Genetic Search Based on Multiple Mutations

Veljko Milutinovic and Dragana Cvetkovic, University of Belgrade
Jelena Mirkovic, University of California, Los Angeles

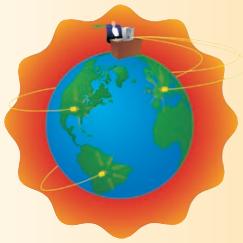
Because of the fast growth in the quantity and variety of Web sites, quickly and efficiently retrieving information on the Internet is becoming increasingly difficult. Searches often result in a huge number of documents—many of which are completely unrelated to what the users are looking for. This problem occurs mostly with indexing engines such as AltaVista, Yahoo, and Lycos, which potentially can find almost any desired document, but also use poor evaluation functions.

Forming the right query is difficult: Using just a few keywords generates too many documents, and using too many keywords generates too few. Because of an unsophisticated evaluation function, documents at the beginning of the result list are often less acceptable than those at the end. If the list is long, users may never reach those better documents because they get bored after following a fraction of the several hundred links.

Before the indexing process is completed, it is impossible to locate a specific new product or service, using search engines based on indexing. Because of the volume of new information created daily, the indexing process can take days, weeks, or even longer.

The key is finding the needed information before it gets indexed. Links-based approaches, such as genetic search or simulated annealing, can do this. The assumption that makes links-based approaches work is that related businesses will update their links to a new page before the page gets indexed.

Genetic search can help even if the information is indexed properly and immediately. A genetic algorithm reprocesses the long list of URLs that an indexing search engine finds—automatically eliminating unrelated URLs so users don't have to do this manually.



Genetic search algorithms enable intelligent and efficient Internet searches. They are especially useful when the search space is relatively large.

HOW GENETIC SEARCH WORKS

An adaptive heuristic method based on the "survival-of-the-fittest" principle, genetic search proves particularly effective when the search space is too large for classical search methods to examine efficiently.

Zero-order genetic search

First, we consider a genetic search without mutation or best first search. The search begins from a set of input Web sites given by a user. Next, the software agent system downloads the Web sites that are linked to this site and evaluates the *fitness value*—how closely the downloaded Web sites' contents fit the user's needs. Typically, only a subset of these sites survive.

The agent system then downloads the sites linked to these surviving sites, and so on, until the end-of-search condition is met. Finally, the agent system ranks the Web sites according to their fitness value and presents them to the user.

Introducing mutations in the data

Mutations introduce some randomness in the search, thus slowing down the convergence and covering more of the search space. Typical mutations use specialized mutational databases (H. Chen et al., "Intelligent Spider for Internet Searching," *Proc. 30th Ann. Hawaii Int'l Conf. System Sciences*, IEEE CS Press, Los Alamitos, Calif., 1997).

Topic-oriented mutations. Traditional database mutation involves choosing from a subset of URLs covering the selected topic. The software agent system enters the specific topic-oriented subdatabase using the URL that would otherwise be used for retrieval of a next Web site. Using an appropriate algorithm, the agent system selects a new URL from the topic-oriented subdatabase to retrieve the next Web site.

Temporal mutations. Mutations based on temporal locality involve maintaining information about previous search results and picking URLs from that set (database). The software agent system periodically returns to a previously fruitful Web site or to other Web sites developed by the same author.

Spatial mutations. Spatial locality exploitation examines the URLs on the same server or local network as the best-ranked URLs. After the agent system finds a desirable Web site using a traditional genetic algorithm with mutation, the engine uses the same Internet service provider to search for other sites that cover a similar topic.

In chaotic markets, competing businesses in the same geographic area probably do not reference one another on their Web sites. So finding one desirable site will most likely not directly lead to another. However, many of these competing businesses may use the same ISP, and spatial mutation algorithms can catch these cases.

After a successful side trip based on spatial mutation, the agent system continues with traditional database mutation until it finds the next-best site or finishes its search.

