Interactive Visualizations to Facilitate Human-Machine Collaboration in Web Browsing

Denis Lalanne Paul P. Maglio
IBM Almaden Research Center
650 Harry Rd, NWED-B2
San Jose, CA 95120 USA
Denis.Lalanne@epfl.ch pmaglio@almaden.ibm.com

ABSTRACT

Browsing and searching are two common techniques for finding information on the World Wide Web (WWW). Though some searching is reasonably well supported by search engines, and many sites provide maps to their contents, browsing has minimal support. In this paper, we describe CoBrowse, a computational assistant intended to collaborate with a user to provide WWW browsing and searching support. It does this by inferring a set of keywords that represent the user's current interests based on the content and sequence of pages visited. With the current interests as a guide, CoBrowse suggests what pages to visit next, displays maps of the WWW neighborhoods around the user's current page, and displays the user's history in terms of the user's previous tasks. In addition to the design and implementation of CoBrowse, we report the results of a preliminary user study that suggests that the support CoBrowse provides can in fact help users find information on the web.

Keywords

Web browsing support, human-machine collaboration, attentive systems, user modeling, and web visualization.

INTRODUCTION

As the World Wide Web (WWW) increases in size, complexity, and information content, it becomes increasingly difficult for users to find information and increasingly easy for users to be overloaded with information. Because of the web's exponential growth, tools that help users browse and search this vast information space are now essential. Web browser applications, such as Netscape and Internet Explorer, generally keep track of the pages a user has visited, but do not provide assistance or advice on what pages to visit next or on what the current page's context is. Search engines suggest pages based on a specific query, but only when a user knows more or less what he or she is looking for and only when the user has decided to query them. Site maps can also be used to support information-seeking, as they display the structure (often hierarchical structure) of a document collection. Other approaches to supporting information-seeking on the web include information visualization [6,9], user modeling for adapting what is displayed to the user [2,13,18], annotations to facilitate navigation [5,7], tools for interpersonal awareness and interaction [1,14,16,17], and tools for information management and sharing [12].

Though the distinction can be problematic, we view the process of searching as differing from the process of browsing in that searching is deliberate and goal-directed, whereas browsing continuos and aims at covering a larger information space (see also [19]). Searching is used for answering specific questions. Browsing is used for learning about a topic or set of topics. Few tools support a natural interleaving of browsing and searching, for instance by enabling the user to search in the neighborhood of a specific document that has been found by browsing (but see [14]). The problem of finding target information when starting from a particular document is in general difficult, as there may be no good *residue* of the target information that points the way [8]. Indeed, even if the desired information is very close to a particular page (e.g., two or three

links away), the proper path is not always obvious.

Our main goal was to create a system that assists web browsing by pointing users in the direction of the information they desire. Our approach was to try to infer the user's intentions from the user's behavior, including search terms typed and pages visited. Of course, there are alternative approachs, such as relying on semantic information about content visited [23], structural information about how pages in a region are connected [11], or even social information about what others have done [10,17]. Our hope was to create a kind of *attentive system* that collaborates with the user during browsing, suggesting where to go by reflecting on information about the user's intentions and location. Attentive systems observe the user unobtrusively, trying to anticipate needs so they can propose relevant and helpful information [15].

CoBrowse is our implementation of an attentive web assistant. It creates a simple, keyword-based model of the user from queries the user has typed into search engines as well as words on pages the user has visited. It presents a graphical visualization depicting how well links that follow from the current page relate to the keywords that describe the user. The keywords naturally change as the user moves from page to page, reflecting how the user's interests change. In addition, the user can manually change the keywords, effectively searching the neighborhood around the current page. In this way, CoBrowse blends support for browsing and searching at the same time, providing an interactive framework for human-machine collaboration.

In this paper, we present our system CoBrowse in some detail. First we describe generally how CoBrowse works and what CoBrowse displays. Second, we outline its implementation. Third, we report results of a preliminary user study aimed at determining whether CoBrowse is an effective web browsing assistant.

COBROWSE

CoBrowse is intended to improve a user's WWW browsing results by enabling the user to collaborate with the system in exploring a topic of interest. In particular, CoBrowse aims to increase the space browsed by the user, and to enhance the user's perception of this information space by presenting visual summaries of the regions browsed. Moreover, CoBrowse attempts to augment the user's memory by keeping track of previously performed tasks along with task-related interests.

