



PERGAMON

Information Processing and Management xxx (2001) xxx–xxx

www.elsevier.com/locate/infoproman

**INFORMATION  
PROCESSING  
&  
MANAGEMENT**

## A test of genetic algorithms in relevance feedback

Cristina López-Pujalte <sup>a,\*</sup>, Vicente P. Guerrero Bote <sup>a</sup>, Félix de Moya Anegón <sup>b</sup>

<sup>a</sup> *Facultad de Biblioteconomía y Documentación (Alcazaba de Badajoz), Universidad de Extremadura, 06071 Badajoz, Spain*

<sup>b</sup> *Facultad de Biblioteconomía y Documentación, Universidad de Granada, Campus Cartuja, Granada, Spain*

Received 9 May 2001; accepted 9 November 2001

---

### Abstract

10 There have been recent applications of genetic algorithms to information retrieval, mostly with respect to  
11 relevance feedback. Nevertheless, they are yet to be evaluated in a way that allows them to be compared  
12 with each other and with other relevance feedback techniques. We here implement the different genetic  
13 algorithms that have been applied in the literature together with some of our own variations, and evaluate  
14 them using the residual collection method described by Salton in 1990 for the evaluation of relevance  
15 feedback techniques. We compare the results with those of the Ide dec-hi method, which is one of the  
16 traditional methods that yields the best results. © 2001 Published by Elsevier Science Ltd.

17 *Keywords:* Genetic algorithms; Relevance feedback; Information retrieval; Test collections

---

### 18 1. Introduction

19 Everyone is aware of the deficiencies in the first query that a user puts to an information re-  
20 trieval (IR) system. Hence the importance of query optimization techniques that allow the user to  
21 obtain better results. One such technique is *relevance feedback*. This uses the information supplied  
22 by the user in response to a first retrieval of information from the system to modify the next query.

23 At the same time there have been major advances in the application of artificial intelligence (AI)  
24 techniques in the field of information science, particularly in IR which forms one of the main lines  
25 of AI research. First there were the applications of knowledge-based systems (of which expert  
26 systems are of one specific type) (Poulter, Morris, & Dow, 1994), and of neural networks

---

\* Corresponding author. Tel.: +34-242-59910; fax: +34-242-59957.

*E-mail addresses:* clopez@alcazaba.unex.es (C. López-Pujalte), vicente@alcazaba.unex.es (V.P. Guerrero Bote), felix@goliat.ugr.es (F.M. Anegón).

27 (Guerrero Bote, Moya Anegón, & Herrero Solana, 2001), followed by genetic algorithms  
 28 (Goldberg, 1989; Holland, 1992; Michalewicz, 1995), an AI technique which emulates the genetic  
 29 evolution of species (Belew, 1989; Gordon, 1988a,b; Kwok, 1989, 1990; Raghavan & Agarwal,  
 30 1987). Nevertheless, one would say that research uniting evolutionary computation with IR is still  
 31 in its early stages.

32 A GA is an AI technique which uses adaptive procedures to solve problems of searches that  
 33 satisfy certain requirements, and is inspired in the mechanisms of biological evolution, based on  
 34 the principle of natural selection and on species' genetic codes.

35 The input to a GA is a population of individuals called *chromosomes*. These represent the  
 36 possible solutions to the problem. The chromosomes may be randomly generated, or, if one al-  
 37 ready has some knowledge of the optimal distribution, one may use that information to create  
 38 part of the initial set of potential solutions (Michalewicz, 1995; Yang & Korfhage, 1994).

39 These individuals evolve in successive iterations called *generations*, by processes of *selection*,  
 40 *crossover*, and *mutation*. These iterations stop when the system ceases to improve or when a set of  
 41 maximum number of generations is reached. The output of the GA is the best individual of the  
 42 final population or a combination of the best chromosomes of that population.

43 Fig. 1 represents the algorithm of this process. The iterative procedure can be summarized as  
 44 follows: during iteration (generation)  $t$  the GA has a population  $P(t)$  of potential solutions (the  
 45 chromosomes or vectors). Each chromosome is evaluated by means of the fitness function to give  
 46 a measure of its suitability. The most suitable individuals are selected to form part of the re-  
 47 production population. Some of the members of this intermediate reproduction population un-  
 48 dergo alterations (recombination) due to the action of genetic operators (crossover and/or  
 49 mutation) to form new solutions which make up a new population  $P(t + 1)$ , and which in turn will

```

begin
    t = 0;
    Initialize P(t);
    Evaluate P(t);
    While (not termination-condition) do
        begin
            t = t+1;
            Select P(t) from P(t-1);
            Alter (cross and mutation) P(t);
            Evaluate P(t);
        end
    end
end.

```

Fig. 1. Structure of a genetic algorithm (adapted from Michalewicz, 1995).

50 be the initial population of the following generation. The process is iterated until the termination  
51 condition is reached.

