



HAL
open science

A Flexible Bipolar Querying Approach with Imprecise Data and Guaranteed Results

Sébastien Destercke, Patrice Buche, Valérie Guillard

► **To cite this version:**

Sébastien Destercke, Patrice Buche, Valérie Guillard. A Flexible Bipolar Querying Approach with Imprecise Data and Guaranteed Results. *Fuzzy Sets and Systems*, 2011, 169 (1), pp.51-54. 10.1016/j.fss.2010.12.014 . lirmm-00611940

HAL Id: lirmm-00611940

<https://hal-lirmm.ccsd.cnrs.fr/lirmm-00611940>

Submitted on 29 May 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

0001
0002
0003
0004
0005
0006
0007
0008
0009
0010
0011
0012
0013
0014
0015
0016
0017
0018
0019
0020
0021
0022
0023
0024
0025
0026
0027
0028
0029
0030
0031
0032
0033
0034
0035
0036
0037
0038
0039
0040
0041
0042
0043
0044
0045
0046
0047
0048
0049

A FLEXIBLE BIPOLAR QUERYING APPROACH WITH IMPRECISE DATA AND GUARANTEED RESULTS

SÉBASTIEN DESTERCKE, PATRICE BUCHE, AND VALÉRIE GUILLARD

ABSTRACT. In this paper, we propose an approach to query a database when the user preferences are bipolar (i.e., express both constraints and wishes about the desired result) and the data stored in the database are imprecise. Results are then completely ordered with respect to these bipolar preferences, giving priority to constraints over wishes. Additionally, we propose a treatment that allows us to guarantee that any query will return a result, even if no element satisfies all constraints specified by the user. Such a treatment may be useful when user's constraints are unrealistic (i.e., cannot be all satisfied simultaneously) and when the user desires a guaranteed result. The approach is illustrated on a real-world problem concerning the selection of optimal packaging for fresh fruits and vegetables.

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, since they specify which values or object 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 preference should be satisfied by query results, satisfying the second type of preference can be considered as optional, as the user does not consider them as necessary requirements. Also, it may be useful to have procedures that send back to the user some answer to its query, even if this answer does not completely fulfil its need (suggestions proposed by modern web search engines when entering mistyped entries can be assimilated to such procedures). Finally, there may be imprecise data in the database, and there is also a need to take this imprecision into account. In this paper, we propose to consider these three problems in a

Key words and phrases. Bipolarity, Possibility theory, Fuzzy databases, Food engineering, Applications.

0050 common framework, using the notion of bipolar information and of fuzzy
0051 pattern matching.

0052
0053 The notions of bipolar preferences and of bipolar information in general
0054 have recently received increasing attention [2, 23]. Roughly speaking, in-
0055 formation is said to be bipolar when there is a positive and a negative part of
0056 the information. These negative and positive parts of the information may
0057 have different natures, and should therefore be processed in parallel, and
0058 not as a single piece of information. This kind of bipolarity [14], coined
0059 as asymmetric, is the one we are concerned with. For example, we may
0060 feel both positive and negative about something, without being able to fuse
0061 these two feelings in a unique one (for example, eating ice cream gives a
0062 gustative pleasure, but one can also feel guilty about it).

0063
0064 In the case of database queries, asymmetric bipolarity is useful to dis-
0065 tinguish negative preferences or constraints (i.e. criteria that a satisfying
0066 answer **must** satisfy) from positive preferences or wishes (i.e. criteria that a
0067 good answer **should** satisfy, if possible). For example, in the query "a new
0068 car not too expensive and if possible red", "not too expensive" is clearly a
0069 requirement while "red" merely expresses a wish.

0070
0071 In this paper, we propose a method to treat bipolar preferences in data
0072 bases where data can be uncertain. In particular, this method uses the bipolar
0073 nature of preferences to induce an (pre-)ordering between query results,
0074 so that priority is given to those instances that are the most likely to satisfy
0075 all expressed constraints. Section 2 describes the method, while Section 3
0076 illustrates the approach on a use case coming from a new decision support
0077 system (DSS) currently developed in our lab where a (industrial/researcher)
0078 user wants to select a packaging material that best suits his/her needs. Fi-
0079 nally, we give some elements of comparison with previous works in Section
0080 4.
0081
0082

0083 2. METHOD

0084
0085
0086 This section first recalls some basic tools that will be used in the method,
0087 before describing the method itself.

