Improving Query Expansion with Stemming Terms: A New Genetic Algorithm Approach [*]

Lourdes Araujo and José R. Pérez-Agüera

[1]Dpto. Lenguajes y Sistemas Informáticos. UNED, Madrid 28040, Spain, lurdes@lsi.uned.es
[2]Departamento de Ingeniería del Software e Inteligencia Artificial, UCM, Madrid 28040, Spain, jose.aguera@fdi.ucm.es

Abstract. Nowadays, searching information in the web or in any kind of document collection has become one of the most frequent activities. However, user queries can be formulated in a way that hinder the recovery of the requested information. The objective of automatic query transformation is to improve the quality of the recovered information. This paper describes a new genetic algorithm used to change the set of terms that compose a user query without user supervision, by complementing an expansion process based on the use of a morphological thesaurus. We apply a stemming process to obtain the stem of a word, for which the thesaurus provides its different forms. The set of candidate query terms is constructed by expanding each term in the original query with the terms morphologically related. The genetic algorithm is in charge of selecting the terms of the final query from the candidate term set. The selection process is based on the retrieval results obtained when searching with different combination of candidate terms. We have obtained encouraging results, improving the performance of a standard set of tests.

1 Introduction

Providing answers automatically to client's information needs has become a crucial task nowadays. The spectacular growth of the World-Wide-Web has called for new solutions to access the information needed. An area of particular interest is the reformulation of the queries that users present to the browsers or digital libraries to access the information they need. The information contained in the huge amount of available documents is characterized by a small set of representative terms, called index terms. These terms can be composed of one or more words. Index terms are usually obtained by means of statistical techniques [1], such as the construction of an inverted index[1] whose terms have associated a list of pointers to the occurrences of the term in the text collection.

[*] Supported by projects TIN2007-68083-C02-01 and TIN2007-67581-C02-01.

[1] Given a set of documents containing words, the *inverted index* "inverts" that, so that it consists of a set of words each listing all the documents containing that word.

It very often happens that user queries are composed of terms which are different of the index terms of the collection, despite they refer the same concept, or they correspond to the same stem. Another reason that hinder the retrieval of the requested information is that very often users employ too small sets of search terms because they rely on implicit knowledge that the search engine lacks. An additional problem for retrieval is word and language ambiguity.

All these reasons have made query reformulation an interesting area of research. This operation amounts to changing the original query by adding, removing or replacing terms. In other cases, what is modified is the weights assigned to the terms in the search, to indicate their relevance. Let us consider an example. Suppose that a user, who is interested in learning about modifying the search queries to improve the results of his searches, has posed the following query:

modify query technique

As these three terms are very general, the list of documents that a Web searcher, such as Google or Yahoo, retrieves is extremely large (820,000 documents), and the order in which these documents are presented to the user is inappropriate. In fact, among the first ten documents presented to the user, there are only two more or less related to the concept of *query modification* or *query reformulation.*

One can easily think of expanding the search term by adding to the original query synonyms and related terms. In our example, the query could become:

modify query technique modification transform transformation method

Though the expansion reduces the amount of retrieved documents (now they are about 490,000), there are still many documents among the first ones which have nothing to do with *query reformulation.* Specifically, among the ten first retrieved documents only four are somehow related to the intended topic. Furthermore, a blind query expansion can worsen the result by lowering the precision. Searchers usually present first those documents which contain every term in the query, and this can lead to exclude many relevant documents. On the other hand, there exist other possible transformations which can provide a much better query for the user needs. Suppose that we have a method available that provides us with terms which are truly related to the information the user is looking for, not just to the terms of the query. In our example these terms could be:

query transformation
query modification
query expansion
information retrieval
web search

Now, an appropriate selection of terms among those of the original query and the related ones can lead to much more precise results. In the example, let us assume that the query is transformed to

"query expansion" "query transformation"

Now the amount of retrieved documents is 212 and all of them are relevant to the user needs.

Because of the extended use of information search in our society, query reformulation has become a very active research area and different approaches have been applied. Salton [16] and Robertson and Sparck Jones [15] use relevance feedback to reformulate the query. Systems based on relevance feedback modify the original query taking into account the user relevance judgements on the documents retrieved by this original query.

