Control and Cybernetics

vol. **38** (2009) No. 2

Ontologies and soft computing in flexible querying*

by

Manolis Wallace

Department of Computer Science and Technology,
University of Peloponnese
End of Karaiskaki Str., 22100 Tripolis, Greece
e-mail: wallace@uop.gr

Abstract: Flexible querying is all about providing users with more intuitive and efficient means of interaction with information or document bases, so that retrieving the items they are interested in becomes easier and more pleasant. In this process three steps may be identified as crucial in determining the overall result: understanding the user's request, understanding the content of what is available, and matching between the two. In this paper we present an integrated soft computing framework to tackle all three steps by incorporating fuzzy relational knowledge stored in modified ontological structures, a modified agglomerative clustering approach to knowledge based information processing, a detailed profile modeling methodology, and a novel approach to handling of large and sparse transitive relations.

Keywords: context, content understanding, ontologies, fuzzy relations, transitivity.

1. Introduction

Information retrieval is a field of study that received augmented attention back in the 1970s, when collections of textual documents first started to appear. At that time pioneering works such as Rocchio (1971) or Salton (1971) applied a closed world concept to model information space and thus relied on statistical analysis approaches to provide for efficient indexing and searching in documents. This approach reasonably soon reached an upper bound of performance, reducing the attention that information retrieval received, with efforts reported in the annual TREC (Text Retrieval) conferences being one of the few exceptions; developments in the broader area of information retrieval are best reviewed and summarized in Baeza-Yates and Ribeiro-Neto (1999).

With information retrieval reaching its limits, flexible querying emerged, as an attempt to provide for different, more intuitive means and approaches to

*Submitted: June 2008; Accepted: August 2008.

querying. Flexible querying goes beyond pre-existing information retrieval approaches in that it acknowledges additional issues that need to be tackled. For example, flexible querying considers the imperfect way in which user queries are formed, whereas in conventional information retrieval a "good" query was taken for granted, see FQAS, the Flexible Query Answering Systems Series. (Flexible querying is sometimes considered to be associated strictly with database querying. In this work the term is used in its more general sense and is associated with general purpose information retrieval.)

Naturally, it was not long before it became apparent that soft computing, with its inherent ability to model uncertainty and real life information, is a suitable tool for the field of flexible querying. For example, Dubois and Prade (1997) already discussed the utilization of fuzzy sets in order to model the user query, thus providing the user with more descriptive power for the expression of queries. A wide range of works since then have also discussed the utilization of fuzzy sets to model the user query (Bordogna and Pasi, 2002), and some have even discussed practical implementation issues (Callens et al., 2003).

Still, it is quite clear that soft computing remains a very useful tool towards extending even further the state of the art in flexible querying. More recent works in the field discuss the utilization of generic fuzzy sets (de Calmes, Prade and Sedes, 2007), possibility theory (Matthé et al., 2005) and fuzzy bags (Rocacher and Bosc, 2005) in the matching process, as well as in the representation of the user query (Buche et al., 2006). All of these works focus solely on the representation of the query (and possibly the index) and the matching between the two, still relying on a more conventional approach to other processes involved in flexible querying, such as understanding the true meaning of the query and/or of the documents; this will be in part the focus of this work.

Another new era was signaled with the emergence of the knowledge based approach to information processing. More than anything else, it was the developments in the field of the semantic web, with its very structured, standardized and detailed representation of human knowledge, that lead to new hopes for enhanced and more intuitive information processing and, why not, understanding, via automated methodologies (Berners-Lee and Fischetti, 1999). In this approach, algorithms are expected to exploit knowledge stored in ontologies in order to process data in the same way as humans do.

Even in this direction the necessity for fuzziness is evident. For example, Bulskov, Knappe and Andreasen (2002) discuss a conceptual querying approach that is based on crisp ontologies and utilize path distances to define concept similarities; the approach is equivalent to a fuzzy ontology approach, in which all degrees are defined to be equal. In such an approach the distance between "animal" and "cat" may originally be one hop, but the enhancement of the ontology with the addition of the term "mammal" would change the distance to two hops, thus also altering the calculated similarity between the two terms. Of course, this is counter-intuitive, indicating that meaningful similarities need to be defined independently from graph based metrics, thus making fuzziness in

ontologies a necessity for flexible ontology based querying. Sanchez and Yamanoi (2006) and Baziz et al. (2007) provide literature reviews of preliminary works towards the incorporation of fuzziness into conceptual querying.

In this work, partially building on previous works, we provide a complete soft computing framework for flexible querying; the proposed framework greatly depends on knowledge stored in fuzzy ontological relations. The main difference of the proposed framework with respect to most other applications of soft computing in flexible querying is that the application of fuzziness is not limited to the representation of the user query and the matching mechanism, as for example in Calegari and Sanchez (2007). The framework applies soft computing to define and detect context, which it then uses in understanding the meaning of the user query, understanding the content of the available documents, relating user preferences to the user query and personalizing the results of the retrieval process.

The structure of the remaining of the paper is as follows: In Section 2 we discuss knowledge representation in ontologies and the inclusion of fuzzy degrees. Section 3 focuses on size and time considerations related to the utilization of large ontological relations in practical applications and Section 4 provides a quantified modeling of context. Based on these, Section 5 comes to present an integrated and uniform approach to the handling of user query understanding, content understanding and personalized query-content matching. Finally, Section 6 discusses the implementation of the presented framework and compares it to other approaches, while Section 7 lists our concluding remarks.

2. Fuzzy relational knowledge and ontologies

With semantics being one of the current "hot" topics of research, ontological information tends to be considered as a "must have" for knowledge based systems. Still, the breadth and practical reach of these systems is often limited due to the crisp and inflexible information, typically conveyed by ontological relations. One issue that has not yet been fully exploited is that of the inclusion of degrees in ontologies, so that human knowledge about the real world can be modeled in a more intuitive manner.

In general, an ontology O may be modeled mathematically as follows:

$$O = \{S, \{R_i\}\}, i = 1 \dots m \tag{1}$$

$$R_i := S \times S \rightarrow \{0, 1\}, i = 1 \dots m \tag{2}$$

where O is the ontology, S the set of semantic concepts it describes and R_i the i-th semantic relation among the concepts out of a total of m supported semantic relations. The formal definition of ontologies of course also supports an inference layer, but herein we omit it for the sake of simplicity as it is not relevant to this work. Although any type of relation may be contained in an ontology, the two main categories found are taxonomic (i.e. ordering) and compatibility (i.e. symmetric) relations; in this work we focus on ordering relations,

although the presented framework can also be applied without modifications to compatibility relations as well as their combinations.

It is well understood that relations among real life concepts are most often a matter of degree, and are therefore best modeled using fuzzy relations. Ontological taxonomies, on the other hand, are crisp in principle. Thus, they fail to fully describe real life information, and are limited to α-cuts of the desired relations. This is a very important drawback that renders such relations insufficient for the services that an intelligent information processing system aims to offer. In this work we move beyond the conventional crisp taxonomies typically contained in ontologies and propose the extension of ontological relations with the inclusion of fuzzy degrees, thus allowing them to implement more realistic representations of real life information. Incorporating this approach in the ontological field would mean that the mathematical modeling of ontologies would have to be extended as follows:

$$O = \{S, \{R_i\}\}, i = 1 \dots m \tag{3}$$

$$R_i := S \times S \rightarrow [0, 1], i = 1 \dots m. \tag{4}$$

Although simple in conception, this extension is not directly supported by current standards (at least pending the results of the latest standardization efforts of the W3C). In Semantic Web languages, such as RDF and OWL, binary relations between two individuals are represented by means of object properties. An object property is characterized by a domain and a range but it cannot have its own properties such as the severity, strength or relevance of the relation. This, consequently, means that fuzziness cannot be incorporated in RDF or OWL in a seamless way. In order to deal with this constraint we utilize an ontology design pattern known as reification (Mylonas and Wallace, 2006). The basic idea of this pattern is that an object property that needs to have its own properties is represented as a class. Then, the instances of this property become instances of the class (see *W3C, RDF Reification*).

