A Realistic Information Retrieval Environment to Validate a Multiobjective GA-P Algorithm for Learning Fuzzy Queries*

Oscar Cordón[1], Enrique Herrera-Viedma[1], María Luque[1], Felix Moya[2], and Carmen Zarco[3]

[1] Dept. of Computer Science and A.I. E.T.S. de Ingeniería Informática
University of Granada. 18071 - Granada (Spain)
{ocordon,viedma,mluque}@decsai.ugr.es
[2] Dept. of Information Sciences. Faculty of Information Sciences
University of Granada. 18071 - Granada (Spain) felix@ugr.es
[3] PULEVA S.A.
Camino de Purchil, 66. 18004 - Granada (Spain) czarco@puleva.es

Summary. IQBE has been shown as a promising technique to assist the users in the query formulation process. In this framework, queries are automatically derived from sets of documents provided by them. However, the different proposals found in the specialized literature are usually validated in non realistic information retrieval environments. In this work, we design several experimental setups to create real-like retrieval environments and validate the applicability of a previously proposed multiobjective evolutionary IQBE technique for fuzzy queries on them.

1 Introduction

Information retrieval (IR) may be defined as the problem of the selection of documentary information from storage in response to search questions provided by a user [2]. Information retrieval systems (IRSs) deal with documentary bases containing textual, pictorial or vocal information and process user queries trying to allow the user to access to relevant information in an appropriate time interval.

The paradigm of Inductive Query by Example (IQBE) [4], where queries describing the information contents of a set of documents provided by a user are automatically derived, has proven to be useful to assist the user in the query formulation process. This is especially useful for fuzzy IRSs [3], as they

* This work was supported by the Spanish Ministerio de Ciencia y Tecnologa under projects TIC2003-07977 and TIC2003-00877, including FEDER fundings.

consider complex queries composed of weighted query terms joined by the logical operators AND and OR, which are difficult to be formulated by non expert users.

The most known existing approach is that of Kraft et al. [15], based on genetic programming (GP) [14]. Several other approaches have been proposed based on more advanced evolutionary algorithms (EAs) [1], such as genetic algorithm-programming (GA-P) [12] or simulated annealing-programming, to improve Kraft et al.'s [7, 8].

In view of the latter, the IQBE paradigm seems to perform properly but most of the existing proposals are validated by only running the algorithms on a whole document collection selected from those typical in IR, as Cranfield, CISI, etc. This does not show the real environment in which an IRS will be used.

In this paper, we will design real-like environments to test these kinds of algorithms by: i) dividing the documentary base (Cranfield, in our case) into a training document set, in which the queries will be learned, and a test document set, against to which the obtained queries will be tested; and ii) using training collections with a number of irrelevant documents that shows the real behaviour of the users in an IQBE or a user profile [13] environment. Then, we will validate a specific algorithm able to generate several queries with a different trade-off between precision and recall in a single run, the multiobjective GA-P IQBE technique proposed in [9], in the designed IR environments.

The paper is structured as follows. Section 2 is devoted to the preliminaries, the basis of FIRSs and a short review of IQBE. Then, Section 3 describes the design of realistic IR test environments. The multiobjective GA-P proposal is reviewed in Section 4. Section 5 presents the experiments developed and the analysis of results. Finally, the conclusions are pointed out in Section 6.

2 Preliminaries

2.1 Fuzzy Information Retrieval Systems

FIRSs are constituted of the following three main components:

The documentary data base, that stores the documents and their representations (typically based on index terms in the case of textual documents).

Let D be a set of documents and T be a set of unique and significant terms existing in them. An indexing function $F : D \times T \rightarrow [0, 1]$ is defined as a fuzzy relation mapping the degree to which document d belongs to the set of documents "about" the concept(s) represented by term t. By projecting it, a fuzzy set is associated to each document ($d_i = \{< t, \mu_{d_i}(t) > | \, t \in T\}$; $\mu_{d_i}(t) = F(d_i, t)$) and term ($t_j = \{< d, \mu_{t_j}(d) > | \, d \in D\}$; $\mu_{t_j}(d) = F(d, t_j)$).

In this paper, we will work with Salton's normalized *inverted document frequency* (IDF) [2]: $w_{d,t} = f_{d,t} \cdot log(N/N_t)$; $F(d,t) = \frac{w_{d,t}}{Max_d \, w_{d,t}}$, where $f_{d,t}$

is the frequency of term t in document d, N is the total number of documents and N_t is the number of documents where t appears at least once.