There is an underlying common background in many of the works done in query reformulation, namely the appropriate selection of a subset of search terms among a list of candidate terms. Because the number of possible subsets can be very large, it makes sense to apply heuristic search techniques to the problem, such as genetic algorithms (GAs).

GAs have been previously applied [4] to different aspects of information retrieval. Proposals devoted to the query expansion problem with GAs can be classified into relevance feedback techniques and Inductive Query by Example (IQBE) algorithms. In systems based on relevance feedback [21] the user gives feedback on the relevance of documents retrieved by his original query. IQBE [2] is a process in which the user does not provide a query, but document examples and the algorithms induce the key concepts in order to find other relevant documents.

Several works apply GAs to assigning weights to the query terms [14, 22, 18, 9, 8], while others are devoted to selecting the query terms. Let us review some proposals in the latter case, the one on which we focus our work. Chen et al. [2] apply a GA as an IQBE technique, i.e. to select the query terms from a set of relevant documents provided by the user. In this work, the authors propose an individual representation that has also been used in later works: chromosomes are binary vectors of fixed size in which each position is associated with one of the candidate terms. In [10], the authors propose a genetic programming (GP) algorithm to learn boolean queries encoded as trees whose operators are AND, OR and NOT. This work was later extended in [5] by incorporating multiobjective optimization techniques to the problem. Fernández-Villacañas and Shackleton [11] compared two evolutionary IQBE techniques for boolean query learning, one based on GP and the other on a classic binary GA, which obtained the best results. Kraft et al. [12] propose the use of GP to learn fuzzy queries. The queries are encoded as expression trees with boolean operators as inner nodes, and query terms with weights as terminal nodes. Cordón et al. [3] extend Kraft's proposal by applying a hybrid simulated annealing-genetic programming scheme, what allows them to use new operators to adjust the term weights. Tamine et al. [19] use knowledge-based operators instead of the classical blind operators, as well as niching techniques.

A common factor of the above mentioned works is that they relay on some kind of information provided by the user. In some cases, the user has to provide a set of documents that are used for the inductive learning of terms. In other cases, the user provides relevance judgements on the retrieved documents, that are use to compute the fitness.

In this work we propose a new application of GAs to the selection of query terms. The novelty is that our system does not require any user supervision: new candidate terms for the query are provided by a morphological thesaurus. Some of the new terms will allow to retrieve new relevant documents. However, including the whole set of candidate terms in the query will, in general, degrade the system performance because documents fetched by expanded terms can relegate documents relevant for the original query terms. Accordingly, we apply a GA to perform a selection of terms. The GA works with individuals which represent different combinations of candidate query terms or queries. The selection process is based on the relevance, with respect to the original query, of the first N (a parameter) documents retrieved when submitting the query to an automatic searcher[2]. To compute the relevance of a retrieved document we use the classical vector space model of information retrieval (IR) [6]. We have performed experiments to investigate the limit of the performance that can be reached by the GA, by assuming that we have available the relevance judgements provided by the user. After studying the GA parameters for this case, we have obtained an important improvement of the performance. Then, we have studied the GA which is applied without user supervision.

The rest of the paper proceeds as follows: section 2 describes the general scheme of the process to construct the set of candidate query terms; section 3 is devoted to describe the evolutionary algorithm used to select the terms of the final query, including different fitness functions tested in the experiments; section 4 presents and discusses the experimental results, and section 5 draws the main conclusions of this work.

2 The query expansion system

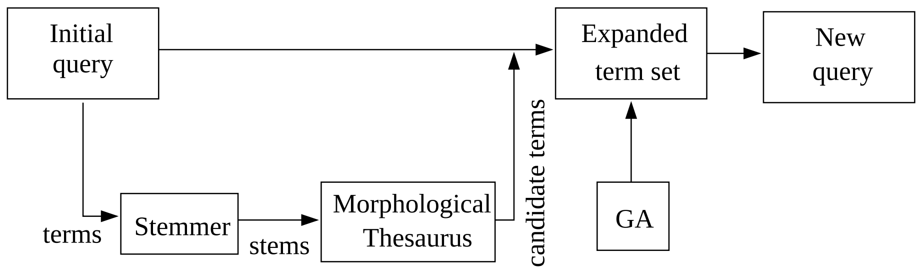

Fig. 1. Scheme of the process to select candidate terms to expand the query and the later selection of them by the Genetic Algorithm.

