RJ 9816 (84879) May 16, 1994
Computer Science

Research Report

FINDING WHAT I AM LOOKING FOR: AN INFORMATION RETRIEVAL AGENT

R. C. Barrett
E. J. Selker

IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

IBM Research Division
Yorktown Heights, New York ■ San Jose, California ■ Zurich, Switzerland

RJ 9816 (84879) May 16, 1994
Computer Science

Finding What I'm Looking For: an information retrieval agent

R.C. Barrett and E.J. Selker
IBM Almaden Research Center
650 Harry Rd.
San Jose, CA 95120

Abstract

We are developing a user-interface agent which is aimed at aiding an information searcher. Through the methods of relevance feedback, we have implemented an information retrieval system which works over a range of real repositories of publication text. The primary contribution concerns a scenario supporting a conversational style of user-interface which flows well with the searcher's normal method of working, providing suggestions for refining the search. We have found that these non-intrusive suggestions typically reduce the search space by a third without degrading the quality of the search. The system runs smoothly on a low-powered workstation in a client-server arrangement. In this paper, we present the background of our design, the details of the relevance algorithms, and the results we have obtained. We also discuss our directions of future work as we study the problem of guiding an information searcher to the desired data.

keywords

information retrieval
relevance feedback
query refinement
Boolean query
user model
user-interface agent

Introduction

The information explosion is all around us: the Information Superhighway, Internet, UseNet, WAIS, World-Wide Web, Gopher, etc., etc., etc. The information we are looking for is buried in gigabytes of data that has no interest to us. How do we find the things that are important to us? We are left to sift through these databases, searching for the quality material that answers our questions. This sort of browsing can be valuable, as there are many jewels waiting to be discovered. But in many cases we wish to focus on one topic and look for some particular information. The purpose of this work is to help make the search for information more efficient and more rewarding.

How can we provide the best environment for a user to control the machine and for the machine to respond to the user? An interface is defined by Webster as "a point at which independent systems interact." The world of information retrieval is a classic computer-human interface problem. Here we have two independent systems: a searcher with an information goal, and a computer with information. The user-interface question is how to most efficiently communicate between the searcher and the information source.

Information Retrieval

The classical method for connecting the searcher and the source is a batch model. It has a repeated command-response structure. The searcher issues a search command and the computer replies with the results of the search. The searcher then examines the results, thinks about the correlation between the search and the results, creates an improved query, and executes the new search. This process is continued until a satisfactory result is obtained. A notable feature of this method is that the computer does not have any information about the searcher's reactions. Each search command is considered to be independent and the search results are assumed to be the correct answer to the request. No allowance is made for the searcher's difficulty in producing a suitable query. The system reacts to the searcher's commands but does nothing to proactively help formulate the search.

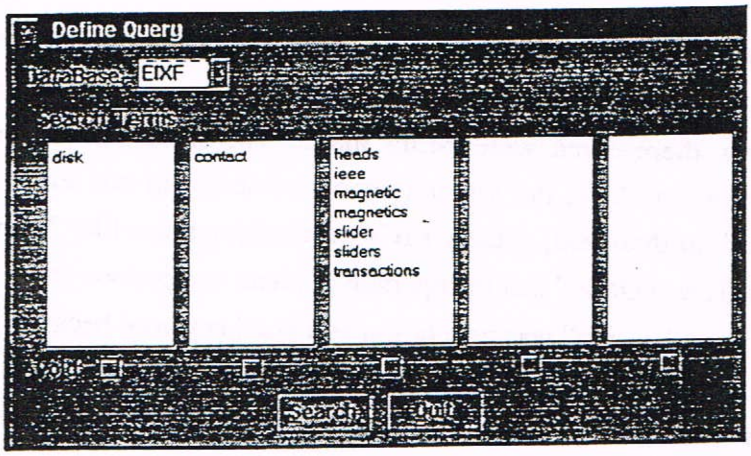
An alternative search methodology is one in which the user is in conversation with a user-interface agent which aids in generating an appropriate search command [1,2]. Instead of the searcher having all of the responsibility for formulating the next query, an agent provides suggestions. Specifically, when the search results are returned, the searcher provides feedback by marking certain articles as relevant and others as irrelevant to the search being performed. Given this with a window as shown in Figure 1. The boxes are used to enter search terms. Like terms are entered in the same box and unlike terms are entered in separate boxes. Checkboxes are included to mark a certain box as containing "terms to avoid". This structure is much more intuitive than the usual Boolean AND, OR, and NOT syntax. In the example shown in the figure, the searcher is looking for information on tape library systems, but is not interested in helical-scan formats. So the fundamental search terms are `tape`, `library`, and `helical`. In order to broaden the search, synonyms have been included for `library` and `helical`. Finally, the helical search category has been negated by checking the "Avoid" box. We have found that searchers easily understand this type of query definition, without needing to understand Boolean logic. Of course, this search is seen by the machine as: `tape AND (library OR automation OR loader) NOT (helical OR rotary OR DAT)`, but this form is never seen by the searcher.

