Flexible bipolar querying of uncertain data using an ontology

Patrice Buche, Sébastien Destercke, Valérie Guillard, Ollivier Haemmerlé, Rallou Thomopoulos

Flexible bipolar querying of uncertain data using an ontology

P. Buche, S. Destercke, V. Guillard, O. Haemmerlé, and R. Thomopoulos

Abstract In this paper, we propose an approach to query a database where the user preferences can be bipolar (i.e., express both constraints and wishes about the desired result) and the data stored in the database can be uncertain. Query results are then completely ordered with respect to these bipolar preferences, giving priority to constraints over wishes. Furthermore, we consider user preferences expressed on a domain of values which is not "flat", but contains values that are more specific than others according to the "kind of" relation. These preferences are represented by specific fuzzy sets, called "Hierarchical Fuzzy Sets" and defined over a simple ontology. We propose a use of "Hierarchical Fuzzy Sets" for query enlargement purposes. The approach is illustrated on a real-world problem concerning the selection of optimal packaging material for fresh fruits and vegetables.

Patrice Buche
INRA IATE, 2, place Pierre Viala F-34060 Montpellier Cedex 02 FRANCE
LIRMM/CNRS-UM2, F-34392 Montpellier, France, e-mail: buche@supagro.inra.fr

Sébastien Destercke
CNRS HEUDYASIC, Centre de recherches de Royallieu F-60205 Compiegne Cedex FRANCE e-mail: sebastien.destercke@hds.utc.fr

Valérie Guillard
UM2 IATE, cc 023 Pl. E. Bataillon F-34095 Montpellier FRANCE e-mail: guillard@univ-montp2.fr

Ollivier Haemmerlé
IRIT-Melodi, Université Toulouse le Mirail, 5, Allées Antonio Machado F-31058 Toulouse Cedex 9 FRANCE e-mail: ollivier.haemmerle@univ-tlse2.fr

Rallou Thomopoulos
INRA IATE, 2, place Pierre Viala F-34060 Montpellier Cedex 02 FRANCE
INRIA GraphiK, F-34392 Montpellier, France, e-mail: rallou@supagro.inra.fr

1 Introduction

In some applications, there may be a need to differentiate, within queries, between negative preferences and positive ones. Negative preferences correspond to constraints, as they specify which values or objects have to be rejected (i.e., those that do not satisfy constraints), while positive preferences correspond to wishes, as they specify which objects are more desirable than others (i.e., satisfy user wishes) without rejecting those that do not meet the wishes. Indeed, while the first type of preferences should be satisfied by query results, satisfying the second type of preferences can be considered as optional, as the user does not consider them to be necessary requirements.

Also, preferences may be expressed over elements organized into a hierarchy rather than on a 'flat' domain. This kind of hierarchy is typically modeled as a simple ontology in which concepts are partially-ordered by the 'kind of' relation. Considering these two extensions (i.e., allowing bipolar preferences expressed over hierarchies) answers a bipolar query enlargement purpose, as the resulting query will send back more results than classical bipolar ones.

Finally, there may be uncertainty in the available data, and there is a need to integrate this uncertainty in the query processing. In this paper, we propose to consider these three problems in a common framework, using the notion of bipolar information and of fuzzy pattern matching.

The notions of bipolar preferences and of bipolar information in general have recently received increasing attention [1, 21]. Roughly speaking, information is said to be bipolar when there is a positive and a negative part of the information. These negative and positive parts of the information may have different natures, and should therefore be processed in parallel, and not as a single piece of information. This kind of bipolarity [11], coined as asymmetric, is the one we are concerned with. For example, we may feel both positive and negative about something, without being able to fuse these two feelings in a unique one (for example, eating ice cream gives a gustative pleasure, but one can also feel guilty about it).

In the case of database queries, asymmetric bipolarity is useful to distinguish negative preferences or constraints (i.e. criteria that a good answer **must** satisfy) from positive preferences or wishes (i.e. criteria that a good answer **should** satisfy, if possible). For example, in the query "a new car not too expensive and if possible red", "not too expensive" is clearly a requirement while "red" merely expresses a wish.

Some preliminary studies of this work have already been published in [14] and [28]. In this paper, we provide a synthetic overview of a method to treat bipolar preferences in databases where data can be uncertain and expressed on a hierarchical domain. In particular, this method uses the bipolar nature of preferences to induce an (pre-)ordering between query results, so that priority is given to those instances that are the most likely to satisfy all expressed constraints and wishes. Section 2 describes the method, while Section 3 illustrates the approach on a use case coming from a new decision support system (DSS) currently developed in IATE laboratory where a (industrial/researcher) user wants to select a packaging material that best

suits his/her needs. Finally, we give some elements of comparison with previous works in Section 4.

2 Method

This section first recalls some basic tools that will be used in the method, before describing the method itself.

2.1 Preliminaries: fuzzy pattern matching

In this paper, we use fuzzy sets [33] to represent preferences in our queries and possibility distributions [18] to represent uncertainty in the data. A normalized fuzzy set μ over a variable X assuming its value on D_X is a mapping $\mu : D_X \to [0,1]$ with at least one $x \in D_X$ such that $\mu(x) = 1$. Here, we assume that D_X is either a finite set of elements (e.g., the colour of a car), possibly partially ordered by the 'kind of' relation (see Section 2.3), or a subset of the real line (e.g., the maximal speed of a car).

Here, fuzzy sets are used to express preferences provided by a user in a query. That is, for a given variable X, the fuzzy value $\mu(x)$ expresses to what degree the value x satisfies the preference represented by μ, with $\mu(x) = 1$ meaning that the preference is fully satisfied and $\mu(x) = 0$ that it is completely unsatisfied.

Example 1. Consider again our car example "a new car not too expensive and if possible red". Assume the user has specified that "not too expensive" means that any price over 18,000 $ is unacceptable, while any price lower than 14,000 $ can be considered as totally satisfactory. The corresponding preference is represented by the fuzzy set μ_{NTE} in Figure 1. Given this representation, we have, for example, that a price of 15,000 $ fulfils the user preferences at a degree $\mu_{NTE}(15,000) = 0.75$.

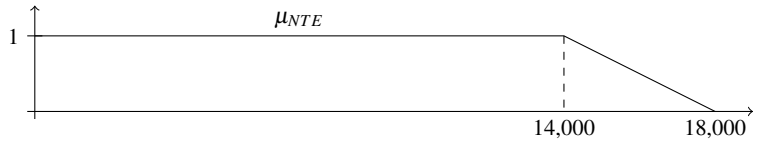

Fig. 1 Fuzzy set μ_{NTE} describing "Not Too Expensive"

Possibility distributions, on the other hand, are simple uncertainty representations allowing to model the ill-known value of some variable. A possibility distribution π over a variable X is also a mapping $\pi : D_X \to [0,1]$ with at least one $x \in D_X$ such that $\pi(x) = 1$. They are therefore equivalent to fuzzy sets from a formal point of

view, but possess different semantics. Indeed, they describe our knowledge about the potential value of X. Two measures or set-functions can be derived from a possibility distribution, namely the necessity and possibility measures, which are such that, for every event $A \subseteq D_X$,

$$\Pi(A) = \sup_{x \in A} \pi(x); \quad N(A) = \inf_{x \in A^c} (1 - \pi(x)) = 1 - \Pi(A^c),$$

where $\Pi(A)$ and $N(A)$ express to what extent it is respectively plausible and certain that the actual value of X lies in A.

Note that possibility distributions can model both precisely known values ($X = x$ corresponds to the distribution $\pi(x) = 1$ and zero everywhere else) and set-valued variables ($X \in A$ corresponds to the distribution $\pi(x) = 1$ if $x \in A$, zero otherwise). In the same way, fuzzy sets can model crisp preferences (i.e., those used in classical queries).

In the rest of the paper, we consider that each query (or preference) P on an attribute X assuming its value on D_X is expressed by a fuzzy set μ_P (possibly degenerated into a crisp preference). Our knowledge D about the attribute value for a particular tuple is given by a possibility distribution π_D (also possibly degenerated in a crisp set). Our knowledge about the imprecise evaluation of P given uncertainty D is summarised by the following lower and upper values [19, 18]:

$$\begin{aligned} \Pi(P;D) &= \sup_{x \in D_X} \min(\mu_P(x), \pi_D(x)), \\ N(P;D) &= \inf_{x \in D_X} \max(\mu_P(x), 1 - \pi_D(x)). \end{aligned} \tag{1}$$

In the following, we will speak of evaluations of a fuzzy preference when talking about the interval $[N(P;D), \Pi(P;D)]$.