CoBrowse was inspired by Lieberman's Letizia [13], a computer agent that operates in tandem with a conventional Web browser. It tracks the user's browsing behavior and anticipates links that might be of interest to the user. Letizia provides recommendations upon request, usually in the form of a list of links. CoBrowse differs in that it continuously informs the user of its recommendations through dynamic and interactive visualizations that can provide hundreds of links displayed along more than one dimension for easy comparison. These interactive visualizations not only allow the user to watch the machine's ongoing work and suggestions, but also allow the user to intervene. Interaction is the key: machine and user do not simply work alone, watching each other's output. The interactive visualization serves as a common workspace in which the user and CoBrowse can collaborate.

User interests

CoBrowse models a user's interests as a set of words culled from the web pages a user visits and web searches a user performs. We have found in practice that simple word frequency (minus stop words) does a reasonable job of capturing the main points of a web page, though slightly more elaborate approaches such as TFIDF (as in [13]) often give a small improvement. CoBrowse uses a simple thresholding scheme based on word frequency to find the five or ten most important

words in the pages recently visited by the user. In this way, the user model is not necessarily continuous, developing smoothly over time. As a user moves from page to page, the set of words that describes his or her current interests can change both slowly and abruptly.

The hope is that current interests represent the task actually performed by a web user. For example, if a user is looking for apartments in the bay area, the list of keywords that describe the user ought to include "Bay Area, apartment, rental, Palo Alto, Cupertino," and so on. CoBrowse assembles a list of keywords by taking into account both the user's interactions with the browser and also with CoBrowse itself. It uses the most frequent words on the current web page, the words in the page's title, the words in the anchor text of the link followed, as well as these sorts of words derived from the previously visited pages. As long as there is some commonality between the accumulated keywords and the words derived from the current page, CoBrowse merges the current keywords into the those accumulated so far. If there is little or nothing in common between the current words and the words derived from the previous pages, CoBrowse begins a new set of words, pushing the previously derived set onto the stack of history. This accumulation of historical sets of keywords (previous user interests) represents the kinds of things the user has done in the past.

Suggestions and the MAP visualization

The set of keywords representing the user's current interest provides input to CoBrowse's main heuristic for suggesting new pages to visit. The algorithm for discovering new pages is a mix of best-first and breadth-first search. The idea is to look around from the current page breadth-first but to prune branches that are too unrelated (i.e., do not match some threshold number of words contained in the user's current interest). At any point, the user can intervene by modifying the current interest keywords. Suggested URLs are found by searching forward from the user's current URL, and evaluating these based on how well they match the user's current interests. In this way, CoBrowse looks ahead for the user, searching the neighborhood around the current page (see also [13]).

A simple way of displaying suggested links to follow is to list the URLs sorted according to how well they match the user's current interests. The Multi-Attribute Pareto (MAP) [20] visualization is a more elaborate way to suggest URLs to the user. The MAP organizes the URLs in a two dimensional graph, providing information about relationships among words to help users evaluate which URL is best. Figure 1 shows a MAP in which the x-axis represents attributes of "word2" and the y-axis represents attributes of "word1". In this example, the points on the graph represent URLs whose content is related to these two words to varying degrees. The farther to the right a URL is represented, the more it relates to "word2", and the higher a URL is represented, the more it relates to "word1". Thus, URLs that are highly related to both words appear in the upper right of the graph.

Figure 1: Pareto visualization of suggested URLs.

Such MAP visualizations enable users to visually perform a tradeoff analysis to choose suggested pages. For example, given the graph in Figure 1, if a user wants to emphasize the keyword "word1" more than the keyword "word2", then a point with a high value on the y-axis and a low value on the x-axis ought to be selected. Though this sort of visualization is feasible for up to three criteria, many real-world problems call for tradeoff analysis in higher dimensions. CoBrowse's MAP visualization overcomes this limit by combining color patterns, visual structures, and interactivity.

Figure 2 shows two examples of the MAP visualization created for CoBrowse. Here, the y-axis represents the center of mass of the N criteria (i.e., the user's current interest keywords), and the x-axis represents the sum of all criteria values. The best overall suggested page (i.e., the one that most closely matches *all* the keywords) is the one with the largest x value. Whereas x represents an absolute performance value, y indicates the distribution of the underlying criteria. Pages displayed near the center of the y-axis are related to more of the keywords than those displayed toward the top or bottom. Pages displayed away from the center of the y-axis are related more narrowly to fewer of the keywords.

