Hierarchical Fuzzy Sets To Query Possibilistic Databases

Rallou Thomopoulos, Patrice Buche, Ollivier Haemmerlé

HIERARCHICAL FUZZY SETS TO QUERY POSSIBILISTIC DATABASES

Rallou Thomopoulos, Patrice Buche, Ollivier Haemmerlé

R. Thomopoulos is with INRA (Institut National de la Recherche Agronomique),
UMR1208 (bâtiment 31), 2 place P.Viala,
34060 Montpellier Cedex 1, France
and with LIRMM (CNRS & Université Montpellier II),
161 rue Ada, 34392 Montpellier Cedex 5, France
E-mail: rallou@ensam.inra.fr
Phone : (+33) 4 99 61 22 17 - Fax : (+33) 4 99 61 30 76

P. Buche is with INRA,
Department of Applied Mathematics and Computer Sciences, Unité Mét@risk
UR 1204, 16 rue Claude Bernard,
75231 PARIS Cedex 05, France.
Phone : (+33) 1 44 08 16 75 - Fax : (+33) 1 44 08 16 66
E-mail: patrice.buche@inapg.inra.fr

O. Haemmerlé is with IRIT,
Département de Mathématiques-Informatique,
UFR Sciences, Espaces et Sociétés, Université Toulouse le Mirail,
5 Allées Antonio Machado, 31058 Toulouse Cedex 1, France.
Phone : (+33) 5 61 50 46 08 – Fax : (+33) 5 61 50 41 73
E-mail: ollivier.haemmerle@univ-tlse2.fr

HIERARCHICAL FUZZY SETS TO QUERY POSSIBILISTIC DATABASES.

ABSTRACT

Within the framework of flexible querying of possibilistic databases, based on the fuzzy set theory, this chapter focuses on the case where the vocabulary used both in the querying language and in the data is hierarchically organized, which occurs in systems that use ontologies. We give an overview of previous works concerning two issues: firstly, flexible querying of imprecise data in the relational model; secondly, the introduction of fuzziness in hierarchies. Concerning the latter point, we develop an aspect where there is a lack of study in current literature: fuzzy sets whose definition domains are hierarchies. Hence we propose the concept of hierarchical fuzzy set and present its properties. We present its application in the MIEL flexible querying system, for the querying of two imprecise relational databases, including user interfaces and experimental results.

INTRODUCTION

In flexible querying systems, fuzzy sets are used to represent preferences in selection criteria. For instance, in the framework of a database about microbiological risk assessment in foods, the users may ask for milk as a first choice or yogurt as a second choice. In possibilistic databases, an imprecise datum is represented by a possibility distribution. For instance, in some kinds of human diseases, the bacterium Escherichia coli is suspected to be responsible, but other bacteria like Listeria are not excluded. Behind those two different purposes, the same homogeneous formalism is used: the fuzzy set theory. In both cases, a relation order is defined on a domain of values. In this chapter, we study the case when the domain of values is not "flat" but hierarchically organized, using the "kind of" relation. For instance, food products, like milk or yogurt, are part of a hierarchy of substrates, in which whole milk is a kind of milk. In the same way, the bacteria Escherichia coli and Shigella are part of a hierarchy of microorganisms. We call a fuzzy set defined on a hierarchy, a ***hierarchical fuzzy set (HFS)***. Contrary to the classical case when the domain of values is "flat", in this case, the assumption that the values are independent does not hold. Two order relations (the preference/possibility order relation and the "kind of" relation) must be put in adequacy. Several issues thus have to be addressed:

- Does the preference/possibility degree associated with a given value in a fuzzy set have implications on the degrees associated with other values of the domain, particularly more specific or more general values?
- What would be the meaning of two comparable values (with the meaning of the "kind of" relation) associated with different preference/possibility degrees?
- Can the "kind of" relation be used to enlarge the user's query in order to obtain more answers while respecting the preference order defined by the user in the selection criteria?

We have designed and realized two instances (for two different relational databases) of a flexible querying system, called MIEL[1], involving hierarchical fuzzy sets. Both databases contain imprecise data and deal with risk assessment in food, respectively microbial risk and chemical risk.

[1] MIEL is a french acronym for Extended Database Search Tool

The need for flexible querying, imprecise data representation and studying fuzzy sets when the domain of values is hierarchically organized is justified, in both databases, by three characteristics of the data:

- Although composed of several thousand entries (ten for the microbial database and fifty for the chemical database), data are not abundant enough to answer every query and therefore there is a need for flexible querying in order to complement exact answers with pertinent answers (i.e. semantically close).
- Data include imprecise values. For instance, the level of contamination of a given food by a given contaminant is not precisely known but is included in a given interval or is inferior to a given threshold.
- Symbolic data are often organized in taxonomies: for example, taxonomies of food products (Ireland & Moller, 2000), of bacteria (Balows et al., 1992), …

The MIEL fuzzy querying system has been especially designed for end-users who are not specialists of computer science. They express their query through a set of pre-written queries we call views. These views can be complemented by the users through the simple graphical user interface of the MIEL system. That interface allows the users to specify their projection attributes and their selection criteria. The taxonomies of the symbolic data can also be browsed by the end-users in order to express their selection criteria as hierarchical fuzzy sets.

In this chapter, firstly we provide some background on the topic and recall some broad definitions useful for understanding the main focus of the chapter. Secondly, we define and explain the concept of hierarchical fuzzy set and compare it to the bibliography. Thirdly, we present the MIEL flexible querying system which uses the concept of hierarchical fuzzy set. Fourthly, the instantiations of the MIEL system for the querying of two imprecise databases in the field of risk assessment in food are presented and we give some experimental results. Fifthly, current projects and future trends are presented. We conclude this chapter in the last section.

BACKGROUND

The bibliography is two-fold: we are concerned in this chapter with the combination of two topics, firstly, flexible querying of imprecise data, which includes flexible querying techniques, the representation of imprecise data, and the combination of both previous topics in the framework of the relational model; secondly, the introduction of fuzziness in hierarchies. We will finish this section by recalling the basics of fuzzy sets required to understand this chapter.

Flexible querying of imprecise data

Flexible querying techniques

The classical implicit assumption in database management systems is the Closed World Assumption: a fact which is not present in the database is assumed to be false. For example, let us consider the following facts stored in a given database: “Whole milk is contaminated by Listeria and Salmonella” and “Skim milk is contaminated by Escherichia Coli”. The query “Which are the contaminants not present in whole milk ?” will retrieve “Escherichia Coli”.

This assumption is embarrassing because, in real applications, it is often impossible to gather all the available information on a given subject. For example, in the field of risk assessment in food, it is difficult to gather information, in particular because of confidentiality problems. Consequently, it is important to be able to "relax" the Closed World Assumption in order to consider that the lack of answer does not mean that the answer is negative but is somewhat unknown: it corresponds to the Open World Assumption (OWA). It comes to consider that a database can be incomplete and that some queries may have an empty answer as a result. To avoid this drawback, one may propose to the user:

- Querying tools which retrieve information which is semantically close from the database,
- Models, parameterized with semantically close information found in the database, to estimate lacking information.

The first proposal, which is the one we consider in this chapter, has been studied in two different ways: by the expression of preferences in the selection criteria of a query and by the generalization of the selection criteria. Those mechanisms permit one to complement an exact answer, potentially empty, with semantically close answers which have been judged pertinent.

In the first family of approaches, the querying system does not check if information stored in the database verifies a selection criterion, but to which extent it somehow satisfies the selection criterion. It implies an order of the answers. Three kinds of works have been proposed to solve this problem: the use of secondary criteria in Lacroy & Lavency (1987), the definition of similarity distances (Ichikawa & Hirakawa, 1986; Motro, 1988), and the expression of linguistic preferences in Rabitti & Savino (1990). It has been shown (Bosc & Pivert, 1992; Bosc et al., 1994) that all those propositions can be restated in a unique formalism: the expression of preferences by fuzzy sets. This formalism permits the user to distinguish ideal values from acceptable values for a given criterion. A pertinence degree is associated with each answer corresponding to the query: it measures the adequation degree of the answer to the fuzzy selection criteria of the query.

In the second family of approaches, the query is modified to become more general (Motro, 1984). Consequently, the querying system retrieves the exact answers completed by other pertinent answers. In a first category of works, a hierarchy of concept is used to generalize the query when the answer is empty (Fargues, 1989; Bidault et al. 2000). In a second category of works, when the selection criterion is expressed by a fuzzy set, several techniques have been proposed to generalize it. Dubois & Prade (1995) propose to use a similarity relation defined on the domain of values. If the fuzzy set is defined on a numerical domain, Bosc et al. (2004) propose a fuzzy generalization operator using a proximity relation between two values based on the calculus of their quotient.

Directed by applications in the field of risk in food where information is structured according to hierarchical symbolic data, we propose to gather both families of approaches which are complementary. We will develop this idea in the concept of hierarchical fuzzy set presented in the main focus of the chapter.

Representation of imprecise data

In the context of database management systems, Codd (1979) has been one of the first to take into account the notion of imprecise datum in the framework of the relational model. He introduced the concept of *null value* representing the value of an attribute which is unknown

or has no sense in the record where it is stored. Lipski (Lipski 1979, Lipski 1981) has extended Codd's approach which was binary (complete knowledge or complete ignorance) in order to be able to express partial knowledge. He introduced the notion of plausible values represented by an exclusive disjunction of possible values. The theory of possibility (Zadeh 1978) has been used in the framework of the relational model by Prade (1984) and Prade & Testemale (1984) to extend Codd's and Lipski's approaches by introducing an order on the possible values. In our system, we propose a representation of imprecise data in the relational databases based on the theory of possibility, close to the representation used in FSQL (Galindo et al. 1998).

Fuzzy querying in the framework of the relational model

The expression of queries using fuzzy values has already been studied in the framework of the relational database model. Theoretical studies have been proposed to extend the SQL language by introducing fuzzy predicates processed on crisp information (Bosc & Pivert, 1995) and implementations have been proposed such as the FQUERY97 system (Zadrozny & Kacprzyk, 1998) under the QBE-like Microsoft Access graphical environment and the FSQL system (Galindo et al., 1998) under Oracle Relational Database Management System (RDBMS). Moreover, as the FSQL system permits the representation of imprecise data, its querying system is able to compare a fuzzy predicate with an imprecise datum. In those previous works, the user has to build himself the query flexibility: for instance, in FSQL, the user has to specify in his query whether he is using a fuzzy join or a standard one. Those systems are more or less dedicated to computer science specialists even if FSQL system, for example, interprets some fuzzy concepts as "approximate", "interval", "crisp" in a very understandable way. As we mentioned in the introduction, the aim of our system is to help any user to make a fuzzy query against a database schema. This is the reason why we have decided to develop our own fuzzy querying system, MIEL, which will be presented in the main focus of this chapter.