3. Size and time considerations

A known issue related to the practical use of ontologies in general purpose applications is that of the complexity related to their size and the scalability issues emerging from that. Existing practical applications of ontologies are typically based on simple ontological representations that in most cases have been developed by few people or even a single person over a very small period of time. As a result, although the methodologies have proven their worth in such limited cases, they cannot be generalized to treat a general case, as, for example, the general purpose analysis of web content. For such an application a detailed and complete (to a reasonable degree) representation of generic human knowledge would be required. The practical utilization of an ontology of such a scale would require at least the following two distinct scalability issues to be tackled:

1. scalability with regards to the memory size needed to load and the time needed to access the ontological information
2. scalability with regards to the processing time required to consider all the information in the ontology.

3.1. Size considerations

Clearly, algorithmic exploitation of the information that is conveyed by the ontological relations cannot be based on direct access of the XML representation structure, when the size of the ontology and the complexity of the application are considerably augmented; the reduced speed of XML based operations alone would be enough to prohibit any thought for such applications. Therefore, in order to practically consider the utilization of large, general purpose ontologies we need to first define a suitable model to be used for their representation in computer memory during semantic operations.

Focusing on the (fuzzy or crisp) binary relations included in ontologies, which are the main focus of this work, we can easily see that in order for such information to be readily available for practical utilization all entries referred to the same semantic relation have to be extracted from the RDF or OWL document and stored as an $n \times n$ relation, n being the count of semantic concepts in the ontology, i.e. the cardinality of S; this $n \times n$ relation is in any case a very sparse one, as not every concept of the real world is related to every other concept.

The conventional modeling for sparse $n \times n$ relations is that of linked lists for sparse matrices, which augments access time from $O(1)$ to $O(n)$. Although the storage requirements are much smaller, the representation model remains insufficient for practical general purpose applications, where complex operations utilizing a binary relation have to be performed before the system response is determined, and number n is large.

The representation model proposed in order to overcome these limitations is as follows: a binary relation is represented using two sorted vectors. The first vector is sorted according to row index i, and in case of identical row positions, column position j is utilized, and vice versa for the second vector.

Following this approach, the two vectors that correspond to the sample sparse relation in Fig. 1 are listed below: [(1,2), (1,5), (1,6), (2,1), (2,4), (2,5), (3,2), (4,4), (4,6), (5,1), (5,4)] [(2,1), (5,1), (1,2), (3,2), (2,4), (4,4), (5,4), (1,5), (2,5), (1,6), (4,6)] This representation model preserves the storage merits of the classical sparse matrix implementation with the linked lists. Moreover, access time for a specific element, row or column has a computational complexity of $O(\log n)$, when utilizing a divide and conquer approach. Moving a step further, we utilize AVL trees for the representation of each of the two vectors (Adelson-Yelskii and Landis, 1962). In this way, not only access but also insertion, deletion and update times are limited to $O(\log n)$, making incremental operations on the ontological relations feasible.

An additional consideration related to size scalability is that of information

$$\begin{bmatrix} & (1,2) & & (1,5) & (1,6) \\ (2,1) & & (2,4) & (2,5) & \\ & (3,2) & & & \\ & & (4,4) & & (4,6) \\ (5,1) & & (5,4) & & \end{bmatrix}$$

Figure 1. A sample sparse relation.

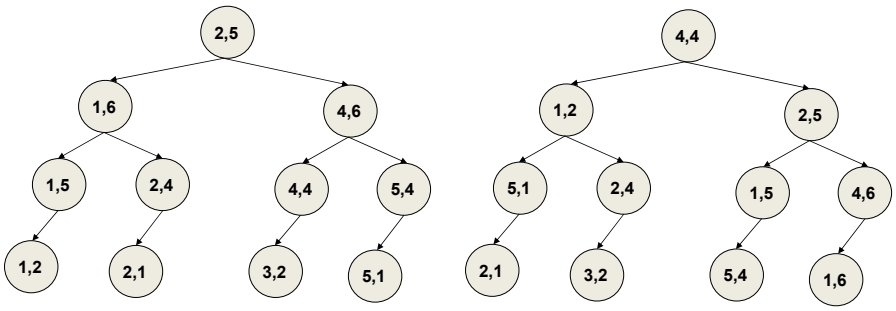

Figure 2. The AVL trees that correspond to the sparse relation in Fig. 1.

dispersion. The existence of numerous semantic relations in the definition of an ontology results in meaningful and potentially useful information being dispersed among them, and therefore no single relation on its own suffices for a complex and ambitious task such as generic content analysis and querying. On the other hand, the utilization of a methodology that considers multiple semantic relations leads to increased memory (and time) requirements that can, in turn, render the methodology practically inapplicable.

In order to overcome this barrier we introduce the notion of knowledge view. We define a view of the relational knowledge included in an ontology as a new single relation that is suitable for a specific task, computed via some combination of the originally available relations. Views have the advantage of being "ready to use" for a specific task, but do not maintain the full extent of semantic information conveyed by the original relations and thus cannot replace them in ontological representations. Quite the contrary, given the original semantic relations in an ontology, a different view can be developed every time a new task is to be tackled. In a general mathematical formulation a knowledge view T is seen as

$$T = \bigcup R_i^k, i = 1 \ldots m, k \in \{-1, 0, 1\}, \tag{5}$$

where $k = 0$ indicates that R_i does not participate in the view in question, $k = 1$ that it participates "as is", while $k = -1$ indicates that R_i participates

in the view after being inverted. A typical knowledge view, which we will be using later in this work for the tasks of query processing, document analysis and information retrieval, is:

$$T = Pa^{-1} \cup Sp \cup Sb \cup Ex \cup Pr^{-1} \cup In \cup Lo^{-1} \tag{6}$$

where the considered semantic relations are Pa for part, Sp for specialization, Sb for subarea, Ex for example, Pr for property, In for instrument and Lo for location.

Degrees in these relations have an intuitive meaning; for the part relation, for example, the degree indicates how extensive or important the part is with respect to the whole, making $Pa(s_1, s_2) = 0.1$ meaningful if s_1 is the "body" and s_2 is the "finger", while $Pa(s_1, s_2) = 1$ could only hold if $s_1 = s_2$; the same goes for all other relations as well.

3.2. Time considerations

Although this approach permits us to load into memory information from considerably larger (as far as count of concepts is concerned) ontological relations, an important computational issue remains: that of recursion. Conventional content based reasoning methodologies have an inherent recursive nature, which makes them inapplicable for cases where large transitive paths may be found in relations. Clearly, the consideration of large ontologies for the automated analysis and querying of generic content, which is the focus of this work, falls into this category.

In order to overcome this problem, in this work we develop and utilize methodologies that are based on the transitivity of fuzzy ontological relations. Taking advantage of this property we may alleviate the need for recursion, thus making the practical exploitation of the information contained in the relations feasible.