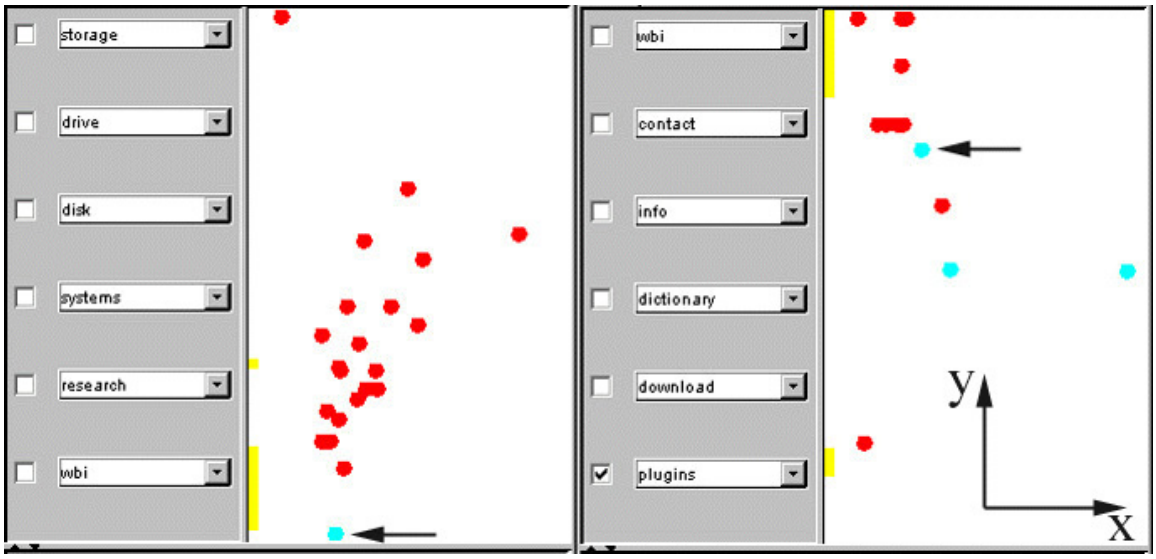

Figure 2: The interactive MAP visualization, multi criteria tradeoff of suggested URLs.

When the user clicks on a URL diplayed in the MAP visualization, the page is opened in the browser window. Because a new page has then been visited, CoBrowse follows the user, updates the keywords reflecting current interest and then looks around the new page to find new suggestions related to the updated interests. On the left part of Figure 2, for instance, the user is visiting a page on research at IBM Almaden. When the user moves the mouse pointer over a suggested page on the MAP display, the dot is highlighted in blue and the keywords related to it specifically are highlighted by a yellow bar drawn against the left side of the display. In this way, the user can interact with the visualization to find relationships among suggested pages without having to actually visit the pages. For example, the blue highlighted dot on the left part of Figure 2 refers mostly to WBI and a little to research, as can be inferred from the distributon of yellow (pages related to this one).

Users can interact with the MAP visualization in other ways too. For instance, the user can manually change one or more keywords from the set of automatically deduced keywords. Given a new set of words representing the user's interest, CoBrowse begins searching from the current page again, updating the visualization to reflect the change. The user can also select a subset of the current keywords by setting the check boxes next to the keywords. In the right part of Figure

2, for instance, the keyword "plugins" has been selected and the related pages are highlighted in blue. Moving the mouse pointer over one of these blue dots, a yellow bar is drawn next to the keywords for each contained in the related document. The longer the yellow bar, the more important the keyword next to it is for the page the mouse pointer is over.

Figure 3 illustrates an ideal scenario of MAP visualization use. In the MAP on the left, the user is looking for pages at New York University (NYU) about studying abroad. Only one URL is highlighted when the user selects the keywords "abroad" and "study". Moving the mouse pointer over this page highlights the keywords contained in the document (abroad, york, university and study). The window directly under the MAP visualization displays and highlights the URLs of the pages shown in the MAP. The window at the bottom displays detailed information about the currently highlighted page, such as title and keywords. Suppose the user clicks on this page, the URL is opened in the browser, and a new MAP is displayed (middle of Figure 3). The user then modifies the keywords by entering the word "Florence" in the third box down and selecting the keywords "Florence" and "abroad". A single page is then highlighted. If the user then moves the mouse pointer over this highlighted page, CoBrowse shows that the page is indeed related to "Florence" and "abroad" as well as "study". Finally, the user clicks on the page to open it and finds the answer to the question (right part of Figure 3).

