OBIWAN - A Visual Interface for Prompted Query Refinement

James W. Cooper and Roy J. Byrd
IBM Thomas J. Watson Research Center, jwcnmr@watson.ibm.com, byrd@watson.ibm.com

Abstract

Typically, users submit very simple search queries to digital document data collections. Often these queries can result in extremely broad answers or answers in which document relevance is hard to assess. Our group has developed a suite of tools which build sophisticated indexes to document collections. These tools can be used to provide cues to help users formulate more effective queries. In order to present these cues and manage the process of query refinement, we have developed a Java-based client server system which uses these indexes and tools.

One of the most powerful parts of our system is its ability to recognize domain-specific multi-word names and terms, using heuristic methods, and to index these vocabulary items in such a way that we can look up vocabulary items that commonly are related to the original query terms. This system responds to an initial query by suggesting additional items that the user can use to focus the query.

1. Introduction

Finding important and relevant documents in an online document collection is becoming increasingly difficult as online document collections and search engines proliferate. The problem is further complicated when the searcher is an infrequent user of a particular search system, but has urgent business or technical needs that he wishes to satisfy without extensive training or experience in the use of these systems and without the use of a search intermediary.

There have been a number of approaches to solving these problems in recent years. For example, [Fowler, Wilson, and Fowler, 1992] have described a multi-window document interface where users can drag terms into search windows and see relationships between terms in a graphical environment. Local feedback was utilized by [Buckley, et al., 1996] and [Xu & Croft, 1996] who also utilized local context analysis using the most frequent 50 terms and 10 two-word phrases from the top ranked documents to perform query expansion. [Schatz. et al., 1996] describe a multi-window interface that offers users access to a variety of published thesauruses and computed term co-occurrence data.

Several groups have addressed the challenge of providing a simple user interface suitable for infrequent use by non-technical users. Envision [Nowell, et al., 1996] is one such interface although it was designed primarily for use in searching technical journal collections. Another approach, termed Scatter/Gather [Hearst & Pedersen, 1996], emphasized clustering of similar documents into several subgroups in response to the initial query. Fireworks [Hendry & Harper, 1996] presents a unique scattered screen environment called SketchTrieve which allows manipulation of queries, documents, authors and results as separate, moveable windows. We have previously reported [Cooper & Byrd, 1997] a web browser-based multi-screen system for navigating through Wall Street Journal articles. In this paper, we describe an extremely simple user interface, suitable for the users of the document collection we describe in Section 3.

When translating an information requirement into a query for a document retrieval system, a user must convert concepts involved in the requirement into query terms which will match terms found in documents in the data base. Creating a successful query against a document collection involves solving the "vocabulary problem" described in [Furnas, et al., 1987]. The problem consists of several interrelated issues, including precision, ambiguity, and synonymy. A query about “IBM Credit Corp.” or “IBM Global Network” is more precise than a query about “IBM,” and will find better documents, assuming that one of the former is part of the user's information requirement. In some document collections, the query term “wireless” could be ambiguous, meaning either “cellular telephone,” or “Motorola's Wireless Data Group.” A more focused query would use one of the non-ambiguous forms. Finally, a query about “on-line analytical processing” would be greatly improved if it also included the synonymous form “OLAP.”

Of course, all of these query improvements rely on knowing what vocabulary items are potentially relevant, because they actually occur in the document collection. They also rely on it being easy for the user to find them when they are needed for improving a particular query. The Talent tools described in the next section help us to

meet the first of these two prerequisites. The user interface devices described in Section 3 address the second.

2. Talent Tools

We have developed a suite of text analysis tools, called "Talent" (for "Text Analysis and Language ENgineering Tools"), which allow us to use the text in a document collection to build lexical resources suitable for solving the vocabulary problem when querying that collection. Before queries begin, the Talent tools are used to extract domain-specific vocabularies, "context thesauruses," and relationships from all the documents in the collection. Further tools are used to organize this material into "lexical networks." Here, we describe these lexical resources in more detail.

Vocabularies

Talent vocabularies consist of "vocabulary items." Each item comprises a "canonical form" and zero or more additional "variant forms." Depending on the item, the canonical form is the form which is the least ambiguous, the most explicit, the longest, the lemma, etc., among all of an item's forms. In addition, Talent contains facilities for recognizing acronyms and abbreviations and for including them among the variants of their full forms. Vocabulary items have a category associated with them, indicating whether the item is an ordinary word, a domain term, or a name denoting a person, place, organization, etc.

