Memoplex Browser++

Clarence Chan

University Of British Columbia, Department Of Computer Science

ABSTRACT

Document search techniques often list search hits as a flat list of results, with very little contextual indication of the relationship between document hits except for their *absolute* relevance to a search query. However, there may be auxiliary structural information to a database of documents that could be pertinent to information retrieval on the user's end, such as connections between documents as identified by a semantic network.

Memoplex++ is a text document browser designed to address these issues of structure in information retrieval; it is designed for viewing articles stored in the Memoplex semantic network **[8]**. Memoplex++ visualizes semantically related documents as text-based previews connected in a radial graph view, exploiting the spatial locality of related articles to support document browsing. Search results are spatially clustered to provide users with general sense of direction and orientation in this abstract document space.

An informal user evaluation revealed that the spatial positioning and general design tradeoffs of Memoplex++ provided no clear advantages over a traditional list-based view of documents. However, there were perceived benefits to the use of semantic links to indicate the relative pertinence of groups of semantically linked articles.

1 INTRODUCTION

Perusal of text documents and articles is a central process of research in many fields and disciplines. However, there are often vast amounts of literature for any given area of discourse. Thus, the process of extracting useful information from textual sources encompasses not only the review of a large numbers of texts in a general area, but also the filtering and selection of texts relevant to one's research from this multitude of documents. As online electronic document repositories become increasingly prevalent, it is important that tools be developed to to support their use so that researchers can and extract and filter the information they need from these document repositories. This scenario of finding relevant documents in a large corpus is a specific example of the kinds of issues dealt with in the field of information retrieval.

Some of the more common approaches to information retrieval from a collection of texts are the rather simple concepts of searching and browsing. When a user wants to do a textual *search* of a collection, she will normally interact with the collection through some query interface; the user provides several keywords to the interface, representing a particular query about the area of interest, and is in turn provided with a set of documents that the interface deems to be relevant to the user's initial query. However, while even naïve search algorithms are capable of globally filtering out irrelevant documents when returning results to the user, the list of relevant results returned is often still rather broad. Unless the user knows *exactly* what she is looking for, it becomes difficult to further narrow down the list of results with more specific keyword searches, and the problem of finding the "right" documents becomes a task of **visually** perusing the returned documents, one at a time.

1.1 Issues with search, browsing as complement

From an analytical standpoint, directed search techniques are less well suited for locating particular items of interest from within a local group of ostensibly relevant results, once the search space has been initially narrowed down. When people begin on research in a given area, they do not always have starting points or specific documents as targets to search for, as they are unfamiliar with the domain anyways. As mentioned earlier, the task of further narrowing down an abstract search becomes one of document perusal, or *browsing*. Browsing is the natural, complementary task to search, when the user is in an exploratory state of mind and wishes to peruse a set of documents for potential good hits. In contrast to search, browsing is a less directed task (does not focus on matching a particular query in a large global set), and banks on the user's ability to identify elements of interest by having her **visually** browse through a small local group of related materials.

We can think of browsing in this context as the "second" step of information retrieval following search. Search narrows down a large domain to a smaller set of related materials that are somewhat relevant to one's query; browsing allows a person to further dive into a specific area to identify the key elements of interest. One analogy often used is the process of going to a library, looking at an index of call numbers to find a specific subject area (search), then going to that section and flipping through the various books in that particular set of stacks to get a feel for the different texts in that area (browsing). At the lower level of browsing through books in the stacks, there is more of an emphasis on local and relational links in the data (article cross-references, related works), rather than the absolute and hierarchical nature of call numbers.

It is then natural to suggest that these kinds of browsing strategies can be carried over to the electronic realm. Visually speaking, we would like to have some kind of visual formalism that displays documents in terms of local relevance to one another so that we can fully utilize the power of browsing and perusal. However, there must exist a structure to a given database of articles beyond a simple flat arrangement of documents for this to work out; an explicit hierarchical or node-link model of relationships, such as a semantic network, is tremendously useful in this regard.

2 RELATED WORK

There have been many different approaches to the general problem of visualizing electronic documents effectively. We present a brief overview of some of the relevant works.

2.1 Individual Document Views

Although there are a huge number of different techniques for viewing individual documents that have been proposed, most of them employ the basic principle of a focus + context view, allowing users to focus on a particular region of interest within a document while maintaining contextual document information at the periphery. Even those methods that are interaction-heavy still use a general notion of focus+context in their implementation. For

example, Hinckley's work on speed-dependent zooming **[7]** uses automatic zoom-outs to relate the general structure and relative size of the document as the user scrolls through the text at varying rates; we can think of the context as the greater overview of the text, and the focus as the static view of a document region.
However, the classic example of a focus + context technique can be seen in Furnas' seminal paper on fisheye views **[5]** and other works, such as using fisheyes to view source code **[4]**. Furnas' original fisheye paper describes the basics of *abstract* foci and and contextual information in a given domain, explaining this view of the world with regards to a generalized *degree-of-interest* framework.