Once the initial query has been entered, it is sent to the database search engine. This engine then returns a list of titles which fit the query. The titles are then presented to the searcher on the left-hand side of a two-paged window (see Figure 2). As the searcher browses the titles and abstracts, she can mark them as relevant or trash. The right-hand side of the page accumulates the relevant articles. As the searcher marks the titles and abstracts, the agent suggests new search terms. These new terms are presented in the same form as the searcher entered their initial query. Having them in the same form provides a semantic link for the user. When the searcher is finished reviewing the results, she can return to the query window where the suggested search terms are automatically added to the original query. If the suggestions are acceptable then the modified search can be performed immediately. Otherwise the suggestions can be edited to the searcher's liking. This modified search is then sent back to the search engine. It should be noted that articles which have already been marked as "relevant" or "trash" are caught by the agent on subsequent searches. They are automatically discarded or moved to the relevant list so that they do not distract from the search. This mechanism saves the searcher from having to re-read these articles. It also makes the relevance information immediately available to the system for future query refinement.

Thus, the search scenario consists of the searcher responding to found articles and suggested search terms. The searcher and agent act cooperatively to refine the search.

Results

To demonstrate this adaptive interface technology for real-sized problems, we used the IBM Technical Information Retrieval Center consisting of 28 repositories which represent "the published information databases of the world"[9]. The databases include the INSPEC science



DataBase: EIXF

Search Terms:

disk

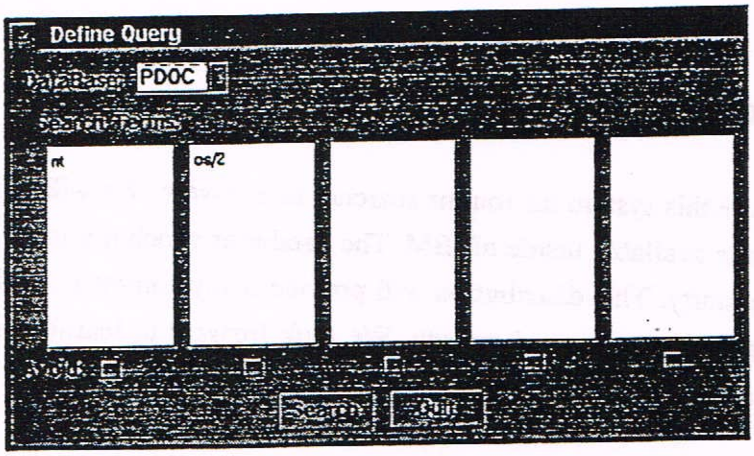
contact

heads
ieee
magnetic
magnetics
slider
sliders
transactions

Avoid: □ □ □ □ □

These terms indicate several different features that the system identified. First, it identified disk drive components which differentiate disk drives from other types of "disks". These components are "heads" and "sliders". It also identified a subject area that goes along with disk drives, namely "magnetic". Finally, it also identified a journal which tends to carry the desired information: IEEE Transactions on Magnetics. Including these search terms as the system recommended reduced the number of articles found by 50% and improved the quality of the search so that 9 out of the first 10 articles found were relevant (versus 3 out of 10 for the initial query). This example illustrates one of the most powerful applications of relevance feedback, namely narrowing an overly broad search to one of several subfields. This entire scenario took about 5 minutes.