For the prompted query refinement application (among others), vocabulary items also contain occurrence statistics. An important statistic, described by [Prager, 1994] is the "information quotient" (or IQ), which characterizes an items distinctiveness within the document collection. Specifically, a vocabulary item's IQ, expressed as a number between 0 and 100, gives an indication of how effective the item would be at distinguishing among documents when the item is used as a query term. We often find it useful to interpret a vocabulary item's IQ as a measure of its "importance" within the domain.

Context Thesauruses

A context thesaurus is a device which, given a probe consisting of a single word, a query, or an entire document, will produce a ranked list of vocabulary items that are related to the probe. The information used to accomplish this is basically information about the co-occurrence of vocabulary items in the sentences and paragraphs of the document collection. Specifically, using an idea inspired by the "phrase finder" procedures described in [Jing and Croft, 1994], the context thesaurus consists of a an ordinary information retrieval system's document index. The key innovation is that the documents are "pseudo documents" derived from the original document collection. Thus, there is one pseudo document for each vocabulary item and it contains contexts in which the item occurs within the collection. In the prompted query refinement system, the IBM NetQuestion query system [IBM, 1997] is used to index and query the pseudo document collection.

Unnamed Relations

Our view of unnamed relations is based on the fact that vocabulary items which often appear together are likely to be related. Furthermore, the frequency with which two items appear together can be used as the basis for a measure of the strength of the relationship. The current implementation of our unnamed relation extractor uses the context thesaurus. For a pair of vocabulary items, we determine the extent to which one item occurs in the context of the other by querying the context thesaurus for each item. When a query to the context thesaurus is a vocabulary item, the rank of a vocabulary item on the resulting hitlist can be interpreted as a numerical value of the strength of association between the query item and the hitlist item. If each item appears on the other's hitlist with a rank that exceeds some threshold and if the ranks are fairly close, we assume that there is a symmetric relationship between the two items. The strength of that relationship is computed from the ranks assigned by the context thesaurus.

It is also worth noting that the context thesaurus itself implicitly contains asymmetric unnamed relations. A relationship is symmetric if each of two vocabulary items appears on the other's context thesaurus hitlist. It is asymmetric if only one of them appears on the other's hitlist. As an example, if the name of an arbitrary employee of IBM were used as a context thesaurus query, the resulting hitlist is likely to contain "International Business Machines". The converse is mostly likely not true, except for a handful of names such as "Louis V. Gerstner." Thus, there would be a symmetric relationship between Gerstner and IBM but only asymmetric relations between IBM and most other employees. Of course, the distinction between symmetric and asymmetric relations is a fluid one and

depends on the text in the document collection and the thresholds set during the extraction process.

Named Relations

Talent extracts named relations between vocabulary items by exploiting grammatical and orthographic constructions in which such relationships are typically expressed in English. These constructions include appositives, possessives, parentheticals, copular clauses, coordination, etc. The named relation extractor operates in "discovery mode." That is, it extracts not only the related vocabulary items but also the names of the relationships from the text, based on the positions that they occupy within the defining text constructions. For this reason, the relationships, as well as the vocabulary items, are domain-specific.

Lexical Networks

In our current prompted query refinement application, named and unnamed relations are combined into a series of database tables comprising a "lexical network." In lexical networks, vocabulary items are the nodes and relations are the links. A lexical network is, in effect, the repository of most of the material extracted from a document collection by Talent tools. They are used within the query refinement server when the user interface client requires lexical information for navigation and refinement activities.

When vocabulary items and relationships among them are used in applications, it is often natural and useful to ignore the distinction between the (symbolic) items themselves and the real-world concepts to which they refer. This is certainly the case with prompted query refinement. Although it is only symbolic information that is extracted from the documents, maintained in our lexical resources, and presented at the user interface, the value of that information is that it elicits query modifications from the user based on his or her grasp of conceptual relationships in the real world. The usefulness of our tools and interfaces derives from their ability to inform or remind the user of conceptual links relevant to the document collection being queried and to the current query.

3. The OBIWAN User Interface

The Giga Information Group Document Collection

We undertook this research as a joint effort with the Giga Information Group, a provider of Information Technology (IT) reports for businesses. Their information delivery model is to use web pages to deliver major reports (called Planning Assumptions) and shorter reports, (called Customer Queries) as web pages which are posted to their site on a daily basis. In addition, they provide selected press releases and articles from the Dow Jones News service on IT companies. The snapshot of the data we used for this study consisted of 700 Planning Assumptions, 5400 Customer Queries, and about 5000 press releases and 5000 news articles. The overall document collection contained about 140 megabytes of text.

