Fuzzy Logic Techniques in Multimedia Database Querying: A Preliminary Investigation of the Potentials

Didier Dubois, *Member*, *IEEE*,
Henri Prade, and Florence Sèdes

Abstract—Fuzzy logic is known for providing a convenient tool for interfacing linguistic categories with numerical data and for expressing user's preference in a gradual and qualitative way. Fuzzy set methods have been already applied to the representation of flexible queries and to the modeling of uncertain pieces of information in databases systems, as well as in information retrieval. This methodology seems to be even more promising in multimedia databases which have a complex structure and from which documents have to be retrieved and selected not only from their contents, but also from "the idea" the user has of their appearance, through queries specified in terms of user's criteria. This paper provides a preliminary investigation of the potential applications of fuzzy logic in multimedia databases. The problem of comparing semistructured documents is first discussed. Querying issues are then more particularly emphasized. We distinguish two types of request, namely, those which can be handled within some extended version of an SQL-like language and those for which one has to elicit user's preference through examples.

Index Terms—Fuzzy logic, multimedia databases, querying by example, semistructured documents, inexact matching.

1 Introduction

WITH the advent of the multimedia age, new functionalities and capabilities are needed for managing new kinds of data: images, sounds, texts, video data, and their combination into composite objects that are generally called "multimedia documents." The spatial and/or temporal, retrieval, browsing, integration, and presentation requirements of multimedia data significantly differ from those for traditional data [57]. A multimedia DBMS [4] must indeed address many requirements beside traditional DBMS capabilities, such as huge capacity storage management, interface and interactivity, performance and QoS (Quality of Service).

The full use of such information systems raises new issues, especially for access and manipulation of information in a more intuitive, less formalized and human-friendlier way. It poses new challenges from data indexing to querying and retrieval; due to the richness of multimedia data content, querying systems must have extended capabilities. Modeling must provide, e.g., through the use of pattern recognition, indexing or classifying tools, high-level descriptions or abstractions of multimedia data content, structure, and presentation. A query support must allow the user to query them by the idea he has of their appearance rather than by their exact content. Querying by example or allowing for flexible queries is natural in this context.

It is well-known that fuzzy logic [71], [72], [73] provides a framework for modeling flexibility and vagueness in the interface between human conceptual categories and data. Such capabilities have already been developed in the database field, especially for handling flexible queries [43], [13]. There have been only a few preliminary and specialized papers on the use of fuzzy logic in multimedia databases systems [7], [21], [20], [74], [75]. In this paper, supposing the tools for indexing video and audio documents are available, we suggest that this methodology might be even more necessary and useful for multimedia applications. Some emerging applications are discussed, although no specific systems are surveyed.

The body of the text is organized into three main sections. First, Section 2 emphasizes the specificities of multimedia databases, especially w.r.t. indexing and querying and ends with a proposal for handling semistructured documents where some flexibility in the matching process is already introduced. Then, Section 3 offers an overview of the use of fuzzy sets techniques in regular database systems, which are also applicable in multimedia. Last, Section 4 suggests two new types of applications of fuzzy-set-based methods in relation with specificities of multimedia systems: the extensions of SQL-like query language and an approach to querying by examples. The concluding section emphasizes the potentials of flexible querying for multimedia systems and also points out some other possible developments in relation to them.

This paper is a fully revised version of an IFIP conference paper [34], previously extended with other recent related developments [35].

2 Multimedia Databases

We can view a multimedia database as a controlled collection of multimedia data items, such as texts, images, graphic objects, sketches, video, or audio sequences. The concept of document refers to any composite object combining elements characterized by different media. The multimedia DBMS should accommodate these special requirements by providing high-level abstractions in order to manage the different data types, along with a suitable interface for their presentation.

Some multimedia data types, such as video, audio, and animation sequences, have temporal requirements which have implications on their storage, manipulation, and presentation. Images, graphics, and video have spatial constraints in terms of their contents and objects in an image present some spatial mutual relationships.

Content-based access to multimedia information [76] refers, for instance, for a picture, to shape, color, texture, etc. This is why textual descriptions are usually associated to it. Additional attributes or features, called "metadata," can also help enrich these descriptions or abstract representations [38], [41].

Thus, representing multimedia information such as image or sound sequences poses the problem of video or audio understanding. Textual descriptions are commonly used to describe them in current systems which raise problems for information retrieval due to the limitations of these descriptions, their subjectivity to the representation norms and the massive information available according to the context, the interpretation, the user's profile, etc. The lack of a standard structure of potential information means that it can be difficult "straight away" to express a precise query.

On the whole, characteristic features of multimedia data are: lack of a standard structure, heterogeneity of formats, self-describing information, inadequacy of textual descriptions, multiplicity of data types and source databases, spatial and temporal characteristics.

2.1 Indexing

By contrast, a low-level representation of multimedia data encodes the physical reality. Automated indexing based on this type of representation uses features, such as color, shape, texture, spatial information, symbolic strings, for indexing images; see, for instance, [39]. For audio data [66], describing can involve an analysis of the signal or speech recognition

• *The authors are with the IRIT (Institut de Recherche en Informatique de Toulouse), 118, route de Narbonne, 31062 Toulouse cedex 4, France. E-mail: {dubois, prade, sedes}@irit.fr.*

Manuscript received April 1999; revised November 1999; accepted December 1999.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 111278.

followed by keyword-based indexing; other perceptual and acoustic features are used for musical data, such as note, pitch, duration, or rhythm signature, chord, and melody.

From the information retrieval area, we know how difficult it is to characterize the contents of textual objects. Problems are encountered on the one hand in specifying the contents of the objects and, on the other hand, for describing the objects one is looking for. In multimedia databases, the question is thus how to characterize the "contents" of image, audio, or video data. Indeed, we must face the problem of giving an interpretation to a photo or a song, for which many interpretations exist in general, according to the context, the user's point of view [5], etc.

Multimedia data must be preferably interpreted before they can be queried in order to generate content descriptions (otherwise, it might be more costly to make it online). Multimedia information can be retrieved using identifiers, attributes, or keywords. Keywords are by far the predominant method used for indexing multimedia data. These keywords are metadata since they are "data about (multimedia) data." A user is supposed to select keywords from a set of words belonging to a prescribed vocabulary (specialized or not) or thesaurus. This method is simple and intuitive, but depends on the subjectivity of the person who indexes or on the underlying hypotheses of an automatic indexing system and obviously depends on the given vocabulary. Thus, indexing is context-dependent. Introducing abstractions allows the user to refer to the data in terms of high-level features which constitute his model of the application domain. For the retrieval and organization of the multimedia data, it should be possible to provide several layers of abstractions according to the indexing levels and the interpretation.

However, it is difficult to completely automate indexing. While the computer can easily analyze a picture containing works of art in terms of colors, textures, and shapes [50], it is almost impossible to automatically determine interpretive or aesthetical features of an art object (e.g., is the red circle a rising sun on the landscape or the brim of a hat?).

2.2 Querying and Information Retrieval

Queries in relational systems are exact match queries: The system is able to return exactly those tuples a user is precisely asking for and nothing more. To specify the resulting relation, conditions concerning known attributes of known relations can be formulated.

In the context of this paper, roughly speaking, we can distinguish between queries aiming at retrieving one document by means of some distinctive pattern and topically oriented requests aiming at collecting a family of related documents. For instance, searching for data that contains specific audio samples or spoken parts, such as broadcast news about a specific issue specified by some terms. In addition to single media-based search, one may want to access data on the basis of a multiple media specification. In this case, a query usually involves different multimedia data types, various attributes, possibly keyword-based or content-oriented, or even contextual information. Retrieval algorithms must support content and context-based retrieval and multimedia DBMS should offer support for spatial and temporal queries [77].

Extensions of conventional concepts of query languages—e.g., all the retrieved objects exactly match the query—require that these characteristics be taken into account. It requires approaches that can deal with the temporal and spatial semantics of multimedia data or query languages that can incorporate flexibility in the expression of requests. Indeed, as queries are usually imprecise, relevance feedback and meaning similarity, rather than exact matching, and mechanisms for displaying ranked results, are important. This is particularly important in combination with "content-based" access, where the specifications often have to be considered as approximate and imprecise. In Section 4, we explore how multimedia querying should offer support for new requirements such as querying by examples and flexible querying of spatial or temporal data, using fuzzy predicates.

2.3 Looking for Similar Semistructured Documents

One of the main problems of documentary applications is that a query may refer to a document in terms of its appearance from the user's point of view and not only in terms of its attribute contents. The lack of a standardized structure of the document(s) to be retrieved calls for the use of flexibility in querying.

