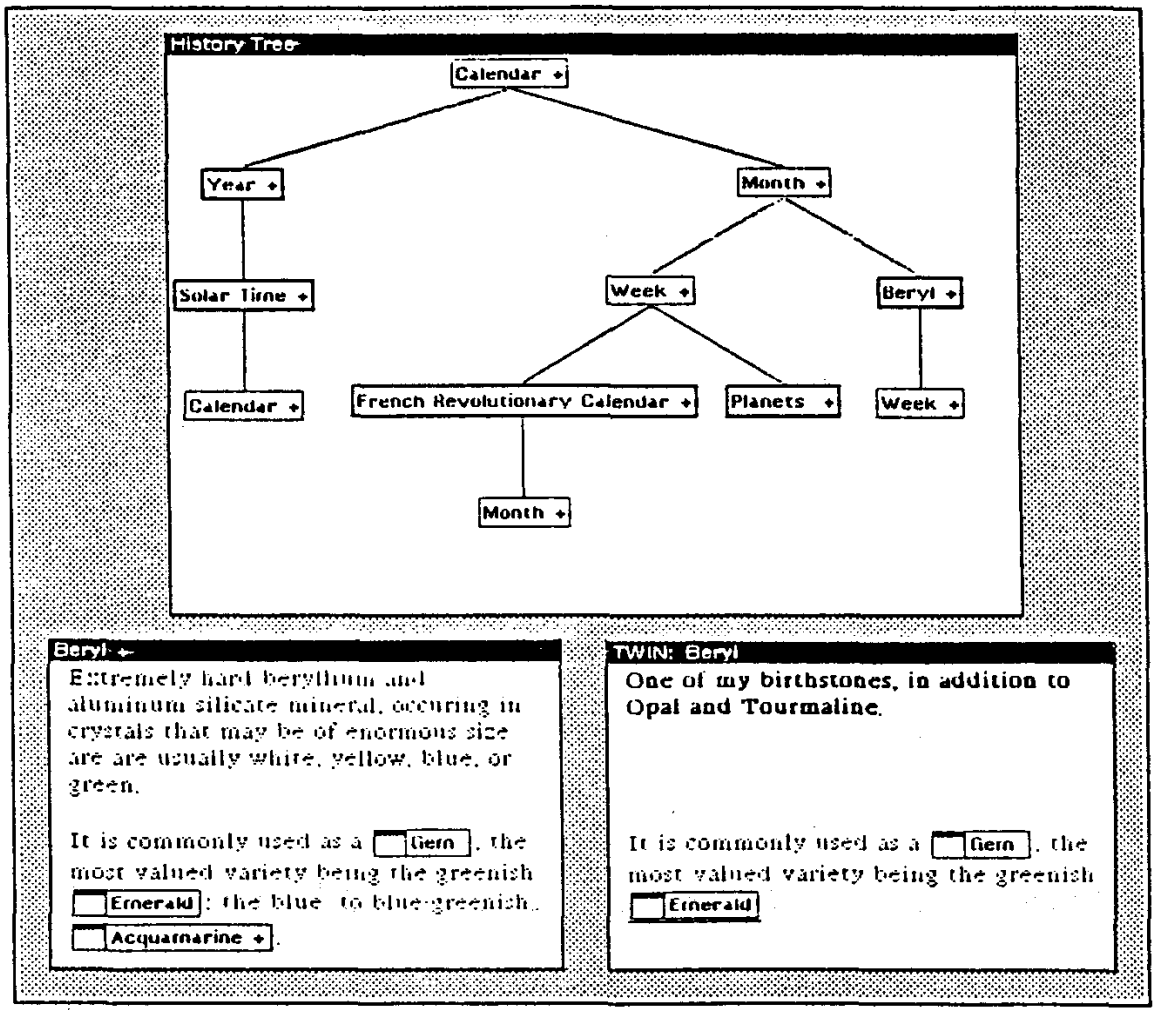

Figure 2.2. NoteCards history tree (Utting and Yankelovich, 1989, p. 72).

Similarly, Nielsen's HyperCard-based hypertext system places a checkmark next to nodes that have been visited within the overview map (Nielsen, 1990a).

2.1.4 Paths

The concept of paths or trails in hypertext was first proposed by Vannevar Bush in his seminal paper, "As We May Think" (Bush, 1945). Bush described the design of a mechanized system called the Memex ("memory extender") that would allow an individual to store all of their records and access them easily and quickly. An important feature of the Memex was the user's ability to link together items they considered related, thereby forming a *path* through the records that could be revisited at a future time.

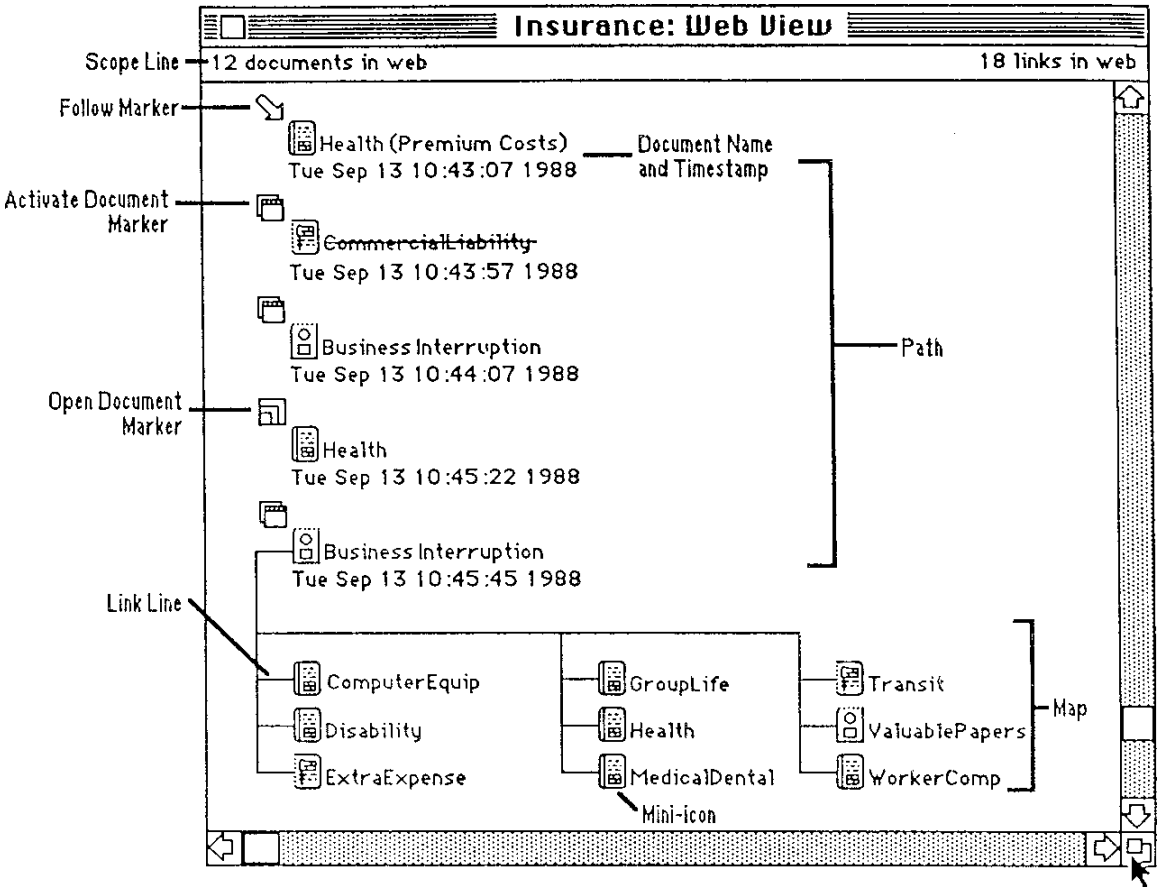

Figure 2.3. Intermedia Web View (Utting and Yankelovich, 1989, p. 78).

In the hypertext systems of today, *paths* are typically associated with the idea of a guided tour, where the author determines an appropriate order of presentation for a given audience. The guided tour may even include annotations explaining the items on the path. For example, Trigg (1988) extended this concept to NoteCards and called *stops* along the tour *tabletops*. Tabletops consist of sets of cards and annotations arranged on the screen in a particular layout.

However, some hypertext systems use paths to present history information. For example, in Intermedia a path is a linear list of documents users visited earlier in a browsing session (Utting and Yankelovich, 1989). A user's path is saved when closing an Intermedia *web* (set of hypertext nodes), and restored when the web is later reopened. As Figure 2.3 demonstrates, Intermedia paths display the name of the document (e.g. *Health (Premium Costs)*), an icon indicating the type of event (*following* a link, *activating* an

already open document, or *opening* a document), and a timestamp specifying when the event occurred. The *map* at the bottom of Figure 2.3 displays all the documents linked to the current document (*Business Interruption*), and is updated as the current document changes.

2.1.5 Personalized lists of nodes

Personalized lists of nodes can take two forms: bookmark lists, or a personalized node that contains links to nodes of interest. Bookmark lists allow the user to identify nodes that they may want to return to later. Thus, the user must explicitly take an action to set a bookmark which differs from history lists that are automatically updated. Bookmark lists tend to be smaller and more manageable than history lists though they may not include everything of relevance (Nielsen, 1990a).

One example is Monk's (1989) Personal Browser. It was designed to assist directed (versus exploratory) navigation—situations in which the user has a specific navigational goal. This system was developed as a HyperCard prototype to allow the user to create a customized card of links to nodes of interest. The Personal Browser card would replace the current card on the display when invoked, and include a *go back* button. The user could organize the buttons (links) on their personal card, and add text and graphics. Monk proposed extending his idea by having the application prompt the user to add cards to their personal browser card if they have been visited a certain number of times.

2.2 Graphical WWW Browsers

This section surveys history mechanisms found in graphical World Wide Web browsers. Web browsers and non-distributed hypertext systems share many of the same history mechanisms: backtracking, history list, "already-visited" cues, and personalized lists of nodes. In addition, graphical WWW browsers, and their add-on tools, may include search mechanisms, inter-sessional history views, and even alternative browsing mechanisms.

2.2.1 Backtracking

Three commands in graphical WWW browsers assist the user in revisiting previously visited nodes: *Back*, *Forward*, and *Home*. These commands are typically activated through a button click, menu item selection, or shortcut key. Back and Forward determine the current position in a stack of previously visited or *recallable* pages, as they are defined by Cockburn and Jones (1996); they do not control the browsing of a temporal ordering of previously visited pages (see Section 6.1.6). Back returns the user to the previous page on the Mosaic History list or Netscape Go list. Forward is a *next* document operation which undoes the effect of Back. This is important, as Forward retains the integrity of the history list while choosing a hyperlink at this point destroys it. Home takes the user to the Web page that is defined as their *home* document leaving the stack intact.

Our study, and a study by Catledge and Pitkow (1995) show that Back is a very heavily used navigation command, while Forward and Home are infrequently used (see Section 4.3). Catledge and Pitkow reported that Back accounted for 40.6% of all navigation actions, Forward accounted for 2%, and Home accounted for 0.5%. In our study, Back accounted for 30% of navigation actions, Forward accounted for 0.8%, and Home accounted for 0.9%. The 10% difference in the use of Back may be due to the users skill differences. The Catledge and Pitkow study took place when many users were discovering the Web for the first time while our subjects were experienced Web users. The latter group may have exhibited more purposeful browsing as opposed to exploratory browsing that likely leads to greater use of Back.

2.2.2 Linear History List

The linear history list is a common feature of graphical WWW browsers, and tends to be implemented as either a menu or dialog box containing the page titles that are recallable. One difference between browsers is whether the most recent pages appear at the top or the bottom of the list. Mosaic's History list and TkWWW's Recall list put the most recent page at the bottom, while Netscape's Go list displays it at the top (Figure 2.4). As mentioned earlier, bottom-up ordering of most recent pages runs counter to the normal top-down scanning of lists.

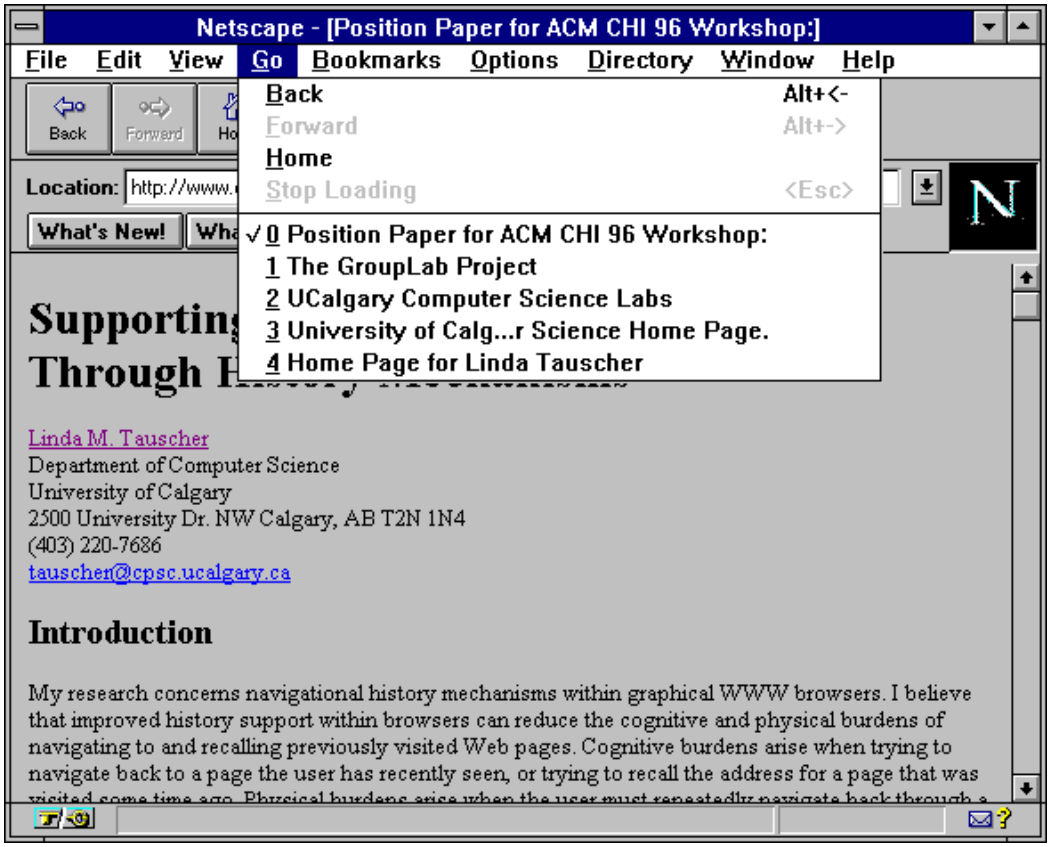

Figure 2.4. Netscape Go list

2.2.3 Branching History List

The branching history list provides a two-dimensional representation of the Web pages the user has visited. Its main benefit compared to the linear history list is that the branching list provides more information about the structure of the Web space the user has visited. Both Cockburn and Jones (1996), and Ayers and Stasko (1995) have proposed history mechanisms of this type. Cockburn and Jones have implemented their tool, *WebNet*, in Tcl/Tk; it is designed to run along-side any standard web-browsing client. WebNet displays a scrollable graphical overview of the web subspace visited in a session (Figure 2.5). 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 corresponding page.

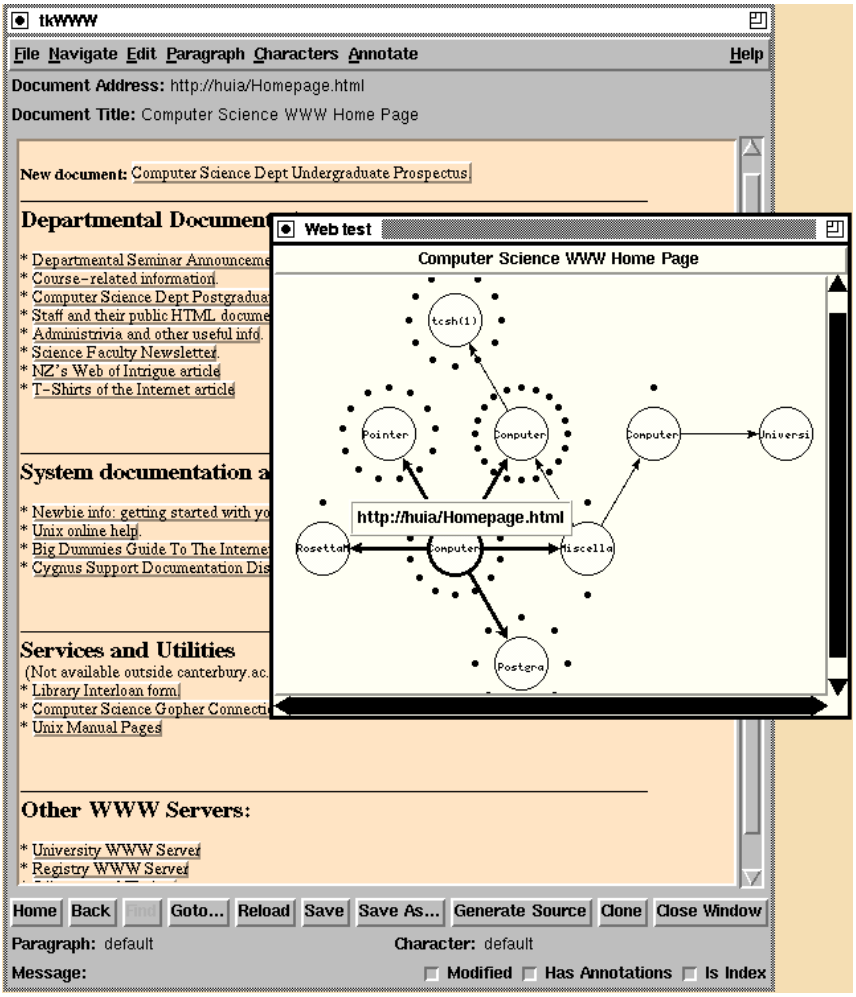

Figure 2.5. WebNet (Cockburn and Jones, 1996)

Information overload in WebNet is dealt with by two methods.

1. WebNet partitions the navigation history into *web subspaces*. A new subspace is created when the user *directly accesses* a URL (e.g. selects it from their hotlist or types it into the URL field). A *dynamic page menu* lists the page title for the directly accessed URL, and a cascading menu displays the remaining pages accessed while in that subspace (Figure 6.1).

WebNet contains a view filter that alters the size of the nodes proportionately with respect to the selected criterion. Current criterion include the following: 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

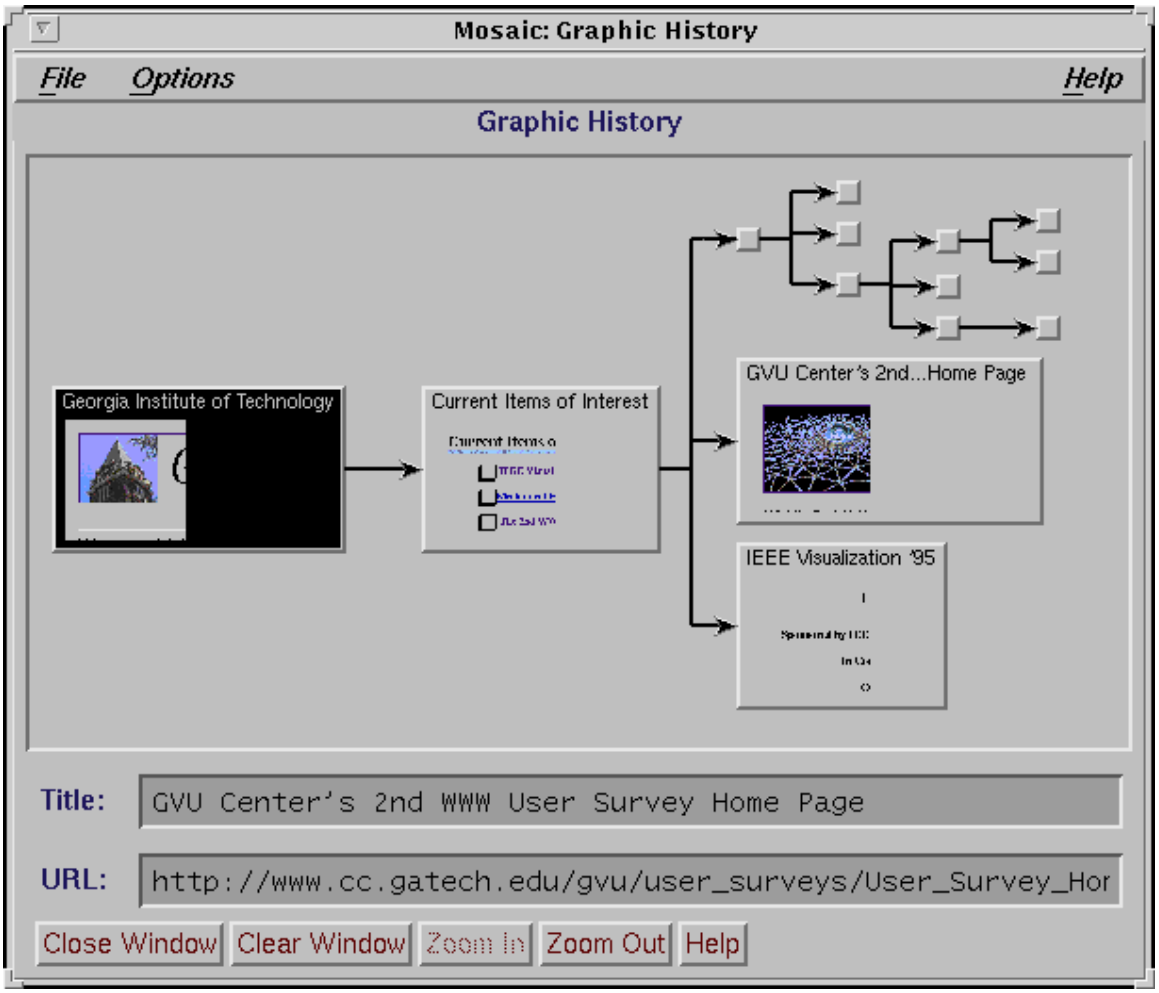

Figure 2.6. MosaicG Graphic History View (Ayers and Stasko, 1995)

are shown largest, and the fewer the number of visits, the smaller the node appears (Cockburn and Jones, 1996).

Ayers and Stasko (1995) have implemented MosaicG, a modified version of Mosaic 2.5 which provides a two-dimensional view of the documents a user has visited in a session. The Graphic History View presents titles, URLs, and thumbnail images of the documents visited in a session, according to user preferences (Figure 2.6). The graphical layout as a two-dimensional tree built from left to right with visual cues is effective in providing both spatial and temporal context, important for reducing user disorientation. Several approaches are taken to address potential scalability issues. The user can: zoom out for a smaller representation of all documents in the tree, condense branches of the tree

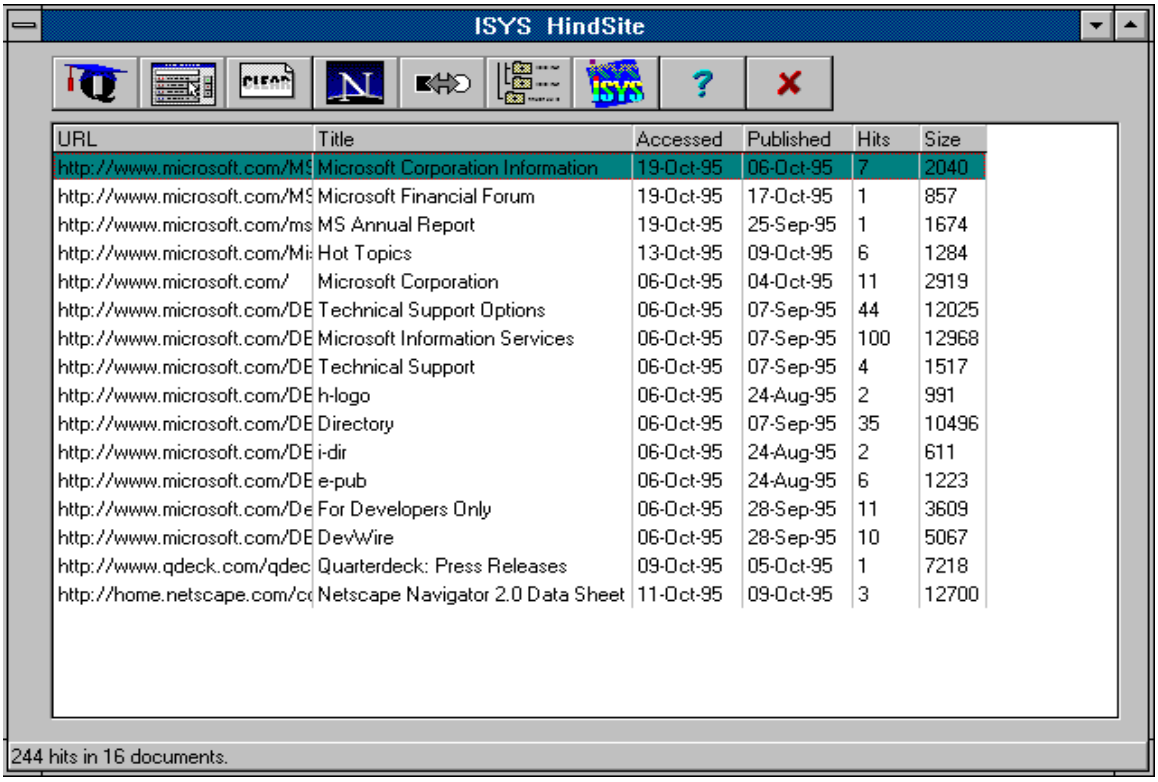

Figure 2.7. ISYS HindSite search results window

that are no longer of interest, and manually control the amount of abbreviation of page titles. All of these solutions involve additional effort on the part of the user, though users of MosaicG have expressed interest in having more power to manipulate the documents and tree structure e.g. reparent a node as the root of a tree, erase branches completely (Ayers and Stasko, 1995).

2.2.4 "Already-visited" Cues

WWW inter-sessional history is limited to telling the user that a hyperlink has been visited previously by changing the colour or style of the link on any page it is encountered. This is the major "already-visited" cue in graphical WWW browsers, and is based upon a user preference (history expiry date) and a global history list maintained by the browser.

2.2.5 Search Mechanisms

The global history list maintained in a file by the browser is not intended for use by the typical user. However, some sophisticated computer users use file manipulation

commands (e.g. grep in Unix) to search this file for a pattern. There is now a Windows product, ISYS HindSite, that provides a search interface to one's history list (Figure 2.7). HindSite indexes the contents of every page the user has visited which makes it a powerful search tool.

2.2.6 Inter-sessional History Views

Early browser designs were sorely lacking in support for capturing, accessing, and viewing inter-sessional history i.e. navigations collected across browsing sessions. Today, inter-sessional history is being integrated into browsers or is being provided as add-on software applications. Their effectiveness in reducing the cognitive and physical effort of locating the desired URL within the inter-sessional history view varies. Various methods of pruning, organizing, and representing the URLs within a history are discussed below.

Wetherall (1995) was one of the first to research the access to and viewing of WWW inter-sessional history. Wetherall developed a script that presents browser history information as a Web page. This page makes the user's history list available for others as well as allowing the user to relocate URLs. Two views of the data are provided: URLs sorted by date last visited, and URLs sorted by server. For the listings by date, the URL is displayed only for the most recent date accessed which is a very reasonable choice. A technique called *path compression* is used to compress links that follow a directory path into a single line of the listing. Each portion of the URL can be selected to take the user to the link up to and including that section of the URL. This presents several URLs within a single line, economizing on the length of the history list by about 10% (Wetherall, 1995).

Navinet Inc. has a web log feature within their Overdrive navigation software that works in conjunction with Netscape (Figure 2.8). 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, and should be integrated with the Logger itself.

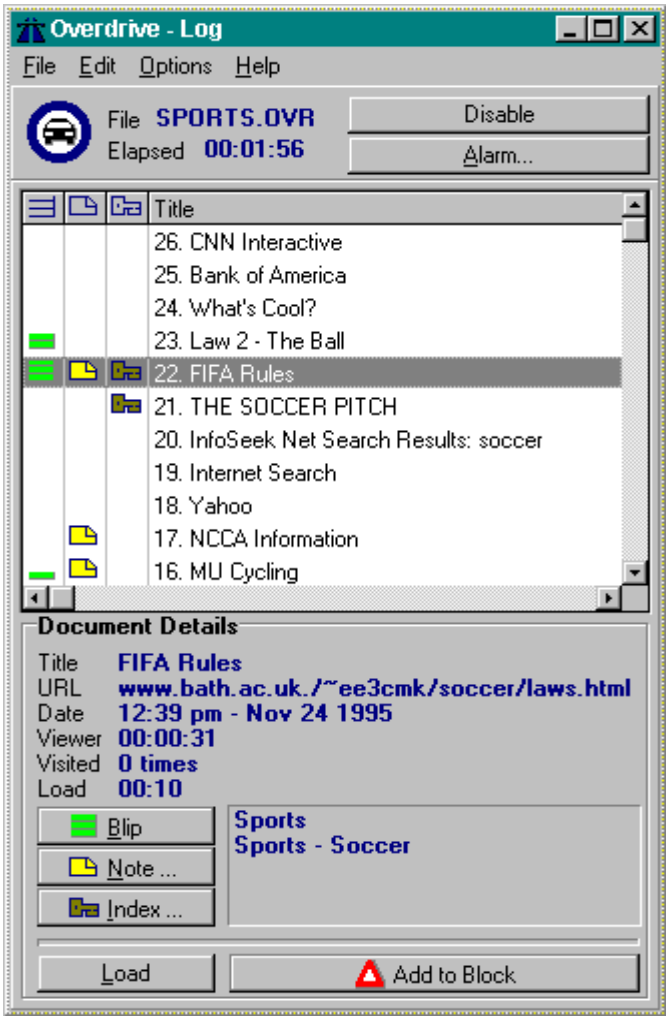

Figure 2.8. Navinet's Overdrive Logger.

WebNet (Cockburn and Jones, 1996) is also capable of storing inter-sessional views, or *web subspaces* that can be used as a sort of guided tour by other users. The user can save web subspaces generated during a browsing session (see Section 2.2.7).

Ayers and Stasko's (1995) Graphic History View (see Section 2.2.3) allows the user to save a browsing session as a text file; the user must explicitly invoke the *Save* command to do so. The thumbnail images of each Web page are not preserved though they are updated if the page is revisited.

Apple's Internet Explorer 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

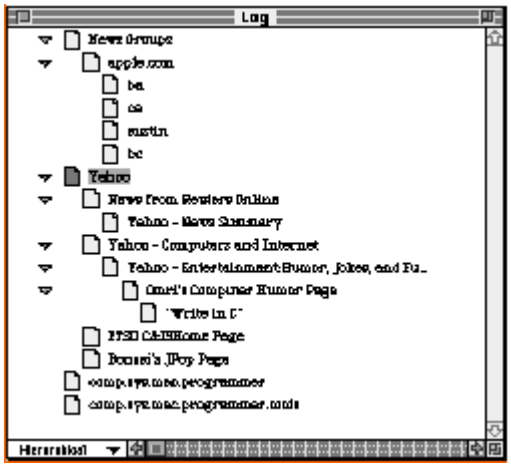

Figure 2.9. Internet Explorer Log window

alphabetically, chronologically, or hierarchically. Figure 2.9 displays a hierarchical representation. Items are named by title, and the user can view the address by applying the *Get Info* command to the item. 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 (Section 2.2.6), or saved as Internet references on the desktop, allowing the user to share them with others.

2.2.7 Personalized Lists of URLs

All graphical WWW 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. In general, personalized lists of URLs 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 it on the display or entering its URL into a dialog box.