The problem is, of course, that this constantly changing information eludes simple attempts to organize and categorize it and unexpected connections between these reports may eventually develop since they are written by a number of consultants. Therefore, when a customer logs into the Giga service and simply searches the collection for information on a given keyword or topic, he may miss useful, related information that a simple search does not provide. Further, as the document collection grows, it is much more difficult to find the most useful documents without actually reading large numbers of them.

Observations of large numbers of users working with search engines reveal that most new or inexperienced users tend to compose queries made up of one or two words, rather than using qualifying terms that might help limit or focus the query. Studies by the Prodigy on-line service revealed that most queries average 1.8 words and that virtually no one seeks to use either the "advanced" query formulation or looks at the help screens. In addition, according to [Selzer, et al., 1997] the average lengths of English queries to the AltaVista system range from 1.8 words for queries about places and travel to 3.7 words for queries about world news events.

Moreover, since every search engine has unique characteristics, infrequent users are much less likely to know how to use the sophisticated features of these engines. Many of the users of the Giga information service are business people rather than technical experts, hence they are less likely to have time to learn the intricacies of constructing multiple term queries with Boolean qualifiers.

Thus, we set out to design a simple "one button" user interface, named OBIWAN (for "One Button Interface

With Associated Network"), that would suggest additional query terms based on the lexical network, and make it easy to use these terms to refine the query. While the main interface panel actually has 3 buttons (**Search**, **Clear**, and **View**), the intent of the interface is preserved: the **Search** button performs different search functions depending on the information which is currently displayed.

Construction of the Data Files

We constructed an index of the more than 16,000 documents using the IBM NetQuestion document search and retrieval system. Next, we constructed a context thesaurus for each of the four document collections and combined them into a single context thesaurus, consisting of one NetQuestion search index. We used a standard relational database to store the lexical network, consisting of vocabulary items together with named and unnamed relationships among them. Finally, we constructed an "items database," also using a relational database, containing vocabulary items, documents, and information about which items occur in which documents. For convenience, we also included the title and abstract for each document in the Giga collection, in order to make it easier to display search results.

Design of the User Interface

In Figure 1, we illustrate the basic OBIWAN navigation display. Here, the user can enter a search word or phrase and click once on the **Search** button. This, then, displays items, ranked by IQ, which are actually found in the collection as the labels of check boxes. Note that since the Context Thesaurus indexes the text found near all of the items it identifies, the user does not need to enter an exact match for a item in order for the thesaurus to suggest useful related items.

Note further that nearly all of these are multi-word items, which usually rank much higher by IQ than single word items.

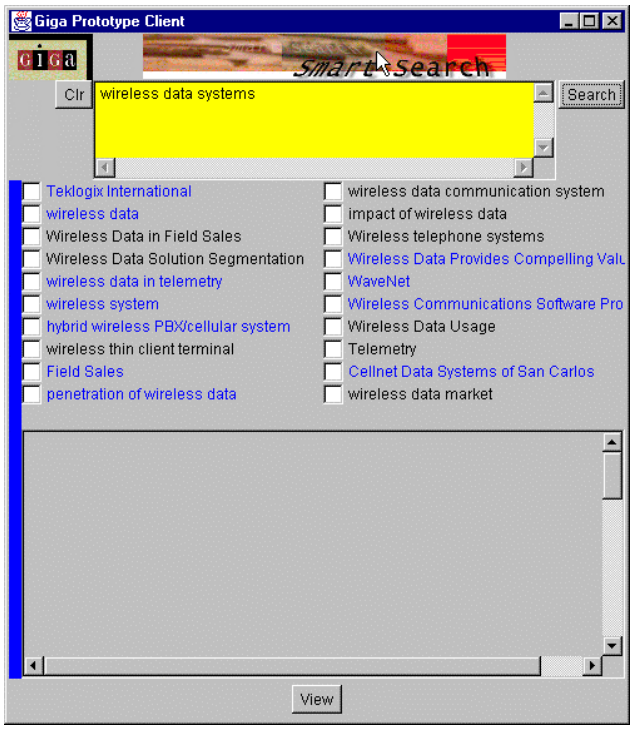

Figure 1. Items suggested by the Context Thesaurus after clicking on the Search **button.**

Clicking on a **Search** button a second time causes the program to search the collection using the same query. The titles of the top 5 documents that were found are shown at the bottom of Figure 2.