Introducing fuzziness in hierarchies

Introducing fuzziness in a hierarchy can be seen in different ways. In our concern, the issue is to be able to define an order relation – represented by degrees that express preferences or possibility – on a hierarchically organized set of elements, on which a relation order is thus already defined by the "kind of" relation. The aim is thus to properly define, and reason with, a fuzzy set whose definition domain is a hierarchy. This issue is not trivial since the degree associated with an element must be coherent with those associated with sub-elements or super-elements, and there is currently a lack on this subject in the literature.

In the bibliography concerning fuzzy methods, we have identified three main categories of papers which present some similarities; two are quite distant from our concern and the third one is closer to our concern and confirms some of the ideas we propose in this chapter. We can distinguish, especially in recent research:

- the use of linguistic labels in ontologies. In studies about possibilistic ontologies (Loiseau et al., 2005), each term of an ontology is considered as a linguistic label and has an associated fuzzy description. Fuzzy pattern matching between different ontologies is then computed using these fuzzy descriptions. This approach is related to those concerning the introduction of fuzzy attribute values in the object model (Rossazza et al., 98);

- the use of fuzzy relations between the terms of a thesaurus. Studies about fuzzy thesauri have discussed different natures of relations between concepts, where relations are gradual and moderated by degrees. Fuzzy thesauri have been considered for instance in Miyamoto (1986) and De Cock (2004). In this approach, a query composed of a set of terms is enlarged to similar terms thanks to fuzzy pseudo-thesauri. Similarity is based on the co-occurrence frequency of terms in a given set of documents.

- The use of a fuzzy conceptual structure for document indexing and user query expressing in the framework of information retrieval (Boughanem et al. 2004, Baziz et al. 2006). The conceptual structure is hierarchical and it encodes the knowledge of the topical domain of the considered documents. In this approach, the evaluation of conjunctive queries is based on the comparison of minimal sub-trees containing the two sets of nodes corresponding to the concepts expressed in the document and the query respectively.

However in our context the terms of the hierarchy and the relations between terms are not fuzzy as in the two first categories of papers. Therefore, we could not inspire from those works to solve the questions we mentioned at the beginning of the introduction. We found more analogies with the third category of papers where the terms of the hierarchy and the relations between terms are not fuzzy as in our approach. Even if the interpretation of the weights are different and therefore leads to a different evaluation procedure, these authors show that the completion of the fuzzy sets representing the query and the document description, using the "kind of" relation of the conceptual structure, lead to better results. This idea is close to the notion of generalization of HFS we will present in this chapter.

Basics of fuzzy sets

We briefly present fuzzy sets, which will be used in the following to represent the required values in a flexible query or the possible values in an imprecise datum. We also introduce comparisons between fuzzy sets that will be used to compare an imprecise datum to a flexible query. Fuzzy sets (Zadeh, 1965) were introduced to represent concepts that are not strictly delimited, like "young" or "far" for instance. Unlike the case of a classic set, an element may belong partially to a fuzzy set.

Definition 1: A fuzzy set *A* on a domain *X* is defined by a membership function μ_A from *X* to [0, 1] that associates the degree to which *x* belongs to *A* with each element *x* of *X* .

The domain *X* may be continuous or discrete. Figure 1 presents two examples: the fuzzy sets *ProductPreferences* and *ResponsibleBacterium.* They are also denoted, respectively, *1/Milk + 0.5/Yoghourt*, and *1/Escherichia coli + 0.7/Shigella*, which indicates the degree associated with each element. These fuzzy sets are user-defined, during the choice of the querying selection criteria (*ProductPreferences*), or during the entry of an imprecise datum (*ResponsibleBacterium*).

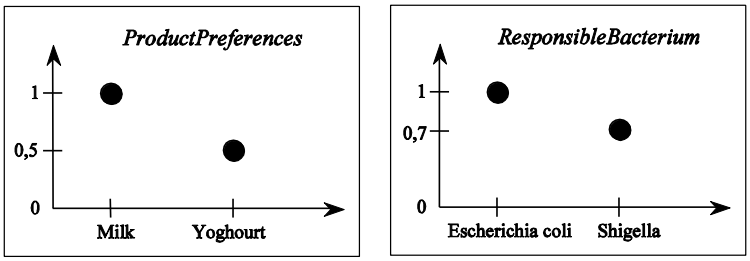


Figure 1 The fuzzy sets ProductsPreferences and ResponsibleBacterium

In the following, we focus on two different comparisons between fuzzy sets: the inclusion relation, that we use to determine in a binary way whether an imprecise datum is an answer to a flexible query or not, and fuzzy pattern matching, which allows one to determine in a graduate way whether an imprecise datum somehow answers a flexible query.
In the most commonly used inclusion relation between fuzzy sets, a fuzzy set *A* (in our case, an imprecise datum) is included in *B* (in our case, a flexible query) if its membership function is "below" the membership function of *B*, that is, if each element that somehow belongs to *A* belongs at least as much to *B*. More formally:

Definition 2: Let A and B be two fuzzy sets defined on a domain X. A is included in B (denoted $A \subseteq B$) if and only if their membership functions μ_A and μ_B satisfy the condition:

$$\forall x \in X, \mu_A(x) \leq \mu_B(x).$$

In fuzzy pattern matching (Dubois & Prade, 1995), two scalar measures are classically used to evaluate the compatibility between an imprecise datum and a flexible query: (i) a possibility degree of matching (Zadeh, 1978); (ii) a necessity degree of matching (Dubois & Prade, 1988).

Definition 3: Let *Q* and *D* be two fuzzy sets defined on a domain *X* and representing respectively a flexible query and an imprecise datum: *D* is compatible with *Q* with the possibility degree $\Pi(Q, D)$ and the necessity degree *N(Q,D):*

- the possibility degree of matching between *Q* and *D*, denoted $\Pi(Q, D)$, is an "optimistic" degree of overlapping that measures the maximum compatibility between *Q* and *D*, and is defined by

$$\Pi(Q, D) = \sup_{x \in X} \min(\mu_Q(x), \mu_D(x)).$$

- the necessity degree of matching between *Q* and *D*, denoted *N(Q,D)*, is a "pessimistic" degree of inclusion that estimates the extent to which it is certain that *D* is compatible with *Q*, and is defined by

$$N(Q, D) = \inf_{x \in X} \max(\mu_Q(x), 1 - \mu_D(x)).$$

MAIN FOCUS OF THE CHAPTER

In the first part of this section, we introduce the concept of hierarchical fuzzy set (HFS). Then in the second part, we present the MIEL flexible querying system which uses the concept of HFS. In the third part, we present two applications that rely on the MIEL querying system, including examples of Graphical User Interfaces (GUI) illustrating use-cases of the MIEL querying system, and experimental results.

Hierarchical Fuzzy Set

In this part, we propose a definition of a hierarchical fuzzy set (HFS) and specify its semantics. Then, we explain why and how we compute the closure of a HFS. Next, we extend the comparison operations between fuzzy sets we have recalled in the background section. We show that the notion of closure permits to group hierarchical fuzzy sets in equivalence classes and that each equivalence class has a unique representative, called *minimal*. Finally, we propose a method of generalization of a HFS based on the minimal HFS.

Definition and semantics

For a given selection attribute, if its domain of values is hierarchized, users always express their preferences on a subset of the domain. Indeed, they only choose the elements they are interested in and implicitly consider that: (i) the elements more specific than those they have chosen must be taken into account by the system, (ii) the other elements must not be taken into account (as the non-comparable elements for example). In the following, we say that an element *elt* of the domain is more general than an element *elt'* (denoted by $elt' \leq elt$) if *elt'* is a predecessor of *elt* in the partial order induced by the hierarchy. An example of such a hierarchy is given in Figure 2 (for instance $Meat \leq Substrate$).

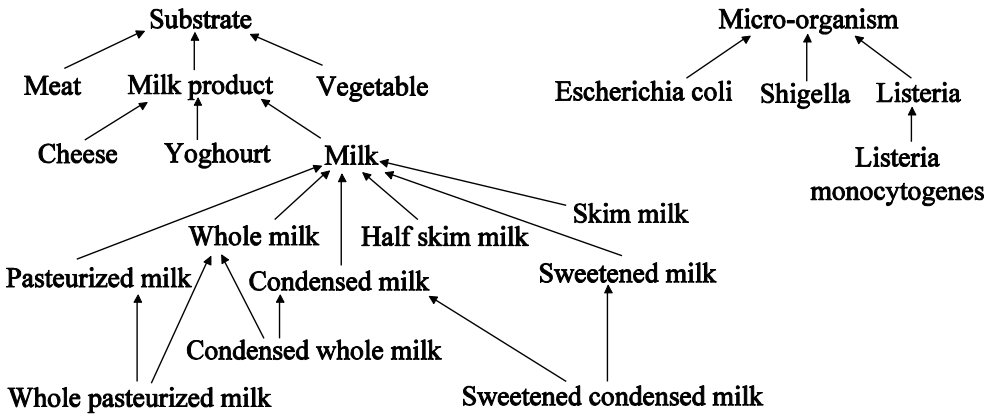


Figure 2 Example of a hierarchy

A hierarchical fuzzy set is then defined as follows.

Definition 4: A *hierarchical fuzzy set (HFS)* is a fuzzy set whose definition domain is a subset of the elements of a finite hierarchy partially ordered by the "kind of" relation.

Example 1: The example of HFS shown in Figure 3 has for definition domain the set of elements {Whole milk, Half-skim Milk, Skim milk} which is a subset of the set of the elements belonging to the hierarchy presented in Figure 2. This HFS may also be noted 1.0/Whole milk + 0.9/Half-skim milk + 0.8/Skim milk.

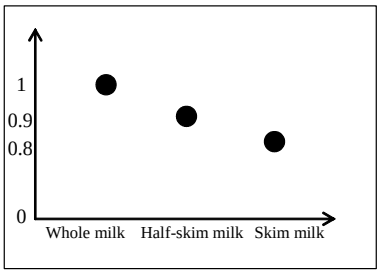


Figure 3 Example of a HFS

We can note that no restriction has been imposed concerning the elements that compose the definition domain of a hierarchical fuzzy set. In particular, the user may associate a given degree *d* with an element *elt* and another degree *d'* with an element *elt'* more specific than *elt*. $d' \leq d$ represents a semantic of restriction for *elt'* compared to *elt*, whereas $d' \geq d$ represents a semantic of reinforcement for *elt'* compared to *elt*.

For example, if there is a particular interest in *Skim milk* because the user studies the properties of low fat products, but also wants to retrieve complementary information about other kinds of milk, these preferences can be expressed using for instance the following fuzzy set: *1/Skim milk + 0.5/Milk*. In this example, the element *Skim milk* has a greater degree than the more general element *Milk*, which corresponds to a semantic of reinforcement for *Skim milk* compared to *Milk*. On the contrary, if the user is interested in all kinds of milk, but to a lesser extent in *Condensed milk* because of its smaller water content, the preferences can be expressed using the following fuzzy set: *1/Milk + 0.2/Condensed milk*. In this case, the element *Condensed milk* has a smaller degree than the more general element *Milk*, which corresponds to a semantic of restriction for *Condensed milk* compared to *Milk*.

