December 1997

Improving Recall in Web Information Retrieval via Genetic Programming

Karen Cheung

Nabil Kamel

Improving Recall in Web Information Retrieval via Genetic Programming

Karen S.K. Cheung[1]
Tel: (852) 2788-8476
Email: iskare@msmail.is.cphk.hk

Nabil Kamel
Tel: 2788-8697
Email: iskame@msmail.is.cphk.hk

Executive Summary
The rapid growth and availability of information in Internet pose new challenges. The World Wide Web indeed serves as a good medium for users to wander on and retrieve information from the Internet. However, as more people create Web pages, the problems of precision and recall become more pronounced. It is more difficult to find all and only the information one needs, efficiently. In order to improve recall — the focus of our work — the technique of relevance feedback is often used.

Relevance Feedback is a technique to significantly improve recall. It is an iterative query reformulation process used in information retrieval systems. The idea behind relevance feedback is that we can add additional keywords to the original query in each round of search (Salton 1983). One of the fundamental uses of relevance feedback is that it allows us to iteratively refine our search by adding new keywords to modify the initial query to find other relevant information. However, this method can be quite slow, especially if the documents are widely distributed. In order to enhance the performance of relevance feedback, we propose to use Genetic Programming (GP). We describe how to apply GP to improve recall performance in a distributed environment, and discuss related issues.

1 Introduction and Background

Because of the explosive growth of the Internet, the desirability of searching for and retrieving relevant information is a major issue. Slow response time is another aspect that needs great attention. To address these issues, numerous advanced tools have been developed, so as to counter the effects of this explosive growth and to increase the availability of information in a well organized and efficient fashion. There are two main methods for finding information in the Internet: *navigation* and *querying* (Dumais 1995, Gilster 1994 and Ramesh 1995). Navigation is characterized by menu-based systems or hypertext systems such as HyperText Markup Language (HTML). Querying is another method to locate information on the Internet. Boolean matching and keyword matching are common querying methods.

Using boolean matching to search sounds straight forward, however, a boolean search sometimes retrieves no records. The problem of *precision* and *recall* still persist in keyword searching methods. Precision and recall are two fundamental parameters of the effectiveness of information retrieval. Precision is a measure of the proportion of retrieved documents actually relevant. It can be obtained by dividing the number of relevant documents actually obtained by the total number of retrieved documents. Recall measures the proportion of relevant information actually retrieved by a search, that is, the number of relevant documents actually obtained divided by the total number of relevant documents in the collection (Salton 1983).

Besides getting *irrelevant* information, missing a certain amount of information is another great concern. This is due to the various ways in expressing the same concept or naming a particular topic. For instance, some people may use the term "hemoglobin" instead of "blood". If we use "blood" as the search keyword, we will never find those documents which use "hemoglobin" as a keyword.

---

[1] The authors are with the Department of Information Systems, City University of Hong Kong, Tat Chee Avenue, Kowloon Tong, Hong Kong. Fax: (852) 2788-8694

Relevance Feedback is an effective technique to significantly improve recall, and has been receiving much attention. In early research, Rocchio in (Rocchio 1967) showed by conducting experiments that the *vector space model* has very significant improvements in the performance of relevance feedback. Also, Ide in (Ide 1971) extended Rocchio's work to verify the results and further study two other different feedback strategies. Unlike doing research experimentally, Bruza and Huibers (1971) theoretically proposed a framework, *information field*, so to compare information retrieval mechanisms inductively.

However, relevance feedback can be quite slow, especially if the documents are widely distributed. In order to improve the efficiency of relevance feedback, we propose to use Genetic Programming to reduce the cost of increased recall. Our approach to the problem is similar in spirit to the semijoin approach used in relational database systems (Ceri and Pelagatti 1984), however it differs in the nature of the retrieval process and requires a different formulation. It is common knowledge that information are distributed across the Internet. When users submit search queries to the networked system, the answers which satisfy the query may require data from more than one site. In order to speed up the searching process, the *semi-retrieve* operation can be used to reduce the size of a document set, that needs to be transmitted. In a semi-retrieve operation, a set of keywords is extracted from a local document base and transmitted to a remote site, where the documents that match the extracted keywords, according to some match criteria, are retrieved at the remote site. Using semi-retrieve operation saves the cost of data movement between sites, and further speeds up the search process.