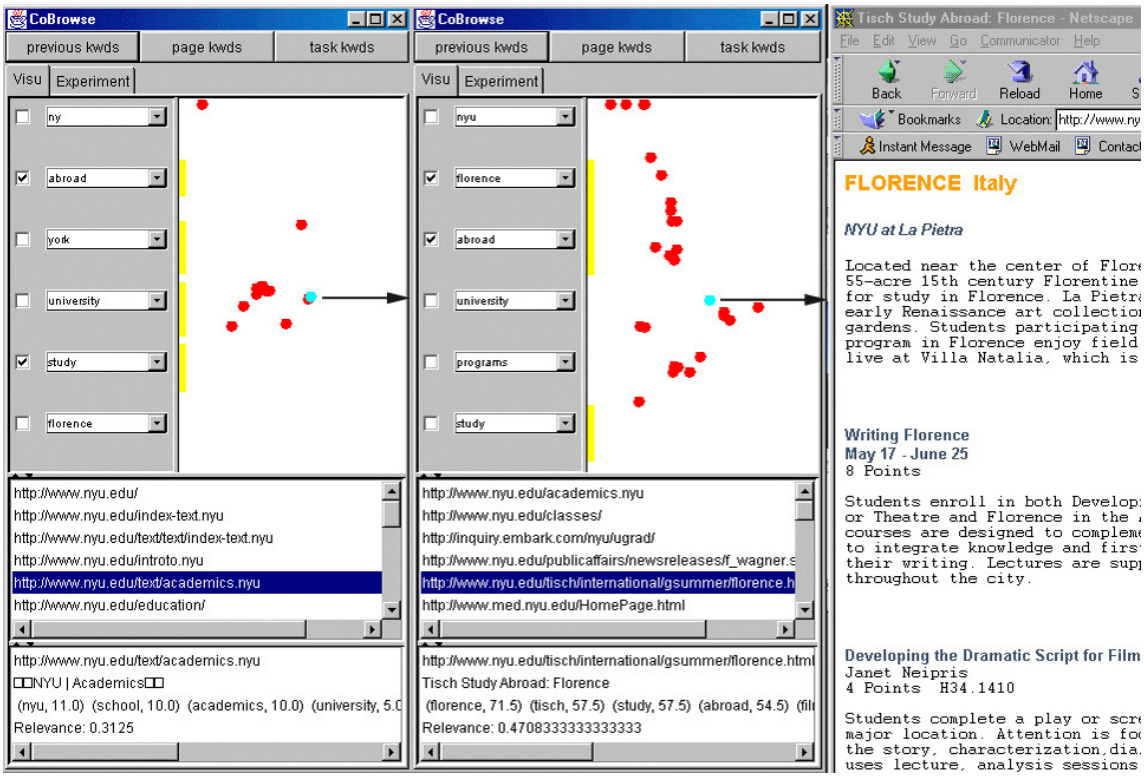

Figure 3: Starting from the page www.nyu.com, an ideal sequence using CoBrowse (in two clicks) to answer the question: "Who is the person to contact for studying film writing in Florence with NYU?"

Neighborhood Visualization

CoBrowse provides another simple visualization to help the user locate him- or herself in the information space of the web. CoBrowse relies on clustering techniques to build a map of the neighborhood around a page. Each page is represented by a small set of keywords (e.g., highest frequency words) that are used to hierarchically cluster pages that can be reached within a few links from the current page. This technique allows the use of a simple tree visualization. The closer a keyword is of the root of the tree, the more representative it is of the neighborhood. The first level of the tree represents the main clusters, which can then be further decomposed into

more discriminating cluster, and so on. The user can set the depth of the tree, as well as the algorithm's accuracy or threshold for creating clusters. Figure 4 shows the neighborhood visualization used in CoBrowse. The right part of the figure displays the continuously changing tree in terms of the keywords that descibe the clusters. In this case, we were starting at the IBM Almaden site. Clicking on one branch of the tree highlights the corresponding URLs in the graph display on the left.