Most browsers now support hierarchical hotlists though few users seem to use them. This may change with the new drag and drop method of creating and organizing folders to store URLs within Netscape Navigator 2.0, and Internet Explorer's *Notebook*. Cockburn

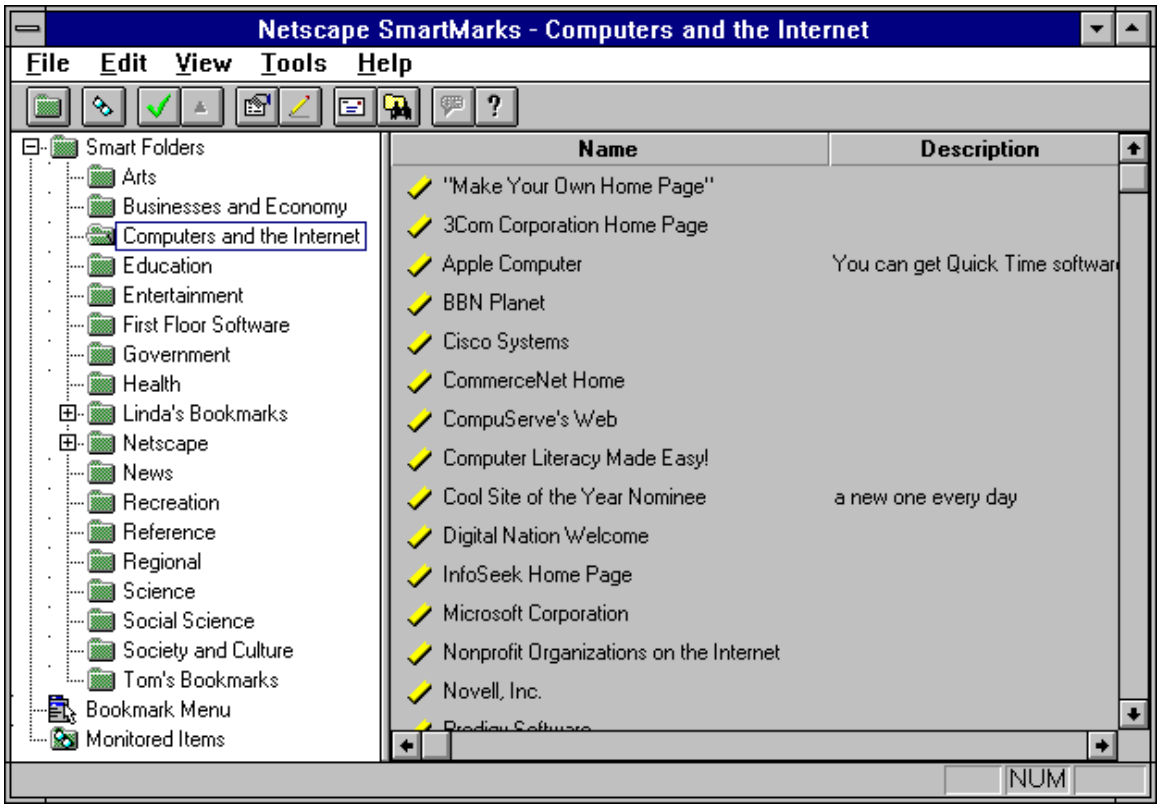

Figure 2.10. Netscape SmartMarks

and Jones (1996) describe WebNet's feature for saving Web subspaces as providing a rich variant of a bookmark or hotlist facility.

Netscape SmartMarks is an advanced bookmark facility for Netscape Navigator. It was the first software system with the ability to organize bookmarks in a hierarchy of folders using drag-and-drop editing (Figure 2.10). Additional interesting features include the following: a sample catalog with folders and links to popular topics and Web sites, the ability to save a search as a bookmark, the ability to search local folders, and the ability to inform the user of changes to pages.

Authoring personal home pages is another method for organizing personalized lists of URLs. It requires more skill and effort than managing bookmarks. The user must be capable of authoring a document using HyperText Markup Language (HTML) or dedicated authoring tools, though extensions to common word processors are reducing

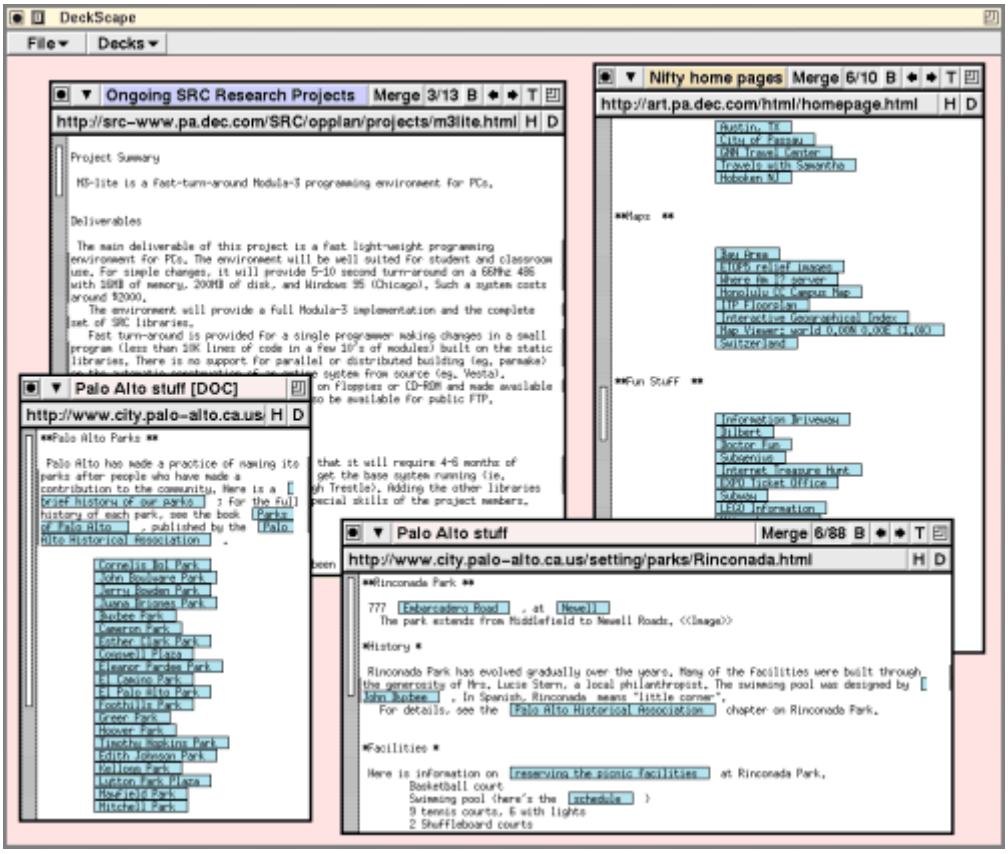

Figure 2.11. DeckScape with 3 decks and away page (Brown and Shillner, 1995)

this need. One of the first non-distributed hypertext systems to implement the concept of a personal home page was Monk's (1989) Personal Browser (see Section 2.1.5).

2.2.8 Alternative Browsers

Alternative browsing metaphors may present Web browsing history in innovative ways. For example, DeckScape is an experimental WWW browser based on a "deck" metaphor (Brown and Shillner, 1995). A deck consists of a collection of Web pages. Only the page at the top of the deck is visible, though the *away page* feature allows multiple pages from the same deck to be displayed simultaneously. For example, in Figure 2.11 we see three decks: *Ongoing SRC Research Projects*, *Nifty home pages*, and *Palo Alto stuff*. The narrow window in the lower-left shows an away page from the *Palo Alto stuff* deck. The user can easy jump to the top or bottom of a deck. Multiple decks can exist within a

single browser window. Various operations can be performed upon these decks: iconize, move or copy pages between decks, create new decks, delete decks or pages, etc.

DeckScape is interesting for several reasons. The deck metaphor provides a powerful method of organizing material that integrates several components of current browsers including hierarchical hotlists, history list, and the ability to open multiple windows. This integration and a multi-threading feature permit advanced browsing techniques such as expanding all links on the current page (the results of which DeckScape returns as a deck), and quickly revisiting pages by flipping through the deck without re-fetching the pages.

In terms of history, DeckScape retains all pages visited until the user discards them (rather than only those pages on the path from the root page to the current page). 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.

2.3 Concluding Remarks

Many history mechanisms that were common in earlier nondistributed hypertext systems are now standard features of WWW browsers. However, our hypothesis is that WWW browsers currently make use of history to a limited degree, and in a piecemeal and ad-hoc manner. A tighter integration of a variety of history mechanisms needs to occur that is based upon empirical data about how users make use of history. Such data is lacking for hypertext systems but does exist for command-line systems. There are sufficient differences between the two domains to warrant a reapplication of the investigative techniques applied to command-line systems, and the addition of other techniques that are more appropriate to the hypertext domain. This research will examine the extent of page reuse within the World Wide Web, today's most popular, distributed hypertext system. Specifically, the following research questions will be investigated:

- 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 reuse patterns and thereby make it easier to navigate back to pages of interest?

3. WWW Navigation Usage Study - Methodology

This chapter describes the methodology of a study of WWW browser use that focused on navigation activities. The first section explains the four objectives of the study. Subsequent sections describe the subjects, instructions to subjects, apparatus, data collection, method, and data selection. The chapter concludes with a discussion of potential problems within the study, and their probable impact.

The following two chapters present the results of the study. Chapter 4 addresses summary statistics for navigation methods and recurrence rate. Chapter 5 looks at patterns over time that include the following: the growth of URL vocabulary; URL recurrence rate as a function of distance; frequency of URL accesses; locality; and longest repeated subsequences.

3.1 Objectives of the Study

Our major objective was to understand the aspects of browsing that relate to repetition when users navigate to Web pages. This led to several specific objectives.

1. First, we wanted to identify the rate that different navigation methods are used to access a URL. This replicates a previous study by Catledge and Pitkow (1995). Our results are presented in Section 4.1, Navigation Methods.
2. Second, we wanted to examine the use of existing history mechanisms within WWW browsers. Catledge and Pitkow (1995) report some relevant statistics in their study. We took this one step further by conducting follow-up interviews with our own subjects to gather qualitative data about history use and the types of pages that are revisited. These results are also presented in Section 4.1.
3. Third, we wanted to understand how often WWW users revisit Web pages. This statistic is called *recurrence rate*, and it is described and analyzed in Section 4.2. We also wanted to examine a related concept—the growth in each user's repertoire of new pages visited (Section 5.1).

4. Finally, we were interested in whether usage patterns can be detected that could be used to predict which page a user will revisit next. Several metrics were analyzed. The analysis of usage patterns of recency (Section 5.2), frequency (Section 5.3), and locality (Section 5.4) are modelled after similar analyses done with command line data (Greenberg, 1993a; Lee 1992). Similarly, analysis of longest repeated subsequences (Section 5.4) are contrasted against those found by Catledge and Pitkow (1995).

The next sections describe the subjects that participated in the study, instructions to subjects, apparatus, method, data selection, and potential problems.

3.2 Subjects

The subjects were 28 unpaid volunteers that belonged to one of two groups.

1. Nineteen subjects were associated with the University of Calgary either as computer science students, researchers, professors, or support staff. There were eighteen males and one female in this group, ranging in age from 22 to 41 years.
2. Nine subjects were employed by a local telecommunications company as computer scientists, engineers, or technologists. There were eight males and one female in this group, ranging in age from 25 to 35 years.

All subjects work within the Unix environment on a daily basis, and had used a graphical WWW browser for at least one year at the point the study was undertaken.

Should the results of these two groups be pooled or considered separately? Though the characteristics of both groups are similar, summary statistics for each group were calculated to report the relative proportion of user actions (see Section 4.1), and recurrence rate (see Section 4.2). The major difference between them was the number of log statements generated, with the corporate group reporting a higher mean than the university group. However, the standard deviation for the corporate group was much higher, and a simple *t*-test comparing the means for the proportion of navigation events as well as the recurrence rate showed no statistically significant difference (df=21, p<.01). Therefore, data from the two groups was pooled for the analysis.

3.3 Instructions to subjects

During the orientation session, subjects were informed verbally that:

- data about various WWW actions, including URLs visited, would be accessible only to the investigator;
- the identity of the subjects 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).

Full instructions to subjects are found in Appendix A, which includes a copy of the consent form (Appendix A.1) and the investigator's script for the orientation session (Appendix A.2). The script describes configuration changes necessary to replicate the subject's previous browsing environment as closely as possible. The script also walks the subject through a training session that explains the differences between Mosaic 2.6 and Netscape Navigator 1.12 (Appendix A.3).

After the orientation session, the investigator checked to ensure that the data was being logged correctly for the new subject. Any data logged during the training session was removed from the data files for the subsequent data analysis. The subjects did not require nor did they receive any additional instructions during the actual study period. No subject asked to be withdrawn from the experiment, and only one asked to see their personal data.

3.4 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. One copy of the software resided at each location, and the investigator assisted the subject in configuring their account to access the modified version rather than their normal browser. Mosaic 2.6 was the latest supported release at the time

Figure 3.1. Main browser window for Mosaic 2.6

the study was undertaken. It is written in the C language, and uses the Motif libraries for its graphical interface components (Figure 3.1).

We instrumented Mosaic 2.6 to generate log files that recorded certain user actions. One log file was created for each subject. We used the Mosaic 2.4 source code graciously provided to us by Catledge and Pitkow as a starting point for our own modifications to Mosaic 2.6. Because this study differs, we recorded a smaller subset of actions, and only the final result of an action. For example, Catledge and Pitkow (C & P) logged the opening of the hotlist dialog box as a separate log statement; we logged this event only if the user actually selected a URL from the dialog. Also, we captured data about additional features (e.g. forms), and we recorded additional data (e.g. page title).

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 3.1. Fields contained by each log entry with examples

3.5 Data collection

Table 3.1 lists the fields in each log entry. Three fields will be explained further: final action; event/action path; same/new window. The *final action* field recorded the high-level user actions logged for this study e.g. *Exit*, *Back*, *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. Appendix A.4 contains a complete list of the 82 possible combinations of these three fields that Mosaic 2.6 was instrumented to record.

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 3.2). 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-navigation actions that we considered relevant to the objectives of the study.

Most of the final actions listed in Table 3.2 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. These are referenced in Chapter 4 though they will be defined now:

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

Table 3.2. Browser actions logged

- *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 for approximately 6 weeks. Logging began after the orientation session (held the week of October 23, 1995) until December 6, 1996. 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 orientation session included a review of the differences between the two browsers.

Six weeks after the study, each subject participated in a one hour interview. During the interview, the investigator asked the subjects questions about their browsing activities and methods based upon the statistics and plots generated. Subjects were also asked to describe the importance of their top 15 most frequently accessed URLs.

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 4.2). 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 was not considered. By these criteria, data from 5 of the 28 original participants was rejected.

3.7 Potential problems and probable impact

There were three potential problems associated with the study: incomplete capture of browsing activity, errors in the data, and viewing pages containing Netscape extensions. Given the objectives of this study, we conclude that these problems do not have a significant impact upon our results.

For some subjects, the data collected does not represent their entire browsing activity during the study period. The major reason for this is that some subjects browse the Web from other locations that do not have the instrumented version. For example, the modified version of Mosaic was a Unix version only, and it was not installed on any subject's home computer. Since subjects with very low browsing activity were omitted from the study, this anomaly likely has a minor impact upon the results as we believe that enough of a record was captured to indicate normal uses.

Next, interaction, logging, and interrupt errors were identified in the data. Interaction errors arose when the browser was unable to display a requested Web page. This could occur for a variety of reasons. The user may have mistyped the URL, the page may no longer exist, the server may be inaccessible, or the server may be too busy to transmit the page. Logging errors arose from our modifications to Mosaic. They were identified by

ensuring that all fields were present, and that the date field was valid. For example, if a problem occurred while data was being written to the log file, an erroneous entry might result. Log statements in error were disregarded by all analysis programs. Interrupt errors occurred when the user successfully performed an interrupt action, and an extraneous log statement was generated. A mean of 19.91 interaction errors were identified for each subject whereas only 1.96 logging errors and 8.96 interrupt errors were noticed. Given the small numbers, we expect these errors to have no effect on our analysis.

A third problem that occurred was the difficulty in viewing pages containing Netscape's extensions to HyperText Markup Language (HTML) within the Mosaic browser. Most problematic were pages using tables when the table cells contained hyperlinks or images, for these features are not supported by Mosaic 2.6. Subjects were encouraged to invoke Netscape Navigator to view these pages. This likely had a minimal impact on the data collection because subjects stated in the post-study interview that they used Netscape Navigator only when necessary. This cannot be validated since an electronic record of when subjects invoked Netscape Navigator was not kept.

3.8 Summary

This chapter described the methodology for our study of WWW browser use. The main objectives of the study were to assess the use of different navigation methods and existing history mechanisms, determine how often Web pages are revisited, and detect the presence of usage patterns. Twenty-eight subjects used a modified version of Mosaic 2.6 for approximately six weeks. Data about all navigation events, and other salient user activities were captured. Twenty-three subjects exceeded the minimum level of browsing activity we established, so the analyses in Chapters 4 and 5 are based upon their data. Three potential problems were discussed though we do not expect them to have a significant impact upon our results.

4. Summary Statistics

This chapter presents summary statistics for the WWW navigation usage study. The first section discusses navigation methods and includes a comparison with the Catledge and Pitkow (C & P) study. The second section addresses *recurrence rate*—the frequency of repeats in URL visits. Both sections describe the analysis method, present the results, and conclude with a discussion. Chapter 5 presents the results of the other analyses performed on the study data—patterns that evolve over time.

4.1 Navigation Methods

This section examines navigation methods used during Web browsing. An important objective in performing this analysis is to identify the primary methods people use to access a URL, and their extent of usage. This somewhat replicates work by Catledge and Pitkow (1995), who captured client-side user events of XMosaic 2.4 during a three week period in August, 1994. From their results we anticipate a high proportion of both hyperlink selections and navigating back one document, and low usage of the *Forward* navigation method. Still, several new browser features are not present in XMosaic 2.4 (see Section 4.1.3), and the users in our study have used browsers more extensively. While we do not expect major differences, the two studies are contrasted.

We also wanted to assess the extent to which current browser history mechanisms are used. The C & P data, as well as past research into history mechanisms within Unix (Greenberg, 1993a; Lee, 1992), both showed a low frequency of history mechanism use. Thus, we anticipate that *Window History* and hotlist usage within Mosaic will be low in our own study, and through analyzing patterns of usage and gathering qualitative data we hope to explain why this is so. An understanding of how history is currently used, and how users browse in general, are fundamental to improving browser history mechanisms.

4.1.1 Analysis method

Much of the data analysis presented in this chapter and Chapter 5 is based upon navigation events only. Table 3.2 identifies whether each browser action logged is an

Navigation Category	Navigation Action
Open URL	Open URL
Open Local	Open Local
Back	Back
Forward	Forward
Home	Home; StartUp Document
Reload	Reload Current
Binary Transfer	Binary Transfer
Helper Applications	External Viewer; Telnet Window
New Window	Clone Window; Help; Mosaic Comment; New Window
Submit Form	Submit Form

Table 4.1. Correspondence between navigation categories & navigation actions

event that results in the navigation to a URL. For data presentation purposes, the 15 navigation actions have been reduced to 10 categories as defined in Table 4.1.

4.1.2 Results

Results from the analysis of navigation methods are presented in four subsections. First, all browser actions recorded are presented. The following subsection focuses only on navigation actions since they are of particular interest to our study. In the third subsection, we analyze the *Open URL* navigation action further, since it comprises several methods of accessing a new URL. The final subsection addresses how navigation actions are used over time.

4.1.2.1 Browser actions

Table 4.2 summarizes the mean, standard deviation, median, minimum and maximum values for all *browser actions* logged across all subjects. The important results are as follows.

- A median of 701, and a mean of 902 log statements were generated by the 23 subjects. There is a wide standard deviation (676) with the lowest subject reporting 303 log statements and the highest subject reporting 3299 log statements.
- 90% of all events logged were related to navigating to a URL with a standard deviation of only 4%. (Note: since this study did not attempt to capture all possible user interactions the percentage is only relative to recorded events. User interactions

	Log Stmts	Navigation Events	Hotlist Editing	Manage History	News	Interrupt	Exit
Mean	901.87	90.46	3.36	0.03	0.62	2.57	2.96
SD	675.99	4.16	4.33	0.13	1.80	1.99	1.88
Median	701	90.68	1.78	0	0	1.82	2.60
Min	303	82.52	0	0	0	0.70	0.17
Max	3299	97.57	14.14	0.64	6.77	8.01	6.55

Table 4.2. Frequency of browser actions as a percentage of total log statements

not recorded were either at too low a level (e.g. opening a dialog box), or of too low usage and not pertinent to our study objectives (e.g. printing a Web page). Still, all navigation events were recorded.)

- The remaining browser actions logged are: *Interrupt*, *Exit*, hotlist editing, managing the history list, and reading Usenet news. News reading actions were not classified as a navigation events since they do not involve typical Web site navigation, and can be considered a separate task. Note the low proportion of actions relating to hotlist editing; only 3% of navigation actions involved modifying the hotlist by: adding, deleting, modifying, or inserting URLs; loading an alternate hotlist; saving the hotlist. The low value for Manage History is not surprising given that the only non-navigation actions one can perform upon the history list are clearing all items from it or mailing it to an Internet user.

4.1.2.2 Navigation actions

Table 4.3 summarizes the mean, standard deviation, median, minimum and maximum values for all navigation actions. The important results are as follows.

- *Open URL* was the most frequently used navigation action (50%). Since it comprises several methods of invoking a new URL, Open URL will be discussed separately in Section 4.1.2.3.
- *Back* is a highly used navigation action, comprising 30% of navigation actions. *Forward*, however, was infrequently used; it generated only 1% of navigation actions.

	Open URL	Open Local	Back	Fwd	Home	Reload	Bin Xfr	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
SD	7.51	0.33	8.46	0.90	2.36	4.02	1.44	5.02	0.83	2.60
Mdn	51.33	0	31.33	0.60	4.38	1.15	0.29	3.32	0.44	4.33
Min	34.36	0	16.70	0	1.65	0.28	0	0	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

Table 4.3. Frequency of navigation actions as a percentage of total navigation events

- The *Home* navigation action includes both the automatic display of the StartUp document, and user selections of the *Home* document. 5% of navigation actions resulted in the display of this page.
- Three subjects had very high use of *Reload Current*. The mean for all navigation actions was 3% though there were occurrences of 10%, 13%, and 15%.
- Only 1% of navigation actions involved the downloading of a file (*Binary Transfer*).
- *Helper Applications* typically involved image viewers (Xv) or document viewers (Ghostscript); telnet applications were also included in this category of Table 4.1. A mean of 5% of navigation actions were of this type.
- The New Window category encompasses the navigation actions that cause an additional browser window to display (see Table 4.1). Few of these actions occurred (<1%).
- The *Submit Form* action arises when the user submits a form using either the GET or POST method. Search engines use forms extensively as do interactive Web applications. 4% of all navigation actions involved a forms page.

4.1.2.3 Open URL navigation action

Figure 4.1 shows the different types of *Open URL* actions, and the proportion of each observed across all subjects.

- The most popular method of selecting a URL is via clicking a hyperlink or *anchor*; 83% of *Open URL* actions were invoked in this way.

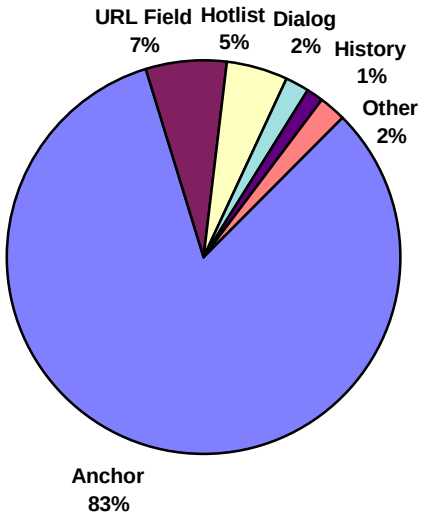

Figure 4.1. Breakdown of Open URL actions

- Mosaic's *Window History* was used very little to select a URL (1% of all *Open URL* actions). The hotlist did not fare much better; only 5% of all *Open URL* actions were initiated from the hotlist.
- Users favoured typing the URL into the URL field (7%) versus the *Open URL* dialog box (2%).

4.1.2.4 Navigation action use over time

Further insight into how navigation actions are used can be gained by analyzing their occurrence over time. The URL vocabulary graphs, Figures 5.1-5.3, described in Section 5.1.2, provide this information.[1] The *All* curve plots the unique versus total URLs visited at each point in the user's browsing timeline. For example, referring to Figure 5.1, after 300 URLs have been visited by subject15, about 70 of these are unique with the remaining 230 being visits to previously seen pages. Shifted above this line by equal steps are curves

[1] Figures 5.1-5.3 are more completely discussed in Section 5.1 which deals with the URL vocabulary growth. However, the graphs also provide an interesting visual representation of navigation use over time.

indicating the breakdown of the *All* plot into most frequent navigation actions: *Open URL, Back, Reload*, *Forms* and *Helper Applications*. Evident from these graphs are:

- regular and extensive use of *Open URL*;
- extensive use of *Back*;
- *Reload* actions usually occur in a cluster;
- use of *Forms* tends to be intermittent though there are some clusters of forms activity;
- intermittent use of *Helper Applications*.

4.1.3 Discussion

The discussion of navigation methods is divided into three topics: a comparison with the C & P study, individual variations in navigation method use, and individual similarities.

4.1.3.1 Comparison with Catledge and Pitkow (1995) study

The navigation data from our study were compared to the Catledge and Pitkow (1995) data in order to see if our subject pools are the same or different. The caveat is that there are several differences between the two studies that make our comparisons slightly suspect. There are differences in the type of user actions logged, the number of subjects, the length of the study period, and the type of subjects.

1. Catledge and Pitkow recorded all possible user actions and their intermediate steps (e.g. opening a menu, selecting a menu item, canceling a dialog box) whereas we recorded all navigation actions and particular other actions.
2. The studies used two different versions of Mosaic; our later version contains features lacking in XMosaic 2.4. These features include the following: reading Usenet news, typing a URL into the URL field, and displaying help pages in a new window. These events were logged during our study. Also, while both versions of Mosaic had forms capability, C & P did not log this data whereas our study did.
3. C & P analyzed data from 101 subjects whereas our study only involved 23 subjects.
4. They collected data for a period of 3 weeks while we collected data for approximately 6 weeks.

	http	file	ftp	gopher	news	telnet
C & P	80	8	4	4	—	—
U of C	92.93	1.76	3.19	0.25	1.71	0.06

Table 4.4. Comparison of document request protocols with C & P (1995) data (%)

5. We hypothesize that our subjects were more experienced Web browsers, for we chose subjects that had used a Web browser for at least one year. Conversely, the C & P study occurred when Mosaic was first becoming popular, and anyone running the normal Mosaic browser at their site was asked if they wished to participate in their study.

Still, reasonable comparisons can be made between the groups. First, consider document request protocols. Table 4.4 compares our results with those reported by C & P (1995). Our higher percentage for *http* is likely due to the larger number of Web servers versus other methods of serving documents in existence at the end of 1995 compared to August 1994. The lower *file* percentage in our study may be due to a larger proportion of new Web users in C & P's subject pool as new users may have been keen upon authoring a home page. Of course, news reading was not a feature of XMosaic 2.4 so the previous study could not report that protocol. Note also the dramatic drop in gopher requests, evidence of the popularity of serving documents via httpd versus gopher.

Table 4.5 compares the different proportion of navigation actions found by C & P and our study. For analysis consistency, we derived the C & P statistics directly from their original log file, which they graciously provided us. Since their study was only 3 weeks long (versus 6 weeks), we set the minimum log statement requirement to 150 (half of our 300 cutoff). As a result, data from only 55 subjects are included in Table 4.5.

The mean number of log statements for the C & P subjects was 735 statements (versus 902) with a standard deviation of 930 statements (versus 676). Navigation actions that were not possible with XMosaic 2.4 (e.g. *Helper Applications*) or not logged by C & P (e.g. *Submit Form*) were omitted from the comparison. This confounds the results somewhat because our study reports that 4.16% of navigation actions are due to *Submit Form* and 4.56% are produced by *Helper Applications*.

	Open URL	Open Local	Back	Fwd	Home	Reload	Bin Xfr	New Win	Open Anchor	Open Htlst	Open Hist
Catledge & Pitkow											
Mean	54.51	0.70	37.20	1.40	0.70	4.71	0.41	0.38	49.70	2.38	0.11
SD	8.80	1.16	9.39	1.65	1.21	9.03	1.86	0.62	9.45	2.93	0.38
University of Calgary											
Mean	50.07	0.16	30.24	0.84	5.04	3.07	1.11	0.75	41.57	2.68	0.75
SD	7.51	0.33	8.46	0.90	2.36	4.02	1.44	0.83	6.18	2.35	1.11

Table 4.5. Comparison of navigation actions with C & P (1995) data (%)

However, in terms of the other actions, there are many similarities with only minor differences. A *t*-test comparing the means of each navigation action found statistically significant differences for: *Back*, *Home*, *Open Anchor*, and *Open History* (df=76, p<.01). The *Back* and *Open Anchor* actions can be explained because the more browser-sophisticated subjects in our study used more types of navigation actions. Our *Home* category includes the StartUp Document event (see Table 4.1) whereas C & P did not log this event. Our subjects likely used *Open History* more because they are more experienced browsers. Overall, it appears that the subjects chosen are fairly typical in their Web navigation activities.

4.1.3.2 Individual variation

The conventional view of why many users browse the Web is personal interest—people browse for information pertaining to hobbies or interests outside of work or education. Browsing is apparent by the high activation of hyperlinks on a Web page, and extensive site exploration would involve high use of the *Back* action. While our subjects did exhibit a high level of browsing activity and often indicated that they were motivated by personal interest, they also showed differences in their purposes for using the Web. Examples of these different purposes are described below; they were gleaned from post-study interviews with the subjects, individual subject data, and summary statistics from Section 4.1.2.

- Some subjects performed extensive authoring of Web pages during the study. They created pages for courses, workshops, or projects. These subjects showed higher use

of *Reload Current* (3 rates were above 9% compared to the mean of 3%), as this action is commonly used to view changes made during authoring episodes.

- Some subjects used the Web in a directed search mode to locate and/or download research papers, conference information, software patches, etc. These subjects used forms frequently to search for relevant Web sites. Sometimes the research episode generated only one visit to the site, sometimes the site was revisited for further information, and there are also sites that were visited on a regular basis. We suspect that these subjects may have made greater use of their hotlist to record and reaccess important Web sites (4 hotlist editing rates were above 10% compared to the mean of 3%; 3 *Open Hotlist* rates were above 5% compared to the mean of 2.7%). They may have used helper applications (e.g. Ghostview) to read papers (4 rates were above 10% with the mean equal to 4.5%), or *Binary Transfer* to download papers and/or software (3 rates were higher than 3% with the mean equal to 1%).
- Other subjects used applications that had been integrated into the Web such as a knowledge elicitation tool, print queue query, phone book query, and documentation status query. These sites were revisited more so than regular Web sites, especially if they are an integral part of the subject's work. Web applications use forms to request data from the subject, and four subjects reported forms activity above 7% with the mean equal to 4%.

4.1.3.3 Individual similarities

The most notable similarities at the subject level include the following: percentage of events relating to navigation, proportion of events that involve local navigation, popularity of accessing a URL via selecting a hyperlink, high use of the *Back* button (30% of all navigation events), low percentage of multiple windows (<1%), infrequent use of *Window History* (<1%), and low proportion of hotlist selections (<3%). Similarities in browser use are due to the nature of hypertext, and current limitations within graphical Web browsers:

- Subjects preferred using *Back* (30%) versus *Window History* (<1%) to return to a URL. They cited several reasons for their low usage of the history list. First, accessing the *Window History* dialog involves different or more physical actions. Second, the list

of URLs must be scanned to locate the item of interest. Third, sometimes the title of a URL does not appear or it does not coincide with the headers on a page.