The transitive property of binary relations, due to its physical meaning, is closely related to the study of graphs. In that framework, transitive closure of a relation is equivalent to the detection of the pairs of vertices that are either directly connected or connected via some path. Thus, the majority of existing literature on transitive closure algorithms has focused mainly on the cases of undirected crisp graphs (Seidel, 1995) and crisp graphs (Baker, 1962; Warren, 1975).

Computationally enhanced transitive closure methodologies found in the literature, such as Lee (2001), De Meyer, Naessens and De Baets (2004), cannot be used in our case as their application is limited to the case of $\max - \min$ transitivity, which is not compatible with the semantics of ontological relations; $\sup - t$ transitivity is required, with t being an Archimedean norm. The computation of the transitive closure for relations such as the ones considered herein using the conventional approach requires several relation compositions (Warshall, 1962; Dunn, 1974) and is not practically applicable for the case of relations of the

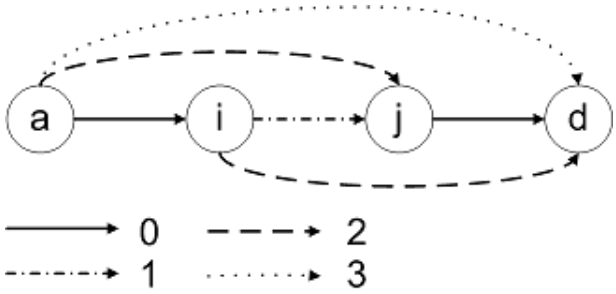

Figure 3. The main concept behind ITU and ITC algorithms

size discussed herein. What is worse, given the long computation times for the transitive closure, an incremental trial and error approach to the development of the ontology would not be possible.

A solution to this is the exploitation of the sparse nature of the relations in order to reduce the computational complexity of the closure operation. As seen in Fig. 3, when a new element is inserted into the originally transitive relation T, only three additional operations need to be performed in order to maintain transitivity, rather than a complete relation composition; this is the main concept of the ITU (Incremental Transitive Update) algorithm. In the following, the algorithm is formalized, with parameters T, i and j standing for the originally transitive relation and the source and destination, respectively, of the new element that has been added to the relation.

Algorithm ITU:

Parameters: T i j

Output: T

1. Identify the fuzzy set A of ancestors of entity i in relation T. Degrees in A are determined as $A(s) = T(s, i), s \in S$.
2. Identify the fuzzy set D of descendants of entity j in relation T. Degrees in D are determined as $D(s) = T(j, s), s \in S$.
3. For each element s appearing in A assign

$$T(s, j) \leftarrow \sup(T(s, j), A(s) \wedge_t T(i, j)). \tag{7}$$

4. For each element s appearing in D assign

$$T(i, s) \leftarrow \sup(T(i, s), T(i, j) \wedge_t D(s)). \tag{8}$$

5. For each element s_1 appearing in A and s_2 appearing in D assign

$$T(s_1, s_2) \leftarrow \sup(T(s_1, s_2), A(s_1) \wedge_t T(i, j) \wedge_t D(s_2)). \tag{9}$$

In the above notation $\wedge_t$ indicates a fuzzy intersection, where the t norm is used instead of min. When the algorithm terminates we have T being equal to the transitive closure of the original T fuzzy binary relation.

If relation T is reflexive, then $T(i,i) = 1$ and $T(j,j) = 1$ and thus $A(i) = 1$ and $D(j) = 1$. In this case, the above process can be simplified by omitting steps (3) and (4), as they are included in step (5).

This can easily be extended to compute the transitive closure of any fuzzy binary relation, thus producing the ITC (Incremental Transitive Closure) algorithm

Algorithm ITC:

Parameters: T

Output: T'

1. Create an empty binary relation T'.
2. For each non zero element $T(i,j)$ in the initial relation T

   (a) Assign

   $$T'(i,j) \leftarrow \sup(T'(i,j), T(i,j)). \tag{10}$$

   (b) Run the incremental update algorithm with parameters T', i, j:

   $$T' \leftarrow ITU(T', i, j). \tag{11}$$

When the algorithm terminates, we have T' being equal to the transitive closure of the original fuzzy binary relation T.

4. The notion of context

In general, the term "context", for many the holy grail of semantic processing, refers to whatever is common among a set of elements. In this work, where the elements are semantic entities, queries and documents, the term "context" may refer to the common meaning of a set of entities, or to the overall topic of a query or document, respectively.

A query, as well as a document, will be represented in this work by its mapping to semantic entities. Therefore, the context of a query is again defined via the semantic entities that are related to it. The fact that relation T described in Section 3 is (almost) an ordering relation allows us to use it in order to define, extract and use the context of a query, or of a set of semantic entities in general. Relying on the semantics of the relation T, we define the context $K(s)$ of a semantic entity $s \in S$ as the set of its ancestors in relation T:

$$K(s) = \sum a/T(a,s), a \in S \tag{12}$$

This set also includes the semantic entity s in question.

Assuming that a set of entities $S' \subset S$ is crisp, i.e. all considered entities belong to the set with degree one, the context of the group, which is again a set of semantic entities, can be defined simply as the set of their common descendants

$$K(S') = \bigcap K(s_i), s_i \in S'. \tag{13}$$

Obviously, as more entities are considered, the context becomes narrower, i.e. it contains less entities and to smaller degrees. When the definition of context is extended to the case of fuzzy sets of semantic entities, this inequality must still hold. Moreover, we demand that the following are satisfied as well:

- $S''(s) = 0 \rightarrow K(S'') = K(S'' - \{s\})$, i.e. no narrowing of context.
- $S''(s) = 1 \rightarrow K(S'') \subseteq K(s)$, i.e. full narrowing of context.
- $K(S'')$ decreases monotonically with respect to $S''(s)$,

where S'' is a fuzzy set of entities (i.e. a set similar to S', in which each element s_i is accompanied by a degree of membership in the set). Taking these into consideration, we demand that, when S'' is fuzzy, the "considered" context $\mathcal{K}(s)$ of s, i.e. the entity context when taking its degree of participation in the set into account, becomes low when the degrees of taxonomy are low and the degree of participation $S''(s)$ is high. Therefore:

$$cp(\mathcal{K}(s)) \doteq cp(K(s)) \cap (S''(s) \cdot S) \tag{14}$$

where cp is an involutive fuzzy complement, and $\cap$ and $\cup$ correspond to a t-norm and a t-conorm, which are dual, with respect to cp. By applying de Morgan's law, we obtain:

$$\mathcal{K}(s) \doteq K(s) \cup cp(S''(s)). \tag{15}$$

Then the context of the set is easily calculated as follows:

$$K(S'') = \bigcap \mathcal{K}(s_i), s_i \in S''. \tag{16}$$

Considering the semantics of the relation T and the process of context determination, it is easy to realize that when the entities in a set are highly related to a common meaning, the context will have high degrees of membership for the entities that represent this common meaning. Therefore, the quantity $h(K(S''))$, where $h(\cdot)$ symbolizes the height of a fuzzy set, may be used as a measure of the semantic correlation of entities in the set S''. We will refer to this measure as intensity of the context.

5. Flexible querying

Flexible querying is a term coined for and originally associated with database querying. In that scope, it refers to the alleviation of the rigidity of traditional

database querying by providing users with more user friendly and human intuitive approaches to querying. Since then, the term has exceeded the boundaries of database querying and is more generally utilized to describe the offering of more intuitive querying tools in all aspects of information retrieval (see, for example, Bordogna and Pasi, 2002). The same approach is used herein, with flexible querying referring the the development of more intuitive ways to perform free text searches in open- or closed-world collections of documents.