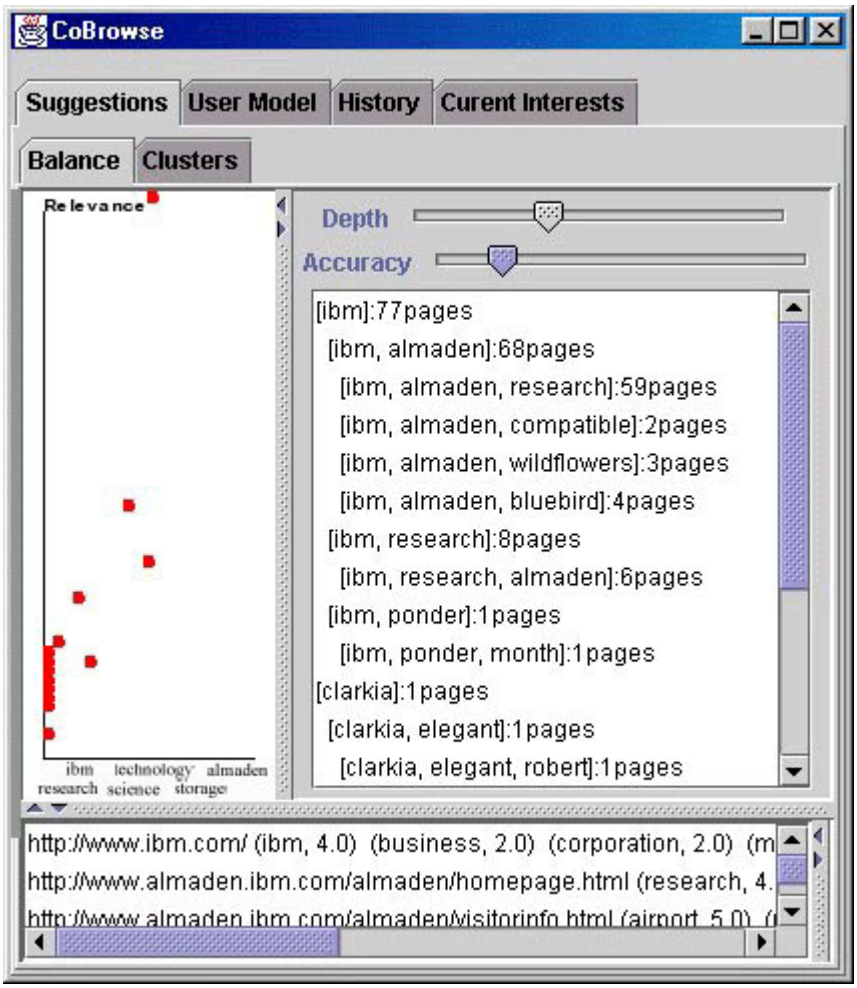

Figure 4: Visualizing a page's neighborhood with a simple tree.

Many systems use similar methods to assist browsing and selection of search results through visualization [6]. Many of these tools are more visually appealing than the simple one we have devised, see for instance, the links contained in Atlas of Cyberspaces, Visual Browsing in Web and Non-Web Databases and OLIVE - On-line Library of Information Visualization Environments. For instance, Cartia is a good example of a neighborhood visualization technique. Cartia displays an interactive landscape of information (a topographical map) that shows the user what is inside documents and web pages. Such maps are effective at showing points of interest and the informational distance between them. The greater the similarity between two documents, the closer together they appear on Cartia's map. Peaks appear on the map where there is a concentration of several documents about the same topic, and the distance between peaks shows how closely the topics are related. With such a visualization, its is easy determine what topics are covered in thousands of documents, how much emphasis is placed on a given topic, and how different topics relate to one another.

All these sorts of visualizations, however, are tailored to answering specific user queries, meaning the user must stop browsing and decide exactly what information is desired. Again, the

CoBrowse's collaborative approach is unique: No explicit user queries are required. CoBrowse follows user interests and takes advantage of the user's idle time (reading a page for example) to explore and display where the user is, what is around, and where it it makes sense to go.

IMPLEMENTATION

CoBrowse was built using the Web Intermediaries (WBI) Development Kit, a framework for adding intermediary functions to the WWW [3,4,5]. *Intermediaries* are computational elements that lie along the path of web transactions. They may operate on web data as the request is sent from the browser, passes through firewalls and proxies, is satisfied by the generation of a document, and as the document is returned to the browser. Intermediaries have access to web data at all these points, and are able to observe, respond to requests, and modify both the request and the resulting documents. WBI is essentially a programmable proxy that was designed for easy development and deployment of intermediary applications.

In WBI, intermediary applications are constructed from four basic building blocks: request editors, generators, document editors, and monitors. We refer to these collectively as MEGs, for Monitors, Editors, Generators. Monitors observe transactions without affecting them. Editors modify outgoing requests or incoming documents. Generators produce documents in response to requests. WBI dynamically constructs a data path through the various MEGs for each transaction. To configure the route for a particular request, WBI has a rule associated with each MEG that specifies a boolean condition indicating whether the MEG should be involved in a transaction. An application is usually composed of a number of MEGs that operate in concert to produce a new function.