Self-describing data has been considered recently in the database research community. Researchers have found this data to be fundamentally different from relational or object-oriented data and called them *semistructured data.* Semistructured data [80], [81], [83] are motivated by the problems of integrating heterogeneous data sources, modeling sources such as Web data and structured text documents, e.g., SGML and XML. Research on semistructured data has addressed data models, query-language design [84], [85], query processing and optimization, schema languages (tutorials describing some of the works on semistructured data can be found in [1] and [82]); here, we focus on schema extraction and comparison.

The key observation is that most of the Web documents are semistructured data. Indeed, an important class of documents consists of semistructured documents, like HTML or XML marked-up documents [18], [23]. Here, the idea of allowing for flexibility in the exploitation of semistructured information means to refer to the structure in an approximate matching procedure.

Different types of queries can address a collection of semistructured documents.

A first type of query developed later in Section 4.1, is:

Find the paper including imperatively keywords "x..." and preferably keywords "y..." and which refers mostly to *good* papers in the domain.

The retrieving of relevant paper requires 1) the identification of keywords and references in the document structure, the document itself, or in any description and 2) the checking of if most of the authors belong to a set of "*good*" authors on the domain associated with the present keywords.

Another type of request may refer to the description of a document structure, e.g., "It had a title, author(s), probably an abstract, maybe keywords, a body structured in sections and subsections, certainly followed by references, among which we preferably may find some author's name, ...," to retrieve an already seen document where the description refers to its structure in a flexible way.

Thus, an important level of querying is to ask for "similar" documents, in terms of their structure, e.g., from a given sample structure. To cope with it, the problem of comparing a semistructured document is addressed in the following. This problem is challenging due to the irregularity, incompleteness, and lack of schema that characterizes semistructured data.

More generally, structural matching and discovery in documents are useful for data warehousing, version management, hypertext authoring, digital libraries, and web databases, for instance, to discover modifications in an HTML document or to find common substructures. Indeed, in addition to offering access to large collections of heterogeneous and semistructured data, the Web allows this information to be updated at any time. These rapid and often unpredictable changes create the problem of detecting these modifications. A user may visit documents repeatedly and is interested in knowing how each document has changed since the last visit. This may be achieved by saving a snapshot of the previous HTML pages at the site (something that

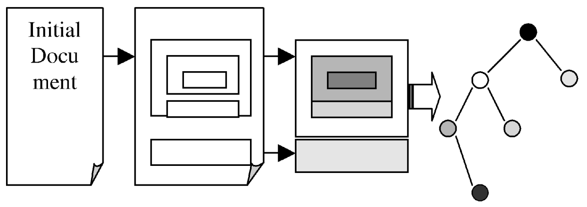

Fig. 1. Eliciting and extracting the document structure.

most browsers are able to do anyway). Assuming we have saved the old version of the document, we want to periodically detect the changes due to its evolutions by comparing the old and new versions of the document and knowing how much similar they are.

Most previous works in change management have dealt only with text files [54], [44], flat files or relational data [2], [47], presenting algorithms for efficiently comparing sets of records that have keys. A system, called *LaDiff*, has been implemented to detect, mark, and display changes in structured documents. It is based on analysis of their hierarchical structure, taking two versions of a LaTex document as input and producing as output a LaTex document with the changes marked [56]. Much database research has been conducted about structured documents, such as SGML [15], [16], [19], [17], and [52]: Given a Document Type Definition (DTD), any document should conform to it. This DTD is, like a database schema, a generic structure for a class of documents.

Few systems have been built to support semistructured data. Indeed, these data are irregular, incomplete, and do not necessarily conform to a fixed schema. Semistructured data may have some structure but, if it exists, it is enclosed into the instance and a priori unknown. It must be elicited in order to discover patterns. From a given mark-up language, a parser can analyze the document content in order to elicit the structure items. Documents are represented as ordered labeled trees since hierarchically structured information can be represented as trees in which the children of each node have a designated position (see Fig. 1).

We must identify changes not just to the nodes content, but also to their relationships. For example, if a node (and its children) is moved from one location to another or if a leaf node has been split up into its children, we would like to detect it, e.g., given a level l, the structure has been reorganized, directly attaching the children at level $l+1$ to the father at level $l-1$ (see Fig. 2).

Hence, one of our problems is to find an appropriate matching for the trees we are comparing. The matching is difficult because it is based on labels of marks and not only on the content of the text. Two documents can be compared using tree matching techniques [63], [70], [65]. Furthermore, not only do we want to match trees that are identical (with respect to the labels and values of the nodes and their children), but we also want to match trees that are "approximately equal," according to their similarity. Graph structure is not adapted to comparison other than exact identity. In the following, we suggest a process adapted to comparison algorithms [63], [70], by successive adaptation, for example,

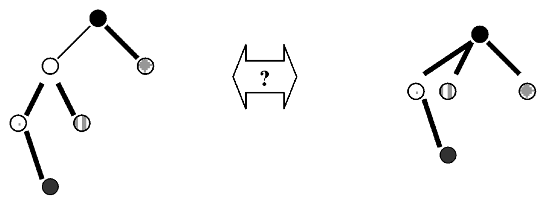

Fig. 2. Comparing two structures.

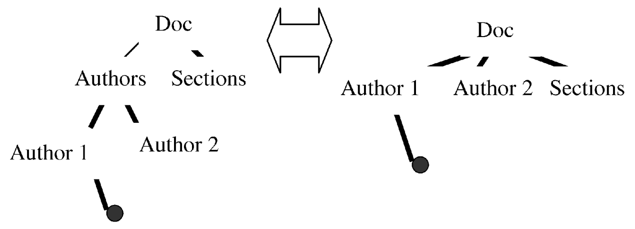

Fig. 3. Example of similar structures.

deleting a structure level if this one is considered as not very relevant (see Fig. 3). One of the main characteristics of this model is that it saves ordering between elements. Without it, the problem of "subtree" relations between two trees cannot be handled properly [45].

The possible cohabitation of multiple views of the same document handicaps any approach based on strict comparison. Additional semantic knowledge about elements (nodes) or relationships (arcs) is necessary to allow an approximate comparison. This knowledge can be extracted from the element name (mark-up content), from the attributes or their values, or from any external user's specification. Our approach to the problem of detecting modifications in pieces of hierarchically structured information has two main original features:

1. *Semistructured data.* Although some database systems, particularly active database systems, build change detection facilities into the system itself, we focus on the problem of detecting changes given old and new versions of the data. What can be known about the structure of a semistructured document should be extracted from the encoding of the document itself.
2. *Approximate matching.* By approximate matching of graphs, we mean that the graphs at least match for their essential parts and, if possible, for their less important subparts. The matching becomes a matter of degree.

A first version of the elicitation process of the structure (without any grading of the importance of each subpart of the document) has been implemented in the rewriting tool called eXrep [49]. The ordered tree-like structure of the document is built by recognizing structure items from the content itself through automatic identification of marks and their rewriting by means of dictionaries [62]. From this rewriting process, it is possible to identify, in the tree, edges which can be considered as more important than others: From the document analysis, a syntactic and semantic marking can allow us to infer that some edges have a lower weight than other ones. The ideas at the basis of the weights computation are that the level of importance of the edge depends on the recognized mark, its content, the text associated with the pending node, and the depth of the tree representing the structure. Indeed, comparison cannot only rely on structural similarities while text associated with the nodes is ignored. The weighting process first deals with structural comparison and mark-up semantics, second with content. In practice, one could only distinguish between a rather small number of levels in the importance scale.

Given two tree structures T^1 and T^2 whose nodes are weighted on a scale

$$\mathrm{w}_1 = 1 > \mathrm{w}_2 > \ldots > \mathrm{w}_\mathrm{n} > \mathrm{w}_{\mathrm{n}+1} = 0,$$

their similarity can be evaluated in the following way:

Let T_i be the tree built from T by only keeping edges whose level is greater or equal to w_i and fusing the nodes belonging to a suppressed edge. Then, $\mathrm{similarity}(\mathrm{T}^1, \mathrm{T}^2) = \mathrm{w}_{\mathrm{n}-\mathrm{i}+1}$ if T^1_i and T^2_i are identical but $\mathrm{T}^1_{\mathrm{i}+1}$ and $\mathrm{T}^2_{\mathrm{i}+1}$ are different and $\mathrm{similarity}(\mathrm{T}^1, \mathrm{T}^2) = 0$ if T^1_1 and T^2_1 are different.

So, $\mathrm{similarity}(\mathrm{T}^1, \mathrm{T}^2) = 1$ if T^1 and T^2 are similar at each level, from 1 to n (at each level, similarity is binary, equals 0 or 1).