[2] the system does not require the user supervision

A classical operation in information retrieval to improve the performance of the systems is to use morphological variants. In most cases, morphological variants of words have similar semantic interpretations and can be considered as equivalent for the purpose of IR applications. For this reason, a number of so-called stemming algorithms, or stemmers, have been developed. They reduce a word to its stem or root form. One can view stemming as a form of global query expansion: we expand a term in the query with all the terms in the dictionary sharing the same stem. Our system is based on this idea of expanding the query with morphological variants.

Figure 1 shows a scheme of the process followed to transform a query. First of all, we obtain the stems which correspond to the original query terms. This is done with the well known Porter stemmer [20, 13]. Then, we use the *morphological thesaurus* for Spanish available along with the Porter stemmer, which provides the different forms (plural and grammatical declinations) corresponding to each stem. All these terms are candidate for the final query. Finally, the GA is in charge of selecting, from the candidate term set, the final query terms, that are submitted to the searcher.

3 The genetic algorithm

Chromosomes [7] of our GA are fix-length binary strings where each position corresponds to a candidate query term. A position with value one indicates that the corresponding term is present in the query. Individuals of the initial population are randomly generated. Because of some preliminary experiments we have performed have shown that, in most cases, the elimination of the original query terms degrades the retrieval performance, we force to maintain them among the selected terms of every individual. The set of candidate terms is composed of the original query terms, along with related terms provided by the applied thesaurus.

The selection mechanism to choose individuals for the new population uses roulette wheel. We apply one-point crossover operator. Our algorithm uses random mutation which flips a bit randomly chosen. We also apply elitism.

3.1 Fitness functions tested

The fitness functions that we have proposed are different measures of the degree of similarity between a document belonging to the document collection and the submitted query. To compute this similarity, we apply the vector space model of information retrieval [6, 1]. In this model, a document d_j and a query q are represented as n-dimensional vectors. To construct the vectors, we have to assign weights to index terms in queries and in documents. The classic vector space model computes the term weight as:

$$w = tf \cdot log\frac{D}{d}$$

where *tf* stands for term frequency, $log\frac{D}{d}$ is the inverse document frequency, D is the total number of documents in the document set and d is the number of documents containing the term.

Then, as proposed by Salton and McGill [17], the degree of similarity of the document d_j with respect to the query q is evaluated as the distance between the vectors $\boldsymbol{d_j}$ and $\boldsymbol{q}$. This distance can be quantified by the cosine of the angle between these two vectors. Based on this similarity measure, we have considered three alternative fitness functions: $\sqrt{\cos\theta}$, $\cos\theta$, and $\cos^2\theta$. They differ on the distance among the values assigned to different individuals, which can affect the GA selection process.

4 Experimental results

The system has been implemented in Java, using the JGAP library[3], on a Pentium 4 processor. We have used a set of tests provided by CLEF (Cross-Language Evaluation Forum) for the Spanish language. The collection and tests used come from EFE94. This document collection came from the international news agency EFE, from all the news received during 1994 and consists of 215.738 documents stored in files with SGML format.

In the first place, we have performed experiments to know the limit to the improvement we can reach according to the data provided by the collection used in the experiments, and in this way to have an idea of the quality of the fitness function that have been tested later.