Example 2. Consider the preference of Example 1, and a car for which the price is known to belong to the interval $[14,500; 16,000]$, with $15,500$ the most likely value. Figure 2 illustrates both the preference and the knowledge about the price. From this information, we have (using Eq. (1)) that

$$\Pi(P;D) = 0.7 \quad \text{and} \quad N(P;D) = 0.55.$$

2.2 Notations and problem

The problem we consider is as follows: we assume that we have a database consisting of a set $\mathscr{T}$ of T objects o_t, $t = 1, \ldots, T$, with each object taking its values on the Cartesian product $\times_{i=1}^{N} D_{X_i}$ of N domains $D_{X_1}, \ldots, D_{X_N}$. An object o_t is here described by a set of N possibility distributions π_t^i, $i = 1, \ldots, N$, where $\pi_t^i : D_{X_i} \to [0,1]$ is the possibility distribution describing our knowledge about the value of the i^{th}

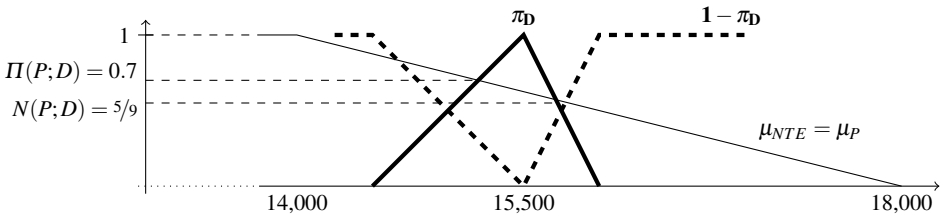

Fig. 2 Evaluation of a fuzzy preference with uncertain data.

attribute of object t. When D_{X_i} is finite, its elements are partially ordered in an ontology according to the 'kind of' relation (classical finite sets are retrieved when all elements are incomparable w.r.t. this order, see Section 2.3). We also assume that the user provides the following information:

- a set $\mathscr{C} = \{C_1^{i_1}, \ldots, C_{N_c}^{i_{N_c}}\}$ of N_c constraints ($N_c \leq N$) to be satisfied by the retrieved objects, where $C_j^{i_j} : D_{X_{i_j}} \rightarrow [0,1]$ is a normalised fuzzy set defined on the attribute i_j ($1 \leq i_j \leq N$).
- a set $\mathscr{W} = \{W_1^{i_1}, \ldots, W_{N_w}^{i_{N_w}}\}$ of N_w wishes ($N_w \leq N$) that the retrieved objects should satisfy if possible, where $W_j^{i_j} : D_{X_{i_j}} \rightarrow [0,1]$ is a normalised fuzzy set defined on the attribute i_j ($1 \leq i_j \leq N$).
- complete pre-orderings $\leq_c$ and $\leq_w$ between the constraints to be satisfied and between the wishes, respectively. These pre-orderings take account of the fact that some constraints may be considered as more important to satisfy than others (and similarly for wishes). In the following, we denote by $\mathscr{C}_{(i)}$ (resp. $\mathscr{W}_{(i)}$) the constraints (resp. the wishes) that have rank i w.r.t. to the pre-ordering[1] $\leq_c$ (resp. $\leq_w$). We denote by $|\leq_c|$ and $|\leq_w|$ the total number of ranks (i.e., of equivalence classes) induced by the two orderings.

Note that constraints and wishes may well be defined on the same attribute. For example, having an acceptable price may be considered as a constraint, but since a lower price (all other things being equal) is always preferable, lowering the price may become a wish for prices lower than completely satisfying prices (in Example 1, one can define a wish that would start from $14,000$ \$).

The problem we consider now is how to retrieve from a set $\mathscr{T}$ of objects, those that primarily satisfy the constraints, and among the latter, those that fulfill the most wishes. Of course, the querying approach has to take account of the bipolar nature of the information, of the possible uncertainty in the data, and of the user's preferences among the constraints and wishes. The next section presents how user preferences are handled when defined over a domain of elements partially ordered by the "kind of" relation. In this latter case, a special kind of fuzzy sets, called *hierarchical fuzzy sets*, will be used.

[1] Note that since $\leq_c$ and $\leq_w$ are complete pre-orderings, each constraint/wish has a well-defined rank.

2.3 Fuzzy sets defined on a hierarchical domain

The notion of hierarchical fuzzy set rose from our need to express fuzzy values in the case where elements receiving a membership value are part of an ontology domain (e.g., packaging material components). First (Section 2.3.1), a fuzzy set is created directly by the user and defined on a part of the hierarchy (only some elements are given membership values). Second, for reasons explained in Section 2.3.2, we extend the fuzzy set to the whole hierarchy, thus obtaining the *closure* of the fuzzy set. Section 2.3.3 defines how we extend the evaluation of fuzzy preferences when classic fuzzy sets are extended to hierarchical fuzzy sets.

2.3.1 Presentation

The definition domains of the fuzzy sets that we define below are subsets of hierarchies composed of elements partially ordered by the "kind of" relation, i.e. they are defined over a subset $B \subseteq D_X$ of the domain of attribute X. An element $x \in D_X$ is more general than an element $x' \in D_X$ (denoted $x' \prec x$), if x' is a predecessor of x in the partial order induced by the "kind of" relation (denoted $\prec$) of the hierarchy. An example of such a hierarchy is given in Figure 3. A hierarchical fuzzy set is then defined as follows.

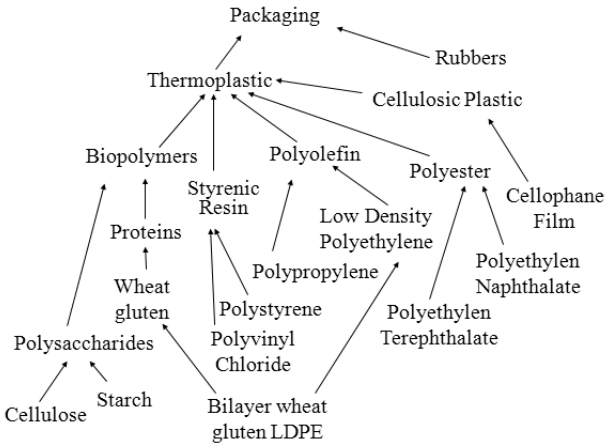

Fig. 3 Example of a hierarchy

Definition 1 *A* **hierarchical fuzzy set** *is a fuzzy set whose definition domain $B \subseteq D_X$ is a subset of the elements of a finite hierarchy partially ordered by the "kind of" relation $\prec$.*

For example, the fuzzy sets *HFS1* and *HFS2* represented in Figure 4 conform to Definition 1. Their definition domains are subsets of the hierarchy given in Figure 3.

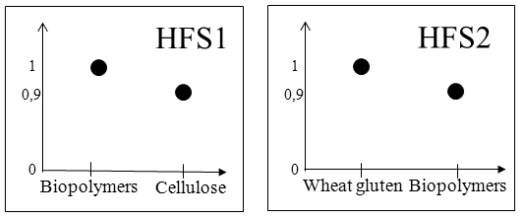

Fig. 4 Fuzzy sets *HFS1* and *HFS2*

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

For example, if there is particular interest in wheat gluten because the user is studying the properties of wheat chain by-products to design packaging, but also wants to retrieve complementary information about other kinds of biopolymers, these preferences can be expressed using for instance the following fuzzy set[2]: $\{(Wheatgluten,1),(Biopolymers,0.9)\}$. In this example, the element *Wheat gluten* has a larger degree than the more general element *Biopolymers*, which corresponds to a semantic of reinforcement for *Wheat gluten* compared to *Biopolymers*. On the contrary, if the user is interested in all kinds of biopolymers to design packaging, but to a lesser extent in *Cellulose* because of its higher value to make bioethanol rather than packaging, the preferences can be expressed using the following fuzzy set: $\{(Biopolymers,1),(Cellulose,0.9)\}$. In this case, the element *Cellulose* has a smaller degree than the more general element *Biopolymers*, which corresponds to a semantic of restriction for *Cellulose* compared to *Biopolymers*.

2.3.2 Closure of a hierarchical fuzzy set

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

- the first one is semantic. Let $\{(Polysaccharides,1),(Biopolymers,0.9)\}$ 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 *Polysaccharides* and *Biopolymers*. For instance, it can be deduced that the user does not

[2] Here, we adopt the usual notation (x,y) for specifying fuzzy sets over symbolic variables, where (x,y) means that modality x has membership value y.

expect results concerning packagings like *Rubber* or *Polyolefin*, even if the degree 0 has not explicitly been associated with these packagings. It is also possible to assume that any kind of *Polysaccharides* (*Cellulose* and *Starch* for example) interests the user with the degree 1;

- the second one is operational. The problem rising from Definition 1 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 for example Eq. (1)). For instance, $\{(Wheat\ gluten,1),(Biopolymers,0.9)\}$ and $\{(Biopolymers,1),(Cellulose,0.9)\}$ are defined on two different subsets of the hierarchy of Figure 3, respectively $\{Wheat\ gluten, Biopolymers\}$ and $\{Biopolymers, Cellulose\}$, and thus are not comparable.