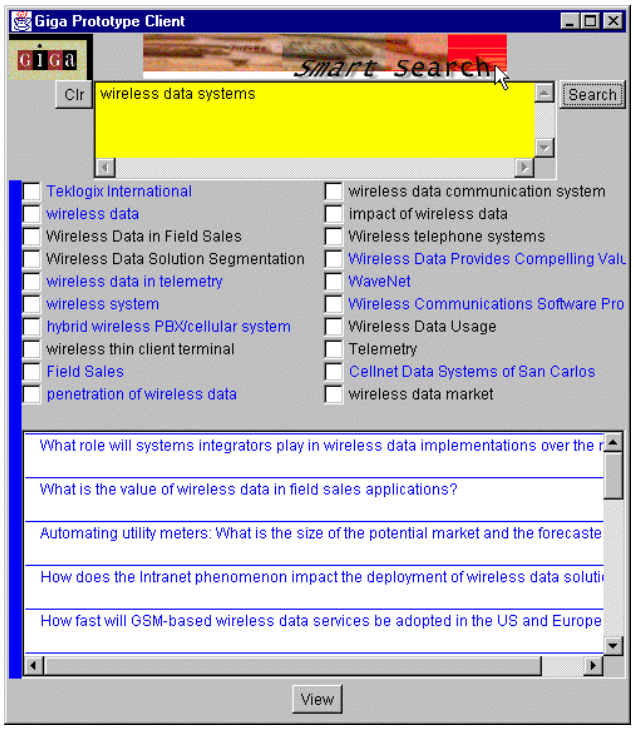

Figure 2. Top documents selected by searching the collection using just "Wireless data access."

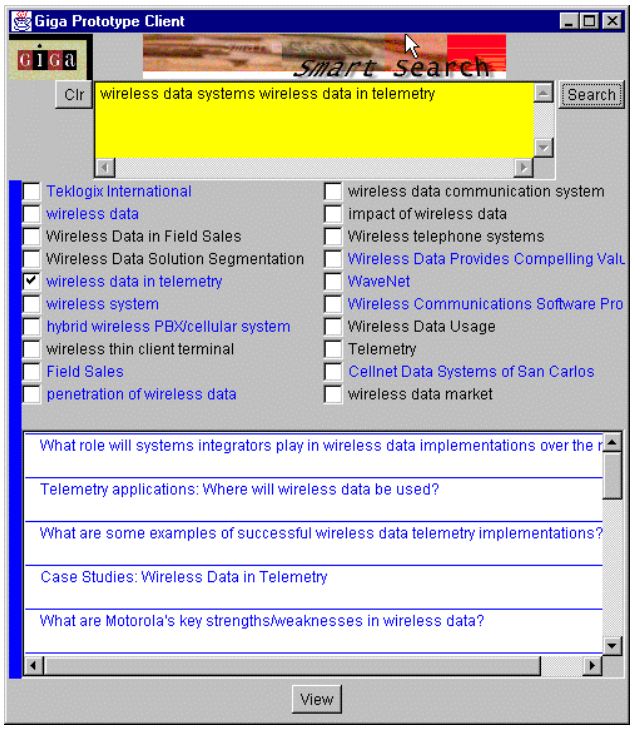

Figure 3. Top documents selected by enhancing the original query with the checked item.

The user can carry out a more focused search, however, if he selects one or more of the items suggested by the Context Thesaurus. As each of them is checked, it is added to the query field at the top of the form, and any un-checked items are removed from the query field. Then, if the user clicks on **Search** a second time, the program searches using this refined query, and in general produces considerably improved results. This is shown in Figure 3.

While OBIWAN has a superficial resemblance to the Visual Live Topics display provided by the AltaVista search service, see, for example, [DeJesus, 1997] we note that that technology provides only single words and not necessarily those in the current collection, while we index and suggest vocabulary items consisting of single- and multi-word phrases which actually exist in the collection.

The user can also narrow the scope of the query terms suggested further by adding one or more items to the query field and having OBIWAN recalculate the neighboring items. For example, we added "Communications product" to the search text and searched for new co-occurring items. The markedly different result is shown in Figure 4.

Figure 4. A new set of Context Thesaurus items generated when "communications product" is added to the query.

Related Items

Vocabulary items which have unnamed or named relations to other items are represented in blue in the checkbox list. Right-clicking on any such blue-highlighted item (shown in Figures 2 through 4 as lighter gray) displays a list of surrounding items that are closely related, as shown in Figure 5.