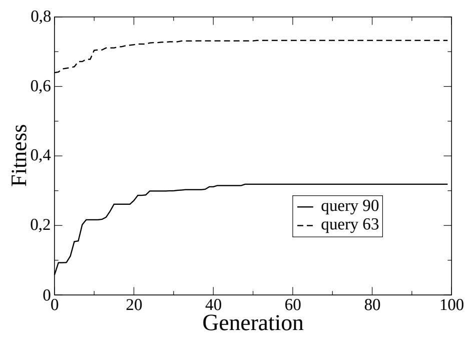

Fig. 2. Fitness evolution for two queries of the test set. The GA parameters have been a population size of 100 individuals, a crossover rate of 25% and a mutation rate of 1%.

To this purpose, we take the user relevance judgements as the best fitness function that we can use. Since for the CLEF collection used in the experiments

[3] http://jgap.sourceforge.net/

we have the user relevance judgements, we have used them to guide the selection process. Specifically, we have used as fitness function the standard precision measure ($\frac{|Ra|}{|A|}$) defined as the fraction of retrieved documents (the set A) which are relevant (R_a). Table 1 shows the precision of the best individual obtained by the GA. We have compared our system performance with the results of the original user query (*Baseline*) and with the results obtained expanding with the stems provided by the Porter stemming (*Porter Stemming*). The latter works by substituting the original query terms by their stems and performs the search in the document collection indexed by stems. We can observe that our system achieved an important improvement of the performance, greater than the one achieved with other stemming methods traditionally used in query expansion, such as Porter. We consider the improvement achieved a ceiling for the improvement of the unsupervised GA. Figure 2 shows the evolution of two queries of the test set. We can observe that both of them reach convergence very quickly.

	Prec.	Prec10	Improvement
Baseline	0.3567	0.4150	–
Porter Stem.	0.4072	0.45	+12.40%
Genetic Stem.	0.4682	0.54	+23.81%

Table 1. Global precision results for the whole set of tested queries. Each individual datum has been computed as the average over 5 different GA runs. Prec. stands for precision (all documents), Prec10 stands for the precision of the results for the first ten documents retrieved. Last column is the rate of precision (all documents) improvement.

4.1 Selecting the Fitness Function

Let us now investigate the proposed unsupervised GA. To select the fitness function to be used in the remaining experiments, we have studied the fitness evolution for different queries of our test set. Figure 3 compares the fitness evolution for the query which reaches the greatest improvement (*best_query*). The most relevant point in this figure is the generation at which each function reaches its optimum. The three functions converge to different numerical values that correspond to the same precision value (.68). We can observe that the square-root cosine function is the first one to converge to its optimum. Probably because this function emphasizes the distance between the values assigned to different individuals, thus improving the selection process of the GA. Accordingly, the square-root cosine has been the fitness function used in the remaining experiments.

4.2 Tuning the GA Parameters

The next step taken has consisted in tuning the parameters of the GA. Figure 4 shows the fitness evolution using different crossover rates for two queries, the

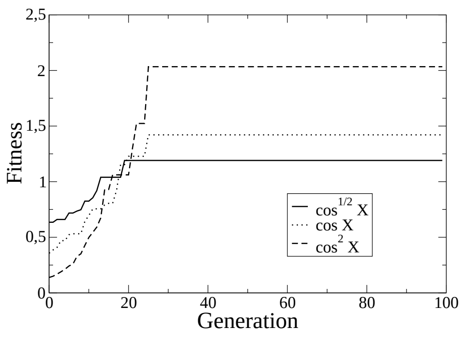

Fig. 3. Fitness functions comparison for the *best_query*, the one for which the greatest precision improvement is achieved. The GA parameters have been a population size of 100 individuals, a crossover rate of 25% and a mutation rate of 1%.

best one (Figure 4(a)), and the worst one (Figure 4(b)). Results show that, in both cases, we can reach a quickly convergence with values around 25%.