Generally, focus + context methods have been empirically determined as being superior to plain linear views; an example of such a study can be found in Frokjaer's work **[3]**. Frokjaer compared linear views of documents to fisheye views and overview + detail views (the latter of which is a focus + context-style technique, displaying a multi-page overview of a document in a side panel in addition to the main document text as in the commercially available Adobe Acrobat Reader).

What we would like to expand upon from these papers is the notion of providing context to not only a single document, but to a set of documents by cleverly manipulating the visual space in which they are presented. Context in multiple document views should theoretically allow for the user to browse through contextually similar or relevant items that are adjacent to the main item of interest.

2.2 Multi-Document Visualizations

Another approach to document visualization involves the viewing of particular attributes of directories or databases of texts, and the ability to spot trends across documents in a given corpus. These trends are then typically visually presented to the user to give them some sense of information regarding the corpus at a higher level (i.e. categories or general subject matter). Most of these techniques actually pre-date the single-document visualization techniques, and are often concerned with concise semantic representations of document content across corpora.

One of the earliest works by Hearst, TileBars**[6]**, accepts search queries and ranks the relevance of documents in the database with respect to the terms. For each document, a set of horizontal bars is displayed; each bar represents a single search term, and the darkness of an individual tile in a bar represents the frequency of the search term in that section of the document.

Related work by Byrd **[1]** describes and implements a search highlighting tool within an online document's scrollbar; the scrollbar is populated with small coloured tiles which denote search term occurrences throughout the document as a whole, with different colours representing different terms. These kinds of data representations arguably are the inspiration for the later overview + detail interfaces for single documents that provide large-scale views, often in conjunction with search data.

While these mechanisms reveal interesting high-level data about multiple documents, they do not represent the actual document content very well at the low level, and so the actual text of two documents sharing relevance with respect to a set of keywords or concepts is not immediately available to the user to browse through. Our approach aims to tackle this problem by providing the user with the actual meat of the text.

That having been said, there has also been research into the use of 3-D depth cues to organize multi-document data. The concept of Piles **[10]**, followed by Data Mountains **[11]**, investigate the notion of arranging iconic representations of documents in a similar manner to the real world, making use of piles and layers. Documents can be stacked upon each other and in the case of Data Mountains, can be filed individually in the foreground or background, generating a sense of depth.

However, these methods again either attempt to capture the totality of a document in a single glyph, or try to represent too much information in a single panel; additionally, these methods are heavily interaction-dependent and are not strictly intended to support browsing, as it were. Simple interaction methods and a reasonable amount of information density in the actual text are needed for browsing behaviours.

2.3 Semantic networks

Semantic networks are computational constructs designed to support the notions of local and relational links in data; in particular, semantic similarity networks represent sets of items as graphs wherein individual nodes (items) are connected by edges to other nodes based on the strength of their relation to one another.

In the case of text documents, semantic networks can be constructed that indicate how closely related different documents in the graph are; documents sharing many keywords, and more importantly, that share many higher-order attributes (as determined by some metric of similarity), are linked together by an edge and can be thought of as tightly related to one another. There are many different ways in which these relational links between documents can be generated; some of the more prominent examples in the field of information retrieval that perform this important task include the *tfidf* algorithm and the process of latent semantic analysis **[2]**. Because of this, semantic networks are tools that can potentially effectively support browsing of documents.

One example of a semantic network implementation is seen in Mike Huggett's Memoplex server **[8]** (which happens to be the backend for our document visualization interface). The Memoplex server maintains and administers several sets of document corpora which have been structurally arranged into a semantic network. Document relationships are calculated and weighted within the Memoplex server according using the *tfidf* algorithm mentioned previously, which inducts similarity relationships between documents based on non-trivial keyword matchings.

Our work attempts to treat this similarity metric as a meaningful measure of locality of documents in an abstract information space, and visualize strongly related documents accordingly so that these similar documents can be browsed and viewed at a high level together. In particular, we build on the original Memoplex Browser **[9]** designed by Lanir, which is an interface for visualizing the large-scale structure of the semantic network itself, rather than the individual nodes (documents and their text). A brief discussion of the original Memoplex Browser (fromhere on referred to as Memoplex Browser version 1, or Memoplex V1 for short) is useful at this point for some more insight into our background motivation.

