The Re:Search Engine – Helping People Return to Information on the Web

Jaime Teevan
Massachusetts Institute of Technology, CSAIL
Cambridge, MA 02139 USA
teevan@csail.mit.edu

ABSTRACT
Re-finding information is commonly cited as a problem on the Web. One reason re-finding is hard is that while people rely on context to return to information (*e.g.*, by following the original path taken to it), the Web makes no guarantee that the context will remain static. The *Re:Search Engine* is designed to help people return to information in dynamic environments like the Web by maintaining consistency in the results it returns. For example, if Connie, while looking to purchase a Global Positioning System, found several she liked via a search for "GPS", she would expect to be able to use the same query to locate the same systems again. Returning the original result list for a re-issued query provides consistency, but omits new information, such as new GPS systems. An ideal result list contains both the systems Connie remembers having seen and high quality new systems. Because people remember little of what is presented in a search result list, when a person repeats a query, the Re:Search Engine can preserve what is remembered about the original result list while presenting new information.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces – Graphical user interfaces. H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval – Search process, Relevance feedback.

General terms: Design, Experimentation, Human Factors

Keywords: Re-finding, personalization, information management, implicit feedback, user profiling.

PEOPLE RELY ON CONSISTENCY TO RE-FIND…

Supporting the successful re-finding of information on the Web is important. In a study of Web users [3], 17% of those surveyed reported "Not being able to return to a page I once visited," as one of "the biggest problems in using the Web." People rely on consistency in their information environment to re-find [2]. Consider the example search shown in Figure 1. If Connie, while looking to purchase a Global Positioning System, found several systems she liked through a search for "GPS", she would expect to be able to use the same query to locate the exact same systems again.

The importance of consistency is emphasized by two studies conducted to give insight into how people return to information. One, a modified diary study of fifteen computer science graduate students performing personally motivated searches in their email, in their files, and on the Web, found that even among this technically savvy population, participants preferred navigating to their search target along known paths over jumping directly to it [10]. This preferred search strategy is fragile, failing if any part along the known path changes. A naturalistic study analyzing instances mined from the Web where people expressed difficulty re-finding found that lost information was commonly described using the path originally taken to find it [12].

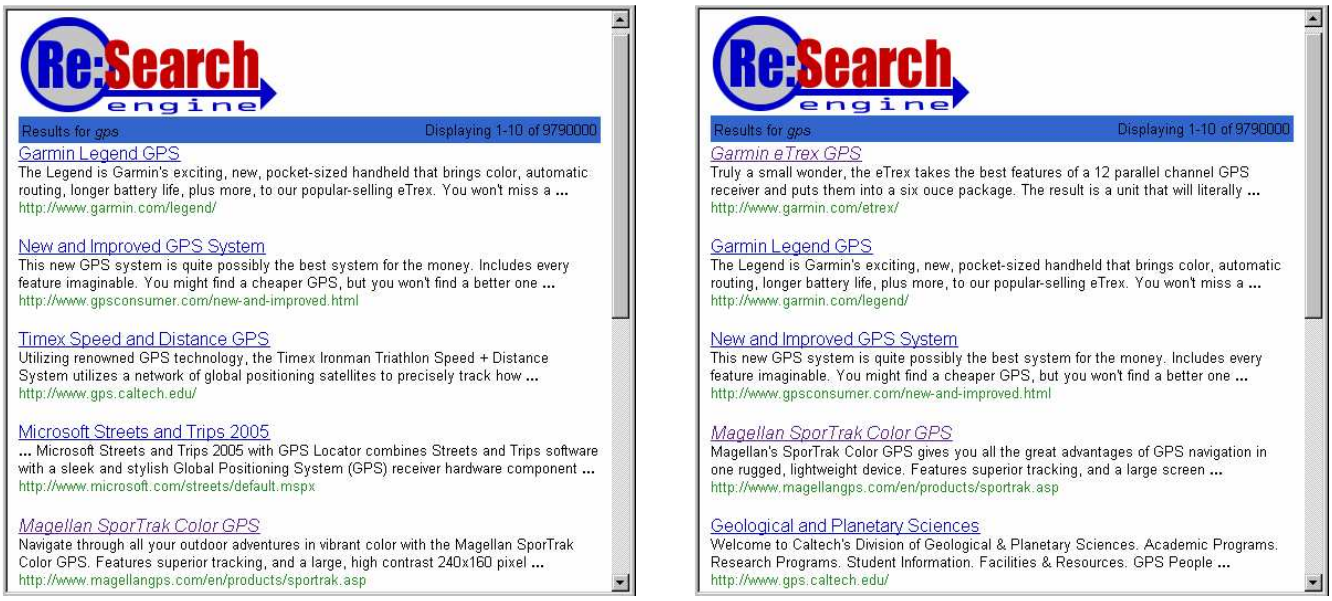
…BUT THE WEB CHANGES

Despite the importance of consistency in re-finding, information on the Web frequently changes. For example, search results, often an important step in the original access path to a piece of information, change when search engines update their indices to reflect the current state of the Web or users misremember past queries. Attempts to improve retrieval quality through personalization or collaborative filtering will further increase the frequency of such changes.