5.1. Definitions

Given a set of available elements $\tilde{D}$, an information retrieval system (IRS) undertakes the tasks of organizing the elements, using a suitable modeling, and controling/facilitating access to them. It is also involved in the handling of the interaction with end users, the extraction of information concerning their desires and the recall of elements $\tilde{P} \subseteq \tilde{D}$ suiting these desires. In order for all these to be realized, three main tasks need to be supported, namely understanding of element content, understanding of user desire, and matching between the two. These correspond to the components of document indexing, query processing and matching mechanism of an IRS, as is seen in Fig. 4.

Through the indexing, each element $\tilde{d}$ of set $\tilde{D}$ is associated with its modeled representation $d \in D$, with $d = \sum s_i/w_i, s \in S$, thus generating relation I, the index of the system. In the above notation w_i is the degree, to which entity s_i participates in fuzzy set d, i.e. the degree, to which it is associated to document $\tilde{d}$, and D is the space in which documents are modeled by the IRS:

$$f_1 \doteq I : \tilde{D} \times S \to [0,1]. \tag{17}$$

In a similar manner, query processing associates the user query $\tilde{q}$ to its modeled representation $q \in Q$, with $q = \sum s_i/w_i, s \in S$,

$$f_2 : \tilde{Q} \times S \to [0,1] \tag{18}$$

where $\tilde{Q}$ is the space, in which user queries are expressed, and Q the space, in which they are modeled by the system.

Finally, the matching mechanism uses the modeled expressions of both user query and available elements in order to provide users with a suitable system response,

$$f_3 : Q \times D \to [0,1]. \tag{19}$$

In all the above we have assumed the general case, in which indexing, query processing and matching are all a matter of degree.

Obviously, the efficiency and effectiveness of the IRS it tied to the calculation and meaningfulness of f_1, f_2 and f_3. In this section we explain how we exploit the models and methodology presented in Sections 2 and 3 in order to incorporate knowledge, semantics and reasoning under uncertainty in the definition and calculation of the functions that constitute the backbone of the IRS.

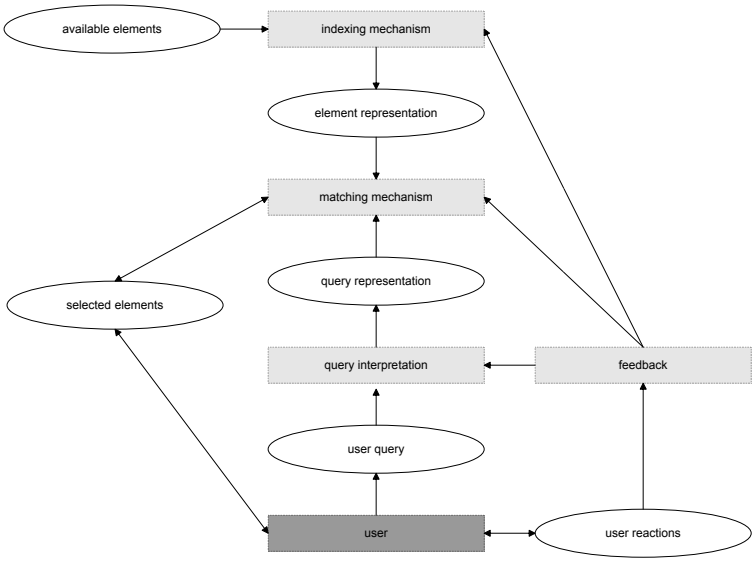

Figure 4. The structure of an information retrieval system

5.2. Query disambiguation

In Section 4, although never stated directly, it was implied that the mapping of the terms provided by the user in a textual query to the corresponding semantic entities is one-to-one, error free and trivial. This can be true for some cases, but in the general case there are important issues one needs to consider as for example that distinct semantic entities may have common textual descriptions. As a simple example, let us consider the case of term "element"; at least two distinct semantic entities correspond to it: "element$_1$", which is related to chemistry, and "element$_2$", which is related to XML.

Let us now suppose that a query containing the term "element" is given by a user. If the remaining terms of the query are related to chemistry, then it is quite safe to suppose that the user is referring to semantic entity "element$_1$" rather than to semantic entity "element$_2$". This implies that the context of the query can be used to facilitate the process of semantic entity determination. However, the estimation of the query context, as defined in Section 4, cannot be performed before the mapping of the query to semantic entities is completed. Therefore, query interpretation needs to take place simultaneously with context detection. We propose the following method:

Let the textual query $\tilde{q}$ contain the terms

$$\tilde{q} = \{t_1, t_2, \ldots, t_{n_q}\}. \tag{20}$$

Also for each term t_i let

$$S_i = \{s_{i1}, s_{i2}, \ldots, s_{in_i}\} \tag{21}$$

be the set of semantic entities that may be expressed via t_i in a text. Then, the count of distinct combinations of semantic entities that may be used to model the user query is

$$|S_q| = \prod n_i, i = 1 \ldots n_q \tag{22}$$

where

$$S_q = S_1 \times S_2 \times \ldots \times S_{n_q}. \tag{23}$$

For each of these alternatives we calculate the context using the methodology developed in Section 4. The combination that produces the context with the greater intensity is the one that is selected as the valid modeling for the user query,

$$q_{selected} = \{q \in S_q : h(K(q)) = \max(h(K(q_i))), q_i \in S_q \tag{24}$$

where h is the intensity function defined earlier.

This algorithm, proposed for query disambiguation, i.e. for the implementation of f_2 as defined in the beginning of this section, is exhaustive. Still, this is not an important drawback that may lead to scalability concerns as:

- queries do not contain large numbers of terms,
- the terms for which more than one semantic entity may be chosen are rare and
- the number of distinct semantic entities that may have a common textual description is not large.

An approach, different in implementation but similar in concept (i.e. based on the abstract notion of context) appears in Baziz, Boughanem and Aussenac-Gilles (2005), where once the words have been extracted from the text, the best semantic network to model them is selected; in this network similarity measure values between connected nodes weigh the links.

5.3. Document analysis

The mapping of the terms found in a textual document to semantic entities may be performed using an approach similar to the one presented above for query disambiguation, by limiting each time the scope of the context to a single paragraph or sentence. The index can also be populated via other automated methodologies, as for example automatic audio transcription, video analysis and object detection and so on, thus also providing fuzziness based on the certainty or degree to which entities are detected.

Still, the true role of the document indexing component is to "comprehend" the contents of a document in order to provide an indexing that accurately and fully carries the meaning of its contents. The approach utilized in this work for this analysis of the document is to extract the topics associated with the document. This is achieved based on the definition of context provided in Section 4; this definition of context has the requirement that the input fuzzy set of semantic entities is normal, and therefore we also demand that the index is normal for each document, i.e.:

$$\forall \tilde{d} \in \tilde{D} \exists s \in S : I(\tilde{d}, s) = 1. \tag{25}$$

Based on the index, and the knowledge contained in the considered knowledge view T, we aim to detect the degree, to which a given document $d \in D$ is related to a topic/thematic category $tc \in TC$, with $TC \subseteq S$. We will refer to this degree as $R_{\mathrm{TC}}(tc, d)$. In other words, we aim to calculate the relation:

$$R_{\mathrm{TC}} : TC \times D \to [0, 1]. \tag{26}$$

In order to simplify some of the formulas to follow, we shall use this notation interchangeably with

$$S_{\mathrm{TC}}(\tilde{d}) : TC \to [0, 1] \tag{27}$$

with $S_{\mathrm{TC}}(\tilde{d})$ being the fuzzy set of thematic categories that are related to document $\tilde{d}$ and

$$S_{\mathrm{TC}}(c) : TC \to [0, 1] \tag{28}$$

with $S_{\mathrm{TC}}(c)$ being the fuzzy set of thematic categories that are related to fuzzy set c.