The query subsystem, allowing the users to formulate their queries and presenting the retrieved documents to them. Fuzzy queries are expressed using a query language that is based on weighted terms, where the numerical or linguistic weights represent the "subjective importance" of the selection requirements.

In FIRSs, the query subsystem affords a fuzzy set q defined on the document domain specifying the degree of relevance (the so called *retrieval status value* (RSV)) of each document in the data base with respect to the processed query: $q = \{< d, \mu_q(d) > | \, d \in D\} \quad ; \quad \mu_q(d) = RSV_q(d)$.

The matching mechanism, that evaluates the degree to which the document representations satisfy the requirements expressed in the query (i.e., the RSV) and retrieves those documents that are judged to be relevant to it.

When using the *importance* interpretation [3], the query weights represent the relative importance of each term in the query. The RSV of each document to a fuzzy query q is then computed as follows [17]. When a single term query is logically connected to another by the AND or OR operators, the relative importance of the single term in the compound query is taken into account by associating a weight to it. To maintain the semantics of the query, this weighting has to take a different form according as the single term queries are ANDed or ORed. Therefore, assuming that A is a fuzzy term with assigned weight w, the following expressions are applied to obtain the fuzzy set associated to the weighted single term queries A_w (*disjunctive queries*) and A^w (*conjunctive ones*):

$$A_w = \{< d, \mu_{A_w}(d) > | \, d \in D\} \quad ; \quad \mu_{A_w}(d) = Min\ (w, \mu_A(d))$$
$$A^w = \{< d, \mu_{A^w}(d) > | \, d \in D\} \quad ; \quad \mu_{A^w}(d) = Max\ (1 - w, \mu_A(d))$$

If the term is negated in the query, a negation function is applied to obtain the corresponding fuzzy set: $\overline{A} = \{< d, \mu_{\overline{A}}(d) > | \, d \in D\} \quad ; \quad \mu_{\overline{A}}(d) = 1 - \mu_A(d)$.

Finally, the RSV of the compound query is obtained by combining the single weighted term evaluations into a unique fuzzy set as follows:

$$A\ AND\ B = \{< d, \mu_{A\ AND\ B}(d) > | \, d \in D\}\ ;\ \mu_{A\ AND\ B}(d) = Min(\mu_A(d), \mu_B(d))$$
$$A\ OR\ B = \{< d, \mu_{A\ OR\ B}(d) > | \, d \in D\}\ ;\ \mu_{A\ OR\ B}(d) = Max(\mu_A(d), \mu_B(d))$$

2.2 Inductive Query by Example

IQBE was proposed in [4] as "a process in which searchers provide sample documents (examples) and the algorithms induce (or learn) the key concepts

in order to find other relevant documents". This way, IQBE is a technique for assisting the users in the query formulation process performed by machine learning methods. It works by taking a set of relevant (and optionally, non relevant documents) provided by a user and applying an off-line learning process to automatically generate a query describing the user's needs from that set. The obtained query can then be run in other IRSs to obtain more relevant documents.

3 Real-like IR Environments to Test IQBE Algorithms

As said, the experimental studies developed in most IQBE contributions [4, 6, 7, 8, 15] do not represent a real environment where an IRS will be used. In this work, we aim at designing realistic retrieval environments to test IQBE algorithms.

Two problems are found in the experimental setups usually considered. On the one hand, the document set provided to the IQBE algorithm is the whole documentary collection. In this set, the relevant documents are those which are relevant to the selected query, while the irrelevant documents are the rest of them. Hence, the amount of irrelevant documents is very high (for example, in the first Cranfield query, 1369 of the 1398 documents are irrelevant). This does not represent a realistic environment as when the user provides a set of (relevant and irrelevant) documents, for which he wants to learn the best possible query retrieving them, the amount of irrelevant documents provided uses to be significantly smaller.

On the other hand, the real goal of the query learning system (the derivation of queries modeling the information needs represented by the set of documents provided by the user, which are able to retrieve new relevant documents when applied to a different documentary collection) is not actually tested.

Considering the previous aspects, and following the usual machine learning operation mode, we propose to divide the documentary base into a training document set from which the queries will be learned, and a test document set against to which those queries will be tested. Besides, in order to confront the former aspect, we use training collections with a realistic number of irrelevant documents that matches with the real behaviour of the users. In short, we will implement four different environments – two corresponding to a usual IQBE framework and other two coming from the user profile field –, by considering a different number of irrelevant documents for the training set in each case[4].