3 Memoplex Browser, Version 1

3.1 System Description

The Memoplex V1 interface is a front-end to the Memoplex server comprised of four main components (see Figure 1):

- Keyword search pane
- Radial graph view
- Document focus pane
- Visited node (history) timeline

Given a corpus of documents and its corresponding semantic network, Memoplex V1 is designed to visualize the structure of the network in the radial graph area when a search result is selected in the keyword search pane. The radial graph view itself shows no document text, but merely individual nodes and edges representing documents and their semantic connections. All the panes and the timeline are initially blank until the user selects a search result from a successful keyword query. This selection action causes the radial graph view to be populated with a local network of nodes that are either directly or indirectly connected to the currently selected search result (represented in the graph by the node in the middle). Clicking on a node triggers several actions: an animation places that node in the middle of the graph and re-orients its neighbours accordingly, while displaying the node's contents in the document focus pane. Memoplex V1 also features the ability to automatically colour-code all nodes into their respective clusters (clusters being determined by a *k*-means process with the semantic similarity measure as the metric).

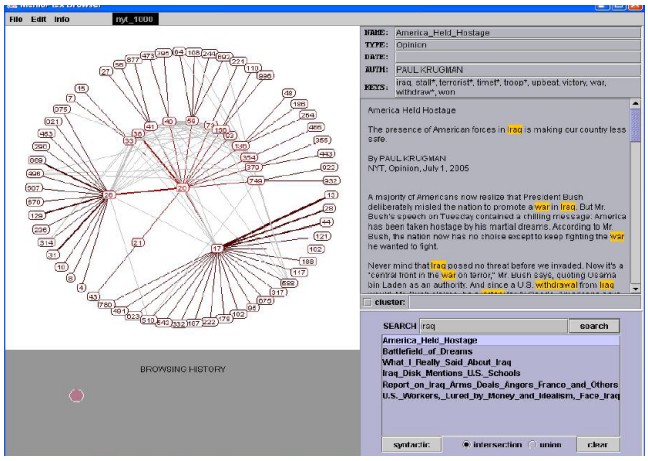

Figure 1. Memoplex Browser, version 1.

The document text itself is presented in the document focus pane in the upper right, along with other useful information such as the document's title and author. The node timeline at the bottom left presents the user with a list of previously visited nodes. However, the bulk of the visualization content lies in the radial graph view. Ostensibly, the ability to see the structure of the network provides the user with a contextual view of similar documents, and an understanding of the general nature of articles in that general theme and how they relate to each other.

3.2 Criticisms

To understand the motivation behind the changes made in Memoplex++ (section 4), we first discuss a number of criticisms of the Memoplex V1 system below.

One set of issues with the existing Memoplex Browser interface is that only one actual document's text is physically visible at any one time, and more space is devoted to visualizing the entirety of the semantic network than the document of interest itself. It is arguable that being able to visualize the large-scale structure of the semantic network is not as useful as being able to see the actual text contained within those documents, as it is the text that ultimately is evaluated by the user when deciding whether or not a document is useful or not. The graph structure be designed to support browsing by offering some kind of context as to the actual *content* of the document, *within* the existing context provided by the edges and general spatial arrangement.

From a task-based point of view, the purpose of the semantic graph should not be to visualize the entirety of the network, but to provide connections between the current document of interest and other related documents that might be worth browsing and looking at. The rational for using semantic networks in the first place is to support browsing, which is inherently a locally directed task; thus it stands to reason that only a small number of nodes that are directly connected need to be visualized on the screen at any one time.

Furthermore, the spatial arrangement of the nodes should be used in such a way that some notion of proximity and similarity of documents can be encoded in the actual physical location of the nodes as well. Spatial encoding is known to be among the strongest encodings, and capturing some kind of data along this dimension would allow us to visualize more pertinent information at minimal cost.

There are also a host of other issues with the general usability of the system. One notable problem was the general lack of flexibility of the search engine and its limited keyword recognition, which limited the variety of different keyword searches that could be performed, as well as effectively hindering the user's ability to search with more than a single query term. Another problem with the system was the excessive devotion of screen real estate to the history timeline panel, which serves no clear benefit given that all the relevant nodes that have been previously viewed would have been in proximity to the current node in the graph view anyways.

4 MEMOPLEX++

We addressed these issues with a second iteration of Memoplex Browser, called Memoplex Browser++. Memoplex Browser++ aims to more effectively support local browsing through a number of augmentations to the interface and its underlying engine. It keeps the general setup of a radial graph view on the left, a search pane in the bottom right and a document focus pane in the top right, but does away with the history bar and makes even more space for the graph view by shrinking the document focus and search panes. More details are provided below.