From these remarks can be defined the concept of *closure* of a hierarchical fuzzy set, which is a developed form of the hierarchical fuzzy set defined on the whole hierarchy. The closure of a hierarchical fuzzy set is computed by propagating the degree of an element according to the "kind of" relation: the degree associated with an element is propagated to its sub-elements (more specific elements) in the hierarchy, provided the latter have no degree yet. For instance, in a query, if the user is interested in the element *Biopolymers*, we consider that all kinds of *Biopolymers* – *Polysaccharides*, *Proteins*, etc. – are of interest. On the other hand, we consider that the super-elements (more general elements) of *Biopolymers* in the hierarchy – *Thermoplastic*, *Packaging*, ...– are too general to be relevant for the user's query.

Definition 2 *Let F be a hierarchical fuzzy set defined on a subset B of the elements of a hierarchy D_X. Its membership function is denoted μ_F. The closure$^+$ and the closure$_-$ of F, denoted $\mu_{clos^+(F)}$ and $\mu_{clos_-(F)}$, are two hierarchical fuzzy sets defined on the whole set of elements D_X.*

For each element x of D_X, let $E_x = \{x_1,\dots,x_n\}$ be the set of the smallest super-elements of x in B, i.e. elements such that for any $i = 1,\dots,n$, $x \preceq x_i$ ($x \in E_x$ if $x \in B$) and there is no $x' \in B$ such that $x \prec x' \prec x_i$. Then:

- *if E_x is not empty,*

$$\mu_{clos^+(F)}(x) = \max_{1\le i\le n}(\mu_F(x_i)) \tag{2}$$

and

$$\mu_{clos_-(F)}(x) = \min_{1\le i\le n}(\mu_F(x_i)); \tag{3}$$

- *otherwise $\mu_{clos^+(F)}(x) = \mu_{clos_-(F)}(x) = 0$.*

In other words, the *closure*$^+$ and the *closure*$_-$ of a hierarchical fuzzy set F are built according to the following rules. For each element x of D_X:

1. if $x \in B$, then x keeps the same degree in both closures of F, i.e., $\mu_{clos^+(F)}(x) = \mu_{clos_-(F)}(x) = \mu_F(x)$ (case where $E_x = \{x\}$);

2. if E_x has a unique smallest super-element x_1 in B, then the degree associated with x_1 is propagated to x in both closures of F, i.e., $\mu_{clos^+(F)}(x) = \mu_{clos_-(F)}(x) = \mu_F(x_1)$ (case where $E_x = \{x_1\}$ with $x \prec x_1$);
3. if x has several smallest super-elements $\{x_1, \ldots, x_n\}$ in B, with different degrees, the proposition made in Definition 2 consists in choosing the maximal degree associated with $x_1, \ldots, x_n$ in the $closure^+$, and the minimal degree in the $closure_-$;
4. all the other elements of D_X, i.e., those that are more general than, or not comparable with the elements of B according to $\prec$ are considered as non-relevant. The degree 0 is associated with them (case where $E_x = \emptyset$).

Example 1 *Figure 5 shows the two closures of the hierarchical fuzzy set* $\{(Wheat\ gluten, 1), (Biopolymers, 0.8), (Cellulose, 0.3), (LowDensityPolyethylene, 0.2)\}$.

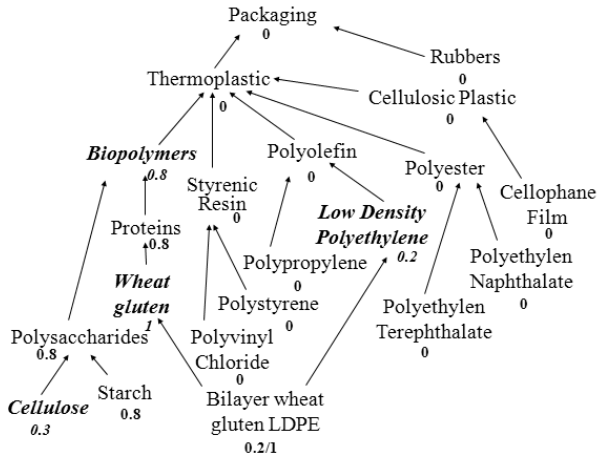

Fig. 5 Closures of a hierarchical fuzzy set: $closure^+$ and $closure_-$ only differ for the element *Bilayer wheat gluten LDPE* for which $\mu_{clos^+(F)}(x) = 1$ and $\mu_{clos_-(F)}(x) = 0.2$.

The use of both a permissive ($closure^+$) and a restrictive ($closure_-$) closure is due to the bipolar nature of the preferences involved. In the case of a wish, Equation (2) ensures a semantic of reinforcement by the use of the max operator (i.e., an element outside B is at least as desirable as its most desirable super-element in B), while the use of min operator in Equation (3) ensures a semantic of restriction for constraints (i.e., an element outside B is at most as desirable as its least desirable super-element in B).

2.3.3 Pattern matching for hierarchical fuzzy sets

Using the concept of closure, all fuzzy sets defined on a given hierarchy can be extended to the same definition domain (the whole hierarchy D_X) and thus can be compared using the classical comparisons and operations between fuzzy sets (e.g., those presented in Equation (1)).

Similarly to preferences, our knowledge about data will usually be expressed on a subset B of D_X, here by a possibility distribution π (note that here, we assume that the ontology structure and concepts are certain, only the actual value of some data on this ontology is uncertain). Computing the closure of π over D_X is slightly different, as we do not consider bipolarity in the information (only negative information in the form of π is given) and as the semantic of possibility distributions is different.

Let us define, for an element $x \in D_X$, the set $\underline{E}(x) = \{y_1, \ldots, y_m\}$ of the biggest sub-elements of x in B, i.e. elements such that for any $i = 1, \ldots, m$, $y_i \prec x$ and there is no $y \in B$ such that $y_i \prec y \prec x$. The closure $clos(\pi)(x)$ of π is defined as follows:

- if $x \in B$, then $clos(\pi)(x) = \pi(x)$;
- if $E_x = \{x_1, \ldots, x_n\}$ is not empty, then $clos(\pi)(x) = \max_{1 \leq i \leq n}(\pi(x_i))$;
- if E_x is empty and $\underline{E}(x) = \{y_1, \ldots, y_m\}$ is not, then $clos(\pi)(x) = \max_{1 \leq i \leq m}(\pi(y_i))$;
- else $clos(\pi)(x) = 0$.

This procedure may give quite imprecise possibilities, but it corresponds to the desire not to miss any interesting data. It is also consistent with usual procedures modifying uncertainty models in the case of refinement or coarsening of an initial non-hierarchical space (in the example of Figure 4, *Biopolymers* can be seen as a coarsening of the elements *Polysaccahrides*, *Proteins* and as an element of the refinement of *Thermoplastic*).

Definition 3 *Let π and F be two hierarchical fuzzy sets defined on the same hierarchy, respectively defining some knowledge about the variable value and some preferences about these values. Then:*

1. *the possibility degree of matching between π and F a positive preference (resp. a negative one) $\Pi(\pi; F)$ is defined as*
   $\Pi(clos(\pi); clos^+(F)) = \sup_{x \in D_X} \min(clos(\pi)(x), \mu_{clos^+(F)}(x))$
   (resp. $\Pi(clos(\pi); clos_-(F)) = \sup_{x \in D_X} \min(clos(\pi)(x), \mu_{clos_-(F)}(x))$);
2. *the necessity degree of matching between π and F a positive preference (resp. a negative one) , $N(\pi; F)$, is defined as*
   $N(clos(\pi); clos^+(F)) = \inf_{x \in D_X} \max(\mu_{clos^+(F)}(x), 1 - clos(\pi)(x))$
   (resp. $N(clos(\pi); clos_-(F) = \inf_{x \in D_X} \max(\mu_{clos_-(F)}(x), 1 - clos(\pi)(x)))$).

We will see in the next section how bipolar preferences, including positive and negative preferences defined by hierarchical fuzzy sets, are used to query uncertain data.