In another example, a searcher desired to find articles which compared the OS/2 and Windows NT operating systems. The initial query was:



DataBase: PDOC

Search Terms:

nt

os/2

Avoid: □ □ □ □ □

After marking several articles, the relevance module suggested several new search terms. These included "operating" and NOT "announcement". It was observed that the retrieved articles were of basically two classes: software products that would run under both Windows NT and OS/2 (trash), articles by the same author will provide a good tool for looking beyond the initial query. The user-interface will have to be expanded to include author information. More general query broadening would also be helpful. Our current paradigm uses the relevance information to narrow the initial search. But sometimes the initial search has been too narrow, eliminating other useful articles. The challenge lies in teaching the system about the breadth of articles in the database so that the search is not broadened too far. This idea lies in the field of machine learning with only positive examples.

We are also looking at expanding the system so that it will automatically search for new information as it appears in the databases. By collecting relevance information about these continuously-gathered articles, the system can begin to describe a user model. This model seeks to answer the question "what is my user interested in?".

Where to look is as important as what to look for. In its current incarnation, the searcher has some 28 databases from which to choose. Even this number can be overwhelming and can be difficult for the searcher. Are the poor search results due to an improper query or is it the wrong database? The problem grows tremendously as we consider the thousands of smaller, more specialized databases on the Internet. Where do we begin to look? We are currently developing a multi-dimensional information taxonomy which can help describe the type of information in a given database. Much of the database analysis can be performed automatically, generating database profiles. These profiles can then be matched to the searcher's needs. Examples of different dimensions in this taxonomy are subject, timeliness (news vs. encyclopedia), reading level, and exposition vs. discussion. This sort of analysis offers the possibility of guiding the searcher to the information she desires without the "luck-of-the-draw" which so pervades the Internet today.

Appendix: Relevance Algorithm

The core of our search system is the query improvement engine. This module is given the task of improving a query based on the relevance information provided by the searcher. Once the various articles have been marked by the searcher as "relevant" or "trash", this module begins to search for a Boolean query which will differentiate between the two sets. No *a priori* information about the format of the text is assumed, making all of the words of the text equivalent. This assumption makes the system applicable to any text, rather than limiting it to known formats.

As each article is marked, its words are added to an inverted index which lists the articles which contain any given word. Forty common words such as "and" and "the" are automatically

References

[1] Selker, T. *A Framework for Proactive Interactive Adaptive Computer Help*. PhD thesis, Computer Science Department, City University of New York, NY, NY, 1992.

[2] Maes, P., and Kozierok, R. Learning interface agents. In *Proceedings of AAAI-93* (1993), AAAI Press, Menlo Park, CA, pp. 459-464.

[3] Dennis, S. F. The design and testing of a fully automatic indexing-searching system for documents consisting of expository text. In E. Schecter (Ed.), *Information retrieval – a critical view*, Washington, D.C.; Thompson Book Co. (1967).

[4] Salton, G., and Buckley, C. Improving Retrieval Performance by Relevance Feedback. *Journal of the American Society for Information Science*, 41 (1990), 288-297.

[5] Frants, V. I., and Shapiro, J. Control and Feedback in a Documentary Information Retrieval System. *Journal of the American Society for Information Science*, 42 (1991), 623-634.

[6] Aalbersberg, I. J. Incremental relevance feedback. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval 15* (1992), 11-22.

[7] Nickerson, G. Getting to know wide area information servers. *Computers in Libraries*, 12 (1992), 53-55.

[8] Verhoeff, J., Goffman, W., and Belzer, J. Inefficiency of the use of the boolean functions for information retrieval systems. *Communications of the ACM*, 4 (1961), 557-558, 594.

[9] IBM Technical Information Retrieval Center, IBM publication #ZZ38-3001-06 (1993).