These four proposals are analyzed as follows and their characteristics are summarized in the left side of Table 1, where $\%non-rel$ and $\%rel$ refer to the percentages of the whole irrelevant and relevant documents, respectively, included in the training document set.

[4] Notice that two different variants are obtained from each of these four environments by taking two different values for the number of relevant documents considered.

3.1 Classic IQBE Test Environments

We include within this group those environments where the training document set is assigned a number of irrelevant documents similar to that a user would provide in a real IQBE case. It is expected that a normal user can afford up to 30 or 40 irrelevant documents to represent an information need. On the other hand, other real (and very usual) case is that where the user does only provide relevant documents and does not give any irrelevant one at all.

As said, we are working with Cranfield, which has a total of 1398 documents. We have decided to design two different classic IQBE environments: one of them where the training set has a 2% of the whole irrelevant documents (randomly selected) and another where no irrelevant document is included on that set.

Both variants will allow us to check if the amount of documents normally provided by a user to a IQBE process is enough to derive queries satisfying the user's needs or if there is a need of using any additional assistance mechanism.

Table 1. Characteristics of the IR environments designed

	A1/A2	B1/B2	C1/C2	D1/D2
% non-rel	0	2	10	50
% rel	50,25	50,25	50,25	50,25

	A	B	C	D
Pob-Size	200	400	400	800
#Eval	1000	25000	25000	50000

3.2 User Profile-based IQBE Test Environments

These two environments are based on incorporating a large number of irrelevant documents to the training documentary set, a 10% and a 50% of the overall number existing in the whole collection. At first sight, we could think there is a discrepancy with a realistic IR environment, since a user is not able to provide so many irrelevant documents (around 690 when working with Cranfield). However, this operation mode is clearly justified when considering user profiles.

User profile derivation [13] is based on a relevance feedback framework where a user runs queries on an IRS and judges the relevance of the retrieved documents to his informations needs. Then, the system makes use of this information to build a *user profile*, representing the user information needs, considered to enhance the retrieval efficacy of future queries of that user. Hence, the system can store user relevance judgements from different queries in an automatic way. As said in [11], these techniques will be useful for users having a persistent need for the same type of information in order to increase the retrieval effectiveness.

4 A Multiobjective GA-P Algorithm for Automatically Learning Fuzzy Queries

The components of our multiobjective IQBE algorithm to learn fuzzy queries based on the GA-P paradigm [9] are described next.

Coding Scheme: The expressional part (GP part) encodes the query composition – terms and logical operators – and the coefficient string (GA part) represents the term weights, as shown in Figure 1. A real coding scheme is considered for the GA part.

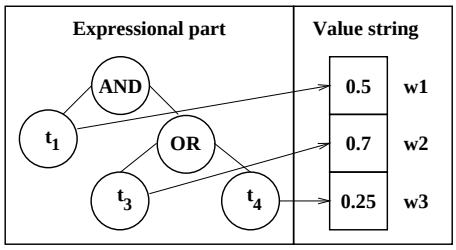

Fig. 1. GA-P individual representing the fuzzy query $0.5\ t_1\ AND\ (0.7\ t_3\ OR\ 0.25\ t_4)$

Fitness Function: The multiobjective GA-P (MOGA-P) algorithm is aimed at jointly optimizing the precision and recall criteria [2], as follows:

$$Max\ P = \frac{\sum_d r_d \cdot f_d}{\sum_d f_d} \quad ; \quad Max\ R = \frac{\sum_d r_d \cdot f_d}{\sum_d r_d}$$

with $r_d \in \{0,1\}$ being the relevance of document d for the user and $f_d \in \{0,1\}$ being the retrieval of document d in the processing of the current query.

Pareto-based Multiobjective Selection and Niching Scheme: The Pareto-based multiobjective EA considered is Fonseca and Fleming's Pareto-based MOGA [5]. Therefore, the selection scheme of our MOGA-P algorithm involves the following four steps:

1. Each individual is assigned a rank equal to the number of individuals dominating it plus one (non-dominated individuals receive rank 1).
2. The population is increasingly sorted according to that rank.
3. Each individual is assigned a fitness value according to its ranking in the population: $f(C_i) = \frac{1}{rank(C_i)}$.
4. The fitness assignment of each group of individuals with the same rank is averaged among them.