52 As one can see that, for each problem to be solved, one has to provide an *evaluation* or *fitness*  
53 *function*  $f$ . Its choice is crucial for the GA to function well. Given some chromosome, the fitness  
54 function must return a numerical value proportional to the utility of the solution the chromosome  
55 represents. This score is used in the process of parent selection and in sorting or classifying the  
56 chromosomes. The fitness function must therefore be suited to the problem at hand, since the  
57 efficacy of the GA will to a great degree be determined by how faithfully  $f$  characterizes the  
58 function being optimized.

59 A GA is a versatile domain-independent technique with a certain implicit degree of parallelism  
60 which is well suited to the huge spaces of document databases.

61 Unlike neural networks, a GA is not specifically a learning algorithm, but it can be used in  
62 many learning tasks because of its adaptive nature: the possible solutions evolve towards better  
63 solutions, approaching the optimal case.

64 According to the study performed by Cordón, Moya, and Zarco (1999), there exist three main  
65 areas of GA application to IR:

- 66 • document indexing,
- 67 • clustering,
- 68 • query optimization.

69 This third group of applications, query optimization, is the most numerous. They all have in  
70 common the use of a GA to perform the technique of relevance feedback. There has been no  
71 comparison of the results of these isolated experiments, however, with the traditional algorithms  
72 of relevance feedback. We shall here evaluate the different GA's applied to relevance feedback in  
73 the literature, following the vector model (the most commonly used model in this sort of appli-  
74 cation). We shall use the Cranfield collection, and the residual collection method, and then  
75 compare the results with those of the classical Ide dec-hi feedback method.

## 76 2. Antecedents

77 The first use of GA's was to solve problems in which there was feedback from the environment.  
78 In particular, they were used to fit parameters in, for instance, oil field simulations, market  
79 analysis, classification, etc. (Glover, 1987; Goldberg, 1989; Grefenstette, 1986, 1987; Hilliard &  
80 Liepins, 1987, Robertson, 1987). It is not surprising therefore that they began to be investigated in  
81 the IR field with respect to their application to relevance feedback (Yang & Korfhage, 1994).

82 In the following, we discuss only the literature on which our current research is based, high-  
83 lighting the fitness functions used by the GA's implemented in those studies. These are the  
84 functions (sometimes with certain modifications to adapt them to our experiment) that in the  
85 present work we shall implement within the previously determined optimal feedback GA.

86 Robertson and Willett (1996) implement a GA which learns the weights of the query terms by  
87 means of relevance feedback. To calculate the individuals' fitness, they first find the inner product  
88 of each query of the collection with each document of the database. They then retrieve a fixed  
89 number of documents, and finally calculate the recall of the retrieval. We shall implement our

90 functions *fitness 1* and *fitness 2* on the basis of the fitness function used in this work. Our function  
91 *fitness 2* is a variant of *fitness 1*, using the cosine as similarity measure instead of the inner product.  
92 Chen and co-workers (Chen, 1995; Chen & Iyer, 1998; Chen, Chung, & Ramsey, 1998) use a  
93 GA to learn the query terms which best represent a set of documents supplied by the user (this  
94 process they call *inductive query by examples*), and to end up with the construction of an intelligent  
95 agent which uses this genetics to implement the feedback module. Their fitness function finds the  
96 Jaccard index for each chromosome (possible query) with the rest of the chromosomes of the  
97 population to then obtain the mean value of these similarities. We derive our functions *fitness 3*,  
98 *fitness 4* and *fitness 5* from this work. The last two are practically the same as the first, but use the  
99 inner product and the cosine, respectively, as similarity measure.

100 Another work that merits special attention is that of Yang and Korfhage (1994), which pro-  
101 poses a GA that optimizes the weights of a query by means of relevance feedback, after carrying  
102 out (as we also do) a process of eliminating stopwords and stemming the remaining terms. Each  
103 individual's fitness is calculated by comparing each query with the documents of the collection  
104 using the Euclidean distance as the similarity measure, and those documents whose distance is less  
105 than some given threshold are retrieved. Finally, the fitness is obtained from the following  
106 equation:

$$F_1 = Rr - Rn - Nr,$$

108 where  $Rr$  is the number of documents retrieved that are in the set of relevant documents provided  
109 by the collection (this might also be the number of documents judged as relevant by the user, if the  
110 user is present in the experiment),  $Rn$  is the number of documents retrieved that are not included  
111 in the standard relevant set, and  $Nr$  is the number of relevant documents that have not been  
112 retrieved.

113 We took our functions *fitness 6* and *fitness 7* from this work. *Fitness 7* is similar to *fitness 6*, but  
114 instead of calculating a retrieval threshold, a quota of documents is set (the first 10 documents) in  
115 order to be able to compare the results with other functions that employ a quota instead of a  
116 threshold.