Two structures of documents are all the more similar as there are fewer subparts to neglect to make them identical; this is why $\mathrm{similarity}(\mathrm{T}^1, \mathrm{T}^2)$ is computed by means of an order-reversing map of the scale $\{1, \ldots, \mathrm{n}\} : \mathrm{i} \rightarrow \mathrm{n} - \mathrm{i} + 1$.

What occurs with the content? When comparing T^1 with an approximation of T^2 where some edge has been deleted, we may have to detect whether the text associated with the suppressed node, if there is one, is pasted elsewhere in the approximation of T^2. More generally, we may think of computing the similarity of adding requirements on identity or subsumption of the attached texts (in order to have a nonzero similarity).

Thus, given a document, it is possible to retrieve and rank-order approximately similar documents.

3 Fuzzy Set-Based Methods in Databases

Research on "Fuzzy databases" (see [10], [58]) has been developed for about 20 years by a few small groups of scholars, with only marginal connections to the main trends of database research. If we except the more recent use of fuzzy techniques in data mining, these works have been mainly concentrating on three issues:

1. flexible querying in classical languages (e.g., [43], [13]) and in object-oriented languages [22];
2. handling of imprecise, uncertain, or fuzzy data (e.g., [59], [32]);
3. defining and using fuzzy dependencies (e.g., [61], [9]).

An introduction to these different issues may be found in a survey [14].

These tasks involve the three basic semantics that can be naturally attached to a fuzzy set [33], namely, preference, uncertainty, and similarity. Indeed, the flexibility of a query reflects the preferences of the end-user. Using a fuzzy set representation, the extent to which an object described in the database satisfies a request then becomes a matter of degree. Besides, the information to be stored in a database may be pervaded with imprecision and uncertainty. Then, ill-known attribute values can be represented by means of fuzzy sets viewed as possibility distributions. Moreover, a query may also allow for some similarity-based tolerance: Close values are often perceived as similar, interchangeable. Indeed, if, for instance, an attribute value v satisfies an elementary requirement, a value "close" to v should still somewhat satisfy the requirement. The idea of approximate equality, of similarity, also plays a key role in the modeling of fuzzy dependencies. However, this last research trend will not be reviewed in the following since this issue does not seem to be immediately relevant for multimedia databases.

An advantage of fuzzy set-based modeling is that it is mainly qualitative in nature. Indeed, in many cases, it is enough to use an ordinal scale for the membership degrees (e.g., a finite linearly scale). This also facilitates the elicitation of (context-dependent) membership functions for which it is enough, in practice, to identify the elements that totally belong and those which do not belong at all to the fuzzy set.

Fuzzy set-based techniques have also raised interest in information retrieval for a long time (see [46], [53], [6]). In these approaches, degrees of relevance can be attached to each pair (keyword, document). Besides, we may also think of fuzzy thesauri. But, then we have to distinguish between a statistical degree which reflects the co-occurrence of noninterchangeable keywords in the description of documents belonging to the same corpus and a degree of (approximate) synonymy between keywords (or a technical keyword used for indexing and a user word). The analysis of relevance might be further refined by distinguishing, for a given vocabulary, between keywords which more or less certainly pertain to the document and others which are relevant but somewhat optional [60].

3.1 Advantages of Flexible Querying

Fuzzy set membership functions [67] are convenient tools for modeling user's preference profiles. The large panoply of fuzzy set connectives can capture the different user attitudes concerning the way the different criteria present in his/her query compensate or not; see [12] for a unified presentation in the fuzzy set framework of the existing proposals for handling flexible queries.

Thus, the interest of fuzzy queries for a user is twofold:

1. A better representation of users' preferences. For instance, "the user is looking for an apartment which is not too expensive and not too far from downtown." In such a case, there does not exist a definite threshold for which the price becomes suddenly too high. Rather, we have to discriminate between prices which are perfectly acceptable for the user and other prices, somewhat higher, which are still more or less acceptable (especially if the apartment is close to downtown). Obviously, the meaning of vague predicate expressions like "not too expensive" is context/user dependent, rather than universal. The large panoply of fuzzy set connectives can capture the different user's attitude concerning the way the different criteria present in his/her query compensate or not. Moreover, in a given query, some part of the request may be less important to fulfill; this leads to the need for weighted connectives. Elicitation procedures for membership functions and connectives are thus very important for practical applications.
2. Fuzzy queries, by expressing user's preferences, provide the necessary information for rank-ordering the answers contained in the database according to the degree to which they satisfy the query. It contributes to avoiding empty sets of answers when the queries are too restrictive, as well as large sets of answers without any ordering when queries are too permissive.

As we just said, flexible queries are often motivated by the expression of preferences or tolerance and of relative levels of importance. However, the use of queries involving fuzzily bounded categories may also be due to an interest for more robust evaluations. This is the case in a query like "find the average salary of the *young* people stored in the database," where the use of a predicate like "young" (whose meaning is clearly context-dependent) does not here refer to the expression of a preference; here, it is rather a matter of convenience since the user is not obliged to set the boundaries of the category of interest in a precise and, thus, rather arbitrary way. In such a case, a range of possible values for the average salary, instead of a precise number, will be returned to the user. This range can be viewed as bounded by the lower and the upper expected values of a fuzzy number; see [30]. It is a robust evaluation which provides the user with an idea of the variability of the evaluation according to the different possible meanings of "young" (in a given context).

Another important class of flexible requests oriented toward robustness are those involving linguistic quantifiers, such as "most," e.g., "do most of the international trains leave on time?" In such a query, "most" should be understood as a potential proviso for exceptions rather than as the approximate specification of a proportion to be checked [11].

3.2 Flexible Queries and Their Evaluation

1. *Enlarging a pattern by a tolerance relation*. Two values u_1 and u_2 belonging to the same attribute domain U may be considered as approximately equal even if they are not identical. For instance, if the pattern requires somebody who is 40 years old, an item corresponding to a person who is 39 may be considered in some cases as approximately matching the request. An approximate equality can be conveniently modeled by means of a fuzzy relation R which is reflexive (i.e., $\forall\, u \in U, \mu_R(u, u) = 1$) and symmetrical (i.e., $\forall\, u_1 \in U, \forall\, u_2 \in U, \mu_R(u_1, u_2) = \mu_R(u_2, u_1)$). The closer u_1 and u_2 are, the closer to 1 $\mu_R(u_1, u_2)$ must be. The quantity $\mu_R(u_1, u_2)$ can be viewed as a grade of approximate equality of u_1 with u_2. R is then called a proximity or a tolerance relation. When the query pattern is represented by a subset P of U (P may be fuzzy), but the retrieved item is a (precise) constant d, the tolerance R can be taken into account in the degree of matching by replacing P by the enlarged subset P o R, defined by

$$\mu_{P\,o\,R}(d) = \sup_{u \in U} \min(\mu_P(u), \mu_R(u, d)) \geq \mu_P(d). \qquad (1)$$

   Note that, when P is fuzzy, $\mu_P(d) = 1$ still means total compatibility with P and $\mu_P(d) = 0$ means total incompatibility with P. Intermediary degrees of matching account for partial compatibility.

2. *Prioritized combination of matching degrees*. In the case of *logical* conjunctive combination of several requirements P_i, which corresponds to an egalitarian view between the different requirements, the elementary degrees of matching are aggregated by the min operation, which may be prioritized for taking into account the importance of each requirement (min is the largest associative aggregation operation which extends ordinary conjunction; it is also the only idempotent one). Thus, for a piece of information $d = (d_1, \ldots, d_n)$, we obtain the global matching degree

$$\min_{i=1,\ldots,n} \max(1 - w_i, \mu_{P_i}(d)), \qquad (2)$$

   with $\mu_{P_i}(d) = \mu_{P_i}(d_i)$, where u_i is the precise value of the item d for the attribute pertaining to P_i and where the following condition should be satisfied by the priorities w_i:

$$\max_{i=1,\ldots n} w_i = 1, \qquad (3)$$

   if there is at least one requirement that is imperative and, thus, can eliminate an item d when it is violated. Clearly, when $w_i = 0$, the degree of matching $\mu_{P_i}(d)$ is ignored in the combination, then P_i has absolutely no importance; the larger w_i, the smaller the degrees of matching concerning P_i which are effectively taken into account in the aggregation. The normalization (3) expresses that the most important requirement has the maximal priority (i.e., 1) and is compulsory.