The first application constructed with WBI was *personal history*, which uses a monitor that records the sequence of pages visited by each user along with the text of each page [5]. The user can access his or her own personal history through generators that search through the stored text or that display paths taken previously. In addition, the markup of individual pages can be changed to reflect patterns of repeated use, for instance, adding shortcut links to pages that are visited repeatedly within some radius of the current page. WBI has also been used (a) to cluster a user's history of web usage around key nodes found on the way to some goal [18], and (b) to enable web users to interact in real time [16,15].

CoBrowse is implemented as WBI plugin, that is, as a set of MEGs. In particular, CoBrowse monitors the user's HTTP traffic, collecting keywords, augmenting the user model, and updating the visualizations to support browsing.

EVALUATION

To determine whether the particular user modeling and visualization techniques we implemented in CoBrowse actually facilitate web browsing, we conducted a preliminary user study on some of the system's functions. As mentioned, our rough distinction between browsing and searching mirrors the distinction between exploration and goal-directed activity. If browsing is exploration, then one episode of browsing might be more successful than another if the user covers more of the space or learns more about a topic. Though CoBrowse is intended to aid browsing, it is very difficult to set up a user test that could show a cost or benefit of a tool specifically for browsing. Thus, we chose to test how well CoBrowse facilitates searching for information in particular information neighborhoods. We compared the performance of users browsing with CoBrowse and without CoBrowse to find answers to several specific questions.

There are at least three general types of questions that can be answered by searching the web (see also [18]). First, there are questions for which there is a single answer and only a single page or

document on the web that contains the answer. For instance, a single site, single answer (SSSA) question might be "Does the University of Western Ontario offer a master's degree in psychology?", as there is a definite "yes" or "no" answer which is likely to be found only at the University of Western Ontario site. Second, there are questions for which there is only a single answer but this might be found on any of several sites. For instance, a multiple site, single answer (MSSA) question might be "How many U.S. states was Ralph Nader on the presidential ballot in 1996?", as there is a single number that is the correct answer, but this number might be found at several sites, such as the U.S. Federal Election Commission or the Green party, and so on. Third, there are questions for which there might be several answers and these might be found at several different sites. For instance, a multiple site, multiple answer (MSMA) question might be "Name three drugs currently being tested to treat Alzheimer's patients", as there are likely many such drug trials and it might require visiting many sites to find three distinct drugs.

To constrain our study, we considered only SSSA questions. Furthermore, we barred the use of search engines, preferring instead to start users at a specific page for each question and then see how difficult it was to find the answer by browsing. In an attempt to control for browsing difficulty, we relied on Furnas's notion of *residue* [8], which we operationally defined as the indication that target information will be available on page by the anchor text of the link that leads to it. More precisely, we defined residue as the fraction of words (minus stop words) the question shares in common with a link's anchor text and text one sentence around the anchor text. For instance, suppose page A contains the answer to the question "Does the University of Western Ontario offer a master's degree in psychology?". If page B has a link to page A with the anchor text "Psychology Degree Programs", the residue of that link for the question at hand would be 2 out of 8 (or .25). We constructed a set of questions with high (>.25), medium (>.1) and low (<.1) residue from starting pages five links away. Our hypothesis was that CoBrowse should help especially in cases where the residue is low, as its look ahead ought to provide useful information to the user.

Methods

The test was set up as a two factor, within-subjects design. One factor varied whether a web browser (Internet Explorer) and CoBrowse MAP visualization was available or whether the browser alone was available (Control condition). The second factor varied the residue of the answers from the designated starting page set five links away (low, medium, and high). Thus, this was a 2x3 design in which each participant saw each of six questions.

An upper bound of ten minutes was set for answering each question. We tracked the browsing behavior of the participants, including the URLs visited, the time, where it has been opened from (browser, or visualization), and whether the question was answered successfully.

Participants

Four experienced computer users participated in this preliminary study. They were compensated for their time.

Questions

1. Starting from http://members.aol.com/DKoppen/spaceodd4.html, how did the hostess manage to turn upside down without falling, in "2001: A Space Odyssey"? [Answer: Velcro; Residue: >.25]

2. Starting from http://www.castles.org/Chatelaine/index.html, find the name of a clan whose territory was based on the isle of Skye. [Answer: Clan MacNicol; Residue: >.25]

3. Starting from http://www.nyu.edu, who is the person to contact for studying film writing in Florence with NYU? [Answer: Janet Neipris; Residue: >.1].

4. Starting from http://www.library.mcgill.ca/, find the césars movie awards page. [Answer: http://www.ecran-noir.com/evenements/prix/cesar/; Residue: >.1].

5. Starting from http://hci96.open.ac.uk/~hci96/, find who is responsible for the masters degree in HCI at the Queen Mary and Westfield College University of London. [Answer: Peter Johnson; Residue: <.1].