- Hotlist selections occur occasionally (5% of *Open URL* actions). In reaccessing a URL, the user must have gone to the effort to add the item to their hotlist. As the hotlist grows it must be reorganized, and few subjects did so during the study. Some users said that they preferred adding an important URL to their home page versus including it in their hotlist. Another subject stated that they keep no more than 10 items in their hotlist at any one time. While the hotlist could be an important tool for revisiting pages, most subjects did not want to go to the extra effort of setting it up and maintaining it. This is because people have to take time out of their current task to do work that has no benefit to them now (though it will in the future).
- Subjects engaged in a great deal of *local* navigation—once they accessed a page, they often returned to it to access other pages or while they were on their way back to a previous page. This is apparent from the high use of the *Back* action (30%). While it is natural to reaccess information and explore links to other pages, this type of navigation pattern also results from the default behaviour of only displaying one node or page at a time, and the lack of an overview map that puts the current node into context with where the user has been and where they may go next.

4.1.4 Concluding remarks

This section presented and discussed summary statistics about navigation methods from our WWW usage study. We compared our results to a previous study where possible. Both studies showed high use of *Open URL* and *Back* actions, and low use of the hotlist and history list to access a URL. We hypothesize that the effort involved in performing the latter two actions contribute to their low usage. Subjects show both similarities and differences in how they use navigation methods. Similarities are due to the nature of hypertext, and current browser design. Differences arise due to different purposes in using the Web. Overall, these results indicate that there are opportunities to improve browser design.

4.2 Recurrence Rate

History systems are only useful if users actually repeat their activities. While Web browsers contain a history mechanism that presents a stack-based list of some URLs recently visited, there is little empirical evidence confirming the extent of recurrences for Web browsing. However, research has quantified the repetition of user interactions in other domains. Greenberg (1993a) found repetition in telephone numbers dialed (57%), information retrieval in a technical manual (50%), and Unix command lines (75%). Lee (1992) obtained qualitative results that indicate that Unix users repeat individual command lines or portions of them. We hypothesize that people browsing the Web also exhibit repetition in accesses to Web pages. We analyze our own data and the Catledge and Pitkow (1995) data to derive a *recurrence rate*, that is, the probability that any URL visited is a repeat of a previous visit. We also hypothesize why users revisit pages and why they access new pages. These results can provide insights into effective browser history designs (Chapter 7).

4.2.1 Analysis method

The frequency of repeats in URL visits is called the *recurrence rate* of this activity: the probability that any *activity* is a repeat of a previous one (Greenberg, 1993a). An activity is loosely defined as "the formulation and execution of one or more actions whose result is expected to gratify the user's immediate intention (Greenberg, 1993a, p. 65)." In Web browsing, the activity of interest is a navigation event—any user action that results in the display of a Web page within the browser window. *Total activities* is the count of all URLs navigated to by the user, whereas *different activities* count only those URLs that are different (or unique). The *recurrence rate R* over a set of user activities is thus calculated as:

$$R = \frac{total\ activities - different\ activities}{total\ activities} \times 100\%$$

Although many Web pages are revisited, new pages are constantly added to the repertoire. The rate at which new activities are composed and introduced to the dialog is the *composition rate C* (Greenberg, 1993a):

$$C = \frac{different\ activities}{total\ activities} \times 100\% = 100 - R$$

Greenberg (1993a) coined the term *recurrent system* to identify those systems in which users predominately repeat activities they had invoked before. Recurrent systems are open-ended in that there are many activities possible but most activities tend to be repetitions of previous ones. Greenberg (1993c) cites the following typical characteristics of recurrent systems:

- many activities are repeated with *R* ranging between 40-85%;
- new activities are incorporated regularly with *C* ranging between 15-60%;
- a small subset of the total activities available are used;
- frequency of repetition for items varies though *R* tends to be constant over time;
- recently submitted items are more likely to be repeated;
- the sets of activities invoked by different users are mostly disjoint;
- common activities are repeated by different people at different rates.

In our study, the recurrence rate was calculated for each of the 23 subjects. Only navigation events were considered as we are interested in revisits to Web pages, not other browser actions (see Table 3.2). Of these, only jumps to different pages were counted, with jumps within a page removed from the analysis. For example, if the user generated three Open URL navigation actions as follows:

http://www.foo.com/foo#one

http://www.foo.com/foo#two

http://www.foo.com/foo#three

the internal anchor (e.g. *#one*) was removed, so that the three jumps were considered to occur on the same page (*http://www.foo.com/foo*).

4.2.2 Results

Table 4.6 summarizes the mean, standard deviation, median, minimum and maximum values for the overall recurrence rate and composition rate. The mean recurrence rate of 58% falls within the 40-85% range for recurrent systems, as defined by Greenberg (1993a). To show the individual variation, Figure 4.2 plots the recurrence rate observed at

	Log Statements	Navigation Events	Total URLs	Recurrence Rate	Composition Rate
Mean	910.87	90.46	817.09	57.98	42.02
SD	675.99	4.16	606.88	8.79	8.79
Median	701	90.68	632	57.33	42.67
Min	303	82.52	251	42.50	25.73
Max	3299	97.57	2926	74.27	57.50

Table 4.6. Recurrence and composition rates for navigation events

the end of the study period versus the total number of URLs visited for each subject. This graph displays the spread in the browsing activity of subjects (251 to 2926 total URL visitations), their recurrence rates (43% to 74%), and the relationship between these two variables.

Is recurrence rate affected by the extent of browsing activity? A simple correlation analysis was performed by contrasting the total URLs visited and the recurrence rate for each subject. A statistically significant correlation was not found ($r = 0.484$, $df = 21$, $p < .01$). Consequently, R is considered independent of the total URLs. This independence is important because if the recurrence rate for a user did change with activity levels, the history mechanism must be designed to adapt to this. This, however, is not the case.

4.2.3 Discussion

This section assesses the frequency of revisits to Web pages using the concept of recurrence rate. We will compare our findings to the C & P data (our own calculations), and results from other domains. We will also present qualitative findings—the various reasons given by our subjects for both revisiting pages and visiting new pages.

An overall recurrence rate of 58% shows that users do revisit Web pages. They do not fixate on a small set; they incorporate new pages into their repertoire as well. This seems to be a generalizable result, for we also analyzed the C & P data and found similar results. Our analysis was based upon 55 of their subjects (see Section 4.1.3). We found a mean recurrence rate of 61% (SD = 9%) with the extremes ranging from 39% (153 log statements) to 84% (652 log statements).

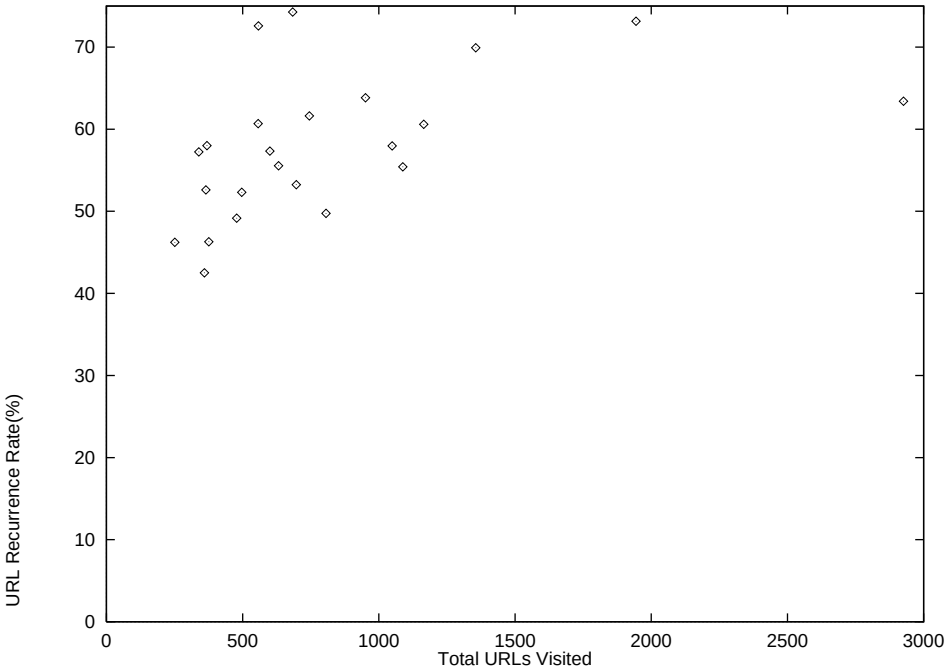

Figure 4.2. Recurrence rate (%) versus total URLs visited by each subject

Repetition within Web browsing compares favourably to other domains, particularly telephone usage and information retrieval (Greenberg, 1993a). However, our results are lower than Greenberg's study of Unix command lines which showed a 75% recurrence rate. As outlined in Section 1.4, there are several differences between these two domains and we now know that this has an impact upon the recurrence rate.

Post-study interviews gave us the opportunity to learn more about users' reasons for visiting new pages, and revisiting old ones. The following are reasons users gave for revisiting pages:

- to access information that changes on a regular basis;
- to further explore the information on a page, or to remind themselves of the contents;
- to access a search engine or jumping-off page that can lead to a page of interest;
- to review pages one is authoring;
- startup document or home page;
- the page is on a path to another revisited page.

Users stated that they visit new pages because:

- information needs change due to changing work demands or personal interests;
- they notice something interesting while browsing for another item (serendipitous browsing);
- they learn about a new site from a colleague, newsgroup, newsletter, etc.;
- they explore new information accessible from previously visited pages that was not explored exhaustively.

4.2.4 Concluding Remarks

This section assessed the extent of recurrences for Web browsing. The 58% recurrence rate that we found qualifies Web browsing as a recurrent system. We also calculated *R* for the Catledge and Pitkow data and obtained similar results (61%). During post-study interviews, we found that people revisit pages because the information changes, they wish to explore the page further, the page has a special purpose (e.g. search engine, subject index), etc. People visit new pages because their information needs change, they wish to explore more of a particular site, a page is recommended by a colleague, etc.

User interfaces that exhibit the properties of recurrent systems should be designed to take advantage of this potential for activity reuse, and thus reduce the cognitive and physical effort of specifying future activities. The key is to give preferential treatment to the large number of repeated actions. This involves identifying patterns in history use. Chapter 5 presents an analysis of five patterns that we examined.

5. Patterns that evolve over time

This chapter presents analyses of the WWW usage study data that assesses navigation patterns that evolve over time. The first section addresses the growth of URL vocabulary, or how new pages are incorporated into the user's repertoire of visited pages. The second section deals with recurrence rate as a function of the distance between revisited pages. The third section concerns a popular pattern—frequency—as it applies to URL accesses. Following this is an analysis of locality that replicates similar research applied to Unix command lines. The final section addresses longest repeated subsequences, or paths of URL visits.

5.1 Growth of URL vocabulary

Do users extend their vocabularies continuously and uniformly over the duration of their browsing episodes? Greenberg (1993a) addressed this question for Unix command line usage and found that the formation of new command lines was linear and regular. We expect the same results in Web browsing because people's information needs change, and they learn about new Web pages from various sources as well. We also believe that examining the rate of change in local patterns of vocabulary growth as well as the navigation actions involved will provide insight into the nature of individuals' browsing patterns. Understanding how Web browsers are used and for what purposes are crucial factors in effective browser design.

5.1.1 Analysis method

The recurrence rate identifies the probability that the next URL visited has already been seen by the user. It is based on the total URLs visited, and the number of unique URLs—or the URL *vocabulary*. We know that over all subjects, there is a 58% probability that the next URL selected has already been visited by the user. Now we must examine whether users extend their vocabularies continuously and uniformly over the duration of their browsing episodes. From the correlation analysis of recurrence rate versus URLs visited (Section 4.2.2), we know that these two variables are independent. A

second and perhaps more convincing method of observing their independence is by examining in detail the URL vocabulary growth of our subjects. This type of analysis, performed at an individual level, can highlight both differences and similarities in subject's browsing behaviour.

URL vocabulary plots were generated for each subject. Each plot contains a curve (*All*) representing the overall URL vocabulary; for each URL visited, the number of unique URLs visited at that point is plotted. Major navigation actions are also plotted as separate curves shifted above the vocabulary line by a constant amount. These navigation actions are: *Open URL*, *Back*, *Reload*, *Forms*, *Helper Applications*, and *Other*. The *Other* category includes all remaining navigation actions. Taken together, these 6 curves comprise the *All* curve. The additional curves are intended to show approximately when the most common navigation actions were invoked; they were discussed in detail in Section 4.1.2.

5.1.2 Results

Plots of three subjects are shown as Figures 5.1, 5.2, and 5.3. These plots are typical of most subjects, although they have been chosen because they illustrate interesting aspects of URL vocabulary growth. There are similar patterns evident in all graphs as well as variation in the nature and extent of patterns for particular subjects.

The most striking similarity across all 23 subjects is the regular increase in URL vocabulary as portrayed by the approximately linear shape of the *All* curve. The slope of this curve is roughly equal to each subject's composition rate, *C* (see Section 4.2.1). Its shape clearly indicates that new pages are continuously and regularly being added to the user's repertoire of pages visited.

While there is an overall linearity in the vocabulary curve, local variations are also evident, the nature and extent of which vary amongst individuals. These local variations (and their concomitant navigation actions) highlight several browsing patterns.

- *First-time visits* to pages produces a steeper sloped area on the curve.
- *Revisits* to pages produces an area of horizontal slope.

- *Hub-and-spoke* navigation involves a series of revisits to an index or table-of-contents page to access the next hyperlink on that page. It will appear moderately sloped on the vocabulary curve if this is the first visit to these pages. C & P (1995) noticed hub-and-spoke navigation patterns in their data. They discovered that users tended to browse in one small area within a particular site, and made frequent use of backtracking to return to a page from which they would explore additional links.
- *Guided tour* navigation involves exclusive use of the *Open URL* action as the user selects *Next* and *Previous* links on the page; the slope will vary depending on the number of first time visits versus revisits.
- *Depth-first search* involves a series of *Open URL* actions followed by a series of *Back* actions; the curve appears as a *step* with an initial steep area followed by a level area.
- *Authoring* is usually apparent when a cluster of *Reload* actions occur, as the user redisplays the page after modifying the HTML source; the slope of this area will be horizontal since the same page is being revisited.
- *Web-based applications* typically use forms to request input from the user, so a cluster of *Submit Form* and *Back* actions will appear; the slope appears horizontal since the same pages are usually reaccessed extensively.

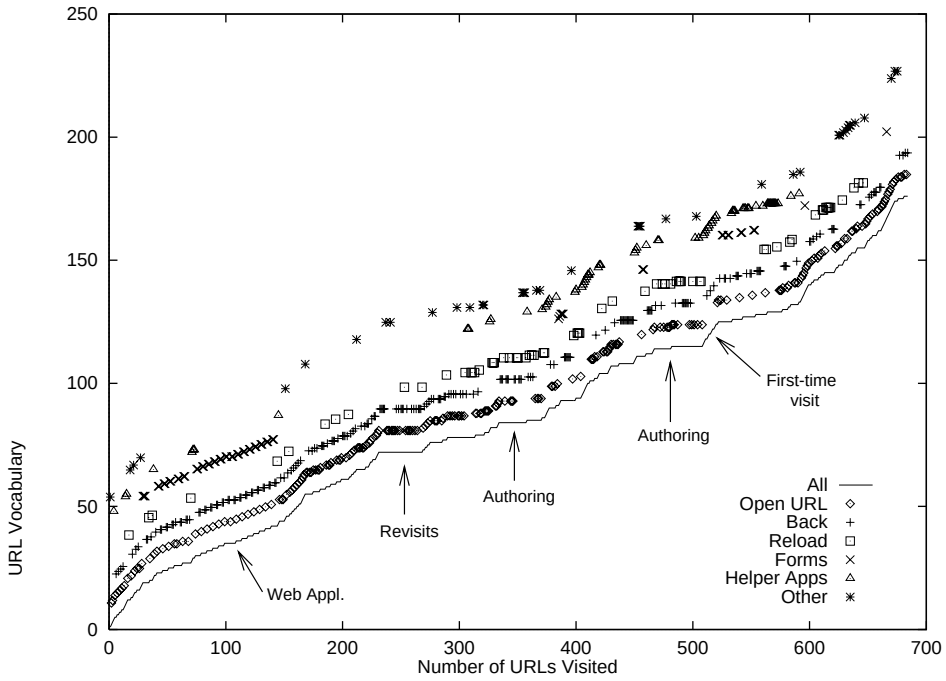

Figure 5.1. URL Vocabulary for subject15

Several interesting patterns are evident in Figure 5.1, and during the post-study interview we explored their meaning with the subject. First, the subject used a Web-based application (a custom-developed knowledge elicitation tool) between URLs 50 to 150 (x-axis) where there is an initial moderately sloped area with a combination of *Open URL*, *Back* and *Forms* activity. Authoring is evident at the plateau (horizontally sloped) areas where the subject used *Reload,* for example, URLs 480 to 510. Plateau areas in combination with *Back* or *Open URL* actions show revisits to pages as when the subject reviewed online course notes at URLs 240 to 280.

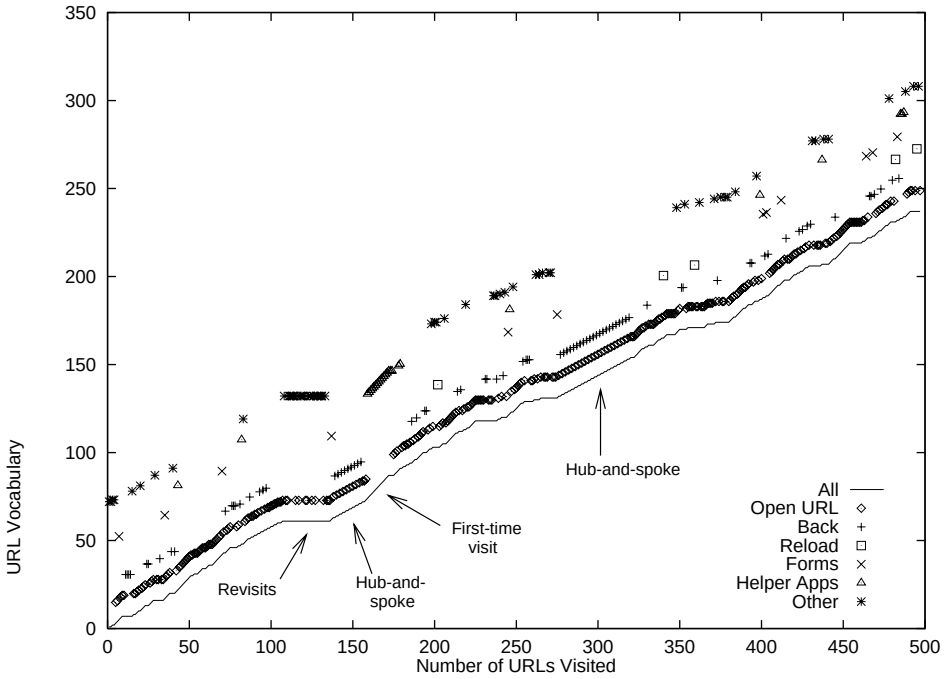

Figure 5.2. URL Vocabulary for subject17

Figure 5.2 shows evidence of hub-and-spoke navigation, and first time visits. An excellent example of the hub-and-spoke pattern is located at URLs 280 to 320 (x-axis). *Open URL* and *Back* actions were consistently used to return to an index page to access another URL on the page. This user also visited an extended set of pages for the first time at URLs 160 to 175 as evidenced by the curve's steep slope. These URLs were images of a popular daily comic strip that the subject viewed with an external viewer.

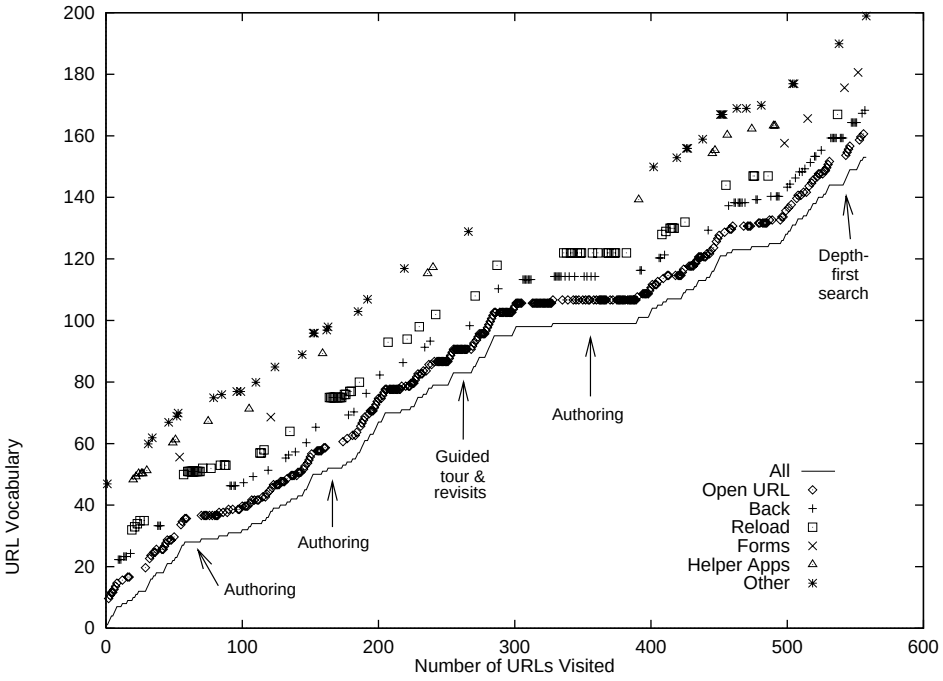

Figure 5.3. URL Vocabulary for subject23

Figure 5.3 shows an extensive authoring activity, the guided tour pattern, and an example of a depth-first search. The plateau and numerous *Reload* actions between URLs 300 to 400 indicate authoring. A short plateau between URLs 245 to 265 represents navigation through a set of pages using *Next* and *Previous* hyperlinks (as opposed to the hub-and-spoke style) embedded into the Web pages authored by the subject. The horizontal slope during this guided tour indicate that these are revisits to the pages. Between URLs 547 to 557, the subject performed a search and then used *Open URL* to consecutively navigate through the four results pages; they then applied consecutive *Back* actions to return to the search form. This is an example of a depth-first search browsing pattern.

Does URL vocabulary size depend on the number of URLs visited? This seems to be the case by qualitatively viewing individual graphs. We also did a simple regression analysis to contrast the total URLs visited by each subject with the number of unique URLs visited. The regression line is plotted in Figure 5.4, where each point in the scattergram represents the values observed for each subject as the end of the study period.

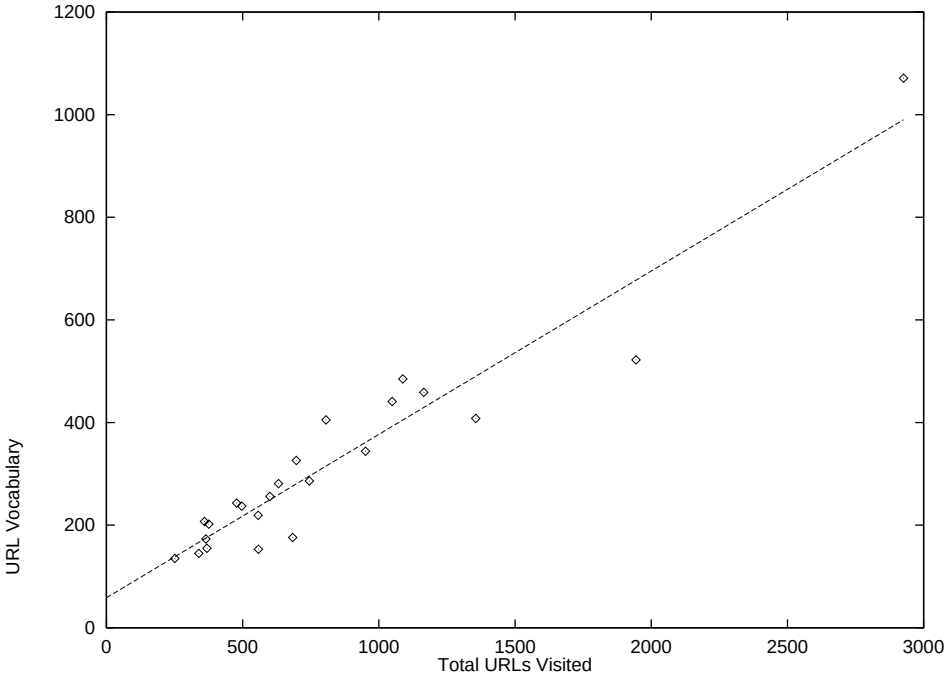

Figure 5.4. Regression: URL vocabulary size for each subject

A statistically significant and strong correlation was found (r = .955, df = 21, p < .01). The moderate slope (C = 32%) of the regression line makes the correlation of practical significance as well. We conclude that new URLs are visited regularly by users.

5.1.3 Discussion

Analysis of URL vocabulary (the rate at which new pages are incorporated into the repertoire of visited pages) reveals striking similarities across all subjects with interesting local variations in browsing patterns.

URL vocabulary growth curves for all subjects possess a definite linear and positive slope. These data and interview results indicate that browsing includes both URL revisits and new visits, and that new URLs are incorporated into the repertoire at a regular rate. Greenberg's (1993a) analysis of Unix command line vocabulary growth showed the same trend.

Local variations in the vocabulary rate and changes in the use of various navigation actions indicate the presence of different browsing patterns. Seven patterns were proposed and illustrated in Section 5.1.2. They are summarized as follows:

- plateaus or horizontally sloped areas for any navigation action indicate revisits to many URLs;
- steeply sloped areas indicate a series of pages are being visited for the first time;
- plateaus in conjunction with *Reload* actions suggest authoring activity, or updates to *live* pages whose content changes frequently;
- plateaus in conjunction with *Forms* indicate use of forms-based Web applications (e.g. custom-made application);
- alternating use of *Open URL* and *Back* indicate hub-and-spoke navigation episodes where the user backtracks to a particular page several times; slope is moderate at this point if this is the first visit;
- a series of consecutive *Open URL* actions followed by a series of *Back* actions indicate a depth-first browsing strategy; if this is a first time visit, the curve will have a step-like appearance;
- a series of consecutive *Open URL* actions may indicate a guided tour or discussion group where *Next* and *Previous* hyperlinks have been 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.

History mechanisms should consider the variety of browsing patterns users exhibit, and they currently do to some extent. For example, the hub-and-spoke pattern and the depth-first search pattern can work well with the stack-based history mechanism and the *Back* action found in most browsers. Authoring is well-supported by the presence of a *Reload* button on the toolbar. Guided tours contain hyperlinks that encourage a linear pattern of navigation. However, improvements in history designs can be made. For example, excessive backtracking that results from the hub-and-spoke and depth-first navigation styles could be reduced by a graphical overview map or by retaining the index page within the browser window as can be done with Netscape Navigator 2.0's *frames* feature.

While Web pages are recalled, new pages are incorporated at a regular rate. Thus, removing pages that are unlikely to be revisited is crucial to reducing information overload in a history system. The *pruning* method should also ensure that the remaining items have

a high probability of being on the list when the user wants them. This warrants an investigation into the distance between recurrences, which is the topic of the next section.

5.2 URL recurrence rate as a function of distance

For any URL visited, the probability that it has already been seen by the user is quite high (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 mechanism displayed the previous *p* entries as a list, what is the probability that this includes the next entry (Greenberg, 1993a)?

Recency is one perspective of recurrence in which recent, individual user interactions are referenced repeatedly during a user session (Lee, 1992). Two studies performed by Greenberg (1993a) indicate that the most recently submitted activities are the most likely to be repeated. Greenberg (1993a) found that telephone numbers that have just been dialed are more likely to be repeated than those dialed long ago. For the most prolific caller studied, 41% of all calls are repeats of one of the previous ten dialed. Thus, the probability of an item recurring is related to its recency of selection. Greenberg's (1993a) study of Unix command lines revealed similar results with a 47% chance that the next command line issued by the user would appear in a standard sequential history list containing the ten previous command lines.

Given the high use of the back action in Web browsing (30% of navigation events), we expect that there will also be a recency effect for recurrences in this domain. Current browser history lists partially base their pruning mechanism on the recency of a URL visit. Quantifying this effect will provide insight into the effectiveness of this method, and suggest a reasonable list length for presenting the most recently accessed URLs.

5.2.1 Analysis method

The recurrence distribution as a measure of distance was calculated for each subject. The algorithm to calculate the recurrence rate (R) at a given distance (d) for a single user (s), $R_{s,d}$, can be found in Greenberg, 1993a, and is summarized in Figure 5.5.

Given

- a trace of URLs numbered from 1 through 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

Average the number of recurrences at a particular distance

- for (distance := 1 to n)
  counter[distance] := (counter[distance]/n) * 100;

Figure 5.5. Algorithm to calculate $R_{s,d}$ (Greenberg, 1993a)

The mean recurrence rate distance *d* over all *S* subjects is then calculated as (Greenberg, 1993a):

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

5.2.2 Results

5.2.2.1 Pooled data

Table 5.1 summarizes the mean, standard deviation, median, minimum and maximum values for the probability of a recurrence at a given distance in percent. Figure 5.6 plots this data up to a distance of 50 for all subjects. The horizontal axis shows the position of the repeated URL on the history list relative to the current one. The vertical axis represents the rate of URL recurrences, R_d.

According to Table 5.1 and Figure 5.6, there is a R_{d1} = 9.8% probability that the current URL is a repeat of the previous URL (distance = 1), R_{d2} = 18.8% for a distance of 2, R_{d3} = 2.3%, and so on. As illustrated in Figure 5.7, a distance of 1 occurs in Web navigation when the user reloads the current page, or successfully interrupts the transmission of a page. 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.

	1	2	3	4	5	6
Mean	9.97	18.81	2.25	4.93	1.07	2.47
SD	4.89	4.21	1.14	1.04	0.90	0.85
Median	9.26	18.83	1.80	4.69	0.83	2.48
Min	2.83	11.92	0.50	3.22	0.27	0.80
Max	19.39	27.51	5.26	7.02	4.68	3.84

	7	10	20	30	40	50
Mean	0.88	0.91	0.39	0.17	0.12	0.07
SD	0.52	0.43	0.34	0.24	0.29	0.11
Median	0.83	1.09	0.29	0.10	0.00	0.00
Min	0.00	0.00	0.00	0.00	0.00	0.00
Max	1.77	1.33	1.20	0.77	1.36	0.40

Table 5.1. Probability of a recurrence at the given distance *d* in % (R_d)

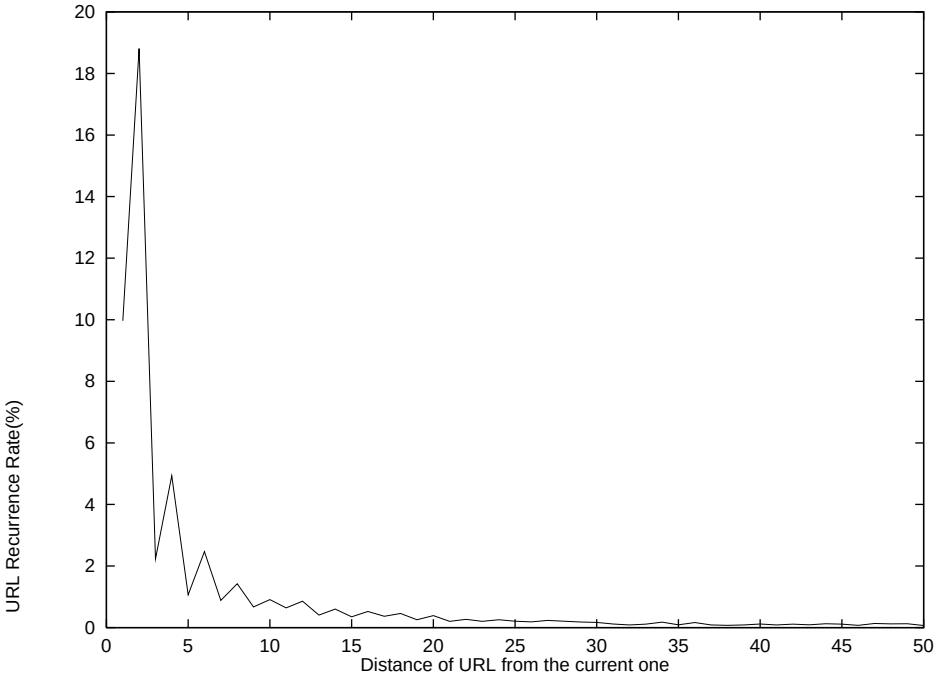

Figure 5.6. URL Recurrence rate as a function of distance (all subjects)

However, the most striking feature of the data is the extreme recency of the distribution. The previous 6 or so URLs contribute the majority of recurrences. Although the probability values of R_d continually decrease after the second item, the rate of decrease and low values make all distances beyond the previous 8 or so items equivalent for

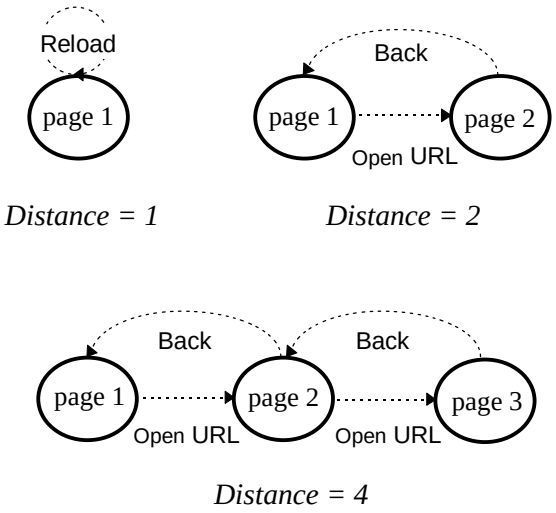

Figure 5.7. Distances generated by *Reload*, *Open URL* and *Back* actions

practical purposes. The actual mean recurrence rate for any distance greater than 8 is less than 1%.