0088
0089 **2.1. Preliminaries: fuzzy pattern matching.** In this paper, we use fuzzy
0090 sets [33] to represent preferences in our queries and possibility distribu-
0091 tions [19] to represent the possible imprecision in the data. A normal-
0092 ized fuzzy set μ over a variable X assuming its value on \mathcal{X} is a mapping
0093 $\mu : \mathcal{X} \rightarrow [0, 1]$ such that there is at least one $x \in \mathcal{X}$ such that $\mu(x) = 1$.
0094 Here, we assume that \mathcal{X} is either a finite set of elements (e.g., the colour
0095 of a car) or a subset of the real line (e.g., the maximal speed of a car).

0096
0097 Here, fuzzy sets are used to express preferences provided by a user in
0098 a query. That is, for a given variable X , the fuzzy value $\mu(x)$ expresses

0099
0100
0101
0102
0103
0104
0105
0106
0107
0108
0109
0110
0111
0112
0113
0114
0115
0116
0117
0118
0119
0120
0121
0122
0123
0124
0125
0126
0127
0128
0129
0130
0131
0132
0133
0134
0135
0136
0137
0138
0139
0140
0141
0142
0143
0144
0145
0146
0147

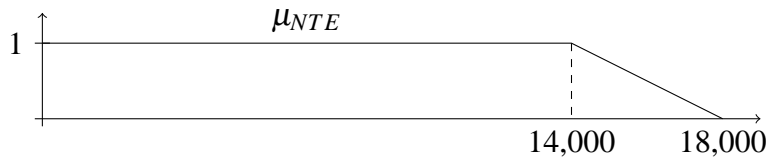


FIGURE 1. Fuzzy set μ_{NTE} describing "Not Too Expensive"

to which 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 $\mu_{NTE}(15,000) = 2/3$

Possibility distributions, on the other hand, are simple uncertainty representations allowing to model data whose value is imprecisely known. A possibility distribution π over a variable X is also a mapping $\pi : \mathcal{X} \rightarrow [0, 1]$ such that there is at least one $x \in \mathcal{X}$ 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 $A \subset \mathcal{X}$,

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

where $\Pi(A)$ and $N(A)$ express to which 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 ways, 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 is expressed by a fuzzy set μ_P (possibly degenerated into a crisp preference) and 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

0148
0149
0150
0151
0152
0153
0154
0155
0156
0157
0158
0159
0160
0161
0162
0163
0164
0165
0166
0167
0168
0169
0170
0171
0172
0173
0174
0175
0176
0177
0178
0179
0180
0181
0182
0183
0184
0185
0186
0187
0188
0189
0190
0191
0192
0193
0194
0195
0196

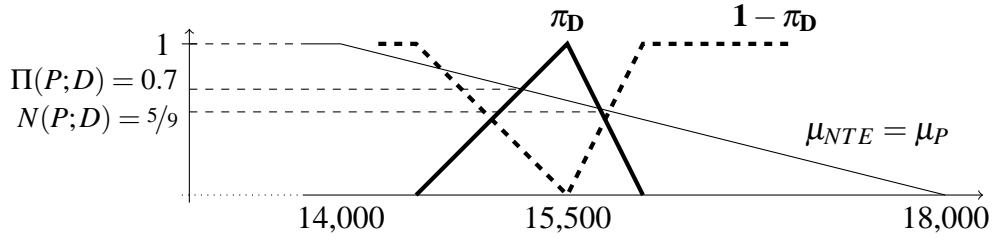


FIGURE 2. Evaluation of a fuzzy preference with uncertain data.

of P given uncertainty D is summarised by the following lower and upper values [20, 19]:

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

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

$$\Pi(P;D) = 0.7 \quad \text{and} \quad N(P;D) = 5/9$$

2.2. Notations and problem. The problem we consider is as follows: we assume that we have a database consisting in a set \mathcal{T} of T objects o_t , $t = 1, \dots, T$, with each object taking its values on the Cartesian product $\times_{i=1}^N \mathcal{X}_i$ of N domains $\mathcal{X}_1, \dots, \mathcal{X}_N$. An object o_t is here described by a set of N possibility distributions π_t^i , $i = 1, \dots, N$, where $\pi_t^i : \mathcal{X}_i \rightarrow [0, 1]$ is the possibility distribution describing our knowledge about the value of the i^{th} attribute of object t .