2.4 From bipolar querying with imprecise data to answer ordering

Previous sections have dealt with the problem of modeling and specifying bipolar preferences and uncertain data over hierarchies defined by simple ontologies. We now detail methods allowing the retrieval and ordering of answers from these preferences and propose some elements explaining this ordering to the users.

2.4.1 Ordering answers

As underlined by [1], when bipolar information concerns preferences, satisfying constraints should be a primary aim, while satisfying wishes remains secondary. To do this, a solution is to first retain all the objects that may satisfy the constraints, order them w.r.t. the degree to which they satisfy these constraints, and then refine this order by using degrees to which objects satisfy those wishes. If the user has specified preferences between constraints (resp. between wishes)[3], we also provide a means to take these preferences into account.

We propose, for constraints $\mathscr{C}_{(i)}$ of rank i, to summarise the way an object o_t satisfies these constraints by an aggregated interval $[N_t^{(i)}, \Pi_t^{(i)}]_c$ given by the following formula:

$$N_t^{(i)} = \top_{C_k^{j_k} \in \mathscr{C}_{(i)}} N(C_k^{j_k}; \pi_t^{j_k}), \quad \text{and} \quad \Pi_t^{(i)} = \top_{C_k^{j_k} \in \mathscr{C}_{(i)}} \Pi(C_k^{j_k}; \pi_t^{j_k}), \tag{4}$$

with $N(C_k^{j_k}; \pi_t^{j_k})$, $\Pi(C_k^{j_k}; \pi_t^{j_k})$ given by Eq. (1) or definition 3 if the domain associated with $C_k^{j_k}$ is a hierarchy, and $\top$ a t-norm[4] [23]. T-norms are conjunctive aggregation operators and are chosen here for the reason that ALL constraints have to be satisfied simultaneously. Here, we take $\top = \min$, the minimum operator.

Similarly, we build, for each $\mathscr{W}_{(i)}$ and object o_t satisfying the constraints, the interval $[N_t^{(i)}, \Pi_t^{(i)}]_w$, such that

$$N_t^{(i)} = \oplus_{W_k^{j_k} \in \mathscr{W}_{(i)}} N(W_k^{j_k}; \pi_t^{j_k}), \quad \text{and} \quad \Pi_t^{(i)} = \oplus_{W_k^{j_k} \in \mathscr{W}_{(i)}} \Pi(W_k^{j_k}; \pi_t^{j_k}), \tag{5}$$

where $\oplus$ is an aggregation operator that can be a t-norm, an averaging operator such as an OWA [32] operator or a t-conorm, depending on the behaviour we want to adopt w.r.t. the satisfaction of wishes. Indeed, since satisfying wishes is not compulsory, we can adopt different attitudes [1]. For instance, using a t-conorm means that we are satisfied as soon as one wish is fulfilled, while using a t-norm means that we still want all the wishes to be satisfied to increase our overall satisfaction. In this paper, we consider the latter case, and will take $\oplus = \min$.

[3] No preferences means here that all constraints (or wishes) have the same rank, i.e., are of equal importance.

[4] A T-norm $\top : [0, 1]^2$ to $[0, 1]$ is an associative, commutative operator that has 1 for neutral element and 0 for absorbing element.

It is necessary then to order objects that could satisfy the constraints and some wishes, according to the previous evaluations. To do so, we will use a lexicographic order and a dominance relation $\leq_{[N^{(i)},\Pi^{(i)}]}$ between objects such that, for two interval evaluations $[N_t^{(i)},\Pi_t^{(i)}]$, $[N_{t'}^{(i)},\Pi_{t'}^{(i)}]$ related to objects o_t and $o_{t'}$ and to a group of constraints $\mathscr{C}_{(i)}$ or a group of wishes $\mathscr{W}_{(i)}$, $o_t \leq_{[N^{(i)},\Pi^{(i)}]} o_{t'}$ if $N_t^{(i)} \leq N_{t'}^{(i)}$ and $\Pi_t^{(i)} \leq \Pi_{t'}^{(i)}$ (with $o_t <_{[N^{(i)},\Pi^{(i)}]} o_{t'}$ if at least one inequality is strict). That is, an object $o_{t'}$ dominates another one o_t if its satisfaction bounds are pair-wise higher than the satisfaction bounds of o_t. The lexicographic order is then used to take account of the difference between negative and positive preferences and of the orderings $\leq_c$ and $\leq_w$ (i.e. objects are first ordered using constraints of rank one, then two, ...).

Note that, although $\leq_{[N^{(i)},\Pi^{(i)}]}$ is a partial order, we will induce from it a complete pre-order that refines $\leq_{[N^{(i)},\Pi^{(i)}]}$, for the reason that users are more at ease with complete orderings. However, we will use the fact that $\leq_{[N^{(i)},\Pi^{(i)}]}$ is a partial order to differentiate negative and positive preferences. The procedure consists in building iteratively an ordered partition $\{\mathscr{T}_0,\ldots,\mathscr{T}_M\}$ of $\mathscr{T}$. Rejected objects that do not satisfy all constraints are put in $\mathscr{T}_0$, while objects in $\mathscr{T}_M$ can be considered as the most satisfactory.

In a preliminary step, Algorithm 1 rejects those objects of $\mathscr{T}$ that do not at all satisfy some constraints.

Algorithm 1: Determination of $\mathscr{T}_0$, the set of rejected objects which will not belong to the query result

Input: The set of objects $\mathscr{T} = \{o_1,\ldots,o_T\}$
Output: Ordered partition$\{\mathscr{T}_0, \mathscr{T} \setminus \mathscr{T}_0\}$ of $\mathscr{T}$

```
1 T_0 = ∅;
2 foreach o_t ∈ T do
3     if Π_t^(i) = 0 for some i = 1,...,| ≤_c | then
4         T_0 = T_0 ∪ {o_t} ;
```

Algorithm 2 describes how results are ordered within a subset of $\mathscr{T} \setminus \mathscr{T}_0$ (called $\mathscr{T}'$), according to constraints of a given rank. The whole procedure consists in building a partition of $\mathscr{T} \setminus \mathscr{T}_0$. The partition is refined iteratively by applying, at every rank i ($i \in [1,|\leq_c|]$), Algorithm 2 within each equivalence class of objects obtained at the previous rank $i-1$. When $i=1$, the unique initial equivalence class $\mathscr{T}'$ is $\mathscr{T} \setminus \mathscr{T}_0$. In every run of Algorithm 2, equivalence classes $\{\mathscr{T}'_1,\ldots,\mathscr{T}'_n\}$ are incrementally built, starting from the worst ($\mathscr{T}'_1$) and ending with the best ($\mathscr{T}'_n$). At each step, the objects included and then suppressed from $\mathscr{T}'$ are those objects that do not dominate other objects (line 4), in the sense of $\leq_{[N^{(i)},\Pi^{(i)}]}$. This means that objects with imprecise evaluations (i.e., $[N_t^{(i)},\Pi_t^{(i)}]$ with larger width) will be in lower classes, along with objects having low evaluations (i.e., low $\Pi_t^{(i)}$). This corresponds to a pessimistic attitude towards imprecision, since imprecise evaluations are as-

sociated with poorly satisfying objects. Such an attitude is coherent with negative preferences, as the possibility of not satisfying a constraint is penalised.

Algorithm 2: Query result ordering for constraints of rank (i)

Input: $\mathscr{T}' \subseteq \mathscr{T} \setminus \mathscr{T}_0$ with $\mathscr{T}'$ an element of the partition issued from rank $(i-1)$, $[N_t^{(i)}, \Pi_t^{(i)}]_c$ for each $o_t \in \mathscr{T}'$
Output: Ordered partition$\{\mathscr{T}'_1, \ldots, \mathscr{T}'_n\}$ of $\mathscr{T}'$

```
1 K = T'; j=1;
2 while K ≠ ∅ do
3     foreach o_t ∈ K do
4         if ∄o_j ∈ K s.t. o_t ≥_[N^(i),Π^(i)] o_j then
5             Put o_t in T'_j
6     K = K \ T'_j ;
7     j = j + 1;
```

After having applied Algorithm 1 once and Algorithm 2 $|\leq_c|$ times, the complete pre-order is further refined according to wishes by using Algorithm 3. There are two main differences with Algorithm 1 and Algorithm 2. First, no objects are rejected, as we are dealing with positive preferences (satisfying them is not compulsory). Second, we start here from the best equivalence class and finish with the worst[5], and at each step the objects included and then suppressed from $\mathscr{T}'$ are those objects that are not dominated by other objects (line 7), in the sense of $\leq_{[N^{(i)},\Pi^{(i)}]}$. Contrary to Algorithm 2, objects with imprecise evaluations will be in the upper classes. This corresponds to an optimistic attitude towards imprecision, which is coherent with positive preferences, as it promotes the possibility of satisfying more wishes. Note that inconsistency problems between positive and negative information [1] do not occur here, since constraints and wishes are treated separately and lexicographically.