117 Another very innovative work, especially with respect to the fitness function, is that of Horng  
118 and Yeh (2000). This also uses a GA to fit the weights associated with the query terms. The fitness  
119 function, as well as including the number of relevant and irrelevant documents retrieved, takes  
120 their order of appearance into account, since it is not the same that the relevant documents appear  
121 at the end of the list of retrieved documents as at the beginning. It is thus not necessary to set a  
122 threshold to determine whether a document should be retrieved. The authors see this as very  
123 important, since they argue that the value of the threshold affects the behaviour of the retrieval  
124 operation. If the value is too low, many irrelevant documents will be retrieved. If the value is too  
125 high, very few documents may be retrieved.

126 The fitness function that they employ (Chang & Hsu, 1999; Kwok, 1997) is constructed as  
127 follows. One calculates the similarity of the query vector with all the documents using the inner  
128 product. The documents are sorted by decreasing similarity, and finally the chromosome's fitness  
129 is given by the formula:

$$F = \frac{1}{|D|} \sum_{i=1}^{|D|} \left( r(d_i) \sum_{j=1}^{|D|} \frac{1}{j} \right),$$

131 where  $|D|$  is the total number of documents retrieved, and  $r(d)$  is a function giving the relevance of  
 132 the documents  $d$ , being unity if the document is relevant and zero if it is irrelevant. This function is  
 133 our *fitness 8*.

134 Another approach is that of Martín-Bautista and co-workers (Martín-Bautista, Vila, & Larsen,  
 135 1999; Martín-Bautista, 2000). While this does not follow the vector space model, we find it very  
 136 interesting in its contribution of new ideas, amongst them two new fitness functions. The authors  
 137 present *GIRAF* (Genetic Information Retrieval Agent Filter), an intelligent agent that can work  
 138 off-line filtering documents retrieved from Internet. It is based on a GA with fuzzy genes that  
 139 perform the adaptive learning of the user's requirements.

140 In this algorithm the chromosome population represents the user's possible preferences, and the  
 141 different preferences compete to obtain the best representation of the said requirements. To cal-  
 142 culate an individual's fitness, the authors propose in the first work (Martín-Bautista et al., 1999)  
 143 that, since the chromosome is a set of genes, this value should be calculated as some aggregation  
 144 class of fuzzy values of each gene. In particular, the authors used the arithmetic mean of the genes  
 145 as follows:

$$C_i^j(\omega) = \frac{1}{l} \cdot \sum_{h=1}^l \mu_g^h(x), \quad i = 1, \dots, m, \quad j = 1, \dots, k,$$

147 where  $C_i(\omega)$  is the score of the  $i$ th chromosome on document  $\omega$  in generation  $j$ ,  $l$  is the length of  
 148 the chromosome, and  $\mu_g^h$  is the value of the membership function of the  $h$ th gene of type  $g$ .

149 This function, modified to suit our experiments in the vector space model, will be our *fitness 9*.

150 In their following work (Martín-Bautista, 2000), the author performs another experiment with  
 151 a real representation as well as an experiment with a fuzzy representation, with the main aim of  
 152 comparing the two methods. For the fitness function in the real case, they use the following  
 153 combination of the classical measures of precision and recall:

$$F = v \cdot \text{recall} + (1 - v) \cdot \text{precision}$$

155 since it seems reasonable that precision should have greater importance than recall (in the trials  
 156 the value of  $v$  is set at 0.4). From this we obtain our *fitness 10*.

157 Besides the work on genetics that we have consulted, we cannot end this section without  
 158 mentioning the work on relevance feedback of Salton and Buckley (1990), which examines in  
 159 depth six traditionally used methods. We have taken this work as a model for our experiments on  
 160 feedback, and have also compared the behaviour of our GA's with the method that gave the best  
 161 results in that earlier study – the Ide dec-hi method (Ide, 1971).

162 This method consists simply in adding directly to the original query terms the terms of all the  
 163 relevant documents of the set of documents provided for feedback (in our case 15), and removing  
 164 those of the first irrelevant document obtained in the retrieval that belongs to the said set.

165 Formally, the modified query vector is the following:

$$Q' = Q + \sum_{\text{all relevant}} D_i - S,$$

167 where  $Q$  is the original query vector;  $D_i$  is the vector of the relevant document  $i$ ; and  $S$  is the vector  
168 of the first irrelevant document in the ranking.

### 169 3. Experiment and evaluation

170 As we mentioned above, one of our primary goals was to compare the behaviour of the dif-  
171 ferent GA's with each other and with some traditional method, in particular with the Ide dec-hi  
172 method, which was the method that obtained the best results in the study of Salton and Buckley  
173 (1990).

174 To this end, it was necessary to generate a trial database, created from a test collection – the  
175 *Cranfield* collection. A prime motive for choosing this collection is that it has been used in  
176 feedback and GA experiments on many occasions, so that it offers the greatest possibility for  
177 comparing results.

178 The collection consists of 1398 documents on aspects of aeronautical engineering, and 225  
179 queries for which relevance judgements are known. From these, we had to make a selection of  
180 queries that are suitable for testing the relevance feedback technique that is the subject of the  
181 present work.

