26

A Multi-Objective Genetic Algorithm for Learning Linguistic Persistent Queries in Text Retrieval Environments*

María Luque[1], Oscar Cordón[2], and Enrique Herrera-Viedma[2]

[1] Dept. of Computer Science and N.A.
University of Córdoba. 14071 - Córdoba (Spain)
mluque@uco.es

[2] Dept. of Computer Science and A.I. E.T.S. de Ingeniería Informática
University of Granada. 18071 - Granada (Spain)
{ocordon,viedma}@decsai.ugr.es

Summary. Persistent queries are a specific kind of queries used in information retrieval systems to represent a user's long-term standing information need. These queries can present many different structures, being the "bag of words" that most commonly used. They can be sometimes formulated by the user, although this task is usually difficult for him and the persistent query is then automatically derived from a set of sample documents he provides.

In this work we aim at getting persistent queries with a more representative structure for text retrieval issues. To do so, we make use of soft computing tools: linguistic information is considered for weighting the terms of Boolean queries by means of ordinal linguistic values (linguistic queries), and multiobjective evolutionary algorithms are applied to build the linguistic persistent query. Experimental results will show how using an expressive linguistic information-based query structure and a proper learning process to derive it, we can get more flexible, comprehensible and expressive user profiles.

26.1 Introduction

Persistent queries (PQs) are useful tools for information retrieval system (IRS) users having a relatively specific information need remaining fixed during a certain time period [23, 38]. By the definition of these kinds of queries, the information filtering process can be put into effect by delivering interesting information to a user, thus getting the her or him permanently updated on topics the user is interested [26].

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

Although different structures can be used to represent a PQ, it is usually difficult for a user to formulate the query independent of its structure [23, 24, 38]. Therefore, explicit PQs automatically learned from a training set of documents by means of user's relevance feedback are normally considered in information routing systems.

Soft computing tools have demonstrated to be useful in the personalization of IRSs, providing them with flexibility and some kind of "intelligence". The latter is viewed as the capability of automatically adapting to a context or service based on implicit behavior and learning instead of explicit solicitation from users [14, 22, 39].

One of the ways to add flexibility to an IRS is to make it tolerant to uncertainty and imprecision — both inherent to the user-system interaction — which can be achieved by allowing a more natural expression of users' needs [39]. For example, some flexible query languages based on the application of fuzzy set theory have been proposed which make possible simple and approximate expressions of subjective information needs [6]. In this contribution, we will deal with linguistic queries, considering them to improve the representative power of classic Boolean ones when used as PQ structures.

On the other hand, the IRS self-adaptativeness can be tackled by the machine learning perspective of soft computing, put into effect by evolutionary algorithms [1], neural networks and Bayesian networks, among others. These techniques can be hybridized with the representative power of flexible query languages to get "intelligent" IRSs [14]. In particular, evolutionary algorithms has obtained promising results in IR [15]. We will consider the use of multiobjective evolutionary algorithms [13] to automatically derive several linguistic PQs representing the user's information needs in a single run.

The aim of this contribution is to propose the use of a new, more flexible query structure —the linguistic query— to appropriately represent PQs for text retrieval and to introduce an evolutionary learning process to explicitly derive PQs of this composition. The latter will be based on a multiobjective technique able to automatically generate several PQs with a different trade-off between precision and recall in a single run.

The proposed method will be validated in a simulated text retrieval environment considering seven different information needs extracted from the classic Cranfield collection. Its efficacy will be compared with user profiles derived by one of the state-of-the-art algorithms [23, 24].

This chapter is structured as follows. Section 26.2 is devoted to introduce the preliminaries, including the PQ framework as well as the main aspects of linguistic IRSs and multigranular linguistic information. Then, Section 26.3 presents an IRS based on multigranular linguistic information that accepts linguistic weighted queries. The multiobjective GA algorithm to construct linguistic PQs is described in Section 26.4. Section 26.5 presents the experiments developed to test it and the analysis of results, while the conclusions are pointed out in Section 26.6.

26.2 Preliminaries

26.2.1 Construction of Persistent Queries

A) Information Filtering and Persistent Queries

Information filtering refers to an information seeking process where the user is assumed to be searching for information addressing a specific long-term interest [26, 38].

In an information filtering system, the user's permanent information need is represented in the form of a "profile". The most common profile structure is the "bag of words", which is based on a set of keywords representing the user's interest. Many systems assume an implicit definition of the profile by the user, although this comes with the traditional human-computer interaction "vocabulary problem", involving the difficulty for the user to select the right words to communicate with the system. This is specially important in this case as the profile can neither be too broad —as in that case the information filtering system would retrieve so many non relevant documents— nor too specific —as much valuable information can be lost.

Due to this reason, machine learning techniques have been applied to construct "implicit profiles" [24, 38]. In this case, the profile is automatically learned by the system from a training set of documents provided by the user.

Belkin and Croft suggested that IR techniques can be successfully applied to information filtering [3]. This way, the profile can be represented as a query formulated by using any IR retrieval model [2], the so called PQ [23]. Besides, IR query formulation techniques such as relevance feedback [2] or inductive query by example [12] can be applied in information filtering.

B) Flexible Persistent Queries

As different query structures from different IR retrieval models can be used to represent a PQ, the obtaining of effective retrieval results depends on the user's ability to express his information needs in the form of a query both in information filtering and in IR. It has been shown that the user often does not have a clear picture of what he is looking for and can only represent his information need in vague and imprecise terms, which results in a situation known as fuzzy-querying [37].

Flexible query languages can help to solve this problem due to their capability of personalization. A flexible query language is a language that enables a simple and approximate expression of subjective information needs [39]. For example, different linguistic IR models that use a fuzzy linguistic approach [45] to model the weights of queries and the retrieval status value of documents have been proposed in the literature [7, 8, 9, 29, 30, 31, 33], as we will see in Section 26.2.2.

Therefore, the modeling of user profiles in the form of flexible PQs (in particular, of linguistic PQs) can help us to improve both the comprehensibility and the retrieval efficacy of the obtained PQs.

C) Inductive Query by Example of Persistent Queries

Inductive Query by Example (IQBE) [12] was proposed as "a process in which searchers provide sample documents and the algorithms induce the key concepts in order to find other relevant documents". It works by taking a set of relevant (and optionally, non relevant documents) provided by a user and applying an off-line machine 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.

Hence, IQBE techniques can be directly applied to construct PQs for information filtering, as they work in the same way as explicit profile learning methods. In this contribution, we propose a new IQBE technique, based on a multiobjective genetic algorithm, to derive several flexible PQs with different retrieval efficacy trade-offs in a single run.

26.2.2 Linguistic Information Retrieval Systems

The main activity of an IRS is to gather pertinent archived documents that best satisfy the user queries [2]. IRSs consists of three components:

1.- *A Database:* which stores the documents and the representation of their information contents (index terms).

2.- *A Query Subsystem:* which allows users to formulate their queries by means of a query language.

3.- *An Evaluation Subsystem:* which evaluates the documents for a user query obtaining a Retrieval Status Value (RSV) for each document.

The query subsystem supports the user-IRS interaction, and therefore, it should be able to account for the imprecision and vagueness typical of human communication. This aspect may be modeled by means of introducing weights in the query language. Many authors have proposed weighted IRS models using Fuzzy Set Theory [5, 8, 10, 11, 34, 42]. Usually, they assume numeric weights (values in [0,1]). However, the use of query languages based on numeric weights forces the user to quantify qualitative concepts (such as "importance"), ignoring that many users are not able to provide their information needs precisely in a quantitative form but in a qualitative one.

In fact, it seems more natural to characterize the contents of desired documents by explicitly associating a linguistic descriptor to a term in a query, like "important" or "very important", instead of a numerical value. To this end, some fuzzy linguistic IRS models [7, 33] have been proposed using a fuzzy linguistic approach [45] to model query weights and document scores.

A useful fuzzy linguistic approach which allows us to reduce the complexity of the IRS design [30] is called the ordinal fuzzy linguistic approach [27]. In this approach, the query weights and document scores are ordered linguistic terms, as we will see in the next Section.