The knowledge uncertainty is fully acknowledged through the use of the partial order $\leq_{[N^{(i)},\Pi^{(i)}]}$ (which considers both end-points of intervals $[N^{(i)}, \Pi^{(i)}]$) in algorithms 2 and 3 which allow us to make a clear distinction in the treatment of negative and positive aspects of bipolar preferences. However, a possible drawback for huge databases is the complexity that the use of these algorithms represents. Indeed, each run of Algorithms 2 and 3 requires comparing each object with all the other objects of a same equivalence class. If n objects have to be ordered, then in the worst case $(|\leq_c|+|\leq_w|)n^2$ comparisons are performed, assuming that no object strictly dominates another for any rank of constraints or wishes. In the best case, i.e. when objects are completely ordered after a first run, n^2 comparisons have to be made. It must be noted that n is reduced to $|\mathscr{T} \setminus \mathscr{T}_0|$ thanks to Algorithm 1. Such complexities are quite acceptable for most databases, but could be problematic for databases counting billions of objects. In such a case, it is possible to use other propositions presenting a lower complexity where object ordering is solely based on one of the

[5] The shift loop (Lines 3-5) is there to keep the same indexing of subsets $\mathscr{T}_j$

Algorithm 3: Query result ordering for wishes of rank (i)

Input: $\mathscr{T}' \subseteq \mathscr{T} \setminus \mathscr{T}_0$ with $\mathscr{T}'$ an element of a partition issued from rank $(i-1)$, $[N_t^{(i)}, \Pi_t^{(i)}]_w$ for each $o_t \in \mathscr{T}'$
Output: Ordered partition$\{\mathscr{T}'_1, \ldots, \mathscr{T}'_m\}$ of $\mathscr{T}'$

$K = \mathscr{T}'$; j=0;
while $K \neq \emptyset$ **do**
  for $i = j, \ldots, 1$ *(skip if* $j = 0$*)* **do**
    $\mathscr{T}'_{j+1} = \mathscr{T}'_j$
  $\mathscr{T}'_1 = \emptyset$;
  foreach $o_t \in K$ **do**
    if $\nexists o_j \in K$ *s.t.* $o_t \leq_{[N^{(i)}, \Pi^{(i)}]} o_j$ **then**
      Put o_t in $\mathscr{T}'_1$
  $K = K \setminus \mathscr{T}'_1$;
  $j = j + 1$;

two numbers $N^{(i)}$ or $\Pi^{(i)}$ [19]. However, using orderings based on single numbers means that the imprecision in $[N^{(i)}, \Pi^{(i)}]$ is not fully taken into account and some of the information contained in the interval is lost.

Example 3. Let us consider a set $\mathscr{T}$ of six objects $o_1, \ldots, o_6$, two ranks of constraints and only one rank of wish. The intervals $[N_t^{(i)}, \Pi_t^{(i)}]_c$ $(i = \{1,2\})$ and $[N_t^{(1)}, \Pi_t^{(1)}]_w$ are summarized in Table 1.

	$[N_t^{(1)}, \Pi_t^{(1)}]_c$	$[N_t^{(2)}, \Pi_t^{(2)}]_c$	$[N_t^{(1)}, \Pi_t^{(1)}]_w$
o_1	$[0.1, 0.4]$	$[0.8, 1]$	$[1, 1]$
o_2	$[0.5, 0.8]$	$[0.5, 0.6]$	$[0.6, 0.9]$
o_3	$[0.3, 1]$	$[0.4, 0.8]$	$[0.2, 0.5]$
o_4	$[0.8, 1]$	$[0, 0]$	$[0.5, 0.7]$
o_5	$[1, 1]$	$[0.2, 0.4]$	$[0, 0]$
o_6	$[0, 1]$	$[0.6, 0.9]$	$[0.3, 0.7]$

Table 1 Example 3 evaluations for constraints and wishes.

Running Algorithm 1 gives $\mathscr{T}_0 = \{o_4\}$. o_4 is the only rejected object, because $\Pi_4^{(2)} = 0$, even if it satisfies rank one constraints necessarily to a high degree. After a first run of Algorithm 2, we obtain the following partition:

$$\mathscr{T}_0 = \{o_4\} < \mathscr{T}_1 = \{o_1, o_6\} < \mathscr{T}_2 = \{o_2, o_3\} < \mathscr{T}_3 = \{o_5\}.$$

All elements potentially satisfy constraints in $\mathscr{C}_{(1)}$ (although o_6 does not necessarily satisfy them). Note that o_6, for which information is fully imprecise, is at the end of the ordering (whereas it would have been at the front if we used Algorithm 3). Since there are two ranks of constraints, a second run of Algorithm 2 gives

$$\mathscr{T}_0 = \{o_4\} < \mathscr{T}_1 = \{o_6\} < \mathscr{T}_2 = \{o_1\} < \mathscr{T}_3 = \{o_2, o_3\} < \mathscr{T}_4 = \{o_5\}.$$

This second run refined the ordering between o_1 and o_6. Also note that the bad scores of o_5 w.r.t. constraints of rank two do not change its order, due to the constraint preferences and the use of a lexicographic order. Finally, a run of Algorithm 3 gives

$$\mathscr{T}_0 = \{o_4\} < \mathscr{T}_1 = \{o_6\} < \mathscr{T}_2 = \{o_1\} < \mathscr{T}_3 = \{o_3\} < \mathscr{T}_4 = \{o_2\} < \mathscr{T}_5 = \{o_5\}.$$

Note that o_5 is not rejected, since satisfying wishes is not a requirement.

2.4.2 Explaining the ordering

Answers provided by DSS, expert systems or multi-criteria decision making methods can be hard to interpret for end-users. It is therefore useful to provide them with simple and understandable (e.g., expressed in natural language) elements of explanation [24].

We therefore propose such explanations of our ordering. As Algorithms 1- 3 use a lexicographic ordering implicitly based on pair-wise comparisons, such explanations can only concern a single rank of constraint or wish and will therefore remain simple. These explanations can be stored in an $n \times n$ matrix $Expl$ where the element $Expl(\ell,k)$ will contain the explanation of why object ℓ has been judged better/worse than object k. This matrix is somehow anti-symmetric, as the reason $Expl(\ell,k)$ will be the opposite of $Expl(k,\ell)$. Note that we do not need to consider objects in $\mathscr{T}_0$, as such objects will not be part of the answer received by the user.

Consider first Algorithm 2 and assume that we are running it on the ith rank of constraints and that loop of lines 3-4 has just ended for the jth time (i.e., the set $\mathscr{T}'_j$ has just been built). Then, for each o_ℓ, o_k such that $o_\ell \in \mathscr{T}'_j$ and $o_k \in K \setminus \mathscr{T}'_j$, we propose the following explanation in $Expl(\ell,k)$:

- if $N_k^{(i)} > \Pi_\ell^{(i)}$, then $Expl(\ell,k) = \{o_\ell$ is judged worse than o_k because it is certainly worse on constraints of priority i, and they are indistinguishable on more important constraints$\}$;
- else, $Expl(\ell,k) = \{o_\ell$ is judged worse than o_k because it is possibly worse on constraints of priority i, and they are indistinguishable on more important constraints$\}$.

Note that explanations make a distinction between the relation $\geq_{[N^{(i)},\Pi^{(i)}]}$ and the more constraining (but stronger) relation (known as interval dominance) that consists in saying that $o_k > o_\ell$ if and only if $N_k^{(i)} > \Pi_\ell^{(i)}$.

Proposed explanations are similar for Algorithm 3, except that users should be informed that wishes are now considered. Assume that we look at the ith rank of wishes and that loop of lines 3-9 has just finished (the new set $\mathscr{T}'_1$ has just been built). Then, for each o_ℓ, o_k such that $o_\ell \in K \setminus \mathscr{T}'_1$ and $o_k \in \mathscr{T}'_1$, we propose the following explanation in $Expl(\ell,k)$:

- if $N_k^{(i)} > \Pi_\ell^{(i)}$, then $Expl(\ell,k) = \{o_\ell$ is judged worse than o_k because it is certainly worse on wishes of priority i, and both satisfy constraints in an indistinguishable way$\}$;
- else, $Expl(\ell,k) = \{o_\ell$ is judged worse than o_k because it is possibly worse on wishes of priority i, and both satisfy constraints in an indistinguishable way$\}$.