182 We initially set the number of documents for the implementation of feedback to 15 (we also ran  
183 trials with 10, 20, and 25 documents). I.e., for each query, the first 15 documents were examined  
184 for their relevance, and this information was used as feedback to the algorithm.

185 Not all the queries included in the collection were suitable for testing this technique however.  
186 Those which retrieve either all or none of the relevant documents are of no use for our purposes.  
187 We hence chose for the evaluation a set of (33) queries which had at least 3 relevant documents  
188 retrieved in the first 15, and lacked at least 5 relevant documents which still remained to be re-  
189 trieved.

### 190 4. Document vectorization

191 Firstly, we performed the following steps to determine the terms that we would use to describe  
192 the documents of the collection:

193 1. Extracting all the words from each document.

194 2. Removal of stopwords, using the list of stopwords generated with the frequency dictionary of  
195 Kucera and Francis (1967).

196 3. Stemming the rest of the words, using the *Porter Stemmer* which is the most commonly used  
197 stemmer in English (Frakes & Baeza-Yates, 1992; Porter, 1980).

198 After applying this process to the documents of the *Cranfield* collection, we were left with 4307  
199 terms, so that we shall be working with 1398 document vectors of 4307 components.

200 We assigned weights to the terms using the scheme proposed by Salton and Buckley (1990), so  
201 as to equiparate our experiment as closely as possible with theirs. The formula is

$$a_{ij} = \frac{(0.5 + 0.5(tf_{ij}/\max tf)) \log(N/ni)}{\sqrt{(0.5 + 0.5(tf_{ij}/\max tf))^2 (\log(N/ni))^2}},$$

203 where  $a_{ij}$  is the weight assigned to the term  $t_j$  in document  $D_i$ ,  $tf_{ij}$  is the number of times the term  $t_j$   
 204 appears in document  $D_i$ ,  $n_j$  is the number of documents indexed by the term  $t_j$ , and  $N$  is the total  
 205 number of documents in the document database.

206 Lastly, we decided to normalize the vectors by dividing them by their Euclidean norm, since the  
 207 study of Noreault, McGill, and Koll (1981) showed that the best similarity measures are those  
 208 that compare angles between vectors, and we performed a similar process to the above for the  
 209 queries associated with each collection, obtaining the normalized query vectors which we shall try  
 210 to optimize with relevance feedback.

## 211 5. Experimental design

212 The scheme of the experiment implementing the relevance feedback technique by means of the  
 213 various methods is very simple (López-Pujalte, 2000):

- 214 • Each query of the collection is compared with all the documents in the collection, using the co-  
 215 sine as similarity measure. We thus obtain a list of similarities of each query with all the doc-  
 216 uments.
- 217 • This list is sorted by degree of similarity.
- 218 • The normalized document vectors corresponding to the first 15 documents of the list, with their  
 219 relevance judgements and the normalized query vector, are supplied as input to the algorithm  
 220 responsible for the relevance feedback procedure.
- 221 • The program also generates as output a masking file which contains for each query all the doc-  
 222 uments which are not to be considered in the evaluation process (these will be the first 15 which  
 223 have been used in modifying the queries), since we shall be following the residual collection  
 224 method used by Salton and Buckley (1990).

225 We used recall and precision to evaluate the results. To this end we interpolated the precision at  
 226 fixed intervals of recall (0.1 in value) as described by Salton and McGill (1983), and calculated the  
 227 average of the precision at all the points of recall as was done by Salton and Buckley (1990) in  
 228 their study of feedback, so as to be able to compare the two systems.

229 As we mentioned above, we used the method of *residual collection*, in which all the documents  
 230 previously seen by the user (whether relevant or not) are removed from the collection, and both  
 231 the initial query and the modified query are evaluated on this residual collection, since the rele-  
 232 vance feedback operation must be judged for its ability to retrieve new documents.

233 With respect to the design of the GA, we experimented with several alternatives or improve-  
 234 ments to the classical model (Herrera, Lozano, & Verdegay, 1995; Herrera, Lozano, & Verdegay,  
 235 1998):

- 236 • Different sampling mechanisms: simple universal sampling (Goldberg, 1989; Holland, 1992),  
 237 and stochastic universal sampling (Baker, 1987), both of them with or without elitism (De Jong,  
 238 1975).