26.2.3 Multi-Granular Linguistic Information

An ordinal fuzzy linguistic approach is an approximate technique suited for dealing with qualitative aspects of problems, defined by considering a finite and totally ordered label set $S = s_i, i \in \{0, \ldots, \mathcal{T}\}$ in the usual sense ($s_i \geq s_j$ if $i \geq j$) and with odd cardinality (7 or 9 labels). The mid-term representing an assessment of "approximately 0.5" and the rest of the terms being placed symmetrically around it [4]. The semantics of the label set is established from its ordered structure by considering that each label for the pair $(s_i, s_{\mathcal{T}-i})$ is equally informative. In some approaches [27, 29, 30], the semantics is completed by assigning fuzzy numbers defined on the [0,1] interval to the labels. These membership functions (μ_{s_i}) are described by linear trapezoidal membership functions represented by the 4-tuple $(a_i, b_i, \alpha_i, \beta_i)$ (the first two parameters indicate the interval in which the membership value is 1.0; the third and fourth parameters indicate the left and right widths of the distribution). Furthermore, we require the following operators:

1) Negation: $\text{Neg}(s_i) = s_j$, $j = \mathcal{T} - i$.
2) Maximization: $\text{MAX}(s_i, s_j) = s_i$ if $s_i \geq s_j$.
3) Minimization: $\text{MIN}(s_i, s_j) = s_i$ if $s_i \leq s_j$.

In any linguistic approach, an important parameter to be determined is the *granularity of uncertainty*, i.e., the cardinality of the label set S used to express the linguistic information. The cardinality of S must be small enough so as not to impose useless precision levels to the users, and it must be rich enough in order to allow a discrimination of the assessments in a limited number of degrees.

On the other hand, according to the uncertainty degree that a user qualifying a phenomenon has on it, the label set chosen to provide the user's knowledge will have more or less terms. When different users have different uncertainty degrees on the phenomenon, several label sets with a different granularity of uncertainty are necessary. In the latter case, we need tools for the management of multi-granular linguistic information to model these situations [28].

26.3 The Multi-Granular Linguistic IRS

In this section we review our previous work on an IRS model that accepts linguistic weighted Boolean queries and provides linguistic RSVs expressed using multi-granular linguistic information [31]. Thus, it uses multi-granular linguistic weighted queries and multi-granular linguistic RSVs. Other important property of this IRS is that it models the Boolean operators in a flexible way by means of the OWA operators [44].

Before presenting our algorithm, we provide the basic assumptions in this work. We consider a set of documents D= $\{d_1, \ldots, d_m\}$ represented by means of index terms T= $\{t_1, \ldots, t_l\}$, which describe the subject content of the documents. A numeric indexing function $F : D \times T \rightarrow [0, 1]$ is defined, called *index term weighting*. F maps a given document d_j and a given index term t_i to a numeric weight between 0 and 1. Thus, $F(d_j, t_i)$ is a numerical weight that represents the degree of significance of t_i in d_j. $F(d_j, t_i) = 0$ implies that the document d_j is not at all about the concept(s) represented by the index term t_i and $F(d_j, t_i) = 1$ implies that the document d_j is perfectly represented by the concept(s) indicated by t_i. Using the numeric values in (0,1), F can weight index terms according to their significance in describing the content of a document in order to improve the document retrieval.

26.3.1 Multi-Granular Linguistic Weighted Queries

We assume that each query is expressed as a combination of the weighted index terms which are connected by the logical operators AND ($\wedge$), OR ($\vee$), and NOT ($\neg$), and weighted with ordinal linguistic values. Each term in a query can be simultaneously weighted by means of several weights [29, 30]. Particularly, a term of a query can be weighted by means of three weights associated with different semantics. In such a way, the system is able to support the specification of user's preferences.

Each term of a query can be weighted by means of three weights associated to the following semantics:

1. *Symmetrical threshold semantics* [30]. By associating threshold weights to terms in a query, the user is asking to retrieve all documents about the topics represented by such terms. A symmetric threshold semantics is a special threshold semantics which assumes that a user may employ presence weights or absence weights in the formulation of weighted queries. Then, it is symmetrical with respect to the mid threshold value, i.e., it presents the usual behavior for the threshold values which are on the right of the mid threshold value (presence weights), and the opposite behavior for the values which are on the left (absence weights or presence weights with low values).
2. *Relative importance semantics*. This semantics defines term weights as a measure of the relative importance of each term of a query with respect to the other ones. By associating relative importance weights to terms in a query, the user is asking to see all documents whose content represents to a higher degree the concepts associated to the most important terms than to the less important ones. In practice, this means that the user requires that the computation of the RSV of a document is dominated by the more heavily weighted terms.

3. *Quantitative semantics*. This semantics defines query weights as measures of the quantity of documents that users want to consider in the computation of the final set of documents retrieved for each query term. By associating quantitative weights with the terms in a query, the user is asking to see a set of retrieved documents in which the terms with a greater quantitative weight contribute with a higher number of pertinent documents.

As in [30], we use the linguistic variable *"Importance"* to model every semantics, but with different interpretations. For example, a query term t_i with a threshold weight of value "$High$" means that the user requires documents whose content t_i should have at least a high importance value. However, the same query term t_i with a quantitative weight of value "$High$" means that the user wants a set of documents in which the term t_i contributes with a higher number of pertinent documents; and the same query term t_i with an importance weight of value "$High$" means that the user requires that the meaning of t_i must have a high importance value in the computation of the set of retrieved documents. Therefore, the problem in such a model [30] is that different linguistic weights associated with a term are assessed on the same label set, S. To solve this problem, we propose to represent the linguistic weights using multi-granular linguistic information, i.e., assuming label sets with different cardinality and/or semantics to assess the weights associated with the three semantics, called S^1, S^2 and S^3, respectively.

Then, we assume that a query is any legitimate Boolean expression whose atomic components (atoms) are 4-tuples $< t_i, c_i^1, c_i^2, c_i^3 >$ belonging to the set, $T \times S^1 \times S^2 \times S^3$; $t_i \in T$, $c_i^1 \in S^1$ is a value of the linguistic variable *"Importance"* modeling the symmetrical threshold semantics, $c_i^2 \in S^2$ is a value of the linguistic variable *"Importance"* modeling the quantitative semantics, and $c_i^3 \in S^3$ is a value of the linguistic variable *"Importance"* modeling the relative importance semantics. Therefore, the set of legitimate Boolean queries is a set of multi-granular linguistic weighted queries Q which is defined by the following syntactic rules:

1. $\forall q =< t_i, c_i^1, c_i^2, c_i^3 > \in \mathrm{T} \times S^1 \times S^2 \times S^3 \rightarrow q \in \mathrm{Q}$.
   These queries are called atoms.
2. $\forall q, p \in \mathrm{Q} \rightarrow q \wedge p \in \mathrm{Q}$.
3. $\forall q, p \in \mathrm{Q} \rightarrow q \vee p \in \mathrm{Q}$.
4. $\forall q \in \mathrm{Q} \rightarrow \neg(q) \in \mathrm{Q}$.
5. Every legitimate Boolean query $q \in \mathrm{Q}$ can only be obtained by applying rules 1-4.

26.3.2 Evaluating Multi-Granular Linguistic Weighted Queries

Usually, evaluation methods for Boolean queries work by means of a constructive bottom-up process, i.e., in the query evaluation process, the atoms are evaluated first, then the Boolean combinations of the atoms, and so forth, working in a bottom-up fashion until the whole query is evaluated. Similarly, we propose a constructive bottom-up evaluation method to process the multi-granular linguistic weighted queries. This method evaluates documents in terms of their relevance to queries by supporting the three semantics associated with the query weights simultaneously and by managing the multi-granular linguistic weights satisfactorily. Furthermore, given that the concept of relevance is different from the concept of importance, we use a label set S' to provide the relevance values of documents, which is different from those used to express the queries (S^1, S^2 and S^3).

To manage the multi-granular linguistic weights of queries, we develop a procedure based on the multi-granular linguistic information management tool defined in [28]. This procedure acts making uniform the multi-granular linguistic information before processing queries. To do so, we have to choose a label set as the uniform representation base, called *basic linguistic term set (BLTS)*, and then we have to transform (under a transformation function) all multi-granular linguistic information into that unified label set BLTS. In our case, the choice of the BLTS is easy to perform. It must be the label set used to express the output of the IRS (relevance degrees of documents), i.e., BLTS=S'.

The method to evaluate a multi-granular linguistic weighted query is composed of the following six steps:

1.- Preprocessing of the query.