   In the above model, each priority level is a constant and, thus, does not depend upon the value taken by the concerned attribute for the considered object d. This limitation may create some unnatural behavior of the matching procedure. For instance, the price of an object you are looking for may be of limited importance only within a certain range of values; when this price becomes very high, this criterion alone should cause the rejection of the considered object, in spite of its rather low priority. To cope with this limitation, it has been proposed [36] that the priority level becomes a function of the concerned attribute value.

   Recently, some authors [37] have advocated the use of another weighted aggregation mode, namely,

$$\sum_i (w_i - w_{i+1}) * \ i \ * f(x_1, \ldots, x_i),$$

   where $\sum_i w_i = 1$ and $w_1 \geq \ldots \geq w_i \ldots \geq w_n$ and f is an aggregation function. The advantage of this scheme is its generality; moreover, it enjoys a restricted linearity property with regard to the weights w_i which makes it similar to a Choquet integral (see, e.g., [40]). However, this framework requires a *numerical* scaling, while (2) requires an *ordinal*, linearly ordered scale only ($1 - (\cdot)$ being the order-reversing map of the scale). Indeed, since the numerical meaningfulness of degrees of relevance of documents or of degrees of importance of keywords is debatable, qualitative aggregation schemes are sometimes more desirable.

   We may think that, in some cases, the min-based aggregation leads to a ranking of retrieved items that is insufficiently discriminating because it relies on the comparison of the least satisfied properties. This ranking can be greatly improved by the use of refinements of the min ordering [26].

3. *Hierarchical requirements.* This framework also allows for conditional requirements. A conditional requirement is a constraint that applies only if another one is satisfied. This notion will be interpreted as follows: A requirement P_j conditioned by a hard requirement P_i is imperative if P_i is satisfied and can be dropped otherwise. More generally, the level of satisfaction $\mu_{P_i}(d)$ of a fuzzy conditioning requirement P_i for an instance d is viewed as the level of priority of the conditioned requirement P_j, i.e., the greater the level of satisfaction of P_i, the greater the priority of P_j is. A conditional constraint is then naturally represented by a fuzzy set $P_i \rightarrow P_j$ such that:

$$\mu_{P_i \rightarrow P_j}(d) = \max(\mu_{P_j}(d), 1 - \mu_{P_i}(d)), \qquad (4)$$

   where P_j is a prioritized constraint with a variable priority.

   Nested requirements with preferences, of the form "P_1 should be satisfied and, among the solutions to P_1 (if any), the ones satisfying P_2 are preferred and, among those satisfying both P_1 and P_2, those satisfying P_3 are preferred, and so on," where $P_1, P_2, P_3 \ldots$ are hard constraints [48], can be understood in the following way: Satisfying P_2 if P_1 is not satisfied is of no interest; satisfying P_3 if P_2 is not satisfied is of no use even if P_1 is satisfied. Thus, there is a hierarchy between the constraints. One has to express that P_1 should hold (with priority 1) and that if P_1 holds, P_2 holds with priority α_2 and if P_1 and P_2 hold, P_3 holds with priority α_3 (with $\alpha_3 < \alpha_2 < 1$). Thus, this nested conditional requirement can be represented by means of the fuzzy set P^*:

$$\begin{aligned}\mu_{P^*}(d) = {} & \min(\mu_{P_1}(d), \max(\mu_{P_2}(d), 1 - \min(\mu_{P_1}(d), \alpha_2)), \\ & \max(\mu_{P_3}(d), 1 - \min(\mu_{P_1}(d), \mu_{P_2}(d), \alpha_3)).\end{aligned} \qquad (5)$$

   Another type of sophisticated flexible queries which can be handled in the fuzzy set framework are those which call for an extended division, e.g., "find the items which satisfy at a sufficient degree all the important requirements" [8], [27].

4. *Logical and compensatory ANDs.* Queries are usually compound and this raises the issue of finding the appropriate aggregation operation for combining the elementary degrees of matching. Even if the combination is linguistically expressed by the conjunction AND, it may correspond to very different aggregation attitudes ranging from logical to compensatory ANDs. Logical ANDs are modeled by prioritized $\min$ operations as explained above. Many other (weighted) operations exist (for example, weighted averages correspond to an utilitarian view where compensation makes sense). Ordered Weighting Averaging operations are arithmetic means quantified using a fuzzy quantifier and range between minimum and maximum [79]. Choquet integrals are more general aggregation functions with the same range that can account for dependent attributes [78]. Procedures for the practical elicitation of the right AND operator can be based on the ranking by the user of a few prototypical examples presented to him in order to identify what kind of aggregation he implicitly used [28]. More generally, examples can be used for identifying an operator in a parameterized family.

3.3 Uncertain Data

Viewing tuples as lists of attribute values, fuzzy data are associated with lists of fuzzy sets. Such lists contain possibly ill-known attribute values pertaining to the description of objects. Namely, a component in a list refers to only one (ill-located) element of the scale or domain of the concerned attribute; the corresponding fuzzy set, which restricts the possible values of this attribute, is called a possibility distribution.

The basic dissymmetry of the pattern-data matching is preserved by this modeling convention. Indeed, a fuzzy pattern represents an ill-bounded class of more or less preferred objects, while a fuzzy item represents an ill-known object whose precise description is not available. Namely, let P and D be, respectively, a pattern atom and an item component pertaining to the same single-valued attribute, which are to be compared. P and D refer to the same scale U conveying their meanings. Let μ_{P} be the membership function associated to atom P and π_{D} be the possibility distribution attached to D. Both are mappings from U to [0, 1], but $\pi_{\mathrm{D}}(\mathrm{u})$ is the grade of possibility that u is the (unique) value of the attribute describing the object modeled by the item. D is a fuzzy set of *possible* values (only one of which is the genuine value of the ill-known attribute), while P is a fuzzy set of *more or less* compatible values. For instance, $\pi_{\mathrm{D}}(\mathrm{u}) = 1$ means that u is totally possible, while $\pi_{\mathrm{D}}(\mathrm{u}) = 0$ means that u is totally impossible as an attribute value of the object to which the item pertains. In the following, μ_{P} and π_{D} are always supposed to be normalized, i.e., there is always a value which is totally compatible with P and a value totally possible in the range D.

Two scalar measures are used in order to estimate the compatibility between a pattern atom P and its counterpart D in the item list, namely, a degree of possibility of matching $\Pi(\mathrm{P};\mathrm{D})$ and a degree of necessity of matching $\mathrm{N}(\mathrm{P};\mathrm{D})$ which are, respectively, defined by (see [69] and [28]):

$$\Pi(\mathrm{P};\mathrm{D}) = \sup_{\mathrm{u}\in\mathrm{U}} \min(\mu_{\mathrm{P}}(\mathrm{u}), \pi_{\mathrm{D}}(\mathrm{u})), \tag{6}$$

$$\mathrm{N}(\mathrm{P};\mathrm{D}) = \inf_{\mathrm{u}\in\mathrm{U}} \max(\mu_{\mathrm{P}}(\mathrm{u}), 1-\pi_{\mathrm{D}}(\mathrm{u})). \tag{7}$$

The limiting cases where $\Pi(\mathrm{P};\mathrm{D})$ and $\mathrm{N}(\mathrm{P};\mathrm{D})$ take values 0 and 1 are useful to study in order to lay bare the semantics of these indices. For any fuzzy set, F on U, let $\mathrm{F}^\circ = \{\mathrm{u} \in \mathrm{U} | \mu_{\mathrm{F}}(\mathrm{u}) = 1\}$ be the core of F and $\mathrm{s}(\mathrm{F}) = \{\mathrm{u} \in \mathrm{U}, \mu_{\mathrm{F}}(\mathrm{u}) > 0\}$ be its support. Then, it can be checked that

1. $\Pi(\mathrm{P};\mathrm{D}) = 0$ if and only if $\mathrm{s}(\mathrm{P}) \cap \mathrm{s}(\mathrm{D}) = \oslash$,
2. $\Pi(\mathrm{P};\mathrm{D}) = 1$ if and only if $\mathrm{P}^\circ \cap \mathrm{D}^\circ \neq \oslash$,
3. $\mathrm{N}(\mathrm{P};\mathrm{D}) = 1$ if and only if $\mathrm{s}(\mathrm{D}) \subseteq \mathrm{P}^\circ$,
4. $\mathrm{N}(\mathrm{P};\mathrm{D}) > 0$ if and only if $\mathrm{D}^\circ \subset \mathrm{s}(\mathrm{P})$ (strict inclusion).