Although Web search engines have traditionally sought to return the search results that are the most relevant to a query without consideration of past user context, some recent search systems, such as A9 [1], allow users to mark pages of interest to return to later. However, people are unlikely to employ keeping strategies that require active involvement [6]. Some search engines also allow people to explicitly search within information they have seen before [1, 4], but these systems do not maintain consistency in result presentation, requiring the user to take a different

Figure 1. Connie's initial results for the query "GPS"

UIST '05, October 23-27, 2005, Seattle, Washington, USA.
ACM 1-59593-023-X/05/0010...$5.00.



(a) Current Web results

(b) Results show to user by Re:Search Engine

Figure 2. An example of the Re:Search Engine in action. Figure 1 shows the search results when Connie first searched for "GPS" (visited links are *italicized*). Figure 2(a) shows the results when the query is next performed, and Figure 2(b) shows how the Re:Search Engine combines what Connie is likely to remember from Figure 1 with what is new in Figure 2(a).

path to the same information. Information management systems that preserve consistency in dynamic environments permit their users choose to interact with a cached version of their information space [5, 9]. While employing similar methods to keep the results for repeat queries static would make re-finding simpler, it would deny users the opportunity to discover new information. For example, if Connie re-issues her "GPS" search, in addition to re-finding the systems she liked before, it is possible she would also be interested in learning about newly available systems. Even though changes to search results associated with a query can potentially hinder returning to previously viewed information, they benefit users by providing new information.

SOLUTION – THE RE:SEARCH ENGINE

The *Re:Search Engine* addresses the dual goals of maintaining search result consistency and providing new information by seamlessly integrating old relevant information with new. The engine interfaces with a preexisting search engine (*e.g.,* Google or Yahoo!). When a person issues a query that has been issued before, the Re:Search Engine first fetches the current results for that query from the underlying search engine. It then merges this newly available information with a cached copy of the results that were previously presented to the user, leaving unchanged what the user *remembers* about the initial result set. Because people tend to remember very little about the search result list they originally saw, it is possible to preserve the salient features of the old results while still presenting new information.

Consider as an example Connie's search. Recall that Figure 1 shows the results when Connie first searched for "GPS". Later, when she re-performed the same query, the results had changed to include several new GPS systems (Figure 2a). Instead of directly returning the new results, which could be disorienting, or the original results, which might omit items Connie would want to see, the Re:Search Engine merged the two (Figure 2b). The merging preserves memorable aspects of the original results, such as followed links (*italicized*) and the ordering of the first and last results, while including new results and an updated result summary.

An exploratory paper prototype study of people interacting with lists of document summaries suggests that many changes, such as changes to the summary wording and to the document ordering, go unnoticed, even when the changes occur as the person interacts with the information [11]. A challenge in designing a system that takes advantage of the fact that people don't notice all changes is to identify which aspects of the information a person interacts with are memorable (and thus should only be changed with care), and which are not (and can change as needed).

THE RE:SEARCH ENGINE ARCHITECTURE

The architecture of the Re:Search Engine is shown in Figure 3. The design of the system is influenced by a study of what 119 people found memorable about search result lists. In the study, participants were initially asked to interact with a search result list and then later asked to recall their query and the results they interacted with without referring back to the original information.

When a person performs a search via the Re:Search Engine, the query is passed through an *index of past queries* that returns similar previously issued queries. These matching queries are used to retrieve the associated previously viewed search results from a *result cache*. Using information stored in the *user interaction cache*, the past results are then *merged* with the live results for the current query and