The user query is preprocessed to put it into either conjunctive normal form (CNF) or disjunctive normal form (DNF), in such a way that every Boolean subexpression must have more than two atoms. Weighted single-term queries are kept in their original forms.

2.- Evaluation of atoms with respect to the symmetrical threshold semantics.

According to a symmetrical threshold semantics, a user may search for documents with a minimally acceptable presence of one term in their representations, or documents with a maximally acceptable presence of one term in their representations [29, 30]. Then, when a user asks for documents in which the concept(s) represented by a term t_i is (are) with the value *High Importance*, he/she would not reject a document with an F value greater than *High*. On the contrary, when a user asks for documents in which the concept(s) represented by a term t_i is (are) with the value *Low Importance*, he/she would not reject a document with an F value less than *Low*. Given a request $< t_i, c_i^1, c_i^2, c_i^3 >$, this means that the query weights that imply the presence of a term in a document $c_i^1 \geq s^1_{\mathcal{T}/2}$ (e.g. *High, Very High*) must be treated differently to the query weights that imply the absence of one term in a document $c_i^1 < s^1_{\mathcal{T}/2}$ (e.g. *Low, Very Low*). Then, if $c_i^1 \geq s^1_{\mathcal{T}/2}$, the request $< t_i, c_i^1, c_i^2, c_i^3 >$ is synonymous with the request $< t_i, \textit{at least } c_i^1, c_i^2, c_i^3 >$, which expresses the fact that the desired documents are those having F values as high as possible; and if $c_i^1 < s^1_{\mathcal{T}/2}$, the former request is synonymous with the request $< t_i, \textit{at most } c_i^1, c_i^2, c_i^3 >$, which expresses the fact that the desired documents are those having F values as low as possible. This interpretation is defined by means of a parameterized linguistic matching function

$g^1 : D\times$ T $\times S^1 \rightarrow S^1$ [29]. Given an atom $< t_i, c_i^1, c_i^2, c_i^3 >$ and a document $d_j \in D$, g^1 obtains the linguistic RSV of d_j, called $RSV_j^{i,1}$, by measuring how well the index term weight $F(d_j, t_i)$ satisfies the request expressed by the linguistic weight c_i^1 according to the following expression:

$$RSV_j^{i,1} = g^1(d_j, t_i, c_i^1) = \begin{cases} s^1_{min\{a+\mathcal{B},\mathcal{T}\}} & \text{if } s^1_{\mathcal{T}/2} \leq s^1_b \leq s^1_a \\ s^1_{max\{0,a-\mathcal{B}\}} & \text{if } s^1_{\mathcal{T}/2} \leq s^1_b \text{ and } s^1_a < s^1_b \\ Neg(s^1_{Max\{0,a-\mathcal{B}\}}) & \text{if } s^1_a \leq s^1_b < s^1_{\mathcal{T}/2} \\ Neg(s^1_{Min\{a+\mathcal{B},\mathcal{T}\}}) & \text{if } s^1_b < s^1_{\mathcal{T}/2} \text{ and } s^1_b < s^1_a \end{cases} \quad (26.1)$$

such that, (i) $s_b^1 = c_i^1$; (ii) s_a^1 is the linguistic index term weight obtained as $s_a^1 = Label(F(d_j, t_i))$, being $Label : [0, 1] \rightarrow S^1$ a function that assigns a label in S^1 to a numeric value $r \in [0, 1]$ according to the following expression:

$$Label(r) = Sup_q\{s_q^1 \in S^1 : \mu_{s_q^1}(r) = Sup_v\{\mu_{s_v^1}(r)\}\}; \quad (26.2)$$

and (iii) $\mathcal{B}$ is a bonus value that rewards/penalizes the value $RSV_j^{i,1}$ for the satisfaction/dissatisfaction of request $< t_i, c_i^1, c_i^2, c_i^3 >$, which can be defined in an independent way, for example as $\mathcal{B} = 1$, or depending on the closeness between $Label(F(d_j, t_i))$ and c_i^1, for example as $\mathcal{B} = round(\frac{2(|b-a|)}{\mathcal{T}})$.

3.- Evaluation of atoms with respect to the quantitative semantics.

In this step, documents is evaluated with regard to their relevance to individual atoms of the query, the restrictions imposed by the quantitative semantics are considered.

The linguistic quantitative weights are interpreted as follows: when a user establishes a certain number of documents for a term in the query, expressed by a linguistic quantitative weight, then the set of documents to be retrieved must have the minimum number of documents that satisfies the compatibility or the membership function associated with the meaning of the label used as linguistic quantitative weight. Furthermore, these documents must be those that better satisfy the threshold restrictions imposed on the term.

Therefore, given an atom $< t_i, c_i^1, c_i^2, c_i^3 >$ and assuming that $RSV_j^{i,1} \in S^1$ represents the evaluation according to the symmetrical threshold semantics for d_j, we model the interpretation of a quantitative semantics by means of a linguistic matching function, called g^2, which is defined between the $RSV_j^{i,1}$ and the linguistic quantitative weight $c_i^2 \in S^2$. Then, the evaluation of the atom $< t_i, c_i^1, c_i^2, c_i^3 >$ with respect to the quantitative semantics associated with c_i^2 for a document d_j, called $RSV_j^{i,1,2} \in S^1$, is obtained by means of the linguistic matching function $g^2 : D \times S^1 \times S^2 \rightarrow S^1$ as follows

$$RSV_j^{i,1,2} = g^2(RSV_j^{i,1}, c_i^2, d_j) = \begin{cases} s_0^1 & \text{if } d_j \notin \mathcal{B}^S \\ RSV_j^{i,1} & \text{if } d_j \in \mathcal{B}^S \end{cases} \quad (26.3)$$

where $\mathcal{B}^S$ is the set of documents such that $\mathcal{B}^S \subseteq Supp(\mathcal{M})$ where,

$$\mathcal{M} = \{(d_1, RSV_1^{i,1}), \ldots, (d_m, RSV_m^{i,1})\} \quad (26.4)$$

is a fuzzy subset of documents obtained according to the following algorithm:

```
1. $K = \#Supp(\mathcal{M})$
2. REPEAT
   $M^K = \{s_q \in S : \mu_{s_q}(K/m) = Sup_v\{\mu_{s_v}(K/m)\}\}$.
   $s^K = Sup_q\{s_q \in M^K\}$.
   $K = K - 1$.
3. UNTIL $((c_i^2 \in M^{K+1})$ OR $(c_i^2 \geq s^{K+1}))$.
4. $\mathcal{B}^S = \{d_{\sigma(1)}, \ldots, d_{\sigma(K+1)}\}$, such that $RSV_{\sigma(h)}^{i,1} \leq RSV_{\sigma(l)}^{i,1}, \forall l \leq h$.
```

According to g^2, the application of the quantitative semantics consists of reducing the number of documents to be considered by the evaluation subsystem for t_i in the later steps.

4.- Evaluation of subexpressions and modeling of the relative importance semantics

We argue that the relative importance semantics in a single-term query has no meaning. Then, in this step we have to evaluate the relevance of documents with respect to the subexpressions of queries composed of two atomic components.

Given a subexpression q_v with $\mathcal{I} \geq 2$ atoms, we know that each document d_j presents a partial $RSV_j^{i,1,2} \in S^1$ with respect to each atom $< t_i, c_i^1, c_i^2, c_i^3 >$ of q_v. Then, the evaluation of the relevance of a document d_j with respect to the whole subexpression q_v implies the aggregation of the partial relevance degrees $\{RSV_j^{i,1,2},\ i = 1, \ldots, \mathcal{I}\}$ weighted by means of the respective relative importance degrees $\{c_i^3 \in S^3,\ i = 1, \ldots, \mathcal{I}\}$. Therefore, as $S^1 \neq S^3$, we have to develop an aggregation procedure of multi-granular linguistic information. As said, to do so, we first choose a label set BLTS to make linguistic information uniform. In this case, BLTS=S' which is used to assess RSVs (relevance degrees of documents). Then, each linguistic information value is transformed into S' by means of the following transformation function:

Definition: *Let $A = \{l_0, \ldots, l_p\}$ and $S' = \{s'_0, \ldots, s'_m\}$ be two label sets, such that $m \geq p$. Then, a multi-granularity transformation function, $\tau_{AS'}$ is defined as $\tau_{AS'} : A \longrightarrow \mathcal{F}(S')$*

$$\tau_{AS_T}(l_i) = \{(s'_k, \alpha_k^i)\ /k \in \{0, \ldots, m\}\},$$
$$\forall l_i \in A,\ \alpha_k^i = \max_y \min\{\mu_{l_i}(y), \mu_{s'_k}(y)\}, \tag{26.5}$$

where $\mathcal{F}(S')$ is the set of fuzzy sets defined in S', and $\mu_{l_i}(y)$ and $\mu_{s'_k}(y)$ are the membership functions of the fuzzy sets associated to the terms l_i and s'_k, respectively [28].