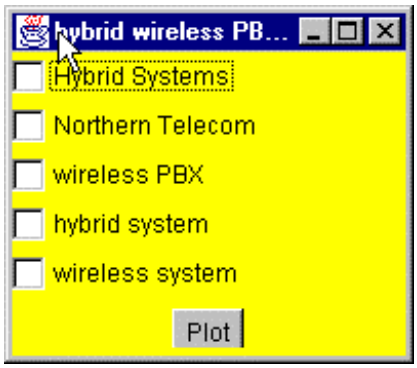

Figure 5. A list of items closely related to "hybrid wireless PBX."

Then the user can click on any of these items as well and add them to his query. In order to make this window as easy to use as possible, it pops up if he clicks the right mouse button anywhere on a blue-highlighted checkbox, and stays up as long as the mouse hovers over it. It disappears as soon as the mouse moves out of the window.

Graphical Relations Between Items

It is also possible to view the relationships between items graphically as a network of named and unnamed relations as shown in Figure 6 [Tunkelang, Byrd and Cooper 1997]. The user can also select any item and add it to the query using the **Add** button.

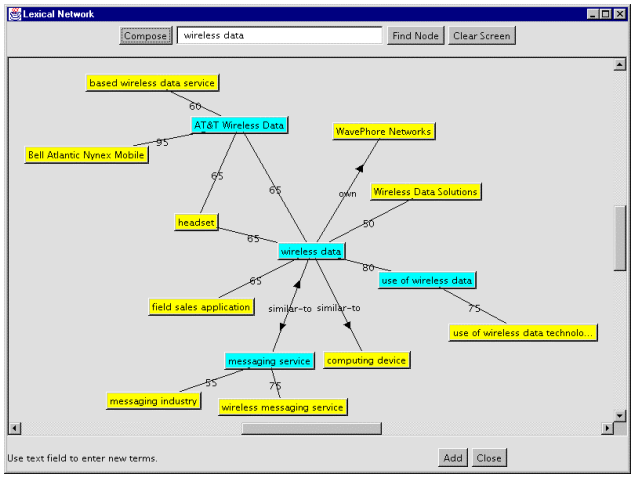

Figure 6. A plot showing named and unnamed relationships around "wireless data" graphically.

Many of the items in this display can be further expanded by double clicking on them to show additional, more distant relationships. This provides a way of navigating through the lexical network in a way which is difficult to represent in any other fashion.

The Items Database

The items database is a group of three tables, one of document filenames, one of vocabulary items and one of the document keys versus the item keys. Further, the item relations are encoded as a table of relation keys and a table of relation names. From this database, we can easily construct a table of all of the items that were found in a document, even though the items are stored by canonical form, and display them sorted into any useful order. In our user interface, we allow the user to right-click on any document title and see a display of the names and domain terms found in that document. Here, also, the checkbox provides a way to add items to the original query, in a type of user-controlled relevance feedback. This is illustrated in Figure 7.

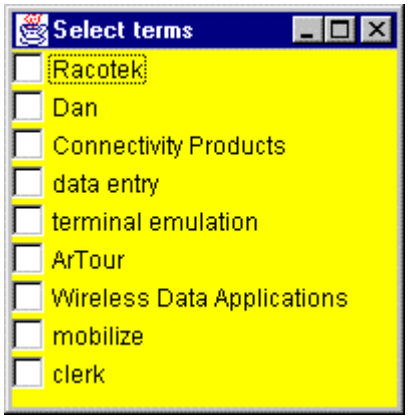

Figure 7. The top vocabulary items found in a document titled "What is the danger in using terminal emulation in wireless data projects?"

We quickly note that the items shown in Figure 7 provide an extremely concise summary of the contents of that document. These items provide a good summary partly because they are ranked by IQ, and thus are some of the most selective items for that particular document. Adding any selected items from this list to the initial query will find more documents "like this one," a common requirement in document retrieval systems.

Experiments in Representing Lexical Neighborhoods

Using the items database, OBIWAN can allow the user to find all of the documents containing an individual item. We refer to this document cluster as the item's "lexical neighborhood." Further, if an item can be found in a large number of documents, it is possible to find the intersection of several sets of items to reduce the total number of documents to only those documents which contain all of the items of interest. This is, of course, just

a Boolean conjunction of the documents containing the selected items.