Then, a niching scheme is applied in the objective space to obtain a well-distributed set of queries with a different trade-off between precision and recall (see [9] for details). Finally, the intermediate population is obtained by Tournament selection [16].

Genetic Operators: The BLX-α crossover operator [10] is applied twice on the GA part to obtain two offsprings; it ramdoly generates offsprings's genes from the associated neighborhood of the genes in the parents, allowing a suitable balance between exploration and exploitation. Michalewicz's non-uniform mutation operator [16] is considered to perform mutation on that part.

The usual GP crossover [14] is considered for the GP part. Two different mutation operators are applied: random generation of a new subtree, and random change of a query term by another not present in the encoded query.

5 Experiments Developed and Analysis of Results

As said, the documentary set used to design our IR frameworks has been the *Cranfield* collection, composed of 1398 documents about Aeronautics. It has been automatically indexed by first extracting the non-stop words, applying a stemming algorithm, thus obtaining a total number of 3857 different indexing terms, and then using the normalized IDF scheme (see Section 2.1) to generate the term weights in the document representations.

Among the 225 queries associated to the Cranfield collection, we have selected those presenting 20 or more relevant documents (queries 1, 2, 23, 73, 157, 220 and 225). The number of relevant documents associated to each of these seven queries are 29, 25, 33, 21, 40, 20 and 25, respectively.

For each one of these queries and each retrieval environment, the documentary collection has been divided into two different, non overlapped, document sets, training and test, each of them composed of the percentage of relevant and irrelevant documents showed in the left side of Table 1.

MOGA-P has been run ten different times on the training document set associated to each query during a fixed number of evaluations (see the right side of Table 1 [5]). The common parameter values considered are a maximum of 20 nodes for the expression parts, a Tournament size t of 1% of the population size, 0.8 and 0.2 for the crossover and mutation probabilities in both the GA and the GP parts.

The Pareto sets obtained in the ten runs performed for each query have been put together, and the dominated solutions removed from the unified set. Then, five queries well distributed on the Pareto front were selected from each

[5] We have tried with different parameter values choosing those for which the systems behaves better.

of the seven unified Pareto sets and run on the corresponding test set once preprocessed[6].

Table 2. Statistics of the Pareto sets obtained by the MOGA-P algorithm (option C1)

$\#q$	$\#p$	$\sigma_{\#p}$	$\#dp$	$\sigma_{\#dp}$	M_2^*	$\sigma_{M_2^*}$	M_3^*	$\sigma_{M_3^*}$
1	30.7	4.318	2.9	0.263	6.455	1.305	0.601	0.025
2	45.1	5.098	1.8	0.190	8.421	1.192	0.448	0.052
23	34.0	4.825	3.9	0.411	9.216	2.061	0.74	0.042
73	44.3	3.699	1.7	0.145	6.735	1.714	0.338	0.071
157	33.5	5.301	4.8	0.395	8.944	2.046	0.803	0.035
220	36.6	2.021	1.1	0.095	1.308	1.241	0.044	0.041
225	36.9	6.082	2.2	0.19	8.899	2.605	0.485	0.056

Table 3. Statistics of the Pareto sets obtained by the MOGA-P algorithm (option D1)

$\#q$	$\#p$	$\sigma_{\#p}$	$\#dp$	$\sigma_{\#dp}$	M_2^*	$\sigma_{M_2^*}$	M_3^*	$\sigma_{M_3^*}$
1	110.0	10.5	5.3	0.348	39.901	4.574	0.918	0.044
2	127.4	7.462	4.3	0.202	47.023	3.235	0.895	0.033
23	133.8	5.805	6.9	0.170	52.156	3.004	1.042	0.015
73	93.0	12.435	2.6	0.210	24.893	5.133	0.730	0.041
157	118.9	7.886	7.8	0.310	45.264	3.943	1.066	0.006
220	91.1	6.897	1.9	0.221	18.987	4.395	0.437	0.094
225	98.1	6.243	2.3	0.202	22.931	4.266	0.626	0.083

As there is not enough space in the contribution to report every experiment developed, several illustrative results have been selected to be showed. Tables 2 and 3 collect several data about the composition of the ten Pareto sets generated for each query in environments C1 and D1, always showing the averaged value and its standard deviation. From left to right, the columns contain the number of non-dominated solutions obtained ($\#p$), the number of different objective vectors (i.e., precision-recall pairs) existing among them ($\#dp$), and the values of two of the usual multiobjective metrics $\mathcal{M}_2^*$ and $\mathcal{M}_3^*$ [5].

