1 Introduction

There is a great deal of research and practical tools for accessing information. Web search engines are becoming a common tool used by the general population. Many data-driven applications like email clients provide native search tools to locate information. However, a disproportionate amount of effort has been placed on users’ initial discovery of information, as opposed to re-accessing information they discovered in the past. It is estimated that between 58% and 81% of web hits are not the first time the user has seen the page [7, 8]. In one study [3], 17% of web users report not being able to return to a previously visited page as one of their biggest problems using the web. Clearly the problem of re-accessing information is one that deserves more study.

In this paper we develop methods for better re-access of information and implement them in a tool that searches the history of a user’s web documents. Section 2 describes the problem in more detail. Section 3 surveys other related work. Section 4 discusses how users’ behavior during their initial information discovery is used for re-accessing the information later. In section 5, the implementation of the tool is detailed. Section 6 presents examples of how the tool performs. Finally, sections 7 and 8 discuss how the tool can be improved in future work and offers some final remarks.

2 Motivation

This project was initially motivated by a common problem: A user browses to a page and several months later would like to find the same page, but cannot remember how to get to the page. Perhaps the user tries unsuccessfully to use a search engine to re-find the page. The user needs a better way to search her history. Currently, history searches bring up all pages the user has visited that contain the query, regardless of relevance. This project attempts to calculate the relevance to the user of every page the user visits. This score is approximated using heuristics modeled after user browsing behavior. These scores are then used to boost relevant documents when the user searches her history, in order to re-access information.
3 Related Work

Much research has been done on information access, but most either assumes the user is making their first discovery of the information, or ignores the specific case of users re-accessing information. However, there have been some studies conducted and prototype tools created for information re-access.

Keeping Found Thing Found\(^1\) is a project that has conducted several studies in this area. In [4], users are studied in their place of work to see how they record information, or pointers to information, for later access. It was found that the methods used varied widely depending on the user and the type of job they performed. How people retrace their steps to previously found information when they have not purposefully stored it was studied in [1]. A study specific to re-finding information on the web is done in [9].

In addition to user studies, systems have been implemented that support information re-access. The RE:Search Engine [10] records users’ previous searches and their selected results and integrates the previous results with new searches, so that previously selected pages and new pages are ranked higher. Stuff I’ve Seen [2] is a tool used to locally search all of a user’s information, including emails, webpages, and other documents. Google\(^2\) has also created tools related to information re-access, including personalized search and desktop search.

4 Behavioral Scoring

Two heuristics are used to judge the relevance of a document. These heuristics are the length of time a page is viewed and how deep a link is. Depth of a link can be defined as the number of clicks away from a search engine’s results page or a URL entered by the user. The motivation behind these two heuristics is based on simple observations of browsing behavior. The longer a user looks at a page, the more likely it is to be relevant. In fact, if a user looks at a page for a very brief amount of time, the page is likely to be irrelevant and should be scored lower. The second observation is that search engines often bring up a general page or home page for a topic, the user then browses from this page, narrowing down on the specific information she seeks.

Each heuristic is scored as a probability. The scoring functions are rather naive at the moment because they need to be tuned by actual user studies. The length of time a page is viewed is divided by 30 minutes to calculate its probability score. If the page is viewed for more than 30 minutes, the probability is set to one. The cap of 30 minutes is chosen somewhat arbitrarily, but a cap of some sort needs to exist. This is to prevent a website that a user leaves open on their desktop overnight from being boosted excessively.

The link depth is scored in a similar manner to the length viewed score. The link depth starts at 1 for pages entered manually into URL bar and 3 for search pages. All links from this start page will have a depth of \(start\_depth + 1\). Each linked page’s links will have a depth of \(start\_depth + 2\) and so on. We compute the probability score for link depth by dividing the current page’s depth by 8. This cap of 8 and the starting

---

\(^1\)http://kff.ischool.washington.edu/
\(^2\)http://www.google.com
page initial depths are again chosen arbitrarily; proper user studies are needed to tune the constants.