This is illustrated further in Table 5.2 and Figure 5.8, which report the same data for all subjects as a running sum of the probability. In Figure 5.8, the horizontal line at the top of the graph is the maximum mean recurrence rate for all subjects of 58%. The most recently visited URLs are responsible for most of the cumulative probabilities. For example, there is a 39% chance that the next URL visited will match a member of a set containing the 6 previous submissions. In comparison, all further contributions are minimal (though their sum total is not).

5.2.2.2 Individual data

The pattern of spikes for distances that are multiples of two is evident in the individual plots for 19 of the 23 subjects (though the percentage of URLs which recur within a distance of 6 varies from approximately 26% to 53%). The remaining 4 subjects display a greater recurrence rate for a distance of 1 versus a distance of 2. Figure 5.9 contrasts the two patterns by displaying curves from 2 subjects.

	1	2	3	4	5	6
Mean	9.97	28.78	31.01	35.94	37.01	39.48
SD	4.89	5.52	5.39	5.64	5.95	6.21
Median	9.26	30.20	32.32	36.22	37.83	39.24
Min	2.83	18.35	19.68	23.40	23.94	26.33
Max	19.39	42.90	44.55	49.59	50.46	53.29
	10	**20**	**30**	**40**	**50**	***R***
Mean	43.37	48.23	50.35	51.47	52.52	57.98
SD	6.71	6.79	7.07	7.27	7.43	8.79
Median	43.37	48.23	50.35	51.47	52.52	57.98
Min	28.99	36.97	39.72	40.83	41.39	42.50
Max	57.61	62.57	65.35	66.52	68.10	74.27

Table 5.2. Cumulative probabilities of a recurrence at the given distance *d* in % (R_d)

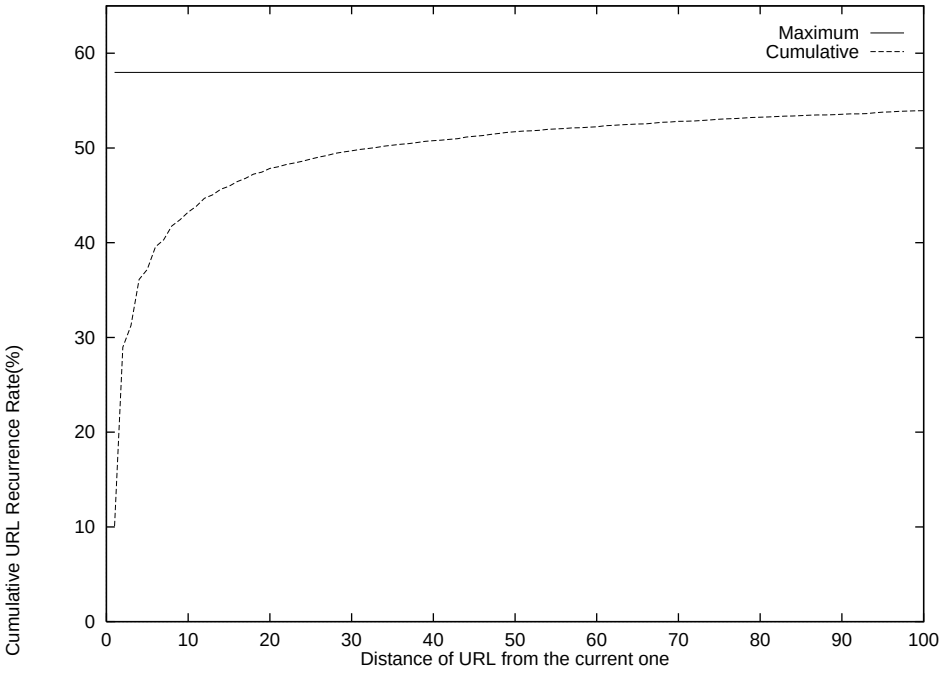

Figure 5.8. Cumulative URL Recurrence rate as a function of distance (all subjects)

For 3 of the 4 subjects the spike at a distance of 1 can be explained by the high use of *Reload* during authoring tasks. A combination of factors produced this pattern for the fourth subject; they had below average browsing activity, below average use of *Back*,

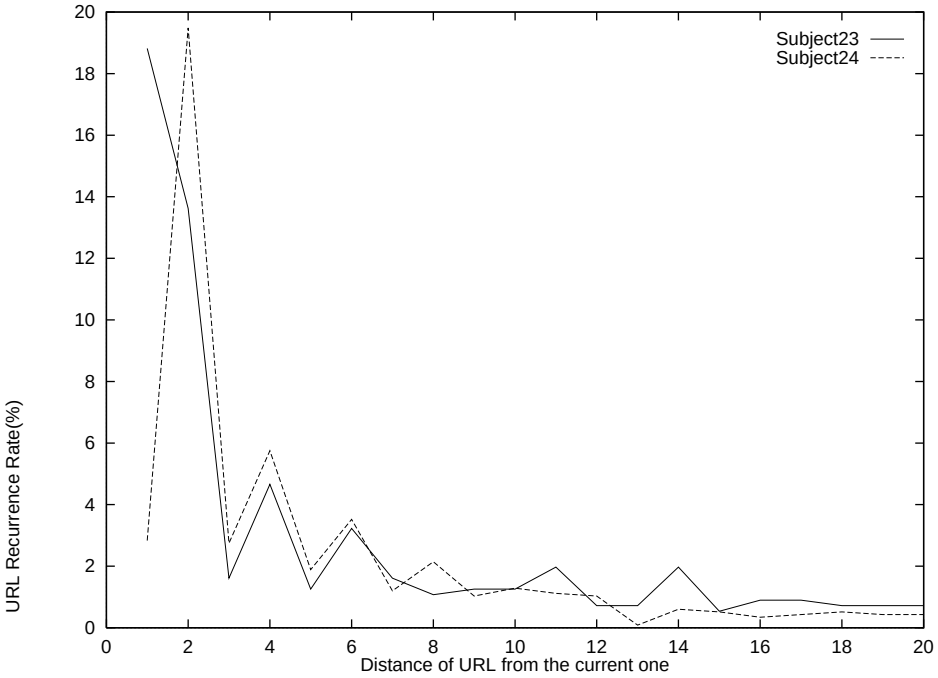

Figure 5.9. URL Recurrence rate as a function of distance for 2 subjects

often generated consecutive sequences of *Open URL* actions, sometimes browsed named anchors within a page, and showed above average use of *Window History*.

5.2.3 Discussion

There are two important findings from the analysis of URL recurrence rate as a function of distance: the extreme recency of the distribution, and two variations in recency patterns.

First, our results show that the most recently visited URLs are responsible for the majority of recurrences. For example, given a history list of the last 10 URLs visited, there is a 43% probability that the next URL the user visits will appear on that list. The maximum this value could be is 58%, the recurrence rate. This finding is similar to Greenberg's (1993a) Unix study in which there is about a 47% chance that the next command line issued will match one of the previous 10 commands.

These statistics confirm the potential of reuse facilities in general, and, according to Greenberg (1993a), verify the assumptions of recency made by most history mechanisms. The notion of temporal recency has a cognitive appeal in that users likely remember the

commands they have just entered, and will thus be able to predict whether the history mechanism will make those available for reuse. However, as our application of various conditioning methods shows (Chapter 7), there are more effective ways to present history than a strict sequential list.

Two variations in recency patterns were noticed for R_{d1} and R_{d2}. For most subjects, URLs at a distance of 2 generated the highest recurrence rate. High use of *Back* seems to explain the dominance of multiples of 2 on the overall distribution. However, a distance of 1 showed the highest recurrence rate for four subjects. Differences in browser use can explain the variation. One major factor was the reloading of pages during authoring tasks. *Reload* can be invoked by a simple button action on current browsers, an important feature as this data shows. Beyond R_{d1} and R_{d2}, the patterns between subjects are remarkably similar.

5.3 Frequency of URL accesses

Frequency is a popular method for ranking items of interest. Greenberg (1993a) reports that several investigators have examined the frequency of computer command use by a user population. All studies report results approximated by a Zipf distribution, which has the property that a relatively small number of items have high usage frequencies, while a very large number of items have low usage frequencies (Greenberg, 1993a). For example, when considered as a pooled statistic, Greenberg (1993a) found that in Unix command usage across four user groups, 10% of the commands used by each group accounted for 84-91% of all usage. Web browsing is a different task and the frequency distribution of page accesses by individuals is not known. If pages are frequently revisited, do they tend to be of a special type? In this section we examine the frequency distribution of URL accesses. Through user interviews, we also develop a taxonomy of frequently visited page types that may be useful in the design of browser history mechanisms.

5.3.1 Analysis method

Frequency of URL accesses was examined in two ways. First, we generated frequency graphs for each subject. Each graph plots the number of times each unique URL was

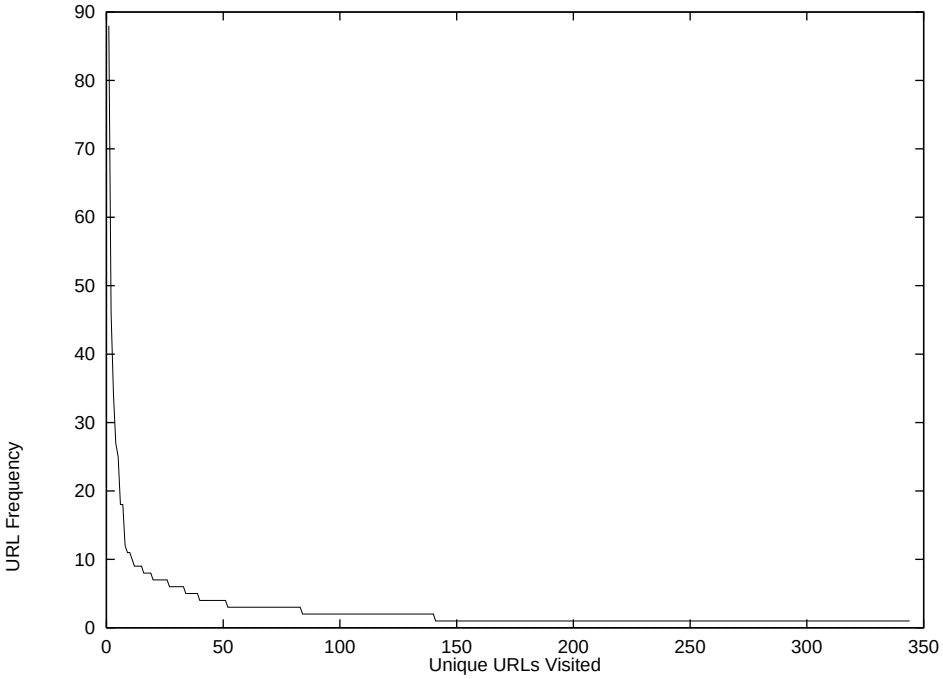

Figure 5.10. Frequency of URL accesses for subject16

visited for the duration of the study. Second, we interviewed each subject several weeks after the study about the type of pages they most frequently visit. Subjects were shown summary reports listing their top 15 URLs by frequency. They were asked to describe characteristics about these pages that explained why they were frequently accessed, and the rate at which they accessed them (e.g. browser startup, daily, weekly, monthly, as required, one-time access and will not return to the page, one-time access and may return to the page). From this, a preliminary taxonomy of conceptual page types was derived.

5.3.2 Results

Figure 5.10 shows the frequency of URL accesses for a typical subject (subject16). The horizontal axis represents the unique URLs visited (344 in this case); the vertical axis represents the frequency as a count of the number of times the page was visited. For example, the most frequently accessed URL for this subject was visited 88 times. Note the large number of URLs that were only visited once or twice. 204 of the 344 unique URLs (59%) were visited once, 57 (17%) were visited twice, 32 (9%) were visited three times.

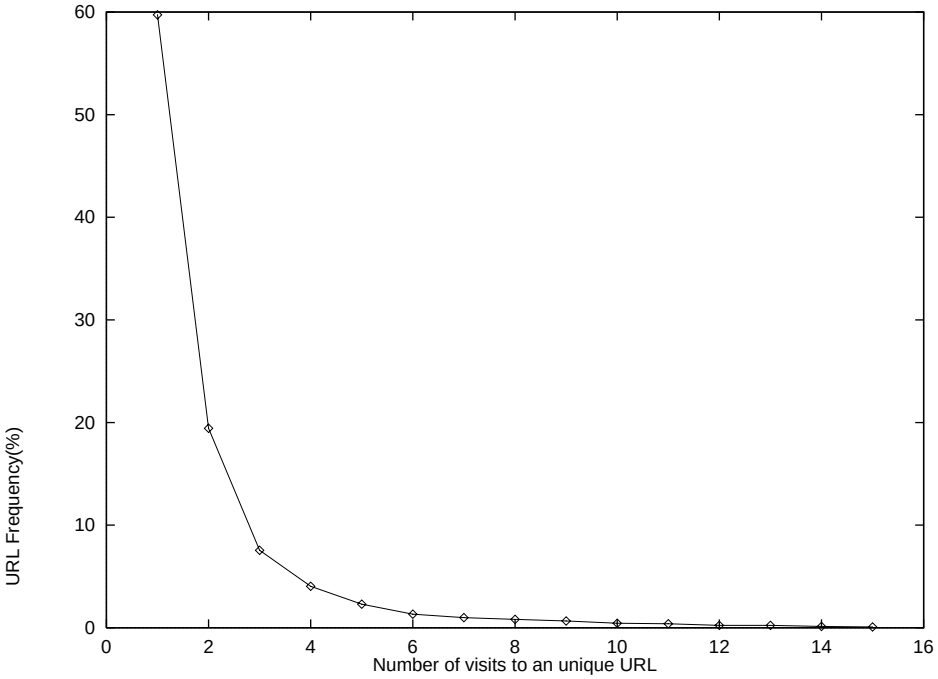

Figure 5.11. Frequency of URL visits for all subjects

Only 51 unique pages (15%) were visited four or more times. All subjects' frequency distributions resemble this typical case.

Figure 5.11 shows the mean percentage of URLs visited 1 through 15 times for all subjects. The purpose of this plot is to show the dramatic proportion of URLs that are only visited a few times. For example, 60% of URLs were visited once, 19% were visited twice, 8% were visited 3 times, and 4% were visited 4 times.

From interviewing each subject about their most frequently visited 15 URLs, the following taxonomy of frequently-visited conceptual page types is proposed. The taxonomy is based upon three elements: content of the page, owner/author of the page, and how the page is used by the user. The intent of the taxonomy is to classify a Web page as predominately one of these types. This exercise may provide insight into possible methods for organizing and presenting navigation history. This taxonomy is considered preliminary; additional research should be performed to verify the categories of page types.

- *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. 18 of 23 subjects had a personal home page that they accessed at least once during the study. For 9 subjects, the personal home page was the most frequently accessed page. For 7 subjects, their home page appeared either second, third, fourth or sixth on the list. For 2 subjects, this page did not appear on the top 15 list.
- *Start-up page*. The personal home page is often set by the user to be the browser's start-up document—the page that is automatically displayed when the browser is invoked. For 15 of the 23 subjects, the startup document was the most frequently accessed page. For the 8 corporate subjects, this page was set by the system administrator to the local corporate home page, though 2 subjects did choose to change this to their personal home page. The 15 University of Calgary subjects were permitted to set their startup document to whatever page they wished. 12 subjects used their personal home page; one used the University of Calgary Computer Science department home page; another subject used a group research home page they had authored; the final subject showed a variety of startup documents because he used a URL Launcher application that invoked the Mosaic browser with a particular URL that was entered into the launcher's dialog box.
- *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.
- *Index page*. This page contains links pertaining to a particular topic so it is a good starting point when the user is searching for information about that topic e.g. WWW Servers, newsgroup article list, Intel Web Server Table of Contents. It also acts as a gateway to other pages.

- *Search engine*. This page type includes Web-based applications that specifically assist the user in locating a Web page or site e.g. WebCrawler, OpenText, Lycos, or site-specific search engines.
- *Web application.* 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-English dictionary, and a documentation status interface.
- *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 the path the navigation page is on.
- *Authored page*. This type of page will be visited often during the authoring phase.

5.3.3 Discussion

There are two important findings from examining the frequency of URL accesses: frequency distributions indicate that only a small number of URLs are frequently accessed, and certain types of pages tend to be more frequently accessed.

All subjects produced a similar frequency distribution in which a small number of URLs are highly visited, while a very large number of URLs have very low usage frequencies. This finding agrees with previous research into the frequency of command usage.

Second, frequently visited pages tend to be of certain types based upon their content, purpose and author/owner status. A taxonomy of page types was proposed in Section 5.3.2 based upon post-study interviews with subjects about their top 15 most frequently accessed pages. What the frequency reports do not show is the regularity of access. That is, some URLs have a high frequency count because they are visited on a regular basis while other URLs may be revisited several times during the same browsing session but are no longer of interest after that. Page types that tend to be accessed on a regular basis include the following: personal home page, index page, and search engine. Other page types are accessed intensely for a brief period. For example, the user may update an

authored page twice a year or use a document status Web application at the end of a project. In the latter cases, frequently visited pages are of little use once the need for the information ends. A history mechanism that presented the user's most frequently accessed pages might thus include items that the user is no longer interested in.

5.4 Locality

Locality is another perspective on recurrence in which a user repeatedly references a small and related set of user interactions (Lee, 1992). While recency characterizes recurrences in terms of distance, locality characterizes recurrences in terms of periods of time where references are made solely to a small and related group of user interactions (Lee, 1992). That is, for locality to exist, user interactions must exhibit a temporal and spatial bias.

Lee (1992) took this concept from computer program memory reference research, and reapplied it to Unix command lines. Lee found that locality does exist in a user's command line trace, especially for small set sizes. However, the mean number of occurrences of locality sets for even small set sizes was low. Also, the *locality rate*—proportion of items that exhibit locality—for command line references was poor compared to program memory references (Lee, 1992 reports that program memory reference exhibit locality rates of > 90%). Web browsing, however, could show locality since it is reasonable to think that clusters of pages may be revisited.

Finally, Lee concludes that user activities that are reused in a history facility are attributed to locality. Within Greenberg's Unix command-line data, 65% of history tool usages occurred within locality periods. Lee surmises that the 35% of history usages that cannot be accounted for are due to the modification and recall of parts of previous command line but this is actually an infrequently used feature of *csh* history. Also, it can be argued that history use is due to repetition of activities. And, within the command-line data, history usage accounted for only 4% of all user interactions, which is likely due to the difficulty in using Unix history because it is known that the recurrence rate for Unix commands is 75%.

Visit no.	Unique URL no.	URL	Navigation Action
1	1 -	www.cpsc.ucalgary.ca/projects/grouplab/coop/faq.hml	Open URL
2	1 -	www.cpsc.ucalgary.ca/projects/grouplab/coop/faq.html	Reload
3	2 -	www.cpsc.ucalgary.ca/projects/grouplab/	StartUpDoc
4	3 -	www.cpsc.ucalgary.ca/projects/grouplab/coop/coop.html	Open URL
5	3 -	www.cpsc.ucalgary.ca/projects/grouplab/coop/coop.html	Reload
6	3 -	www.cpsc.ucalgary.ca/projects/grouplab/coop/coop.html	Reload
7	4 -	www.cpsc.ucalgary.ca/projects/grouplab/coop/workterm.html	Open URL
8	5 *	www.cpsc.ucalgary.ca/projects/grouplab/coop/cover.html	Open URL
9	4 *	www.cpsc.ucalgary.ca/projects/grouplab/coop/workterm.html	Back
10	3 *	www.cpsc.ucalgary.ca/projects/grouplab/coop/coop.html	Back
11	1 *	www.cpsc.ucalgary.ca/projects/grouplab/coop/faq.html	Open URL
12	3 *	www.cpsc.ucalgary.ca/projects/grouplab/coop/coop.html	Back
13	2 *	www.cpsc.ucalgary.ca/projects/grouplab/	Back
14	3 *	www.cpsc.ucalgary.ca/projects/grouplab/coop/coop.html	Forward
15	4 *	www.cpsc.ucalgary.ca/projects/grouplab/coop/workterm.html	Open URL
16	5 *	www.cpsc.ucalgary.ca/projects/grouplab/coop/cover.html	Open URL
17	6	www.cpsc.ucalgary.ca/projects/grouplab/481/guidelin.html	Open URL

Table 5.3. An example locality set of size 5, phase length 9(*) for subject23

5.4.1 Analysis method

Lee (1992) reports that *locality* is the phenomenon in which a program's memory references are limited to a small subset of *segments* (portions of a computer's memory) for an extended time interval. Locality thus has two components: a favoured subset of segments (i.e. a *locality set*) and a definite reference interval (i.e. a *phase*) over which this favoured subset remains unchanged. Locality sets and *working sets* (the chronological, sequential history of commands) differ in that the latter includes command references that appear in phases as well as the transitions between phases. The locality detection algorithm applied is described in Lee (1992) who developed it from Madison and Batson (1976).

In the domain of Web browsing, the locality set members are URLs that the user has visited. The sequential list of URLs visited by the user over time, ignoring boundaries between login sessions, is analyzed for sequences of URLs that constitute a locality set. A set of URLs is a locality set if and only if all its members are referenced at least once after

the set is formed and no URLs are referenced which are not in the set. Its *phase* is the maximal interval over the URLs visited in which all references made after the formation of the locality set are only to URLs in the locality set. Thus, the minimum length of a phase will be the locality set size.

Table 5.3 contains an actual example of a locality set extracted from subject23's browsing session. The first column, *Visit no.*, identifies each access to a URL in the browsing sequence; 17 accesses are displayed. A *Unique URL no.* is listed in column 2 to identify each unique URL; there are 6 different URLs in this example. The "-" symbol after the Unique URL no. shows phase formation which occurs from Visit no. 1 to Visit no. 7. At Visit no. 8, the fifth and final member of the locality set is accessed for the first time; this event marks the beginning of the phase for this locality set. The phase is 9 URL visits in duration (denoted by the "*" symbol in column 2)—it is in effect from Visit no. 8 to Visit no. 16. The phase for the set ends when subject23 accesses a URL outside of the set (URL no. 6) at Visit no. 17. This locality set contains 5 unique URLs, so it has a set size of 5.

Several methods were used to analyze the nature and extent of locality within Web browsing.

5.4.1.1 Locality rate

Locality rate represents the extent of locality in user sessions, and is calculated as (Lee, 1992):

$$R_{locality} = \frac{number\ of\ locality\ activities}{number\ of\ activities} \times 100\%$$

where *number of locality activities* is the number of activities in a session appearing in phases. Within the domain of Web browsing, the *number of activities* represents the number of URLs visited, and the *number of locality activities* represents the number of URLs visited that belonged to a locality set. Locality rate was calculated for each subject, and the mean over all subjects was also derived.

5.4.1.2 Locality set size

The mean number of locality sets of a particular size was calculated for each subject. The mean and standard deviation for these values was also calculated for all subjects.

5.4.1.3 Phase durations

The average phase or duration of all sets of a particular size was calculated. This indicates how long the user remained within the locality set. Two graphs were produced to visually represent this data.

- *Phase durations per locality set size* plots each locality set ordered by set size for its duration over the URLs visited. This plot shows where locality sets existed during the subject's browsing timeline, the phase length of each set, and the size of each set.
- *Mean length of locality set phases* shows how the phase length increases as the locality set size increases. It is based on the data from all 23 subjects. Recall that the minimum phase length will equal the locality set size. Phases show how long the user remained in the locality set before a reference to a non-set member occurred.

5.4.1.4 Locality set recurrences

The extent of locality set recurrences was assessed by reporting the number of unique sets versus the total number of locality sets. Two graphs were produced to visually represent this data:

- *Locality sets by locality set number* clearly identifies which locality sets were repeated, and where in the user's browsing timeline these recurrences occurred. This was accomplished by assigning each unique locality set a set number; the plot thus shows the number of unique sets generated, and when and how often the sets were repeated.
- *Total and unique occurrences for each locality set size* shows how many sets are repeated and how many sets occur only once during the study period. It is based on the data from all 23 subjects.

5.4.1.5 The activities that comprise locality sets

Reports for each subject were produced that listed each unique locality set found, where it occurred, and the URLs comprising the set. Larger sets, frequently repeated sets,

	1-U	1-T	1-D	2-U	2-T	2-D	3-U	3-T	3-D	4-U	4-T
Mean	36.35	45.91	2.78	18.78	21.52	4.57	6.65	7.04	6.57	2.83	2.96
SD	23.36	39.96	0.52	23.21	26.27	1.03	9.04	9.95	4.68	4.11	4.25
Mdn	20	31	2.70	8	11	4.28	3	3	5.25	2	2
Min	8	12	2.13	2	2	3.25	0	0	0	0	0
Max	116	186	4.20	100	110	7	36	38	23	18	19
WMn			2.87			4.36			5.74		
WSD			0.52			1.05			4.70		

	4-D	5-U	5-T	5-D	6-U	6-T	6-D	7+U	7+T	7+D	
Mean	6.85	1	1	3.34	0.43	0.43	1.90	0.96	0.96	2.85	
SD	3.92	2.04	2.04	4.41	1.20	1.20	4.33	2.58	2.58	5.71	
Mdn	6.33	0	0	0	0	0	0	0	0	0	
Min	0	0	0	0	0	0	0	0	0	0	
Max	15.5	9	9	11.75	5	5	13.7	11	11	18.3	
WMn	8.72			8.70			11.1			13.6	
WSD	3.17			1.90			2.47			3.40	

Table 5.4. Locality set frequency (Unique, Total) and avg. Duration for set sizes 1-7+

and sets of longer duration were discussed with each subject during the post-study interview. The objective was to identify the types of browsing behaviours that may give rise to significant locality sets.

5.4.2 Results

Table 5.4 summarizes the mean, standard deviation, median, minimum and maximum locality set occurrences and their average duration across the 23 subjects. Locality set sizes of 1 through 6 are reported in the first 6 major columns; the last columns contain the data for sets of size 7 and larger. Three values are reported for each set size: the number of unique sets (U), the total number of sets (T), and the average duration or phase length (D). The weighted mean (WMn) and weighted standard deviation (WSD) are also presented for the average durations. These values are calculated by averaging the product of the total number of sets and their duration for each subject (as opposed to simply averaging the durations for each subject). It is important to report WMn and WSD at larger set sizes since subjects are more likely to report zero occurrences at larger sizes, and this will produce a lower mean duration (see Table 5.4).

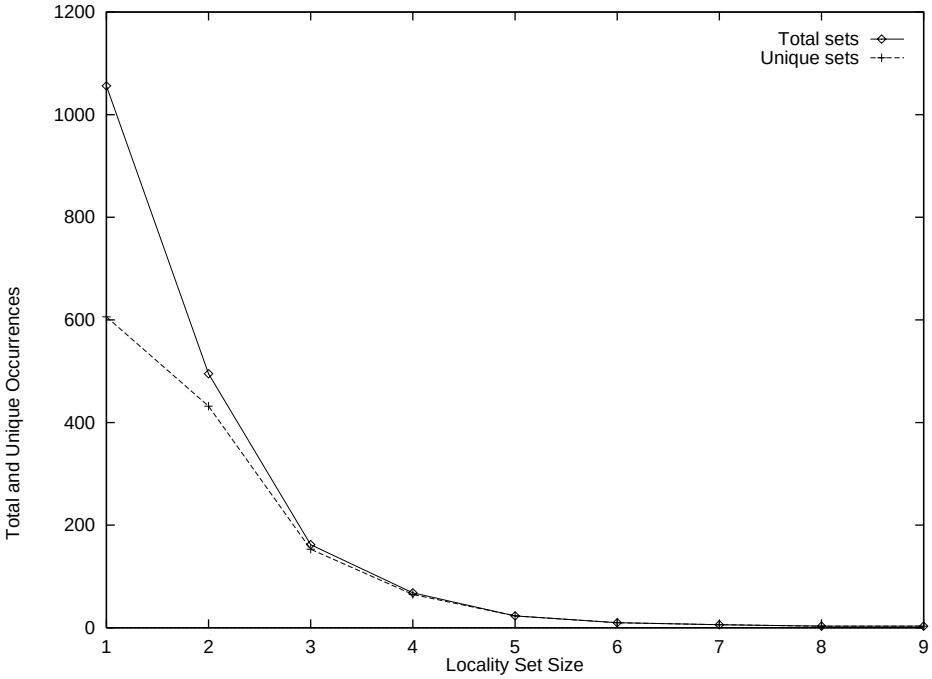

Figure 5.12. Total versus unique locality sets per set size

Table 5.4 shows that locality sets of various sizes did present themselves in the WWW study data. All subjects had locality sets of size one URL with a mean of 46 occurrences per subject. All subjects also had locality sets of size two URLs with a mean of 22 occurrences. 22 of 23 subjects had sets of three URLs with a mean of 7 occurrences. Beyond this, the number of occurrences drops dramatically with set size. Also, there is a great deal of variability in the number of sets identified, as indicated by the standard deviations of 40, 26, and 10 for set sizes 1, 2, and 3 respectively. This wide variation in locality set occurrences can be partially explained by the wide variation in browsing activity (see Section 4.1.2).