In designing an algorithm that is able to calculate this relation, in a meaningful manner, a series of issues need to be tackled:

1. A semantic entity may be related to multiple, unrelated thematic categories.
2. A document may be related to multiple, unrelated thematic categories.
3. The semantic index may have been created in an automated manner. Thus, existence of random, and therefore misleading semantic entities cannot be excluded.
4. Semantic relations are always a matter of degree. Therefore, correlation between a document and a thematic category is also a matter of degree.

According to issue 1, it is necessary for the algorithm to be able to determine, which thematic categories are indeed related to a given document. In order for this task to be performed in a meaningful manner, the common meaning of the remaining entities that index the given document needs to be considered as

well. On the other hand, when a document is related to more than one thematic categories, as issue 2 points out, we should not expect all the terms that index it to be related to each one of the thematic categories in question; quite the contrary, we should expect most entities to be related to just one of them. Therefore, a clustering of semantic entities, based on their common meaning, needs to be applied. In this process, entities that are misleading will probably not be found similar with other entities that index a document. Therefore, the cardinality of the clusters may be used to tackle issue 3. Finally, issue 4 is easily solved by allowing the overall algorithm to be fuzzy.

As far as the clustering of the semantic entities is concerned, agglomerative methods are more flexible than their partitioning counterparts, in that they do not need the number of clusters as an input (Theodoridis and Koutroumbas, 1998). Their generic structure is as follows:

1. Turn all considered elements into singletons, i.e. into independent clusters of one element.
2. Compute all cluster-to-cluster distances.
3. If a termination criterion is met, i.e. the smallest computed distance is larger than a given threshold then terminate.
4. Merge the two clusters for which the smallest distance has been computed.
5. If a termination criterion is met, i.e. the number of remaining clusters is equal to a given termination threshold, then terminate.
6. Continue at step 1.

Herein we follow an agglomerative approach to cluster the crisp (we ignore degrees) set of semantic entities that index a document, using the context to define both cluster distance and termination criteria. Still, the agglomerative approach is less robust in other ways:

- It only creates crisp clusterings, i.e. it does not support degrees of membership in their output.
- It only creates partitions, i.e. it does not allow for overlapping among the detected clusters.

Both of the above are great disadvantages for the problem at hand, as they are not compatible with the task semantics: in real life, a semantic entity may be related to a topic to a degree other than 1 or 0, and may also be related to more than one distinct topics. In order to overcome such problems, we describe in the following a method for fuzzyfication of the partitioning we have produced via the original agglomerative clustering. In this way the cluster cardinalities will be corrected, so that they may be used for the meaningful extraction of thematic categories.

Each cluster c is described by the crisp set of semantic entities S_c that belong to it. Using those, we may create a fuzzy classifier, i.e. a function C_c that will

measure the degree of correlation of a semantic entity s with the cluster c,

$$C_c : S \to [0, 1]. \tag{29}$$

Obviously, a semantic entity should be considered correlated with c, if it is related to the common meaning of the semantic entities in S_c. Therefore, the quantity

$$C_1(c, s) = h(K(S_c \cup \{s\})) \tag{30}$$

is a meaningful measure of correlation, where h is again the intensity function defined earlier. Of course, not all clusters are equally compact; we may measure cluster compactness using the similarity among the entities a cluster contains, i.e. using the intensity of the cluster context. Therefore, the aforementioned correlation measure needs to be adjusted, to the characteristics of the cluster in question:

$$C_2(c, s) = \frac{C_1(c, s)}{h(K(c))}. \tag{31}$$

It is easy to see that this measure has the following properties:

- $C_2(c, s) = 1$ if the semantics of s imply it should belong to c. For example $C_2(c, s) = 1, \forall s \in S_c$
- $C_2(c, s) = 0$ if the semantics of s imply it should not belong to c.
- $C_2(c, s) \in (0, 1)$ if s is neither totally related, nor totally unrelated to c.

These are the very same properties that we desire for the fuzzy classifier of the cluster, and therefore:

$$C_c(s) \doteq C_2(c, s). \tag{32}$$

Using such classifiers, we may expand the detected crisp partitions, so as to include more semantic entities, as follows: partition c is replaced by cluster

$$c' = \sum s / C_c(s), s \in I(d). \tag{33}$$

Obviously $c' \supseteq c$. Having computed the fuzzy clustering of the semantic entities that index a document, we are now ready to perform the actual extraction of the thematic categories that are related to it.

We have defined the context of a set of semantic entities as a fuzzy set of semantic entities itself; this contains the entities that describe the common meaning of the original set. The thematic categories that are contained in the context of a cluster of semantic entities are obviously thematic categories that are related to it and to the whole document. Based on this concept, the following steps lead to the thematic categorization of documents:

First of all, the process of fuzzy hierarchical clustering has been based on a crisp set, thus ignoring fuzziness in the index. In order to incorporate this information in the results of the clustering process, we adjust the degrees of membership for clusters as follows:

$$c''(s) = c'(s) \wedge_t I(\tilde{d}, s). \tag{34}$$

The semantic nature of this operation demands that an Archimedean t-norm be used.

From each one of those clusters, we may extract the corresponding thematic categories. Obviously, thematic categories that are not contained in the context of c'' cannot be selected as being related to it. Therefore

$$S_{\mathrm{TC}}(c'') \subseteq S'_{\mathrm{TC}}(c'') \doteq w(K(c'') \cap TC) \tag{35}$$

where w is a weak modifier. Modifiers, which are also met in the literature, like linguistic hedges, are used (in this work) to adjust mathematically computed values so as to match their semantically anticipated counterparts.

In the case that the semantic entities that index the document $\tilde{d}$ are all clustered in a unique cluster c'', $S_{\mathrm{TC}}(\tilde{d}) = S'_{\mathrm{TC}}(c'')$ would be a meaningful approach. On the other hand, when more than one clusters are detected, then cluster cardinalities have to be considered as well. Clusters of extremely low cardinality probably only contain misleading entities, and therefore need to be ignored in the estimation of $S_{\mathrm{TC}}(d)$. On the contrary, clusters of high cardinality almost certainly correspond to the distinct topics $\tilde{d}$ is related to, and need to be considered in the estimation of $S_{\mathrm{TC}}(\tilde{d})$. The notion of "high cardinality" is modeled with the use of a large fuzzy number L. $L(\cdot)$ is the truth value of the proposition "the cardinality of a is high".

The set of thematic categories that correspond to a document is computed from the remaining clusters, after adjusting membership degrees according to scalar cardinalities, as follows:

$$S_{\mathrm{TC}}(\tilde{d}) \doteq \bigcup S_{\mathrm{TC}}(c''), c'' \in G \tag{36}$$

with

$$S_{\mathrm{TC}}(c'') = S'_{\mathrm{TC}}(c'') \cdot L(|c''|) \tag{37}$$

where a fuzzy co-norm (not necessarily max) is used for the calculation of the sum, G is the set of fuzzy clusters with adjusted membership degrees and $|c''|$ the scalar cardinality of c''.

It is easy to see that $R_{\mathrm{TC}}(\tilde{d}, tc)$ will be high if a cluster c'', whose context contains tc, is detected in the indexing of the document, and additionally, the cardinality of c is high (i.e. the cluster is most probably not composed of misleading entities) and the degree of membership of tc in the context of c'' is high.