Closure

We can make two remarks concerning the use of hierarchical fuzzy sets:

- the first one is semantic. Let *1/Skim milk + 0.5/Milk* be an expression of preferences in a query. We can note that this hierarchical fuzzy set implicitly gives information about elements of the hierarchy other than *Skim milk* and *Milk*. For instance, one can deduce that the user does not expect results concerning products like meat or vegetable, even if the degree 0 has not explicitly been associated with these products. One may also assume that any kind of skim milk (sterilized, pasteurized, raw skim milk for example) interests the user with the degree 1;
- the second one is operational. The problem rising from Definition 4 is that two different fuzzy sets on the same hierarchy do not necessarily have the same definition domain, which means they cannot be compared using the classic comparison operations of fuzzy set theory (see Definitions 2 and 3). For example, *1/Skim milk + 0.5/Milk* and *1/Milk + 0.2/Condensed milk* are defined on two different subsets of the hierarchy of Figure 2 and thus are not comparable.

These remarks led us to introduce the concept of *closure* of a hierarchical fuzzy set, which is a developed form defined on the whole hierarchy. Intuitively, in the closure of a hierarchical fuzzy set, the "kind of" relation is taken into account by propagating the degree associated with an element to its sub-elements (more specific elements) in the hierarchy. For instance, in a query, if the user is interested in the element *Milk*, we consider that all kinds of *Milk* – *Whole milk, Skim milk, Pasteurized milk, etc.* – are of interest. On the opposite, we consider that the super-elements (more general elements) of *Milk* in the hierarchy – *Milk product, Substrate ...* – are too general to be relevant for the user's query.

Definition 5: Let *F* be a hierarchical fuzzy set defined on a subset *D* of the elements of a hierarchy *H*. Its membership function is denoted μ_F. The *closure* of *F*, denoted *clos(F)*, is a hierarchical fuzzy set defined on the whole set of elements of *H* and its membership function $\mu_{clos(F)}$ is defined as follows. For each element *elt* of *H*, let $E_{elt} = \{elt_1, \ldots, elt_n\}$ be the set of the closest super-elements[2] of *elt* in *D* (in the broad sense, i.e. $elt_i \geq elt$):

- if E_{elt} is not empty, $\mu_{clos(F)}(elt) = \max_{1 \leq i \leq n} \mu_F(elt_i)$;
- otherwise $\mu_{clos(F)}(elt) = 0$.

In other words, the closure of a hierarchical fuzzy set *F* is built according to the following rules. For each element *elt* of *H*:

- if *elt* belongs to *F*, then *elt* keeps the same degree in the closure of *F* (case where $E_{elt} = \{elt\}$);
- if *elt* has a unique smallest super-element elt_1 in *F*, then the degree associated with elt_1 is propagated to *elt* in the closure of *F* (case where $E_{elt} = \{elt_1\}$ with $elt_1 > elt$);
- if *elt* has several smallest super-elements $\{elt_1, \ldots, elt_n\}$ in *F*, with different degrees, a choice has to be made concerning the degree that will be associated with *elt* in the closure. The proposition made in Definition 5 consists in choosing the maximum of the degrees associated with $\{elt_1, \ldots, elt_n\}$. This choice is discussed in the following;
- all the other elements of *H*, i.e. those that are more general than, or not comparable with the elements of *F*, are considered as non-relevant. The degree 0 is associated with them (case where $E_{elt} = \varnothing$).

Example 2: Figure 4 shows an example of closure presented on the hierarchy. The elements of the HFS and their associated membership degree appear in bold italic.

[2] with the meaning of the « kind of » relation.

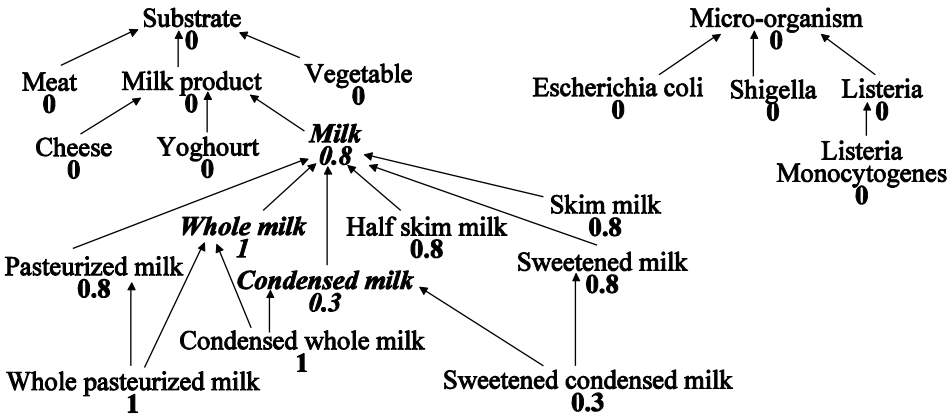


Figure 4 Closure of the hierarchical fuzzy set *0.8/Milk + 1/Whole milk + 0.3/Condensed milk*

In the hierarchical fuzzy set *0.8/Milk + 1/Whole milk + 0.3/Condensed milk* of the au-dessus~~above~~ Figure, the user has associated the degree 1 with *Whole milk* but only 0.3 with *Condensed milk*. The maximum of these two degrees is thus associated with their common sub-element *Condensed whole milk* in the closure. The case of *Sweetened condensed milk* is different: the user has associated the degree 0.8 with *Milk* but has given a restriction on the more specific element *Condensed milk* (degree 0.3). As *Sweetened condensed milk* is a kind of *Condensed milk*, it inherits the degree associated with *Condensed milk*, that is 0.3.

In the case where an element *elt* of the hierarchy, which does not appear in the initial hierarchical fuzzy set, has several smallest super-elements that appear in the hierarchical fuzzy set with different degrees, associating the maximum of these degrees with *elt* in the closure is a choice that may be discussed. We distinguish two cases:

- if the hierarchical fuzzy set expresses preferences in a query, the choice of the maximum allows us not to exclude any possible answer (the possibility and the necessity degrees of matching can be higher). In real cases, the lack of answers to a query generally makes this choice preferable, because it consists in enlarging the query rather than restricting it. This is actually the case in our project;
- if the hierarchical fuzzy set represents an ill-known datum, the choice of the maximum allows us to preserve all the possible values of the datum, but it also makes the datum less specific. We chose this solution in order to homogenize the treatment of queries and data. In a way, it also participates in enlarging the query, as a less specific datum may share more common values with the query (the possibility degree of matching can thus be higher, although the necessity degree can decrease).

We have shown in Thomopoulos et al. (2006) that computing the closure *clos(F)* of a fuzzy set *F* defined on a domain $dom(F) \subset H$ has a complexity in $|H|.|dom(F)|^2$, provided that the comparison of two elements of the hierarchy can be done in constant time. Generally, the definition domain of *F* is limited to a few elements, so that the actual computing time remains moderate. The closure operation has been implemented in the MIEL querying system. For a given query, MIEL computes the closures of the HFS associated with selection attributes before submitting it to the RDBMS.

Comparisons of HFS

The introduction of the concept of closure allows all the fuzzy sets that are defined on a given hierarchy to have the same definition domain (the whole hierarchy) and thus to be compared using the classical comparison operations between fuzzy sets.

Definition 6: Let F_1 and F_2 be two hierarchical fuzzy sets defined on the same hierarchy. Then:

1. $F_1 \subseteq F_2$ *if* $clos(F_1) \subseteq clos(F_2)$;
2. the possibility degree of matching between F_1 and F_2, $\Pi(F_1, F_2)$, is defined as $\Pi(clos(F_1), clos(F_2))$;
3. the necessity degree of matching between F_1 and F_2, $N(F_1,F_2)$, is defined as $N(clos(F_1),clos(F_2))$.

Example 3: We present in Figure 5 the closures of the hierarchical fuzzy sets *1/Skim milk + 0.2/Milk* and *1/Milk + 0.5/Condensed milk*. The elements of the HFS and their associated membership degrees appear in bold italic. Their comparison shows that *1/Skim milk + 0.2/Milk* is included in *1/Milk + 0.5/Condensed milk* because the membership function of the former associates lower degrees with every element of the hierarchy.

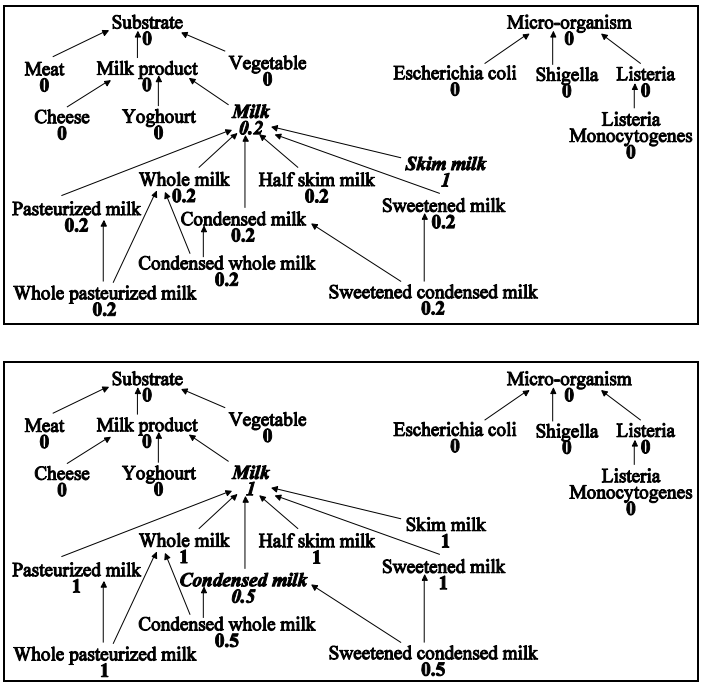


Figure 5 The closures of the hierarchical fuzzy sets *1/Skim milk + 0.2/Milk* (upper part) and *1/Milk + 0.5/Condensed milk* (lower part).

Minimal HFS

In the previous section, we saw that each hierarchical fuzzy set has an associated closure that is defined on the whole hierarchy. We now focus on the fact that two different hierarchical fuzzy sets, defined on the same hierarchy, can have the same closure, as in the following examples.

The hierarchical fuzzy sets $Substrate_1$ = *1/Milk* and $Substrate_2$ = *1/Milk + 1/Skim milk* have the same closure: the degree 1 is associated with *Milk* and every more specific element, the degree 0 is associated with all the other elements of the hierarchy.

The hierarchical fuzzy sets $Substrate_3$ = *1/Milk + 0.8/Whole milk + 1/Pasteurized milk* and $Substrate_4$ = *1/Milk + 0.8/Whole milk + 1/Whole pasteurized milk* have also the same closure, represented in Figure 6.