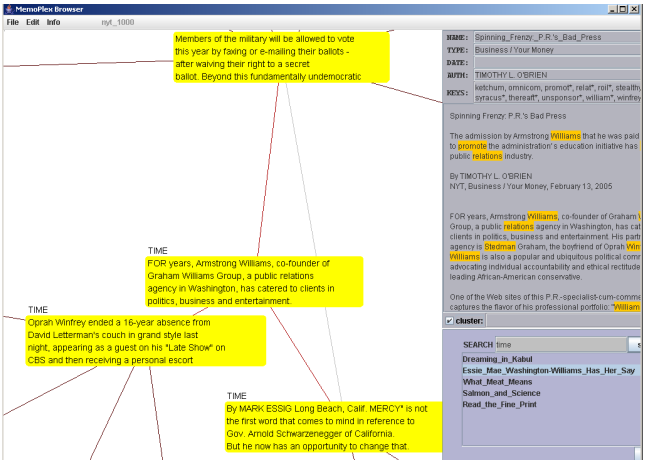

Figure 2. Memoplex Browser++

4.1 Implementation Tools

Our work was built on top of the original Memoplex Browser, which itself was created using Java and the Prefuse toolkit for drawing graphs and providing interactive controls. The Memoplex server operated as the data backend for our visualization tool, serving documents from a semantic network of 1000 New York

Times articles. We also used the Google Desktop API to supplement the search pane in its search tasks.

While the Prefuse and Google Desktop API's provided very basic functionality for creating basic graphs and simple searches, much of the manipulation of specific visual elements, search construction and clustering of data had to be coded in. In each following subsection we describe a main feature of the augmented system, as well as a brief description of the work involved in implementing that feature.

4.2 Radial Graph View

4.2.1 Document Previews

The arguably most important aspect of the Memoplex++ system is the introduction of document previews into the radial graph view itself. Text snippets of the actual documents that the nodes represent are parsed and inserted into the visual representation of the node. Because the New York Times corpus of articles has many non-standard formats in their representation, and no 'canonical' structure to abide by, some amount of low-level textual parsing and manipulation was required to extract meaningful headers that captured enough content to be representative of the general idea of the article.

Additionally, a non-trivial amount of effort was invested into making the Prefuse node renderer accept multiple lines of text in the display of the node while maintaining proper node boundaries and shape around the featured text, depending on the size at which the text was rendered at.

The purpose of rendering actual text snppets of the article within each node is to provide the user with a convenient mechanism through which to visually browse the content of peripheral nodes without the substantial investment cost of setting it actually opening up that document or viewing it full size in the document focus pane. In this situation, the complement to locality in browsing is context; the radial graph is meant to provide a suitable context to browse within, using a limited amount of screen space, and so a reasonable compromise is to provide just enough textual data for the user to casually glance at and make inferences about the domain of articles, without committing to any particular choice of action.

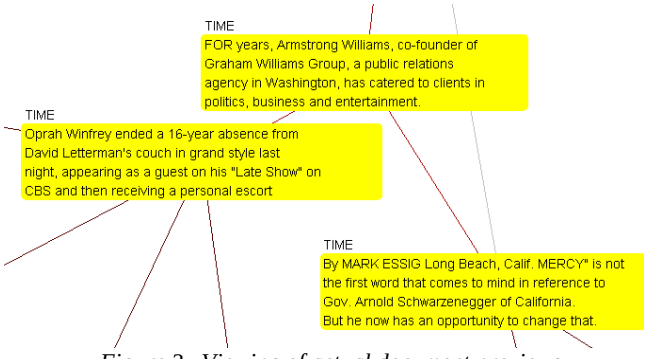

Figure 3. Viewing of actual document previews.

4.2.2 Constrained Network View

To accommodate the increased size of individual nodes in the radial graph view, the depth of the visible graph, as well as the total number of nodes visible, had to be culled from the radial graph view. However, as it turns out, we can provide just as much relevant information in the context of browsing in this reduced environment. In fact, the reduction of clutter in the radial graph view serves an auxiliary purpose in *only* presenting the information necessary for local awareness of documents, thus augmenting the user's capacity to visually browse a select group of strongly connected items rather than a whole host of weakly connected ones.

The depth of the visible graph layout is severely reduced in the final implementation of Memoplex Browser++. Instead of showing a potentially unconstrained depth of semantic levels as in the original browser, our augmented system *only* shows items that are one 'hop' away from the focus node of interest at any given point in time; to access elements that are any further out than one 'hop', the user must bring a peripheral node into focus by clicking on it to see *that* node's surrounding periphery. This allows for the user to concentrate on the most directly relevant items when doing visual browsing rather than be distracted by nodes that are several semantic hops away, which have a much looser attachment to the element of focus and most likely have a much lower degree of interest relative to the current focal point (to use Furnas' terminology).