We assume that the user provides the following information:

- a set $\mathcal{C} = \{C_1^{i_1}, \dots, C_{N_c}^{i_{N_c}}\}$ of N_c constraints ($N_c \leq N$) to be satisfied by the retrieved objects, where $C_j^{i_j} : \mathcal{X}_{i_j} \rightarrow [0, 1]$ is a normalised fuzzy set defined on the attribute i_j ($1 \leq i_j \leq N$).
- a set $\mathcal{W} = \{W_1^{i_1}, \dots, W_{N_w}^{i_{N_w}}\}$ of N_w wishes ($N_w \leq N$) that the retrieved objects should satisfy if possible, where $W_j^{i_j} : \mathcal{X}_{i_j} \rightarrow [0, 1]$ 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 sequel, we denote by $\mathcal{C}_{(i)}$ (resp. $\mathcal{W}_{(i)}$) the constraints (resp. the wishes) that have rank i w.r.t. to the (pre-)ordering¹ \leq_c (resp. \leq_w). We denote by $|\leq_c|$ and $|\leq_w|$ the total number of ranks 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 \$).

Now, the problem we consider here is how to retrieve from a set \mathcal{T} of objects, those that primarily satisfy the constraints, and among these latter ones, those that fulfil the most wishes. Of course, the querying approach has to take account of the bipolar nature of the information, of the possible imprecision in the data, and of the users preferences among constraints and wishes.

We also require that an answer is returned for every query, even if no object is found that satisfies the specified constraints. The reason is that the user may be ready to relax some of his/her constraints, but is often unaware of the constraints that would have to be relaxed in order to have a non-empty answer. Hence, it may be helpful to provide those objects that are close to satisfying the constraints.

2.3. From bipolar querying with imprecise data to answer ordering.

As underlined by [2], 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)², we also provide a means to take these preferences into account.

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

²No preferences means here that all constraints (or wishes) have the same rank, i.e., are of equal importance.

We propose, for constraints $\mathcal{C}_{(i)}$ 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:

$$(2) \quad N_t^{(i)} = \top_{C_k^{jk} \in \mathcal{C}_{(i)}} N(C_k^{jk}; \pi_t^{jk}), \quad \text{and} \quad \Pi_t^{(i)} = \top_{C_k^{jk} \in \mathcal{C}_{(i)}} \Pi(C_k^{jk}; \pi_t^{jk}),$$

with $N(C_k^{jk}; \pi_t^{jk})$, $\Pi(C_k^{jk}; \pi_t^{jk})$ given by Eq. (1), and \top a t-norm³ [25]. 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 $\mathcal{W}_{(i)}$ and object o_t satisfying the constraints, the interval $[N_t^{(i)}, \Pi_t^{(i)}]_w$, such that

$$(3) \quad N_t^{(i)} = \oplus_{W_k^{jk} \in \mathcal{W}_{(i)}} N(W_k^{jk}; \pi_t^{jk}), \quad \text{and} \quad \Pi_t^{(i)} = \oplus_{W_k^{jk} \in \mathcal{W}_{(i)}} \Pi(W_k^{jk}; \pi_t^{jk}),$$

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 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 [2]. 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 consider $\oplus = \min$.

It is then necessary 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 $\mathcal{C}_{(i)}$ or a group of wishes $\mathcal{W}_{(i)}$, $o_t \leq_{[N^{(i)}, \Pi^{(i)}]} o_{t'}$ 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

³A T-norm $\top : [0, 1]^2 \rightarrow [0, 1]$ is an associative, commutative operator that has one for neutral element and zero for absorbing element.