It is interesting to note that if we simply display the most selective (Top IQ) items in this fashion, they will be of limited use, since they by definition occur in very few documents. Instead, we must display items that we know occur in several documents in order to determine an intersection of documents containing several such items .

Figure 8 shows the first step in this navigation process, in a slightly modified OBIWAN interface. First, the user enters a simple search phrase, here "Java." Then he clicks on **Search** to find all the items in the collection which contain that string. These are displayed in the left-hand list box. Clicking on any phrase in that list box produces a list of titles of documents which contain the phrase.

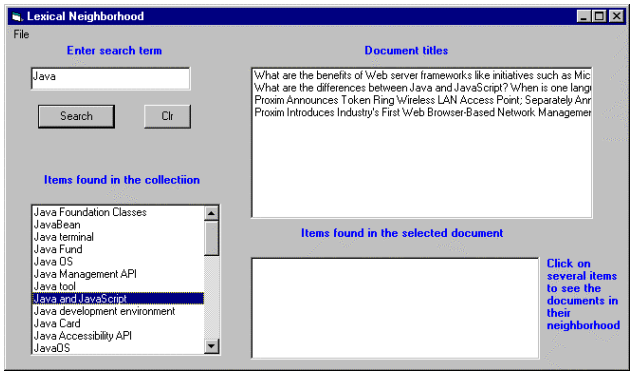

Figure 8. An interface for selecting documents on the basis of the items they contain. The selected phrase "Java and JavaScript" appears in the 4 documents shown.

Now, if the user clicks on any one of these documents, he can see the items it contains as shown in Figure 9.

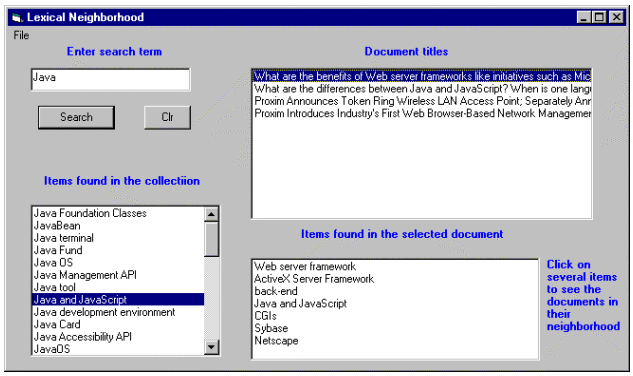

Figure 9. The items contained in the top, highlighted document are show in the bottom right list box.

Finally, the user can see which documents contain any combination of items by clicking on them. In Figure 10, the user has selected "Sybase" and "Netscape." The document display list then changes to show those documents which contain *both* of these items. This provides a powerful way to navigate through the concept space surrounding the collection by viewing the lexical neighborhood of the intersection of several items of interest.

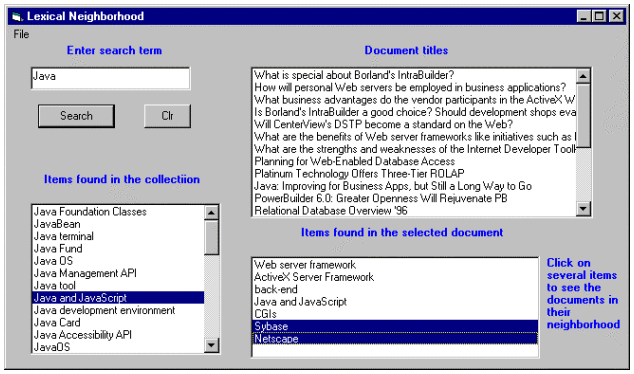

Figure 10. Documents in the upper list box contain both "Sybase" and "Netscape."

Construction of the OBIWAN Program

The program is written in Java 1.1 [Arnold and Gosling, 1996] as a stand-alone client application running on Windows 95 and a server running on Windows NT. The server consists of a Java 1.1 program whose methods are called using Java Remote Method Invocation [Cooper, 1997; Orfali and Harkney, 1996]. The server, in turn, makes queries to one or more of 4 separate indexes:

1. The NetQuestion document index
2. The Context Thesaurus
3. The Lexical Network
4. The Items Database

The server accesses the NetQuestion search engine using Java to spawn an external query process and receive the results through the standard-out port. The search of the context thesaurus amounts to a search of a second NetQuestion index. The server uses JDBC, Java's database connectivity system and the JDBC-ODBC Bridge [Cooper, 1997] to access the lexical network and the items database. A Java object is returned for each resulting item.