On the other hand, Table 4 shows the retrieval efficacy of the five queries selected from the unified Pareto sets for several Cranfield queries in three of the retrieval environments (B2, C1 and D1). In that table, Sz stands for the query size, P and R for the precision and recall values, and $\#rr/\#rt$

[6] As the index terms of the training and test documentary bases can be different, there is a need to translate training queries into test ones, removing those terms without a correspondence in the test set.

for the number of relevant and retrieved documents, respectively. Finally, the following subsections summarize the conclusions drawn in the different experiments developed.

Table 4. Retrieval efficacy of the selected queries on the training and test collections

			Training set				Test set			
$\#q$			Sz	P	R	$\#rr/\#rt$	Sz	P	R	$\#rr/\#rt$
B2	1	1	19	1.0	1.0	7/7	19	0.036	0.364	8/221
		2	19	1.0	1.0	7/7	19	0.063	0.455	10/158
		3	19	1.0	1.0	7/7	19	0.057	0.500	11/192
		4	19	1.0	1.0	7/7	19	0.062	0.500	11/176
		5	15	1.0	1.0	7/7	15	0.094	0.636	14/149
	225	1	19	1.0	1.0	6/6	17	0.026	0.211	4/151
		2	17	1.0	1.0	6/6	13	0.043	0.263	5/117
		3	17	1.0	1.0	6/6	17	0.021	0.211	4/193
		4	19	1.0	1.0	6/6	19	0.050	0.579	11/220
		5	15	1.0	1.0	6/6	15	0.045	0.579	11/244
C1	2	1	19	0.923	1.0	12/13	11	0.096	0.769	10/104
		2	19	0.923	1.0	12/13	17	0.131	0.846	11/84
		3	19	0.923	1.0	12/13	19	0.133	0.769	10/75
		4	19	0.923	1.0	12/13	19	0.110	0.846	11/100
		5	19	1.0	0.833	10/10	17	0.046	0.231	3/65
	225	1	19	0.923	1.000	12/13	17	0.016	0.077	1/61
		2	19	0.923	1.000	12/13	19	0.000	0.000	0/52
		3	19	1.000	0.917	11/11	13	0.017	0.077	1/60
		4	19	0.800	1.000	12/15	17	0.016	0.077	1/61
		5	17	1.000	0.833	10/10	11	0.021	0.077	1/48
D1	157	1	19	0.299	1.0	20/67	15	0.195	0.8	16/82
		2	19	0.39	0.8	16/41	19	0.119	0.25	5/42
		3	19	0.593	0.8	16/27	17	0.3	0.3	6/20
		4	19	0.789	0.75	15/19	15	0.25	0.15	3/12
		5	19	1.0	0.5	10/10	15	0.375	0.15	3/8
	225	1	17	0.324	1.0	12/37	15	0.0	0.0	0/33
		2	17	0.324	1.0	12/37	15	0.0	0.0	0/33
		3	19	0.579	0.917	11/19	11	0.0	0.0	0/15
		4	19	0.688	0.917	11/16	15	0.0	0.0	0/8
		5	19	1.0	0.917	11/11	15	1.0	0.077	1/1

5.1 Classic IQBE *Versus* User Profile Test Environments

In the two classic IQBE environments, A and B, both the precision and the recall of every learned query is always equal to 1 in the training set, regardless the number of relevant documents. Besides, both values are also very close to 1 in most of the cases in the user profile-based evironment C (see B2 and C1

in Table 4). However, in the other user profile-based framework D, it is very difficult to find a query with both recall and precision equal to 1 (see D2 in Table 4). Hence, as the number of irrelevant documents increases, it is more difficult for the learned query to only retrieve relevant documents.

On the other hand, in the test results, as the number of irrelevant documents in the training set increases, the precision values also increase, whereas the recall values diminish. The reason is that when there are a lot of irrelevant documents in the test set (a few of them in the training set), the queries get all the relevant documents by retrieving a very large number of documents, thus obtaining a very low precision. However, there are cases in option D where the query does not retrieve any document or just one, showing a usual machine learning phenomenon called *over-learning* (see the results for query 225 in C1 and D1 in Table 4).