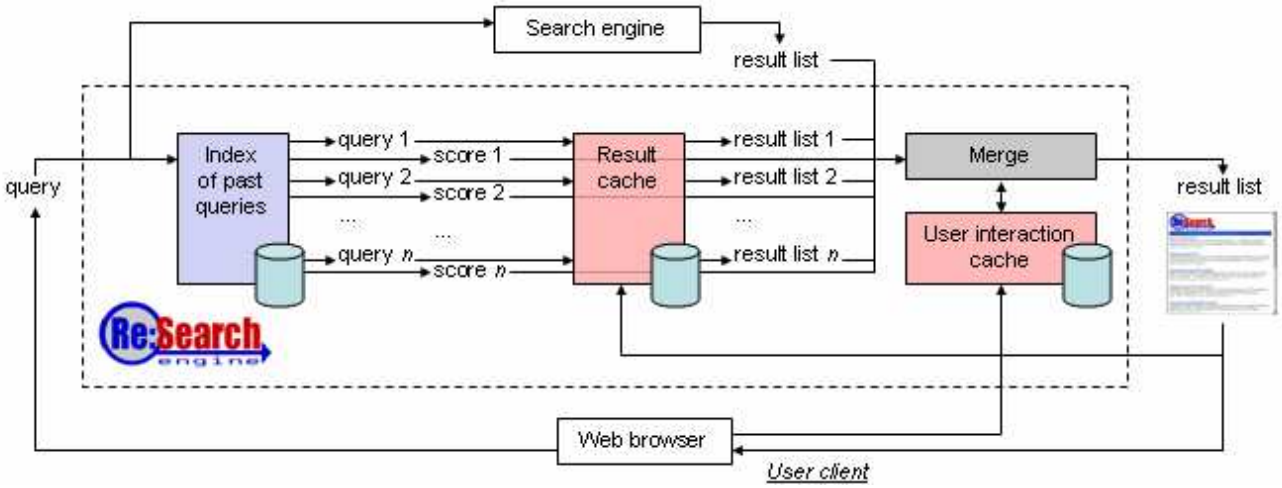


Figure 3. The architecture of the Re:Search Engine. The user's current query is matched to past queries, and the results for the past queries are retrieved from a cache. These results are then merged with the live search engine results based on how memorable the results are, and the resulting result list is presented to the user.

the merged list is returned to the user. Finally, the current query is added to the index of past queries, the merged result list is added to the result cache, and the user interactions with the returned result list are logged.

Index of Past Queries

The purpose of the *index of past queries* is to identify repeat searches. The index functions similarly to a traditional document index, except that the "documents" indexed are query strings. An index was chosen for the query matching both for efficiency and because it accurately reflects how people remember their past queries. For example, word ordering, tense, capitalization and stop words are commonly forgotten when recalling search terms, and these features are removed when a query is indexed.

Not all queries that contain similar text to a past query are repeat searches. During search sessions, people commonly explore variants of the same query, actively seeking new results with each variant. For example, if Connie thought the results returned for "GPS" contained too many expensive systems, she might try searching for "GPS, cheap". These results should not be merged with the results for the query she issued immediately prior. For this reason, past queries that are similar but occurred recently are ignored.

Result Cache

If the query the user issued is determined to be related to one or more previous searches run by the user, the results corresponding to the previous searches are fetched from a *result cache* using the pervious queries returned by the past query index as keys. Only the most recently viewed set of results for a particular query is stored in the cache. For example, when Connie issued the query "GPS" a second time, the results shown in Figure 2(b) replaced the results shown in Figure 1 in her result cache.

User Interaction Cache

Past results of possible relevance to the current query are fetched and merged with the live search results to produce a list containing both old and new results. The merge algorithm is designed to help people take advantage of the context built during past searches, and thus draws on an understanding of how memorable past results are. Implicit measures of attention, like those discussed by Kelly and Teevan [7], can suggest what one might notice during a search. These measures are collected by instrumenting the user's browser to observe the user's interactions with previous result sets and are stored in a *user interaction cache*. The implicit measures currently recorded by the Re:Search Engine are click through and click ordering.

Merge Algorithm

In the merging of old and new result lists, each old result is given a *memorability* score. This score was developed by analyzing what people remember about search result lists, and is computed using past user interactions with the result (*e.g.*, whether the associated Web page was visited), static information about the result (*e.g.*, its rank in the result list), and the result's associated query (*e.g.*, the query's relevance to the current query and its recency). For example, the Magellan SportTrak system that Connie found during her GPS search originally ranked third. The result's memorability is the observed probability of the third result being remembered given it was clicked. This value is increased slightly because the SportTrak system is last result she clicked, and discounted by the elapsed time since the original query.

Changing the presentation of a memorable result incurs a cognitive cost, represented as a *cost of change*, because the result no longer occurs where expected. Changing a result's rank incurs a small cost, while removing a result from the list incurs a large cost. Like the memorability score, the cost of change is based on actual user behavior. For example, the recalled rank of a result tends to be higher than the result's true rank, so the cost of change for the SportTrak system to be listed second is relatively small compared with the cost of its being ranked last. Because a change to a memorable result incurs a greater cognitive cost than a

change to a result that is hardly remembered, the cost of a result list is a function of result memorability and the cost of making the changes necessary to produce that list.

Additionally, each result in the new result list for the current query is given a *benefit of new information* score based on the expected benefit the as-yet-unseen result will provide to the user. If scoring information is available from the underlying search engine, the result's score can be used to represent the expected benefit. However, scoring information is often not available, so the Re:Search Engine uses the result's rank as a proxy. Beneficial results are more likely to be seen if they occur high in the returned result list, and the benefit of a result list is based both on each individual result's benefit and its location in the list.