The final behavior score is computed by combing the probability score given by each heuristic. The probability scores are treated as independent and simply multiplied together to compute the overall probability. This independence assumption is not justified, as relevant pages are likely to boost all of the heuristic scores in a similar manner, but this simplified model has worked well thus far. A method of weighting heuristics has also been provided. Each probability score has a constant power that it is raised to that can be adjusted. This allows more accurate or more important heuristics to be emphasized in the final behavior score.

\[ \text{behaviorScore} = \text{Prob}(\text{Depth})^\alpha \ast \text{Prob}(\text{Time})^\beta \]  

(1)

5 Implementation

The implementation consists of three components: the indexer, the firefox extension, and the search interface. The indexer takes webpages’ and their metadata and stores them in an index. The Firefox extension monitors user behavior, creates a behavior score based on the user’s browsing behavior, and sends all the webpages viewed along with their behavior scores to the indexer for indexing. The search interface queries the index, retrieves document metadata and content-based score, re-scores the results using the behavior score metadata, and displays the results.

5.1 Indexing

The input to the indexing component is a single file consisting of a website’s HTML and its metadata. There are three metadata fields: url, the URL of the page, docid, a unique number to identify the document, and behaviorScore, the behavior-based relevance probability. Upon receiving a new website, the indexer adds the individual website to the index immediately.

*Indri* [6] is used to do the indexing. It supports single document additions, parses HTML documents, and supports additional metadata fields. Metadata fields are stored in a forward index so that the behavior score and URL can be retrieved, given a document ID, during searching.

5.2 Firefox Extension

The user’s browsing behavior is tracked by a Firefox extension. The extension is implemented in *JavaScript* and *XUL*. The extension is completely silent, no direct user interaction is necessary to track the behavior besides the initial installation of the extension. The flow of the extension for each web page visited is as follows. First, the extension makes note of the current page being viewed and then keeps track of whichever heuristics are specified. After the user browses away from the site, a score is computed, the document and score are written to disk, and a shell script is called to add the document to the index.
The code that calculates the behavior score is quite extensible and supports the addition of new heuristics through a few simple changes. Each heuristic implements a scoring function that is called between page transitions. These functions compute the probability score for the old page before we load the new page. Then, these probabilities are combined to compute the behavior score and the document and score are added to the index. This combination is performed in the logarithmic domain to preserve precision.

\[ \text{behaviorScore} = \text{Prob}(\text{Depth})^\alpha \ast \text{Prob}(\text{Time})^\beta \]  

\[ \log(\text{behaviorScore}) = \alpha \ast \log(\text{Prob}(\text{Depth})) + \beta \ast \log(\text{Prob}(\text{Time})) \]  

### 5.3 Search Interface

The search interface is a PHP-based website that can be run on the user’s local machine. Apache was used for the web server. Indri [6] comes with a PHP library, allowing direct access to the index.

After the user inputs a query, the top 10 documents are retrieved from the index using content-based scoring. The content-based score is done by Indri’s scoring methods, but any scoring system can be used. Once these documents are loaded, their URL and behavior score are retrieved from the metadata index. The documents are then rescored and re-ranked by combining the content-based score and the behavior score. Since Indri’s scoring is based on a log probability, the final reweighted score is simply the sum of the content-based score and the behavioral score (which is stored as a log probability in the index metadata). The content and behavioral scores can be given different weights, though in our case we weighted them equally, which we found to be sufficient. The results are then displayed in the users browser. Each results shows the document’s score, URL, title, and a snippet of relevant text where query terms were found.

The interface allows users to choose between behavioral and content-based scoring, where the behavior score is not used to re-rank the results. An example set of results from the interface is shown in figure 8.

### 6 Results

The results of this project are limited at present. While promising results have been obtained for some sample scenarios, full user studies are needed. These studies would provide more observations of common browsing behavior, which would be used to fine tune the heuristics. The observations would also likely suggest new heuristics to measure.