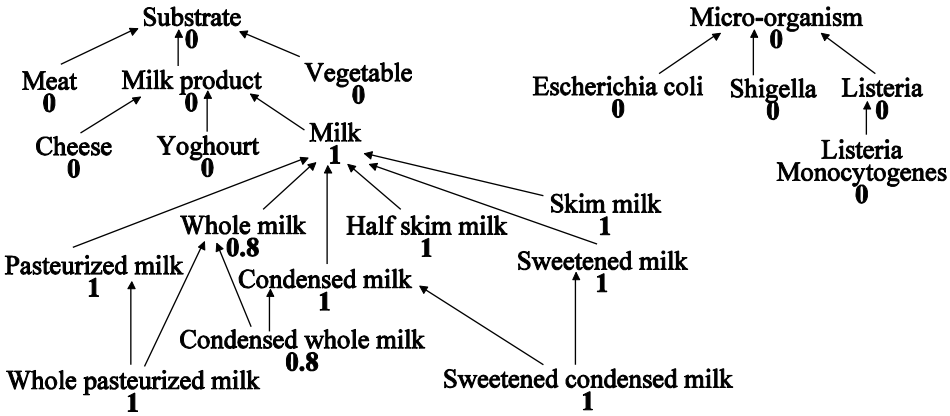


Figure 6 Common closure of the hierarchical fuzzy sets $Substrate_3$ and $Substrate_4$.

Such hierarchical fuzzy sets form equivalence classes with respect to their closures.

We can note that $Substrate_2$ contains the same element as $Substrate_1$ with the same degree, and also one more element (*Skim milk*, with the degree 1). The degree associated with this additional element is the same as in the closure of $Substrate_1$. We say that the element *Skim milk* is ***deducible*** in $Substrate_2$.

Definition 7: Let *F* be a hierarchical fuzzy set, with $dom(F) = \{elt_1, \ldots, elt_j, \ldots, elt_n\}$, and F_{-j} the fuzzy set resulting from the restriction of *F* to the domain *dom(F)* \ $\{elt_j\}$. elt_j is ***deducible*** in *F* if $\mu_{clos(F_{-j})}(elt_j) = \mu_F(elt_j)$.

As a first intuition, we could say that removing a deducible element from a hierarchical fuzzy set allows one to eliminate redundant information. But an element being deducible in *F* does not necessarily mean that removing it from *F* will have no consequence on the closure: removing *elt* from *F* will not impact the degree associated with *elt* itself in the closure, but it may impact the degrees of the sub-elements of *elt* in the closure. For instance, the element *Pasteurized milk* is deducible in $Substrate_3$, according to Definition 7. Removing 1/Pasteurized milk from $Substrate_3$ would not modify the degree of *Pasteurized milk* itself in the resulting closure, but it would modify the degree of its sub-element *Whole pasteurized*

milk (which would have the degree 0.8 instead of 1). Thus, this remark leads us to the following definition of a minimal hierarchical fuzzy set.

Definition 8: In a given equivalence class (that is, for a given closure *C*), a hierarchical fuzzy set is said to be *minimal* if its closure is *C* and if none of the elements of its domain is deducible (here the term "minimal" does not have the meaning of cardinality).

Example 4: The hierarchical fuzzy sets $Substrate_1$ and $Substrate_4$ are minimal (none of their elements is deducible), contrary to $Substrate_2$ and $Substrate_3$.

We propose an algorithm, given below, to calculate a minimal hierarchical fuzzy set. We have proven in Thomopoulos et al. (2006) that the stopping condition of this algorithm is always reached and that the HFS obtained with this algorithm is minimal. Computing the minimal fuzzy set *mnl* of a given closure *C* defined on a hierarchy *H* has a complexity in $|H|.|dom(mnl)|^2$. Moreover, we have also proven in Thomopoulos et al. (2006) that the minimal HFS is unique for a given closure.

Calculation of a minimal fuzzy set *mnl* having a given closure C

```
Begin
    mnl ← ∅
    If (clos(mnl) = C)
    Then
        stop (case where C is the hierarchical fuzzy set that associates
        the degree 0 with every element of the hierarchy)
    Else
        let lin be an order such that each element of the hierarchy
        is examined after its super-elements (that is, a linear
        extension of the opposite order of that induced by the
        "kind of" relation)
        Repeat
            elt ← next element according to lin
            If ( μ_clos(mnl)(elt) ≠ μ_C(elt))
            Then
                mnl ← mnl ∪ {elt}
                μ_mnl(elt) = μ_C(elt)
            Endif
        Until (clos(mnl) = C)
    Endif
End
```

Generalization of a HFS

Using a HFS representing preferences in a query does not guarantee to retrieve an adequate number of answers. A complementary solution to retrieve pertinent answers in addition to exact answers consists in generalizing the HFS. Approaches proposed in the bibliography to generalize fuzzy sets on flat domains, already presented in the background section, are not well-adapted to HFS. Some approaches only concern fuzzy sets defined on a numerical domain (Bouchon-Meunier & Yao 1992, Bosc et al. 2004). Tolerant fuzzy pattern matching (Dubois & Prade, 1995) uses a similarity relation between elements to enlarge the preferences, but it does not take into account the case of hierarchically organized domains. For instance, elements may be added to the support of the fuzzy set in the enlargement mechanism, but more specific elements than those may remain outside of it, which is a major drawback for hierarchical domains (see Buche et al. 2005 for more details).
In this section, more than a unique solution, we propose a methodology in order to generalize a hierarchical fuzzy set expressing preferences. Firstly, we define an elementary generalization operation of a HFS. Then, we introduce the notion of generalization rule which permits to parameter the generalization of a HFS using several criteria. Finally, we propose a generalization operation which applies iteratively several elementary generalizations.

Elementary generalization of a HFS: The elementary generalization of a HFS consists in creating, given a hierarchical fuzzy set F, a more general hierarchical fuzzy set F_g, with the meaning of the inclusion relation extended to HFS. The proof of this property can be found in Thomopoulos et al. (2006).

Definition 9: The elementary generalization of a HFS F is an operation that creates from F a hierarchical fuzzy set F_g obtained by adding a super-element of an element *elt* of *dom(F)*, denoted elt_g, with a given membership degree d_g. The element elt_g must satisfy the following condition: elt_g may neither be an element of *dom(F)* nor be more specific than any element of *dom(F)*.

Example 5: Let F be the following hierarchical fuzzy set: *F = 1/Condensed whole milk + 0.5/Cheese*. For *elt = Condensed whole milk*, we consider in the hierarchy of Figure 2 the super-element elt_g = *Milk* and d_g = 0.2. We obtain: F_g = *1/Condensed whole milk + 0.5/Cheese + 0.2/Milk*.

Generalization rule: We consider that the generalization of a HFS F essentially depends on three parameters: (i) which elements of F will be generalized and in which order, (ii) for a given element of F, which super-elements will be considered for the generalization, (iii) how the membership degree associated with this super-element is determined. A generalization rule permits to determine those three parameters.

Definition 10: A generalization rule R_g is a 3-tuple *(ord, gen, calc)*, where:

- *ord* is a total traversal order through the elements of a hierarchical fuzzy set F, defined on a hierarchy H;
- *gen* is a mapping that associates a set of more general elements in H with each element *elt* in *dom(F)*;
- *calc* is a mapping that associates a degree between 0 and 1 with each pair (elt, elt_g) such that $elt \in dom(F)$ and $elt_g \in gen(elt)$.

Example 6:

- *ord* may be, for instance, an order through the elements of *F* by decreasing degrees. This choice allows one to generalize in priority the elements of *F* that have the higher degrees, that is, the elements for which the user has expressed the higher preference;
- *gen(elt)* may be, for instance, the set of smallest super-elements of *elt* in the hierarchy; this choice permits to minimize the risk to obtain too general answers;
- $calc(elt,\ elt_g) = \min_{\{x \in dom(F) | \mu_F(x) > 0\}} \mu_F(x) \times \mu_F(elt) \times 0.9$ is an example of mapping that permits to retrieve in priority the elements specified by the user.

Each element of *F* does not necessarily have a more general element that may be added to *F* for the generalization operation: as we saw previously in Definition 9, this more general element must satisfy a condition. Here we define the notion of generalizable element of *F*, according to a given generalization rule.

Definition 11: Let *F* be a hierarchical fuzzy set. An element *elt* of *dom(F)* is said to be ***generalizable*** in *F*, according to a generalization rule R_g, if *elt* has a more general element elt_g in *gen(elt)* that satisfies the condition: elt_g may neither be an element of *dom(F)* nor be more specific than any element of *dom(F)*.

Generalization of a HFS: As we saw in section Closure, the MIEL querying system computes the closures of the HFS belonging to a query before submitting it to the RDBMS. Consequently, two queries using two different HFS which belong to the same equivalence class, retrieve the same answer. In order to preserve this property when MIEL performs the generalization of a HFS, this operation is not directly processed on the HFS, but on the minimal HFS, unique representative of the equivalence class which the HFS belongs to.

Definition 12: The generalization of a hierarchical fuzzy set *F*, according to a generalization rule R_g, denoted *gen(F)*, is an operation that provides a hierarchical fuzzy set F_g obtained as follows:

- we call 0-degree generalization of *F*, denoted F_0, the minimal fuzzy set that is equivalent to *F*;
- let F_n be the n-degree generalization of *F*:
  - if there exists an element *elt*, first element (with the meaning of the order *ord*) of $dom(F_0) \subseteq dom(F_n)$ generalizable in F_n according to R_g, then F_{n+1} is obtained by an elementary generalization of F_n according to R_g, in which elt_g is the first element of $dom(F_0) \subseteq dom(F_n)$, generalizable in F_n, and $d_g = calc(elt, elt_g)$;
  - if not, the generalization of *F* is the fuzzy set $F_g = F_n$.

Example 7: Let R_g be the generalization rule proposed in example 6 and *F* the following hierarchical fuzzy set: *F = 1/Whole milk + 1/Condensed whole milk + 0.8/Half skim milk + 0.2/Yoghourt*.

- F_0, the minimal fuzzy set that is equivalent to *F*, is the following: F_0 = *1/Whole milk + 0.8/Half skim milk + 0.2/Yoghourt*;
- the first generalizable element of F_0, in the order *ord*, is *Whole milk*, as $calc(Whole\ milk, milk) = \min_{\{x \in dom(F) | \mu_F(x) > 0\}} \mu_F(x) \times \mu_F(elt) \times 0.9 = 0.2 \times 1.0 \times 0.9 = 0.18$ the generalization provides F_1 = *1/Whole milk + 0.8/Half skim milk + 0.2/Yoghourt + 0.18/Milk*;

- the first element of dom(F_0) generalizable in F_1 is *Yoghourt*, as $calc(Yoghourt, Milk\ Product) = 0.2 \times 0.2 \times 0.9 = 0.036$, the generalization provides F_2 = *1/Whole milk + 0.8/Half skim milk + 0.2/Yoghourt + 0.18/Milk + 0.036/Milk product*;
- there is no element of dom(F_0) generalizable in F_2, so $F_g = F_2$.

We have proven in Thomopoulos et al. (2006) that the number of iterations of the generalization operation is finite and that the fuzzy set F_g obtained by generalization is more general than F with the meaning of the inclusion relation extended to HFS.