Note that $\Pi(\mathrm{P};\{\mathrm{d}\}) = \mathrm{N}(\mathrm{P};\{\mathrm{d}\}) = \mu_{\mathrm{P}}(\mathrm{d})$ in the case of a precise attribute value ($\pi_{\mathrm{D}}(\mathrm{d}) = 1$ and $\pi_{\mathrm{D}}(\mathrm{u}) = 0$ if $\mathrm{u} \neq \mathrm{d}$).

Besides, fuzzy relations R on the real line can be used to grasp such usual notions as "much before," "closely after," etc., that can be applied to the comparison of (fuzzy or nonfuzzy) time-points and time-intervals. Thus, $\Pi(\mathrm{D}_1 \mathrm{~o~} \mathrm{R};\mathrm{D}_2)$ and $\mathrm{N}(\mathrm{D}_1 \mathrm{~o~} \mathrm{R};\mathrm{D}_2)$ evaluate, respectively, to what extent it is possible and certain that the fuzzy point D_1 be in relation R with the fuzzy point D_2. See [29] for an extension of Allen's temporal relations [3] to the case of fuzzy data and/or fuzzy relations.

4 Fuzzy Set Techniques in Multimedia Systems Querying

In the previous section, we provided a brief survey of fuzzy set techniques applicable to regular databases as well as to more general multimedia systems, working on abstract representations or metadata. As we already said, audio data, for example, can be segmented according to the speaker, indexed by signal analysis or speech recognition, followed by keyword-based indexing. So, the fuzzy set techniques referred to in the previous section can be applied in such multimedia data querying. Indeed, the handling of flexible queries may also require the evaluation of features at the signal level. For instance, in a remote image database, queries may refer to the "average surface area of the large parcels." For such a query, the first step is a matter of image processing: detecting boundaries of parcels, computing their surface. The second step consists of entering, via the interface, what the user means by "large" (1 or 10 ha?). The relative position of two objects in an image can be also computed at the pixel level using fuzzy set methods [51]. Indeed, the current information systems technology is widely available and provides the support on which fuzzy specifications can be based.

In this section, we further explore two types of querying which are of particular interest for multimedia systems: The next subsection deals with multimedia-oriented extensions of SQL/OQL, the second one considers queries implicitly specified through examples.

4.1 SQL/OQL-Like Queries

Making the most open form of querying possible addresses the need for a richer formulation in terms of predicates by integrating new operators into queries.

One of the main differences between a multimedia database and an ordinary one from a querying point of view is that, in the first case, the request may refer to a document in terms of its appearance, e.g., in terms of features to be looked for such as size, color, shape of pictures included in it, and not only in terms of attribute values already stored in a database. The lack of standardized structure of the document(s) to be retrieved calls for the use of flexible queries where traditional intervals and relaxation techniques do not work well. We are going to illustrate these different points by several examples.[1]

Q1. Retrieving an already seen document. The request refers to characteristic details in terms of appearance (spatial constraints).

1. In which the features pertaining to the uncertainty of the description are highlighted.

"Find **THE** paper published in the *early 90s* which deals with "x..." and "y..." and *maybe* also with "z..." with two pictures on the first page, of which the red one is *rather on the right*."

An extended OQL translation of this query could be [55]:

```
select q
from q in doc_lib
where q.creation_date in early_90()
/*on-line process, expression of a fuzzy set */
and q.description#'x....' and q.description#'y...'
and (maybe) q.description#'z...'
and count(meets_criteria(q.first_page(), picture))=2
and exists p1 in picture where meets_criteria(q,p1)
and meets_criteria(p1.histogram, «Red»)
and exists p2 in picture where meets_criteria(q,p2)
and meets_criteria(p1, (rather) right p2)
```

early_90() is a fuzzy predicate function which will be used for estimating the plausibility that a current document is the document we look for. *Maybe* will be understood in the following way: The overall relevance of a document which does not deal with "z..." will be discounted with respects to the ones which deal with "x..., " "y...," and "z..." (in other words, the latter documents are hierarchically preferred, see Section 3.2, (5)). The evaluation of the fuzzy predicate expression "*rather* on the right" exploits the HTML description of the structure of the document.

Q2. Retrieving an already seen document (temporal constraints).

"Find **THE** video sequence which comes after a scene in which there's a man waiting *close to* a tree during *about 30 seconds* and where *a little after* we hear the sound of an engine"

```
select q
from q in video_lib
where exists s where (
exists m where m.class#human and meets_criteria(s,m)
and exists o where o.class#…
and meets_criteria(m,(close_to) o)
and meets_criteria(s.length(), (about) 30 s))
and (exists sound where sound.class=engine
and meets-criteria(sound, (little_after) s))
and meets_criteria(q, after s)
```

about 30 s and *little_after* are temporal fuzzy predicates which can be evaluated from the story board.

The evaluation of *close_to* presupposes either a very precise indexing or an online evaluation of the image. If it is not possible (e.g., indexing only mentions a man and a tree without any distance information), the evaluation process should not reject the document even if it discounts it. This is a very general issue in such a problem where, due to the lack of known structure of the document, we cannot know if some feature is available or not. In such a case, the relevance of the current document will depend on the number of available features of the description (a document mentioning a man and a tree will be preferred to a document mentioning only a man or only a tree, etc).

Q3. Looking for a collection of never seen documents. Visual display.

"Find the documents dealing with "x..." in which *most* pictures are from the 90s."

```
select q
from q in doc_lib
where
q.description#'x....'
and exists p in picture where meets_criteria(q,p)
and (most) p.creation_date in 90()
/* extended division */
```

This supposes that the detailed description of the picture includes its creation date.

This is an example where *most* is a proviso for exceptions (a document will be all the more preferred as it includes fewer photos before 1990 and more photos of the 90s).

"Find the painting**S** with *warm* colors."

```
select q
from q in photo_lib
where
q.class#art_object
and q in warm_colors()
```

By contrast, with the previous example, here, *warm_colors* rather reflects user's preferences. *warm_colors* is a fuzzy set of colors whose definition may be context-dependent.

Auditory display.

"Find the piece**S** where cellos *dominate*."

```
select q
from q in audio_lib
where
q.class#music
and cello.duration%q.duration in dominate()
```

The evaluation of a document supposes that it is known when cellos play and when they don't.

4.2 Querying by Examples

The user is not always able to easily express his request, even in a flexible way. First, it may be more convenient for him to express what he is looking for by means of examples. Second, since he may have absolutely no a priori knowledge of the amount of retrievable documents (for a given request), it may be useful if the system is able to guide him about what exists by incrementally building queries from examples.

Querying based on examples, for eliciting user's preferences, can provide the necessary information for building a query. Thus, the user may say to what extent a few examples of documents are representative of what he is looking for by using some (finite) similarity scale. Then, relevance of current documents should be evaluated in terms of their similarity with respect to these examples and maybe counterexamples. Issues are then close to fuzzy case-based reasoning [24].

A first request, if it is too general, may retrieve too large a number of documents. Many techniques (fuzzy or not) exist for clustering such a set of documents. Then, these clusters have to be summarized in terms of prototypical elements (e.g., implicit keywords in an abstract) which are provided to the user. Once the user has chosen a cluster, he can refine his specification by means of a more accurate flexible query. This specification can be expressed in terms of weighted keywords (and in terms of similarities with other words). In such a case, the potential examples/counterexamples are provided by the system and the user has just to rate them.

Examples, as well as counterexamples, are supposed to be described in terms of precisely known attribute values. These attributes are also supposed to be relevant and sufficient for describing their main features from a user's point of view. Let a_{ij} with $i = 1, m$ and $j = 1, n$ be the value of attribute i for example j. Let b_{ik} with $k = 1, p$ be the value of attribute i for counterexample k. Let λ_j and ρ_k be the extent to which example j and counterexample k, respectively, are highly representative (from the user's point of view). Let us now consider a document d described by the attribute values $d_1, \ldots, d_m$, supposed to be precisely known (the attributes are supposed to be the same as the ones used for describing the examples and counterexamples). Clearly, the more *similar* d to *(at least)* a representative example and the more *dissimilar* to *all* counterexamples, the more *possible* d is eligible for the user.

This can be estimated by the following expression:

$$\mu(\mathrm{d}) = \min(\mathrm{ex}, \mathrm{cex}),$$

where

$$\mathrm{ex} = \max_\mathrm{j}\ \min(\lambda_\mathrm{j}, \mathrm{r_j})$$

is the extent to which there exists *at least one* highly representative example similar to d provided that $\mathrm{r_j}$ be the similarity between example j and d and

$$\mathrm{cex} = \min_\mathrm{k} \max(1 - \rho_\mathrm{k}, \mathrm{s_k})$$

is the extent to which *all* highly representative counterexamples are dissimilar to d provided that $\mathrm{s_k}$ is the dissimilarity between counterexample k and d.