For instance, in Example 3, the element $Expl(6,1)$ would have been "Object 6 is judged worse than Object 1 because it is possibly worse on constraints of priority 2, and they are indistinguishable on more important constraints". In practical applications, the names of attributes concerned by the constraints or wishes separating two objects can be explicitly cited rather than giving ranks, as they will be more meaningful to the user.

A possible inconvenience of this method is that values of Eq. (4) and (5) are aggregated on many attributes, meaning that a detailed explanation on each attribute of rank i cannot be given. Possible solutions to solve this issue are (1) to consider complete orderings for $\leq_c$ and $\leq_w$ (i.e., $|\leq_c| = N_c$ and $|\leq_w| = N_w$) or (2) to use decision strategies not based on aggregated values (e.g., a voting rule on each constraint/wish of the same rank).

3 A new decision support system for food packaging design

In this section, we present a new decision support system (DSS) for fresh fruit and vegetable packaging design in which the flexible bipolar querying approach plays a central role. To the best of our knowledge, only one DSS for fresh fruits and vegetables packaging already exists (see [25]), but it does not take into account the criteria ensuring a sustainable design (a critical issue in food science). Such a sustainable design must satisfy, at least, three kinds of criteria: economic, environmental and societal. An example of the economic aspect may be the cost of the packaging material. Concerning environmental aspects, important criteria are the biodegradability of the package or the optimization of product preservation at ambient temperature (in order to decrease the use of the energy-greedy cold chain). Societal aspects can concern the fact that consumers may reject the use of some additives or of nanotechnology in the packaging material because of the unknown consequences on their health, or more simply they may prefer transparent rather than opaque packaging.

In our DSS, starting from a given fruit or vegetable, the user specifies his/her needs in terms of several criteria (e.g., conservation temperature, transparency, material cost, ...) in order to determine a list of packaging. These types of packaging are ordered according to their degree of satisfaction of the criteria. The bipolar approach gives the user the possibility to specify in a flexible way what criteria must be considered as constraints and what other criteria will be used to refine the ranking of packaging satisfying the constraints. Starting from the user specifications, a flexible bipolar query is executed against a database containing information about packaging materials. This information has been collected from different sources which may be technical descriptions of commercial packaging materials or data extracted from

scientific publications concerning new packaging materials. This information may be uncertain, due to the variability of engineered packaging and the biological variability of vegetables. The bipolar approach proposed in this paper deals with this uncertainty. In Section 3.1, we present the global architecture of the DSS. A use case concerning endive packaging will be presented in Section 3.2.

3.1 Decision support system architecture

Starting from the name of the vegetable/fruit of interest specified by the user (see figure 6), the system scans in the first step the vegetable/fruit database in order to retrieve the O_2 respiration rate (and associated parameters) of the studied vegetable/fruit. In the second step, the optimal O_2 permeance[6] of the targeted packaging is computed thanks to a model of gas exchanges inside the package called PassiveMap (see [12] for more details about the model). In the third step, using the targeted optimal O_2 permeance and the other user requirements about criteria of various types (economical, environmental or societal), a query is executed against the packaging database using the flexible bipolar querying engine, which is the central part of the DSS. A list of packaging materials ordered according to the method presented in the previous sections is finally presented to the user. The use case presented in the next section focuses on the DSS flexible bipolar querying engine.

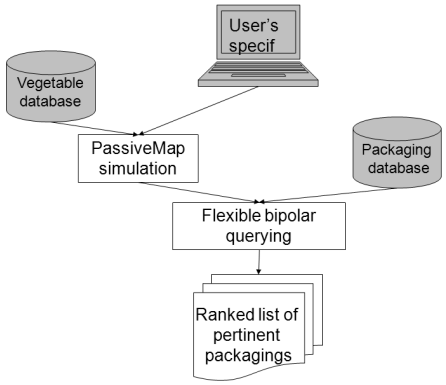

Fig. 6 Global architecture of the DSS

[6] A measure of the ability of a package to conduct gas fluxes.

3.2 Endive packaging use case

In this section, we present a use case of the DSS concerning the choice of a packaging material for endives. The user has to specify a set of parameters needed by the DSS to determine the optimal O_2 permeance of the targeted packaging: the mass of the vegetable (500 grams), the surface, the volume and the thickness of the targeted packaging (respectively 0.14 m^2, 0.002 m^3 and 5e-5 m), the shelf life of the vegetable (7 days) and the storage temperature (20 $^\circ C$). Using the O_2 respiration rate (and associated parameters) retrieved from the vegetable database, an optimal O_2 permeance of 3.65E-11 $mol.m^{-2}.s^{-1}.Pa^{-1}$ is computed. The optimal permeance and the temperature will be considered as criteria to scan the package database.

We consider in this use case that the user is also interested in two other criteria: the biodegradability and the transparency of the package. An extract of the packaging database content is presented in Tables 2 and 3 and will be used to illustrate the flexible bipolar querying process. Note that imprecise data are here reduced to degenerated possibility distributions (given by the $\min - \max$ permeance span), since currently there is no possibilistic uncertainty in the database (however, such uncertainty will be integrated in future evolutions of the DSS including robust design methods [17]).

o_{id}	$PackagingType$	$Permeance_{min}$ $(mol.m^{-2}.s^{-1}.Pa^{-1})$	$Permeance_{max}$ $(mol.m^{-2}.s^{-1}.Pa^{-1})$	$Temperature$ $(^\circ C)$
o_1	$Polyolefin$	$1,29E-13$	$1,29E-13$	23
o_2	$Polyolefin$	$4,05E-11$	$4,05E-11$	23
o_3	$Cellophane$	$1,55E-14$	$1,55E-14$	23
o_4	$Polyolefin$	$1,96E-11$	$2,39E-11$	20
o_5	$Cellulose$	$1,55E-14$	$1,55E-14$	23
o_6	$Polyester$	$4,46E-12$	$4,46E-12$	23
o_7	$Polyolefin$	$1,50E-11$	$1,50E-11$	23
o_8	$Polyester$	$1,55E-13$	$1,55E-13$	23
o_9	$Polystyrene$	$1,03E-12$	$1,03E-12$	23
o_{10}	$Polyester$	$6,23E-12$	$6,23E-12$	23
o_{11}	$Wheatgluten$	$1,55E-11$	$1,67E-11$	25
o_{12}	$PolyVinylChloride$	$7,47E-11$	$7,47E-11$	25

Table 2 Permeance at a given temperature for an extract of the packaging database

We will consider two examples of queries expressed by the user (in the current case, they were given by one of the co-authors, V. Guillard). In the first one, the user specifies one constraint and two wishes. The user first requires the package to be transparent in order to be accepted by the consumer who wants to see the endive through the package. It will be expressed as the first and unique constraint. Concerning his/her wishes, the user would like to maximize the shelf life of the product at an ambient temperature (and consequently to select a packaging whose oxygen permeance is close to the optimal one). It will be expressed as the wishes, here of equal rank.

o_{id}	*PackagingType*	*Transparency*	*Biodegradability*
o_1	*Polyolefin*	*transparent*	*no*
o_2	*Polyolefin*	*transparent*	*no*
o_3	*Cellophane*	*transparent*	*yes*
o_4	*Polyolefin*	*transparent*	*no*
o_5	*Cellulose*	*transparent*	*yes*
o_6	*Polyester*	*transparent*	*yes*
o_7	*Polyolefin*	*transparent*	*no*
o_8	*Polyester*	*translucent*	*yes*
o_9	*Polystyrene*	*translucent*	*no*
o_{10}	*Polyester*	*translucent*	*yes*
o_{11}	*Wheatgluten*	*translucent*	*yes*
o_{12}	*PolyVinylChloride*	*transparent*	*no*

Table 3 Transparency and biodegradability for the same extract of the packaging database

In the second query, the user specifies two constraints and two wishes. To design a sustainable package, the user expresses that the packaging must be biodegradable as a first constraint (rank one) and must be made of renewable resources (i.e. constraint of rank two on the packaging type). Then, the user expresses as first wish that the packaging should be transparent in order to be accepted by the consumer and as second wish that it should maximize the shelf life of the product at an ambient temperature for economic reasons.

As already said in Section 2.1, the user preferences are, for each criterion, expressed by a fuzzy set used as a general formalism which enables the representation of fuzzy, interval or crisp values. Concerning the permeance criterion, 60% of variation is authorized around the optimal value computed by the PassiveMap subsystem, with decreasing degrees of preferences. For the temperature, a total variation of 100% is authorized, with no preference for the different values. The fuzzy sets associated with the permeance and temperature preferences are presented in Figure 7.

