Dana Spiegel
spiegel@media.mit.edu

April 6,1997
6.038

WebResearcher 2000™ — an Intelligent Method for Searching the Web

Overview

Finding useful information on the Internet can be a harrowing experience. The most common (and perhaps only) way to discover the content of the Internet is through the use of a search engine. These utilities, available on the World Wide Web, index the millions (if not billions) of Web pages and UseNet postings available, and provide the user with a (sparse) interface with which to query the index.

Often, the list of sites returned to the user is large. Worse, most of the pages are irrelevant (or not what the user is interested in), and the list returned is unordered (or only incrementally ordered) with respect to the user's preferences and search goals. This is due, in a large part, to the small number of search terms supplied by the user: often only two or three words are given with which to search the tremendous database. As a result, the search utilities usually perform only basic searches on words, and return all pages that match the criteria. The user is forced to read each page themselves to weed out the important information from the rest of the garbage.

The limitation of WWW based search engines is twofold: They provide only a rudimentary *User Interface* (often times only a single field to enter search terms), and they provide no means of *feedback* (the results of a search are dumped into the users lap, with no provision for further searching the information). The first limitation prevents the user from being more specific about the general content of the pages to be retrieved (in many cases it causes the user to be even less specific than they would be otherwise; The user, seeing only a single text entry field, will pare down their search criteria to the few most important words in their search). The second limitation forces the user to do most of the work on their own, which wastes a large amount of time and energy (both important and costly commodities in today's workplace).

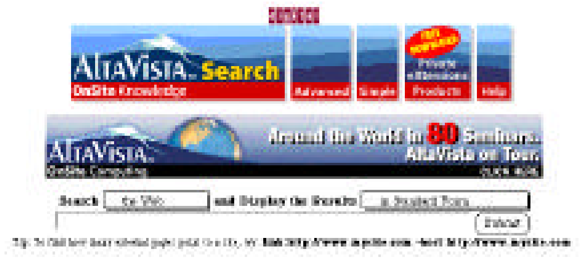


Figure 1 - Most WWW based search engines provide only one line to specify the search parameters. This forces the user to request a search that will not be specific enough to filter out documents that are of no interest.

WebResearcher 2000™ ("R2000") addresses both of these limitations of current WWW based search Engines. R2000 provides a natural language interface to Web searching, which not

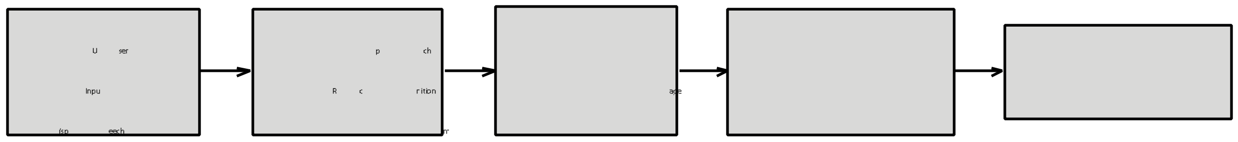


Figure 2 - The Language System takes the user's speech input, and converts it into text, which is then parsed and converted into a database query.

only simplifies the entry of search terms, but provides a medium conducive to the specification of more specific searches.

Additionally, R2000 provides an interactive search method, which allows a user to rate the pages returned by the search engine. Using statistical analysis of the contents of the WWW pages, along with incremental user ratings of the pages, R2000 can provide the user with only the most important and relevant WWW pages, cutting the time to find relevant documents on the World Wide Web from hours to minutes.

User Interface

The User Interface (UI) to R2000 is spoken, natural language. For example, to search the Web for any document relating to the Mets baseball team, the user simply speaks "I want all pages relating to the Mets." Given the complexity and multitude of subject matter available on the Web (and therefore available for search), R2000 is designed to understand only a given domain, such as baseball, and general search terms, such as "I want all pages on..." or "Find pages from the last two weeks... ."

The language system R2000 employs consists of a Speech Recognition system, a Natural Language Parser and Semantic Interpreter. The Speech Recognition system receives spoken input from a microphone located on the user's desk (or other suitable location), and converts the spoken sounds into text sentences. Such systems are available commercially, such as Dragon Dictate, and are likely candidates for use in R2000 (as opposed to using a homemade Speech Recognition System, which would be invariably inferior to commercially available ones, given the short development time allowed for this product).