- 239 • Types of crossover: simple crossover (Goldberg, 1989; Holland, 1992; Michalewicz, 1995;  
240 Wright, 1991), multi-point crossover (De Jong & Spears, 1992; Eshelman, Caruana, & Schaffer,  
241 1989), and another quite different type, natural crossover, used in the work of Hornig and Yeh  
242 (2000).
- 243 • Mutations: random mutations (Radcliffe, 1991; Michalewicz, 1995), and random mutations  
244 within the interval  $[-1, 1]$ , since the GA of the work of Robertson and Willett (1996) showed  
245 improved behaviour when negative weights were introduced.
- 246 • Proposal of other operators: since the mean of the vectors of the final population gave the best  
247 results in some cases, we implemented an averaging operator that was applied alternatively with  
248 the crossover operator (although with a lower probability).
- 249 • Modification of the fitness function  $f$ : in order at times to accentuate the differences in some  
250 evaluations that gave very similar results, and to smooth out the differences of others, we scaled  
251 the fitness function as  $f^2$ ,  $f^3$ ,  $\sqrt{f}$  and  $10^f$ .
- 252 • Adaptive techniques: we also used an adaptive technique to adjust the two control parameters,  
253 the crossover probability ( $p_c$ ) and the mutation probability ( $p_m$ ), during the execution of the al-  
254 gorithm, in particular the method denoted by Yang and Korfhage as the “hybrid method” in  
255 which  $p_c$  falls from 0.9 to 0.6 (by 0.05 each three generations) and  $p_m$  rises simultaneously from  
256 0.01 to 1, although  $p_m$  is reduced rapidly in the last three generations so as not to lose the good  
257 solutions (Yang & Korfhage, 1994).
- 258 • Hybridization: we used Hybrid Genetic Algorithms (Davis, 1991).  
259 We also carried out numerous trials with different values of the control parameters, since these  
260 were obtained experimentally.

## 261 6. Results

- 262 After performing the experiments described above, we had as the result that the relevance  
263 feedback GA that showed the best performance presented the following characteristics:
- 264 • It is a hybrid GA, i.e., a GA which is the result of incorporating the information associated  
265 with another type of algorithm specific to the problem at hand (Davis, 1991), and which re-  
266 ceives as an initial population the most numerous and varied of all those that we experimented  
267 with. The population consists of the chromosomes corresponding to the relevant documents, to  
268 the irrelevant documents, to the latter with their terms negated, and to the optimized query.  
269 These document vectors were converted into chromosomes using the procedure presented by  
270 Chen (1995) in which the chromosomes are formed only by the terms of those first 15 docu-  
271 ments which are different from zero (as well as those of the original query). First one calculates  
272 the set of terms contained in those documents. The size of the chromosomes is equal to the  
273 number of terms in the said set.
  - 274 • As a selection mechanism it uses simple universal sampling with elitism. Simple universal sam-  
275 pling is based on constructing a roulette wheel with as many slots as there are individuals in the  
276 population, and with each slot’s width being proportional to the value of the individual’s fitness  
277 (Holland, 1992). Elitism (De Jong, 1975) consists in introducing the best element of a genera-  
278 tion into the next, if that element has been eliminated by chance.
  - 279 • As genetic operators it uses simple one-point crossover and random mutation.

- 280 • It was notable that the control parameters  $p_c$  and  $p_m$  were considerably higher than those which  
 281 are normally used, especially  $p_m$  ( $p_c = 0.8$  and  $p_m = 0.2$ ), since it is this which helps to maintain  
 282 the population's diversity and to avoid premature convergence of the algorithm.
- 283 • We also found that including a process to normalize all the population's chromosomes each  
 284 time that the genetic operators were applied to them improved the GA's performance (even  
 285 though the chromosomes came from documents that had already been normalized).
- 286 • Finally, the GA sometimes returns the best chromosome found during the process, and at other  
 287 times returns the centroid of the chromosomes of the final population whose fitness is equal to  
 288 the maximum value found in that generation, depending on the fitness function that was used.
- 289 • The maximum number of generations is 20, since from then on the population no longer im-  
 290 proves.
- 291 Having found the GA that gave the best relevance feedback results, we set out to investigate  
 292 further the fitness function. We had found that this was the decisive element in giving a well-  
 293 behaved GA, since the rest of the alternatives cited above led to minimal variations in the results  
 294 given by the algorithm, and which were in no case significant. The variations were significant,  
 295 however, when we changed the fitness function used by the GA. We implemented all the fitness  
 296 functions presented in the literature for this type of experiment, with the necessary adaptations for  
 297 their use in the present setup.
- 298 The final results are listed in Table 1, giving the mean precision for each of the algorithms  
 299 implemented, and the percentage improvement with respect to the initial unoptimized query. With  
 300 respect to the manner the GA calculates the solution, as we mentioned above it depended on the  
 301 fitness function that was used, with there being two alternatives as shown in the table:
- 302 • The “best” solution: the best chromosome found during the entire process.  
 303 • The “centroid” solution: the centroid of the chromosomes of the final population whose fitness  
 304 is the maximum value calculated for that population.