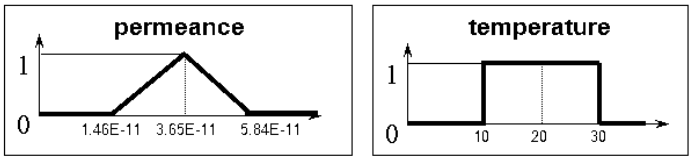

Fig. 7 Preferences for permeance and temperature

The fuzzy set associated with the transparency (resp. biodegradability) criterion is $Pref_{transparency}$={(transparent,1),(translucent,0),(opaque,0)} (resp. the fuzzy set $Pref_{biodegradability}$={(yes,1),(no,0)}). They correspond to crisp requirements provided by the user, as the concept of graded biodegradability made little sense to the user, while translucency is not graded in our current data. The hierarchical fuzzy set associated with the packaging type is $Pref_{packagingType}$={(Biopolymers,1)}. It ex-

presses that the user preferences are for renewable resources but without specifying a specific type of biopolymer.

Using the notations introduced in Section 2.1, the first query is built as follows: $\mathscr{C}_{(1)} = \{Pref_{transparency}\}$ and $\mathscr{W}_{(1)} = \{Pref_{permeance}, Pref_{temperature}\}$.

	$[N_t^{(1)}, \Pi_t^{(1)}]_c$	$[N_t^{(1)}, \Pi_t^{(1)}]_w$
o_1	$[1,1]$	$[0,0]$
o_2	$[1,1]$	$[0,817, 0,817]$
o_3	$[1,1]$	$[0,0]$
o_4	$[1,1]$	$[0,228, 0,427]$
o_5	$[1,1]$	$[0,0]$
o_6	$[1,1]$	$[0,0]$
o_7	$[1,1]$	$[0,021, 0,021]$
o_8	$[0,0]$	$[0,0]$
o_9	$[0,0]$	$[0,0]$
o_{10}	$[0,0]$	$[0,0]$
o_{11}	$[0,0]$	$[0,043, 0,098]$
o_{12}	$[1,1]$	$[0,0]$

Table 4 Evaluations for the constraint and the wishes of the first query.

Let us consider the set $\mathscr{T} = \{o_1, \ldots, o_{12}\}$ of the twelve packages whose characteristics are given in Tables 2 and 3 and whose evaluations for the constraint and wishes of query 1 are given in Table 4 (as the two wishes are of the same rank, they have been aggregated in $[N_t^{(1)}, \Pi_t^{(1)}]_w$ according to Eq. (5)). After running Algorithm 1, we obtain $\mathscr{T}_0 = \{o_8, o_9, o_{10}, o_{11}\}$. After running Algorithm 2 with $\mathscr{C}_{(1)}$, we obtain the following partition:

$$\mathscr{T}_0 = \{o_8, o_9, o_{10}, o_{11}\} < \mathscr{T}_1 = \{o_1, o_2, o_3, o_4, o_5, o_6, o_7, o_{12}\}.$$

After running Algorithm 3 with $\mathscr{W}_{(1)}$, we obtain the following partition:

$$\mathscr{T}_0 = \{o_8, o_9, o_{10}, o_{11}\} < \mathscr{T}_1 = \{o_1, o_3, o_5, o_6, o_{12}\} <$$

$$\mathscr{T}_2 = \{o_7\} < \mathscr{T}_3 = \{o_4\} < \mathscr{T}_4 = \{o_2\}.$$

The second query is built as follows:
$\mathscr{C}_{(1)} = \{Pref_{biodegradability}\}$, $\mathscr{C}_{(2)} = \{Pref_{packagingType}\}$, $\mathscr{W}_{(1)} = \{Pref_{transparency}\}$, $\mathscr{W}_{(2)} = \{Pref_{permeance}, Pref_{temperature}\}$. The first constraint is judged more important than the second one: one wants biodegradable packaging to preserve the environment (first constraint) which is sustainable, thus made of renewable resource (second constraint).The first wish is judged more important than the second one: one wants transparent packaging to fulfill consumers' preferences (first wish) and optimized shelf life of the packed food thanks to a fine control of O_2 permeance (second wish).

Consider again the set $\mathscr{T}$ of packages described in Tables 2 and 3 and whose evaluations for the constraints and the wish of query 2 are given in Table 5. After

	$[N_t^{(1)}, \Pi_t^{(1)}]_c$	$[N_t^{(2)}, \Pi_t^{(2)}]_c$	$[N_t^{(1)}, \Pi_t^{(1)}]_w$	$[N_t^{(2)}, \Pi_t^{(2)}]_w$
o_1	$[0,0]$	$[0,0]$	$[1,1]$	$[0,0]$
o_2	$[0,0]$	$[0,0]$	$[1,1]$	$[0,817, 0,817]$
o_3	$[1,1]$	$[0,0]$	$[1,1]$	$[0,0]$
o_4	$[0,0]$	$[0,0]$	$[1,1]$	$[0,228, 0,427]$
o_5	$[1,1]$	$[1,1]$	$[1,1]$	$[0,0]$
o_6	$[1,1]$	$[0,0]$	$[1,1]$	$[0,0]$
o_7	$[0,0]$	$[0,0]$	$[1,1]$	$[0,021, 0,021]$
o_8	$[1,1]$	$[0,0]$	$[0,0]$	$[0,0]$
o_9	$[0,0]$	$[0,0]$	$[0,0]$	$[0,0]$
o_{10}	$[1,1]$	$[0,0]$	$[0,0]$	$[0,0]$
o_{11}	$[1,1]$	$[1,1]$	$[0,0]$	$[0,043, 0,098]$
o_{12}	$[0,0]$	$[0,0]$	$[1,1]$	$[0,0]$

Table 5 Evaluations for the constraints and the wish of the second query.

running Algorithm 1, we obtain $\mathscr{T}_0 = \{o_1, o_2, o_3, o_4, o_6, o_7, o_8, o_9, o_{10}, o_{12}\}$. Packaging which are not biodegradable have been discarded. Moreover, the hierarchical fuzzy set associated with the packaging type, $Pref_{packagingType}$, permits to express a generic constraint in a simple way: packaging which are not bio-sourced have been discarded too. It must be noticed that the use of a classical fuzzy set for $Pref_{packagingType}$ instead of a hierarchical fuzzy set would have delivered an empty set of answers (all the objects in $\mathscr{T}_0$) after running Algorithm 1. After the first run of Algorithm 2 with $\mathscr{C}_{(1)}$, we obtain the following partition:

$$\mathscr{T}_0 = \{o_1, o_2, o_3, o_4, o_6, o_7, o_8, o_9, o_{10}, o_{12}\} < \mathscr{T}_1 = \{o_5, o_{11}\}.$$

The second run of Algorithm 2 with $\mathscr{C}_{(2)}$ ($[N_t^{(2)}, \Pi_t^{(2)}]_c$) keeps the partition unchanged. After the first run of Algorithm 3 with $\mathscr{W}_{(1)}$, we obtain the following partition:

$$\mathscr{T}_0 = \{o_1, o_2, o_3, o_4, o_6, o_7, o_8, o_9, o_{10}, o_{12}\} < \mathscr{T}_1 = \{o_{11}\} < \mathscr{T}_2 = \{o_5\}.$$

The second run of of Algorithm 3 with $\mathscr{W}_{(2)}$ keeps the partition unchanged.

We can see with the result obtained for the second query, from which only two results are retrieved, that the constraints may be very restrictive compared to the content of the database. In those cases where no answer is found, we have proposed in [14] an approach to provide to the users "best" answers among all the rejected ones (i.e., answers that are the closest to satisfying the constraints).

4 Related works

There exist many works that propose to use fuzzy sets to introduce graded preferences and possibility distributions to handle uncertainty in databases. Our work can be related to these two complementary propositions.

The fuzzy set framework has been shown to be a sound scientific choice to model flexible queries [3]. It is a natural way of representing the notion of preference using a gradual scale. In [5], the semantics of a language called SQLf has been proposed to extend the well-known SQL language by introducing fuzzy predicates processed on crisp information. Other approaches have also been proposed to introduce preferences into queries in the database community [8, 22, 13]. However, in all these approaches, preferences are of the same nature. It is only recently that the concept of bipolarity and its potential use in flexible queries has been studied [20, 21]. This extended approach discriminates between two types of preferences, one acting as compulsory constraints, the other acting as optional wishes. Several works have recently been proposed in order to extend the relational algebra with this concept of bipolarity [6, 7] or to propose a framework to deal with bipolarity in regular relational databases [31]. It should be noticed that, to the best of our knowledge, the introduction in bipolar flexible querying of preferences expressed on a hierarchical domain is an original point of our approach.