The remainder of this paper is organized as follows. Section 2 discusses the search problem. Section 3 gives a brief introduction of Genetic Programming (GP). Section 4 describes our research in progress to apply GP to optimize the search. Finally, Section 5 presents a brief conclusion and future research.

2 The Search Problem

Let us use an example to explain the process of applying semi-retrieve operations in relevance feedback search problems. Assume that we have three document bases, at Sites 1, 2, and 3. A user at Site 1 requests information about blood. The followings are the steps of the keyword search:

1. User at Site 1 enters the term "blood" to perform a keyword search. The details for each document base are as follows:

Site 1	
Docu No	Keyword
11	Blood,Hemoglobin
12	Graph
13	Robot
14	Blood
15	Aircraft
16	Blood,Hemoglobin
17	Hemoglobin
18	Diving
19	Hemoglobin
20	Festival

Site 2	
Docu No	Keyword
21	Join
22	Hemophilia,Hemoglobin
23	Stamp
24	Hemoglobin,Blood
25	Topology
26	Blood,Hemophilia
27	Hemophilia,Hemoglobin
28	Blood
29	Sports
30	Film

Site 3	
Docu No	Keyword
31	Jewelry
32	Hemophilia,Hemoglobin
33	Crystal
34	Hemoglobin,Hemophilia
35	Diamond
36	Hemophilia
37	Red Blood Cell
38	Hemoglobin
39	Animal
40	Christmas

2. After the query is submitted, a local search is done in Site 1 and a list of references which fulfill the keyword "blood" are obtained. Then all keywords from this first set of references are extracted. In this example, they include keywords from documents 11, 14 and 16. The extracted keywords include "Blood" and "Hemoglobin".
3. The extracted keywords are transmitted, i.e., "Blood" and "Hemoglobin", to Site 2 to retrieve documents that contain these keywords. The documents selected are 22, 24, 26, 27 and 28.
4. Instead of transmitting the extracted keywords to Site 2 in Step (3), we can send the keywords to Site 3. The documents retrieved will then be 32, 34 and 38.

Relevance feedback is an iterative process. After the first round of search using the keyword "blood", we will then reformulate the initial query with another new keyword to conduct the second round of search. This process will continue until the list is stabilized, i.e., until no more relevant documents can be retrieved. Of course, a series of semi-retrieve operations will be performed in each round of the search.

The semi-retrieve is a reduction operator. Note that the semi-retrieve operation is not symmetric, i.e. the result will differ if we reverse the order, and is non-associative. In addition, different sequences of semijoins have different cost. It is not difficult for us to use semi-retrieve to handle document retrieval operations between two sites. However, suppose we have a query which involves retrieval of more than, say, seven sites, the situation will be more complicated because the sequences of semi-retrieve will greatly affect the communication costs.

3 Genetic Programming (GP)

According to the evolution theory of Darwin, biological structures that are more successful in grappling with their environment survive and reproduce at a higher rate. This principle is used in Evolutionary Algorithms, for which Genetic Algorithms and Genetic Programming are two of its major classes.

Genetic Algorithm (GA) operates on populations of strings of binary digits, called chromosomes, with the strings encoded to represent individual chromosomes. Three processes, namely, *reproduction*, *crossover*, and *mutation* are applied to the string populations to create new ones. Reproduction takes place on a single chromosome and allows fitter chromosomes to reproduce more. Crossover, the sexual recombination, creates different chromosomes in the offspring by combining materials from the chromosomes of the two parents. Mutation causes the chromosomes of biological offspring to be different from those of their biological parents.

Genetic Programming (GP), proposed by John Koza (1992), is an extension of Genetic Algorithm. Though both GP and GA are mimicking Darwinian evolution for tackling search problems, they differ in the problem and solution representations in actual implementation -- while GA encodes the problem with numeric character strings for further manipulation, GP adopts a symbolic representation.