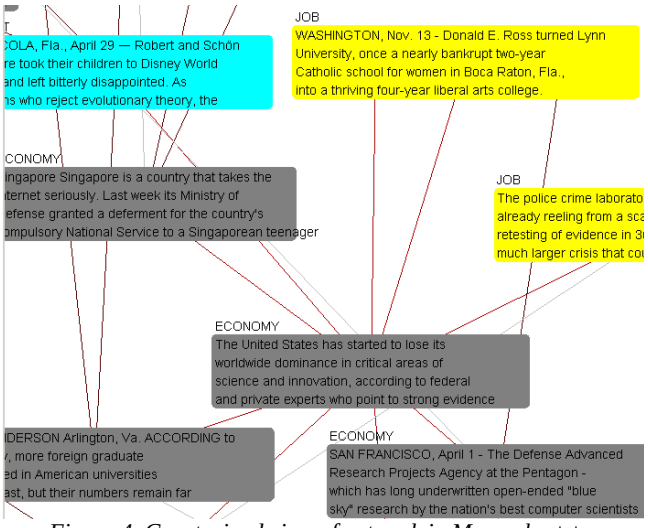

Figure 4. Constrained view of network in Memoplex++. Compare with Figure 5 below.

Accomplishing this required the manipulation of several structures and methods within the Prefuse graph layout abstraction layer so that an arbitrarily large depth of nodes would not be brought into view at a single time. Along similar lines was the decision to cull out the total number of nodes in the given radial graph of a search result. While initially we had intended to show the most strongly connected nodes and edges in the graph and

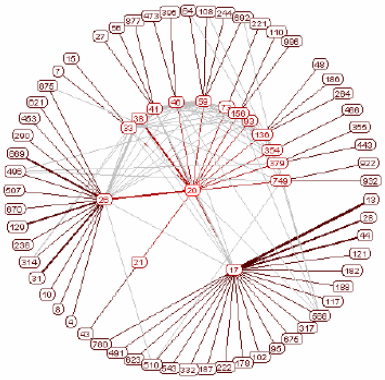

Figure 5. Memoplex Browser v.1. Unconstrained view.

hide the weaker ones to be revealed upon request, this was found to be considerable work for little perceived benefit; it would make more sense to increase the threshold for generating semantic links

between documents in the server and order the potential candidates for nodes when importing the actual graph structure into Prefuse.

This is a reasonable compromise, given that by making our conditions for generating connections tighter, we can weed out those documents which had little keyword relevance to the main search target and surrounding documents anyways, thus automatically eliminating those possibilities as candidates for browsing without troubling the user with them.

One other feature that arose out of this condensed graph representation was the creation of a panning control to move around the graph. While the existence of such a control undermines the intent to create a system designed chiefly for local browsing, as the panning allows the user to violate the assumed interaction model of a local browsing scheme. However, because we found early on that the graph structure and transition animations were not perfect and left items at the fringes, while also confusing users as to the location of certain items, a panning control would be included as a concession to these kinds of tricky situations (so that one could navigate back to a familiar area or to bring a partially obscured item at the border back into focus).

4.2.3 Refined Clustering / Spatial Layout

While the clustering algorithm of the original system coloured items appropriately and provided global information in terms of the overall cluster membership of each individual document, it suffered from the problem of clusters not being easily interpretable by any human user (i.e. She would not be able to identify the conceptual differences between articles in different clusters). Additionally, the cluster visualization relied on merely colouring nodes, yet the very nature of clustering suggests the use of spatial proximity and collocation of similarly clustered items as a visual representation.

We address this by trying to refine the clustering approach, albeit in a rather naïve way. Firstly, we re-worked the search algorithm into a more flexible form; as previously mentioned, the search engine was not very friendly and had trouble returning search hits at times unless very explicit syntax was used, so we rewrote part of the search mechanism using the Google Desktop API, so that the search engine could take up to three arbitrary keywords as query terms.

Then, when the system is given a search result in the form of a document, each of the document nodes in its local network is rated according to each of the query terms entered. This step is carried out by running each individual query as a search on its own, and assigning each document to a cluster corresponding to which keyword search it ranked highest on. Put more concisely, each article semantically linked to a main document is classified into a cluster based on one of the keywords of the search.