It is also important to examine how many locality sets were repeated. 79% of sets of size 1 were unique (and thus never repeated), 87% of sets of size 2 were unique, and 94% of sets of size 3 were unique. These results highlight another trend: as set sizes increase, their repeatability drops dramatically. Figure 5.12 displays the relationship between total and unique sets graphically. The horizontal axis plots the locality set size while the vertical

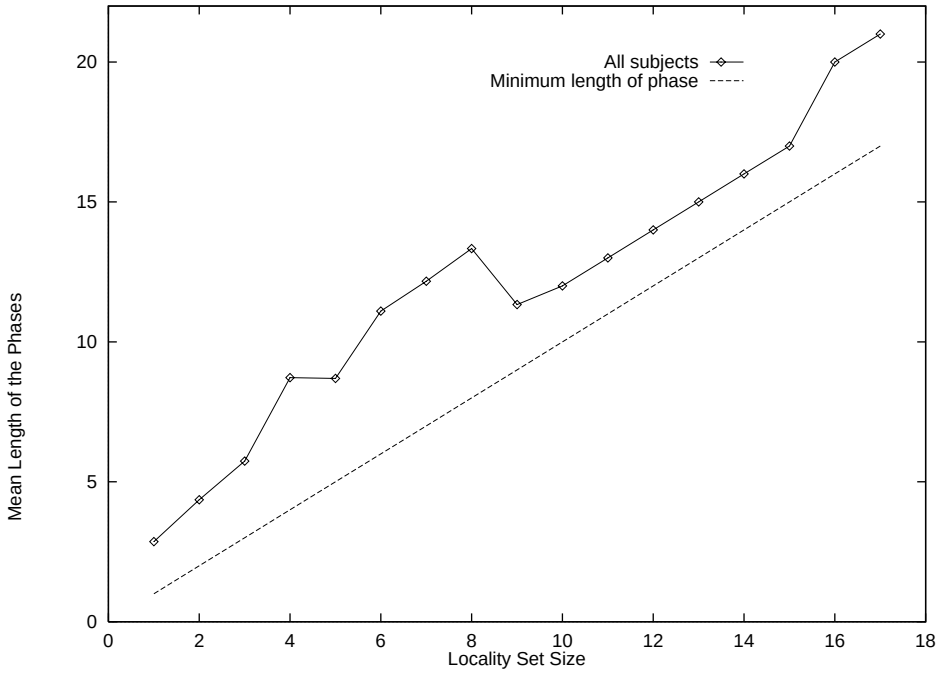

Figure 5.13. Weighted mean length of locality set phases

axis plots the total number and unique number of sets at that size for all 23 subjects. The graph shows that all sets of size 5 and greater were never repeated by the user.

Table 5.4 also reports the average duration or phase length (D) for each locality set size. One would expect to see an increasing phase length with increasing set sizes; also, phase length must be at least the same as the set size, according to the locality detection algorithm. The weighted mean (WMn) presents a more accurate picture of the length of a locality set (i.e. how long people stay in it) especially at larger set sizes where many subjects report zero occurrences. Thus, sets of size 1, 2, 3, and 4 exist for an average of 2.9, 4.4, 5.7, and 8.7 URLs respectively. This data is also presented in Figure 5.13 which plots locality set size versus the weighted mean phase length for all subjects. The data show that Web locality sets do not go much beyond this minimum phase length; sets exist for only a short number of URL visits.

The appearance of the curve in Figure 5.13 for larger set sizes is heavily dependent upon only a few subjects. For example, the regular, linear slope between set sizes 9 through 17 is due to one particular subject that performed several Open URL actions to

read all chapters of an online book, and then used the Back action to return to the first chapter. Before backtracking, the subject accessed the table of contents, and then the epilogue which caused a phase length of two plus the locality set size for all subsequent locality sets that were caused by each consecutive Back action.

The locality rate, i.e. the percentage of URLs occurring within a phase, was calculated for each subject. The mean locality rate, averaged over all 23 subjects, was 15% with a standard deviation of 9%. The lowest locality rate for a subject was 5% while the highest locality rate was 42%. Thus, over all subjects, only 15% of URLs visited were members of a locality set.

Because there was variation in browsing behaviour, locality was also examined at the individual level. Two types of graphs display locality results for each subject: phase durations and locality set size over the subject's browsing period, and repetitions of locality sets. Figures 5.14 and 5.15 plot a line for each locality set according to its set size (vertical axis) and when it occurred during the browsing period (horizontal axis). Figure 5.14 plots data for subject15 who generated the highest locality rate (42%) while Figure 5.15 plots data for subject24 who generated the lowest locality rate (5%).

Subject15 had below average browsing activity (701 versus 902 log statements) while subject24 had above average browsing activity (1257 log statements). Their recurrence rates also differed: subject15 reported 74% while subject24 reported 61%. Both of these rates are above average. However, subject24 generated considerably fewer locality sets than subject15.

Figures 5.16 and 5.17 show that few of the sets were repeated. The URLs visited appear on the horizontal axis while each unique locality set is assigned a number upon termination that represents the vertical axis. Thus, recurrences of the same set will appear to the right of the set's first occurrence. Besides showing the number of repetitions these graphs also shows recency effects since they display when the set was repeated.

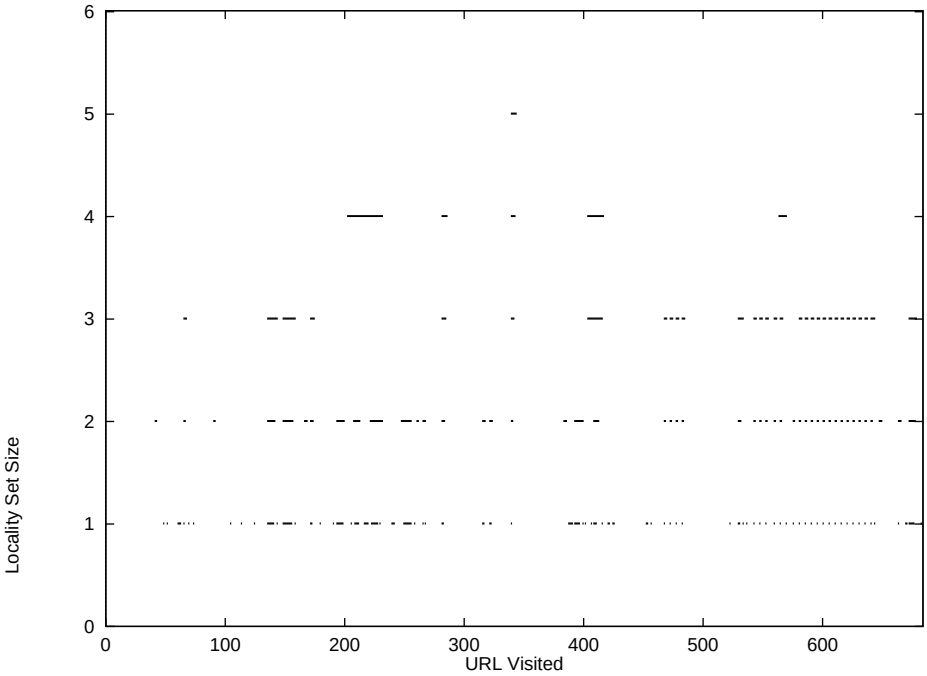

Figure 5.14. Phase Durations per Locality Set Size for subject15

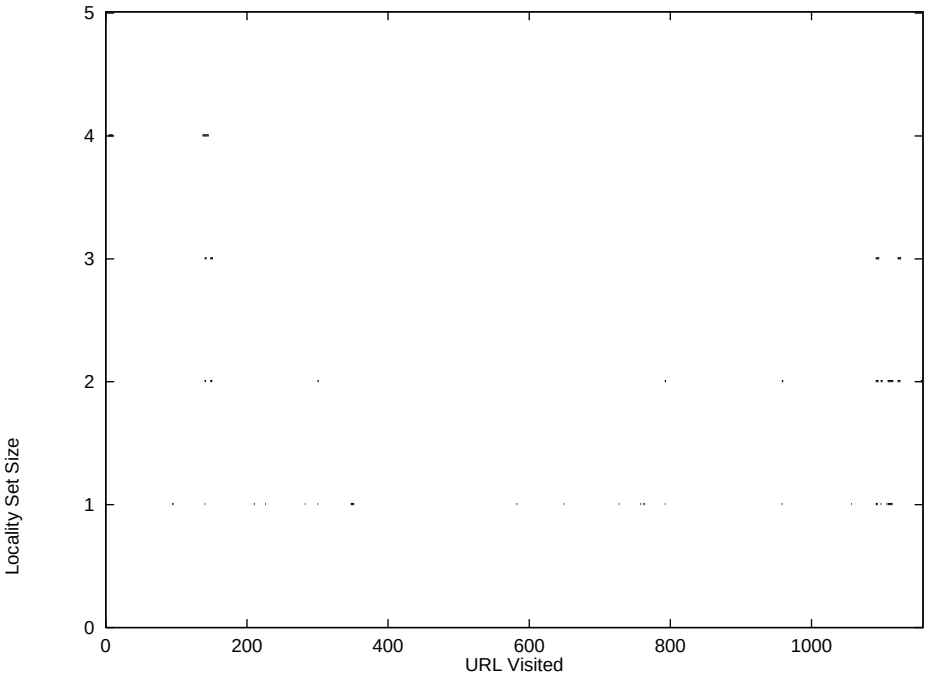

Figure 5.15. Phase Durations per Locality Set Size for subject24

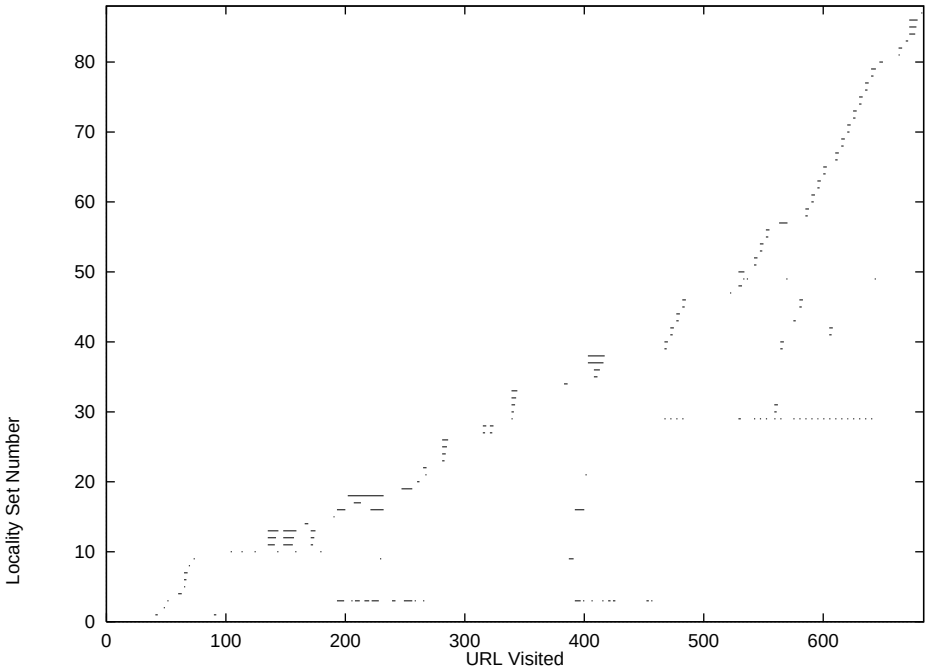

Figure 5.16. Locality Sets by Locality Set Number for subject15

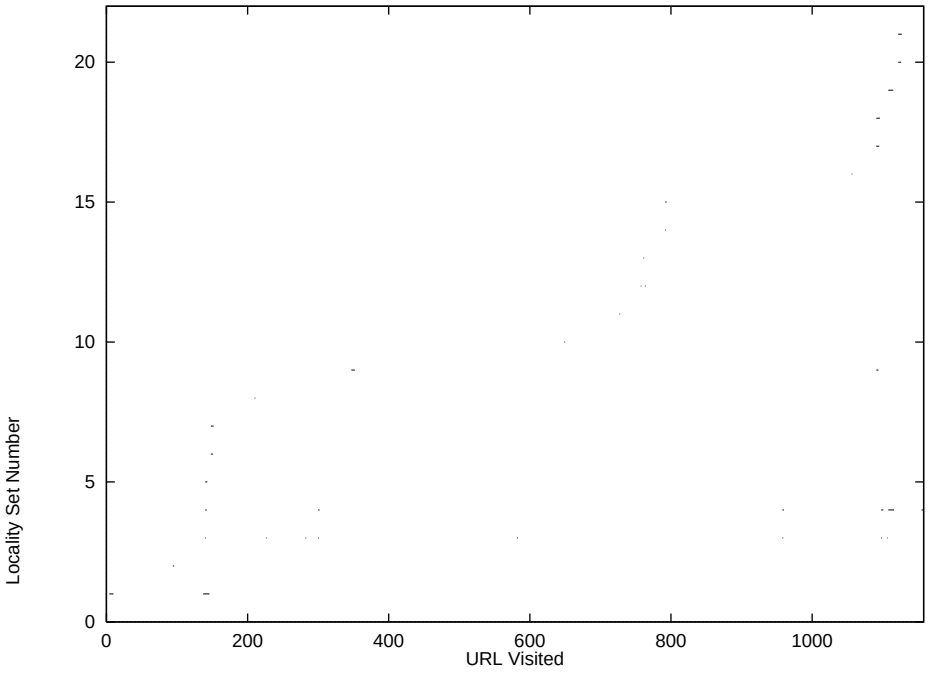

Figure 5.17. Locality Sets by Locality Set Number for subject24

Subject15 had 3 locality sets that were repeated 4 times or more while subject24 only had 2 such sets. The major differences that resulted in the locality rate spread are the greater number of locality sets generated by subject15 and their longer duration. From post-study interviews, it appears that subject15 was more directed in his use of Mosaic. He performed authoring, accessed course notes, and performed a course project using a Web-based knowledge elicitation tool. While subject24 also used the Web extensively for research during the study, his browsing was more exploratory—searching for conferences, papers, organizations, and researchers in his research area.

5.4.3 Discussion

The locality detection algorithm was applied to the Web browsing data to determine whether users browse within a cluster of related pages. While locality sets were found in the data, this pattern has minimal value for the design of history systems.

First, most locality sets were very small consisting of only one or two unique URLs. A locality set of size one is already captured by a simple history list as the last item entered. Sets of size two are of limited usefulness for it is highly likely that the other URL in the set is a hyperlink on the current page, and thus very easy to access in this manner. Lee (1992) found similar results with Unix command lines; most of the sets contained only one or two Unix commands. Larger sets are rare and probably not worth the effort in capturing.

Second, these sets lasted for only a short time (usually 2.5 to 4.5 URLs). Lee (1992) found similar results with Unix command lines.

Third, few locality sets were repeated as illustrated by Figures 5.12, 5.16, and 5.17. Some repetition of smaller set sizes occurred but as stated above, sets of size one or two would be of little use in browser history mechanisms. Lee (1992) also found few repetitions of locality sets for Unix command lines. For example, locality sets of size 2 showed a range of mean occurrences across four user groups of 1.28 to 2.94. Sets that were repeated tended to be of size one and two.

Fourth, the locality rate was very low; only 15% of URLs visited were part of a locality set. Lee's (1992) results were better for Unix command lines, but even then 31% is still poor considering that this method would ultimately be used to predict the user's

next command line submission, and that the recurrence rate is 75%. However, Lee still claims merit to locality, although her arguments are somewhat puzzling. Lee improved upon the 31% by basing locality rate upon commands lines found in phases only. The locality rate rose to 90% and the recurrence rate to 88%. It seems obvious that only considering commands lines within phases, or periods of high repetition, would increase these rates. Also, Lee concludes that locality rate is a better estimate of reuse than recurrence rate though there is only a 2% difference between the two values. Still, Lee does point out that a program's working set provides a good approximation of a program's locality set (Lee, 1992).

Several interesting patterns in the locality data are evident in the Web browsing domain.

1. Repeated use of *Back* generates a hierarchy of locality sets as evidenced by the "tornado"-like group of lines that appear on the subject plots (see Figure 5.16 for several good examples).
2. Use of *Reload* generates locality sets because the same URL is reaccessed. Thus, individuals performing authoring tasks showed higher than average locality rates.
3. Subjects who navigated through Web pages using internal links generated locality sets because the same URL was accessed repeatedly (the named anchor was removed).
4. Subjects using Web-based applications such as the knowledge elicitation tool generated more locality sets because their browsing was more confined to a set of URLs. Though Web-based applications were not used extensively by most subjects, a larger proportion of a user's Web browsing will likely occur in this mode in the future. Locality as a method of predicting future user interactions may then be more applicable within the Web domain.

In conclusion, though locality is an interesting concept, it does not appear to offer more than recurrence rate in terms of predicting the user's next activity within Web browsing. Locality set sizes tend to be very small, and even the small sets occur infrequently. Finally, it is unclear how locality could be practically applied within a working history system. In general, locality is too constrainted 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.

5.5 Paths / Longest Repeated Subsequences

The concept of *paths*, an ordered traversal of hypertext links, has been associated with hypertext ever since Vannevar Bush envisioned hypertext in the 1940's. If paths do exist, it may be useful to record them to make them explicit for the user and shareable with others. Also, paths may be followed because the user knows that the route will take them to a destination page, and perhaps shortcuts through the path would allow a user to go directly to it.

Consequently, we wanted to assess the extent to which World Wide Web users repeatedly make the same sequences of document accesses when browsing. We applied the Pattern Detection Module (PDM) algorithm (Crow and Smith, 1992) to the data to identify longest repeated subsequences (LRSs) of page visitations. A modified version of the PDM algorithm was used by Catledge and Pitkow (1995) to perform a site analysis on their data. They developed a characterization of sites based on frequency and path length relations. For example, a long path length of low frequency would indicate a one-shot resource or a directed search whereas a short path length of low frequency would indicate a dead-end or un-useful page (Catledge and Pitkow, 1995). In their study, the average frequency per path length was plotted and a negative linear slope of -0.24 was calculated. This indicates that frequency declines as path length increases, though this relationship is a weak one. Hence, we expect paths to exist in our data and anticipate that the relationship between path length and frequency will tend towards an inverse association.

Our research also examines the relationship between paths and LRSs. That is, are LRSs a reasonable measure of a path through hypertext? We question the usefulness of the PDM algorithm in our domain for two reasons:

- hypertext encourages the user to explore so they may deviate slightly from their normal route on occasion;

URL No.	URL	Navigation Action
1	www.cpsc.ucalgary.ca/projects/grouplab/wwwresume/teach.html	Open URL
2	www.cpsc.ucalgary.ca/projects/grouplab/wwwresume/talks.html	Open URL
3	www.cpsc.ucalgary.ca/projects/grouplab/wwwresume/grants.html	Open URL
4	www.cpsc.ucalgary.ca/projects/grouplab/wwwresume/academ.html	Open URL
5	www.cpsc.ucalgary.ca/projects/grouplab/wwwresume/uniserv.html	Open URL
6	www.cpsc.ucalgary.ca/projects/grouplab/wwwresume/pubs.html	Open URL

Table 5.5. LRS with length of 6 and frequency of 2 for subject23

- the PDM algorithm, like the locality detection algorithm, is quite restrictive in that the *exact* same sequence of URLs must exist for it to identify the LRS.

5.5.1 Analysis method

The Pattern Detection Module (PDM) algorithm, described by Crow and Smith (1992), finds the longest repeated subsequences (LRSs) in a logfile, where:

- *repeated* is defined as a subsequence occurring at least twice; thus, the minimum frequency for a LRS will be two;
- *subsequence* is a set of consecutive symbols;
- *longest* means that, although a subsequence may be part of another repeated subsequence found by the algorithm, there is at least one occurrence of this subsequence in the logfile 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".

In the domain of Web browsing, the LRS members are URLs that the user has visited. The sequential list of URLs visited by the user over time, ignoring boundaries between login sessions, is analyzed for sequences of URLs that constitute a LRS. The LRSs found may sometimes overlap. For example, 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 and Smith, 1992).

Three methods were used to analyze the nature and extent of LRSs within Web browsing.

	2#	2AF	3#	3AF	4#	4AF	5#	5AF	6#	6AF	7+#	7+AF
Mn	42.22	2.02	21.26	1.89	8.57	1.76	4.35	1.57	2.13	0.98	2.91	1.54
SD	39.68	0.20	23.27	0.13	9.68	0.57	3.89	0.67	2.97	0.97	3.67	0.84
Mdn	32	2.04	14	1.89	6	2	3	1.86	1	1.5	2	2
Min	11	1.66	2	1.59	0	0	0	0	0	0	0	0
Max	187	2.55	92	2.14	36	2.25	14	2.14	11	2	16	2.14
WMn		2		1.88		1.91		1.81		1.86		1.97
WSD		0.20		0.13		0.15		0.28		0.16		0.13

Table 5.6 LRS occurrences (#) and avg. frequencies (AF) for lengths 2-7+

5.5.1.1 LRS frequency and length

LRS frequency and length were analyzed at the individual and aggregate levels. For each subject, the mean number of LRSs and the average frequency of LRSs of a particular length were calculated. Aggregate results include the overall mean and frequency for each LRS length, and a summary graph that displays the total number of occurrences for LRSs of a particular length. This plot shows how the path length relates to the overall number of LRSs found.

5.5.1.2 LRS recurrences

LRS recurrences were also reported at the subject and aggregate levels. For each subject, recurrences were visually represented by plotting the LRSs that occurred during the subject's browsing timeline. This graph clearly identifies which LRSs were repeated, and where in the user's browsing timeline these recurrences occurred. A summary graph shows how frequently LRSs are repeated for a particular path length.

5.5.1.3 The activities that comprise LRSs

Reports for each subject were produced that listed each unique LRS found, where it occurred, and the URLs comprising the set. LRSs were printed in descending order of length, and shorter LRSs that occurred within longer ones were printed together. Larger and more frequently repeated LRSs were discussed with each subject during the post-study interview. The objective was to identify the types of browsing behaviours that may give rise to significant LRSs. For example, subject23 generated a LRS of length 6 and a frequency of 2 during the browsing sequence shown in Table 5.5. In this case, the

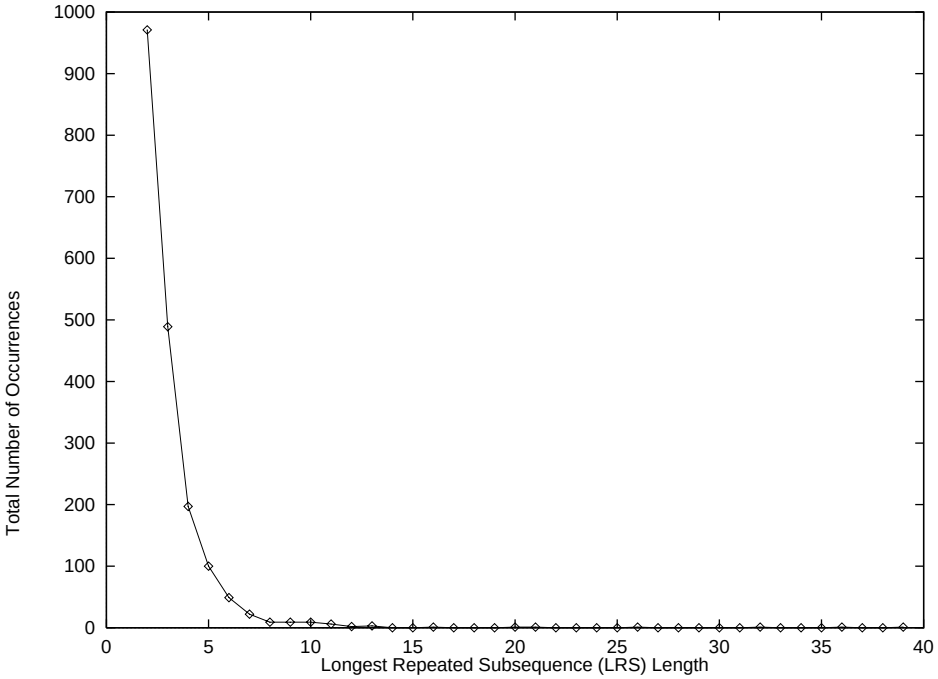

Figure 5.18. Total number of occurrences for LRSs of a given length

navigation actions were the same for both occurrences of the LRS. The subject was reviewing a sequence of authored pages using their *Next* and *Previous* hyperlinks.

5.5.2 Results

Table 5.6 summarizes the mean, standard deviation, median, minimum and maximum LRS occurrences and their average frequency for the 23 subjects. LRS lengths of 2 through 6 are reported in the first 10 columns; columns 11 through 12 contain the data for sequences of length 7 and larger. Two values are reported for each sequence length: the number of LRSs(#), and the average frequency (AF). The weighted mean (WMn) and weighted standard deviation (WSD) are also presented for the average frequencies; this is important at larger lengths when subjects are more likely to report zero occurrences.

Table 5.6 shows that LRSs of various lengths did present themselves in the WWW study data. All subjects had LRSs of two URLs with a mean of 42 occurrences per subject. All subjects also had sequences of three URLs with a mean of 21 occurrences. 21 of 23 subjects had sequences of four URLs with a mean of 9 occurrences. Note that as sequence lengths increase, the number of occurrences drops dramatically. Also, there is a

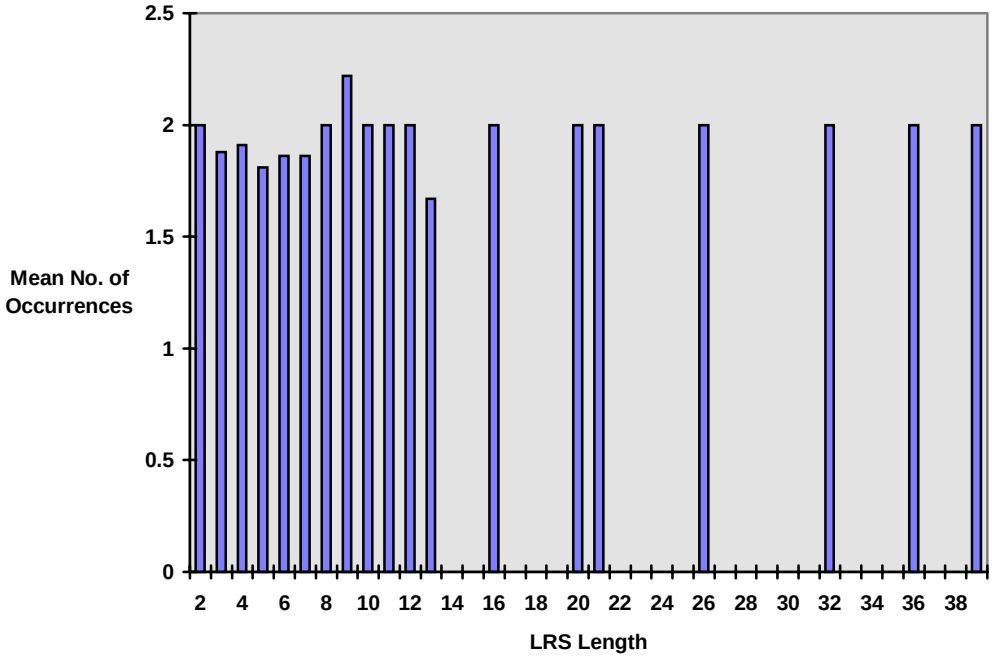

Figure 5.19. Weighted avg. of occurrences for LRSs of a given length

great deal of variability in the number of LRSs identified, as indicated by the standard deviations for lengths 2, 3, and 4.

Figure 5.18 also shows the dramatic drop in LRS occurrences as the sequence length increases. The horizontal axis represents the LRS length while the vertical axis displays the total occurrences of LRSs for all subjects. While 971 LRSs were found of length 2, only 67 LRS of length 7 or greater exist.

It is also important to examine how frequently these LRSs were repeated. Figure 5.19 plots the weighted average of occurrences for LRSs of a given length for all 23 subjects. The horizontal axis shows the sequence length while the vertical axis displays the weighted average number of occurrences. The frequency for all sequence lengths that do exist hovers close to two. This is the minimum as a sequence must have a frequency of at least two for it to be considered *repeated*. However, some average frequencies lie below two because a particular sequence may contain a smaller subsequence that appears at least once in the logfile.

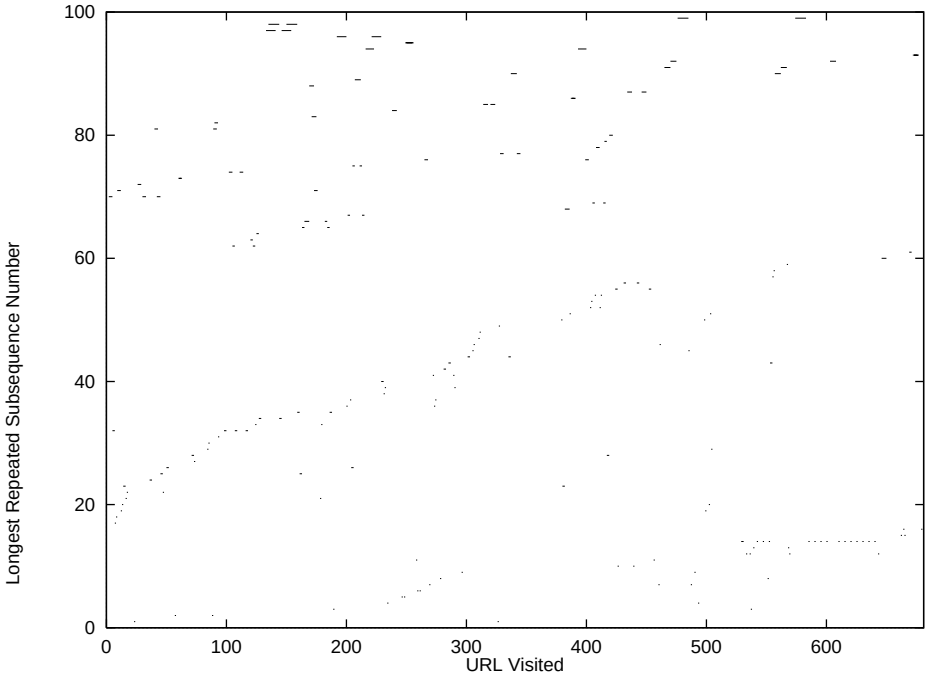

Figure 5.20. LRSs for subject15

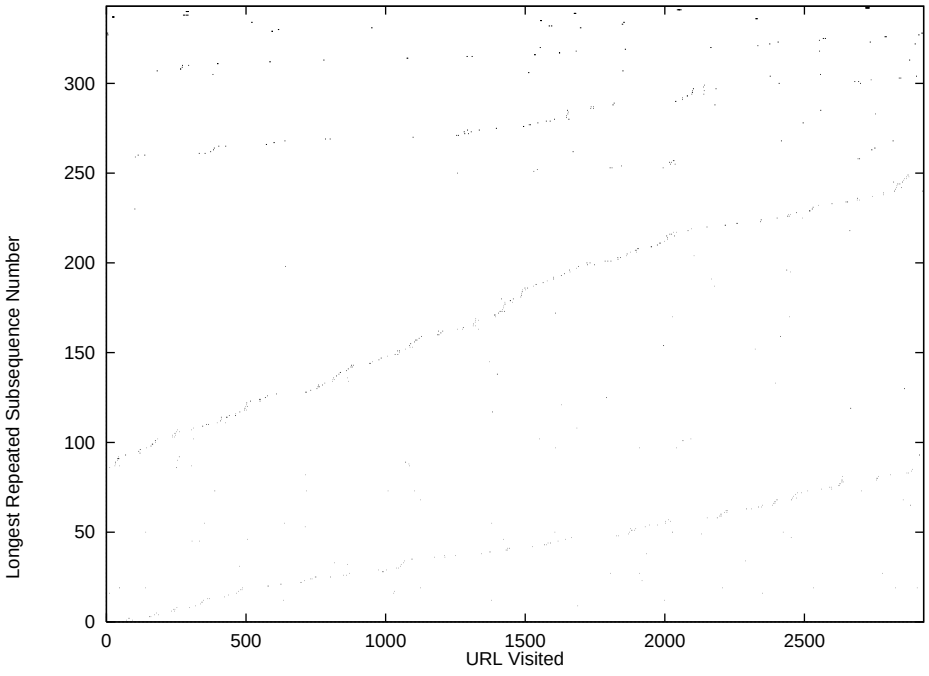

Figure 5.21. LRSs for subject26

This situation occurs frequently with the WWW data, and in many cases only one occurrence does exist. This factor causes the weighted average to lie below the minimally expected frequency of two for some sequence lengths, and it leads to a frequency of about 2 for sequence lengths overall.