DESIGN ISSUES

Developing new genetic algorithms and related tools requires innovations in several domain areas. First, you must properly characterize the application environment—determining, for example, the best methods for mutual referencing and anticipating future trends and asymptotic situations foreseen for the time of project finalization. Properly characterizing the application environment is critical to a successful project focus. You must also decide whether the genetic algorithm will emphasize search speed, search sophistication, specific effects of interest for a given institution or customer, or a combination of these. Such decisions affect the applicability (scope of application) of the final product or tool.

Then you must develop an efficient mutation algorithm useful to the application. Incorporating the knowledge about spatial and temporal locality further enhances the genetic algorithm's performance. Algorithms include semantic mutations, based on the principles of spatial and temporal locality, which involve logical reasoning and semantics considerations, in choosing the URLs for mutation.

One major implementation challenge is using the right technology to maximize performance and minimize complexity—for example, selecting the right client and server computing platforms and effectively using mobile-agent technologies. With static agents, the agent system must download megabytes of information from the Internet onto the local disk because content examination and evaluation must occur offline. A huge amount of valueless data passes through the network, and only a small percentage of fetched documents are actually useful (J. Mirkovic et al., "Genetic Algorithms for Intelligent Internet Search: A Survey and a Package for Experimenting with Various Locality Types," *IEEE TCCA Newsletter*, Dec. 1999). Mobile agents, on the other hand, could browse through the network and perform the search locally on the remote servers, transferring back only the needed documents and data (D. Cvetkovic et al., "Architecture of the Mobile Environment for Intelligent Genetic Search and Proxy Caching," *Proc. 33rd Ann. Hawaii Int'l Conf. System Sciences*, IEEE CS Press, Los Alamitos, Calif., 2000).

Using the right technology to maximize performance and minimize complexity is a major challenge.

SIMULATION RESULTS

We have designed a set of software packages (http://rti7020.etf.bg.ac.yu/rti/ebi) that conduct Internet searches using genetic algorithms with various types of mutations. These packages can operate as stand-alone applications or together, so you can combine and upgrade different search methods.

In an experiment to explore how different mutation strategies affect search efficiency, we measured the average Jaccard's score for the output documents, while changing the type and rate of mutation. We fixed the input document set at three files, and the desired number of output documents at 10. At compile time, the topic-mutation database was generated to optimize simulation results. We measured the average Jaccard's score in the static domain for searches with the following criteria:

- no mutation,
- topic mutation,
- both topic and spatial mutation,
- both topic and temporal mutation, and
- all three types of mutation.

Topic mutation significantly improved the quality of the pages found using our search tool, although increasing the mutation rate lessened this improvement (see figure at http://computer.org/computer/IW11.htm). Spatial mutation yielded only sporadic quality improvements; the quality of spatial mutation is highly application-dependent and can be major or minor, depending on the application. Temporal mutation generated a constant improvement in the quality of pages. Employing all three types of mutation generated a constant improvement in the quality of pages that was greater than the improvement from using any of the three mutation types alone or in pairs.

Genetic search has significant potential for improving information retrieval in the Internet domain. On the algorithmic level, one research direction is in database architecture and design. Another direction involves using the elements of various semantics-based mutations. These mutations are especially important for chaotic markets in developed countries or emerging markets in underdeveloped countries.

Veljko Milutinovic *is a professor of computer engineering at the University of Belgrade and a consultant for computer and Internet technologies. Contact him at vm@etf.bg.ac.yu; http://galeb.etf.bg.ac.yu/~vm.*

Dragana Cvetkovic *is a graduate student, research assistant, and computer laboratory and network administrator in the Department of Computer Science and Engineering, School of Electrical Engineering, at the University of Belgrade. Contact her at dana@desert.etf.bg.ac.yu.*

Jelena Mirkovic *is a graduate student and teaching assistant in the Computer Science Department at the University of California, Los Angeles. Contact her at sunshine@cs.ucla.edu.*

Editor: Ron Vetter, Department of Computer Science, University of North Carolina at Wilmington, 601 South College Rd., Wilmington, NC 28403-3297; voice +1 910 962 7192, fax +1 910 962 7457; vetterr@uncwil.edu