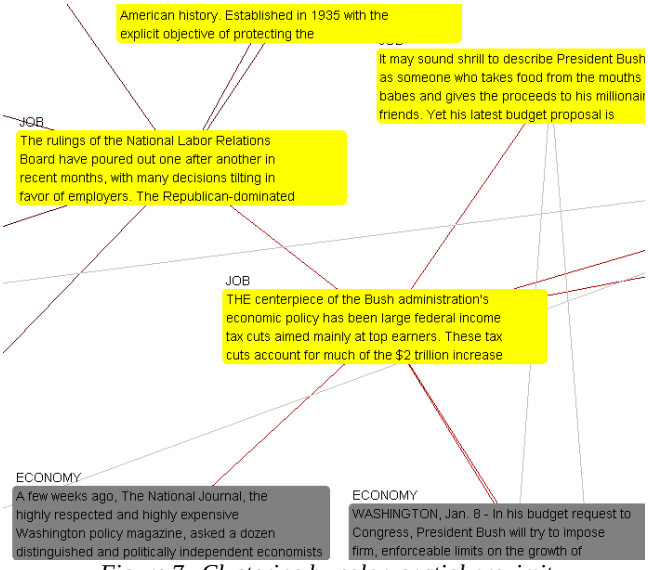

Figure 7. Clustering by color, spatial proximity.

Finally, each node is coloured according to its cluster, and each cluster is assigned a certain region on the screen, where all nodes of that colour will rotate to when the focus is changed. The figure below shows how each cluster is roughly relegated to a certain portion of the screen. If any particular item is brought into focus, the animation that brings its surrounding nodes into view swings them into those general regions according to their cluster group. As seen in the picture, the name of the particular keyword cluster is placed at the top of the node so that the user has some indication of the conceptual meaning of the clustering.

Additionally, this enhanced clustering gives users some notion of orientation in an abstract space of text documents; because documents always re-orient themselves to the appropriate spatial location when re-focusing animations occur, users can utilize the redundant coding of color and space to move themselves more towards documents that are relevant and/or similar with regards to a particular keyword by moving in a single general direction.

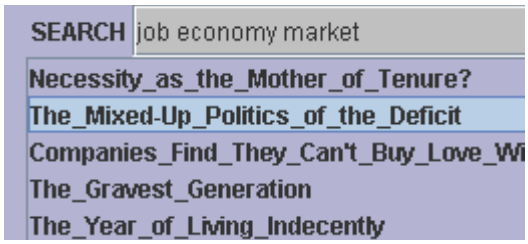

Figure 8. Search pane.

5 Usage and Evaluation

Given these augmentations, we describe a possible scenario of usage that these enhancements are meant to support, and then mention an informal user evaluation carried out to analyze the usefulness of our system in comparison to a flat list system.

5.1 Scenario Of Use

5.1.1 General Browsing And Research

Joe is doing research on interactive fisheye views in visualizations. He has at his disposal an online library of infovis documents that cover many different kinds of visualizations, such as ZUI's, fisheyes, overview+detail interfaces, map visualizations and tree/graph views. However, he is not sure of the exact information he is looking for, and thinks that browsing a database of articles might be helpful in getting him some ground footing. He opens up his augmented Memoplex Browser and types 'fisheye visualization interactive'. The Memoplex Browser returns a graph with the most relevant document as suggested by the search engine: Furnas' 1986 paper on fisheye visualizations. He clicks on the document and casually browses through it in the document focus window, before realizing that it is mostly abstract theory on the concept of fisheyes and degrees of interest, and while he keeps this document in mind because it is in his general area of interest, it does not have examples of interactive fisheye visualizations and so he looks at the semantic graph to see what else is in the immediate vicinity that might be of use.

In the area of the 'visualization' cluster at the top he sees an infoviz taxonomy paper written by the same author. However, more importantly, on the bottom-left hand cluster for articles with the keyword “interactive” he sees a document on interactive focus+context views. This paper is a little more general than interactive fisheye views, but Joe figures that a lot of related work should fall under the subset of interactive focus+context views that might be fisheye work. He clicks on this document, and it moves to the center, replacing the fisheye paper by Furnas. This new document is now surrounded by two other documents in addition to the Furnas paper which belongs to the “fisheye” cluster. The other documents are classified as “visualization” and “interactive” papers respectively; Joe double-clicks on the former, peruses it and finds that it is very useful to his research. He sees that the “interactive” paper is really more about HCI interaction in general, just by looking at the document preview, and so ignores it.

However, Joe is not done. He wants additional examples so he looks at the documents surrounding the paper he has just double-clicked on, which has now moved into the center. He sees two other related items, semantically linked by the graph; he peruses their snippets quickly and saves them for later use as he sees that they seem to be quite relevant based on the abstract alone.

5.1.2 “Fuzzy” Search And Browsing

Eric wishes to find a recent article in the New York Times that he read about concerning the Iraq war, and oil reserves. However, he doesn't remember the title or the author so he fires up Memoplex Browser++ and enters the terms “iraq war oil” into the search box. He double-clicks on the first search hit he gets and is presented with a radial graph view. Most of the articles he sees are about war in Iraq but none of them happen to be the particular article he is looking for, which he can easily tell by browsing around and looking through the snippets.