The structures undergoing adaptation in GP are a population of individual *computer programs* from the search space. These computer programs are generally hierarchically organized and of dynamically varying size and shape. In our case, an individual consists of a specific sequence of semi-retrieve operations. The population is all possible different sequences of semi-retrieve operations.

4 Application of GP to Optimize the Search

There are five preparation steps in order to apply GP, in our case, to improve relevance feedback (Koza 1992).

1. **Determining the set of terminals**
   The first step is to identify a set of terminals that is used in the individual computer programs in the population. The terminals are the inputs, i.e., information that will be processed, for a problem. In our case, the terminals are the different document bases.
2. **Determining the set of functions**
   The second step is to identify a set of functions that will be applied to the set of terminals, i.e., the inputs. The functions, in our case, are the semi-retrieve operations.
3. **Determining the fitness measure**

The third step is to evaluate how good each solution of the problem at hand is. Fitness can be measured in different ways. The most common types of fitness used are the error measure and payoff value (Koza 1992). We consider the performance measured as largely the communication costs to be the fitness measure in our work.

4. **Determining the parameters and variables for monitoring the run**

The fourth step involves selecting the values of parameters to control the runs. *Population size* is one of the important parameters. The population size is chosen according to the complexity of a problem. Generally, the larger the population, the better. But the improvement due to a larger population may not be proportional to the price paid for the increase in computational resources. GP is controlled by 19 control parameters, which includes 2 major numerical parameters, 11 minor numerical parameters, and 6 qualitative variables (which can be neglected in most applications, including the retrieval application addressed in this paper). The two major parameters are population size and the maximum number of *generations* to be run. In our case, the population size is set according to the problem size. Thus problems with more nodes will offer a larger number of semi-retrieve sequences, and the number of generations is limited by the amount of time available for the problem.

5. **Determining the method of designating a result and the criterion for terminating a run**

The final step is to specify the criterion for designating a result and for terminating a run. The method of *result designation* assigns the best individual that ever appeared in any generation of the population (i.e., the best-so-far individual) as the result of a run of genetic programming. The best-so-far individual is then reported as the result of the entire run when the run eventually terminates according to the termination criterion. The run of genetic programming terminates when the *termination criterion* is satisfied. Usually, we terminate a run either when the prespecified number of generations has been reached, or when the *success predicate* has been satisfied. In our work, the success predicate is satisfied if the best solution obtained meets a certain prespecified threshold.

5 Conclusion and Future Research

Users wishing to retrieve information on a certain topic often specify a subset of the keywords needed to retrieve all relevant documents. In order to improve recall, the technique of relevance feedback is often used, with dramatic improvement on recall performance. Relevance feedback, however, is a time consuming technique, especially in a distributed environment. In our work we introduce how to apply GP to efficiently improve recall via relevance feedback technique. In our future research, we will implement a prototype of our GP approach in Common Lisp and use it for simulation.

References

Bruza, P.D. and Huibers, T.W.C. "Investigating Aboutness Axioms using Information Fields." Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, Dublin, Ireland, July 1994, PP. 112-121.

Ceri, S. and Pelagatti, G. Distributed Databases Principles & Systems. New York: McGraw-Hill Book Company, 1984.

Dumais, S. "Finding What You Want: New Tools and Tricks." IEEE Software, Volume 12, No. 5, September 1995, PP. 79-81, 86.

Gilster, P. Finding It on The Internet. New York: John Wiley & Sons, Inc., 1994.

Ide, E. "New Experiments in Relevance Feedback.": In G. Salton (ed.), The SMART Retrieval System, Englewood Cliffs, N.J.: Prentice-Hall, Inc., PP. 337-354.

Koza, J.R. Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, Massachusetts: The MIT Press, 1992.

Ramesh, V.C. "Browse the Power Engineering Virtual Library." IEEE Computer Applications in Power, Volume 8, No. 4, October 1995, PP. 35-38.

Rocchio, J.J. "Relevance Feedback in Information Retrieval.": In G. Salton (ed.), The SMART Retrieval System, Englewood Cliffs, N.J.: Prentice-Hall, Inc., PP. 313-323.

Salton, G. and Mcgill, M.J. Introduction to Modern Information Retrieval. New York: McGraw-Hill Book Company, 1983.