MIEL querying system

In this section, we present the MIEL flexible querying system which uses the concept of HFS. The MIEL graphical user interface allows the users to specify a query. Such a query is expressed in a view (selected by the user from a list of available views). The users also specify in the query a set of projection attributes and a set of selection criteria. Then the MIEL user interface sends the MIEL query to the relational subsystem. The relational subsystem adapts the query to the formalism it uses (an SQL query), then asks the RDBMS query processor to execute the query. Finally, the answers to the query are returned to the MIEL interface which presents them to the users. Firstly, we present the choices we made in the design of the MIEL data model. Then, we successively present the MIEL query language and the MIEL query processing.

MIEL data model

The MIEL data model is composed of an abstract data model, called the *ontology*, and several concrete data models which depend on the actual data model chosen to store the data (RDB or other formalisms). In this chapter, we are only concerned with the ontology and with the RDB concrete data model, i.e. the RDB schema.

Ontology of the MIEL data model: the ontology contains the knowledge of the domain used by the MIEL system. The basic notion of the ontology is the concept of *attribute* which must be understood in its classic database meaning. In order to take into account the imprecision of the values stored in the data of the MIEL system, we propose to use, instead of crisp values, imprecise values expressed as possibility distributions represented by fuzzy sets (see Zadeh, 1978). A variation domain and a definition domain are associated with each attribute. The variation domain corresponds to the universe of discourse; the definition domain is the set of fuzzy sets which can be defined on the variation domain: it corresponds to the actual domain in the classical database meaning.

Definition 13: A is the finite set of attributes of the MIEL data model. Each attribute $a \in A$ is characterized by its type *Type(a)*, its variation domain $dom_v(a)$ and its definition domain *dom(a)*. The type *Type(a)* of an attribute a can be numerical, symbolic or hierarchized. Depending on its type, the variation domain $dom_v(a)$ of an attribute a is:

- if *Type(a)* is numerical, $dom_v(a)$ is defined as a subset of $\Re$, the set of the real values;
- if *Type(a)* is symbolic, $dom_v(a)$ is defined as a set of symbolic constants;
- if *Type(a)* is hierarchized, $dom_v(a)$ is defined as a set of symbolic constants and a partial order defined on it.

In all cases, *dom(a)* is defined as the set of all the possible fuzzy sets on $dom_v(a)$.

Definition 14: The value of an attribute *a* belongs to *dom(a)* and is denoted $\tau(a)$. It is a map π of $dom_v(a)$ to [0,1]. We denote $\pi(x)$ the degree of possibility that the effective value of *a* is x ($x \in dom_v(a)$).

Example 8: The variation domain of the numerical attribute *pH* is the interval [0,14] on $\Re$. The variation domain of the symbolic attribute *Author* could be the set {S.Ajjarapu, C.P.Rivituso, M.Zwietering}. A part of the variation domain of the hierarchized attribute *Substrate* is represented in Figure 7.

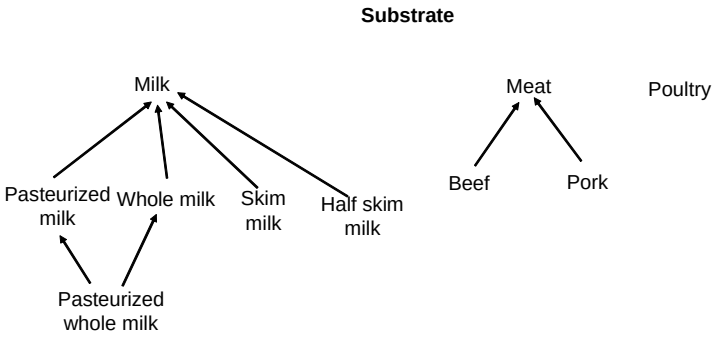


Figure 7 A part of the variation domain of the attribute Substrate

The value *pH_value* of Figure 8 schematizes an example of value for the attribute *pH* (that value belongs to *dom(pH)*: it is a map of $dom_v(pH)$ into [0,1]). The value *Substrate_value* of Figure 8 schematizes an example of value for the attribute *Substrate* (that value belongs to *dom(Substrate)*: it is a map of $dom_v(Substrate)$ into [0,1]; the elements of $dom_v(Substrate)$ having a degree equal to 0 are not represented).

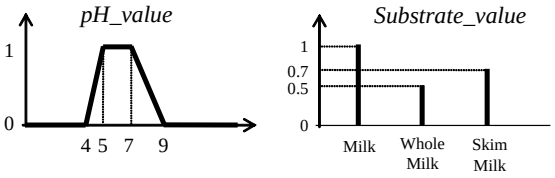


Figure 8 Two examples of imprecise values

Note that in our data model, we consider that all the values are imprecise values. The case of a crisp value for an attribute *a* is a particular case of an imprecise value, such that $\exists x \in dom_v(a)\left[\pi(x) = 1, and\ \forall y \neq x, \pi(y) = 0\right]$.

For simplicity, and since it corresponds to the application needs, we chose to limit the representation of numerical values to trapezoidal functions in the actual database. These trapezoidal functions are stored by means of 4 characteristic points defining the limits of the support and the kernel of the fuzzy set. In the example of Figure 8, these 4 characteristic points are [4, 5, 7, 9].

Schema of the relational database: In the following, we do not present in detail the relational database schema (which is a classic RDB schema), but we focus on the choices we have made in order to map the ontology of the MIEL data model presented previously onto the RDB schema. We present how the attributes and their variation domains are represented in the RDB. We successively consider the way of representing an attribute belonging to the

ontology of the MIEL data model, when that attribute is respectively of type numerical, symbolic or hierarchized.

Representation of a numerical attribute: The representation of a value of a numerical attribute in the relational schema is done by means of a row of an additional table which contains the unique identifier of the numerical fuzzy set and 4 attributes which correspond to the 4 characteristic points of the trapezoidal function. Existing techniques (see Galindo et al. 2006) could be used to represent other kinds of values (such as ranges, unknown, undefined…).

Example 9: **The tables in Figure 9 present an example of a numerical attribute represented in the relational database.**

ExpeId	Substrate	FuzzyPHId
10	Pork	200
11	Skim Milk	231

FuzzySetId	MinSupp	MinKer	MaxKer	MaxSupp
200	4	5	6	7
231	6	6	6	6

Figure 9 The upper table presents an example of relation referencing numerical fuzzy values. The lower table contains a part of the relation *NumericalFuzzySet* which stores the actual numerical fuzzy sets (the second row corresponds to a crisp value in this example).

Representation of a symbolic attribute: The representation of a value of a symbolic attribute *a* of *A* in the relational schema is done by means of one or several rows of an additional table which contains three columns: the unique identifier of the fuzzy set, an element of $dom_v(a)$ and its associated membership degree in that fuzzy set. We remind that a fuzzy set on a symbolic variation domain is defined as a set of pairs (*element, degree*).

Example 10: Tables in Figure 10 present an example of the value of a symbolic attribute represented in the relational database.

Substrate	FuzzyOriginId
Pork	100

FuzzyOriginId	Country	Degree
100	USA	1
100	Germany	1
100	France	0.8

Figure 10 The left table presents an example of relation referencing symbolic fuzzy values. The right table contains a part of the relation *SubstrateOrigin* which contains symbolic fuzzy sets (only one fuzzy set is represented in this example).

In addition, the variation domain of each attribute *a* of *A* of type *symbolic* used in the relational schema is stored in a reference table which contains all the possible values that compose $dom_v(a)$.

Representation of a hierarchized attribute: The representation of a value of a hierarchized attribute of *A* in the relational schema is done in exactly the same way as the representation of a symbolic attribute (see above).
The variation domain of each attribute *a* of *A* of type *hierarchized* used in the relational schema is stored in two specific tables: a table which contains all the possible values that compose $dom_v(a)$ and a table which contains all the pairs $\{v_i, v_j\}$ of the cover relation of the partial order of $dom_v(a)$.

Example 11: Tables presented in Figure 11 are partial instances of relations *Ref_Substrate* and *Hier_Substrate* describing the hierarchized variation domain for substrates in the relational schema.

Substrate
Milk
Full milk
Pasteurized milk
Pasteurized full milk

SubstrateSup	**SubstrateInf**
Milk	Full milk
Milk	Pasteurized milk
Pasteurized milk	Pasteurized full milk
Full milk	Pasteurized full milk

Figure 11 The left table presents a part of relation Ref_Substrate. The right table presents a part of relation Hier_Substrate.

When an attribute is known to be a "crisp" value (for example the substrate in

Figure 9

~~**Figure 9**~~

~~Figure 9~~), the database designers have used classic database attributes of type real, integer or string instead of fuzzy values.

As presented above, the ontology of the MIEL data model is stored in specific tables of the relational database schema. This is due to the fact that we have to proceed to referential integrity control in the data we store.

Flexible query language

A query asked on the MIEL system is expressed in the MIEL query language through the MIEL graphical user interface. In the following, we present the notions we use in a way close to domain relational calculus (Ullman, 1988). We use this query language in a data integration system in which data are stored using different models (relational model, conceptual graph model or XML model presented in section Futures Trends). It is the reason why a MIEL query is always expressed in a view. It provides two advantages: (i) the user always interacts with a unique querying language and does not need to know which models are used to store the data (ii) the data integration system is extensible: to add a new data model, one have to design a new mediator which translates a MIEL query expressed in a view into a query adapted to the data model.

The notion of view: A view is a usual notion in relational databases: it is a virtual table built from the actual tables of the relational database schema by means of a query. In the MIEL system, a set of views (which are pre-written queries) is proposed to the users in order to hide the complexity of the database schema.

Definition 15: A view V on n ($n > 0$) queryable attributes $a_1,\ldots,a_n$ of the MIEL ontology is defined by $V = \{a_1,\ldots,a_n | P_V(a_1,\ldots,a_n)\}$ where P_V is a predicate which characterizes the construction of the view.

Example 12: The view *OneFactorExperience* is defined on 5 attributes: *OneFactorExperience = {Substrate, PathogenicGerm, PH, Factor, ResponseType | $P_{OneFactorExperience}$ (Substrate, PathogenicGerm, PH, Factor, ResponseType)}*. The predicate $P_{OneFactorExperience}$ defines the way the attributes involved in the view are linked together. That view characterizes the result of experimentations in which only one factor is controlled (for example: the temperature). The response type can be, for example, the growth speed of a pathogenic germ in a given substrate.

Expression of a query: A query in the MIEL system is a specialization of a given view by the end-user, who specifies a set of projection attributes as a subset of the queryable attributes of the view and a set of conjunctive selection criteria on some other attributes.

Definition 16: A query *Q* asked on a view *V* is defined by: $Q = \left\{a_1,\ldots,a_p \middle| \exists a_{q+1},\ldots,a_n \left(P_V(a_1,\ldots,a_n) \wedge \left(a_{p+1} \approx v_{p+1}\right) \wedge \ldots \wedge \left(a_q \approx v_q\right)\right)\right\}$ and $g_{p+1},\ldots,g_q$ and $\left(\prod_{\min}, N_{\min}\right) \in \left([0,1]^2\right)$ where P_V is the predicate which characterizes the view *V*, $a_1,\ldots,a_p$ are the projection attributes, $a_{p+1},\ldots,a_q$ are the selection attributes and their respective values $v_{p+1},\ldots,v_q$ given as selection values by the user, $g_{p+1},\ldots,g_q$ are boolean values specifying whether the fuzzy set $v_{p+1},\ldots,v_q$ must be generalized or not, $\left(\prod_{\min}, N_{\min}\right) \in \left([0,1]^2\right)$ its minimum possibility and necessity degrees. The attributes $a_{q+1},\ldots,a_n$ are the queryable attributes of the view which are not used in that query.