4. Results and Discussion

Testing of this methodology and of the user interface has been carried out primarily as a by-product of testing of the technology. We have obtained both comments about usability and suggestions fo enhancing search flexibility.

Anecdotally, in about 80% of the cases where a user has made an elementary query to OBIWAN, he has improved its accuracy by including one or more of the items suggested by the context thesaurus. Since the items which are shown actually occur in the collection, they help the user locate actual documents on the subject he has selected. Since the items are discovered automatically, they are not perfect, and some of them must be ignored by the user. Our experience has been that users act as extremely good filters for these "noise" items and do not find a few of them objectionable as long as some are useful and enhance their ability to locate documents of interest.

Users reported that the context thesaurus allowed them to navigate through metadata associated with the query and allowed them to learn about the subject without reading any of the articles. They noted that the suggested items were intrinsically interesting, beyond any role they played in query refinement.

In another test associated with TREC-6, we observed that users expected that the items proposed by the Context Thesaurus be closely related to the original query. This, of course, depends on whether such terms exist in the collection that was indexed. We also observed that as users became more familiar with the system, they tended to pose longer, more conceptual queries which were then not always easily resolved. For example, when trying to discover information about ferry accidents, they would pose a query of the form "ferries that have sunk killing more than 100 people." Processing of such conceptual queries is the subject of further work.

In conclusion, one the main difficulties in gleaning information from an ever growing collection is the necessity for the user to read large numbers of documents to find out if they are relevant to his interests. The OBIWAN system obviates this requirement by presenting the user with keywords to narrow the search to useful, existing documents and by allowing him to view the keywords in a document directly.

Finally, the ability to select peripherally related keywords allows the user to explore new avenues of interest, based purely on the occurrence of these keywords, thus expanding his technical knowledge without going through the undirected browsing and searching that "web surfing" usually requires.

Applications to Other Media

Clearly, the techniques we describe here are intended to enhance the searching of text collections. Work in other groups on categorizing video data from news reports indicates that most of the same techniques would apply there, as well. Such reports contain large quantities of descriptive text or other metadata produced, for example, by closed captioning, which can be indexed just as we describe here.

In addition, when documents contain hypertext links to multi-media objects, the text surrounding the links usually contains descriptions of those objects. We intend to experiment with our techniques to determine whether these descriptions may be exploited to enhance search for multi-media objects.

Future Work

We have applied categorization techniques based on nearest-neighbor classifiers [Duda and Hart, 1973; Stanfill and Waltz, 1986] to map a reduced set of existing Giga-assigned categories to new and old documents, thus providing the ability to suggest related categories that users might look into for information of interest. Incorporating these categories as well as the lexical neighborhood approach into a single interface will be the subject of further study.

We are also experimenting with using data mining techniques to analyze vocabulary item cooccurrence data, as an alternative to using the context thesaurus, for deriving unnamed relations.

Acknowledgments

We'd like to acknowledge the continuing, helpful cooperation of Bela Labovitch, Susan Funke and Rhoda Nafziger of the Giga Information Group, and Andrew Singleton and Tom Laramee of Cambridge Interactive. We are also deeply indebted to our talented colleagues at IBM Research. They include Mary Neff, Yael Ravin and Misook Choi, who built the vocabulary extractors, Herb Chong and Aaron Kershenbaum, who originated many of our dictionary subsystems, Daniel Tunkelang, who designed and built the graphical layout algorithms with suggestions from Mark Wegman, Birgit Schmidt-Wesche, who made a number of usability suggestions, and Alan Marwick, who inspired us to turn our ideas into a working system.

References

Apte, C., Damerau, F. and Weiss, S.M. Automated learning of decision rules for text categorization. ACM Transactions on Information Systems,12(3) 223-251, 1994.

Arnold, Ken and Gosling, James, *The Java Programming Language*, Addison-Wesley, 1996.

Bates, Marcia J. “Human, Database, and Domain Factors in Content Indexing and Access to Digital Libraries and the Internet,” Allerton, 1996.

Brajnik, G., S. Mizzaro, and C. Tasso "Evaluating User Interfaces to Information Retrieval Systems: A Case Study on User Support" in Proceedings of the 19th Annual ACM-SIGIR Conference, 1996, pp. 128-136.

Buckley, C., Singhal, A., Mira, M & Salton, G. (1996) “New Retrieval Approaches Using SMART:TREC4. In Harman, D, editor, Proceedings of the TREC 4 Conference, National Institute of Standards and Technology Special Publication.