However, he comes across one heavily linked article with many semantic connections to other articles about the Iraq war. Using this particular article as a 'landmark' of sorts, Eric reasons that the article he's looking for should be in close proximity since almost every article is connected to this main 'landmark' one. By bringing the 'landmark' article into focus and looking at its local “oil” cluster, Eric sees the article he is looking for amongst the group and saves it, his goal having been accomplished.

5.2 Informal Evaluations

5.2.1 Description

A set of informal heuristic evaluations of the Memoplex Browser++ software were carried out with four users between the ages of 24 and 30. Users were asked to compare the Memoplex Browser system to a simplified listing of Google Desktop search results in Mozilla Firefox with an adjacent document focus pane (which was merely a separate window controlled by the main one).

While there were dissimilarities between the Google Desktop and Memoplex Browser++ software in terms of the number of search results returned (despite the search results being drawn from the same algorithm, the Memoplex Browser++ system does not allow for more than 10 results in a given query), and the fact that the Google Desktop system had no clear analog to the radial graph view of related documents (aside from the “Similar Pages” link beside each document, which users were encouraged to use), we felt that this was a reasonable comparison to make given that were evaluating the utility of the different kinds of views of snippets (linear vs. spatially arranged), given reasonably workable search modules for both systems.

5.2.2 Task

After instructions on the usage of each interface, users were asked to browse around using each system, and were suggested two sets of search terms to start off as example queries in each system: “economy job market” and “iraq war oil”. Users were asked to do this with one system at a time; order of presentation was counter-balanced. Users were asked to browse through each system first using those search terms, and to make mental notes of the usability of each system. Also, users were asked to make a note of a) for which system was it easier to find the most relevant article according to those search terms, and b) for which system was it easier and/or quicker to get an understanding of the general area those queries were based in. After trying out the first two search queries, users were asked to play around with the system for another 9-10 minutes each in freeform to try to get a feel for the search/browsing experience of each.

5.2.3 Results

While users found that the Memoplex interface was interesting and novel to use, they still preferred the Google Desktop search interface overall. First and foremost, they found the navigation techniques used in the Memoplex interface to be confusing and unintuitive at times. They mentioned that the animations actually interfered with their perusal of documents, because document snippets that stayed in context on a focus change would reposition themselves almost arbitrarily when being viewed, despite the fact that they were only being glanced at and had left the users' center of visual attention by the time they re-focused the graph. We reason that this is the case because the contextual area becomes very perceptually volatile due to the shifting clusters and documents, making it difficult to keep track of the periphery when jumping from node to node.

Users also pointed out that the scrolling mechanism of a flat list such as Google Desktop was much less error-prone and frustrating than the clicking action for the Memoplex interface. If users double-clicked on the wrong item or wanted to navigate back to a previous element, they would have to take the time to locate said correct element (or previous element) and re-focus it, causing another spatial shift. When the cluster-smart spatial shifting was mentioned most users didn't even notice. In fact, for the majority of their own searches in the freeform session, users only typically used one or two keywords, using the cluster as very secondary information. One user mentioned that the clustering of keywords for the first two searches was heavily skewed towards the first keyword and that the radial graph view seemed devoid of but a few results in the other clusters, which she felt to be largely meaningless and somewhat arbitrary.

By contrast, the flat list had a much wider tolerance for skipping over an item, so to speak, since users could always go back any number of pages to find that item in the same, static location, whereas with the Memoplex software one's view was constrained to the local existence of document nodes and their snippets, which meant that an 'alternate branch', as put by one user, would require considerable effort to return to. Despite not having any of the advantages of related document links or clusters on the radial graph view, users felt the Google Desktop interface worked in a more efficient and workable manner as the cluster labels and edges did not clutter the view. In particular, two users explicitly mentioned that there was no clear advantage to the spatial layout that a flat layout could not do, given the same snippets of text.

However, one user pointed out that the local nodes and the surrounding view of the Memoplex software made it easier to fit *more* information on the text than the standard flat list. Because there are no overlapping items in the flat list, we found it a pleasant surprise this user was able to tolerate the occasionally overlapping text boxes in the Memoplex software, which perhaps suggests that the length of the snippet can be cut down even further, and that excessive contextual information may not actually be strictly necessary.

He also mentioned that being able to immediately see a large number of close links to a document influenced his opinion of the document and encouraged him to look at it despite initial reservations based on the snippet. We find this interesting because it is a good example of supporting local browsing by hinting at strong relationships between documents rather than asking users to analyse and evaluate each document on its own merits as in the flat list interface.