Note that:

1. If all the examples are fully representative (i.e., $\forall_\mathrm{j}$, $\lambda_\mathrm{j} = 1$), then $\mathrm{ex} = \max_\mathrm{j}\ \mathrm{r_j}$ as expected; if $\lambda_\mathrm{j} = 0$, example j is not considered.
2. If all the counterexamples are fully representative (i.e., $\forall \mathrm{k}, \rho_\mathrm{k} = 1$), then $\mathrm{cex} = \min_\mathrm{k}\ \mathrm{s_k}$ reduces to a conjunctive aggregation; if $\rho_\mathrm{k} = 0$, counterexample k is not considered.

Then, let $\mathrm{S_i}$ be a fuzzy similarity relation on the attribute domain of i with membership function μ_Si ($\mathrm{S_i}$ is supposed to be reflexive and symmetrical) and $\mathrm{w_i}$ be the level of importance of attribute i in a description. Then:

- The similarity between d and example j is computed as

  $$\mathrm{r_j} = \min_\mathrm{i} \max(1 - \mathrm{w_i}, \mu_\mathrm{Si}(\mathrm{d_i}, \mathrm{a_{ij}}))$$

  (examples j and d are similar as far as all the highly important attribute values which describe them are similar);
- The dissimilarity between d and counterexample k is computed as

  $$\mathrm{s_k} = \max_\mathrm{i} \min(\mathrm{w_i}, 1 - \mu_\mathrm{Si}(\mathrm{d_i}, \mathrm{b_{ik}}))$$

  (counterexamples k and d are dissimilar as far as there exists a highly important attribute for which their respective values are very dissimilar).

When $\mathrm{w_i} = 1\ \forall i$, $\mathrm{r_j}$ is just the conjunctive aggregation of the similarity degrees and $\mathrm{s_k}$ the disjunctive combination of dissimilarity degrees.

Finally, we get

$$\mu(\mathrm{d}) = \min[\max_\mathrm{j} \min(\lambda_\mathrm{j}, \min_\mathrm{i} \max(1 - \mathrm{w_i}, \mu_\mathrm{Si}(\mathrm{d_i}, \mathrm{a_{ij}})), \min_\mathrm{k} \max(1 - \rho_\mathrm{k}, \max_\mathrm{i} \min(\mathrm{w_i}, 1 - \mu_\mathrm{Si}(\mathrm{d_i}, \mathrm{b_{ik}}))].$$

This evaluation might be improved by requiring the similarity of d with *most* examples (see, e.g., [36] for the introduction of a soft quantifier in a weighted min expression).

It is worth pointing out that the above expression can be used in another way that we now briefly outline and that is a topic for further research, apart from the ranking of documents w.r.t. a query. This expression also provides the starting point for generating a description of the documents that are looked for. This description may then be used for interface needs with the user. Indeed, letting d unspecified in the expression, it defines a compound fuzzy set expression which can be logically analyzed. Suppose, for simplicity, that all the weights are equal to 1, we obtain:

$$\mu(.) = \min[\max_\mathrm{j}\ \mathrm{mini}\ \mu_\mathrm{Si}(., \mathrm{a_{ij}}), \min_\mathrm{k}\ \max_\mathrm{i} 1 - \mu_\mathrm{Si}(., \mathrm{b_{ik}})].$$

Clearly, $\mu_\mathrm{Si}(., \mathrm{a_{ij}})$ defines a fuzzy set of values close to $\mathrm{a_{ij}}$, while $1 - \mu_\mathrm{Si}(., \mathrm{b_{ik}})$ defines a fuzzy set of values significantly different from $\mathrm{b_{ik}}$, for each attribute i. Note that the above expression may incorporate interactivity [68] between attributes. For instance, we have two examples of documents on a given topic, one of which is "short" without illustrations and another of which is "long" but having many illustrations. Such a set of examples suggests that acceptable documents may be "long" only if they have "many illustrations" in them. In this case, there is an interaction between the length of the document and the number of illustrations. This interaction may be embodied in the above type of expressions; however, it will make its reading less simple.

It should be pointed out that the set of examples and counterexamples provided by the user, enlarged by the similarity relations, does not usually cover all the cases which can be encountered. Namely, a document in the database may just be dissimilar to all the counterexamples without being similar to any examples. This suggests that if all the stored documents are in this situation, we may only use the cex part of the evaluation in order to avoid an empty answer to a request. However, the set of answers provided on the basis of cex only may be too large if it is not properly focused on the context of interest. For instance, looking for documents on a given topic having some length and number of illustration characteristics, cex should only apply to the latter characteristics in order not to retrieve all the documents on a different topic!

Another method for not "forgetting" the documents which may be in between the examples and the counterexamples would be rather to allow for requests of the form "all the documents satisfying some properties are relevant except the ones similar to counterexamples" or of the form "only the documents somewhat similar to at least one of the examples are relevant." Mixed requests could specify that the documents similar to example(s) are welcome and that those similar to counterexamples should be excluded, while the remaining documents (in the context of interest) have to be provided to the user only if the search is enlarged (then it may help the user in providing further examples and counterexamples).

5 Conclusion

The intended purpose of this paper was to provide a preliminary investigation of potential applications of fuzzy logic techniques in multimedia databases. Emphasis has been put on querying issues: detecting similar semistructured documents, intending to extend SQL/OQL-like queries, and querying by examples. The different uses of fuzzy logic techniques in relation to these two querying problems have been discussed. Last, another use of fuzzy logic techniques in relation with multimedia systems is worth mentioning.

Multimedia document editing environments mainly aim at enriching the representation of spatial placement and temporal layout (e.g., "the two sequences at the same time" in video). Modeling composition between objects and their synchronization (intra/inter component) should account for temporal relationships and spatial ones. One of the problems of multimedia document modeling is the specification of relationships and constraints such as "this sequence finishes *more or less* at the same time as this other one." The display of multimedia objects must be coordinated so that the display meets prespecified and dynamic temporal and spatial constraints. For instance, Madeus [42] is a multimedia authoring and presentation tool that takes into consideration the four dimensions of multimedia documents: logical, spatial, temporal, and hyperlinking. The temporal organization of the document is called scenario. The temporal constraint networks formalism has been chosen to represent the set of temporal relations used to relate objects. Its advantage is being easily interfaced with a declarative symbolic representation based on a set of quantified Allen operators [3] together with a set of causal operators. In such a context, it may be useful to express flexibility in the adjustment constraints (e.g., beginning *nearly* at the same time) as well as in spatial ones (this object should *be rather on the right* of the other one). Fuzzy constraints provide a way of expressing preferences between more or less admissible solutions. For that purpose, tools developed in flexible constraint satisfaction problems [25] are relevant.

The application of these tools to semistructured documents could be extended to hypermedia. Our work could be adapted to the detection of changes in data that can be represented as graphs but not necessarily trees. Concerning querying by examples, instead of counterexamples, we may think of hybrid queries made of examples and of classical (fuzzy) restrictions expressing what attribute values are undesirable. Some other applications clearly require advanced indexing or online content analysis of the documents, which may still lie partially beyond the current available technology.

Clearly, a key issue for the practical use of the ideas proposed above is the computational cost induced by the introduction of fuzzy features in the algorithms. Many fuzzy logic-based systems have been developed, implemented, and successfully used in practice in various areas. However, implementation experiences with fuzzy set methods in flexible querying database systems are still rather scarce, although the implementation of limited extensions of SQL-like languages appear to be feasible at a reasonable computational cost. This might apply as well to querying by examples where the structure of the expressions to be evaluated is somewhat similar.

Besides, recent works in flexible constraint satisfaction problems have shown that the complexity was a linear function of the number of levels used in the satisfaction scale (indeed, in practice, in many cases, a finite rather small number of levels is enough for refining all-or-nothing membership). In other cases, the use of a simple parameterized representation of fuzzy sets amounts to some direct computations on the parameters, which makes them easily tractable. Thus, the search for documents having approximately similar structures seems to resort only to the use of a scale with a small number of levels, which should not increase computational difficulties much.

References

[1] S. Abiteboul, "Semi-Structured Information," *Proc. Int'l Conf. Database Theory,* invited talk, pp. 1-20, 1997.

[2] S. Abiteboul, S. Cluet, and T. Milo, "A Database Interface for File Updates," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 18-26, 1995.

[3] J.F. Allen, "Maintaining Knowledge about Temporal Intervals," *Comm. ACM,* vol. 26, pp. 832-843, 1983.