Once in textual form, the NLP parses the sentences into semantic form. For example, the sentence "I want all pages on the Mets" is converted to:

```
(UNSCOPED ALL V565
  (& ((PLUR PAGE) V565)
    (ON V565 (UNSCOPED THE V575
      (METS V575)))))
```

The semantic form is then converted, by the Semantic Interpreter into a logical search string that can be used to query a database, such as:

```
('retrieve
  ('all
    ('term 'Mets)))
```

By performing this two-fold conversion, R2000 can be asked to perform any valid search in a variety of ways."I want all pages" is logically equivalent to "Find all." Once the input is semantically analyzed, R2000 need only understand the relatively fewer search request strings into which the user input can be converted. Furthermore, because these two systems are separate, they can be independently modified and enhanced. To understand a wider variety of search phrases, the NLP merely needs to be upgraded to convert these new types of phrases into the semantically simpler language of the Semantic Interpreter. Likewise, to enhance search criteria, only the Semantic Interpreter needs to be modified. As long as the new criteria (such as "date posted") can be expressed in

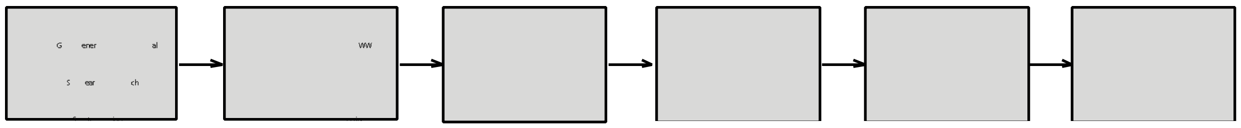


Figure 3 - To generate the database of articles,R2000 first retrieves all Web pages on a particular topic. The pages are then filtered, analyzed, and archived in the database.

in a form known to the NLP, the NLP will understand it and parse it properly.

Once the Language Interface parses the user input into a database search string, it is passed to the Server, which performs all subsequent actions.

Server

The server side of R2000 has two main responsibilities: Preparing the WWW page database, and performing the interactive search.

Database Preparation

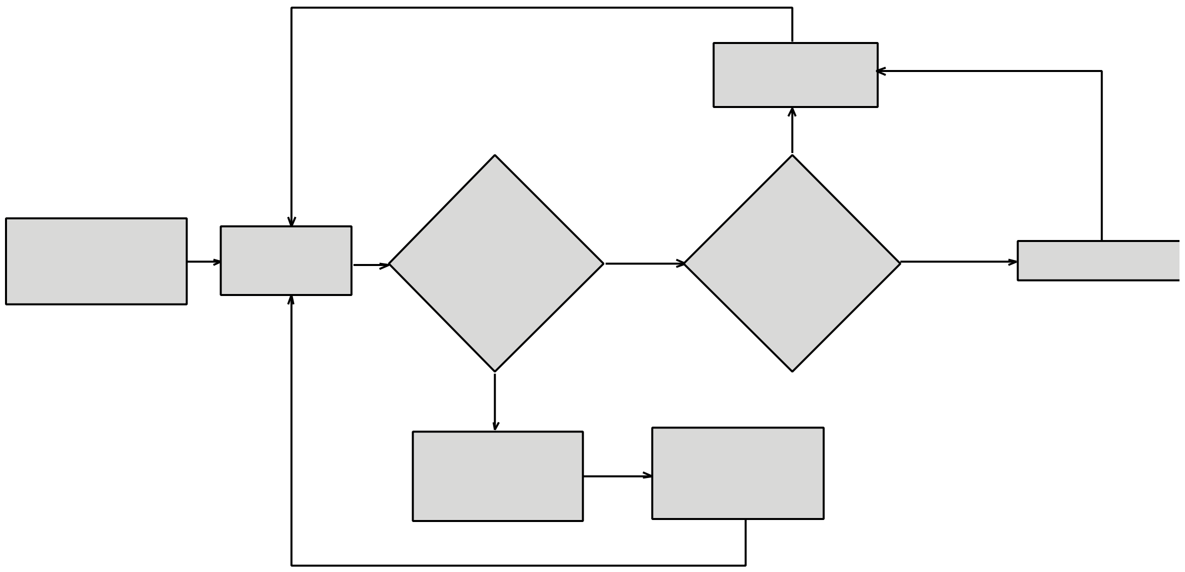
To generate the database, R2000 retrieves, from multiple sources, all WWW pages relating to its scope of knowledge (in this case, baseball). This can be as simple as querying WWW based search engines for all documents matching "baseball," or may be a more specific search (to cut down on the number of documents that need to be retrieved). R2000 can get its data from any online source, such as a general WWW Search engine like AltaVista, or a "trusted" site (one that we know is truthful in its returned documents) like Sports Illustrated (which would likely yield a much higher percentage of pertinent pages than a general Web search, but would also limit the depth and scope of the pages found).