Because there was variation in browsing behaviour, LRSs were also examined at the individual level. LRSs were visually represented with a graph that plots the browsing timeline versus each unique LRS found. This graph thus shows the total number of unique LRSs for the subject (each unique LRS was assigned a number that is represented on the vertical axis), and when they occurred (horizontal axis). For example, in Figure 5.20, LRS no. 14 was repeated 16 times, with the repetitions occurring between URL no. 529 and URL no. 641. LRS no. 14 only consists of two URLs; hence, it appears as a very short bar on the plot.

Two individual plots were chosen to show some of the variation that exists. Figure 5.20 plots LRSs for subject15 while Figure 5.21 plots LRSs for subject26. These subjects' results differ in several ways.

Subject26 generated many more unique LRSs than subject15 (342 versus 99). This may be partially explained by differences in browsing activity—subject15 had below average browsing activity (701 versus 902 log statements) while Subject26 had the highest browsing activity (3299 log statements).

Recency effects are more evident in subject26's data as compared to subject15. That is, LRSs for subject26 appear to be an artifact of how that subject browses; he often navigates through a sequence of URLs, and then returns *Back* through the same set of URLs. During the post-study interview, subject26 stated that he consciously tries to keep his history list "clean" so he frequently uses *Back* to eliminate unwanted URLs from the list.

Subject15 shows more repetitions throughout the study period than subject26. Subject15 performed more Web activities that involved reaccessing the same pages (he only visited 30 different domains) while subject26 visited many more different sites (195 domains). This is supported by previously discussed analyses. Subject15 had the highest

URL No.	URL	Navigation Action
1	www.eccc.uni-trier.de/eccc	Back
2	www.eccc.uni-trier.de/eccc/info/surveys.html	Open URL
3	www.eccc.uni-trier.de/eccc	Back

Table 5.7. LRS with length of 3 and frequency of 2 for subject24

locality rate of all subjects (42%) and the highest recurrence rate (74%), whereas subject26 had a locality rate of 25% and a recurrence rate of 63% (both are above average).

One could question whether the LRS occurrence and frequency statistics give a true picture of the repetition of LRSs. We feel that they present a pessimistic view of the extent of repetition in some situations. We also feel that the algorithm sometimes identifies repeated sequences that are not practically so. One of our objectives in examining the actual LRSs during post-study interviews was to confirm or refute these hunches. Our second objective was to identify patterns in the LRSs that lead to either long sequences or frequent repeats. These are our qualitative findings:

- Many of the longer LRSs arose due to the following actions taken upon the same page: reloading a page several times during an authoring session, navigating within a page using internal anchors, and submitting a form repeatedly (e.g. during a Web search). For example, during a single browsing episode, subject08 made 9 consecutive visits to the URL *www.radio.cbc.ca/radio/bcentre/media-book/sweekday.html* using a combination of *Open URL* and *Back* actions. This URL contained 4 internal anchors (e.g. *www.radio.cbc.ca/radio/bcentre/media-book/sweekday.html#WORLD REPORT*). The navigation sequence was thus partitioned into two paths by the PDM algorithm, one path from URL nos. 1 through 8, and a second path from URL nos. 2 through 9. While technically this navigation sequence can be considered 2 separate sequences, this appears to have little utility in the Web browsing domain.
- In several other cases, only two different pages appeared in the LRS due to navigating between the same two pages using either the *Open URL* and *Back* actions or the *Submit Form* and *Back* actions (search form and results page). Table 5.7 shows an

URL No.	LRS No.	URL	Navigation Action
1	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVX33.html	Back
2	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVVert.html	Open URL
3	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVX33.html	Back
4	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVWing.html	Open URL
5	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVX33.html	Back
6	—	rlv.msfc.nasa.gov/RLV_HTMLs/RLVLift.html	Open URL
7	—	rlv.msfc.nasa.gov/RLV_HTMLs/Gifs/LMSidebySide.gif	Ext. Viewer
8	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVX33.html	Back
9	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVVert.html	Open URL
10	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVX33.html	Back
11	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVWing.html	Open URL
12	1	rlv.msfc.nasa.gov/RLV_HTMLs/RLVX33.html	Back

Table 5.8. Navigation sequence showing LRSs for subject26

example of this for subject24 who navigated between a home page, an information page located at that site, and back to the home page. This sequence was repeated twice during two separate browsing sessions. However, the home page was actually accessed 40 times by the subject, and 26 different LRSs include the URL *www.eccc.uni-trier.de/eccc.*

- Many paths show a recency effect, that is the next LRS follows the previous very closely. This indicates 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. Or, the user may have used *Back* and/or *Forward* to return to a page through several pages. Table 5.8 provides an example for subject26 who generated an LRS of length 5 and frequency 2. The revisit to LRS no. 1 occurred shortly after the first visit, and the subject used *Back* and *Open URL* actions to move through a set of pages in a sequence.
- Some LRSs were generated by following *Next* and *Previous* hyperlinks that were embedded into the Web pages (see Table 5.5). These sequences show more unique URLs as compared to many other types of sequences that involve revisiting some URLs.

- Longer LRSs often contained shorter LRSs. The main problem with this is that it is difficult to assess the extent to which repetition occurs because the PDM algorithm identifies *longest* repeated subsequences. To illustrate this, a LRS of length 6 and frequency 2 (LRS no. 1) for subject22 appears in Table 5.9. The URL numbers from the log file have been preserved, as only the URLs pertinent to these LRSs are shown. LRS no. 1 contains 4 shorter LRSs. LRS no. 1a occurs twice elsewhere and is 5 URLs long; LRS no. 1b occurs 3 times elsewhere and is 4 URLs long; LRS no. 1c occurs once elsewhere and is 2 URLs long; LRS no. 1d occurs once elsewhere and is 4 URLs long. In addition, the URL *www.micro.ucalgary.ca/micro/appleindex.html* was visited 9 times by the subject while it appears 8 times in Table 5.9. This is because at URL nos. 3-34, subject22 visits *www.apple.ca* instead of *www.apple.com*; though the previous four URLs were the same, the algorithm identified this as a distinct LRS. The important point is that though the 5 LRSs in Table 5.9 occurred with a frequency of 1, 2, or 3, they actually involve navigating a similar set of pages.

URL No.	LRS No.	URL	Navigation Action
47	1a	www.cpsc.ucalgary.ca/	StartUp Doc
48	1a	www.micro.ucalgary.ca/	Open URL
49	1a	www.micro.ucalgary.ca/micro/price.html	Open URL
50	1a	www.micro.ucalgary.ca/micro/appleindex.html	Open URL
51	1a	www.micro.ucalgary.ca/micro/price/mac4.pdf	Binary Tsf
80	1	www.cpsc.ucalgary.ca/	StartUp Doc
81	1	www.micro.ucalgary.ca/	Open URL
82	1	www.micro.ucalgary.ca/micro/price.html	Open URL
83	1	www.micro.ucalgary.ca/micro/appleindex.html	Open URL
84	1	www.micro.ucalgary.ca/micro/price/mac4.pdf	Binary Tsf
85	1	www.apple.com/	Open URL
120	1b	www.micro.ucalgary.ca/	Open URL
121	1b	www.micro.ucalgary.ca/micro/price.html	Open URL
122	1b	www.micro.ucalgary.ca/micro/appleindex.html	Open URL
123	1b	www.micro.ucalgary.ca/micro/price/mac4.pdf	Binary Tsf
161	1c	www.cpsc.ucalgary.ca/	StartUp Doc
162	1c	www.micro.ucalgary.ca/	Open URL
164	1b	www.micro.ucalgary.ca/	Back
165	1b	www.micro.ucalgary.ca/micro/price.html	Open URL
166	1b	www.micro.ucalgary.ca/micro/appleindex.html	Open URL
167	1b	www.micro.ucalgary.ca/micro/price/mac4.pdf	Binary Tsf
295	1a	www.cpsc.ucalgary.ca/	StartUp Doc
296	1a	www.micro.ucalgary.ca/	Open URL
297	1a	www.micro.ucalgary.ca/micro/price.html	Open URL
298	1a	www.micro.ucalgary.ca/micro/appleindex.html	Open URL
299	1a	www.micro.ucalgary.ca/micro/price/mac4.pdf	Binary Tsf
443	1b	www.micro.ucalgary.ca/	Open URL
444	1b	www.micro.ucalgary.ca/micro/price.html	Open URL
445	1b	www.micro.ucalgary.ca/micro/appleindex.html	Open URL
446	1b	www.micro.ucalgary.ca/micro/price/mac4.pdf	Binary Tsf
456	1	www.cpsc.ucalgary.ca/	StartUp Doc
457	1	www.micro.ucalgary.ca/	Open URL
458	1	www.micro.ucalgary.ca/micro/price.html	Open URL
459	1	www.micro.ucalgary.ca/micro/appleindex.html	Open URL
460	1	www.micro.ucalgary.ca/micro/price/mac4.pdf	Binary Tsf
461	1	www.apple.com/	Open URL
481	1d	www.cpsc.ucalgary.ca/	StartUp Doc
482	1d	www.micro.ucalgary.ca/	Open URL

483	1d	www.micro.ucalgary.ca/micro/price.html	Open URL
484	1d	www.micro.ucalgary.ca/micro/appleindex.html	Open URL

Table 5.9. Navigation sequence showing LRSs for subject22

5.5.3 Discussion

We now discuss the results of applying the PDM algorithm to Web browsing behaviour. First, we compare our results to previous findings. Then we discuss the following aspects of our longest repeated subsequences data: repetition, recency effects, and the impact of various browsing strategies.

How do these results compare with previous findings? As described in the introduction to Section 5.5, Catledge and Pitkow applied the PDM algorithm to their Web browsing data. They developed a characterization of sites based upon frequency and path length relations. We did not replicate their analysis closely, but we did find a similar negative relationship between path length and frequency. Over all 23 subjects, only 67 LRSs of lengths 7 and greater were found while almost 971 LRSs of length 2 exist (Figure 5.18). From reviewing the LRS reports for each subject, it appears that many long LRSs actually involve consecutive accesses to the same page. For example, the subject used *Reload* repeatedly during authoring, or navigated through a document using its internal anchors. Therefore, few long paths were found and we suspect that many of those that exist reference the same URL. Thus, a history mechanism modelled on LRSs is of little use if the paths just contain repeats of the same page or if the paths are very short.

How often do subjects repeat a particular navigation sequence? Figure 5.19 shows that the weighted average frequency for all path lengths found is about 2 or the minimum necessary to qualify as a LRS. However, assessing repetition is not straightforward with this algorithm. A slight deviation in one's browsing sequence will generate a distinct LRS, as was demonstrated in Table 5.9. The PDM algorithm was successful in identifying *longest* repeated subsequences, but there are many LRSs that share similar URLs that are considered distinct. Thus, it was difficult to assess how much path repetition actually occurred in visits to pages.

Given that path repetitions do occur, are the revisits close to one another or is there considerable spread? By examining graphs of LRSs for each subject, we get the impression that there is a considerable recency effect, though this does vary depending upon the subject's purpose for browsing and their browsing strategy. Figure 5.21 shows more recency effects than Figure 5.20 because the first subject engaged in more exploratory browsing and used *Back* to actively prune their history list. Thus, tracking sequences of URL visits will likely be of more benefit to more focused users. Also, given that many LRSs appear to be an artifact of navigating *Back* and *Forward* through pages, capturing these LRSs for future use is anticipated to be of little benefit.

What is the impact of various browsing strategies upon the identification of LRSs? Certain types of navigation sequences result in consecutive visits to the same page. If this *homogeneous* sequence does not recur, the PDM algorithm will still identify this as an LRS by partitioning the sequence into two. However, there is no benefit in predicting a revisit to the current page. Other long sequences contained only two unique URLs as when the user navigates back and forth between a query page and the results page. Some longer sequences with no URL repetitions did occur, often due to the user following embedded *Next* and *Previous* links. Capturing this sequence in a history mechanism is also of little use since the user probably wants to visit each page, and an easy method to do so already exists. Where sequences would be useful is in pruning out pages that just take the user to a destination page. However, it is difficult to develop a heuristic that can do this.

In spite of these problems with the PDM algorithm, we believe that its utility could be improved if it were modified to handle noise, and included domain knowledge. By *improve utility*, we mean that the addition of these techniques will allow the algorithm to find longer, more frequently repeated, and more meaningful paths.

Crow and Smith (1992) discuss the application of noise handling to their own system, DB_Habits. They surmise that their domain of study is noisy because commands may occur in different orders on different occasions, the task may be interrupted during execution causing interspersed commands, and commands may be mistyped (Crow and

Smith, 1992). As a result, the same task will often be represented by distinct sequences of commands.

Similar situations may occur within Web browsing. A user may use a slightly different method to access a Web page depending upon their current context. For example, if they are on a page that has a hyperlink to the desired page, they may use the hyperlink rather than select the URL from their hotlist. The user's current task may be interrupted as well. The user may wish to visit a certain page but along the way see another hyperlink that they want to briefly explore. Their route will be essentially the same except for one (or a few) visit to a new URL, and a *Back* action to return them to the regular path. Mis-typed URLs are not a problem because we have tried to identify and remove those log statements.

Adding domain knowledge to the PDM algorithm entails identifying some of the features of hypertext that may limit a pattern from being formed. An example of this was described above—the exploration of side trails. The algorithm could allow the occasional navigation to a side trail if the user then continued along a well-traveled path; hypertext, by nature, encourages such explorations but the current algorithm would consider these deviations as distinct. Domain knowledge about browsing strategies could be incorporated. For example, LRSs referencing the same URL, or navigation between a query and results page could be omitted.

Until these modifications are made to the PDM algorithm, its value as a predictive/analysis mechanism in the hypertext domain will be minor.

In summary, the Pattern Detection Module algorithm was applied to the WWW browsing data in an attempt to identify longest repeated subsequences, or paths. We found that though LRSs do exist, they tend to be short. The longer LRSs usually involve references to one or two pages. In terms of repetition, the average frequency for LRSs of all lengths hovered around two which is the minimum requirement to be considered a LRS. Also, there is a strong recency effect in that repeats of LRSs occur within a short distance of each other. These results indicate that history mechanisms which present LRSs are of limited use. However, several changes to the PDM algorithm are proposed which may improve its utility in the hypertext domain.

5.6 Concluding Remarks

This chapter presented five analyses of the WWW usage study data that assessed navigation patterns that evolve over time: growth of URL vocabulary, recurrence rate as a function of distance, frequency, locality, and longest repeated subsequences.

1. We found that across all subjects new pages are incorporated into the user's repertoire of visited pages at a regular rate. However, there were also local variations in the vocabulary growth rate and use of navigation activities across subjects, and across their browsing timeline. These variations indicate the presence of different browsing activities.
2. The analysis of URL recurrence rate as a function of distance showed that there is a very strong recency effect in that most recurrences occur within a short distance of each other. There were two variations in this pattern—some browsing activities cause URLs at a distance of 1 to show a higher recurrence rate though the norm tends to be that a distance of 2 presents the highest rate, largely due to extensive use of the *Back* action.
3. Frequency is a popular pattern that is often applied to identify items of interest. We found that a few pages were heavily accessed while many pages were only referenced once (60%) or twice (19%) by subjects during the study period. Frequently accessed pages tend to fall into certain categories which explains their popularity: personal page, StartUp document, index page, search engine, organization/individual home page, etc.
4. The concept of locality was applied to our data; this analysis replicates previous research involving Unix command lines. All subjects did generate locality sets though they tended to be small, of very short duration, and most were never repeated. Also, a very small number (15%) of URLs were members of locality sets. Analysis at the individual level showed interesting patterns that arise due to navigation actions, particularly *Back* and *Reload*. The locality algorithm was applied as a strict definition from its origin in computer memory research. Relaxing the constraints of the algorithm

to consider noise and domain knowledge may increase the utility of this method for predicting future URL recurrences.

5. The final section addressed longest repeated subsequences, or paths of URL visits; our analysis extends work done by C & P (1995). We found that LRSs exist in the browsing data, but they tend to be short and were only repeated twice on average. Also, repetitions tended to occur within a short distance of each other. However, it was difficult to assess the actual existence and extent of paths or sequences of URL revisits via the PDM algorithm. Like the locality algorithm, the criteria for LRSs were applied according to their strict definition. Simple techniques to deal with noise and to incorporate knowledge of the hypertext domain could improve the results of this method.

6. History List Conditioning Methods

An important consideration in the design of history mechanisms is the effort involved in using history tools. The amount of work to select and submit an activity for reuse must be less than specifying it from scratch. In general, the probability of a recurrence varies depending on the distance, or the position of the recurring item in the history list. This is because shorter lists are quicker to search than longer ones.

This chapter examines methods of conditioning a history list of URLs visited by the user. The purpose of conditioning a history list is to increase the probability that the next URL a person wishes to visit is available in a small set of predictions offered to them for review and reuse. The first section of this chapter discusses the conditioning methods that were applied to the browsing data collected during the WWW study. The second section presents the results of the evaluation. The final section discusses the implications of our findings.

6.1 Conditioning Methods

This section presents background information about applying conditioning methods to the browsing data, and then describes the eight methods that were used.

6.1.1 Conditioning the distribution

Analyses of the WWW navigation study data in Chapters 4 and 5 led us to conclude that there is benefit in capturing a user's navigation history and presenting it to them for reuse. People do revisit Web pages, and Web browsing qualifies as a recurrent system. One of the most striking features of our findings is the strong recency effect in page visits—the last few pages visited are the likeliest to be repeated, and offering these pages to the user in the form of a history list would clearly be beneficial.

However, there are still a considerable proportion of pages that are not recent revisits. Specifically, the recurrence distributions of Section 5.2.2 were derived by considering all page visits for a user as one long input stream with no barriers placed between sessions. We have seen that although a 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 Figure 5.8, there is a C = 42% chance that the next URL has not appeared before, an R_{D10} = 43% chance that it has occurred within the set and a 15% chance that it last appeared further back. This chapter explores the possibility that the distribution can be conditioned, first to increase the recurrence probabilities over a set of a given size, and second to evaluate methods that are currently in use.

Eight conditioning methods are explained in the remainder of this section. Each method has a different way of arranging a user's page visit history that will condition the probability distribution of the next URL given a sequential history list of previous URLs. These include both methods that we expect to perform well, and perhaps more dubious methods that have been implemented in existing Web history facilities. For each method we indicate how the study data will be analyzed to assess its effectiveness. The detailed algorithms used to find $R_{s,d}$ will not be discussed as they are minor variations of the one summarized in Section 5.2.1.

6.1.2 Sequential ordering by recency

This conditioning method was described in Section 5.2, and is simply a time-ordered list of all URLs visited by the user. Table 6.1a illustrates a sequentially ordered history list containing the last 16 submissions, and numbered by order of visit. The most recent visit appears at the top, as the history list is intended to be viewed top down.

According to Greenberg (1993a), there are two benefits of recency. 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 startup instability that other methods do when there are only a few URLs available to present to the user.

6.1.3 Pruning duplicates from a recency list

The sequential ordering by recency method described in Section 6.1.2 contains every URL the user has visited, including revisits to the same URL. These duplicates occupy valuable space on a history list of a limited length. Greenberg (1993a) applied two

URL Visit No.	URL	Navigation Action
a) Sequential ordering by recency		
16	acsl.cs.uicu.edu/kaplan/applets.html	Open URL
15	acsl.cs.uicu.edu/kaplan/worlds-environment.html	Open URL
14	acsl.cs.uicu.edu/kaplan/worlds.html	Open Hotlist
13	www.acm.org/sigchi/chi96/forms/ProcFormat.html	Open URL
12	www.acm.org/sigchi/chi96/call/index.html	Open URL
11	www.acm.org/sigchi/chi96/	Open URL
10	www.acm.org/sigchi/homepage.html	StartUp Document
9	info.sigchi.acm.org/sigchi/homepage.html	Back
8	www.acm.org/sigchi/cscw96/	Back
7	www.acm.org/sigchi/cscw96/dates.html	Open URL
6	www.acm.org/sigchi/cscw96/	Open URL
5	info.sigchi.acm.org/sigchi/homepage.html	Back
4	www.acm.org/sigchi/chi96/	Back
3	www.acm.org/sigchi/chi96/Deadlines.html	Open URL
2	www.acm.org/sigchi/chi96/	Open URL
1	info.sigchi.acm.org/sigchi/homepage.html	StartUp Document

Table 6.1.a) Example of a sequential ordering by recency history list

URL Visit Nos. (top of stack .. bottom of stack)										
b) Recency, duplicates saved in latest position										
16	15	14	13	12	11	10	9	8	7	3
c) Recency, duplicates saved in original position										
16	15	14	13	12	10	7	6	3	2	1
d) Frequency, second key recency										
11	9	8	16	15	14	13	12	10	7	3
e) Stack										
16	15	14	13	12	11	10				
f) Stack, persistent										
16	15	14	13	12	11	10	1			
g) Context-sensitive Web subspace										
14										
	16	15								
10										
	13	12	11							
1										
	8	7	5	4	3					

Table 6.1.b)-g) Examples of history lists conditioned by different methods

strategies for pruning redundant Unix command lines: saving the command line in its original position on the history list, and saving the command line in its latest position.

The latter approach is expected to perform better because local context is maintained, and less visited URLs will migrate to the bottom of the list (Greenberg, 1993a). Table 6.1b and Table 6.1c provide an example of both approaches to pruning duplicates. Note that there are fewer URLs on these lists as compared to the sequential version which retains all URLs. Also, note that the user's StartUp document (a heavily accessed page) occupies the last position on the list when URLs are saved in their original position. Both strategies of pruning duplicates will be applied to the WWW navigation study data.

6.1.4 Frequency ordering

Frequency ordering, where the most revisited page appears at the top of the list and the least visited page appears at the bottom, is perhaps the most obvious way of ranking URLs. The problem with this approach is that user's needs and interests change, and frequency ordering is not very responsive to this. Newer URLs need to be revisited frequently before they can migrate to the top of the list while older and more frequently used items that are no longer of interest remain near the top. Still, Section 5.3 showed that there are certain types of pages that users tend to frequent. We thus expect that the predictiveness of frequency ordering will attain a reasonable level after the distribution has had an opportunity to stabilize (via extended browsing).

An issue associated with frequency ordering is how to break ties, that is, how to order URLs that have the same frequency. Greenberg (1993a) 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 6.1d shows the effect of frequency ordering with secondary sorting by recency upon the example navigation session.

6.1.5 Stack ordering

Current Web browsers maintain a history list that operates as a stack to present the linear path from the first URL visited to the current URL. In many browsers, the most recent page appears at the top of the list while the least recently accessed page appears at

the bottom. This is a simplistic and somewhat inaccurate description for there are some nuances in how the current history mechanism operates. Understanding the subtleties of browser history lists requires differentiating the three classes of user technique for page display as described by Cockburn and Jones (1996): *loading*, *recalling*, and *revisiting*.

- Loading a page occurs when the page is *directly accessed* by selecting a hyperlink, typing the URL, choosing an item from the Hotlist, etc. The page may or may not have been visited prior in the session. Loading a page causes it to be added to the top of the history list. However, loading a page while at some point other than the top of the stack causes all pages above the current position in the stack to be lost (Cockburn and Jones, 1996).
- Recalling a page occurs when the page is revisited through a *Back* action, *Forward* action, or selecting an item from the history list. These actions change the *pointer* to the currently displayed page in the history list effectively allowing the user to move up or down the stack; they do not result in the addition or removal of items from the list.
- Revisiting a page occurs when the user explicitly reloads the current page. There is no effect upon the history list; the current page remains in the same location in the list.

The above behaviours were reapplied to our study data to show how well this method performs. The stack condition method differs from the others previously discussed in the following ways.

1. A new stack is generated at the start of each session, and for each new browser window. URLs visited in a previous session or window are not carried forward.
2. Duplicates may exist on the stack if, for example, the user selected a hyperlink to their *StartUp Document*, or clicked the *Home* button rather than choosing this page from the history list or via the *Back* action.
3. Certain actions do not affect the stack even though they are considered navigation actions (*External Viewer*, *Binary Transfer*, and *Telnet Window*). These were included in the history lists for the other non-stack based methods.

We expect that the stack method will perform reasonably well for very short recurrence distances. However, it will do poorly for long distances because URLs are not

retained on the history list between sessions. This method will be mediocre for intermediate distances because some recent URLs are removed when the user loads a page while at some point other than the top of the stack.

Table 6.1e shows the history list at the end of the example navigation session. The list is short because the sequence actually includes two separate sessions (note the two *StartUp Document* actions in the sequential list), and the example shows the contents of the history list at the end of the second session. In this example, no backtracking occurred so all URLs visited in session two are present in the stack-based history list.

6.1.6 Persistent stack ordering

The stack method described in Section 6.1.6 is sessional as the stack is empty at the start of each session. A persistent stack makes the stack for each browser window available for the next session. Rather than starting from scratch each time the browser or a new window is invoked, the state of the stack at the end of the previous browsing session for a particular window is saved and redisplayed. While browsers do not yet do this, it is a reasonable extension of the current implementation of stack-based history.

We expect that the persistent stack will perform the same as the sessional stack for very short distances. However, the persistent stack will do better for long distances because some URLs are retained between sessions. This method is not expected to do 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 6.1f shows the history list at the end of the example navigation session. The list is one item longer than the sessional stack list because 1 URL was present in the browser's history list at the end of first session, and it is located at the bottom of the list.

6.1.7 Recency ordered hyperlink sublists

This method is similar to the recency ordered history list in which duplicates are pruned and saved in their latest accessed position. The difference is that for each URL on the normal list, a secondary list of the hyperlinks visited from that URL can be raised. The user first scans down *i* entries in the normal list for an exact match that terminates the search, or for an entry that contains the desired hyperlink. In the latter case, the sublist 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*. Given the recency list with duplicates saved in their latest position (see Table 6.1c), for example, the hyperlink sublist on URL *www.acm.org/sigchi/chi96/* (Visit Nos. 11, 4, and 2) would be:

- *www.acm.org/sigchi/chi96/call/index.html* (visit no. 12), and
- *www.acm.org/sigchi/chi96/Deadlines.html* (visit no. 3).

If the URL is found in both the normal list and a sublist, the shortest recurrence distance is used. In the above example the *.../index.html* URL is at a distance of 5 on the main list but a distance of 7 on the sublist (since its parent URL, *.../chi96/*, is at a distance of 6). However, the *.../Deadlines.html* URL is at a distance of 11 on the main list but a distance of only 8 if accessed via its parent URL sublist (*.../chi96/*).

There are two benefits of this approach. First, more URLs are accessible, and if the user was visiting a page simply because it contained a hyperlink to the desired page, that page can now be selected directly from a sublist. Second, because sublists contain only URLs that the user has accessed from a particular page, the user may find it easier to use the sublist to select a URL especially when the page has many links and they must scroll extensively to locate the URL.

Two considerations must be made when evaluating this method. There is additional cognitive and physical effort involved in accessing and selecting an item from a sublist versus a sequential list. The user must scan the sequential list and decide which sublist to display, select that sublist, and scan it for the desired URL. Intuitively, we estimate that the time and effort to do so will be less than accessing the actual Web page, and scrolling and/or scanning it for a hyperlink. Second, our recurrence distance calculation is somewhat biased because when a page is accessed it immediately appears at the top of the history list. Since users often navigate via selecting a hyperlink on a page, those hyperlinks are immediately available in the sublist at a shorter distance than if only the single list had existed. For this reason, as well as the strengths of this approach, we expect recency

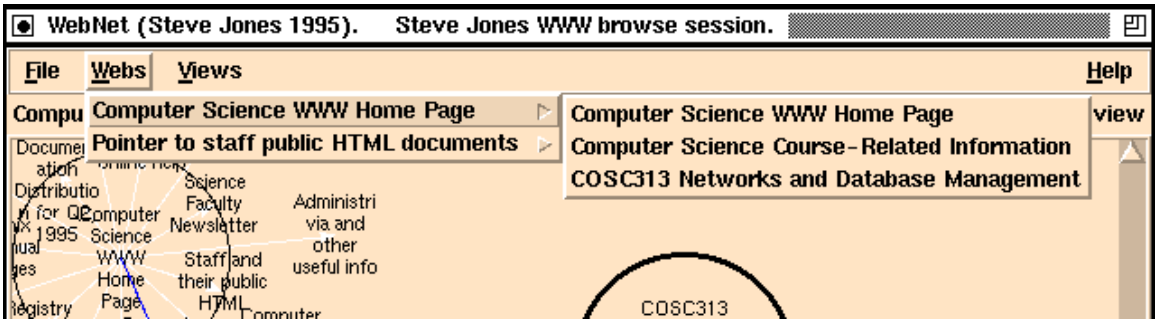

Figure 6.1. WebNet's dynamic page menu (Cockburn and Jones, 1996).

ordered sublists to perform better than recency with duplicates saved in their latest position.

6.1.8 Context-sensitive *Web subspace* history lists

Cockburn and Jones (1996) designed a graphical history display that includes a *Webs* menu to allow users to manage and access collections of distinct WWW subspaces (see Section 2.2). This partitioning of a user's browsing history was designed to deal with the problems of memory overload, and lack of context. Web subspaces also reduce information overload by decreasing the volume of pages and links present within the display. A new subspace is created each time the user *directly accesses* a page. This page is added to the *Webs* menu, and any pages accessed within this subspace are added to a cascading or secondary menu that is linked to the menu entry for the first page in the subspace (see Figure 6.1).

This approach can be modelled as a conditioning method. For our analysis, we considered the following actions as a *direct access* to a URL:

- typing a URL;
- selecting a Hotlist item;
- cloning or opening a new window (including Mosaic *Help*, and browser invocation);
- accessing a URL via client-dependent hard-wired buttons or menus (e.g. *Home*, *Open Local*).

URLs accessed from a direct access URL to the next direct access URL (e.g. selecting a hyperlink) are considered to belong to the same subspace and are therefore accessible in

the cascading menu. Within the main and secondary menu, we sort the URLs based on recency, and remove duplicates, saving according to the 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 subspaces if the user navigated to it in different ways. This is appropriate since subspaces 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. A Web subspace groups these pages together.

We expect that the performance of this method will be affected by a variety of factors.

1. It should do well because it seems to associate URLs that are context-sensitive. However, this association is only an estimate of the actual task context. A user, by simply following hyperlinks, may change their context entirely though all pages accessed until the next direct access URL would appear in the same subspace.
2. For Web subspaces that are browsed extensively, the cascading menu will grow to be very large. Still, the recency ordering will propagate the more likely candidates to the top of the list. Once the direct access URL is revisited, it will move to the top of the main menu and remain there until another URL is directly accessed. In the purely recency method, any URL (direct or not) would be placed above the direct access URL. URLs accessed within the subspace will thus have a recurrence distance equal to their position in the cascading menu plus one to account for the direct access item on the main menu. For this reason, we believe that Web subspaces will show better performance than recency sublists.
3. It is unclear whether the user will grasp the conceptual model for Web subspaces, and the distinction between direct and non-direct URL accesses. Is this an intuitive way of presenting and partitioning one's browsing history?

Table 6.1g shows the history list at the end of the navigation session. There are 3 URLs in the main menu indicating that 3 direct accesses were made (2 *StartUp Documents*; 1 *Open Hotlist*). The last webspace the user browsed is located at the top of the list. The sublists show the contents of the web subspace sorted in recency order.

6.2 Results

This section presents the results of our application of conditioning methods to the browsing data for the eight methods that were evaluated.

6.2.1 Background / Introduction

Results for all conditions are summarized in two tables, each presenting various distributions over the last 50 items of the history list. Table 6.2 presents the percentage of the frequency of visits recurring at a particular distance (R_d), and Table 6.3 presents the same information as a running sum over distance (R_D). The latter includes the total recurrence rate over the complete history list, which differs under certain conditions. Figure 6.2 graphs the results of Table 6.3. As with Figure 5.7, the horizontal axis shows the position of the revisited URL on the history list relative to the current one, while the vertical axis represents R_D, the rate of accumulated URL recurrences, as a percentage. Figure 6.3 graphs only results over the first 10 items of the history list to make it easier to visualize the behaviour for this range.