5.2 50% Relevant Documents *Versus* 25% Relevant Documents

In both classic IQBE environments (A and B), there is no significant differences between option 1 (50% of the relevant documents in the training set) and 2 (25%). However, in those based on user profiles (C and D), the differences are more significant. In both cases, option 1 with more relevant documents in the training set performs better than the other in the test results. This shows that the larger the number of irrelevant documents in the training set, the more relevant documents are needed to get good queries for the test one.

In addition, option C2 usually presents more queries with precision and recall values equal to 0, having a stronger *over-learning* than C1.

6 Concluding Remarks

Several real-like retrieval environments with different characteristics have been proposed to test IQBE algorithms. The Cranfield collection has been considered to define several retrieval needs and, for each of them, relevant and irrelevant documents have been divided into several training-test partitions with a different number of documents. Then, a previous multiobjective evolutionary IQBE proposal for learning fuzzy queries has been tested in the designed environments analyzing the retrieval efficacy obtained in each case.

As future works, we will use retrieval measures considering not only the absolute number of relevant and non relevant documents retrieved, but also their relevance order in the retrieved document list, which is a fuzzy IR ability very useful for the user.

References

1. T. Bäck, D.B. Fogel, and Z. Michalewicz. *Handbook of Evolutionary Computation.* IOP Publishing and Oxford University Press, 1997.

2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Adisson, 1999.
3. G. Bordogna, P. Carrara, and G. Pasi. Fuzzy Approaches to Extend Boolean Information Retrieval. In P. Bosc and J. Kacprzyk, editors, *Fuzziness in Database Management Systems*, pages 231–274. 1995.
4. H. Chen and et al. A Machine Learning Approach to Inductive Query by Examples: An Experiment Using Relevance Feedback, ID3, Genetic Algoritms, and Simulated Annealing. *Journal of the American Society for Information Science*, 49(8):693–705, 1998.
5. C. A. Coello, D. A. Van Veldhuizen, and G. B. Lamant. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Kluwer Academy Publisher, 2002.
6. O. Cordón, E. Herrera-Viedma, and M. Luque. Evolutionary Learning of Boolean Queries by Multiobjective Genetic Programming. In *Proc. PPSN-VII*, pages 710–719, Granada (Spain), 2002. LNCS 2439.
7. O. Cordón, F. Moya, and C. Zarco. A GA-P Algorithm to Automatically Formulate Extended Boolean Queries for a Fuzzy Information Retrieval System. *Mathware & Soft Computing*, 7(2-3):309–322, 2000.
8. O. Cordón, F. Moya, and C. Zarco. A new Evolutionary Algorithm combining Simulated Annealing and Genetic Programming for Relevance Feedback in Fuzzy Information Retrieval Systems. *Soft Computing*, 6(5):308–319, 2002.
9. O. Cordón, F. Moya, and C. Zarco. Automatic Learning of Multiple Extended Boolean Queries by Multiobjective GA-P Algorithms. In V. Loia, M. Nikravesh, and L. A. Zadeh, editors, *Fuzzy Logic and the Internet.* Springer, 2003. In press.
10. L. J. Eshelman and J. D. Schaffer. Real-coded Genetic Algorithms and Interval-Schemata. In L. D. Whitley, editor, *Foundations of Genetic Algorithms 2*, pages 187–202. 1993.
11. W. Fan, M. D. Gordon, and P. Pathak. Personalization of Search Engine Services for Effective Retrieval and Knowledge Management. In *Proceedings of the 2000 International Conference on Information Systems (ICIS)*, Brisbane, Australia, 2000.
12. L. Howard and D. D'Angelo. The GA-P: A Genetic Algorithm and Genetic Programming Hybrid. *IEEE Expert*, 3(10):11–15, 1995.
13. R.R. Korfhage. *Information Storage and Retrieval.* Wiley, 1997.
14. J. Koza. *Genetic Programming. On the Programming of Computers by Means of Natural Selection.* The MIT Press, 1992.
15. D.H. Kraft, F.E. Petry, B.P. Buckes, and T. Sadasivan. Genetic Algorithms for Query Optimization in Information Retrieval: Relevance Feedback. In E. Sanchez, T. Shibata, and L.A Zadeh, editors, *Genetic Algorithms and Fuzzy Logic Systems*, pages 155–173. 1997.
16. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, 1996.
17. E. Sanchez. Importance in Knowledge Systems. *Information Systems*, 6(14):455–464.