Therefore, the result of $\tau_{AS'}$ for any linguistic value of A is a fuzzy set defined in the BLTS, S'. Using the multi-granularity transformation functions $\tau_{S^1S'}$ and $\tau_{S^3S'}$, we transform the linguistic values $\{RSV_j^{i,1,2} \in S^1,\ i = 1, \ldots, \mathcal{I}\}$ and $\{c_i^3 \in S^3,\ i = 1, \ldots, \mathcal{I}\}$ into S', respectively. Therefore, the values $RSV_j^{i,1,2}$ and c_i^3 are represented as fuzzy sets defined on S' characterized by the following expressions:

1. $\tau_{S^1 S'}(RSV_j^{i,1,2}) = [(s'_0, \alpha_0^{ij}), \ldots, (s'_m, \alpha_m^{ij})]$, and
2. $\tau_{S^2 S'}(c_i^3) = [(s'_0, \alpha_0^{i}), \ldots, (s'_m, \alpha_m^{i})]$, respectively.

In each subexpression q_v, we find that the atoms can be combined using the AND or OR Boolean connectives, depending on the normal form of the user query. The restrictions imposed by the relative importance weights must be applied in the aggregation operators used to model both connectives. These aggregation operators should guarantee that the more important the query terms, the more influential they are in the determination of the RSVs. To do so, these aggregation operators must carry out two activities [27]: i) the transformation of the weighted information under the importance degrees by means of a transformation function h; and ii) the aggregation of the transformed weighted information by means of an aggregation operator of non-weighted information f. As it is known, the choice of h depends upon f. In [43], Yager discussed the effect of the importance degrees on the MAX (used to model the connective OR) and MIN (used to model the connective AND) types of aggregation and suggested a class of functions for importance transformation in both types of aggregation. For the MIN aggregation, he suggested a family of t-conorms acting on the weighted information and the negation of the importance degree, which presents the non-increasing monotonic property in these importance degrees. For the MAX aggregation, he suggested a family of t-norms acting on weighted information and the importance degree, which presents the non-decreasing monotonic property in these importance degrees.

Following the ideas shown above, we use the OWA operators ϕ^1 (with orness(W)≤ 0.5) and ϕ^2 (with orness(W)> 0.5) to model the AND and OR connectives, respectively. Hence, when $h = \phi^1$, $f = \max(Neg(weight), value)$, and when $h = \phi^2$, $f = min(weight, value)$.

Then, given a document d_j, we evaluate its relevance with respect to a subexpression q_v, called RSV_j^v, as $RSV_j^v = [(s'_0, \alpha_0^v), \ldots, (s'_m, \alpha_m^v)]$, where

1. if q_v is a conjunctive subexpression then

$$\alpha_k^v = \phi^1(max((1-\alpha_k^1), \alpha_k^{1j}), \ldots, max((1-\alpha_k^{\mathcal{I}}), \alpha_k^{\mathcal{I}j})) \tag{26.6}$$

2. if q_v is a disjunctive subexpression then

$$\alpha_k^v = \phi^2(min(\alpha_k^1, \alpha_k^{1j}), \ldots, min(\alpha_k^{\mathcal{I}}, \alpha_k^{\mathcal{I}j})). \tag{26.7}$$

5.- Evaluation of the whole query. In this step, the final evaluation of each document is achieved by combining their evaluations with respect to all the subexpressions using, again, the OWA operators ϕ^1 and ϕ^2 to model the AND and OR connectives, respectively.

Then, given a document d_j, we evaluate its relevance with respect to a query q as $RSV_j = \{(s'_0, \beta_0^j), \ldots, (s'_m, \beta_m^j)\}$, where $\beta_k^j = \phi^1(\alpha_k^1, \ldots, \alpha_k^{\mathcal{V}})$, if q is in CNF, and $\beta_k^j = \phi^2(\alpha_k^1, \ldots, \alpha_k^{\mathcal{V}})$, if q is in DNF, with $\mathcal{V}$ standing for the number of subexpressions in q.

Remark: *On the NOT Operator.* We should note that, if a query is in CNF or DNF form, we have to define the negation operator only at the level of single atoms. This simplifies the definition of the NOT operator. As was done in [30], the evaluation of document d_j for a negated weighted atom $< \neg(t_i), c_i^1, c_i^2, c_i^3 >$ is obtained from the negation of the index term weight $F(t_i, d_j)$. This means to calculate g^1 from the linguistic value $Label(1 - F(t_i, d_j))$.

6.- Presenting the output of the IRS

At the end of the evaluation of a user query q, each document d_j is characterized by RSV_j which is a fuzzy set defined on S'. Of course, an answer of an IRS where the relevance of each document is expressed by means of a fuzzy set is not easy to understand, and neither to manage. To overcome this problem, we present the output of our IRS by means of ordered linguistic relevance classes, as in [29, 30]. Furthermore, in each relevance class we establish a ranking of the documents using a confidence degree associated to each document.

To do so, we calculate a label $s^j \in S'$ for each document d_j, which represents its linguistic relevance class. We design an easy linguistic approximation process in S' using a similarity measure, e.g., the Euclidean distance. Each label $s'_k \in S'$ is represented as a fuzzy set defined in S', i.e., $\{(s'_0, 0), \ldots, (s'_k, 1), \ldots, (s'_m, 0)\}$. Then, we calculate s^j as

$$s^j = MAX\{s'_l | Conf(s'_l, RSV_j) = min_k\{Conf(s'_k, RSV_j)\}\}, \qquad (26.8)$$

where $Conf(s'_k, RSV_j) \in [0,1]$ is the confidence degree associated to d_j defined as

$$Conf(s'_k, RSV_j) = \sqrt{\sum_{i=0}^{k-1}(\beta_i^j)^2 + (\beta_k^j - 1)^2 + \sum_{i=k+1}^{m}(\beta_i^j)^2}. \qquad (26.9)$$

26.4 A Multiobjective Genetic Algorithm to Automatically Learn Linguistic Persistent Queries

In this section we present a multiobjective Genetic Algorithm (GA) for linguistic PQ learning. The queries to be derived are legitimate queries of the multigranular linguistic information-based IRS defined in Section 26.3, thus allowing us to design expressive user profiles. The next subsections are devoted to the description of the algorithm.

26.4.1 Coding Scheme

It can be seen how the linguistic query structure considered could be represented as an expression tree, whose terminal nodes are query terms and whose inner nodes are the Boolean operators *AND*, *OR* or *NOT*. Besides, each terminal node has associated three ordinal linguistic values corresponding to the

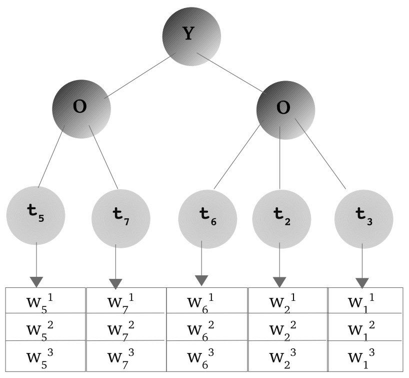

Y
O
O
t_5
t_7
t_6
t_2
t_3
w_5^1 w_7^1 w_6^1 w_2^1 w_1^1
w_5^2 w_7^2 w_6^2 w_2^2 w_1^2
w_5^3 w_7^3 w_6^3 w_2^3 w_1^3

Fig. 26.1. Weighted Boolean query with linguistic weights

three semantics shown in the previous section. Figure 26.1 shows a graphical example of this kind of queries.

Hence, the natural representation would be to encode the query within a tree and to work with a Genetic Programming algorithm [32] to evolve it, as done by previous approaches devoted to the derivation of Boolean queries [16, 17, 40] or extended Boolean queries (fuzzy queries with numerical weights) [18, 19, 20, 21, 35].