6.2.2 Sequential ordering by recency

In Section 6.2, we reported a R_{D10} of 43% for our 23 subjects. This will be used as a benchmark for comparing other conditioning methods.

6.2.3 Pruning duplicates from a recency list

Although pruning duplicates does not change the recurrence rate, it does shorten the total distance covered by the distribution (i.e., the history list is smaller). However, the strategy used has a profound effect on whether the removal of duplicates increases or decreases the predictiveness compared to strict sequential ordering by recency. Consider a 10 item history list. Recurrences saved in their original position fair poorly with R_{D10} = 29% (cf. 43% for strict sequential). Recurrences saved in their latest position fair well with a R_{D10} = 47%, and compared to the sequential ordering that retains all items, this method of pruning duplicates increases the overall probability of a 10 item history list by 4%. Greenberg (1993a) reported similar results for Unix command lines. The remainder of this chapter assumes that history lists with duplicates pruned will use this method.

1	2	3	4	5	6	7	10	20	30	40	50
Recency											
9.97	18.81	2.23	4.93	1.07	2.47	0.88	0.91	0.39	0.17	0.12	0.07
Recency, duplicates saved in latest position											
9.97	20.05	6.06	3.66	2.05	1.43	1.33	0.77	0.24	0.20	0.05	0.09
Recency, duplicates saved in original position											
5.93	7.79	4.51	3.01	2.27	1.93	1.26	0.75	0.27	0.17	0.12	0.15
Frequency, second key recency											
7.60	4.31	3.18	2.77	2.15	1.76	1.64	1.16	0.64	0.38	0.33	0.19
Stack											
8.17	19.92	6.85	3.18	1.37	0.78	0.49	0.25	0.01	0.00	0.00	0.00
Stack, persistent											
8.61	20.50	7.53	3.99	1.96	1.46	1.03	0.53	0.09	0.08	0.03	0.04
Recency, hyperlink sublist											
10.08	23.36	7.14	3.93	2.10	1.35	1.01	0.73	0.14	0.06	0.06	0.01
Context-sensitive Web subspace											
11.35	9.53	16.50	6.28	3.35	1.92	1.32	0.73	0.17	0.06	0.02	0.02

Table 6.2. Probability of a recurrence over distance for various conditioning methods

1	2	3	4	5	6	7	10	20	30	40	50	***R***
Recency												
9.97	28.8	31.0	35.9	37.0	39.5	40.4	43.4	48.2	50.4	51.5	52.5	58.0
Recency, duplicates saved in latest position												
9.97	30.0	36.1	39.7	41.8	43.2	44.5	47.3	51.4	53.0	54.0	54.8	58.0
Recency, duplicates saved in original position												
5.93	13.7	18.2	21.2	23.5	25.4	26.7	29.3	34.0	35.9	37.4	39.0	58.0
Frequency, second key recency												
7.60	11.9	15.1	17.9	20.0	21.8	23.4	27.3	35.2	40.3	44.0	46.8	58.0
Stack												
8.17	28.1	34.9	38.1	39.5	40.3	40.8	41.5	42.3	42.4	42.5	42.5	42.5
Stack, persistent												
8.61	29.1	36.7	40.6	42.6	44.1	45.1	47.1	49.3	50.5	51.1	51.7	53.0
Recency, hyperlink sublist												
10.1	33.4	40.6	44.5	46.6	48.0	49.0	51.3	54.1	55.2	55.7	56.2	58.0
Context-sensitive Web subspace												
11.4	20.9	37.4	43.7	47.0	49.0	50.2	52.8	55.7	56.7	57.0	57.2	58.0

Table 6.3. Cumulative probabilities of a recurrence over distance for various methods

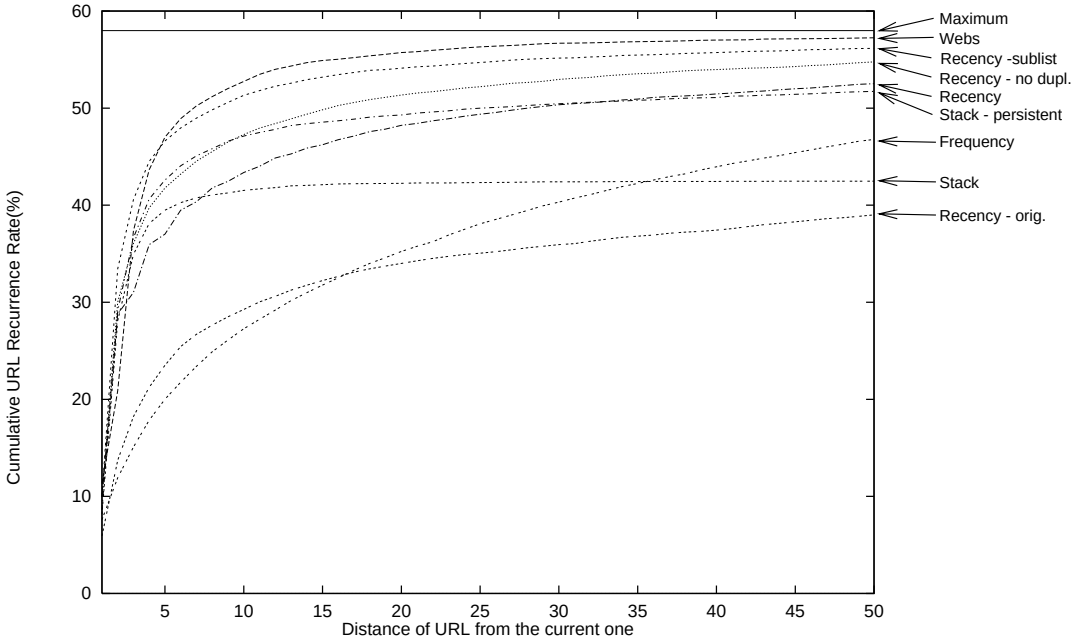

Figure 6.2. Cumulative probabilities of a recurrence over distances up to 50

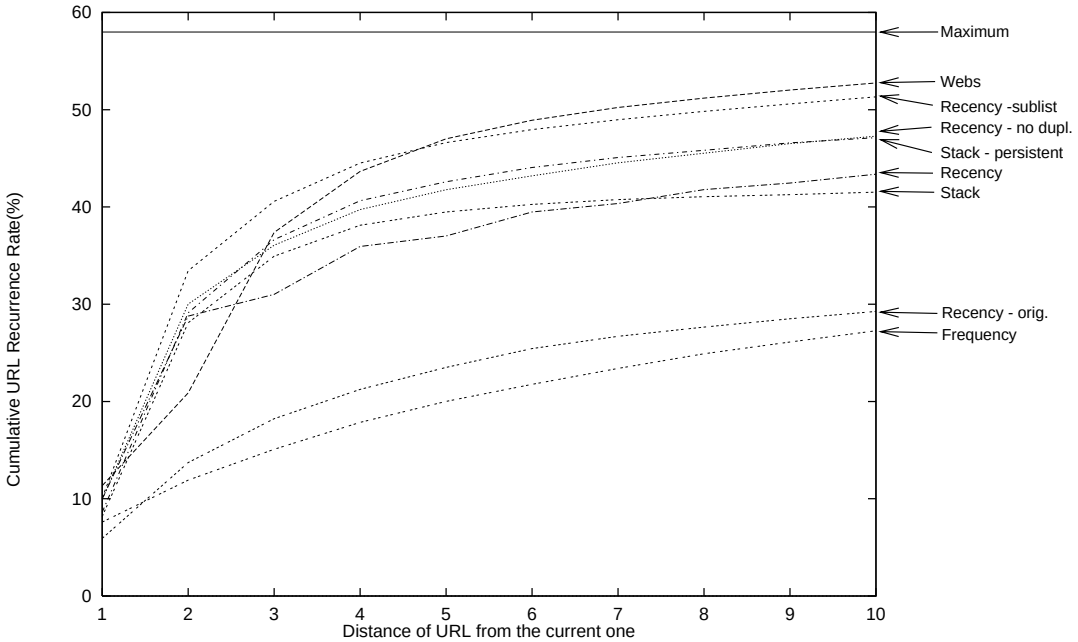

Figure 6.3. Cumulative probabilities of a recurrence over distances up to 10

6.2.4 Frequency ordering

Frequency ordered lists perform surprisingly poorly with R_{D10} = 27% compared to the benchmark of 43%, a difference of 16%. At this distance of 10, frequency exhibits the poorest performance of all 8 conditioning methods. However, its performance does improve for larger distances. For example, frequency ordering is only 6% lower than the sequential recency list at R_{D50}.

6.2.5 Stack ordering

Stack based history lists decrease the overall recurrence rate from 58% in the strict sequential case to 43%. This is because previously visited URLs are not available between sessions, and URLs visited in a session may not be on the stack later in that same session. Stack ordering performs moderately well for short distances e.g. R_{D5} = 40% vs. benchmark of 37%, although not as well as recency with duplicates saved in their latest position (R_{D5} = 42%). Its probability then plateaus, and overall performance fairs poorly over larger distances (e.g. R_{D20} = 42% vs. benchmark of 48%).

6.2.6 Persistent stack ordering

Persistent stack history lists also decrease the overall recurrence rate from the benchmark method (53% vs. 58%), but not as much as stack ordering. The persistent stack performs better than the stack for all distances. It also reports higher accumulated probabilities of recurrences than the benchmark case for distances 2 through 36 (R_{D1} = 9% vs. 10%; R_{D40} = 51.1% vs. 51.5%).

6.2.7 Recency ordered hyperlink sublists

This method outperforms pruning duplicates for all distances reported in Table 6.3. In fact, this method outperforms all others for distances of 2, 3, and 4 (see Figure 6.3). After a distance of 4, the sublists method is second best over all methods. The accumulated probability of a 10 item display is R_{D10} = 51% (cf. 43% for strict sequential) out of the 58% possible.

Note that for a distance of one, recency ordered hyperlink sublists perform better than both the benchmark case, and recency with duplicates saved in their latest position (10.1% versus 9.97% for the latter two methods). This is because the sublists method retains

separate history lists for each browser window that the user invoked. The non-sublist methods use a single history list independent of multiple windows—only the time of the navigation event is considered. Hence, the sublist method performs better because it retains more context—that is, the particular window in which the user invoked the event. And, because of the multiple sublists, the likelihood that the next URL will be at the top of the history list is increased.

6.2.8 Context-sensitive *Web subspace* history lists

The history list comprised of direct access URLs and the Web subspace sublists they encompass shows the best performance of all conditions evaluated for distances = 1, and >= 5. The accumulated probability of a 10 item display is R_{D10} = 53% (cf. 43% for strict sequential) out of the 58% possible.

6.3 Discussion

This section reviews conditioning methods in other work, and discusses our own results within the WWW navigation domain.

6.3.1 Relation to other work

Greenberg (1993a) evaluated 12 different conditioning methods to assess potential reuse opportunities in Unix *csh*. The major categories were: recency, frequency, alphabetic, directory sensitive by recency, commands only by recency, partial matching by recency, and command hierarchy. Greenberg found that the command hierarchy performed best with a R_{D10} = 55.5% out of the 74.4% possible. In his study, his method presented sublists of all command lines that share the same initial Unix command. Partial matching of command lines fared second best over all methods. The poorest method was recency ordering when duplicates were saved in their original position; this was also the case in our own study. Alphabetic ordering performed slightly better for larger distances (and worse for distances <= 16). Frequency ordering did not do as well as the recency sequential case, which agrees with our findings. Replacing duplicates in their original position for the recency list also performed better than strict sequential recency, as we also discovered.

The major difference between the Unix command line domain and the Web navigation domain appears to be the very strong recency effect in the latter. At R_{D10} the best conditioning method in the Unix case reports a probability of accumulated recurrences equal to 55.5%; given that R = 74.4%, which is the best a reuse facility can do in this case, the best predictive method is potentially 75% effective. For the best method in our study, R_{D10} = 53% and given that R = 58%, the method is 91% effective, at least potentially.

6.3.2 WWW navigation results

The recurrence rate R provides a theoretical limit on the performance of a reuse facility. It is reached only if the user reuses previous URL visits at every opportunity. However, finding and selecting URLs from a history list may be more work than some other available methods (e.g. clicking a hyperlink on the current page, clicking the *Back* or *Reload* button) especially if it is necessary to search the complete history list. Pragmatic considerations mean that only a small set of previous submissions are chosen and offered to the user as predictions. Of particular importance is the conceptual model that the history list presents, for this model is closely intertwined with the user's ability to predict whether the item they seek is present on the list. In the following discussion of our results, we consider the simplicity of the method's conceptual model as well as its performance.

The benchmark method, a strict sequential list of URLs ordered by recency performed reasonably well with a small set of URLs e.g. 10 items. 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.

Pruning duplicates is a simple yet effective way of improving the performance of a recency-ordered list. However, duplicates should be saved in their latest (vs. original) position. While this type of list does not show the exact sequence of URLs visited by the user, it still presents a clear conceptual model from which the user should be able to predict if the desired URL will be on the list.

Frequency ordering shows reasonable performance only at large distances. However, given a history list of 10 items, frequency ordering is the poorest method of the 8 evaluated. It may also be difficult for the user to predict which pages would appear on a

frequency-ordered list beyond the two or three that they visit the most. Hence, this method would probably best be applied to a few key URLs, and, if so, it should only be used as an auxillary method in conjunction with one that gives better performance.

The stack method found in most Web browsers shows reasonable performance only at very short distances; at larger distances it is the second worst predictor. Also, there are problems with this method's conceptual model, as Cockburn and Jones (1996) discovered in their usability study.

The persistent stack is an improvement over the stack method in terms of its recurrence probabilities over distance. However, it still suffers from the conceptual model difficulties that the stack method does. Also, the persistent stack can potentially contain many duplicate entries over time, and it may not necessarily contain all URLs visited. Hence, other methods that perform better are recommended over this one.

Recency ordered hyperlink sublists perform very well with the greatest recurrence probability over all methods for distances of 2, 3, and 4. However, this result is optimistic as greater cognitive and physical effort is involved in selecting items from the hyperlink sublists. Also, our algorithm did not account for errors—that is, navigating to a sublist that does not contain the desired URL. To make an accurate selection from a hyperlink sublist, the user must recall which main list item contains the desired URL. Also, note that hyperlink sublists provide access to potentially 55 URLs for a R_{D10} (the main list of 10 items + 9 items on sublist one + 8 items on sublist two + 7 items on sublist three, etc.).

The best method we evaluated—context-sensitive web subspaces—showed that 53% of all URL selections can be successfully predicted with a set of 10 items. Given that R = 58% on average, which is the best a perfect reuse facility could achieve, this method is potentially about 91% effective. However, context-sensitive web subspaces suffer from the same problems that recency ordered hyperlink sublists do. They require greater physical effort to select a sublist item, and greater cognitive effort to recall which sublist might contain the URL; both methods also assume perfect user behaviour. In addition, the context-sensitive web subspaces method's conceptual model may be more difficult to

comprehend. Users need to understand the notion of a direct access URL versus a hyperlink selection to grasp the way the history list is organized.

In conclusion, our analysis of conditioning methods shows that there are methods that can improve upon the effectiveness of the current stack-based method. In particular, recency is a reasonable predictor, especially when duplicates are saved in their latest position, and should be conceptually easy for the user to comprehend. Two methods that performed better may not work as well in practice due to additional physical overheads in making a sublist selection, and cognitive overheads in being able to reasonably predict whether the desired URL is actually on the history list and where it might be located. The limited use of history within current browsers (< 1% of navigation actions) may be due to these very factors. Therefore, further research is required to evaluate these methods in practice via short-term usability studies and long-term usage studies. This research is also important because these history mechanisms may change how a user browses e.g. the hyperlink sublist or web subspace methods will allow the user to *skip over* a page by choosing a hyperlink from a sublist.

7. Implications

This chapter considers the implications of our research. The first section proposes nine guidelines for the design of history mechanisms in WWW browsers. Section 7.2 evaluates Web browsers using these guidelines. Several standalone hypertext systems are also evaluated to provide additional examples of where guidelines are followed or not followed. However, this research does not claim to apply to the entire genre of hypertext systems.

7.1 Guidelines for history mechanism design in WWW browsers

Baecker, Grudin, Buxton, and Greenberg (1995) define *guidelines* as "collections of tests which can be applied to an interface to determine if it is satisfactory" (Baecker, Grudin, Buxton, and Greenberg, 1995, p. 74). Effective guidelines for history mechanism design have several criteria that we have considered:

- *context*: the guidelines specifically concern the design of history mechanisms within WWW browsers;
- *reasonable number*: nine guidelines are described;
- *appropriate level of specificity*: the guidelines are general enough to be applied to a variety of presentation approaches, and are quantitative where appropriate;
- *testable*: several guidelines are supported by our empirical analyses; ongoing analyses with other test methods such as usability studies and GOMS (Goals, Operators, Methods, and Selection) remains;
- *consistent*: the guidelines address different aspects of history mechanisms though contradictions may arise in some cases—these are discussed with each guideline.

The guidelines presented here are derived from those formulated by Greenberg (1993b) for the design of reuse facilities. Greenberg proposed three fundamental design requirements: a user's previously submitted activities should be available for recall, activities should be grouped into high-level task sets, and end-user customization of acitivities and task sets should be supported (Greenberg 1993b). Our discussion will, of

	Design guidelines
1.	Maintain records of URLs visited, and allow users to recall previous URLs from those records.
2.	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.
3.	A selectable history list of the previous 10 URLs visited provides a reasonable set of candidates for reuse.
4.	Other strategies for presenting the history list, particularly pruning duplicates and hierarchical structuring, increase the probability of it containing the next URL.
5.	History based on recency is not effective for all possible recalls because it lists only a few previous events. Alternative strategies must be supported.
6.	URLs already recalled through history should be easily reselectable.
7.	History items should have a meaningful representation.
8.	Support grouping of URLs into high-level Web tasks, and switching between tasks.
9.	Allow end-user customization of history data.

Table 7.1. Design guidelines for WWW browser history mechanisms

course, focus on the reuse of URLs where the *activity* relates to navigating to a URL, and a *task set* is a set of URLs that are related in some way.

The guidelines that Greenberg derived were partially motivated by research that shows that Unix *csh* history is poorly used in practice. Greenberg (1993a) reports that many people never use Unix *csh* history. Those who do tend to be sophisticated Unix users, yet even they do not use it much. On average, Greenberg found that less than 4% of all submissions were retrieved through history (out of the 75% potentially possible). Another interesting finding is that users did not refer very far back when using history. An average of 79-86% of all history uses referred to the last two command lines for novice programmers, while more sophisticated and/or experienced users achieved this rate with the last five command lines. There are several reasons for these findings including the lack of a permanent display of a user history, and the physical and cognitive overhead of recalling and editing previous events.

While our results show high use of the Back action (30%), less than 1% of navigation actions were related to Mosaic's Window History feature. The premise of this thesis is that Web browsers require suitable history mechanisms, and that existing designs can be improved upon. Hence, our objective in presenting these guidelines is to facilitate the

effective design of browser history mechanisms based upon our empirical results. Table 7.1 contains a summary of our nine guidelines.

1. 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 by the definition in Section 4.2. 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, there is an important caveat to this guideline. It may not make sense to present *all* URLs visited in certain situations, such as when users interact with forms-based Web applications. There are two reasons for this. First, Web applications may require the user to follow several steps, each of which generates a different URL. For example, a database query for economic data may have a page from which the user selects variables, another page from which the user selects the date range, and a final page that displays a graph of the result. In this situation it is not meaningful for the user to jump directly to the results page, if they have not selected the variables and dates first.

The second reason that all URLs should not necessarily be presented relates to the POST method of processing forms-based data. The POST method does not put the data from the form into the URL to be processed by the Common Gateway Interface (CGI) script, as the GET method does. Thus, returning to a results page that uses the POST method means that the CGI script has no knowledge of the user's previous inputs. The end result will be a Web page that is not in the format that the user expects.

2. 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. Physical effort may include clicking a hyperlink or button, opening a menu, selecting a menu item, and issuing a keyboard

command. Cognitive effort may include recalling a URL so that it can be typed into the URL field, scanning the history list for a page title, retracing one's steps to a previous URL from the current one, recalling how to navigate to a page from the current page, deciding which hyperlink to follow, formulating a search query, and parsing the results of a search.

Several factors in history mechanism design affect its ability 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 likely to access the history system to retrieve the URL versus navigating to it via other methods.
- The best predictions should be clearly distinguishable (e.g. placed at the top of the list) so that they are the first ones that the user sees.
- A minimum number of physical actions should be required to access and retrieve an item from the history system.
- 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. Instrumental in establishing context and orientation is reminding the user where they currently are and what they have already seen. Because users often pursue multiple paths and digressions (Foss, 1988), cognitive overhead will be reduced if the history mechanism can also show the user where they may want to go to next.
- Accessing the history mechanism should be minimally disruptive to the user's current task.

We have not evaluated the cognitive effort required for reviewing a particular conditioned set of predictions but recommend that this be undertaken as future research. A promising approach is demonstrated by Lee's (1992) work. Lee used a combination of the extended Model Human Processor and GOMS framework to model and predict the mental and physical effort involved in specifying a recurrent Unix command. Three history tool designs (command feature, history menu, step buffer) were compared that involve a combination of recall and/or recognition capabilities. Overall, Lee (1992) found that there

is a trade-off between the cognitive effort for selecting a history item versus the physical effort for reentering the item from scratch. History-based designs require less physical effort (e.g. typing, mouse selection) compared to typing but demand additional mental effort (e.g. recall, visual search). History-based designs are thus effective for poor typists, and when the mental operations required are simple and few. The task that Lee examined was a typing task which is quite different from the selection task that comprises most of our domain. Therefore, additional research is required to assess the cognitive effort associated with history use in graphical user interfaces.

3. A selectable history list of the previous 10 URLs visited provides a reasonable set of candidates for reuse. Greenberg (1993a) 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. For example, our results show that a menu of the previous 10 URLs visited covers, on average, 43% of all inputs. Doubling this to 20 items only increases the probability to 48%. 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 one, two and four (10%, 19%, and 5% respectively). These items should be easily accessible. For this reason, as well as people's natural tendency to scan lists from the top to the bottom, a short history list whose items appear in descending order by recency would work fairly well.

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. Otherwise, the user may face an exhaustive and futile search. Also, if the user can predict that the item will be on the list, they are more likely to use the list to retrieve the URL.

The argument for presenting only a small subset of previously visited items also applies to graphical network views. For example, in the NoteCards system, Foss states that, "in our experience, browser graphs containing 10 or 15 nodes approached the user's saturation point, depending of course, on the familiarity of the user with the network and on how the nodes were organized" (Foss, 1988, p. 88).

4. 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 are not covered by the last 10 items (26% of the recurring total) though doubling or tripling the size of the list does not increase its coverage much (see Figure 5.8). But it is these 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 it is important to explore alternative methods for conditioning the history list.

Pruning duplicates from a recency ordered list is a simple improvement that increased R_{D10} by 4%. The drawback to this method is that it does not preserve the true temporal order of the URLs the user has visited. However, we suspect that removing duplicates will not increase the difficulty of locating an item on the list.

Hierarchical structuring fares even better than pruning. We examined two types: context-sensitive Web subspaces (10% higher), and hyperlink sublists (8% higher). With these methods, a distance of 10 provides access to a maximum of 55 URLs when each main list item is associated with a single sublist. This is an optimistic value, however, since duplicates may occur, and sublists may not contain the maximum number of items. Also, it may be more difficult for the user to predict whether the URL will be in a particular sublist, and where it might be located. Finally, more physical actions are required to manipulate hierarchical lists. Still, the predictability of these methods are quite high, and they also provide some information about the structure of the Web space.

5. 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 the start-up document, and 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 (1993a) 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 3 most frequently visited pages. As discussed in Section 5.3, only a small number of URLs are visited often.

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 WWW domain. Second, for infrequently accessed URLs that have not been visited recently, the ability to search one's history could be beneficial. While the combination of these techniques have not been analyzed, they are worthy of future study.

6. **URLs already recalled through history should be easily reselectable.** If a user has selected an item from their history, the item is probably of more importance to them. Thus, it should be easier for them to retrieve that item in the future. This goal can be facilitated implicitly and/or explicitly. For example, certain conditioning methods favour the reselected item by propagating it to the top of the list based on recency of access, or increasing its access count for frequency ordering. Explicit methods for supporting this guideline might highlight the item on a list or in a graphical overview to show its selection during the current browsing session.

7. **History items should have a meaningful representation.** Semantic information about a history item is necessary to enable the user to easily locate the item whether it appears on a list or in a graphical display of some sort. Web pages are typically referred to by their URL or their HTML title tag. There are several problems with using the title tag: it may be absent, it may not be the same as the page title (which is usually an *H1* tag), and it may be too long to display easily. URLs, on the other hand, may be long and non-intuitive, and thus difficult to recall, type and/or parse. However, URLs can convey a great deal of information, and home page URLs are becoming more visible and widespread (e.g. referred to in the media, publications, etc.).

Given that the URL or HTML title tag are probably the best method for referring to a history item, and given the problems that each presents, there is currently no clear solution

to naming history items. However, there are some techniques that can improve the situation. First, we recommend the user be given a choice as to which method they prefer. The user may also want to specify the maximum length of the URL or title, and where the extra characters should be truncated (e.g. beginning, middle, end). Second, simple techniques can reduce the length of URLs. For example, the *http://* could be removed as could the *.html* or *.htm*. Finally, the user should be permitted to rename the item. This requires additional cognitive and physical effort, and may disrupt the user's current task. However, the benefits may exceed the costs, and the interface could significantly reduce the work involved. For example, changing the name of a bookmark in Netscape Navigator 2.0 requires displaying the Bookmarks menu, and then the Properties dialog. This process could be simplified by allowing the user to click on the bookmark title and edit it directly, similar to editing file names in the Macintosh Finder.

8. Support grouping of URLs into high-level Web tasks, and switching between tasks. Hypertext encourages connections between information that is related in some way. Thus, a sequence of pages that a user browses may have a particular context that would be convenient to present to the user at a higher-level. We explored this concept in several ways that include identifying locality sets and longest repeated subsequences, and evaluating the predictiveness of context-sensitive web subspaces. The latter proved to be the best conditioning method of the eight examined. Locality sets and paths may hold promise if their algorithms were to consider more domain knowledge.

9. Allow end-user customization of history data. We believe that the automatic capture of history data is essential to reduce the physical and cognitive overhead of recording URLs for reuse. However, users may also want to customize the various attributes of a history mechanism or save portions of their history. If users are to take advantage of this feature, it is essential that the physical and cognitive overhead of managing history data be kept to a minimum.

Some of the history mechanism customizations may include the following:

- overriding the system chosen candidates on a *frequently visited pages* toolbar;

- changing the name of a page from its title to its URL, or to a user-specified name;
- removing unwanted Web subspaces and the URLs within them;
- specifying the length of a list;
- moving history items to other browser facilities e.g. Bookmarks/Hotlist.

The last feature is a particulary important one because users often forget to bookmark a page or may not think that the item is of interest until later. The automatic capture of page visits better supports Lucy Suchman's notion of *situated action*—the ad hoc rather than planned responses people make to the actions of others, and to the contingencies of particular situations (Greenberg, 1993a). Because a history facility allows the user to select and revisit Web pages, it responds well to the circumstances of a situation where the user may forget to perform the separate task of bookmarking a page, or only later realize its current value to them.

7.2 Evaluation of current hypertext system history mechanisms

How do current hypertext systems fare against the guidelines proposed in Section 7.1? This section considers each guideline presented in Section 7.1, and provides examples of systems discussed in Chapter 2 that meet or do not meet the guideline.

1. Maintain records of URLs visited, and allow users to recall previous URLs from those records. Web browsers fail to maintain adequate records of URLs visited. While they do maintain a global history list, browsers do not give users access to the history data; it is only used to indicate which hyperlinks have been recently accessed. In fact, Netscape Navigator 2.0 now stores this data in binary form in the Unix environment. Also, the sessional history list may not even retain all URLs visited for the duration of the current session because it is stack-based (see Section 6.2.3).

Add-on software attempts to remedy these problems. For example, the Overdrive Logger (Section 2.2.6) records all URLs visited for one year in individual files for each day. While an improvement over Netscape Navigator 2.0's capabilities, Overdrive is not storing enough data (ie. the user will not be able to find a URL visited long ago), and it is not easy for the user to integrate history data from the multiple files.

The Deckscape experimental WWW browser supports this guideline by retaining all pages until the user explicitly discards them (Section 2.2.8).

2. 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. Minimizing physical and cognitive effort in history mechanism use requires an appropriate ordering of past submissions to increase the predictive ability of the history set. Many history mechanisms found in hypertext systems use a recency-based ordering which is a reasonable predictor. However, some systems place the most recent item at the bottom of the list, which is counter to people's natural tendency to scan in a top-down manner.

Frequency ordering is another method that is found in existing systems. Monk proposed extending the Personal Browser (Section 2.1.5) by having the application prompt the user to add a card to their customized card if it has been visited a certain number of times. However, our assessment of frequency ordering in Section 5.2 shows that it is a mediocre method of assessing a user's current interests.

Cognitive and physical considerations may explain the low percentage of navigation actions that involved history list selections (<1%). Though the Back action gives users access to URLs a distance of two away (the distance with the highest probability of a recurrence), distances 3 through 6 together comprise 11% of recurrences. We expect that URLs at these distances may be easier to access via a history list (where the user can *recall* the page by seeing its title) versus multiple Back actions (where the user may not know exactly which page Back will display). However, the following factors may be contributing to the low use of the list:

- users' inability to predict whether the URL is on the stack-based history list;
- the numerous actions required to access Mosaic's Window History dialog, and its invisibility;
- cognitive overhead associated with switching to the use of the history tool from typical user interactions (clicking hyperlinks or browser buttons);
- time required to find the item on the history list.

Wetherall's Web history page (Section 2.2.6) creates an additional overhead. The history list is presented as an actual Web page so it will replace the current page, or must be viewed in a separate browser window that will probably overlap the current window considerably.

3. A selectable history list of the previous 10 URLs visited provides a reasonable set of candidates for reuse. Most hypertext systems display considerably more or much less than 10 items in their history lists. Hypertext systems tend to take one of three approaches to pruning history items: the history list grows indefinitely, has an arbitrary cut-off, or offers a customizable length.

1. *Indefinite growth*: In Intermedia, a *Web* stores a list of the documents the user visited in a previous session though Intermedia does not appear to control the growth of the Web in any way (Section 2.1.4). Other systems that lack an automatic pruning strategy include Ayers and Stasko's Graphic History View (Section 2.2.3), and WebNet (Section 2.2.3).
2. *Arbitary cut-off*: Bernstein's Hypergate system is an example of a system that uses an arbitrary cut-off (Section 2.1.2). Hypergate offers the user a menu of recently visited pages that is limited to 30 pages. This number seems to have been chosen arbitrarily, and is excessively long. HyperCard Recent offers a maximum of 42 card miniatures on its Recent card, a value chosen based upon the size of the display and miniature (Section 2.1.2). The Overdrive Logger keeps only a year's worth of history, as described in Section 7.2.1. Its heuristic for rolling over log files is not related to user browsing behaviour.
3. *Customizable length*: Web browsers allow the user to specify the maximum number of days for which a hyperlink will change colour to indicate that it was recently visited. Wetherall's Web history page views are dependent upon this parameter for they display the history items found in the user's global history list. Given that the average number of URLs visited over our 6 week study was 817, the history page will contain an excessive number of items. (While a *path compression technique* is used to economize on the length of the display, it does so by only 10% (see Section 2.2.6).)

Web browsers do not allow the user to edit their history list, though Deckscape (Section 2.2.8) and Apple's Internet Explorer (Subection 2.2.6) do permit the user to discard items. Customization does require additional user effort, which is why a sound automatic strategy is also important.

Web browsers do not fit into either of these categories. They prune their history list based upon the stack model. Items are removed when the user loads a page while at some point other than the top of the stack (see Section 6.1.6). 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.

