Displaying Dynamic Information

Jaime Teevan
Massachusetts Institute of Technology
200 Technology Sq. NE43-823
Cambridge, MA 02116 USA
+1 617 452 2839
teevan@mit.edu

ABSTRACT

In this paper I introduce the problem of displaying dynamic information. I give several examples where an individual must interact with information that is changing beyond her control. The challenge in displaying this information is to discover how the user's context can be maintained while giving her access to the new information that becomes available. The user should feel in control of the information despite the fact that it is changing. This can be done effectively by understanding what conceptual anchors the user creates into the data, and keeping them constant while changing the other information as needed.

Keywords

Information retrieval visualization, dynamic information, personalization, user interface.

THE PROBLEM

Imagine that you want to run a search on a particular topic that interests you. This search could be performed in a number of ways. You could query a local index for files that you yourself own on that topic. Or you could query Internet search engines for Web pages related to that topic. Additionally, you could perhaps query library indexes, and even other people. Each of these sources will take varying amounts of time to produce results. Should you be forced to wait for the slowest source to return before you can begin exploring that topic? Or is there some way instead to allow you to begin using the current results and have the new information presented to you as it becomes available?

This example highlights a general problem of how to display information that is changing. Once a user has begun to use the information, she develops a belief about what can be found where. I am investigating how to display new information to a user without destroying the context that she has already built.

There are many more situations of interactions with dynamic information, situations that exist today, and that will exist as challenges to the user interfaces of the future. This situation could arise because the information that one is interested in is partially controlled by family members, colleagues, other collaborators, and even other automated processes. Or it could arise because the information is time dependent. An example of multiple user information sharing is a user's interactions with financial files to which he and his accountant share access. If the accountant reorganizes the files, the changes should become apparent to the user without disorienting him. Examples of time dependency can be seen in the search result example above, and also in things like stock prices and news stories. While there has been some work with displaying dynamic information, such as Ahlberg and Shneiderman's work with dynamic queries [1], the question of how an individual interacts with changes outside of his control is largely unexplored.

APPROACH

I have undertaken to investigate the problem of displaying dynamic information to a user by first performing a user study on a sample problem to understand what elements of context are important, followed by a design of an interface that preserves those important aspects. The sample problem I use is text clustering. Many clustering algorithms are able to roughly group documents into an initial clustering, but may take more time than the user is willing to wait to create a good clustering. The goal is to allow a user to begin working with the initial clustering immediately, while still receiving the benefit of later clustering improvements.

One way to achieve this goal is to ensure that what the user learns about the clusters while using them doesn't change unless the user explicitly changes it or allows it to change. This means investigating further what the user remembers about the information he interacts with.

Clearly a user does not have an understanding of information that she has not seen; so, unseen information is free to change as needed. Since many of the instances where the underlying information is changing involve very large collections of data, it is quite likely that the user will never see most of it, making the issue of how to maintain the user's context trivial. With my implementation of clustering, the information that the user has not seen includes the documents in the clusters that she has not visited and the documents in the clusters she has visited that are not ranked highly enough to be displayed.

However, the information that a user has seen, she expects to be able to find again. The user has developed some *conceptual anchor* into that information, and when she wants the details of the information again, she will use that same anchor into the information to retrieve it. I propose that a good interface for interacting with dynamic information allows as much information as possible to change, while ensuring that those anchors the user has developed remain constant unless she has explicitly understood them to have changed.

Conceptual anchors are a function of what a user expects from the information she is working with. When you watch news in the evening, you may expect the news presented to you on the television to be changing, so in addition to remembering pieces of the story, you may also relate it to a specific time to put it in a context. But unless you have recorded it, you do not expect to be able to return to the story, so you may not develop anchors to aid in returning to the story. Note that this is different from when you read a newspaper article. In this case, you expect to be able to return to the news story. Instead of remembering the time you read the article, you may remember the section of the paper you saw it in.

I found that if I preserved several conceptual anchors in the clustering problem the other information could change as needed. For example, a cluster is described by a set of keywords, found based on common word occurrences within the documents contained in the cluster. From initial tests, it seems that the user generates a general theme for the cluster from the keywords, and does not notice small word changes within the keyword list. This is especially true because I represent each cluster with a unique color. The user quickly associates the color with the cluster, and uses this mapping to navigate between clusters, rather than using the keywords.

Within each cluster a short preview of its documents is displayed, including a title and short summary for each document. Documents are ordered by their relevance to the cluster. From the tests I have performed, the user does not seem to notice the order in which the documents are displayed, as long as the first document remains first, and all of the visible documents remain visible. I believe that the first document's position is important for several reasons. It is likely the first document looked at, and may be remembered for that reason. Additionally, being first is more distinctive than being in the middle. Its position is obvious, where as a document located half way down the list could be perhaps fourth or perhaps fifth. The user does care about which cluster a document was located in when he first saw it, but does not seem to mind if it later also shows up in a cluster where it is related by content.

As long as the anchors are kept constant, each of the test subjects expressed a feeling of total control over the information, and often articulated surprise when I informed them that they had been working with information that was changing. "You say information was changing," one woman said to me, "but I did not feel like it was changing."

Allowing the user to access information that is changing will also involve the user more intimately in the processes that change the data. In my design, in order for the information that the user has seen to change, he must actively approve the change. Because of this, in some cases the user may be able to easily incorporate his personal opinions into the changes, and can even stop mistakes as they are happening. Relevance feedback from the user becomes both a natural and integrated part of the interface.

FUTURE WORK

Above I have presented a new user interface problem, and discussed one solution to that problem. Another possible solution could be to give the user a static snapshot of the data, and then update that snapshot at the user's request. This is, in a way, how current interfaces deal with dynamic information. When you view a Web page, you see what was there at the time you requested that page, and then, if you want, you can actively update the information on that page by refreshing. Similarly, each day you receive a newspaper that replaces the newspaper from the previous day, refreshing your view of the news. I would like to compare this snapshot solution with my proposed solution.

Additionally, within the solution I propose, there are many sub-questions to answer. It is essential to build a greater understanding of what exactly a user uses as conceptual anchors, as well as an understanding of the differences among users of what anchors they develop. Up until now most of the tests I have performed have been with lo-fidelity paper prototypes, and further work with the hi-fidelity implementation will be useful on this front.

I would also like to apply this solution to domains other than clustering. I believe that the conclusions I have drawn are general enough to apply to any piece of dynamic information with which a person may interact. I am particularly interested in applying it to the Haystack project [2]. A person's *haystack* can be seen a repository of all of the information that that individual comes into contact with. The system is constantly looking over this information, and drawing new conclusions about the data it contains. For example, it may decide that a certain email I wrote my advisor is related to a paper I'm writing, and link the two. Or it may decide that I am interested in cooking, and run a search on the Web to collect new recipes for me. In this way, the information within a haystack is constantly changing, both in what is available, and in the relationships between objects. By effectively integrating the display of dynamic information into Haystack, we can create an interactive and dynamic desktop where the system and the user work together to organize the information.

REFERENCES

1. Ahlberg, C. and Shneiderman, B. Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. In *Human Factors in Computing Systems: Proceedings of the CHI '94 Conference*, 1994.

2. Adar, E., Karger, D.R., and Stein, L. Haystack: Per-User Information Environments. In *Proceedings of the eighth international conference on Information knowledge management*, 1999.