[4] D.A. Adjeroh and K.C. Nwosu, "Multimedia Database Management Requirements and Issues," *IEEE Multimedia,* vol. 4, no. 3, pp. 24-33, 1997.

[5] *Multimedia Database in Perspective,* P. Apers, ed. Springer-Verlag, 1997.

[6] G. Bordogna, P. Carrara, and G. Pasi, "Fuzzy Approaches to Extend Boolean Information Retrieval," *Fuzziness in Database Management Systems,* P. Bosc and J. Kacprzyk, eds., pp. 231-274, 1995.

[7] P. Bosc, F. Connan, and D. Rocacher, "Flexible Querying in Multimedia Databases with an Object Query Language," *Proc. Seventh IEEE Int'l Conf. Fuzzy Systems,* pp. 1308-1313, 1998.

[8] P. Bosc, D. Dubois, O. Pivert, and H. Prade, "Flexible Queries in Relational Databases—The Example of the Division Operator," *Theoretical Computer Science,* vol. 171, pp. 281-302, 1997.

[9] P. Bosc, D. Dubois, and H. Prade, "Fuzzy Functional Dependencies and Redundancy Elimination," *J. Am. Soc. Information Systems,* vol. 49, no. 3, pp. 217-235, 1998.

[10] *Fuzziness in Database Management Systems,* P. Bosc and J. Kacprzyk, eds., Heidelberg: Physica-Verlag,, 1995.

[11] P. Bosc, L. Liétard, and H. Prade, "An Ordinal Approach to the Processing of Fuzzy Queries with Flexible Quantifiers. in: Applications of Uncertainty Formalisms," *Lecture Notes in Computer Science 1455,* A. Hunter and S. Parsons, eds., Berlin: Springer Verlag, pp. 58-75, 1998.

[12] P. Bosc and O. Pivert, "Some Approaches for Relational Databases Flexible Querying," *J. Intelligent Information Systems,* vol. 1, pp. 323-354, 1992.

[13] P. Bosc and O. Pivert, "SQLf: A Relational Database Language for Fuzzy Querying," *IEEE Trans. Fuzzy Systems,* vol. 3, no. 1, pp. 1-17, 1995.

[14] P. Bosc and H. Prade, "An Introduction to the Fuzzy Set and Possibility Theory-Based Treatment of Soft Queries and Uncertain or Imprecise Databases," *Uncertainty Management in Information Systems: From Needs to Solutions,* A. Motro and Ph. Smets, eds., chapter 10, pp. 285-324, Boston: Kluwer Acadamic, 1997.

[15] S. Chawathe and H. Garcia-Molina, "Meaningful Change Detection in Structured Data," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 22-32, 1997.

[16] S. Chawathe, A. Rajaraman, H. Garcia-Molina, and J. Widom, "Change Detection in Hierarchically Structured Information," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 34-43, 1996.

[17] C. Chrisment, P.Y. Lambolez, and F. Sèdes, "Hyperdocument Management and Exchange in an OO System," *Proc. Indo-French Workshop OO Systems,* pp. 313-333, 1994.

[18] C. Chrisment and F. Sèdes, "Bases d'Objets Documentaires," *Tutoriel, Informatique des Organisations et Systèmes d' Information et de Décision (INFORSID '98),* pp. 1-40, 1998.

[19] V. Christophides, S. Abiteboul, S. Cluet, and M. Scholl, "From Structured Documents to Novel Query Facilities," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 313-324, 1994.

[20] F. Connan, "Interrogation Flexible dans un Environnement Objet," *Rencontres Francophones sur la Logique Floue et Ses Applications,* Toulouse: Cépaduès, pp. 253-259, 1998.

[21] F. Connan and D. Rocacher, "Gradual and Flexible Allen Relations for Querying Video Data," *Proc. Fifth European Conf. Intelligent Techniques and Soft Computing (EUFIT '97),* pp. 1132-1136, 1997.

[22] R. De Caluwe, ed., *Fuzzy and Uncertain Object-Oriented Databases: Concepts and Models,* Singapore: World Scientific, 1997.

[23] S. Djennane and F. Sedes, *Audio Facilities for Hypermedia Consultation: Natural Language and Databases,* M. Bouzeghoub, eds., pp. 91-101, Amsterdam: IOS Press, 1996.

[24] D. Dubois, F. Esteva, P. Garcia, L. Godo, R. Lopez de Mantaras, and H. Prade, "Fuzzy Set Modelling in Case-Based Reasoning," *Int'l J. Intelligent Systems,* vol. 13, pp. 301-374, 1998.

[25] D. Dubois, H. Fargier, and H. Prade, "Possibility Theory in Constraint Satisfaction Problems: Handling Priority, Preference and Uncertainy," *Applied Intelligence,* vol. 6, pp. 287-309, 1996.

[26] D. Dubois, H. Fargier, and H. Prade, "Refinements of the Maximin Approach to Decision Making in Fuzzy Environments," *Fuzzy Sets and Systems,* vol. 81, pp. 103-122, 1996.

[27] D. Dubois, M. Nakata, and H. Prade, "Find the Items which Certainly Have (Most of) Important Characteristics to a Sufficient Degree," *Proc. Seventh IFSA World Conf.,* pp. 243-248, 1997.

[28] D. Dubois and H. Prade, *Possibility Theory—An Approach to Computerized Processing of Uncertainty.* New York: Plenum Press, 1988.

[29] D. Dubois and H. Prade, "Processing Fuzzy Temporal Knowledge," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 19, pp. 729-744, 1989.

[30] D. Dubois and H. Prade, "Measuring Properties of Fuzzy Sets: A General Technique and Its Use in Fuzzy Query Evaluation," *Fuzzy Sets and Systems,* vol. 38, pp. 137-152, 1990.

[31] D. Dubois and H. Prade, "Semantics of Quotient Operators in Fuzzy Relational Databases," *Fuzzy Sets and Systems,* vol. 78, pp. 89-93, 1996.

[32] D. Dubois and H. Prade, "Valid or Complete Information in Databases: Possibility Theory-Based Analysis," *Proc. Int'l Conf. and Workshop Database and Expert Systems Applications (DEXA '97),* pp. 603-612, 1997.

[33] D. Dubois and H. Prade, "The Three Semantics of Fuzzy Sets," *Fuzzy Sets and Systems,* vol. 90, pp. 141-150, 1997.

[34] D. Dubois, H. Prade, and F. Sedes, "Fuzzy Logic Techniques in Multimedia Databases Querying—A Preliminary Investigation of the Potentials, IFIP DS-8," *Semantics in Multimedia,* R. Meersman, Z. Tari, S. Stevens, eds., chapter 13, Boston: Kluwer Academic, 1999.

[35] D. Dubois, H. Prade, and F. Sedes, "The Use of Some Fuzzy Logic Techniques in Multimedia Databases Querying," *Proc. IEEE Int'l Conf. Fuzzy Systems,* pp. 586-591, 1999.

[36] D. Dubois, H. Prade, and C. Testemale, "Weighted Fuzzy Pattern Matching," *Fuzzy Sets and Systems,* vol. 28, pp. 313-331, 1988.

[37] R. Fagin and E. Wimmers, "Incorporating User Preferences in Multimedia Queries," *Proc. Sixth Int'l Conf. Database Theory (ICDT '97),* pp. 247-261, 1997.

[38] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, "Efficient and Effective Querying Image by Content," *J. Intell. Inf. Syst.* vol. 3, nos. 3/4, pp. 231-262, 1994.

[39] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker, "Query by Image and Video Content: The QBIC System," *Computer,* vol. 28, no. 9, pp. 23-32, Sept. 1995.

[40] M. Grabisch, H.T. Nguyen, and E.A. Walker, *Fundamentals of Uncertainty Calculi with Applications to Fuzzy Inference.* Dordrecht: Kluwer Academic, 1995.

[41] W.I. Grosky, "Managing Multimedia Information in Database Systems," *Comm. ACM,* vol. 40, no. 12, pp. 73-80, 1997.

[42] M. Jourdan, N. Layaïda, and L. Sabry-Ismail, "Time Representation and Management in MADEUS: An Authoring Environment for Multimedia Documents," *Multimedia Computing and Networking,* M. Freeman, P. Jardetzki, and H.M. Vin, eds., pp. 68-79, SPIE 3020, Feb. 1997.

[43] J. Kacprzyk and A. Ziolkowski, "Data Base Queries with Fuzzy Linguistic Quantifiers," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 16, no. 3, pp. 474-478, 1986.

[44] M. Kifer, *EDIFF— A Comprehensive Interface to Diff for Emacs 19,* available through anonymous ftp at ftp.cs.sunysb.edu, 1995.