In the previous definition, the comparator $\approx$ stands for "approximately equal" and will be interpreted in the answer by the two classical scalar measures used in fuzzy pattern matching (Dubois & Prade, 1995) to evaluate the compatibility between an imprecise datum and a fuzzy set representing the selection values.

Example 13: The query *Q* is expressed in the view *OneFactorExperience*:
Q = {Substrate, PathogenicGerm, pH, Factor, ResponseType | $P_{OneFactorExperience}$ (Substrate, PathogenicGerm, pH, Factor, ResponseType) $\wedge$ (Substrate$\approx$ SubstratePreferences) $\wedge$ (pH$\approx$ pHPreferences)} where the fuzzy sets *SubstratePreferences* and *pHPreferences* are presented in Figure 12 and their associated boolean values in the query are false (no generalization). The minimum possibility and necessity degrees are set to $\prod_{\min} = 0.8$ and $N_{\min} = 0.0$.

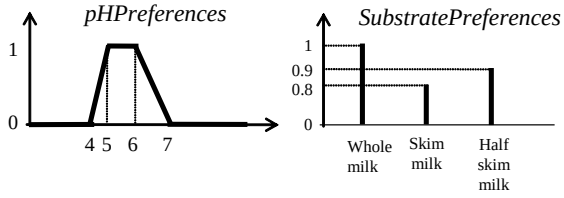


Figure 12 Preferences expressed by the user

The answers: An answer *A* to a query *Q* in the MIEL system is a set of tuples. Each tuple is composed of values (which are fuzzy sets as presented in Definition 14). Each tuple satisfies the selection criteria of the query. If the generalization of a selection value is required in the query, the way this operation is processed depends on the type of the selection attribute. For an attribute of type numerical or symbolic, the generalization is processed as in Dubois & Prade (1995). When the attribute is of type hierarchized, the generalization of the hierarchical

fuzzy set is processed as described in definition 12 of section Generalization of an HFS. In the following definition, for any type of selection attribute, we denote by *gen(F)* the generalization of the fuzzy set *F*, its corresponding fuzzy selection value.

Definition 17: Let $Q = \{a_1,\ldots,a_p \mid \exists a_{q+1},\ldots,a_n\ (P_V(a_1,\ldots,a_n) \wedge (a_{p+1} \approx v_{p+1}) \wedge \ldots \wedge (a_q \approx v_q))\}$ be a query, $g_{p+1},\ldots,g_q$ its boolean values and $(\prod_{\min}, N_{\min})$, its minimum possibility and necessity degrees. The answer *A* to the query *Q* is: $A = \{\tau_1,\ldots,\tau_r\}$, the set of tuples of the form $\{\tau_i[a_1],\ldots,\tau_i[a_p]\}_{1\le i\le r}$, such that every tuple of *A* satisfies all the selection criteria of *Q*, with $\pi_i = t(\Pi(f_{p+1}(v_{p+1}), \tau(a_{p+1})),\ldots,\Pi(f_q(v_q), \tau(a_q)))$ and $n_i = t(N(f_{p+1}(v_{p+1}), \tau(a_{p+1})),\ldots,N(f_q(v_q), \tau(a_q)))$ their respective possibility and necessity degrees of matching (as defined in Definition 3), such as $\pi_i \ge \prod_{\min}$ and $N_i \ge N_{\min}$, *t* a t-norm and $f_j(v_j) = gen(v_j)$ if b_j is true and v_j otherwise.

We have chosen the *min* operation to implement the t-norm *t* which is a classical choice to represent the conjunction but of course other t-norms may be used.

Example 14: An example of answer corresponding to the query of example 13 expressed in the view *OneFactorExperience* is given in Figure 13.

$(\prod, N)$	**Substrate**	**PathogenicGerm**	**pH [min, max]**	**Factor**	**ResponseType**
(1.0, 1.0)	Whole milk	Bacillus Cereus	[5.1, 5.2]	Temperature	Temporal cinetic
(0.9, 0.9)	Half skim milk	Listeria	[5.0, 5.4]	Temperature	Growth speed
(0.8, 0.0)	Skim milk	Listeria	[6.0, 8.0]	Temperature	Level of contamination

Figure 13 Part of an answer

Query processing

The MIEL user interface sends the MIEL query to the relational subsystem. The relational subsystem adapts the MIEL query to the formalism it uses: an SQL query. The views in the relational database of the MIEL system are SQL queries. In the actual implementation of the MIEL system, the views are stored in a specific table of the database called *LViews*, in which each tuple represents a view and is composed of four columns: *IdView* is the unique identifier of the view, *SelectPart* is the list of projection attributes, *FromPart* is the list of relations involved in the view, *WherePart* is the list of join predicates between those relations. We did not use the notion of view because it is not implemented in all RDBMS, especially MySQL which is very popular.

Example 15: The SQL query which corresponds to the view *SubstrateList* defined in Figure 14 is: select P.Title, S.Substrate from Publication P, Substrate S where P.IdPub = S.IdPub.

IdView	SelectPart	FromPart	WherePart
SubstrateList	P.Title,	Publication P,	P.IdPub=S.IdPub

	S.Substrate	Substrate S	

Figure 14 A partial instance of relation *LViews*

The query processing in the database subsystem is processed as follows:

1. selection of the view corresponding to the query;
2. for each selection attribute of type hierarchized, computation of the fuzzy set closure and optionally, if it is expressed in the query, computation of the generalization of the hierarchical fuzzy set;
3. for each selection attribute of type numerical or symbolic, optionally if it is expressed in the query, computation of the generalization of the fuzzy set;
4. transformation of the fuzzy values of the selection criteria into classic SQL conditions (we call that process "defuzzification"). This transformation depends on the type of the selection attribute. If the type is hierarchized, a list of queried values is built from the list of elements which belong to the fuzzy set closure. If the type is symbolic, a list of queried values is built from the list of elements which belong to the associated fuzzy set. If the type is numerical: a list of boolean conditions checks both cases of overlapping between the fuzzy sets which represent the selection criterion and the one representing the imprecise datum (see Haemmerlé et al 2007 for more details);
5. completion of the SQL query corresponding to the view in order to build the actual "defuzzified" SQL query;
6. subsmission of the SQL query to a standard relational database management system (ORACLE or Postgresql in the present version);
7. computation of the adequation degree of each tuple of the answer.

Example 16: We assume that the query asked through the MIEL graphical user interface is: {Title, Substrate | SubstrateList(Title,Substrate) $\wedge$ (Substrate $\approx$ *SubstratePreferences*)}. The selected view is that of example 15 and the fuzzy set *SubstratePreferences* is that of example 13. Using the hierarchy of Figure 7, the fuzzy set closure and the defuzzification of the selection criterion lead to the actual selection criterion: Substrate in ('Whole milk', 'Skim milk', 'Half skim milk', 'Pasteurized whole milk'). The SQL query submitted to the RDBMS query processor is then: select P.Title, S.Substrate from Publication P, Substrate S where P.IdPub = S.IdPub and S.Substrate in ('Whole milk', 'Skim milk', 'Half skim milk', 'Pasteurized whole milk').

Applications and experimentation

In this part, we present two applications based on the MIEL querying system. Then examples of Graphical User Interface (GUI) illustrating a use-case are given. Finally, we give some experimental results about the closure and the generalization of a HFS.

Presentation of two applications

We have designed and implemented two instances of the MIEL flexible querying system, involving hierarchical fuzzy sets for two different relational databases:

- the Sym'Previus (http://www.symprevius.org/) database which contains around 10.000 data about the behaviour of microbial contaminants in foods. This system has been developed with industrial partners (Danone, Bongrain, Pernod-Ricard, …) and governmental institutions (French Ministry of Agriculture and Fisheries);

- the Mét@risk (http://metarisk.inapg.inra.fr/) database which contains around 50.000 data about chemical contamination in foods. This system has been developed by the Mét@risk INRA research unit with a national governmental institution (the French ministry of Agriculture and Fisheries) and an international institution (the WHO[3] Gems Food).

Both systems are operational and a copy is accessible from the Web:

- Sym'Previus database: http://www.symprevius.org/ (contact olivier.couvert@adria.tm.fr to obtain an access)
- Mét@risk database: http://idot.inapg.fr/mielContaminant/web/ (contact Buche@agroparistech.fr to obtain an access)

The Sym'Previus ontology contains 2 attributes of type hierarchized. For the attribute *Substrate*, the ontology contains 507 terms with a maximum specialization depth of 7. For the attribute *MicrobialContaminant (microorganism)*, the ontology contains 167 terms with a maximum specialization depth of 7 too.

The Mét@risk ontology contains 7 attributes of type hierachized. Six attributes describe the food: *ProductType* (481 terms), *FoodSource* (2239 terms), *CookingMethod* (42 terms), *TreatmentApplied* (628 terms), *PackingMedium* (37 terms), *ContainerOrWrapping* (285 terms). For those attributes, the maximum specialization depth is 6. For the attribute *ChemicalContaminant*, the ontology contains 222 terms with a specialization depth of 1.

Examples of GUI

We present some dialogues with the user through an example of query. Although those dialogs could of course be enhanced to help users introducing fuzzy sets, they have been judged enough "intuitive" by our microbiologist partners who helped us to design the GUI. The user wants to query the view *OneFactorExperience* (see example 12). Five queryable attributes are available: *Substrate, PathogenicGerm, pH, Factor, ResponseType.* The user expresses preferences about the *Substrate* using the HFS (1.0/Cheese: soft + 0.9/Cheese) and about the *pH* using the numerical fuzzy set [4, 5, 6, 7]. Figure 15 shows the window in which the user expresses the HFS about the *Substrate*. The part of the ontology concerning the attribute *Substrate* can be accessed by the user in the frame *Hierarchy of food products*. The user's preferences are registered in the list boxes of the frame *Set of values*. The list boxes also give access to the ontology by alphabetic order. In the frame *Hierarchy of food products*, the button *Vizualise the choice* asks the MIEL system to color in red the terms of the ontology which belong to the fuzzy set closure and in purple the terms obtained by a generalization of the fuzzy set. This last operation is computed by the system when the check box *Extended selection* of the frame *Set of values* is marked.