5.4. Personalized information retrieval

To this day, out of the components we have identified in IRSs it is the process of matching between the query and the index that has received the most attention, while query and document analysis have been relatively neglected. The reason behind this is that in the IRSs based mainly on the processing of terms not much intelligence could be incorporated in indexing and query analysis, thus making the task of matching the most important one in the process of information retrieval.

On the other hand, in the approach presented in this work both the query analysis and document indexing steps have been performed in an intelligent, context aware, knowledge based approach that in addition to degrees of association also took under consideration sources of uncertainty such as ambiguity in querying and misleading terms in documents. Moreover, queries and documents are both modeled as fuzzy sets on S, which facilitates their association. Therefore, a simple and straightforward approach to the design of the matching mechanism could be sufficient.

Still, this step does offer one additional opportunity to enhance the effectiveness and flexibility of the overall approach. This refers to the ability to modify matching process in accordance to the user profile, in order to also incorporate personalization in the overall process of information retrieval.

A simple representation for the user profile that also allows for degrees of preference is that of a fuzzy set defined on the set of semantic entities S. Unfortunately, as explained below, it is easy to see that such an approach is not adequate.

First of all, let us consider the (not rare) case in which a user has various preferences. When the user poses a query that is related to one of them, then that preference should be used to drive the selection of relevant documents. Usage of preferences that are unrelated to the specific query may only be viewed as addition of noise, as any proximity between selected documents and these preferences is clearly coincidental, in the given context. In order to limit this inter–preference noise, we need to be able to identify which preferences are indeed somewhat related to the considered user query, and to what extent. Therefore, each preference needs to be stored separately; a single fuzzy set is not sufficient for the representation of user preferences.

Let us now consider the case in which we can infer (for example by monitoring and later analyzing user actions or simply through a questionnaire), that the user is not interested in documents of a specific subject. In such a case, in addition to preferences, special care must be taken for the representation of dislikes.

When applying the above in designing a user preference data model, we need to keep in mind that a user is interested in each subject to some degree. Therefore, interests (and dislikes) need to be characterized by a degree of intensity. Moreover, the process of mining the interests is not free of uncertainty itself,

and therefore, mined interests also need to be accompanied by a degree of confidence. This leads to the use of two distinct, seemingly independent degrees, for each preference.

Still, it is easy to see that an intense interest will probably be mined with a greater degree of confidence, than one that is not as intense. It is this observation that allows us to suppose that the degree of confidence and the degree of intensity are not independent. In other words, although two distinct degrees are related to each interest, an intensity degree and a confidence degree, a single degree is sufficient for their representation.

In compliance with the principles presented above, we use the following formal representation of user preferences P in a user profile:

$$P = \{U^+, U^-\} \tag{38}$$

where U^+ refers to the set of interests and U^- refers to the dislikes,

$$U^- = \sum s_i/u_i^-, \quad i \in N_n \tag{39}$$

where u_i^- is the degree of participation of entity s_i in U^-,

$$U^+ = \{U_i^+\}, \quad i \in N_k \tag{40}$$

where k is the count of distinct positive interests that are contained in the user profile, and

$$U_i^+ = \sum s_j/u_{ij}^+, \quad i \in N_k, j \in N_n \tag{41}$$

where u_{ij}^+ is the degree of participation of entity s_j in U_i^+.

It is easy to see that this definition allows for the overlapping of interests and dislikes. Moreover, it allows for the participation of the same semantic entity in different interests, and to different degrees.

The utilization of two distinct sets to model user desires is also found in bipolar queries (Dubois and Prade, 2008; Zadrożny and Kacprzyk, 2006); contrary to here, where the two sets are utilized to model different types of preference (i.e. positive and negative), in bipolar querying the two sets are used to model different levels of desire (i.e. required elements and preferred elements).

Based on the definition of context, as provided in Section 4, we may detect the fuzzy set $U^K = \sum U_i^+/k_i$ of interests that are related to the query context as follows:

Obviously, in the case that no context can be detected in the query, we would like the whole set of user preferences to be selected. More formally, when $h(K(q))$ approaches zero, U^K approaches U^+. If, on the other hand, the query context is intense, preferences that do not intersect with the context should not be considered, thus eliminating inter – preference noise. The remaining

preferences are considered in proportion to the intensity of their intersection with the context.

A simple formula that complies with the above guidelines is the following:

$$k_i = \frac{h(U_i^+ \cap K(q))}{h_q} \vee c(h_q) \tag{42}$$

where $h_q = h(K(q))$ is the intensity of the query context.

Negative preferences that are out of context do not add noise, and consequently do not have to be filtered with the use of the query context. Therefore, the context adapted user preferences are:

$$P^K = \{U^K, U^-\}. \tag{43}$$

Within a specific query context we may demand, as a minimum consistency criterion, that the context adapted user preferences do not contain both positive and negative preferences for the same semantic entities. As we have not imposed such a criterion in the process of constructing or expanding the profile, it is possible that P^K is not in accordance with this criterion.

In order to specify the optimal way of altering P^K, so as to make it compatible with our consistency criterion, we start, once again, by identifying some necessary conditions. First of all, interests are generally extracted with greater confidence than dislikes; the users themselves may be more aware of their dislikes than their interests, but the latter are easier to detect than the former when relying solely on automated methodologies that do not include polling the user. Therefore, interests should be treated more favorably.

Obviously, if only interests correspond to a specific semantic entity, then their degrees must not be altered. Likewise, if only a dislike corresponds to a specific semantic entity, then its degree must not be altered. In general, the degrees of interests should increase monotonically with respect to their original value, and decrease monotonically with respect to the original value of the corresponding dislike, and vice versa. These conditions are not particularly strict as there are infinite compliant implementations; we choose a simple linear approach:

A degree of favor $a \in [0, 1]$ is defined. It indicates the degree to which interests are favored, with respect to dislikes. When $a = 1$ there is no distinction, while, as a approaches 0, dislikes are completely ignored. The decision on whether interests or dislikes dominate a semantic entity s_i, is based on the sign of the expression

$$\max_j(k_j u_{ji}^+ - a u_i^-). \tag{44}$$

If $k_j u_{ji}^+ - a u_i^-$ is positive for at least one j, then the negative preference cannot dominate, i.e $\hat{u}_i^- = 0$; by $\hat{u}_i^-$ we denote the intensity of the negative preference after the adjustment for consistency. Likewise, if $\max_j(k_j u_{ji}^+ - a u_i^-) < 0 \; \forall j$, then $\hat{u}_i^+ = 0 \; \forall j$.

The adjusting of dominating values is performed using the following formulas:

$$\hat{u}_{ji}^{+} = u_{ji}^{+} - \frac{au_i^{-}}{k_j} \tag{45}$$

$$\hat{u}_i^{-} = u_i^{-} - \frac{max_j(k_j u_{ij}^{+})}{a}. \tag{46}$$

The above approach produces a valid (i.e. consistent) context adapted user profile; this profile may only be used in the process of matching the query, for which it was calculated.