Table 1  
 Results of different relevance feedback methods for the Cranfield collection

Method	Best		Centroid	
	Mean	Improve (%)	Mean	Improve (%)
No feedback	0.098			
Ide (dec-hi)	0.218	120.8		
GA's				
GA with fitness 1	0.218	120.8	0.151	54
GA with fitness 2	0.218	120.8	0.154	57.1
GA with fitness 3	0.043	–	0.043	–
GA with fitness 4	0.060	–	0.030	–
GA with fitness 5	0.053	–	0.053	–
GA with fitness 6	0.148	50.7	0.120	22.4
GA with fitness 7	0.218	120.8	0.150	53
GA with fitness 8	0.218	120.8	0.220	123.6
GA with fitness 9	0.054	–	0.054	–
GA with fitness 10	0.218	120.8	0.151	54

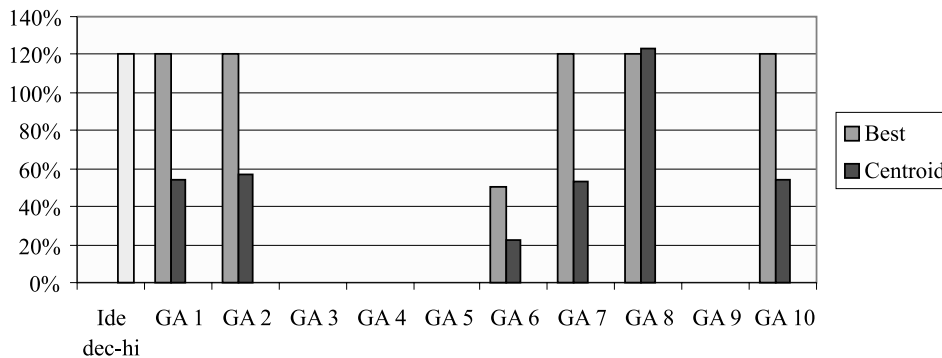


Fig. 2. Improvement percentages with respect to the original query as obtained by different relevance feedback methods for the Cranfield collection.

305 One sees that the results varied greatly depending on the fitness function that was used. The  
 306 results range from yielding no improvement in the GA's which use the fitness function used by  
 307 Chen and its variants, and that used in Martín-Bautista et al. (1999) (possibly in the latter case  
 308 because the function was designed by the authors for a fuzzy model, not a vectorial model), to  
 309 achieving an improvement of 123.6% with the fitness function 8.

310 The relevance feedback technique with the best performance was therefore a genetic technique,  
 311 in particular that which used function 8 designed by Horng and Yeh (2000).

312 As can be seen in Table 1, in most cases the GA gives the best results with the former method  
 313 (the best). The exception is the GA that uses the fitness function 8, which gave the best results  
 314 using the centroid method.

315 As can be seen in Fig. 2, in the Cranfield collection there is a 120% improvement in many of the  
 316 methods produced by relevance feedback with respect to the original query. The GA which uses  
 317 the fitness function 8 and the centroid solution even surpasses the classical Ide dec-hi method. The  
 318 percentage improvement with this GA was 123.6%.

## 319 7. Conclusions

320 GA's potentially have a major field of application in IR. In the case dealt with here, relevance  
 321 feedback, the idea was to optimize the query to improve the results of the retrieval. On applying a  
 322 GA to this technique, one can start out from a set of possible solutions (queries) that evolves  
 323 under the algorithm with the aim of achieving optimization. The residual collection method of  
 324 Salton and Buckley (1990) was found to be particularly well suited to evaluating the results of this  
 325 process.

326 To guide the evolution of the possible solutions, one has to design fitness functions that allow  
 327 the goodness of the solutions to be evaluated, using for the purpose the information coming from  
 328 the user's feedback. It was in these fitness functions where the experiments most varied from one  
 329 to another, so that these functions are the key to achieving a good level of improvement.

330 We tested here the different functions that have been described in the literature, mainly those  
331 that have been applied with the vector model, and others which, while not designed for this model,  
332 we adapted to it.

333 The results showed that these algorithms allow a considerable improvement of the original  
334 query, implementing the technique of relevance feedback (in a single iteration). The improvement  
335 was from 50.7% to 123.6% (which corresponded to the best method acting on the Cranfield  
336 collection, the GA that used function 8), surpassing even the improvement obtained with the  
337 classical Ide dec-hi method which gave an improvement of 120.8%. The Ide dec-hi method is the  
338 best of the classical methods used in relevance feedback, according to the study of Salton and  
339 Buckley (1990).

340 As we have already mentioned, it is worth noting that including a process of normalization of  
341 all the chromosomes each time that the genetic operators are applied to them (even though the  
342 chromosomes came from normalized document vectors) improved the behaviour of the GA.

343 But one can draw the conclusion that the greatest variation in the results of the different al-  
344 gorithms depends on the fitness function that is used, which is therefore of prime importance.  
345 Which function is used can lead to success or to utter failure in the exploration. Amongst the  
346 functions that were tested, the one that gave the best performance took into account not only  
347 which documents were retrieved, but also the order in which they were retrieved. I.e., one wants  
348 fitness functions that value not just that the possible solution retrieves many relevant documents  
349 and few irrelevant documents, but also values whether the relevant documents are at the begin-  
350 ning or end of the list.