[45] P. Kilpelaïnen, "Tree Matching Problems with Application to Structured Text Databases," technical report, Dept. of Computer Science, Univ. of Helsinki, Finland, 1992.

[46] D.H. Kraft and D.A. Buell, "Fuzzy Sets and Generalized Boolean Retrieval Systems," *Int'l J. Man-Machine Studies,* vol. 19, pp. 45-56, 1983.

[47] W. Labio and H. Garcia-Molina, "Efficient Algorithms to Compare Snapshots," available by anonymous ftp from db.stanford.edu in pub/labio/1995/, 1995.

[48] M. Lacroix and P. Lavency, "Preferences: Putting More Knowledge into Queries," *Proc. 13th Int'l Conf. Very Large DataBases,* pp. 217-225, 1987.

[49] P.Y. Lambolez, J.P. Queille, and C. Chrisment, "EXREP: A Generic Rewriting Tool for Textual Information Extraction," *Ingénierie des Systèmes d'Information,* vol. 3, no. 4, pp. 471-485, 1995.

[50] J. Martinez and S. Guillaume, "Colour Image Retrieval Fitted to "Classical" Querying," *Ingenierie des Systemes d'Information,* vol. 6, no. 1, pp. 1-12, 1998.

[51] P. Matsakis, "Relations Spatiales Structurelles et Interprétation d'Images," PhD thesis, Université Paul Sabatier, Toulouse 3, 1998.

[52] T. Milo and S. Zohar, "Using Schema Matching to Simplify Heterogeneous Data Translation," *Proc. Very Large Data Bases,* pp. 122-133, 1998.

[53] S. Miyamoto, *Fuzzy Sets in Information Retrieval and Cluster Analysis.* Kluwer Academic, 1990.

[54] E. Myers, "A Difference Algorithm and Its Variations," *Algorithmica,* vol. 1, no. 2, pp. 251-266, 1986.

[55] V. Ogle and M. Stonebraker, "Chabot: Retrieval from a Relational Database of Images," *Computer,* vol. 28, no. 9, pp. 40-48, Sept. 1995.

[56] Y. Papakonstantinou, H. Garcia-Molina, and J. Widom, "Object Exchange across Heterogeneous Information Sources," *Proc. 11th Int'l Conf. Data Eng.,* pp. 251-260, 1995.

[57] P. Pazandak and J. Srivasta, "Evaluating Object DBMS for Multimedia," *IEEE Multimedia,* vol. 4, no. 3, pp. 34-49, 1997.

[58] F.E. Petry, *Fuzzy Databases: Principles and Applications.* Dordrecht: Kluwer Academic, 1996.

[59] H. Prade and C. Testemale, "Generalizing Database Relational Algebra for the Treatment of Incomplete/Uncertain Information and Vague Queries," *Information Sciences,* vol. 34, pp. 115-143, 1984.

[60] H. Prade and C. Testemale, "Application of Possibility and Necessity Measures to Documentary Information Retrieval," *Uncertainty in Knowledge Based Systems, Lectures Notes in Computer Science 286,* B. Bouchon and R. Yager, eds., pp. 265-274, Berlin: Springer-Verlag, 1987.

[61] K.V.S.V.N. Raju and A.K. Majumdar, "Fuzzy Functional Dependencies and Lossless Join Decomposition of Fuzzy Relational Database Systems," *ACM Trans. Database Systems,* vol. 13, no. 2, pp. 129-166, 1988.

[62] F. Sedes, "Bases d'Objets Documentaires, Hyperbases," Habilitation à Diriger les Recherches, Université Paul Sabatier, Toulouse 3, 1998.

[63] D. Shasha and K. Zhang, "Fast Algorithms for the Unit Cost Editing Distance between Trees," *J. Algorithms,* vol. 11, no. 4, pp. 581-621, 1990.

[64] J. Wang, D. Shasha, G. Chang, V. Relih, K. Zhang, and G. Patel, "Structural Matching and Discovery in Document Databases," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* pp. 560-563, 1997.

[65] J.T.L. Wang, K. Zhang, K. Jeong, and D. Sasha, "A System for Approximate Tree Matching," *IEEE Trans. Knowledge and Data Eng.,* vol. 6, no. 4, pp. 559-571, Aug. 1994.

[66] W. Wold, T. Blum, D. Keislar, and J. Wheaton, "Content-Based Classification, Search, and Retrieval of Audio," *IEEE Multimedia,* vol. 3, no. 3, pp. 27-36, 1996.

[67] L.A. Zadeh, "Fuzzy Sets," *Information and Control,* vol. 8, pp. 338-353, 1965.

[68] L.A. Zadeh, "The Concept of a Linguistic Variable and Its Application to Approximate Reasoning," *Information Sciences,* pp. 219-366, 1975.

[69] L.A. Zadeh, "Fuzzy Sets as a Basis for a Theory of Possibility," *Fuzzy Sets and Systems,* vol. 1, pp. 3-28, 1978.

[70] K. Zhang and D. Shasha, "Simple Fast Algorithms for the Editing Distance between Trees and Related Problems," *SIAM J. Computing,* vol. 18, no. 6, pp. 1245-1262, 1989.

[71] D. Dubois, W. Ostasiewicz, and H. Prade, "Fuzzy Sets: History and Basic Notions," *Fundamentals of Fuzzy Sets: The Handbook of Fuzzy Sets Series,* D. Dubois and H. Prade, eds., pp. 21-124, Boston: Kluwer Academic, 1999.

[72] D.G. Schwartz, G.J. Klir, H.W. Lewis, and Y. Ezawa, "Applications of Fuzzy Sets and Approximate Reasoning," *Proc. IEEE,* vol. 82, no. 4, pp. 482-498, 1994.

[73] J. Yen, "Fuzzy Logic—A Modern Perspective," *IEEE Trans. Knowledge and Data Eng.,* vol. 11, no. 1, pp. 153-166, Jan./Feb. 1999.

[74] S. Medasani and R. Krishnapuram, "A Fuzzy Approach to Complex Linguistic Query Based Image Retrieval," *Proc. IEEE Int'l Conf. Fuzzy Systems,* pp. 590-595, 1999.

[75] M. Detyniecki, C. Seyrat, and R. Yager, "Interacting with Web Video Objects," *Proc. IEEE Int'l Conf. Fuzzy Systems,* pp. 914-918, 1999.

[76] A. Yoshitaka and T.A. Ichikawa, "A Survey on Content-Based Retrieval for Multimedia Databases," *IEEE Trans. Knowledge and Data Eng.,* vol. 11, no. 1, pp. 81-94, Jan./Feb. 1999.

[77] T. Lee, L. Sheng, T. Bozkaya, N.H. Balki, Z.M. Özsoyoglu, and G. Özsoyoglu, "Querying Multimedia Presentations Based on Content," *IEEE Trans. Knowledge and Data Eng.,* pp. 361-386, vol. 11, no. 3, May/June 1999.

[78] M. Grabisch, "The Application of Fuzzy Integrals in Multicriteria Decision Making," *European J. Operational Research,* vol. 89, pp. 445-456, 1996.

[79] R.R. Yager, "An Ordered Weighted Averaging Aggregate Operators in Multicriteria Decision Making," *IEEE Tran. Systems, Man, and Cybernetics,* vol. 18, pp. 183-190, 1998.

[80] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener, "The Lorel Query Language for Semistructured Data," *Int'l J. Digital Libraries,* vol. 1, no. 1, pp. 68-88, Apr. 1997.

[81] P. Buneman, S. Davidson, G. Hillebrand, and D. Suciu, "A Query Language and Optimization Techniques for Unstructured Data," *Proc. ACM-SIGMOD Int'l Conf. Management of Data,* 1996.

[82] P. Buneman, "Tutorial: Semistructured Data," *Proc. ACM SIGMOD Symp. Principles of Database Systems,* 1997.

[83] M. Fernandez, D. Florescu, A. Levy, and D. Suciu, "A Query Language for a Web-Site Management System," *SIGMOD Record,* vol. 26, no. 3, pp. 4-11, Sept. 1997.

[84] J. Le Maitre, E. Murisasco, and M. Rolbert, "SgmlQL, A Language for Querying SGML Documents," *Proc. Fourth European Conf. Information Systems (ECIS '96),* pp. 75-89, 1996.

[85] E. Bruno, J. Le Maitre, and E. Murisasco, "Controlled Hypertextual Navigation in the SgmlQL Language," *Proc. 10th Int'l Conf. and Workshop Database and Expert Systems Applications,* pp. 56-65, 1999.