6. Starting from http://sfsite.com/home.htm, "Do Androids Dream of Electric Sheep?" by Philip K. Dick was the basis for the movie Blade Runner. He lived most of his life in California. What was his last address in Berkeley? [Answer: 1126 Francisco Street; Residue <.1]

The questions and their order of presentation was balanced across the four participants, as shown in Table 1.

Participant 1 1) CoBrowse 1,3,5 2) Control 2,4,6	**Participant 3** 1) CoBrowse 2,4,6 2) Control 1,3,5
Participant 2 1) Control 1,3,5 2) CoBrowse 2,4,6	**Participant 4** 1) Control 2,4,6 2) CoBrowse 1,3,5

Table 1: Questions and order were balanced across conditions.

Results and Discussion

We coded the number of answers as the highest residue page the participant visited for each question. In this case, a score of 1 corresponds to finding the target, whereas a number between 0 and 1 represents the closest page to the target that was visited. The total number of answers found per participant (using this scoring scheme) is shown on the left side of Figure 6, and the total number of pages visited per participant is shown on the right side of Figure 6. As shown, overall about the same number of answers were found in both browser conditions, but participant visited far fewer pages when using CoBrowse.

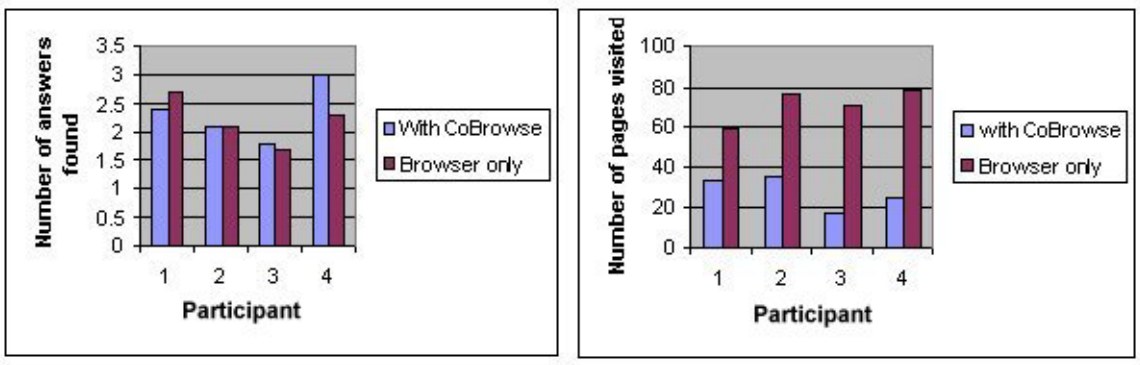

Figure 6: Users visited fewer pages with CoBrowse than they do without CoBrowse. The number of answers found is similar in the two conditions.

With only four participants in this preliminary study (one in each of the cells of Table 1), our conclusions must be considered tentative. Nevertheless, we have a few observations. First, in the control condition (browser only), when the tasks were difficult (i.e., low residue), many participants were unable to find the answers. In this case, they followed most links on the pages and did much more backtracking than when they used CoBrowse. Sometimes participants got stuck on an incorrect path, unable even to return to the starting page. When participants performed the same tasks with CoBrowse, they trusted the tool to help them get out of dead-ends. Some used CoBrowse as a personal search tool, typing keywords for the tool to find in the pages nearby. Others just looked at the optimal links on the visualization and opened them. However, participants used the visualization more easily when they specified the keywords, rather than relying on the automatic user model.

A second observation is that CoBrowse had no significant influence on finding the answers when the residue was high. In this case, participants seemed to simply ignore the suggested pages. When the residue was low, participants were more interested in using the tool. In some cases, participants could use CoBrowse to find the answers hat they were unable to find otherwise. And most of the time, CoBrowse helped participants to get closer to the answer. It seems that CoBrowse helped them focus and reflect more on their task. Indeed, the results show that in the control condition, when the task's residue was low, participants followed more links and backtracked more, but did not get closer to the answers. With CoBrowse, participants spent more time on pages and were more focused on getting to the answer.

CONCLUSION

We have described CoBrowse, an automated web browsing assistant that attempts to support the user *during* browsing through interactive visualizations and human-machine collaboration. We focused mainly on the MAP visualization that was used to suggest pages to a user. And we reported results from a pilot study that suggests that CoBrowse can improve web browsing performance.

ACKNOWLEDGMENTS

We thank Chris Campbell for helping to set up the user study, and Teenie Matlock for editorial comments on an early draft. The Swiss National Science Foundation partially supported this research.