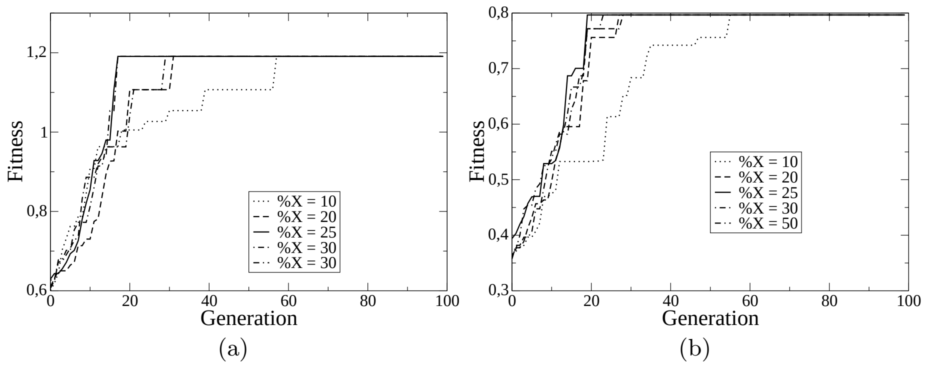

Fig. 4. Studying the best crossover rate for the best_query (a) and the worst one (b). The GA parameters are a population size of 100 individuals and a mutation rate of 1%.

Figure 5 presents the fitness evolution using different mutation rates for the best (a) and the worst (b) queries. Values around 1% are enough to produce a quick convergence.

Figure 6 show the fitness evolution for the best (a) and the worst (b) queries, with different population sizes. The plots indicate that small population sizes, such as one of 100 individuals, are enough to reach convergence very quickly.

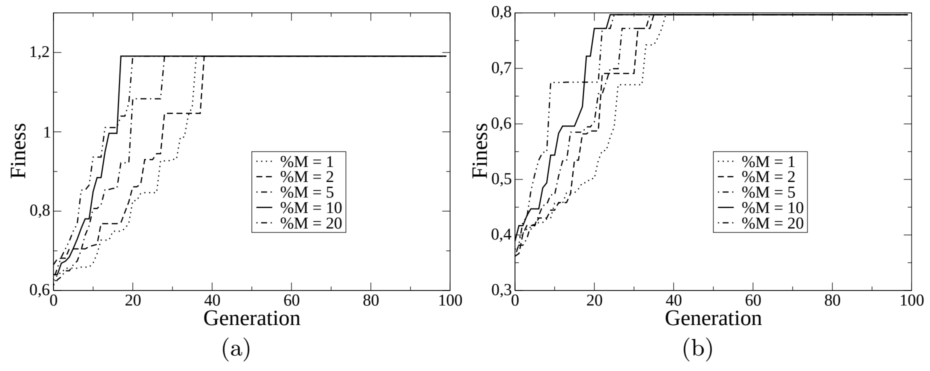

Fig. 5. Studying the best mutation rate for the best_query (a) and the worst one (b). The GA parameters are a population size of 100 individuals and a crossover rate of 25%.

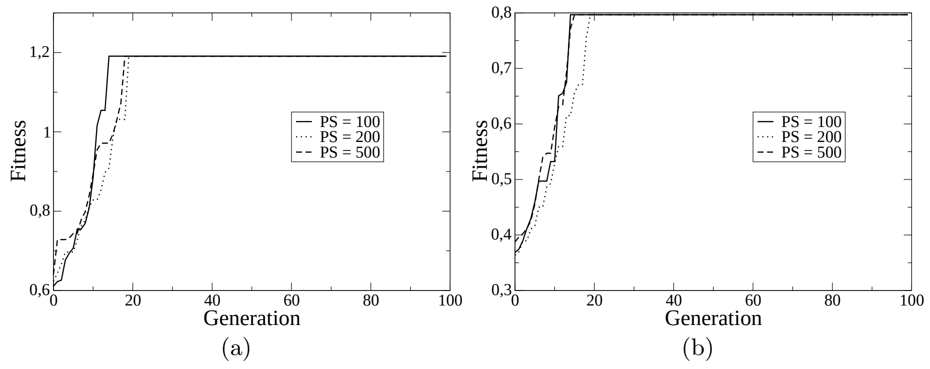

Fig. 6. Studying the population size for the best_query (a) and the worst one (b). The GA parameters are a crossover rate of 25% and a mutation rate of 1%.