[3] World Health Organization

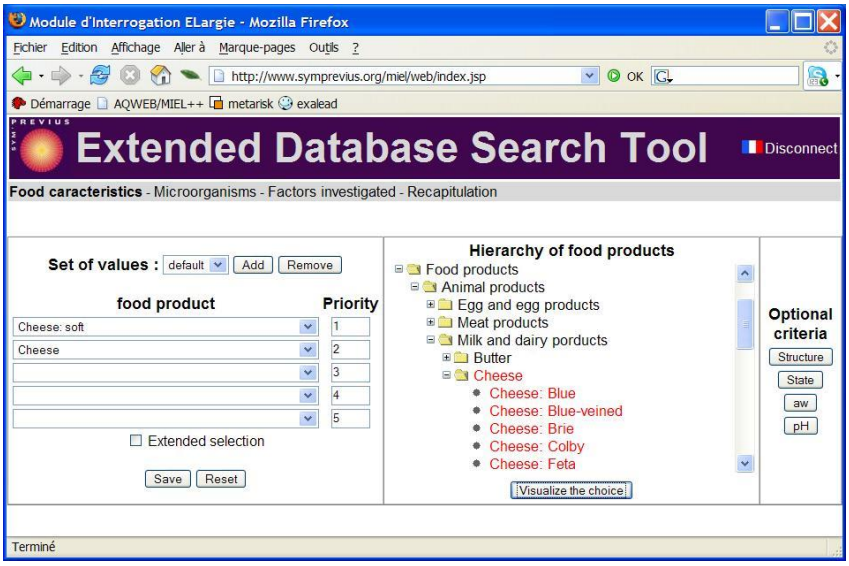


Figure 15 Graphical user interface to register a HFS

Figure 16 shows the window which permits one to define a fuzzy set of a numerical type. It is used in the example to specify *pH* preferences.

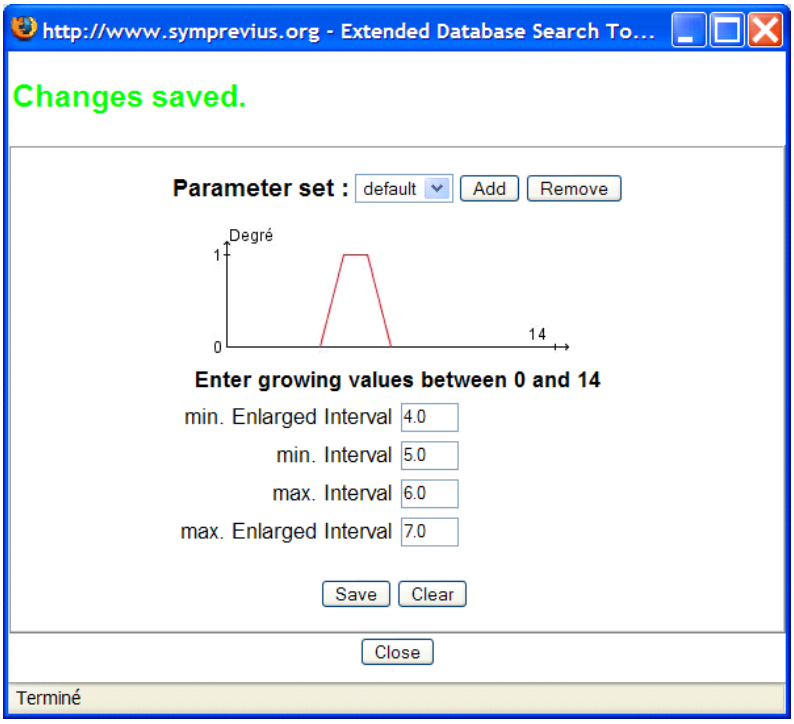
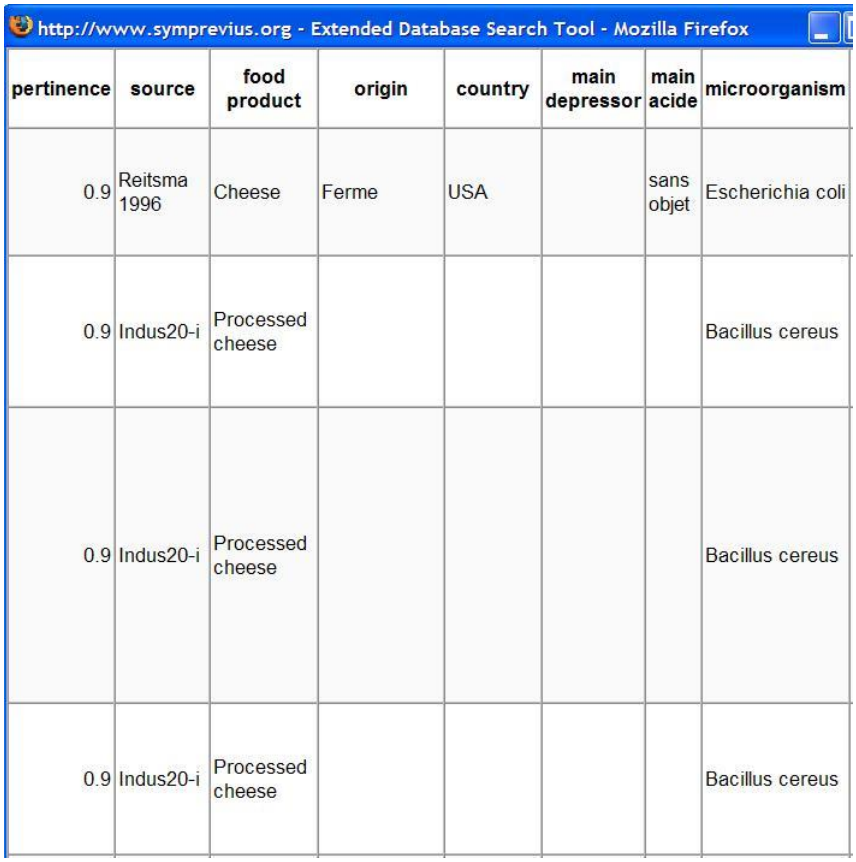


Figure 16 Graphical user interface to register a fuzzy set of type numerical

A part of the answer to the query is presented in Figure 17. The first column contains the possibility degree of matching (see Definition 17) of each tuple. For instance, the first answer is about the behaviour of Escherichia coli in Cheese and has been published in Reitsma (1996). Data have been collected in a farm in the USA. The second answer is about the behaviour of Bacillus cereus in Processed cheese and has been collected by an anonymous industrial partner of the Sym'Previus network. As both answers correspond to a cheese or a kind of cheese (Processed cheese), they are associated with the degree 0.9. Answers can be downloaded on the user's computer in Excel format for further manipulations.



pertinence	source	food product	origin	country	main depressor	main acide	microorganism
0.9	Reitsma 1996	Cheese	Ferme	USA		sans objet	Escherichia coli
0.9	Indus20-i	Processed cheese					Bacillus cereus
0.9	Indus20-i	Processed cheese					Bacillus cereus
0.9	Indus20-i	Processed cheese					Bacillus cereus

Figure 17 An example of answer to a query in the view *OneFactorExperience*

Experimentation

We have made an experimentation in order to evaluate the efficiency of the closure and the generalization operations. We have defined with the experts of the domain 7 test queries which cover around 10 percent of the Sym'Previus database entries (1132 data entries among 10.000 entries). The same query has been executed 3 times: without the computation of the closure (denoted standard queries), with the computation of the closure, with the computation of the closure and the generalization. Exact answers correspond to the answers obtained either by the standard queries or by the queries with the computation of the closure. All the answers obtained with the computation of the closure have been considered as correct by the experts. They represent 99 percent of the exact answers (1 percent of the exact answers correspond to the standard queries). It represents an excellent result for the closure operation. Among the results obtained by the generalization operation, answers that are judged pertinent by the experts (80 percent of the total number of answers obtained by generalization) are those which have the highest matching degrees (between 0.6 and 0.8), whereas the answers that are judged non-pertinent (20 percent) have degrees that go from 0.6 to 0.2. An essential remark is that the value 0.6 can thus be considered as a threshold above which results are classified as pertinent, and below which results are classified as non-pertinent by the experts. The evaluation results are thus also very good for the generalization method, as (i) pertinent results can be clearly identified using their matching degrees; (ii) generalization results bring an important amount of complementary results (56 percent of the total number of results regarded as exact or pertinent).

FUTURE TRENDS

In this section, we expose in more details the continuation of this project within our own research team.

In this chapter, we have presented the concept of hierarchical fuzzy set and its application to query a possibilistic database, thanks to the MIEL flexible querying system, in the framework of the relational model. The concept of hierarchical fuzzy set has also been implemented in two other representation formalisms, the conceptual graph model and XML, in the framework of the design of a data integration system. Our data integration system integrates three subsystems: a relational database (RDB) subsystem, a conceptual graph base (CGDB) subsystem and an XML base (XMLDB) subsystem.

The relational database contains the stable, well-structured part of the information. The conceptual graph base contains the weakly structured pieces of information which do not fit the relational schema. As changing a relational schema is quite an expensive operation, we decided to use an additional base in order to store information that was not expected when the schema of the database was designed, but that is useful nevertheless. We chose to use the conceptual graph model for many reasons, including (i) its graph structure, which appeared as a flexible way of representing complementary information, and (ii) its readability for a non-specialist. The XML base contains information found semi-automatically on the Web by the AQWEB tool. AQWEB scans the Web, retrieves and filters documents which "look like" scientific publications. Tables containing scientific data are extracted automatically from each document and stored in an XML document. In order to be able to query efficiently the XML documents containing data tables, those tables are annotated using the domain ontology. For instance, AQWEB tries to match each term of the table with the closest terms of the ontology. XML documents including annotations are stored in an XML native database to enhance their querying.

The data stored in the three bases are expressed in different formalisms, but conform to a single domain ontology. That domain ontology consists of a set of attributes and their associated variation domain (see Definition 13). The user expresses his query in the MIEL language. This query is then sent simultaneously to the three subsystems which transform the MIEL query into a formalism adapted to each subsystem (an SQL query in the RDB subsystem, a conceptual graph query in the CGDB subsystem and an Xquery query in the XMLDB subsystem). The mediators which process this task have been presented respectively in Haemmerlé et al. (2007) for the CGDB subsystem and Buche et al. (2006) for the first version of the XMLDB subsystem. The query is executed by each subsystem and the answers are retrieved by the MIEL graphical interface to be presented in a homogeneous way to the user.

In the framework of the French national project WebContent (http://www.webcontent.fr), we are currently working on a new version of the AQWEB tool. In particular, we produce a new annotation represented as a fuzzy set, associating terms of the ontology with their similarity to the term of the web table (Hignette et al., 2005). Consequently, we will have to consider that a hierarchical fuzzy set associated with an attribute of type hierarchized may have not only two but three different semantics: a semantics of preference or of possibility distribution (the ones we studied in this chapter), but also a semantics of similarity. We will have to take into

account all the consequences of this extension in the new version of the MIEL XMLDB mediator which will be developed in the framework of the WebContent project.

CONCLUSION

Fuzzy sets are used both in flexible querying, to allow the expression of user's preferences, and in possibilistic databases, to represent imprecise data by means of possibility distributions. In this chapter, we have focused on the case where these fuzzy sets are defined on hierarchically organized domains. Such domains are widely used in ontology-based systems. Defining fuzzy sets on hierarchically organized domains is not a trivial issue since the degree associated with an element in such a fuzzy set must be coherent with those associated with sub-elements or super-elements, and compatible with reasoning using fuzzy set operations.