However, the special characteristics of our linguistic queries allow us to deal with a much simpler representation. As seen in the previous section, the queries of our Multi-Granular Linguistic IRS are always on DNF or CNF. Hence, the query structure is not completely free but it is restricted to a disjunction of several conjunctions, or to a conjunction of several disjunctions, i.e., a fixed three-level tree structure with an OR (respectively, an AND) node in the root, several AND (respectively, OR) nodes in the second level, and the different (positive or negative) terms involved in each subexpression in the third level.

Thanks to this, we are able to design a coding scheme that represents linguistic expression trees as integers vectors, which can be represented using a usual GA chromosome. We should notice that a method of a similar coding scheme for a simpler Boolean query tree structure that does not consider term weights can be found in [25].

In our case, a chromosome C encoding a candidate linguistic PQ will be composed of two different parts, C_1 and C_2, which respectively encode the query composition (the AND-ed or OR-ed subexpressions), and the term weights. This structure presents the next features:

1. As said, every query is in CNF or DNF, so that the chromosome only encodes the subexpressions of the query (the operators are the same and there is no need to keep them in the query representation).
2. The query tree is encoded as an integer vector, where the number 0 acts a separator between subexpressions while the rest of numbers represent the

different index terms in the documentary database. Negative numbers are associated to negated terms in the query (for example, to represent the atomic expression $NOT\ t_{17}$, the number -17 is used).

3. The weights are represented as another integer vector, where each weight is encoded as its position in its label set. This way, the labels are numbered from 1 to the granularity of the label set, and the number corresponding to the selected label for the term weight is stored in the current vector.

To illustrate the coding scheme considered, the query of Figure 26.1 is encoded in a chromosome with the structure shown in Figure 26.2.

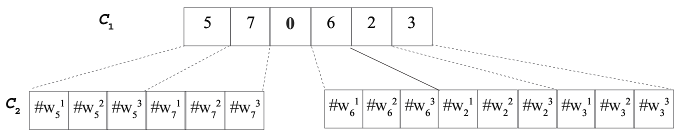


Fig. 26.2. Chromosome structure

26.4.2 Initial Gene Pool

All the individuals in the first population are randomly created, generating separately both parts of chromosomes:

1. Queries (C_1) will be composed of terms selected among those included in the set of relevant documents provided by the user, having those present in more documents a higher probability of being selected.
2. Weights (C_2) are randomly calculated, varying each gene in its respective definition interval: $\{1, ..., label\_set\_granularity\}$.

26.4.3 Fitness Function

The classical precision and recall criteria [2] —computed as shown in equation 26.10— are jointly maximized.

$$P = \frac{\sum_d r_d \cdot f_d}{\sum_d f_d} \qquad ; \qquad R = \frac{\sum_d r_d \cdot f_d}{\sum_d r_d} \tag{26.10}$$

where $r_d \in \{0, 1\}$ is the relevance of document d for the user, and $f_d \in \{0, 1\}$ is the retrieval of document d in the processing of the current query. Notice that both measures are defined in [0,1], where 1 is the best value that can be reached.

26.4.4 Pareto-based Multiobjective Selection Scheme

As said, the objective of our method is to automatically generate several linguistic PQs with a different trade-off between precision and recall in a single run. For this purpose, the SPEA scheme [47] has been employed as the multiobjective GA.

This algorithm introduces the elitism concept, explicitly maintaining an external population P_e. This population stores a fixed number of nondominated solutions which have been found since the start of the run. In each generation, the new nondominated solutions found are compared with the solutions in the existing external population, storing the resulting nondominated solutions on the latter. Furthermore, SPEA uses these elitist solutions, together with those in the current population, in the genetic operations, in the hope to lead the population to good areas in the search space.

The selection scheme involves the following steps:

1. The intermediate population is created from both the current population P and the external population (P_e) by means of binary tournament selection.
2. Genetic operators are used over the new individuals to get a new population (P).
3. Nondominated solutions existing in the new population are copied to the elitist population P_e.
4. The dominated and duplicated solutions are removed.

Therefore, the new elitist population is composed of the best nondominated solutions found so far, including new and old elitist solutions. To limit the growth of the elitist population, the size is restricted to a maximum number of solutions using clustering techniques (see [47] for details).

26.4.5 Genetic Operators

Due to the special nature of the chromosomes involved in this generation process (comprised by two different information levels), the design of genetic operators that is able to deal with it becomes a main task. As there exists a strong relationship between the two chromosome parts, operators working cooperatively in C_1 and C_2 are required in order to make best use of the representation considered. It can be clearly observed that the existing relationship will present several problems if not handled adequately. For example, modifications in the first chromosome part have to be automatically reflected in the second one. It makes no sense to modify the query structure, adding, deleting or changing terms and subexpressions, but continue working with the same weights. On the other hand, there is a need to develop the recombination in a correct way in order to obtain meaningful offsprings.

Taking into account these aspects, the following operators are going to be considered:

Crossover

Two different crossover operators are employed depending of the two parents' scope:

- *Crossover when both parents encode the same query (same C_1 part):* If this is the case, then the genetic search has located a promising space zone that has to be adequately exploited. This task is developed by applying a two-point crossover in C_2 (term weights), and obviously by maintaining the parent C_1 values in the offsprings.
- *Crossover when the parents encode different queries (different C_1 part):* This second case highly recommends the use of the information encoded by the parents for exploring the search space in order to discover new promising zones. In this way, an standard crossover operator is applied over both parts of the chromosomes. This operator performs as follows: a crossover point *cp* is randomly generated in C_1 for each parent and then, the genes between point *cp* and the end of C_1 are interchanged. In C_2, the crossover is developed in the same way, using the corresponding crossover points. The feasible points of crossover are the separators between subexpressions. Figure 26.3 shows an example of the crossover operator.

Mutation

Seven different operators are used, six of them acting on C_1 and one on C_2.

- *Mutation on C_1:* The mutation operators in C_1 are as follows:

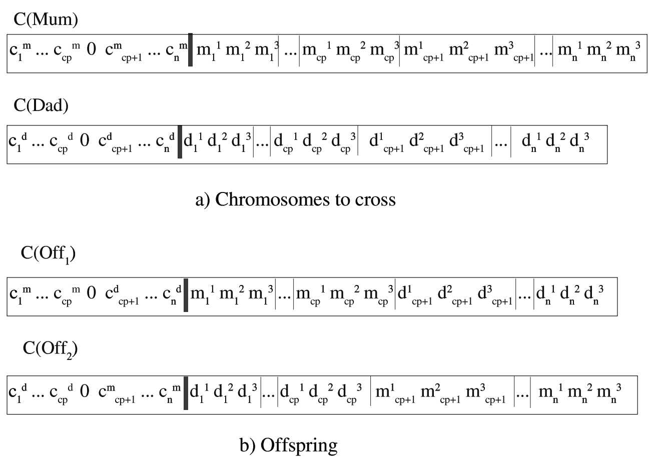


Fig. 26.3. Explorative crossover

1. Replace selected term by a random one.
2. Negation of a term.
3. Deletion of a randomly selected separator.
4. Addition of a new separator in a valid random position.
5. Displacement of a separator.
6. Replace a subexpression by a randomly generated one with more or less terms (C_2 is automatically updated).

- *Mutation on C_2:* The mutation operator selected for C_2 is similar to the one proposed by Thrift in [41] for fuzzy rule base learning with GAs. When a mutation on a gene belonging to the second part of the chromosome is going to be performed, a local modification is developed by changing the current label to the immediately preceding or subsequent one (the decision is made at random). When the label to be changed is the first or last one in the label set, the only possible change is developed.

26.5 Experiments and Analysis of Results

26.5.1 Experiments Developed

This section is devoted to test the performance of the proposed MOGA-LPQ IQBE algorithm that automatically derives profiles represented as linguistic PQs. Since the most common profile structure is the *"bag of words"*, which is based on a set of weighted keywords representing the user's interest, we must compare our algorithm with classical methods of profile construction, in order to verify the performance. As comparison method, we have chosen one of the state-of-the-art algorithms (RSV-OKAPI) [23, 24], based on the vector space model and the probability theory [2]. We consider Robertson Selection Value (RSV) as the approach for profile learning and OKAPI BM25 as similarity function to match profiles and documents.