4.3 Overall Performance

Table 2 presents precision and recall results obtained for the whole set of 40 test queries that we have considered. Recall (Recall $= \frac{|R_a|}{|R|}$), is a coverage measure defined as the fraction of the relevant documents (the set R) which has been retrieved (R_a). We have compared our system performance with the results obtained with the original user query (*Baseline*) and with the results obtained using the Porter stemming (*Porter Stemming*). Each individual datum has been computed as the average over 5 different GA runs. We can observe that our system is able to obtain an improvement in precision of 15.20% over the baseline, being this improvement larger that the one obtained by other methods, such as Porter stemming. Furthermore, our system also achieves a great improvement in the system recall. Table 2 also shows the recall results. Thus, our system is able to improve the coverage, improving precision at the same time. With the current implementation the mean execution time per query is 45 seconds.

	Prec.	Prec5	Prec10	Improvement	Recall	Improvement
Baseline	0.3567	0.4750	0.4150	–	0.7035	–
Porter Stem.	0.4072	0.50	0.45	+12.40%	0.7228	+2.67%
Genetic Stem.	0.4206	0.5150	0.4575	+15.20%	0.7521	+6.46%

Table 2. Global precision and recall results for the whole set of tested queries. Prec. stands for precision (all documents), Prec5 stands for the precision of the results for the first five documents retrieved, and Prec10, stands for precision for the first ten documents retrieved. Last column is the rate of precision (all documents) improvement.

4.4 Analyzing a final query

Apart from evaluating the numerical performance of our system, we have analyzed the queries resulting from the applied expansion process. Let us first consider the query 63 of the collection, the one with best performance (precision increases from .55 to .68, 19.11% of improvement). Table 3 shows the results for this query.

User query: *reserva de ballenas*
Set of candidate terms: *reserva, reservaba, reservaban, reservaciones, reservación, reservada, reservadamente, reservadas, reservado, reservados, reservamos, reservan, reservando, reservandose, reservar, reservara, reservaran, reservarlas, reservarle, reservarles, reservarlo, reservarlos, reservarnos, reservaron, reservarse, reservará, reservarán, reservaría, reservarían, reservas, reservase, reserve, reserven, reservista, reservistas, reservo, reservoir, reservándole, reservándolos, reservándose, reservó, ballenas, ballena, ballén, balléna*
GA final query: *ballenas ballena reserva*

Table 3. Retrieval results of a query example

The original query is a compound term in Spanish, whose meaning is *whale reserve*. The first observation is the large size of the set of candidate terms, what makes clear the need for a selection. In the final query we can observe that the most representative terms related to the topic, such as *ballena* and *ballenas* (singular and plural Spanish words for whale), and the word *reserva* (reserve) have been added to the query. This query suggests investigating the application of a special treatment of compound terms, using "reserva de ballenas" as a single search term. This is a matter of future work.

Another query is *energía renovable* (Renewable Energy), for which the final query is *energías renovables energía renovable*. It achieves an improvement of 39.21%.

In other cases, such as for the query *Verduras, frutas y cáncer* (vegetables, fruits and cancer), the final query and the original query are the same. In this case, the whole point of the query is the relationship between the query terms.

Because of this, searching independently for alternative forms of the query terms can only worsen the precision. However, the GA is capable of clearing the expanded query, recovering the original query and thus maintaining the system performance.

5 Conclusions

In this paper we have shown how an evolutionary algorithm can help to reformulate a user query to improve the results of the corresponding search. Our method does not require any user supervision. Specifically, we have obtained the candidate terms to reformulate the query from a *morphological thesaurus*, with provides, after applying stemming, the different forms (plural and grammatical declinations) that a word can adopt. The evolutionary algorithm is in charge of selecting the appropriate combination of terms for the new query. To do this, the algorithm uses as fitness function a measure of the proximity between the query terms selected in the considered individual and the top ranked documents retrieved with these terms.

We have carried out some experiments to have an idea of the possible improvement that the GA can achieve. In these experiments we have used the precision obtained from the user relevance judgements as fitness function. Results have shown that in this case the GA can reach a very high improvement.