Some hypertext systems do provide a reasonable number of history items. For example, The National Museum of Denmark's museum information system has a limit of 8 items in their visual history "cache", displayed as graphic miniatures along the bottom of the screen (Section 2.1.2). This limit is likely dictated by the constraints of the display space but the 8 miniatures are sufficient to capture the artifacts the user is likely to return to, and recency is an effective method used for selecting history candidates.

4. Other strategies for presenting the history list, particularly pruning duplicates and hierarchical structuring, increase the probability of it containing the next URL. HyperCard's *Recent* facility uses a recency approach and replaces duplicates (Section 2.1.2). However, the revisited card is replaced according to their original position. Our results and Greenberg's study (1993a) show that this is a very poor conditioning method. It means that a recently visited card may not appear on the history list.

Hierarchical structuring is becoming a popular method of presenting history. Ayers and Stasko's (1995) Graphic History View provides a hierarchical representation based upon parent-child relationships between URLs (Section 2.2.3). This representation has some drawbacks especially when the pages browsed are linked in more of a network structure. DeckScape has a similar problem. While it places the child page after its parent page when the user has backtracked to a higher level in the tree, the representation may be confusing if it does not resemble the hypertext network structure upon which the pages are based (Section 2.2.8).

Another strategy for presenting history is to emulate a stack. This method is used by current Web browsers. However, Section 6.2 shows that the stack only performs reasonably well for short distances. This method suffers from other difficulties. First, *loading* a page versus *recalling* it has a different effect upon the stack. The former event occurs when the user revisits a page by selecting a cyclic link to it—this puts the URL on top of the stack. The latter event occurs when the user revisits the page using the Back action, the Forward action or a Window History selection—this adjusts the pointer to the current page in the history list. Users tend to view these events as equivalent since they result in the same change to the browser window. But loading a page while at some point other than the top of the stack causes all pages above the current position to be lost. Thus, in their usability study of history lists in three popular WWW clients, Cockburn and Jones (1996) discovered that users consider navigation behaviour using Back, Forward, history list selections, and cyclic Web page links unpredictable or "non-deterministic." Users also stated that they expected the history list to be temporal or linearly incremental. This may be due to its name (history *list*), and its linear representation. Also, the history list's lack of visibility means that it is more difficult for users to observe its behaviour and modify their incorrect mental models (Cockburn and Jones, 1996).

5. History based on recency is not effective for all possible recalls because it lists only a few previous events. Alternative strategies must be supported. Most of the alternative strategies that exist in the WWW domain are currently provided through helper applications or add-on software. For example, Overdrive Logger provides access to inter-sessional history through browsable lists of URLs, though only one year of history is kept, and the user must recall which day they visited the site to access the appropriate history file (Section 2.2.6).

Searching is another alternative strategy that is being incorporated into large, distributed hypertext systems. ISYS HindSite provides a search interface to the user's global history list (Section 2.2.5). Netscape Navigator 2.0 also incorporates a search feature, but it only searches the user's bookmarks list; a logical extension of this feature would be to search one's global history list as well.

In conclusion, most alternative strategies are add-on applications that are not well-integrated with the browser. However, there is a definite trend towards incorporating add-on features directly into browser software.

6. URLs already recalled through history should be easily reselectable. This guideline is poorly supported in both non-distributed and distributed hypertext systems. While these systems may indicate what nodes have been visited, they do not distinguish between visited nodes and nodes that have been selected from a history view. One could argue that the Back action and stack-based history mechanism in Web browsers supports this guideline to some degree—given the dominant browsing modality (displaying a single document or node at a time combined with a depth-first navigation strategy), Back provides a quick and easy method of backtracking to previous pages. Another example where reselection is considered is within the National Museum of Denmark's museum information system. It presents the eight most recently visited nodes along the bottom of the screen which implicitly facilitates reselection through the history conditioning method of recency.

7. History items should have a meaningful representation. History items in hypertext may be represented in several ways: textual identifiers, graphical miniatures, and/or graphical overviews that show the relationship between nodes.

As discussed in Section 7.1, naming URLs is an issue for many reasons. The objective of the URL's textual description is to make it quick and easy for the user to identify the URL they seek from a group of URLs whether this be in a simple linear list or a graphical view. Current systems take several approaches to this problem. Wetherall's Web page views of inter-sessional history represent history items by the actual URL, which may not appropriate for most users (Section 2.2.6). Cockburn and Jones are investigating different methods of displaying page titles in WebNet (Section 2.2.3). 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 allows users to display Web pages by their URL, page

title, or a thumbnail (Section 2.2.3). Also, the user can also control the abbreviation of page titles.

Some history systems provide graphical miniatures of the nodes to aid in their identification. For example, with HyperCard Recent, the user is expected to recognize the card they want by its layout (Section 2.1.2). This can be difficult if many cards have a similar layout. However, this approach works well for the National Museum of Denmark's information system (Section 2.1.2). Since each screen in this system includes a graphic of a museum artifact that is distinguishable as a miniature, the user can easily select the screen they wish to return to. This representation is unlikely to work as well for the thumbnails in Ayers and Stasko's Graphic History View. While thumbnails of Web pages are an interesting idea, their utility needs to be assessed. For example, are pages really distinct enough to be easily recognizable from a thumbnail image? Is the additional screen space required worth it?

WebNet and the Graphic History View also indicate the structure of the Web space; the former represents it as a network while the latter uses a hierarchical representation.

8. Support grouping of URLs into high-level Web tasks, and switching between tasks. Two experimental WWW systems meet this guideline to various degrees. First, WebNet's web subspaces appears to be an intuitive way of partitioning a user's navigation history (Section 2.2.3). However, the usability of this approach has yet to be assessed.

Second, Deckscape's deck metaphor and its impressive implementation make it a very powerful tool for organizing and switching between groups of Web pages. Some methods for organizing decks that have been proposed include: pages from a particular site, pages accessed during a particular navigation sequence, pages that are hyperlinks on a particular page, pages that meet search criteria, or pages that the user has manually pulled into a separate deck. Deckscape's other strength is the ease at which users can switch between different decks, and the pages in each deck. All decks are displayed in the main browser window where they can be moved, iconified, resized, etc. Individual pages are accessed by leafing through the deck's pages one at a time, jumping to the top or bottom of the deck,

and moving to a particular page by choosing it from a list of the deck's current contents (Brown and Shillner, 1995).

9. Allow end-user customization of history data. Some examples exist of WWW history mechanisms that support this guideline in various ways. Ayers and Stasko's Graphic History View allows the user to zoom into certain portions of the hierarchy, and condense branches of the tree (Section 2.2.3). WebNet allows both manual and automatic creation of web subspaces (Section 2.2.3). Overdrive's Organizer module permits the user to filter, sort, organize and save portions of their browsing history. However, it could be more tightly integrated with the history feature, and reduce the physical effort involved by making the interface more directly manipulatable. For example, 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 (Section 2.2.8). 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 and Shillner, 1995). Finally, decks can be moved, iconified, or resized by the user.

7.3 Concluding Remarks

This chapter presented nine guidelines for history mechanism design in WWW browsers. These guidelines were derived from those formulated by Greenberg (1993b) for the design of reuse facilities. The guidelines chosen from Greenberg's set were easily reapplied to the WWW domain, which clearly demonstrates their generativity. Our guidelines for history mechanism design were then used to evaluate current browser history mechanisms; in some cases, examples of designs that meet or do not meet a particular guideline were also taken from non-distributed hypertext systems.

8. Conclusion

Repetition occurs within many types of user interactions. We have shown that the same is true of the World Wide Web. History mechanisms have been part of user interfaces for some time. They offer previous actions for reuse, and reduce the cognitive and physical overhead of specifying actions from scratch. But how effective are the history mechanisms that graphical Web browsers provide? Can their predictability be improved? Are they even warranted—that is, do users revisit the same Web page?

We began our examination of these questions with a survey of existing history mechanisms, both within non-distributed hypertext systems, and graphical WWW browsers. Many mechanisms common in earlier hypertext systems are now standard features in WWW browsers. However, our research shows that browsers provide incomplete support for reaccessing pages, and existing history mechanisms are piecemeal, ad hoc, and unevaluated.

Much of our research was based upon a WWW navigation usage study in which 23 experienced Web users browsed with a modified version of Mosaic 2.6 for approximately 6 weeks. Their data was analyzed in various ways. First, we calculated summary statistics and compared the navigation methods used with an earlier study by Catledge and Pitkow (1995). We also calculated and compared recurrence rates among the two groups. Second, we examined five different patterns that evolve over time: URL vocabulary growth, URL recurrence rate as a function of distance, frequency of URL accesses, locality, and longest repeated subsequences.

Our findings suggested 7 different conditioning methods that we might compare to the existing stack-based approach for presenting the user's page visit history. We applied these conditioning methods to our usage data, and used the results when formulating design guidelines based upon those developed by Greenberg (1993c) for reuse facilities. The guidelines were used to evaluate existing hypertext-based history mechanisms.

The remainder of this section will outline the contributions of this research, as well as discuss future research directions.

8.1 Contributions

There are three major contributions of this thesis. The first is that our research contributes to a growing body of empirical data influencing the design and use of history mechanisms within user interfaces. Furthermore, our study is the only examination to-date of patterns in users' revisits to WWW pages. Our primary findings are as follows.

1. When accessing URLs, there is high use of *Open URL* and *Back* actions, and low use of the hotlist and history list.
2. 58% of pages are revisited; hence, Web browsing qualifies as a recurrent system.
3. New URLs are incorporated into the user's repertoire of visited URLs at a regular rate. However, there are also variations in: the vocabulary growth rate during a user's browsing history, the use of navigation activities across subjects, and the use of navigation activities across a user's browsing timeline.
4. There is a very strong recency effect in that most recurrences occur within a short distance of each other.
5. A few pages were heavily accessed while many pages were only visited once or twice during the 6 week study period.
6. Users browse within clusters of pages (i.e. locality sets), but the number of pages tends to be very small, and users tend to stay in the cluster for a very short duration. Also, most clusters were not repeated.
7. Users do repeat sequences of URL visits but the sequences tend to be short, and were only repeated twice on average. Incorporating domain knowledge into this algorithm could improve the detection of navigation paths.

The second contribution of this thesis relates to the reapplication of conditioning methods for page visit recurrences, based upon Greenberg's (1993a) work. We found that the current stack-based method of offering previously visited pages for reuse shows reasonable performance only at very short distances, and a study indicates that there are problems with this method's conceptual model. Offering the most recently accessed URLs

is a reasonable predictor, especially when duplicates are saved in their latest position; this method should also be conceptually easy for the user to comprehend, and for the designer to implement. Two conditioning methods gave better results than recency with no duplicates: context-sensitive Web subspaces, and recency ordered hyperlink sublists. While promising, these methods may not work as well in practice due to their additional cognitive and physical overheads.

The third contribution of this thesis concerns the specification of guidelines for WWW history mechanism design. The guidelines were derived from those developed by Greenberg (1993a) for the design of reuse facilities, and several of them are directly supported by our empirical results. Briefly, the guidelines address the following: access to history data (pages the user has visited); reducing the cognitive and physical effort of using a history mechanism; providing a reasonable set of candidates for reuse; improving the conditioning method; supporting alternative strategies; and allowing end-user customization of the history data.

We evaluated both graphical browsers and non-distributed hypertext systems against these guidelines. The evaluation shows that existing browsers do not meet most of our criteria. However, there are commercial helper applications that fill-in some of the gaps, and there are several promising experimental systems that with some modifications could prove very effective.

8.2 Future Directions

There are three research directions in which the work begun in this thesis could be extended: studies that assess the impact of different browser and HTML artifacts upon reuse, studies of other reuse patterns, and studies that apply our results to other domains or aspects of navigation.

First, additional research is required to assess the impact of different browser and HTML artifacts that may alter the reuse patterns that we have discovered. For example, frames are an HTML extension that allow multiple pages to be viewed within the browser window. A common use of frames is to display an index page on the left, and a content page on the right—the index page allows one to change the content page without

navigating back and forth between pages. Alternative browser metaphors such as DeckScape may also affect the reuse patterns. Finally, alternative history mechanisms such as WebNet will likely change how users reaccess pages. Thus, as Web tools evolve, there is a need to continually re-evaluate our results. The evaluations should also involve assessments of the physical and cognitive effort in using different history mechanisms.

Second, additional patterns of reuse should be investigated to improve the probability that the next URL a person chooses to visit is available in a small set of predictions offered to them by the history mechanism.

Third, the results that we have obtained could be reapplied to other domains or other aspects of the WWW where knowledge of page accesses are important. While we have focused on revisits to Web pages, our analyses and results could be extended to research into desktop management systems. In fact, there is a trend towards integrating Web browsers and desktop management systems so that there will be one interface for navigating local files and distributed information. And because of the inflexibility of hierarchical structures for organizing this information, hypertext capabilities will exist. Hence, studies about Web usage may be very relevant to the design of our future, personal desktop.

Our research may also motivate alternative approaches to other aspects of the WWW that need to consider page reuse. First, the strong recency effect we identified has implications for the duration that items are kept in the user's cache. Next, on an aggregate level, extensions to our research may suggest the rate at which pages are refreshed on a proxy server. Third, besides just offering the next URL to the user, the history mechanism could also prefetch the page so that it is already in the user's cache when they request it. Finally, patterns in how people reaccess Web pages may be useful models in the design of intelligent agents that are predicted to browse on our behalf in the future.

9. References

Akscyn, R., McCracken, D., and Yoder, E. (1988). KMS: A Distributed Hypermedia System for Managing Knowledge Organizations. *Communications of the ACM*, 31(7), 820-835.

Apple Computer, Inc. (1996). Cyberdog Feature Information. *http://cyberdog.apple.com/features.html*.

Ayers, E. and Stasko, J. (1995) Using Graphic History in Browsing the World Wide Web. In *Proceedings of the Fourth International World Wide Web Conference*, http://www.w3.org/pub/Conferences/WWW4/Papers2/270/, Boston, MA.

Balasubramanian, V. (1994). State of the Art Review on Hypermedia Issues And Applications. *http://www.isg.sfu.ca/~duchier/misc/hypertext_review/index.html*.

Becker, R., Grudin, J., Buxton, W., and Greenberg, S. (1995). *Readings in Human-Computer Interaction: Towards the Year 2000 (2nd ed.)*. Morgan Kaufmann.

Berghel, H. (1996). The Client's Side of the World Wide Web. *Communications of the ACM*, 39(1), 30-40.

Berners-Lee, T., Cailliau, R., Groff, J., and Pollermann, B. (1992). World Wide Web: the information universe. *Electronic Networking: Research, Applications, and Policy*, 2(1), 52-58.

Bernstein, M. (1988). The bookmark and the compass: Orientation tools for hypertext users. *ACM SIGOIS Bulletin*, 9(4), 34-45.

Brown, M. and Shillner, R. (1995). DeckScape: An Experimental Web Browser. In *Proceedings of the Third International World Wide Web Conference*, http://www.igd.fhg.de/www/www95/papers/90/deckscape-final-v1/paper.html, Darmstadt, Germany.

Bush, V. (1945). As We May Think. *The Atlantic Monthly*, July, 101-108.

Catledge, L. and Pitkow, J. (1995). Characterizing Browsing Strategies in the World Wide Web. In *Proceedings of the Third International World Wide Web Conference*,

http://www.igd.fhg.de/www/www95/papers/80/userpatterns/UserPatterns.Paper4.formatted.html, Darmstadt, Germany.

Cockburn, A. and Jones, S. (1996). Which Way Now? Analysing and Easing Inadequacies in WWW Navigation. *Int. J. Man-Machine Studies*, 44, submitted for publication.

Conklin, J. (1987). Hypertext: An Introduction and Survey. *IEEE Computer*, 20(9), 17-41.

Crow, D. and Smith, B. (1992). DB_Habits: Comparing Minimal Knowledge and Knowledge-Based Approaches to Pattern Recognition in the Domain of User-Computer Interactions. In Beale and Finlay (Eds.), *Neural Networks and Pattern Recognition in Human-Computer Interaction*, 39-61. Ellis Horwood, NY.

December, J. (1994). Challenges for Web Information Providers. *Computer-Mediated Communication Magazine*, 1(6), 8-14, http://sunsite.unc.edu/cmc/mag/1994/oct/toc.html.

Foss, C. (1988). Effective browsing in hypertext systems. In *Proc. RIAO'88 Conf. User-Oriented Context-Based Text and Image Handling*, Cambridge, MA, 82-98.

Greenberg, S. (1993a). *The Computer User as Toolsmith: The Use, Reuse, and Organization of Computer-based Tools*. Cambridge series on human-computer interaction. Cambridge University Press.

Greenberg, S. (1993b). Supporting command reuse: mechanisms for reuse. *Int. J. Man-Machine Studies*, 39, 391-425.

Greenberg, S. (1993c). Supporting command reuse: empirical foundations and principles. *Int. J. Man-Machine Studies*, 39, 353-390.

Greenberg, S. and Witten, I. (1988). Directing the user interface: How people use command-based systems. In *Proceedings of the 3rd IFAC Conference on Man-Machine Systems*, Oulu, Finland.

Hanson, S., Kraut, R., and Farber, J. (1984). Interface design and multivariate analysis of Unix command use. *ACM Transactions on Office Information Systems*, 2(1), 42-57.

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

Keyes, E., Sykes, D. and Lewis, E. (1989). Technology + Design + Research = Information Design. In Barrett, E. (Ed.), *Text, Context, and HyperText*, 251-264. The MIT Press, Cambridge, MA.

Lee, A. (1992). *Investigations into History Tools for User Support*. Ph.D. Thesis, Department of Computer Science, University of Toronto, Ontario, Canada.

Madison, A. and Batson, A. (1976). Characteristics of program localities. *Communications of the ACM*, 19(5), 285-294.

Monk, A. (1989). The Personal Browser: a tool for directed navigation in hypertext systems. *Interacting with Computers*, 1(2), 190-196.

Mylonas, E. and Heath, S. (1990). Hypertext from the Data Point of View: Paths and Links in the Perseus Project. In *Proceedings of the First European Conference on Hypertext*, Paris, France, 324-336.

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

NCSA (1996). NCSA Mosaic ™ for the X Window System. *http://www.ncsa.uiuc.edu/SDG/Software/XMosaic/.*

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

Netscape Communications Inc. (1996b). Netscape Navigator Smartmarks Data Sheet. *http://home.netscape.com/comprod/smartmarks.html*.

Nielsen, J. (1990a). *Hypertext and Hypermedia*. Academic Press, New York, NY.

Nielsen, J. (1990b). "The Art of Navigating Through Hypertext." *Communications of the ACM*, 33(3), 296-310.

Nielsen, J. (1995). *Multimedia and Hypertext: The Internet and Beyond*. AP Professional, Boston, MA.

Trigg, R. (1988). Guided Tours and Tabletops: Tools for Communicating in a Hypertext Environment. *ACM Transactions on Office Information Systems*, 6(4), 398-414.

Utting, K. and Yankelovich, N. (1989). Context and Orientation in Hypermedia Networks. *ACM Transactions on Information Systems*, 7(1), 58-84.

Wetherall, D. (1995). Searching with History on the Web. *http://www.tns.lcs.mit.edu/~djw/history/*

Wright, P. (1991). Cognitive overheads and prostheses: some issues in evaluating hypertexts. In *Proceedings of the ACM Hypertext '91 Conference*, San Antonio, Texas, 1-12.

Appendix A: WWW Browser Usage Study

A.1. Consent Form

University Of Calgary Department of Computer Science

Thesis Research

Title of Investigation: World Wide Web Browser Usage Study

Investigator: Linda Tauscher

Description of Research Project: This research examines patterns of World Wide Web usage with the graphical browser, Mosaic.

This is to certify that I, _____, hereby agree to allow my World Wide Web browsing data to be used by other researchers.

I understand that the data will only be given upon request to credible researchers who will also honour the confidentiality of the data. These researchers will not be allowed to distribute the data further.

I understand that reasonable efforts will be taken to protect my identity; in this case, any references to my process id and username within my data file will be replaced with random values.

However, due to the nature of the data and domain, I understand that it may still be possible for my identity to be detected.

_____ _____

Date Participant's Signature

I, the undersigned, have fully explained the investigation to the above individual.

_____ _____

Date Investigator's Signature

A.2 Instructions to Participant - Orientation Session

In this study you will be helping me to identify usage patterns within World Wide Web browsing. The study will be run in two parts:

1) Today, you will go through a orientation session for the data collection phase. I will help you set-up your computer account so that any browsing you do during an approximately one-month period will be logged to a file.

2) After I have had an opportunity to analyze the data, we will set-up an interview session in which I may ask you about some of your browsing activities during the data collection phase.

After the interview, I will share some of my current insights into WWW browsing if you are interested.

If at any time during the experiment you do not wish to continue participating for any reason you may leave without any repercussions. Please notify me (via email or phone) if you do so.

Give participant consent form now

While using this modified version of Mosaic, some actions that you perform will be logged to a file which only I have permission to read. It is important that you do not modify your browsing behaviour in any way—just do what you normally do. Also, please try to use this version of Mosaic for all of your browsing sessions (unless you are browsing on another platform of course).

I will contact you when the data collection phase is complete; after that you can return to using your favourite browser. I anticipate that you'll be using Mosaic for about one month. If you have any questions during that time, please contact me.

Now we are going to help you to configure your account so that you can run the modified verison of Mosaic.

Configure user's account to run modified version of Mosaic 2.6

1. Set the path

- Put the following as the first item in the subject's last path statement (.cshrc file; set path= statement)

```
/home/grouplab/tauscher/browser_study

source .cshrc
rehash
```

- This directory contains the executable for Mosaic, and three symbolic links (mosaic, netscape, Netscape).
- If you must run Netscape, type netscape-orig (invokes Netscape 1.1N).

2. Update Window Manager menus

- Change Window Manager menu (if this is how they invoke their browser)
- The full path for Mosaic is:
  `/home/grouplab/tauscher/browser_study/Mosaic`

Specify home document in .Xdefaults file

`e.g. Mosaic*homeDocument: http://www.cpsc.ucalgary.ca/~tauscher`

Convert Netscape bookmarks to Mosaic format

Ask subject if they already have a Mosaic hotlist file (`mosaic-hotlist-default.html`); if so, rename it (if they want to use their Netscape bookmarks file instead)

To invoke a script which runs the Perl script to convert the Netscape hotlist, type the following in the subject's home directory:
`~tauscher/grouplab/browser_study/convert_bookmarks`

Testing / Training

1. Ensure that proper version will run (do `which mosaic` at the command line)
2. Start Mosaic
3. Ensure that proper home document appeared
4. Check that hotlist converted okay (view the hotlist)
5. Explain differences between Netscape and Mosaic 2.6 (according to participant's needs)
6. After the session, check that logging is working okay; delete test session data (first set of data) at the end of data collection phase

A.3. Handout: Differences between Mosaic v2.6 and Netscape

The purpose of this document is to explain some of the major differences between Mosaic v2.6 (the latest supported release) and Netscape 1.1 to ease the transition you are probably making from the latter to the former.

1. Session History

The pages that you have visited during the current session can be viewed by selecting Window History from the Navigate menu. To go to one of these URLs, simply double-click on the title. Note that you have to bring up the dialog box to view your history; also, the URL last visited appears at the bottom of the list, not at the top of the list as in Netscape.

2. Hotlist

Mosaic calls its bookmark list a hotlist. You can view it by opening the Hotlist dialog from the Navigate menu. This version of Mosaic allows you to create hierarchical hotlists. To do so, first create a new header by clicking Insert, and selecting List. Then, select a URL to put into this list, click Copy, double-click (or select list name and click Go To) the list name you want to put the URL into, and click Insert. A dialog will appear with the URL and title of the item you copied.

3. Typing URLs

In Mosaic, URLs can be entered directly into the URL field. However, you must include the http:// portion of the URL (this can be omitted from Netscape).

4. Buttons

Mosaic has a variety of command invocation methods. Notice that buttons appear at the bottom of the browser window, rather than the top as with Netscape.

5. Accelerator Keys

Mosaic allows menus to be accessed through accelerator keys. Each menu label contains an underlined letter, which when pressed in conjunction with the Alt key will open the menu. Press the underlined letter for the menu item you want to select that item.

6. Shortcut Keys

Mosaic has shortcut keys for many commands. You can learn about these by choosing the On Window menu item from the Help menu. The News shortcuts are missing from this list. They are as follows:

>	next thread	,	previous article
<	previous thread	.	next article

7. News

Mosaic will retrieve news groups, an index of news articles per group, and actual articles. A URL within an article is not converted into a hyperlink via Mosaic, as it is with Netscape

8. Tables

Images and links inside table cells are properly dealt with by Netscape (since they invented it!) but Mosaic v2.6 cannot handle this feature.

If you have any questions, please contact me at *tauscher@cpsc.ucalgary.ca* or 220-7691.

A.4 Mosaic v2.6 User Interactions Logged

This appendix identifies the user interactions logged during the WWW browser usage study. Not all possible user interactions were logged. We were specifically interested in navigation actions, and activities involving the hotlist and window history. We also tracked exiting from the software, closing a window, and interrupting a URL transfer.

The table below contains the following data:

- *Action*: This column identifies a high level browser user action. There may be a series of user interactions necessary to realize this action (e.g. deleting a hotlist item) or only one user interaction may be required (e.g. going back to the previous page using the mouse).
- *Path to Action*: This column identifies the particular mechanism used to invoke the action. The path is presented as a hierarchy from the general to the specific interface component. For example, if a user selects an item from their hotlist using the mouse, they first choose the *Navigate* menu, then select the *Hotlist* menu item, and then select the URL from the hotlist *Dialog* box. Thus, the path to this *Open URL* action is reported as: *Menu/Navigate/Hotlist/Dialog*.
- *Same or New Window*: The column identifies whether the particular action and path combination occurs in the same browser window or generates a new browser window.
- *Navigation Action*: This column identifies whether the particular action, path, and window combination is a navigation action. Navigation actions result in the display of a page in the browser window.

Action	Path to Action	Same or New Window	Navigation Action
Back	Menu/Navigate/Back	Same_Window	Yes
	Keyboard/Navigate/Back	Same_Window	Yes
Binary_Transfer	Mouse/Document/Anchor	Same_Window	Yes
Clear_Global_History	Menu/Options/Clear_Global_History	Same_Window	No
Clone_Window	Menu/File/Clone_Window	New_Window	Yes
	Keyboard/File/Clone_Window	New_Window	Yes
Close_Window	Menu/File/Close_Window	Same_Window	No
	Keyboard/File/Close_Window	Same_Window	No
Exit	Menu/File/Exit_Program	Same_Window	No
External_Viewer	Mouse/Document/Anchor	Same_Window	Yes
Forward	Menu/Navigate/Forward	Same_Window	Yes
	Keyboard/Navigate/Forward	Same_Window	Yes
Help	Menu/Annotate/Annotate/Dialog	New_Window	Yes
	Menu/Annotate/Audio_Annotate/Dialog	New_Window	Yes
	Menu/File/CCI/Dialog	New_Window	Yes
	Menu/File/Open_URL/Dialog	New_Window	Yes
	Menu/File/Mail_To/Dialog	New_Window	Yes
	Menu/File/Print/Dialog	New_Window	Yes
	Menu/File/View_Source/Dialog	New_Window	Yes
	Menu/File/Find_In_Current/Dialog	New_Window	Yes
	Menu/Help/Demo	New_Window	Yes
	Menu/Help/Manual	New_Window	Yes
	Menu/Help/About	New_Window	Yes
	Menu/Help/On_Window	New_Window	Yes
	Menu/Help/What's_New	New_Window	Yes
	Menu/Help/On_Version_2.6	New_Window	Yes
	Menu/Help/On_FAQ	New_Window	Yes
	Menu/Help/On_HTML	New_Window	Yes
	Menu/Help/On_URL	New_Window	Yes
	Menu/News/Post/Dialog	New_Window	Yes
	Menu/News/Follow_Up/Dialog	New_Window	Yes
	Menu/Navigate/Window_History/Dialog/Mail/Dialog	New_Window	Yes
	Menu/Navigate/Window_History/Dialog	New_Window	Yes
	Menu/Navigate/Hotlist/Dialog/Edit_Or_Insert/Dialog	New_Window	Yes
	Menu/Navigate/Hotlist/Dialog/Mail_To/Dialog	New_Window	Yes
	Menu/Navigate/Hotlist/Dialog	New_Window	Yes

	Menu/Help/Mail_Tech_Support/Dialog	New_Window	Yes
Home_Document	Menu/Navigate/Home_Document	Same_Window	Yes
Hotlist_Add	Menu/Navigate/Add_Current	Same_Window	No
	Menu/Navigate/Hotlist/Dialog	Same_Window	No
Hotlist_Delete	Menu/Navigate/Hotlist/Dialog	Same_Window	No
Hotlist_Edit	Menu/Navigate/Hotlist/Dialog	Same_Window	No
Hotlist_Insert	Menu/Navigate/Hotlist/Dialog	Same_Window	No
Hotlist_Load	Menu/Navigate/Hotlist/Dialog	Same_Window	No
Hotlist_Save	Menu/Navigate/Hotlist/Dialog	Same_Window	No
Interrupt	Mouse/Button/Interrupt	Same_Window	No
Mail_To	Menu/Navigate/Window_History/Dialog	Same_Window	No
Mosaic_Comment	Menu/Help/Comment_Card	New_Window	Yes
New_Window	Mouse/Button/New_Window	New_Window	Yes
	Menu/File/New_Window	New_Window	Yes
	Keyboard/File/New_Window	New_Window	Yes
News_Index	Menu/News/Index	Same_Window	No
News_List_Groups	Menu/News/List_Groups	Same_Window	No
News_Next	Menu/News/Next	Same_Window	No
	Keyboard/News/Next	Same_Window	No
News_Next_Thread	Menu/News/Next_Thread	Same_Window	No
	Keyboard/News/Next_Thread	Same_Window	No
News_Prev	Menu/News/Prev	Same_Window	No
	Keyboard/News/Prev	Same_Window	No
News_Prev_Thread	Menu/News/Prev_Thread	Same_Window	No
	Keyboard/News/Prev_Thread	Same_Window	No
Open_Local	Menu/File/Open_Local	Same_Window	Yes
Open_URL	CCI/Get	Same_Window	Yes
	CCI/Get	New_Window	Yes
	Menu/File/Open_URL	Same_Window	Yes
	Menu/Navigate/Internet_Starting_Points	Same_Window	Yes
	Menu/Navigate/Internet_Resources_Meta-Index	Same_Window	Yes
	Menu/Documents	Same_Window	Yes
	Keyboard/URL_Field/Edit	Same_Window	Yes
	Mouse/Document/Anchor	Same_Window	Yes
	Mouse/Document/Anchor	New_Window	Yes
	Remote_Control/Go_To	Same_Window	Yes
	Remote_Control/New_Win	New_Window	Yes
	Menu/Navigate/Window_History/Dialog	Same_Window	Yes
	Menu/Navigate/Hotlist/Dialog	Same_Window	Yes
Reload_Current	Menu/File/Reload_Current	Same_Window	Yes

	Keyboard/File/Reload_Current	Same_Window	Yes
StartUp_Document	Automatic/Navigate/StartUp_Document	Same_Window	Yes
Submit_Form	Mouse/Document/Form/Post	Same_Window	Yes
	Mouse/Document/Form/Get	Same_Window	Yes
Telnet_Window	Keyboard/URL_Field/Edit	Same_Window	Yes
	Mouse/Document/Anchor	Same_Window	Yes

Appendix B: Statistics

B.1a Event frequency per subject (all events)

B.1b Event frequency per subject (navigation events)

B.1c Event frequency per subject (Open URL events)

B.1d Event frequency per subject (navigation vs. non-navigation events)

B.2 Protocol frequency per subject

B.3 Probability of a recurrence over distance per subject

B.4 Cumulative probability of a recurrence over distance per subject

B.5 Recurrence rate, errors, internal links and domains per subject

B.6 Locality sets per subject

B.7 Longest repeated subsequences per subject