We first proposed an overview of existing works in two fields, whose combination is the core of the chapter: flexible querying of imprecise data, and fuzziness in hierarchies.
Then we introduced the concept of hierarchical fuzzy set (HFS) and presented its properties: two ways of defining it, on a part of a hierarchy or by computing its closure on the whole hierarchy; the extension of fuzzy set operations to hierarchical fuzzy sets, based on the HFS closures; the existence of equivalence classes composed of hierarchical fuzzy sets that share the same closure; the existence of a unique representative which has a property of minimality within each equivalence class; the generalization of a hierarchical fuzzy set, based on its equivalent minimal fuzzy set and useful for flexible querying purposes.
We presented the framework of the MIEL flexible querying system, its data model, its query language, its query processing. It implements the concept of hierarchical fuzzy set and is used for the querying of two different relational databases, containing imprecise data, in the domain of risk assessment in food products.
We illustrated the chapter with the examples of the two applications, their graphical user interface and an experimental evaluation. This evaluation shows that 99 percent of the total number of exact answers is obtained thanks to the closure computation and that 56 percent of the total number of results regarded as exact or pertinent is obtained by the generalization mechanisms presented in the chapter.
Finally we outlined some future trends concerning both hierarchical fuzzy sets and the MIEL flexible querying system.

The essential point to retain from this chapter appears to us as being the relevance of hierarchical fuzzy sets for ontology-based systems, which extends their use beyond the framework of a particular application or a particular representation formalism.

REFERENCES

Ballows, A., Truper, H., Dworkin, M., Harder, W., & Schleifer, K. (Eds.). (1992). *The prokaryotes, a handbook on the biology of bacteria: Ecophysiology, isolation, identification, applications* (2nd ed.). Berlin Heidelberg New York: Springer.

Baziz M., Boughanem M., Prade H., & Pasi G. (2006) A fuzzy logic approach to information retrieval using an ontology-based representation of documents. *In*: *Fuzzy Logic and the*

Semantic Web. (E. Sanchez, eds.), p. 363-377.

Bidault, A., Froidevaux, C., & Safar, B. (2000). Repairing queries in a mediator approach. *Proceedings of the 14th European Conference on Artificial Intelligence* (pp. 406-410).

Bosc, P., & Pivert, O. (1992). Some Approaches for Relational Databases Flexible Querying. *J. Intell. Inf. Syst. 1*(3/4), 323-354.

Bosc, P., Lietard, L., & Pivert, O. (1994). Soft querying, a new feature for database management system. *Proceedings DEXA'94 (Database and EXpert system Application), Lecture Notes in Computer Science, 856* (pp. 631-640). Springer-Verlag.

Bosc, P., & Pivert, O. (1995). SQLf: A relational database language for fuzzy querying. *IEEE Transactions on Fuzzy Systems*, *3*(1), 1–17.

Bosc, P., HadjAli, A., & Pivert, O. (2004). Fuzzy Closeness Relation as a Basis for Weakening Fuzzy Relational Queries. In Christiansen, H.; Hacid, M.-S.; Andreasen, T., & H. L. Larsen (Eds), *Proceedings of Flexible Querying Answering Systems, Lecture Notes in Computer Science, 3055* (pp. 41-53). Springer.

Bouchon-Meunier, B. & Yao, J. (1992). Linguistic modifiers and imprecise categories. *International Journal of Intelligent Systems* 7, 25-36.

Boughanem M., Pasi G., & Prade H. (2004). A fuzzy set approach to concept-based information retrieval. *Proceedings of 10th Inter. Conf. IPMU'04, Perugia (Italy), 4-9/07/2004*, IPMU, p. 1775-1782.

Buche, P., Dervin, C., Haemmerlé, O. & Thomopoulos, R. (2005). Fuzzy querying on incomplete, imprecise and heterogeneously structured data in the relational model using ontologies and rules, *IEEE Transactions on Fuzzy Systems*, *13*(3), 373-383.

Buche, P., Dibie-Bartélemy, J., Haemmerlé, O. & 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.

Codd, E. F. (1979). Extending the Database Relational Model to Capture More Meaning. *ACM Trans. Database Syst.* 4(4), 397-434.

De Cock, M., S., G. & Nikravesh, M., (2004). Fuzzy thesauri for and from the www. In Nikravesh, M., Zadeh, L., Kacprzyk, J., (Eds.) *Soft Computing for Information Processing and Analysis* (pp. 275-284).

Dubois, D., & Prade, H. (1988). *Possibility theory – An approach to computerized processing of uncertainty*. New York: Plenum.

Dubois, D., & Prade, H. (1995). Tolerant fuzzy pattern matching : an introduction. In P. Bosc and J. Kacprzyk (eds), *Fuzziness in Database Management Systems* (pp. 42-58). Heidelberg: Physica Verlag.

Fargues, J. (1989). CG information retrieval using linear resolution, generalization and graph splitting. *Proceedings of the Fourth annual workshop on conceptual graphs*.

Galindo J., Medina J.M., Pons O.,& Cubero J.C. (1998). A Server for Fuzzy SQL Queries. In "Flexible Query Answering Systems", eds. T. Andreasen, H. Christiansen and H.L. Larsen, Lecture Notes in Artificial Intelligence (LNAI) 1495, pp. 164-174. Ed. Springer (http://www.springerlink.com/content/ddyttjwpn31hxer4).

Galindo J., Urrutia A., & Piattini M. (2006). *Fuzzy Databases: Modeling, Design and Implementation*. Idea Group Publishing, Hershey, USA.- ISBN: 1-59140-325-1.- http://www.idea-group.com

Haemmerlé, O., Buche, P. & Thomopoulos, R. (2007). The MIEL system: uniform interrogation of structured and weakly-structured imprecise data. *Journal of Intelligent Information Systems* (online first, http://dx.doi.org/10.1007/s10844-006-0014-z).

Hignette, G., Buche, P., Dibie-Bartélemy, J. & Haemmerlé, O. (2005). Fuzzy semantic annotation of XML documents. *Proceedings of the Conference on Advanced Information Systems Engineering Workshop DisWeb 2005* (pp. 319-332). Porto, Portugal.

Ichikawa, T., & Hirakawa, M. (1986). ARES: A Relational Database with the Capability of Performing Flexible Interpretation of Queries. *IEEE Transactions Software Eng. 12*(5), 624-634.

Ireland, J., & Moller, A. (2000). Review of international food classification and description. *Journal of Food Composition and Analysis*, *13*(4), 529–538.

Lacroix, M., & Lavency, P. (1987). Preferences; Putting More Knowledge into Queries. in Stocker, P. M. ; Kent, W., & Hammersley, P. (Ed.), *Proceedings of 13th International Conference on Very Large Data Bases* (pp. 217-225). Morgan Kaufmann.

Lipski, W. (1979). On Semantic Issues Connected with Incomplete Information Databases. *ACM Trans. Database Syst.* 4(3), 262-296.

Lipski, W. (1981). On Databases with Incomplete Information. *J. ACM* 28(1), 41-70.

Loiseau, Y., Boughanem, M. & Prade, H. (2005). Evaluation of term-based queries using possibilistic ontologies. *In* Herrera-Viedma, E., Pasi, G., & Crestani F. (Eds) *Soft computing for Information Retrieval on the Web*, Springer-Verlag.

Motro, A. (1984). Query Generalization: A Method for Interpreting Null Answers. *Expert Database Workshop* (pp. 597-616).

Motro, A. (1988). VAGUE: A User Interface to Relational Databases that Permits Vague Queries. *ACM Trans. Inf. Syst.* 6(3), 187-214.

Miyamoto, S., & Nakayama, K. (1986). Fuzzy information retrieval based on a fuzzy pseudo-thesaurus. *IEEE Transactions on Systems, Man and Cybernetics* 16, 278-282.

Prade, H. (1984). Lipski's approach to incomplete information data bases restated and generalized in the setting of Zadeh's possibility theory. *Information Systems* 9(1), 27-42.

Prade, H., & Testemale, C. (1984). Generalizing database relational algebra for the treatment of incomplete or uncertain information and vague queries. *Information Sciences* 34, 115-143.

Rabitti, F., & Savino, P. (1990). Retrieval of Multimedia Documents by Imprecise Query Specification. In Bancilhon, F., Thanos, C., & Tsichritzis, D. (Ed.), *EDBT Lecture Notes in Computer Science, 416*, 203-218. Springer.

Rossazza, J., Dubois, D., & Prade, H. (1998). A hierarchical model of fuzzy classes. De Caluwe, R., (Ed.), *Advances in Fuzzy systems - Applications and Theory, Fuzzy and uncertain object-oriented databases: concepts and models.* vol. 13, 21-61. World Scientific.

Thomopoulos, R., Buche, P. & Haemmerlé, O. (2006). Fuzzy sets defined on a hierarchical domain, *IEEE Transactions on Knowledge and Data Engineering*, *18*(10), 1397-1410.

Ullman, J. D. (1988). *Principles of Database and Knowledge-Base Systems, Volume I* . Computer Science Press.

Zadeh, L. (1965). Fuzzy sets. *Information and Control*, *8*, 338–353.

Zadeh, L. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, *1*, 3–28.

Zadrozny, S., & Kacprzyk, J., (1998). Implementing Fuzzy Querying Via the Internet/WWW: Java Applets, ActiveX Controls and Cookies. in *Flexible Query Answering Systems*, eds. T. Andreasen, H. Christiansen and H.L. Larsen, *Lecture Notes in Artificial Intelligence* (LNAI) 1495, 382-392. Ed. Springer,

KEY TERMS

Flexible Querying: methods for querying a database that enhance standard querying expressiveness in various ways such as the expression of user's preferences, query generalization, etc., in order to facilitate the extraction of relevant data.

Fuzzy Set: a mapping from a universe of discourse – definition domain of the fuzzy set – into the interval [0,1]. The concept of fuzzy set extends the notion of Boolean membership to a set to the notion of degree of membership.

Hierarchy: a set of elements that are partially ordered by the "kind of" relation.

Hierarchical Fuzzy Set: a fuzzy set whose definition domain is a part of a hierarchy.

Ontology: a formalization of the description of a domain knowledge at a conceptual level.

Possibilistic Database: a database that contains ill-known data represented by means of the possibility theory.

Possibility Distribution: a fuzzy set whose semantics represents the possible ordered values of an imprecise datum, only one of these values being the effective – but ill-known – value of the datum.

Query Generalization: an operation that creates, from a given query Q1, a query Q2 such that Q1 is included in Q2, that is, the answers to Q1 are included in the answers to Q2, for any database.

MIEL language: a flexible querying language which permits to express in a given view a conjunctive query. Current implementations have been done under Oracle and Postgresql RDBMS.

MIEL query: a conjunctive query where the selection value associated with a queried attribute is expressed by a fuzzy set representing preferences.