However, we should notice that this comparison is not fair to our algorithm due to two reasons. On the one hand, we are designing more expressive user profiles, what usually comes with a retrieval efficacy decrease (the usual interpretability-accuracy trade-off problem). On the other hand, our algorithm is able to derive several linguistic PQs (user profiles) with a different trade-off between precision and recall in a single run, thus giving more chances to the user to retrieve much/less relevant information with a larger/lesser retrieval noise at his own choice. So, the selection of a single PQ to compare it against the user profile derived by RSV-OKAPI will restrict the capabilities of our method.

The documentary database considered to design our experimental setup has been the popular *Cranfield* collection, composed of 1398 documents about Aeronautics [2]. 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 a usual TFIDF indexing 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. The relevance judgments associated to each of these selected queries have been considered to play the role of seven different user's information needs.

For each one of these queries, the documentary base has been randomly divided into two different, non overlapped, document sets, training and test, each of them composed of a fifty percent of the (previously known) relevant and irrelevant documents for the query.

MOGA-LPQ has been run five times with different initializations for each selected query during 50000 fitness function evaluations in a 2.4GHz Pentium IV computer with 1Gb of RAM. The parameter values considered are a population size of 100 individuals, an elitist population size of 25, a maximum of 10 terms by query, and 0.8 and 0.2 for the crossover and mutation probabilities (in both the C_1 and the C_2 parts). The retrieval threshold has been set to the third label of the label set.

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

On the other hand, RSV-OKAPI has been run only one time, since it has no random components, with a profile size of 10 terms and with QTW (the term weights) equal to the RSV value.

Every selected linguistic PQ has been run on the corresponding test set once preprocessed[3] in order to evaluate their capability to retrieve relevant information for the user. The same has been done for the profile derived by RSV-OKAPI.

26.5.2 Analysis of the Pareto Sets Derived

Several quantitative metrics have been proposed in the literature to measure the quality of Pareto sets derived by multiobjective algorithms [46]. Specifi-

[3] 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. Notice that, this is another source of retrieval efficacy loss for our method as two-term subexpressions where one of the terms is not present in the test document set are completely removed due to the restriction imposed on the linguistic query structure of not having single-term subexpressions (see Section 26.3). In the future, we aim at solving this problem, what would significantly improve the performance of our algorithm on the test set.

cally, we have used three different metrics; $\mathcal{M}_2^*$ and $\mathcal{M}_3^*$ and the number of nondominated solutions in the Pareto set.

Table 26.1 collects several data about the composition of the five Pareto sets generated for each query, where both the averaged value and its standard deviation are shown. From left to right, the columns contain the query number ($\#q$), the number of different non-dominated solutions obtained ($\#d$), corresponding to the number of different objective vectors (i.e., precision-recall pairs) existing among them, and the values of the two multiobjective EA metrics selected, $\mathcal{M}_2^*$ and $\mathcal{M}_3^*$, each of which is followed by their respective standard deviation values. Regarding the two later metrics, $\mathcal{M}_2^* \in [0, \#d]$ measures the diversity of the solutions found, while $\mathcal{M}_3^*$ measures the range to which the Pareto front spreads out in the objective values (in our case, the maximum possible value is $\sqrt{2} = 1.4142$). In both cases, the higher the value, the better the quality of the obtained Pareto set.

Table 26.1. Statistics of the Pareto sets obtained by the MOGA-LPQ algorithm

$\#q$	$\#d$	$\sigma_{\#d}$	M_2^*	$\sigma_{M_2^*}$	M_3^*	$\sigma_{M_3^*}$
1	7.600	0.669	3.573	0.311	1.209	0.021
2	6.200	0.522	2.836	0.251	1.175	0.031
23	10.400	0.607	4.701	0.277	**1.276**	0.008
73	5.800	0.522	2.900	0.261	1.137	0.043
157	**12.200**	0.716	**5.581**	0.307	1.229	0.020
220	5.000	0.283	2.500	0.141	1.127	0.034
225	7.000	0.566	3.163	0.244	1.201	0.016

In view of the values shown in Table 26.1, the Pareto fronts obtained are of good quality. We can see how all runs generate a number of linguistic PQs with different precision-recall trade-offs proportional to the number of relevant documents associated with the original query (for those cases where a larger number of relevant documents are provided, a larger number of different PQs are obtained in the Pareto sets); and that standard deviation values are around 0.6. The values of the $\mathcal{M}_2^*$ and $\mathcal{M}_3^*$ metrics are appropriate as well, showing a very good distribution of the Pareto fronts. We should emphasize the values of the latter, very close to 1.4142, the maximum possible value. This shows that the generated Pareto fronts cover a wide area in the space. To illustrate this fact, Figure 26.4 depicts the unified Pareto front obtained for query 157 as an example.

26.5.3 MOGA-LPQ *versus* RSV-OKAPI

To compare the considered algorithms, the average precision over eleven recall levels (P_{avg}) [2] has been taken as comparison measure as both can return the documents ordered according to their relevance. When this measure is used,

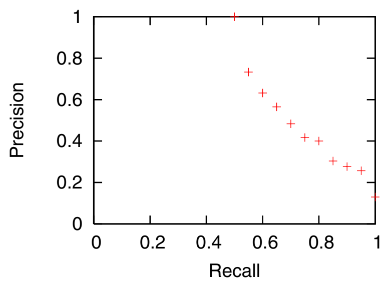


Fig. 26.4. Unified Pareto front generated for query 157

the retrieved document set is descendently ordered according to the RSV values so that the most relevant documents are at the top of the document set.

As said in Section 26.5.1, the five Pareto sets obtained by our method in the five runs performed for each of the seven queries are first unified, and then five different linguistic PQs well distributed on the resulting Pareto front are selected for each query. However, RSV-OKAPI only derives a solution per query as it is run a single time (it has no random components) and generates only one solution per run. In order to ease the comparison, we have considered the following two different performance values for each query for our algorithm:

1. Averaged results: For each Pareto set, we have chosen the PQ with the greater value of P_{avg} on the training document set and the five P_{avg} values obtained have been averaged.
2. Best result on the test set: Of the five selected PQs, we have chosen that with the greater value of P_{avg} on the test document set.

Figure 26.5 illustrates the process to obtain the best results on the test set.

Tables 26.2 and 26.3 show the retrieval efficacy for each query for MOGA-LPQ and RSV-OKAPI, respectively. In those tables, $\#rel$ stands for the number of relevant documents associated with that query, and $\#top$ for the number of relevant documents located in the $\#rel$ first positions of the document set (ordered by their RSV value).

The left side of Table 26.2 shows the average results. In view of them, we should notice the good results obtained by our method on the training document set: the average precision values are around 0.5, whereas the proportion of relevant documents in the first positions is around half the number of relevant documents for the query. However, on the test set, results are rather low compared to training results (we can talk about overlearning, that can be due to the preprocessing made to adapt the derived PQ to the test document collection, as mentioned before), with the P_{avg} values being around 0.1. This way, the access to new documents relevant for the user is not easy, since the user would need to examine a lot of documents before finding a useful one.