6 Results, Discussion

It makes sense at this point to reflect on why the users in our evaluation found the system to be largely unhelpful in supporting browsing tasks. Despite the initial desire at the outset of this project to avoid the tarpits of attempting to visualize network structure rather than the more useful textual information contained in the actual documents, it turned out that our work still replicated too much of the internal structure of the data which was largely not beneficial given the costs incurred. The clutter introduced by imperfect spatial organization of items into clusters, and the introduction of an extra dimension of navigation made the Memoplex Browser++ system much more difficult to navigate than a simple flat list. The perceived benefit of exploiting that extra dimension and set of clusters was not reflective of the way that most users actually chose to utilize it.

Constrained navigation along one dimension captures the essential nature of text as being highly sequential and extremely well-ordered; the use of clusters and extra dimensions to classify and position text along most likely interfered with the users' ability to parse them as sequential elements, making it difficult to interpret in a well-ordered manner. Additionally, any perceived gain in supporting browsing of documents was overwhelmed by the fact that one's spatial map of the general graph view was essentially destroyed and reconstructed to support the clustering mechanism.

When we examine the statement from users above that there is nothing doable on a radial / spatial view of document text that is not already possible on a flat list, we must also reconsider whether or not browsing as a habit can in fact be supported in a flat list. While we still maintain that some kind of spatial layout is more amenable to the *abstract* concept of browsing than a flat list (and in fact may be useful for browsing in domains other than text where the data is not so stringently sequential), perhaps the mental load of maintaining local context in a lower dimension space is markedly less, and given that sequential text has fewer conceptual dimensions than our algorithms assert it to be (at least when we speak of translating these dimensions to a *meaningful* visual mapping), the addition of extra dimensions merely adds more cognitive load onto the browsing aspect, in addition to the overwhelming presence of the semantic network's structure.

However, given that the number of inbound links to a given document was found to be useful by one user, it is possible that some aspects of the network structure may yet be useful when visualized or hinted at to the user. However, a simpler mapping than a clutter of edges may be more useful in visualizing document nodes of high degree, as it would seem that nodes of high degree are interesting not for any particular document that they are linked to, but simply because of the fact that they are so heavily referenced. On the flip side, being able to visualize one or two very strong links between documents may be more helpful than a smattering of moderate or lukewarm links, as when we browse we typically go down one possible avenue at a time rather than attempt to explore many possible links at once; the power of browsing lies in directed suggestion rather than all-out random pathfinding.

7 Conclusion

We have presented an augmentation to Memoplex Browser which attempts to use the semantic structure and clustering of documents in a database to support browsing habits in document perusal. Visually speaking, we have attempted to represent these aspects of the document network in a radial graph layout with actual document previews. Although our attempts to augment the browsing experience beyond simple search and click techniques of a flat list interface were largely unsuccessful, we nevertheless feel that the use of spatial layout to support browsing behaviours through abstract semantic connections may yet be useful in other domains than text where the possible techniques for information browsing are neither constrained by the sequential nature of the data, nor are overwhelmed by the dominance of existing strategies in that domain.

Reference

[1] Byrd, D. A scrollbar-based visualization for document navigation, In Proceedings of the fourth ACM conference on Digital libraries, Berkeley, CA, p.122-129, 1999.

[2] Dumais, S.T. et al. Using latent semantic analysis to improve access to textual information. In Proceedings of the SIGCHI conference on Human factors in computing systems, Washington, DC, p. 281-285, 1988.

[3] Frokjaer, E and Hornbaek, K. Reading of electronic documents: the usability of linear, fisheye, and overview+detail interfaces, In Proceedings of the SIGCHI conference on Human factors in computing systems, Seattle, WA, p.293-300, 2001.

[4] Frokjaer, E and Jakobsen, M. Evaluating a fisheye view of source code. In Proceedings of the SIGCHI conference on Human factors in computing systems, Montreal, QC, p.377-386, 2006.

[5] Furnas, G. W. Generalized fisheye views, In Proceedings of the SIGCHI conference on Human factors in computing systems, Boston, MA, p. 16-23, 1986.

[6] Hearst, M.A. TileBars: visualization of term distribution information in full text information access, In Proceedings of the SIGCHI conference on Human factors in computing systems, Denver, CO, p. 59-66, 1995.

[7] Hinckley, K. and Igarashi, T. Speed-dependent automatic zooming for browsing large documents, In Proceedings of the 13th annual ACM symposium on User interface software and technology, San Diego, CA, p.139-148, 2000.

[8] Huggett, M. Unpublished work.

[9] Lanir, J. Memoplex Browser, CPSC 533 Course Project.

[10] Mander, R., et. al. A “pile” metaphor for supporting casual organization of information. In Proceedings of the SIGCHI conference on Human factors in computing systems, Monterey, CA, p. 627-634, June 1992.

[11] Robertson, G., et. al. Data mountain: using spatial memory for document management. In Proceedings of the 11th annual ACM symposium on User interface software and technology, San Francisco, CA, p. 153-162, November 1998.