We have investigated different proximity measures as fitness functions without user supervision, such as cosine, square cosine, and square-root cosine. Experiments have shown that the best results are obtained with square-root cosine. However, results obtained with this function do not reach the reference results obtained using the user relevance judgements. This suggests investigating other similarity measures as fitness functions.

A study of the queries resulting after the reformulation has shown that in many cases the GA is able to add terms which improve the system performance, and in some cases in which the query expansion spoils the results, the GA is able to recover the original query.

We are now working on reducing the execution time per query following two lines. On the one hand we are improving the GA implementation, and on the other hand we are investigating alternative fitness functions which do not require to retrieve documents at each evaluation.

References

1. R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval.* ACM Press / Addison-Wesley, 1999.
2. H. Chen, G. Shankaranarayanan, L. She, and A. Iyer. A machine learning approach to inductive query by examples: An experiment using relevance feedback, id3, genetic algorithms, and simulated annealing. *JASIS*, 49(8):693–705, 1998.
3. O. Cordón, F. de Moya Anegón, and C. Zarco. A new evolutionary algorithm combining simulated annealing and genetic programming for relevance feedback in fuzzy information retrieval systems. *Soft Comput.*, 6(5):308–319, 2002.

4. O. Cordón, E. Herrera-Viedma, C. López-Pujalte, M. Luque, and C. Zarco. A review on the application of evolutionary computation to information retrieval. *Int. J. Approx. Reasoning*, 34(2-3):241–264, 2003.
5. O. Cordón, E. Herrera-Viedma, and M. Luque. Improving the learning of boolean queries by means of a multiobjective iqbe evolutionary algorithm. *Inf. Process. Manage.*, 42(3):615–632, 2006.
6. G. Salton. *Automatic Information Organization and Retrieval.* McGraw Hill Book Co, 1968.
7. J. J. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, 1975.
8. J.-T. Horng and C.-C. Yeh. Applying genetic algorithms to query optimization in document retrieval. *Inf. Process. Manage.*, 36(5):737–759, 2000.
9. C. Lopez-Pujalte, V. P. G. Bote, and F. de Moya Anegón. A test of genetic algorithms in relevance feedback. *Inf. Process. Manage.*, 38(6):793–805, 2002.
10. S. M. and S. M. The use of genetic programming to build boolean queries for text retrieval through relevance feedback. *Journal of Information Science*, 23(6):423–431, 1997.
11. J. L. F.-V. Martín and M. Shackleton. Investigation of the importance of the genotype-phenotype mapping in information retrieval. *Future Generation Comp. Syst.*, 19(1):55–68, 2003.
12. F. E. Petry, B. P. Buckles, T. Sadasivan, and D. H. Kraft. The use of genetic programming to build queries for information retrieval. In *International Conference on Evolutionary Computation*, pages 468–473, 1994.
13. M. F. Porter. An algorithm for suffix stripping. In *Readings in information retrieval*, pages 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.
14. A. M. Robertson and P. Willet. An upperbound to the performance of ranked-output searching: optimal weighting of query terms using a genetic algorithm. *J. of Documentation*, 52(4):405–420, 1996.
15. S. Robertson and K. S. Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, 1996.
16. G. Salton. *The SMART Retrieval System—Experiments in Automatic Document Processing.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1971.
17. G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill Book Company, 1983.
18. E. Sanchez, H. Miyano, and J. Brachet. Optimization of fuzzy queries with genetic algorithms. application to a data base of patents in biomedical engineering. In *VI IFSA Congress, vol. II*, pages 293–296, 1995.
19. L. Tamine, C. Chrisment, and M. Boughanem. Multiple query evaluation based on an enhanced genetic algorithm. *Inf. Process. Manage.*, 39(2):215–231, 2003.
20. C. van Rijsbergen, S. Robertson, and M. Porter. New models in probabilistic information retrieval. Technical Report 5587, London: British Library, 1980.
21. C. J. van Rijsbergen. *Information retrieval.* Butterworths, London, 2 edition, 1979.
22. J.-J. Yang and R. R. Korfhage. Query modification using genetic algorithms in vector space models. *Int. J. Expert Syst.*, 7(2):165–191, 1994.