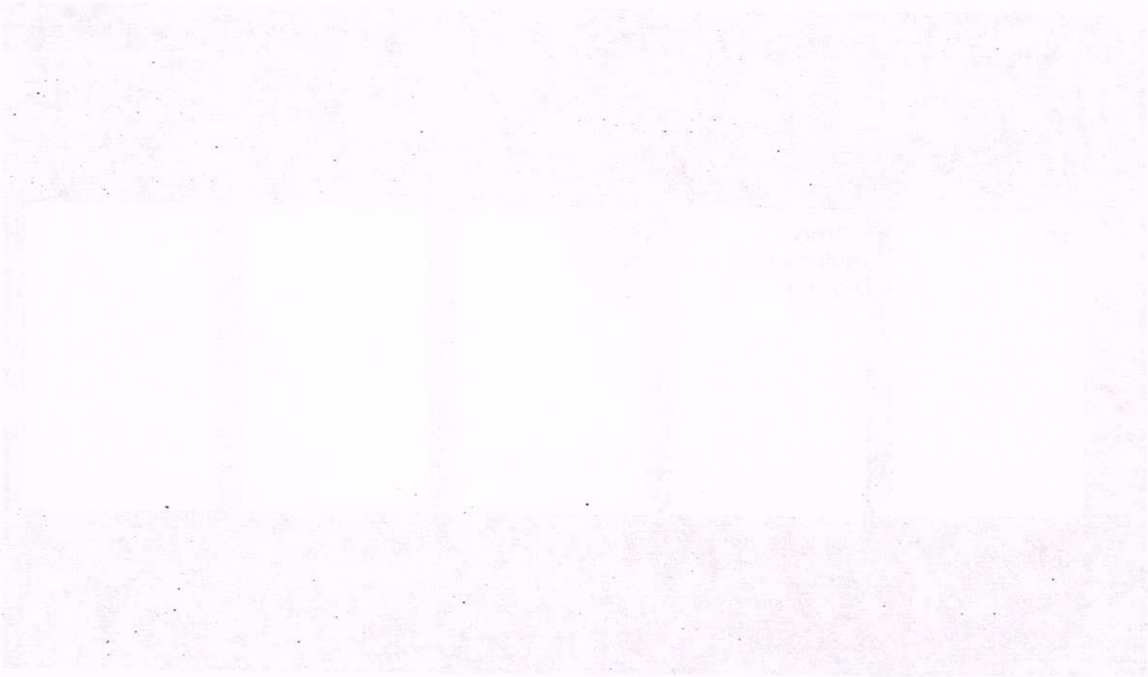
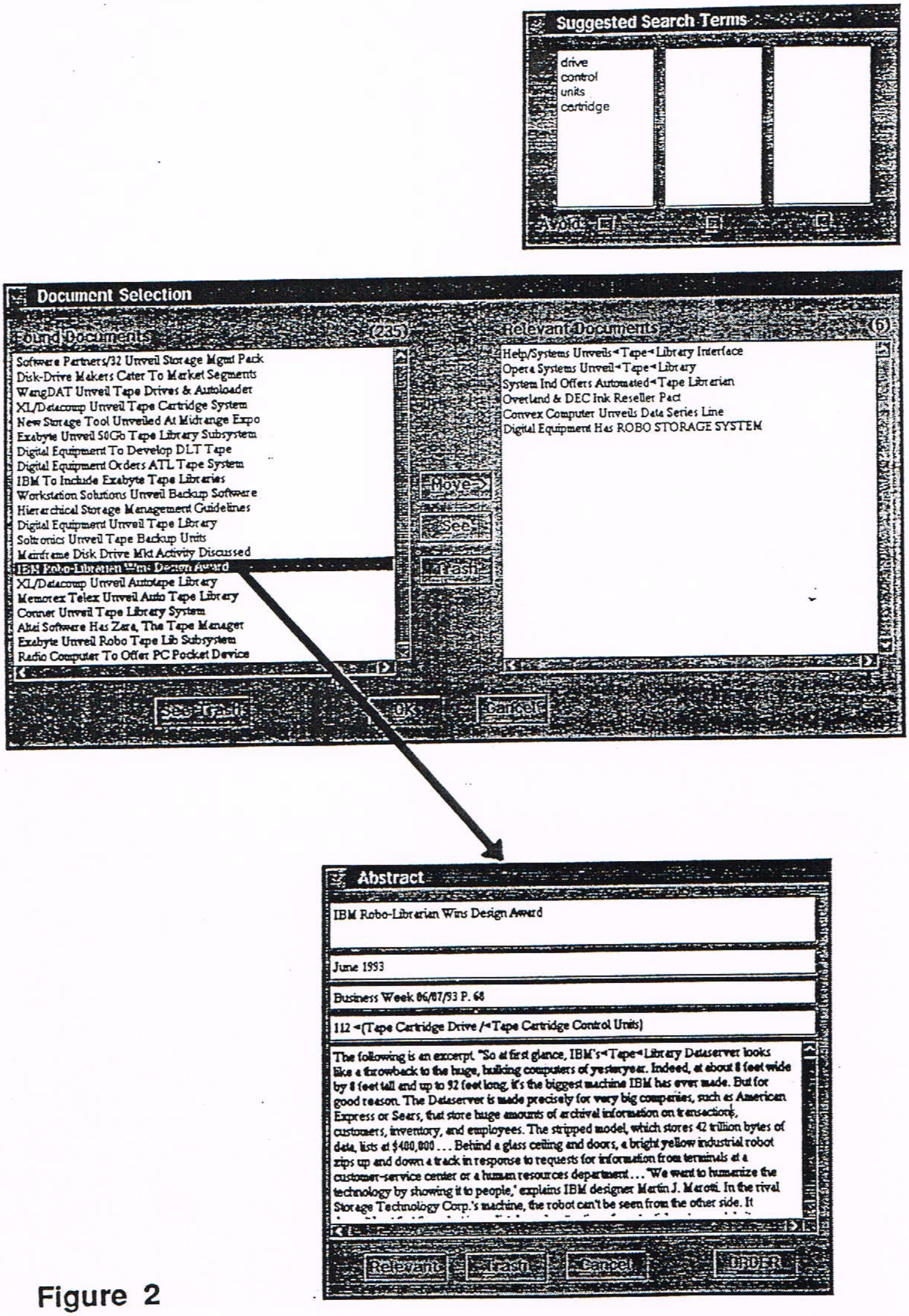