During the merge process, all permutations of possible final result lists that include at least three old results and three new results are considered. Requiring old and new results ensures that some context is maintained while not allowing the result list to stagnate. The result list with the highest total benefit minus cost is selected and returned to the user. Although considering all permutations naively is expensive, the merge algorithm can be implemented efficiently by representing the problem as a min-cost network flow problem.

EVALUATION PLAN

An underlying principle of the Re:Search Engine is that search results should not merely contain the information most relevant to a searcher's immediate need. Instead, results should account for previous interactions with related information and make it easy to take advantage of past context. Since testing such a principle requires direct user involvement in the evaluation, the Re:Search Engine will be evaluated through a series of user studies.

Several studies in the process of being conducted involve the performance of two searches – an initial search and a research for information encountered during the initial search. Such two-search studies allow for the conduct of controlled experiments and permit the exploration of a number of result orderings in addition to the Re:Search Engine's ordering, including a static ordering, a dynamic ordering, and one where the most memorable results are presented first.

An initial two-search study of 113 people suggests that when a result list changes naively, people often notice the change (between 54% and 73% of the time). In contrast, changes made by the Re:Search Engine are noticed only as often as they are in the baseline static case. A further study is being conducted to look at task-based interaction with result lists. The success of each ordering will be measured subjectively ("Does the user like the result list?") and objectively ("Does the user successfully complete the task?").

A drawback to the two-search studies is that they can introduce artificialities that bias behavior, as it difficult to motivate repeat searches without over-specifying the target. To gain a realistic understanding of how the Re:Search Engine will be used in practice, a deployment of the system is planned, and usage data will be collected and analyzed.

GENERALIZING THE SOLUTION

Although the Re:Search Engine focuses on maintaining consistency between old search result lists and new, other types of information people commonly interact with also change. For example, online news sites change when new stories are written, and Web sites change as their hosts edit them. The growing ease of electronic communication and collaboration, the rising availability of time dependent information, and even the introduction of automated agents, suggest information is becoming ever more dynamic. As stated by Levy, "[P]art of the social and technical work in the decades ahead will be to figure out how to provide the appropriate measure of fixity in the digital domain [8]." Understanding how people interact with search results on the Web while re-finding will shed light on how people return to information in dynamic environments in general, and I look forward to applying what I learn from the Re:Search Engine to other problems in the domain.

ACKNOWLEDGMENTS

I appreciate the support of my advisor, David Karger, and committee, Mark Ackerman, Sue Dumais and Rob Miller.

REFERENCES

1. A9, http://www.a9.com
2. Capra, R.G. and Pérez-Quiñones, M.A. Re-Finding Found Things: An Exploratory Study of How Users Re-Find Information. Tehcnical Report cs.HC/0310011, Computing Research Repository (CoRR), 2003.
3. Graphic, Visualization, and Usability Center. GVU's Tenth WWW User Survey, October 1998.
4. Dumais, S.T., Cutrell, E., Cadiz, J.J., Jancke, G., Sarin, R. and Robbins, D.C. Stuff I've Seen: A System for Personal Information Retrieval and Re-Use. In *Proceedings of SIGIR'03*, 2003, 72–97.
5. Hayaski, K., Normura, T., Hazama, T., Takeoka, M., Hashimoto, S. and Grudmundson, S.. Temporally Threaded Workspace: A Model for Providing Activity-Based Perspectives on Document Spaces. In *Proceedings of HyperText'98*, 1998, pp. 87–96.
6. Jones, W., Bruce, H. and Dumais, S.T. How Do People Get Back to Information on the Web? How Can They Do It Better? In *Proceedings of INTERACT'03*, 2003, pp. 793–796.
7. Kelly, D. and Teevan, J. Implicit Feedback for Inferring User Preference: A Bibliography. *SIGIR Forum*, *37*, 2 (Fall 2003), 18–28.
8. Levy, D. Fixed or Fluid? Document Stability and New Media. In *Proceedings of European Conference on Hypertext*, 1994, pp. 24–31.
9. Rekimoto, J. Time-Machine Computing: A Time-Centric Approach for the Information Environment. In *Proceedings of UIST'99*, 1999, pp. 45–54.
10. Teevan, J., Alvarado, C., Ackerman, M.S. and Karger, D.R. The Perfect Search Engine is Not Enough: A Study of Orienteering Behavior in Directed Search. In *Proceedings of CHI'04*, 2004, pp. 415–422.
11. Teevan, J. Displaying Dynamic Information. In *Proceedings of CHI'01 (Extended Abstract)*, 2001, pp. 417–418.
12. Teevan, J. How People Re-Find Information when the Web Changes. MIT AI Memo AIM-2004-012.