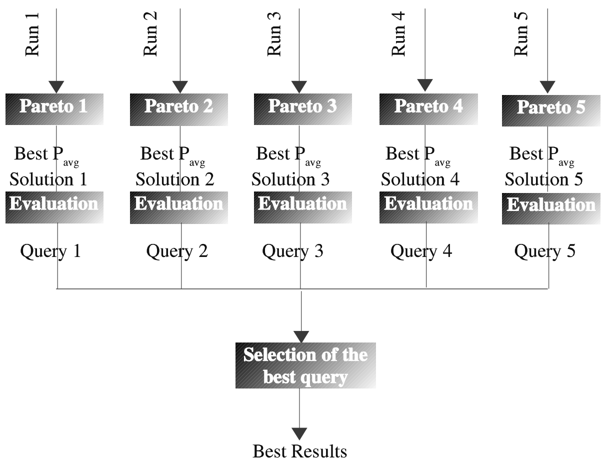


Fig. 26.5. Process to obtain the best results on the test set

Table 26.2. Retrieval Efficacy of MOGA-LPQ Linguistic PQs

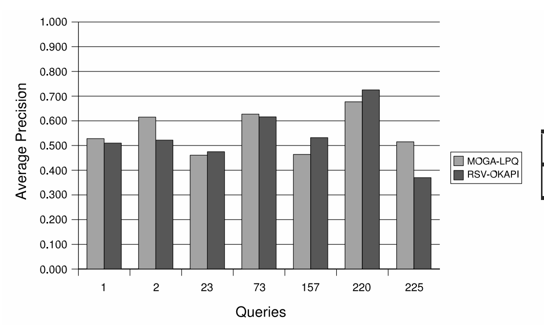
Average Results							**Best Results**						
	Training set			Test set				Training set			Test set		
$\#q$	P_{avg}	$\#rel$	$\#top$	P_{avg}	$\#rel$	$\#top$	$\#q$	P_{avg}	$\#rel$	$\#top$	P_{avg}	$\#rel$	$\#top$
1	0.528	14	7.8	0.070	15	0.75	**1**	0.467	14	8	0.201	15	2
2	0.615	12	7.2	0.143	13	2.75	**2**	0.725	12	7	0.289	13	4
23	0.461	16	8.0	0.113	17	2.4	**23**	0.426	16	8	0.213	17	5
73	0.627	10	5.6	0.057	11	0.75	**73**	0.748	10	7	0.080	11	1
157	0.464	20	9.4	0.112	20	2.2	**157**	0.454	20	11	0.422	20	7
220	0.677	10	6.2	0.173	10	1.2	**220**	0.647	10	5	0.349	10	3
225	0.515	12	6.2	0.078	13	0.75	**225**	0.484	12	5	0.126	13	1

Table 26.3. Retrieval Efficacy of RSV-OKAPI Profiles

	Training set			Test set		
$\#q$	P_{avg}	$\#rel$	$\#top$	P_{avg}	$\#rel$	$\#top$
1	0.510	14	6	0.484	15	6
2	0.522	12	5	0.789	13	9
23	0.475	16	7	0.357	17	7
73	0.616	10	5	0.268	11	2
157	0.532	20	10	0.298	20	7
220	0.725	10	6	0.544	10	5
225	0.370	12	5	0.099	13	2

If we concentrate on the best results (the right half of Table 26.2), the training results are usually not better than those in the left side (but in queries 2 and 73). Since the selected query is the best on the test set, this does not mean that it must be the case on the training set as well. Of course, the best results fully outperforms the average results on the test set, generally around the double, making easy the user access to new information.

On the other hand, the state-of-the-art method for user profile generation shows an appropriate behavior, as can be seen in Table 26.3. P_{avg} values on the training set are over 0.4, whereas the test results are around 0.4.

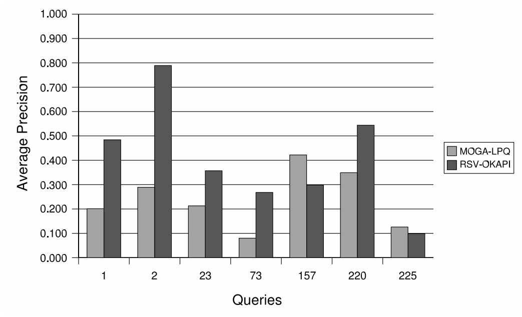


	Average P_{avg}
MOGA-LPQ	0.56
RSV-OKAPI	0.54

Fig. 26.6. Comparison between the mean P_{avg} values of MOGA-LPQ and RSV-OKAPI on the training document set

Figure 26.6 shows a comparative bar chart of the P_{avg} values got by each algorithm on the training document set (the MOGA-LPQ values correspond to the average results). Graphically, we can observe that our method outperforms RSV-OKAPI in four of the seven queries, apart from generating more expressive profiles as we shall see later. In fact, the global mean of the MOGA-LPQ P_{avg} values for the seven queries is slightly higher than that of RSV-OKAPI (0.56 in front of 0.54). Furthermore, the number of relevant documents in the first positions of the retrieved document set is very similar, as can be seen in Tables 26.2 and 26.3. However, notice again that, in each case, we are comparing the averaged results of five linguistic PQs derived by our MOGA-LPQ with that of the single user profile derived by RSV-OKAPI. Hence, if we would have chosen only the best of the five PQs in the training set performance for the comparison, it would have been much more positive for us.

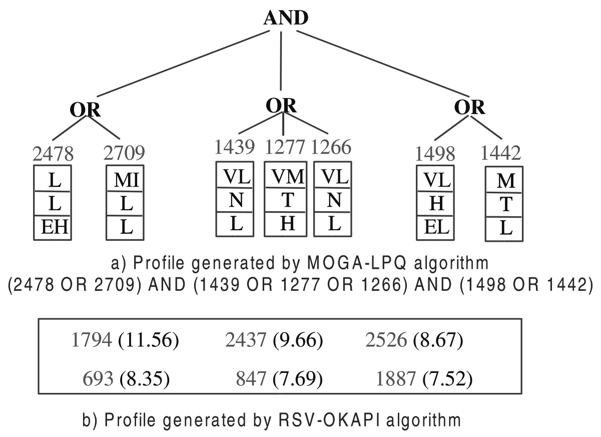
Similarly, Figure 26.7 shows a comparative bar chart of the P_{avg} values got by each algorithm on the test document set (the MOGA-LPQ reported values correspond to the linguistic PQ with the best test results). In view of it, we should notice that RSV-OKAPI outperforms our IQBE method in five of the seven queries, and the differences is rather big, especially for query 2 (around 0.5). Numerically, the global mean of P_{avg} for both algorithms is 0.24



	Average P_{avg}
MOGA-LPQ	0.24
RSV-OKAPI	0.41

Fig. 26.7. Comparison between MOGA-LPQ (Best P_{avg}) and RSV-OKAPI on test document set

and 0.41 for MOGA-LPQ and RSV-OKAPI, respectively. With regards to the position of the relevant documents at the top of the document set, there is no meaningful difference between them. Therefore, the reason of the differences in the P_{avg} values is the position of the rest of the relevant documents.



AND
OR
OR
OR
2478
2709
1439
1277
1266
1498
1442
L L EH
MI L L
VL N L
VM T H
VL N L
VL H EL
M T L
a) Profile generated by MOGA-LPQ algorithm
(2478 OR 2709) AND (1439 OR 1277 OR 1266) AND (1498 OR 1442)
1794 (11.56)
2437 (9.66)
2526 (8.67)
693 (8.35)
847 (7.69)
1887 (7.52)
b) Profile generated by RSV-OKAPI algorithm

Fig. 26.8. User profiles generated by MOGA-LPQ and RSV-OKAPI for query 2

Nevertheless, although our algorithm achieves worse results than RSV-OKAPI on the test set, the modeling of user profiles as PQs clearly improves the expressivity of the user profiles when learning both the terms which is composed of the profile and its structure, instead of learning only the set of terms. As an example, Figure 26.8 shows the profiles derived by both algorithms for query 2.

26.6 Concluding Remarks

The use of soft computing tools to design PQs for text retrieval has been analyzed by constructing linguistic queries from sets of training documents extracted from the Cranfield collection by means of a multiobjective GA. As shown, our method is competitive with the classical method on the training set, behaving both algorithms in a similar way, and with ours having the advantage of a higher profile comprehensibility; whereas the classic state-of-the-art method gets better results on the test set. Nevertheless, our method is promising since it learns both the terms and the profile structure, instead of learning only the set of terms, as well as it derives a set of PQs with different precision-recall trade-off in a single run, although we must refine it with the aim of improving its capability to find new relevant information for the user.

In our opinion, many different future works arise from the present contribution. Firstly, we will search for new functions to measure the similarity between expression trees with the purpose of being able to work in the decision space[4]. On the other hand, we will try to improve our test set performance, either by designing a new preprocessing algorithm with a less aggressive way to adapt the linguistic PQs for the test document collection, or by modifying the validation process by using a division of the data set in training, validation and test document sets, as done in the last contributions proposed in the field [23, 24], instead of using the classic division in training and test sets.