This brings us up to the actual matching between the query model q and the index I (or the thematic categorization R_{TC}), while considering the context adapted user profile. This is accomplished as a two step process: first matching is performed directly without considering user preferences by combining matches for each term $s_i \in q$

$$D_q = \bigcap_i D_{q_i} \tag{47}$$

$$D_{q_i} = \sum \tilde{d}/I(\tilde{d}, s_i). \tag{48}$$

Then interests and dislikes are used in order to adjust degrees for each selected document. For each interest U_i^{+} degrees may be augmented:

$$D_q(\tilde{d}) \leftarrow (D_q(\tilde{d}))^{1+h(K(d \cap U_i^{+}))} \tag{49}$$

while dislikes may reduce them:

$$D_q(\tilde{d}) \leftarrow (D_q(\tilde{d}))^{\frac{1}{1+h(K(d \cap U^{-}))}}. \tag{50}$$

6. Discussion and implementations

Comparing this work to other in the literature is not easy. For the querying part the main reason for this is that, as is evident from the description of our framework in the previous sections, the acquired results are greatly dependent on the contents and completeness of the considered ontology. This does not only make any comparative study difficult when a complete ontology is not readily available (clearly the development of such an ontology is not an easy task either) but also it renders it meaningless when comparing with any methodology that is not ontology-based; better results can always be acquired not only by enhancing the methodology, but also by merely by properly altering the ontology. When it comes to the second part, i.e. personalized information retrieval, this is hindered by the subjectivity issues of personalization. Still, a qualitative comparative study is possible and such a discussion follows.

As has already been explained in the literature, querying based on terms rather than concepts is hindered by a number of difficulties. Therefore, concept

based querying is typically deemed as superior. It would of course be misleading to ignore the fact that quite successful term based information retrieval methodologies have been developed and reported. One thing that the most effective of these approaches have in common is that they are based on the successful statistical analysis of the whole document base, or of an indicative sample of it, thus limiting their application to closed-world or closed-world-like situations. The approach presented herein overcomes this problem, as the knowledge, upon which its operation is based, does not come from the analysis of the documents themselves but from an independently developed ontology. As a consequence, the system can be fully developed when no documents are available and then successfully applied to any document or document collection, provided of course that the associated ontology has been developed correctly and sufficiently.

On the other hand, our approach lacks with respect to conventional information retrieval approaches in exactly that: the fact that the knowledge required for its operation is not generated automatically but rather through a painful and extremely time consuming manual process. Each relation element (except for those that can be inferred from the transitive nature of the considered relations) has to be provided manually by a human.

As has already been mentioned, one characteristic of our approach is that all of the presented methodologies are based on a common fuzzy ontological framework. Therefore, the first step in any implementation of our approach is the implementation of the fuzzy ontological framework. This has been accomplished in three distinct steps:

- A framework has been developed to support the modeling of fuzzy sets and fuzzy relations, as well as the implementation of fuzzy operators and operations.
- Reification was utilized to allow for fuzziness to be incorporated in the definition of ontologies.
- The two aforementioned components were integrated via the development of OWL input and output interfaces for the fuzzy framework.

For the first and last steps the Java^{TM} environment was utilized; the second step is inherently supported by RDF vocabulary.

The fuzzy relational library that we developed (Wallace, no date), in addition to the support for the other methodologies presented in this work, also provides the means to evaluate the theoretically examined properties of the time and space reduction methodologies. Indeed, experiments indicate that double precision fuzzy binary relations between as many as 70000 elements can easily be loaded into memory (impossible when considering conventional array representation) and the transitive closure of such a relation can be achieved within seconds (an operation that when attempted on the same computer, using the proposed sparse representation model but the conventional transitive closure algorithm took longer than 120 hours to complete).

The document analysis methodology has also been implemented and tested as a stand alone tool. The practical application of this tool is demonstrated in Fig. 5, where it is applied towards the analysis and automatic thematic categorization of a document. This application was evaluated in the framework of the FAETHON project, where it was proven that automated text analysis based on fuzzy relational information can provide for effective thematic categorizations. The main limitation identified was the poor performance of the module matching terms to concepts (a problem also augmented by the fact that a portion of the considered documents was in Greek) which led to the development of the disambiguation methodology.

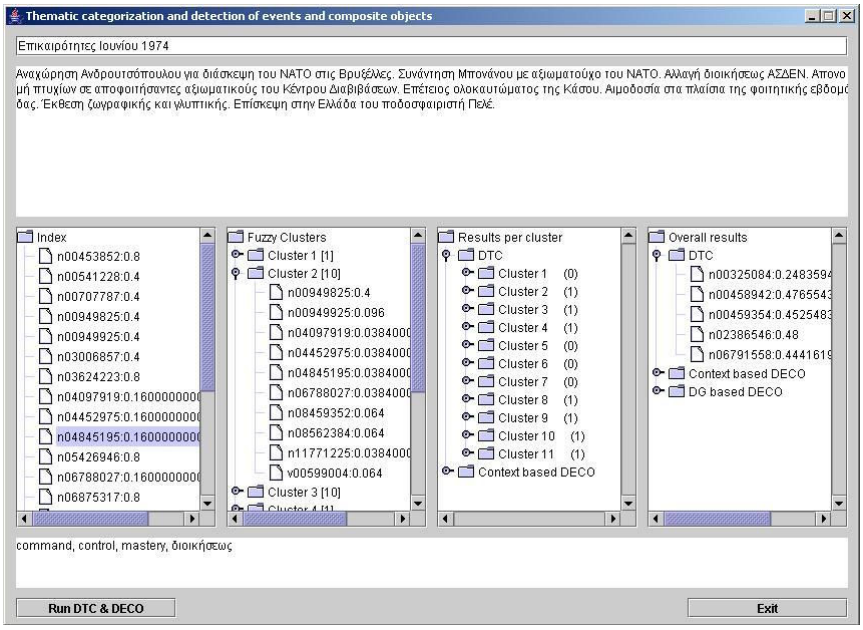

Figure 5. The stand-alone implementation of the document analysis tool.

The reification component was developed and incorporated in the overall framework in the context of the EDRASIS project, which develops a tool for flexible querying of business operational documents; this system also contains the personalized querying component. The complete platform of the project has been implemented, integrated and successfully tested, while at the current stage the preliminary ontology that was used for testing is being further populated in order to allow for application in less controlled document collections.

Future systems are also planned to use the complete framework presented herein, as for example the SAIL system which aims to automatically process educational document bases in order to locate documents related to a learner's

request; in that context automated document analysis is essential as the fact that the users are not field experts who can properly formulate queries describing their needs is at the very heart of the problem addressed.

7. Conclusions

This work has been based to some extent on previous works, both by the author and others. What is new in this paper is that for the first time distinct approaches and methodologies have been brought together in order to provide a complete and integrated soft computing framework for flexible querying; integrated in the sense that all are founded on a common fuzzy ontological basis, which allows for the same definitions, methodologies and algorithms to be reused throughout the retrieval process.

New elements incorporated in the presented framework include:

- the incorporation of the previously proposed fuzzy relational knowledge representation model in ontologies via reification,
- the notion of knowledge views,
- the context aware methodology for query disambiguation and document indexing,
- the notion of inter-preference noise and a user profile model that overcomes it,
- the algorithm to contextualize user profiles,
- the modified matching mechanism that considers user preferences, as well as
- the consideration of all steps of the retrieval process as individual applications of a common overall framework.

The result is a framework that allows for human knowledge to be incorporated into ontologies in a seamless way without the loss of descriptive power and then to be exploited in order to detect context and use it to drive the various steps of information retrieval. As a result, more intuitive queries may be supported and more meaningful results may be provided to users.

As far as future work is concerned, although the algorithmic and modeling aspects of the presented framework have all been tested via application, much room for further improvement still exists. A first concern is how well all presented components work together in a practical setting, as the concurrent consideration of an abundance of different contexts (the context of the query, the context of the document and the context of the user) needs to be investigated practically. Also, the utilization of different fuzzy norms and co-norms in the transitive closure of knowledge views, as well as in the different information processing methods of the framework, needs to be examined as different norms may have lead to more semantically relevant results.