REFERENCES

1. Aberg, J. & Shahmehri, N. (1999). Web Assistants: Towards an intelligent and personal web shop. In P. Brusilovsky, P. De Bra & A. Kobsa (Chairs), *Proceedings of the Second Workshop on Adaptive Systems and User Modeling on the WWW*.

2. Ardissono, L., Console, L. & Torre, I. (1999). Exploiting user models for personalizing news presentations. In P. Brusilovsky, P. De Bra & A. Kobsa (Chairs), *Proceedings of the Second Workshop on Adaptive Systems and User Modeling on the WWW*.

3. Barrett, R. & Maglio, P. P. (1999). Intermediaries: An approach to manipulating information streams, *IBM Systems Journal, 38*, 629-641.

4. Barrett, R. & Maglio, P. P. (1998). Intermediaries: New places for producing and manipulating web content . In *Proceedings of the Seventh International World Wide Web*

Conference (WWW7), Brisbane, Australia.

5. Barrett, R., Maglio, P. P. & Kellem, D. C. (1997). How to personalize the web. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI '97).* New York, NY: ACM Press.

6. Card, S.K. (1996). Visualizing retrieved information: a survey. *IEEE Computer Graphics and Application* 16, 63-67.

7. Campbell, C. S. & Maglio, P. P. (1999). Facilitating navigation in information spaces: Road signs on the World Wide Web. *International Journal of Human-Computer Studies, 50,* 309-327.

8. Furnas, G. W. (1997), Effective view navigation, in *Proceedings of Human Factors in Computing Systems, CHI'97*, pp. 367-374. New York: ACM Press.

9. Gershon, N. (1996). Moving happily through the world wide web, *IEEE Computer Graphics and Applications, 16*, 72--75.

10. Gruen D. & Moody, P. (1998). Synchronous WebPath as a tool for leveraging expertise. Paper presented at *Human Computer Interaction Consortium Conference*, Snow Mountain Ranch, CO.

11. Kumar, R., Raghavan, P. Rajagopalan, S. & Tomkins, A. (1999). Trawling the web for emerging communities. In *Proceedings of the Eighth International World Wide Web Conference*, Toronto, Canada.

12. Li, W. S., Vu, Q., Agrawal, D. Hara, Y., & Takano, H. (1999). PowerBookmarks: A system for personalizable web information organization, sharing, and management. In *Proceedings of the Eighth International World Wide Web Conference*, Toronto, Canada.

13. Lieberman, H. (1995), Letizia: An agent that assists web browsing. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI'95*, pp. 924-929, AAAI Press.

14. Maarek Y.S. & Shaul I. Z. B. (1997). WebCutter: A system for dynamic and tailorable site mapping. In *Proceedings of the Sixth International World Wide Web Conference WWW6*, Santa Clara, California, USA.

15. Maglio, P. P., Barrett, R., Campbell, C. S., & Selker, T. (in press). SUITOR: An attentive information system. To appear in *Proceedings of the International Conference on Intelligent User Interfaces 2000*.

16. Maglio, P. P. & Barrett, R. (1999). WebPlaces: Adding people to the web. In *Poster Proceedings of the Eighth International World Wide Web Conference*, Toronto, Canada.

17. Maglio, P. P. & Barrett, R. (1998). Adaptive communities and web places. In P. Brusilovsky & P. DeBra (Chairs), *Second Workshop on Adaptive Hypertext and Hypermedia.* Pittsburgh, PA.

18. Maglio, P. P. & Barrett, R. (1997). "How to build modeling agents to support web searchers. In Proceedings of the Sixth International Conference on User Modeling, Sardinia, Italy.

19. Marchionini, G. (1995). *Information seeking in electronic environments*. New York:

Cambridge University Press.

20. Pareto, V. Cours d'économie politique, Technical report, Rouge, Lausanne, Switzerland, 1896.

21. Pu, P. & Lalanne, D., (1999). Interactive problem solving via algorithm visualization. To be submitted to *VisSym '00, Joint Eurographics - IEEE TCVG Symposium on Visualization*, Amsterdam, The Netherlands.

22. Selker, T. (1994). COACH: A teaching agent that learns. *Communications of the ACM, 37*.

23. Stefani, A. & Strapparava, C. (1999). Exploiting NLP techniques to build user model for Web sites: the use of WordNet in SiteIF Project. In P. Brusilovsky, P. De Bra & A. Kobsa (Chairs), *Proceedings of the Second Workshop on Adaptive Systems and User Modeling on the WWW*.