Figure 1



Suggested Search Terms

drive
control
units
cartridge

Document Selection

Found Documents (235)

Software Partners/32 Unveil Storage Mgmt Pack
Disk-Drive Makers Cater To Market Segments
WangDAT Unveil Tape Drives & Autoloader
XL/Datacomp Unveil Tape Cartridge System
New Storage Tool Unveiled At Midrange Expo
Exabyte Unveil 50Gb Tape Library Subsystem
Digital Equipment To Develop DLT Tape
Digital Equipment Orders ATL Tape System
IBM To Include Exabyte Tape Libraries
Workstation Solutions Unveil Backup Software
Hierarchical Storage Management Guidelines
Digital Equipment Unveil Tape Library
Soltronics Unveil Tape Backup Units
Mainframe Disk Drive Mkt Activity Discussed
IBM Robo-Librarian Wins Design Award
XL/Datacomp Unveil Autotape Library
Memorex Telex Unveil Auto Tape Library
Conner Unveil Tape Library System
Altai Software Has Zara, The Tape Manager
Exabyte Unveil Robo Tape Lib Subsystem
Radio Computer To Offer PC Pocket Device

Relevant Documents (6)

Help/Systems Unveils◄Tape◄Library Interface
Opera Systems Unveil◄Tape◄Library
System Ind Offers Automated◄Tape Librarian
Overland & DEC Ink Reseller Pact
Convex Computer Unveils Data Series Line
Digital Equipment Has ROBO STORAGE SYSTEM

Abstract

IBM Robo-Librarian Wins Design Award

June 1993

Business Week 06/07/93 P. 68

112 ◄(Tape Cartridge Drive /◄Tape Cartridge Control Units)

The following is an excerpt. "So at first glance, IBM's◄Tape◄Library Dataserver looks like a throwback to the huge, hulking computers of yesteryear. Indeed, at about 8 feet wide by 8 feet tall and up to 92 feet long, it's the biggest machine IBM has ever made. But for good reason. The Dataserver is made precisely for very big companies, such as American Express or Sears, that store huge amounts of archival information on transactions, customers, inventory, and employees. The stripped model, which stores 42 trillion bytes of data, lists at $400,000 . . . Behind a glass ceiling and doors, a bright yellow industrial robot zips up and down a track in response to requests for information from terminals at a customer-service center or a human resources department . . . 'We want to humanize the technology by showing it to people,' explains IBM designer Martin J. Marotti. In the rival Storage Technology Corp.'s machine, the robot can't be seen from the other side. It

Figure 2