References

[1] T. Bäck, D.B. Fogel, and Z. Michalewicz, editors. *Handbook of Evolutionary Computation*. IOP Publishing and Oxford University Press, 1997. 602

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Adisson, 1999. 603, 604, 614, 617, 619

[3] N.J. Belkin and W.B. Croft. Information Filtering and Information Retrieval: Two Sides of the same Coin? *Communications of the ACM*, 35(12):29–38, 1992. 603

[4] P.P. Bonissone and K.S. Decker. Selecting Uncertainty Calculi and Granularity: An Experiment in Trading-off Precision and Complexity. In L.H. Kanal and J.F. Lemer, editors, *Uncertainty in Artificial Intelligence*, pages 217–247. North-Holland, 1986. 605

[5] A. Bookstein. Fuzzy Request: An Approach to Weighted Boolean Searches. *Journal of the American Society for Information Science*, 31:240–247, 1980. 604

[6] 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. Springer-Verlag, 1995. 602

[4] We have worked in the objective space, but the number of solutions presenting different precision-recall values (different objective value arrays) is a little bit low with respect to the size of the elitist population. To solve this, we decided to work in the decision space, utilizing the *edit* or *Levenshtein distance* [36] to measure the similarity between expression trees, but although this measure increased the number of solutions, the run time also increased significantly.

[7] G. Bordogna and G. Pasi. A Fuzzy Linguistic Approach Generalizing Boolean Information Retrieval: A Model and its Evaluation. *Journal of the American Society for Information Science*, 44:70–82, 1993. 603, 604

[8] G. Bordogna and G. Pasi. Linguistic Aggregation Operators of Selection Criteria in Fuzzy Information Retrieval. *International Journal of Intelligent Systems*, 10:233–248, 1995. 603, 604

[9] G. Bordogna and G. Pasi. An Ordinal Information Retrieval Model. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(1):63–75, 2001. 603

[10] D. Buell and D.H. Kraft. A Model for a Weighted Retrieval System. *Journal of the American Society for Information Science*, 32:211–216, 1981. 604

[11] D. Buell and D.H. Kraft. Threshold Values and Boolean Retrieval Systems. *Information Processing & Management*, 17:127–136, 1981. 604

[12] 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 Algoritms, and Simulated Annealing. *Journal of the American Society for Information Science*, 49(8):693–705, 1998. 603, 604

[13] C. A. Coello, D. A. Van Veldhuizen, and G. B. Lamant. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Kluwer Academic Publishers, 2002. 602

[14] O. Cordón and E. Herrera-Viedma. Editorial: Special Issue on Soft Computing Applications to Intelligent Information Retrieval on the Internet. *International Journal of Approximate Reasoning*, 34(2-3):89–95, 2003. 602

[15] O. Cordón, E. Herrera-Viedma, C. López-Pujalte, M. Luque, and C. Zarco. A Review of the Application of Evolutionary Computation to Information Retrieval. *International Journal of Approximate Reasoning*, 34:241–264, 2003. 602

[16] O. Cordón, E. Herrera-Viedma, and M. Luque. Evolutionary Learning of Boolean Queries by Multiobjective Genetic Programming. In *Lecture Notes in Computer Science 2439. Proc. of the PPSN-VII*, pages 710–719, Granada (Spain), 2002. 613

[17] O. Cordón, E. Herrera-Viedma, and M. Luque. Improving the Learning of Boolean Queries by means of a Multiobjective IQBE Evolutionary Algorithm. *Information Processing and Management*, 2005. To appear. 613

[18] O. Cordón, E. Herrera-Viedma, M. Luque, F. Moya, and C. Zarco. Analyzing the Performance of a Multiobjective GA-P Algorithm for Learning Fuzzy Queries in a Machine Learning Enviroment. In *Lecture Notes in Artificial Intelligence 2715. Proc. of the 10th IFSA World Congress*, pages 611–615, Istambul (Turkey), 2003. 613

[19] 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. 613

[20] 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. 613

[21] 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*, pages 47–40. Springer, 2004. 613

[22] F. Crestani and G. Pasi, editors. *Soft Computing in Information Retrieval*. Physica-Verlag, 2000. 602

[23] W. Fan, M.D. Gordon, and P. Pathak. An Integrated Two-Stages Model for Intelligent Information Routing. *Decision Support Systems*, 2004. Submitted. 601, 602, 603, 617, 624

[24] W. Fan, M.D. Gordon, and P. Pathak. Effective Profiling of Consumer Information Retrieval Needs: A Unified Framework and Empirical Comparision. *Decision Support Systems*, 2005. To appear. 602, 603, 617, 624

[25] J.L. Fernández-Villacañas and M. Shackleton. Investigation of the Importance of the Genotype-Phenotype Mapping in Information Retrieval. *Future Generation Computer Systems*, 19(1):55–68, 2003. 613

[26] U. Hanani, B. Shapira, and P. Shoval. Information Filtering: Overview of Issues, Research and Systems. *User Modeling and User-Adapted Interaction*, 11:203–259, 2001. 601, 603

[27] F. Herrera and E. Herrera-Viedma. Aggregation Operators for Linguistic Weighted Information. *IEEE Transactions on Systems, Man and Cybernetics; Part A: Systems*, 27:646–656, 1997. 604, 605, 611

[28] F. Herrera, E. Herrera-Viedma, and L. Martínez. A Fusion Approach for Managing Multi-Granularity Linguistic Term Sets in Decision Making. *Fuzzy Sets and Systems*, 114:43–58, 2000. 605, 608, 610

[29] E. Herrera-Viedma. An Information Retrieval System with Ordinal Linguistic Weighted Queries based on Two Weighting Elements. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(1):77–88, 2001. 603, 605, 606, 608, 609, 612

[30] E. Herrera-Viedma. Modeling the Retrieval Process for an Information Retrieval System using an Ordinal Fuzzy Linguistic Approach. *Journal of the American Society for Information Science and Technology*, 52(6):460–475, 2001. 603, 604, 605, 606, 607, 608, 61

[31] E. Herrera-Viedma, O. Cordón, M. Luque, A. G. López, and A. M. Muñoz. A Model of Fuzzy Linguistic IRS Based on Multi-Granular Linguistic Information. *International Journal of Approximate Reasoning*, 34:221–239, 2003. 603, 605

[32] J. Koza. *Genetic Programming. On the Programming of Computers by Means of Natural Selection*. The MIT Press, 1992. 613

[33] D.H. Kraft, G. Bordogna, and G. Pasi. An Extended Fuzzy Linguistic Approach to Generalize Boolean Information Retrieval. *Information Sciences*, 2:119–134, 1994. 603, 604

[34] D.H. Kraft and D.A. Buell. Fuzzy Sets and Generalized Boolean Retrieval Systems. *International Journal of Man-Machine Studies*, 19:45–56, 1983. 604

[35] D.H. Kraft, F.E. Petry, B.P. Buckles, 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. World Scientific, 1997. 613

[36] V. I. Levenshtein. Binary Codes of Correcting Deletions, Insertions and Reversal. *Sov. Phys. Dokl.*, 6:705–710, 1996. 624

[37] M. Nikravesh, V. Loia, and B. Azvine. Fuzzy Logic and the Internet (FLINT): Internet, World Wide Web and Search Engines. *Soft Computing*, 6(4):287–299, 2002. 603

[38] D.W. Oard and G. Marchionini. A Conceptual Framework for Text Filtering. Technical Report CS-TR-3643, University of Maryland, College Park, 1996. 601, 602, 603

[39] G. Pasi. Intelligent Information Retrieval: Some Research Trends. In J.M. Benítez, O. Cordón, F. Hoffmann, and R. Roy, editors, *Advances in Soft Computing. Engineering Design and Manufacturing*, pages 157–171. Springer, 2003. 602, 603

[40] M.P. Smith and M. Smith. The Use of Genetic Programming to Build Boolean Queries for Text Retrieval through Relevance Feedback. *Journal of Information Science*, 23(6):423–431, 1997. 613

[41] P. Thrift. Fuzzy Logic Synthesis with Genetic Algorithms. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 509–513, 1991. 617

[42] W.G. Waller and D.H. Kraft. A Mathematical Model of a Weighted Boolean Retrieval System. *Information Processing & Management*, 15:235–245, 1979. 604

[43] R.R Yager. A Note on Weighted Queries in Information Retrieval Systems. *Journal of the American Society for Information Science*, 38:23–24, 1987. 611

[44] R.R. Yager. On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decision Making. *IEEE Transactions on Systems, Man, and Cybernetics*, 18:183–190, 1988. 606

[45] L.A. Zadeh. The Concept of a Linguistic Variable and its Applications to Approximate Reasoning. *Part I, II & III, Information Science*, 8:199–249, 8:301–157, 9:43–80, 1975. 603, 604

[46] E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, 2000. 618

[47] E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999. 615