Finally, this discussion would not be complete if we did not underline that every aspect of the presented framework is based on the existence of a proper ontology with detailed and meaningful fuzzy relations defined between the various semantic entities. The development of a detailed domain ontology is a very painful and time consuming process; the development of a detailed domain ontology including semantically aligned degrees for each included relation is obviously a much more difficult task, and therefore the extent to which we are able to develop such relations provides the practical boundaries of the applicability of the presented framework.

Also, the careful reader may have noticed that querying in the proposed framework is performed solely through the considered ontology, which implies that terms and words not included in the ontology (proper names are good candidates for this) cannot be retrieved. This is clearly a limitation on the application of the presented approach to conceptual querying, intended to be utilized as a complement to and not instead of other general purpose information retrieval techniques. The integration of the two approaches into one methodology inheriting all merits and overcoming all drawbacks is also a part of our future work.

References

Adelson-Velskii, G.M. and Landis, E.M. (1962) An algorithm for the organization of information. *Doklady Akademii Nauk SSSR* **146**, 263-266, 1962; English translation in *Soviet Math*, **3**, 1259-1263.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999) *Modern Information Retrieval*. Addison Wesley.

Baker, J.J. (1962) A note on multiplying Boolean matrices. *Comm. ACM* **5** (2), 102.

Baziz, M., Boughanem, M. and Aussenac-Gilles, N. (2005) A Conceptual Indexing Approach based on Document Content Representation. *CoLIS5: Fifth International Conference on Conceptions of Libraries and Information Science*, Glasgow, UK, 171-186.

Baziz, M., Boughanem, M., Loiseau, Y. and Prade, H. (2007) *Fuzzy Logic and Ontology-based Information Retrieval*. Springer Berlin-Heidelberg.

Berners-Lee, T. and Fischetti, M. (1999) *Weaving the Web: Origins and Future of the World Wide Web*. Orion Business.

Bordogna, G. and Pasi, G. (2002) Flexible querying of WEB documents. *Proceedings of the 2002 ACM Symposium on Applied computing*. ACM Press, 675-680.

Buche, P., Dibie-Barthelemy, J., Haemmerle, O. and Hignette, G. (2006) Fuzzy semantic tagging and flexible querying of XML documents extracted from the Web. *Journal of Intelligent Information Systems* **26** (1), 25-40.

Bulskov, H., Knappe, R. and Andreasen, T. (2002) On Measuring Simi-

larity for Conceptual Querying. *Proceedings of the 5th International Conference on Flexible Query Answering Systems*, 100-111.

Calegari, S. and Sanchez, E. (2007) A Fuzzy Ontology-Approach to improve Semantic Information Retrieval. In: F. Bobillo et al., eds., *Proceedings of the Third ISWC Workshop on Uncertainty Reasoning for the Semantic Web*, online at http://ceur-ws.org/Vol-327/pos_paper3.pdf.

Callens, B., de Tre, G., Verstraete, J. and Hallez, A. (2003) A flexible Querying framework (FQF): Some implementation issues. *International symposium on computer and information sciences*. Springer, Berlin-Heidelberg, 260-267.

Chen, S.-M., Horng, Y.-J. and Lee, C.-H. (2003) Fuzzy information retrieval based on multi-relationship fuzzy concept networks. *Fuzzy Sets & Systems* **140**, 183-205.

de Calmes, M., Prade, H. and Sedes, F. (2007) Flexible querying of semi-structured data: A fuzzy-set-based approach. *International Journal of Intelligent Systems* **22** (7), 723-737.

Dubois, D. and Prade, H. (2008) An introduction to bipolar representations of information and preference. *International Journal of Intelligent Systems* **23**(8), 66-877.

Dubois, D. and Prade, H. (1997) *Using Fuzzy Sets in Flexible Querying: Why and How? Flexible Query Answering Systems*. Kluwer Academic Publishers.

Dunn, J.C. (1974) Some recent investigations of a new fuzzy partitioning algorithm and its applications to pattern classification problems. *Journal of Cybernetics* **4**, 1-15.

Kraft, D.H., Petry, F.E. (1997) Fuzzy information systems: managing uncertainty in databases and information retrieval systems. *Fuzzy Sets and Systems* **90**, 183-191.

Lee, H.-S. (2001) An optimal algorithm for computing the maxmin transitive closure of a fuzzy similarity matrix. *Fuzzy Sets & Systems* **123**, 129-136.

Matthe, T., de Tre, G., Hallez, A., de Caluwe, R., Leman, M., Cornelis, O., Moelants, D. and Gansemans, J. (2005) A framework for flexible querying and mining of musical audio archives. *Proceedings of the Sixteenth International Workshop on Database and Expert Systems Applications*, 1041-1045.

De Meyer, H., Naessens, H. and De Baets, B. (2004) Algorithms for computing the min-transitive closure and associated partition tree of a symmetric fuzzy relation. *EJOR* **155**, 226-238.

Miyamoto, S. (1990) *Fuzzy Sets in Information Retrieval and Cluster Analysis*. Kluwer Academic Publishers: Dordrecht-Boston-London.

Mylonas, Ph. and Wallace, M. (2006) Using ontologies and fuzzy relations in the multimedia personalization process. *1st International Workshop on Semantic Media Adaptation and Personalization (SMAP 2006)*, Athens, Greece, 4-5 December. IEEE Computer Society.

Rocacher, D. and Bosc, P. (2005) The set of fuzzy rational numbers and flexible querying. *Fuzzy Sets and Systems* **155** (3), 317-339.

Rocchio, J.J. Jr. (1971) Relevance Feedback in Information Retrieval. *The SMART System - Experiments in Automatic Document Processing.* Prentice Hall, 337-354.

Salton, G. (1971) Relevance Feedback and the optimization of retrieval effectiveness. *The SMART System - Experiments in Automatic Document Processing.* Prentice Hall, 324-336.

Sanchez, E. and Yamanoi, T. (2006) Fuzzy Ontologies for the Semantic Web. **LNCS 4027**, Springer Verlag, 691-699.

Seidel, R. (1995) On the All-Pairs-Shortest-Path problem in unweighted undirected graphs. *J. Computer & System Sciences* **51**, 400-403.

Theodoridis, S. and Koutroumbas, K. (1998) *Pattern Recognition*, Academic Press.

W3C, RDF Reification.
http://www.w3.org/TR/rdf-schema/#ch_reificationvocab.

Wallace, M. (no date) The implementation of the framework in Java™.
http://image.ntua.gr/∼wallace/java/transitive/.

Wallace, M., Athanasiadis, Th., Avrithis, Y., Delopoulos, A. and Kollias S. (2006) Integrating Multimedia Archives: The Architecture and the Content Layer. *IEEE Trans. Systems, Man, and Cybernetics*, Part A: Systems and Humans **36** (1), 34-52.

Wallace, M., Avrithis, Y. and Kollias, S. (2006) Computationally efficient sup-t transitive closure for sparse fuzzy binary relations. *Fuzzy Sets and Systems* **157** (3), 341-372.

Warren, H.S. (1975) A modification of Warshall's algorithm for the transitive closure of binary relations. *Comm. ACM* **18** (4), 218-220.

Warshall, S. (1962) A theorem on Boolean matrices. *J. ACM* **9** (1), 11-12.

Zadrozny, S. and Kacprzyk, J. (2006) Bipolar Queries and Queries with Preferences. *Proceedings of the 17th International Conference on Database and Expert Systems Applications*, 415-419.