351 In light of the results, we can state that, although there have been several works using GA's to  
352 implement relevance feedback, very few have contrasted the results with other more classical  
353 methods of IR, and perhaps have celebrated too much the major improvement given by the GA's  
354 without taking into account that this improvement had already been achieved with some of those  
355 traditional methods.

356 As a future line of research, we shall be looking into the algorithm's search procedure, so that  
357 we shall continue investigating into the design of fitness functions, laying particular emphasis on  
358 procedures which also take the order into account. It is this aspect that we consider to be the key  
359 to the performance of this type of GA.

## 360 Acknowledgements

361 This work was financed by the Junta de Extremadura-Consejería de Educación Ciencia &  
362 Tecnología and the Fondo Social Europeo, as part of the research project IPR99A047.

## 363 References

- 364 Baker, J. E. (1987). Reducing bias and inefficiency in the selection algorithm. In J. J. Grefenstette (Ed.), *Proceedings of*  
365 *the second international conference on genetic algorithms and their applications* (pp. 14–21). Hillsdale, MA: Erlbaum  
366 (Lawrence).

- 367 Frakes, W. B., & Baeza-Yates, R. (1992). In W. B. Frakes, & R. Baeza-Yates (Eds.), *Information retrieval: Data*  
368 *structures & algorithms*. Englewood Cliffs, NJ: Prentice-Hall.
- 369 Belew, R. K. (1989). Adaptive information retrieval. In *Proceedings of the Association for Computing Machinery Special*  
370 *Interest Group on Information Retrieval (ACM/SIGIR) 12th annual international conference on research and*  
371 *development in information retrieval, June 25–28, Cambridge, MA* (pp. 11–20). New York, NY: ACM (ISBN 0-  
372 89791-321-3).
- 373 Chang, C. -H., & Hsu, C. -C. (1999). *The design of an information system for hypertext retrieval and automatic discovery*  
374 *on WWW*. PhD Thesis, Department of CSIE, National Taiwan University.
- 375 Chen, H. (1995). Machine learning for information retrieval: neural networks, symbolic learning, and genetic  
376 algorithms. *Journal of the American Society for Information Science*, 46(3), 194–216.
- 377 Chen, H., Chung, Y., & Ramsey, M. (1998). A smart itty bitsy spider for the web. *Journal of the American Society for*  
378 *Information Science*, 49(7), 604–618.
- 379 Chen, H., & Iyer, A. (1998). A machine learning approach to inductive query by examples: an experiment using  
380 relevance feedback, ID3, genetic algorithms, and simulated annealing. *Journal of the American Society for*  
381 *Information Science*, 49(8), 693–705.
- 382 Cordon, O., Moya, F., & Zarco, M. C. (1999). Breve estudio sobre la aplicación de los algoritmos genéticos a la  
383 recuperación de la información. In *IV Congreso ISKO (Granada)* (pp. 179–186).
- 384 Davis, L. (Ed.). (1991). *Handbook of genetic algorithms*. New York: Van Nostrand Reinhold.
- 385 De Jong, K. A. (1975). *An analysis of the behavior of a class of genetic adaptive systems*. PhD Thesis, Universidad de  
386 Michigan.
- 387 De Jong, K. A., & Spears, W. M. (1992). A formal analysis of the role of multi-point crossover in genetic algorithms.  
388 *Annals of Mathematics and Artificial Intelligence*, 5(1), 1–26.
- 389 Eshelman, L. J., Caruana, A., & Schaffer, J. D. (1989). Biases in the crossover landscape. In J. D. Schaffer (Ed.),  
390 *Proceeding of the third international conference on genetic algorithms* (pp. 86–91). San Mateo, CA: Morgan  
391 Kaufmann.
- 392 Glover, D. E. (1987). Solving a complex keyboard configuration problem through a generalized adaptive search. In L.  
393 Davis (Ed.), *Genetic algorithms and simulated annealing* (pp. 12–31). San Mateo, CA: Morgan Kaufmann.
- 394 Goldberg, D. E. (1989). *Genetic algorithms in search, optimization and machine learning*. Reading, MA: Addison-  
395 Wesley.
- 396 Gordon, M. D. (1988a). Probabilistic and genetic algorithms for document retrieval. *Communications of ACM*, 31(10),  
397 1208–1218.
- 398 Gordon, M. D. (1988b). The necessity for adaptation in modified Boolean document retrieval systems. *Information*  
399 *Processing and Management*, 24(3), 339–347.
- 400 Grefenstette, J. J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems,*  
401 *Man, and Cybernetics*, SMC-16(1), 122–128.
- 402 Grefenstette, J. J. (1987). Incorporating problem specific knowledge into genetic algorithms. In L. Davis (Ed.), *Genetic*  
403 *algorithms and simulated annealing* (pp. 42–60). Los Altos, CA: Morgan Kaufmann.
- 404 Guerrero Bote, V. P., Moya Anegón, F., & Herrero Solana, V. (2001). Document organization using Kohonen's  
405 algorithm. *Information Processing and Management* (in press).
- 406 Herrera, F., Lozano, M., & Verdegay, J. L. (1995). Algoritmos genéticos: fundamentos, extensiones y aplicaciones.  
407 *Arbor, CLII*, 597, 9–40.
- 408 Herrera, F., Lozano, M., & Verdegay, J. L. (1998). Tackling real-coded genetic algorithms: operators and tools for  
409 behavioural analysis. *Artificial Intelligence Review*, 12, 265–319.
- 410 Hilliard, M. R., & Liepins, G. E. (1987). A classifier-based system for discovering scheduling heuristics. In *Genetic*  
411 *algorithms and their applications: Proceedings of the second international conference on genetic algorithms* (pp. 231–  
412 235).
- 413 Holland, J. H. (1992). *Adaptation in natural and artificial systems* (2nd ed.). Cambridge, MA: MIT Press.
- 414 Horng, J.-T., & Yeh, C.-C. (2000). Applying genetic algorithms to query optimization in document retrieval.  
415 *Information Processing and Management*, 36, 737–759.
- 416 Ide, E. (1971). New experiments in relevance feedback. In G. Salton (Ed.), *The SMART retrieval system* (pp. 337–354).  
417 Englewood Cliffs, NJ: Prentice-Hall.