Sample Scenario: Finding the class home page. This scenario shows how browsing based behavior can help determine relevance between almost identical pages. A Google search for the class home page (Figure 8) using the query: “cs498 zhai” returns the current home page at the bottom, several old home pages that look similar and several pages that look promising on the results page but which contain no useful information. A typical user might click on the higher results first, especially if the look like what the user wants. After clicking on one of these links, it is obvious that the page is not the
class home page. The user might also click on the old course home page, too, because
the layout and content is almost identical to this year’s course home page. Again,
though a quick scan and the user will realize that it is the wrong home page. Both of
these types of errors would have low behavior scores because they user looked at them
only for a short while. When the user finally clicks on the real home page at the bottom
of the results page, the behavior score for the home page will be much higher because
the user will spend time examining the syllabus or other content on the home page,
boosting its behavior score. An example session was performed and the results were
as expected. The current class home page comes near the bottom of the results page of
a simple search of the user history (normal scoring, Figure 8) using a standard scoring
function. This same page comes first using behavior based relevance scoring (Figure
8). The more relevant history pages are boosted to the top of the search results, aiding
the user in re-accessing found websites.

7 Future Work

User studies need to be done to fully evaluate the quality of the system. In addition,
monitoring actual user behavior is needed to adjust the parameters. It may be possible
to discover good weights of the different behavior features automatically after record-
ing the users’ browsing behaviors and the pages they re-access. In addition, there may
be many more relevant features about users’ browsing behavior that should be studied
and incorporated into the system. It is not clear whether there is a principled way to
collect and weight relevant browsing features.

Expanding the system beyond just results from users’ page history may lead to bet-
ter access of new information. There are two ways this could be approached. First, the
history results could be incorporated into the results of a general web search. Second,
highly ranked history results could be used as feedback documents for general web
searches.

8 Conclusion

Information re-access is a relatively little-studied research area. In this paper, we cre-
ated a system to improve searching previously accessed web documents based on the
user’s behavior when they originally viewed the page. The method for using the behav-
ioral features in combination with the original scoring was discussed. The implemen-
tation of the system was detailed, and samples of results were presented. The results
show that using behavioral features to score documents during ranking can greatly im-
prove the relevance of highly ranked documents.

References

exploratory study of how users re-find information, Technical Report, Virginia
Tech.


Figure 1: Google Query: cs498
Results for cs498

[query 0.03s, documents 0.01s, total 0.01s, scoring normal]

-4.55 cs498 zhai - Google Search

cs498 zhai - Google Search Sign in Web Images Web Results 1 - 10 of about 98 for cs498 zhai. (0.56 seconds) CS498 / Zhai, CCS498 / Zhai, C. Special Topics in CS. [f...]
[Cached] http://www.google.com/search?q=cs498+zhai&ie=utf-8&oe=utf-8&aqsm=1&aq=0&rls=org.mozilla:en-US:official&bav=on,9, false-1,0.0,asc-x,ved-0ahUKEwj-wLjFm7OsAhXm8HsKHTT1AaQkQ8wIEw&biw=1530&bih=689

-4.99 Course Announcement for CS498-CXZ (Fall 2005) Please Read!!


-5.05 http://coursefire.com/studygroup.php?
courseID = CS498&profName = Zhai,%20C&rs = &PHPSESSID = 4f6da00154c9284ac7a232ac3732c2b6

... Karabulut, K / Nadeau, P / Fitt, L / Todd, M / Woodley, M / Zhai, C / All CS498 / Zhai, C. Special Topics in CS Please login on the left first before using the study groups page. 114722541781: http://coursefire.com/studygroup.php?courseID = CS498&profName = Zhai,%20C&rs = &PHPSESSID = 4f6da00154c9284ac7a232ac3732c2b6
courseID = CS498&profName = Zhai,%20C&rs = &PHPSESSID = 4f6da00154c9284ac7a232ac3732c2b6

-5.11 http://coursefire.com/view_tips.php?
courseID = CS498&profName = Zhai,%20C&PHPSESSID = 4f6da00154c9284ac7a232ac3732c2b6

...all return : window.open('complaint.php?complaint_id='&courseID=CS498&profID=+id='&perce'...: "","_self","... / Todd, M / Woodley, M / Zhai, C / All CS498 / Zhai, C. Special Topics in CS Please login on the left... 1147225411615 http://coursefire.com/view_tips.php?courseID = CS498&profName = Zhai,%20C&PHPSESSID = 4f6da00154c9284ac7a232ac3732c2b6

-5.11 http://silafka.cs.uic.edu/course/cse498/cx204/p1.html


-5.16 http://silafka.cs.uic.edu/course/cse498/cx206/p1.html


Figure 2: Normal Indri Query: cs498
Figure 3: Behavior-based results