Once a Web page list is generated using the above method, R2000 downloads every Web page in the list, and stores them locally. This particular step is time intensive, both because the number of pages that need to be retrieved is large and the speed of the network is relatively slow. The time needed can be reduced by using multiple machines to perform the retrieval, however the full process will likely take at least the better part of a day. This process, however, needs to be done in entirety only once, as subsequent updates to the database are incremental (R2000 retrieves only those documents that have been changed or added to the Web since the last retrieval period).

Using filters, both for generalized Web pages and standardized pages (for "trusted" servers), R2000 removes unwanted text from the pages, such as HTML tags. With a "stop list" (a list of common words that add no meaning to a sentence), R2000 removes all unimportant words from the pages. The pages are then run through a "stemmer," which converts common forms of words into their root words ("dogs" becomes "dog").

Once the pages are prepared, R2000 performs a statistical analysis on the pages. R2000 generates a list of all of the words within the documents to form a vector of terms, and each term is given an intra-document frequency, which represents the number of different pages that contain that word. Then, for each page, R2000 generates another vector of terms, with each term assigned an inter-document frequency, representing the number of times that word appears in that particular page.

Based on these frequency vectors, R2000 assigns each word a weight, proportional to it's inter-document frequency (the more times a



Engines 3 - To generate the database of articles,R2000 first retrieves all Web pages on a particular topic.The pages are then filtered, analyzed, and archived in the database.

word appears in a document, the more important it is to the meaning of that document), and inversely proportional to its intra-document frequency (the more documents a word appears in, the less important that word is for describing the differences between documents).

Using word weights, R2000 plots the pages on an n-dimensional axis (where n is the total number of different words). Pages are clustered using nearest-neighbor with a specified radius, providing R2000 with groups of pages that are similar to each other.

The above preparation of the database, while computationally intensive, needs to be done only once a week (or at another appropriate interval), since the incremental addition of pages to the database doesn't significantly change the word weights. When a new document enters the system, it merely needs to be plotted and added to a cluster; performing the entire statistical analysis again will not change its cluster significantly, and the trade-off between accuracy and time is significant — a small drop in required accuracy will reduce computation time immensely.

To massage even more data out of a statistical analysis of the pages, generalized phrases can also be analyzed in a similar manner. A document that uses a large number of third person past tense phrases is likely to be more technical, and should be grouped with other similar pages, but only if its sentence structure is unique to a small subset of the pages, and not indicative of the pages in general (in which case the sentence structure should be ignored).

Interactive Search

With a prepared database, R2000 can perform an interactive search. Using the search string parsed from user input (see *User Interface*), R2000 generates a preliminary list of documents from the database. These pages, which can be sorted in word frequency order (similar to WWW based search engines), are presented to the user one at a time.

When the user has read a page, they specify it (using the speech interface described above) as either a good representation or a bad representation of the information they are searching for. Using this rating, R2000 returns to the database, and retrieves the page cluster in which that particular page is contained. This cluster of pages are documents the user is likely to find preferable. If a page is marked as a bad representation, then R2000 removes from the original list of retrieved documents all pages within that bad page's cluster. This process continues until the user has accumulated enough good pages, or the original search list is exhausted.

For example, upon searching for all documents relating to the Mets and the World Series, the user is shown a document about the World Champion 1986 Mets. This document satisfies the user, and they specify it as a good document.R2000 finds documents that are clustered with that specified document,and adds them to list of retrieved documents. The user then reads a page about how the Mets didn't deserve to win the World Series another year, and dislikes that page. R2000 removes all similar pages from the list of retrieved documents.

As a result of the interactive search, the user will need to review far fewer pages and spend much less time reading irrelevant documents than if they had performed a basic search on a WWW Search Engine.

Conclusion

WebResearcher 2000™ represents a new type of WWW Search and Retrieval engine that, using Artificial Intelligence techniques, can significantly aid a user searching through a large amount of information. R2000 reduces time spent reading unrelated documents, and quickly brings to the forefront those documents that are most likely to fulfill a user's needs.

The power of this system lies in its ability to gather a user's preferences using an unintrusive method (simple binary questions are very easy and very fast for a user to answer, and the user quickly learns to answer these questions each time they use the system), and then infer about the documents likely to be preferable to the user. By providing a speech interface, R2000 coerces more initial information than current WWW based search engines about the search to be performed. Users find it much easier to speak their needs than classify them with key words.

R2000, as it is designed above, is easily broken into a system of parts, which can be programmed independently: Speech recognition, NLP, WWW page retrieval, etc. It represents a system that can be built by the class, in the time allotted, and would likely see successful completion by term's end.

For information on systems that have been devel - oped using techniques similar to those above, see:
http://www.media.mit.edu/people/mikeb
and
http://fishwrap.mit.edu/
(specifically the "plum"service")