The second proposition is to use possibility distributions (whose formalism is mathematically equivalent to that of fuzzy set) to represent uncertain values [34]. Several authors have developed this approach in the context of databases [26, 27, 4, 2, 29, 10]. To the best of our knowledge, the only other work dealing with the concept of bipolarity in flexible querying of databases including uncertain values, outside some research perspectives in [21], is that of G. De Tré *et al.* [30]. However, they deal with a different aspect of bipolar preferences, as they mainly consider the use of interval-valued fuzzy sets (or similar models) to cope with imprecisely defined preferences, and treat positive and negative preferences in a common framework, rather than considering them separately (as we do here).

5 Conclusion and perspectives

In this paper, we have introduced a method for querying a database when preferences are bipolar (contains both constraints and wishes), data are uncertain and can be expressed on a hierarchical domain. We use fuzzy sets and possibility distributions to model preferences and uncertainty, respectively.

Using basic tools to evaluate query satisfaction, we have proposed methods allowing us to (1) extend fuzzy sets to hierarchical fuzzy sets which put in adequacy two order relations (the preference order relation and the 'kind of' relation) to permit a query enlargement (2) consider orderings between constraints or wishes and (3) pre-order the results according to the bipolar preferences, thus presenting a list of equivalence classes to the user.

The proposed approach is applied in a real-case problem, and is included in a new support decision tool aiming at designing (optimal) packages for fresh fruits and vegetables.

Concerning the method, perspectives include the handling of more generic kinds of uncertainty models [15, 16] that could be included in the database, as well as

methods that would allow to extract information concerning packages from the web automatically [9], since manually entering this information is time-consuming and can only be done by an expert.

Concerning the support decision tool, we are planning to link it with a preliminary step which will combine preferences expressed by the actors of the food packaging chain, which can be potentially in conflict, using argumentation methods.

Acknowledgements The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/ 2007-2013) under the grant agreement FP7-265669-EcoBioCAP project.

References

[1] S. Benferhat, D. Dubois, S. Kaci, and H. Prade. Bipolar possibility theory in preference modelling: representation, fusion and optimal solutions. *Information Fusion*, 7:135–150, 2006.

[2] G. Bordogna and G. Pasi. A fuzzy object oriented data model managing vague and uncertain information. *International Journal of Intelligent Systems*, 14(6):SCI 3495, 1999.

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

[4] P. Bosc, L. Lietard, and O. Pivert. *Fuzziness in Database Management Systems*, chapter Fuzzy theory techniques and applications in data-base management systems, pages 666–671. Academic Press, 1999.

[5] P. Bosc and O. Pivert. SQLf: a relational database language for fuzzy querying. *IEEE Transactions on fuzzy systems*, 3(1):1–17, February 1995.

[6] P. Bosc and O. Pivert. About bipolar division operators. In *Flexible Query Answering Systems, 8th International Conference, FQAS 2009, Roskilde, Denmark, October 26-28, 2009. Proceedings*, volume 5822 of *Lecture Notes in Computer Science*, pages 572–582. Springer, 2009.

[7] P. Bosc, O. Pivert, A. Mokhtari, and L. Lietard. Extending relational algebra to handle bipolarity. In *Proceedings of the 2010 ACM Symposium on Applied Computing (SAC), Sierre, Switzerland, March 22-26, 2010*, pages 1718–1722. ACM, 2010.

[8] N. Bruno, S. Chaudhuri, and L. Gravano. Top-k selection queries over relational databases: Mapping strategies and performance evaluation. *ACM Trans. Database Syst.*, 27(2):153–187, 2002.

[9] P. Buche, J. Dibie-Barthelemy, L. Ibanescu, and L. Soler. Fuzzy web data tables integration guided by an ontological and terminological resource. *IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints), 2011.

[10] P. Buche and O. Haemmerlé. Towards a unified querying system of both structured and semi-structured imprecise data using fuzzy views. In *Proceedings of the 8th International Conference on Conceptual Structures, Lecture Notes in Artificial Intelligence #1867*, pages 207–220, Darmstadt, Germany, August 2000. Springer-Verlag.

[11] J. T. Cacioppo, W. L. Gardner, and G. G. Berntson. Beyond bipolar conceptualizations and measures: The case of attitudes & evaluative space. *Personality & Social Psychology Review*, 1:3–25, 1997.

[12] F. Charles, J. Sanchez, and N. Gontard. Modeling of active modified atmosphere packaging of endives exposed to several postharvest temperatures. *Journal of Food Science*, 8:443–448, 2005.

[13] J. Chomicki. Preference formulas in relational queries. *ACM Trans. Database Syst.*, 28(4):427–466, 2003.

[14] S. Destercke, P. Buche, and V. Guillard. A flexible bipolar querying approach with imprecise data and guaranteed results. *Fuzzy Sets and Systems*, 169(1):51–64, 2011.

[15] S. Destercke, D. Dubois, and E. Chojnacki. Unifying practical uncertainty representations: I generalized p-boxes. *Int. J. of Approximate Reasoning*, 49(3):664–677, 2008.

[16] S. Destercke, D. Dubois, and E. Chojnacki. Unifying practical uncertainty representations: II clouds. *Int. J. of Approximate Reasoning*, 49(3):649–663, 2008.

[17] S. Destercke and V. Guillard. Interval analysis on non-linear monotonic systems as an efficient tool to optimise fresh food packaging. *Computers and Electronics in Agriculture*, 79(2):116–124, Nov 2011.

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

[19] D. Dubois and H. Prade. *Fuzziness in Database Management Systems*, chapter Tolerant fuzzy pattern matching: An introduction. Physica-Verlag, 1995.

[20] D. Dubois and H. Prade. Bipolarity in flexible querying. In *Flexible Query Answering Systems, 5th International Conference, FQAS 2002, Copenhagen, Denmark, October 27-29, 2002, Proceedings*, volume 2522 of *Lecture Notes in Computer Science*, pages 174–182. Springer, 2002.

[21] D. Dubois and H. Prade. An overview of the asymmetric bipolar representation of positive and negative information in possibility theory. *Fuzzy Sets and Systems*, 160:1355–1366, 2009.

[22] W. Kießling and G. Köstler. Preference sql - design, implementation, experiences. In *VLDB Proceedings of 28th International Conference on Very Large Data Bases, August 20-23, 2002, Hong Kong, China*, pages 990–1001. Morgan Kaufmann, 2002.

[23] E. Klement, R. Mesiar, and E. Pap. *Triangular Norms*. Kluwer Academic Publisher, Dordrecht, 2000.

[24] C. Labreuche. A general framework for explaining the results of a multi-attribute preference model. *Artif. Intell.*, 175(7-8):1410–1448, 2011.

[25] P. Mahajan, F. Oliveira, J. Montanez, and J. Frias. Development of user-friendly software for design of modified atmosphere packaging for fresh and fresh-cut produce. *Innovative Food Science and Emerging Technologies*, 8:84–92, 2007.

[26] H. Prade. 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, 1984.

[27] H. Prade and C. Testemale. Generalizing database relational algebra for the treatment of incomplete or uncertain information and vague queries. *Information Sciences*, 34:115–143, 1984.

[28] R. Thomopoulos, P. Buche, and O. Haemmerlé. Fuzzy sets defined on a hierarchical domain. *IEEE Trans. Knowl. Data Eng.*, 18(10):1397–1410, 2006.

[29] G. D. Tré, , and R. D. Caluwe. *Recent research issues on the management of fuzziness in databases*, chapter A generalized object-oriented database model, pages 155–182. Bordogna, G. and Pasi, G. (eds), Studies in Fuzziness and Soft computing, Vol. 53, Physica-Verlag, Heidelberg, Germany, 2000.

[30] G. D. Tré, S. Zadrozny, and A. Bronselaer. Handling bipolarity in elementary queries to possibilistic databases. *IEEE Trans. on Fuzzy Systems*, 18:599–612, 2010.

[31] G. D. Tré, S. Zadrozny, T. Matthé, J. Kacprzyk, and A. Bronselaer. Dealing with positive and negative query criteria in fuzzy database querying. In *Flexible Query Answering Systems, 8th International Conference, FQAS 2009, Roskilde, Denmark, October 26-28, 2009. Proceedings*, volume 5822 of *Lecture Notes in Computer Science*, pages 593–604. Springer, 2009.

[32] R. Yager. On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Trans. on Syst., Man, and Cybern.*, 18:183–190, 1988.

[33] L. Zadeh. The concept of a linguistic variable and its application to approximate reasoning-i. *Information Sciences*, 8:199–249, 1975.

[34] L. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1:3–28, 1978.