Byrd, R. J., Ravin, Y., and Prager, J. "Lexical Assistance at the Information-Retrieval User Interface," Proceedings of the SDAIR (Symposium on Data Analysis and Information Retrieval), UNLV, 1995.

Duda, R. O, and Hart, P.E. *Pattern Classification and Scene Analysis*, John Wiley and Sons, New York, 1973.

Cooper, James W., *Principles of Object Oriented Programming Using Java 1.1*, Ventana, 1997.

Cooper, James W. and Byrd, Roy J. “Lexical Navigation: Visually Prompted Query Expansion and Refinement.” Proceedings of DIGLIB97, Philadelphia, PA, July, 1997.

DeJesus, Edmund X., “The Searchable Kingdom,” Byte June, 1997, 92NA.

Fowler, Richard H., Wilson, Bradley A., and Fowler, Wendy A.L. “Information Navigator: An information system using networks for display and retrieval.” Report NAG9-551, No.92-1. Department of Computer Science, University of Texas, Pan American, Edinburg, TX.

Furnas, G. W., T, K., Landauer, L. M. Gomez, and S. T. Dumais “The Vocabulary Problem in Human-System Communication,” in *Communications of the ACM*, vol. 30, no. 11, November 1987, pp. 964-971.

Hearst, Marti A. and Pedersen, Jan O., “Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results,” Proceedings of the 19th Annual ACM-SIGIR Conference, 1996, pp. 76-84.

Hendry, David G. and Harper, David J., “An Architecture for Implementing Extensible Information Seeking Environments,” Proceedings of the 19th Annual ACM-SIGIR Conference, 1996, pp. 94-100.

IBM. A description of IBM’s NetQuestion text search and retrieval technology, on the World-Wide Web at http://www.software.ibm.com/data/mediaminer, 1997.

Jing, Y. and W. B. Croft “An association thesaurus for information retrieval,” in Proceedings of RIAO 94, 1994, pp. 146-160.

Orfali, Robert and Harkney, Dan. *Client/Server Programming with Java and Corba*. John Wiley and Sons, New York, 1996.

Maarek, Y.S., “Software Library Construction from an IR perspective,” in SIGIR Forum, fall 1991, 25:2, 8-18.

Nowell, Lucy Terry, France, Robert K, Hix, Deborah, Heath, Lenwood S., and Fox, Edward A. “Visualizing Search Results: Some Alternatives to Query-Document Similarity.” Proceedings of the 19th Annual ACM-SIGIR Conference, 1996, pp. 67-75.

Prager, John. IBM Research. Private communication, 1994.

Ravin, Y. “Disambiguating Proper Names in Text,” in Proceedings of MIDDIM 96, International Seminar on Multimodal Interactive Disambiguation, Col de Porte, August 1996.

Ravin, Y. and Wacholder, N. 1996, “Extracting Names from Natural-Language Text,” IBM Research Report 20338.

Schatz, Bruce R, Johnson, Eric H., Cochrane, Pauline A and Chen, Hsinchun, “Interactive Term Suggestion for Users of Digital Libraries.” ACM Digital Library Conference, 1996.

Seltzer, Richard, Ray, Eric J., and Ray, Deborah S. *The AltaVista Search Revolution.*, Osborne-McGraw Hill, 1997.

Spink A, A. Goodrum, D. Robins, and M. M. Wu "Elicitations During Information Retrieval: Implications for IR System Design" in Proceedings of the 19th Annual ACM-SIGIR Conference, 1996, pp. 120-127.

Spink, A. "Term Relevance Feedback and Query Expansion: Relation to Design" in Proceedings of the 17th Annual ACM-SIGIR Conference, 1994, pp. 81-90.

Stanfill, C. and Waltz, D. “Toward Memory-based Reasoning,” *Communications of the ACM*, 29(12) 1213-1228, 1996.

Tunkelang, D. D. "A Practical Approach to Drawing Undirected Graphs", Technical Report CMU-CS-94-161, Carnegie Mellon University, June 1994.

Tunkelang, D. D., Byrd, R. J., and Cooper, J. W., “Lexical Navigation: Using Incremental Graph Drawing for Query Refinement,” Graph Drawing 97.

Xu, Jinxi and Croft, W. Bruce. “Query Expansion Using Local and Global Document Analysis,” Proceedings of the 19th Annual ACM-SIGIR Conference, 1996, pp. 4-11