- 418 Kucera, H., & Francis, N. (1967). *Computational analysis of present-day American English*. Providence, RD: Brown  
419 University Press.
- 420 Kwok, K. L. (1989). A neural network for probabilistic information retrieval. In N. J. Belkin, & C. J. Van Rijsbergen  
421 (Eds.), *Proceedings of the twelfth annual international ACM/SIGIR conference on research and development in*  
422 *information retrieval, Cambridge, MA* (pp. 21–30). New York: ACM Press.
- 423 Kwok, K. L. (1990). Application of neuronal network to information retrieval. In *International Joint Conference on*  
424 *Neuronal Networks, Washington*.
- 425 Kwok, K. L. (1997). *Comparing representations in Chinese information retrieval* (pp. 34–41). Philadelphia, PA, USA:  
426 ACM/SIGIR.
- 427 López-Pujalte, C. (2000). *Algoritmos genéticos aplicados a la retroalimentación por relevancia*. PhD Thesis, Facultad de  
428 Biblioteconomía y Documentación, Universidad de Granada.
- 429 Martín-Bautista, M. J. (2000). *Modelos de computación flexible para la recuperación de información*. PhD Thesis, E.T.S.  
430 de Ingeniería de Informática, Universidad de Granada.
- 431 Martín-Bautista, M. J., Vila, M. A., & Larsen, H. L. (1999). A fuzzy genetic algorithm approach to an adaptive  
432 information retrieval agent. *Journal of the American Society for Information Science*, 50(9), 760–771.
- 433 Michalewicz, Z. (1995). *Genetic algorithms + data structures = evolution programs*. Berlin: Springer.
- 434 Noreault, T., McGill, M., & Koll, M. B. (1981). A performance evaluation of similarity measures, document term  
435 weighting schemes and representation in a Boolean environment. *Information retrieval research*. London:  
436 Butterworths.
- 437 Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3), 130–137.
- 438 Poulter, A., Morris, A., & Dow, J. (1994). LIS professionals as knowledge engineers. *Annual Review of Information*  
439 *Science and Technology*, 29, 305–350.
- 440 Radcliffe, N. J. (1991). Formal analysis and random respectful recombination. In *Proceedings of the fourth international*  
441 *conference on genetic algorithms* (pp. 222–229). San Mateo, CA: Morgan Kaufmann.
- 442 Raghavan, V. V., & Agarwal, B. (1987). Optimal determination of user-oriented clusters: an application for the  
443 reproductive plan. In *Genetic algorithms and their applications: Proceedings of the second international conference on*  
444 *genetic algorithms and their applications* (pp. 241–246).
- 445 Robertson, A. M., & Willett, P. (1996). An upperbound to the performance of ranked-output searching: optimal  
446 weighting of query terms using a genetic algorithm. *Journal of Documentation*, 52(4), 405–420.
- 447 Robertson, G. (1987). Parallel implementation of genetic algorithms in classification systems. In L. Davis (Ed.), *Genetic*  
448 *algorithms and simulated annealing* (pp. 129–140). San Mateo, CA: Morgan Kaufmann.
- 449 Salton, G., & Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American*  
450 *Society for Information Science*, 41(4), 288–297.
- 451 Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval*. New York: McGraw-Hill.
- 452 Wright, A. (1991). Genetic algorithms for real parameter optimization. In G. J. E. Rawlin (Ed.), *Foundations of genetic*  
453 *algorithms 1* (pp. 205–218). San Mateo, CA: Morgan Kaufmann.
- 454 Yang, J. J., & Korfhage, R. (1994). Query modification using genetic algorithms in vector space models. *International*  
455 *Journal of Expert Systems*, 7(2), 165–191.