$\leq_{[N^{(i)}, \Pi^{(i)}]}$ is a partial order to differentiate negative from positive preferences. The ordering procedure consists in building iteratively an ordered partition $\{\mathcal{T}_0, \dots, \mathcal{T}_M\}$ of \mathcal{T} . Rejected objects that do not satisfy all constraints are put in \mathcal{T}_0 , while objects in \mathcal{T}_M can be considered as the most satisfactory.

In a preliminary step, Algorithm 1 describes how the objects of \mathcal{T} that do not satisfy at all some constraints, are rejected. If all objects are rejected, the user is warned and a guaranteed answer method, described in Sec. 2.4, is then used to return possible interesting answers to the user.

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

Input: The set of objects $\mathcal{T} = \{o_1, \dots, o_T\}$

Output: Ordered partition $\{\mathcal{T}_0, \mathcal{T} \setminus \mathcal{T}_0\}$ of \mathcal{T}

- 1 $\mathcal{T}_0 = \emptyset$;
 - 2 **foreach** $o_t \in \mathcal{T}$ **do**
 - 3 **if** $\Pi_t^{(i)} = 0$ for some $i = 1, \dots, |\leq_c|$ **then**
 - 4 $\mathcal{T}_0 = \mathcal{T}_0 \cup \{o_t\}$;
 - 5 **if** $\mathcal{T}_0 = \mathcal{T}$ **then**
 - 6 Send empty result message to user and propose alternative results using method of Sec. 2.4
-

Algorithm 2 describes how results are ordered within a subset of $\mathcal{T} \setminus \mathcal{T}_0$ (called \mathcal{T}'), according to constraints of a given rank. The whole procedure consists in building a partition of $\mathcal{T} \setminus \mathcal{T}_0$. 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 \mathcal{T}' is $\mathcal{T} \setminus \mathcal{T}_0$. In every run of Algorithm 2, equivalence classes $\{\mathcal{T}'_1, \dots, \mathcal{T}'_n\}$ are incrementally built, starting from the worst (\mathcal{T}'_1) and ending with the best (\mathcal{T}'_n). At each step, the objects included and then suppressed from \mathcal{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 associated to 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: $\mathcal{T}' \subseteq \mathcal{T} \setminus \mathcal{T}_0$ with \mathcal{T}' an element of the partition issued from rank ($i-1$), $[N_t^{(i)}, \Pi_t^{(i)}]_c$ for each $o_t \in \mathcal{T}'$
Output: Ordered partition $\{\mathcal{T}'_1, \dots, \mathcal{T}'_n\}$ of \mathcal{T}'

```

1  $K = \mathcal{T}'$ ;  $j=1$ ;
2 while  $K \neq \emptyset$  do
3   foreach  $o_t \in K$  do
4     if  $\nexists o_j \in K$  s.t.  $o_t \geq_{[N^{(i)}, \Pi^{(i)}]} o_j$  then
5       Put  $o_t$  in  $\mathcal{T}'_j$ 
6    $K = K \setminus \mathcal{T}'_j$ ;
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 worst⁴, and at each step the objects included and then suppressed from \mathcal{T}' are those objects that are not dominated by other objects (line 7), in the sense of $\leq_{[N^{(i)}, \Pi^{(i)}]}$. Contrarily 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. Also notes that the problem of inconsistency raised, for example, in [2], does not happen here, since constraints and wishes are treated separately and lexicographically.

By fully acknowledging the knowledge imprecision through the use of the partial order $\leq_{[N^{(i)}, \Pi^{(i)}]}$ (that considers both end-points of intervals $[N^{(i)}, \Pi^{(i)}]$), Algorithms 2 and 3 allow 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 represent the use of these algorithms. Indeed, each run of Algorithms 2 and 3 requires to compare 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, that is when objects are completely ordered after a first run, n^2 comparisons have to be achieved. It

⁴The shift loop (Lines 3-5) is there to keep the same indexing of subsets \mathcal{T}'_j

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

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

Output: Ordered partition $\{\mathcal{T}'_1, \dots, \mathcal{T}'_m\}$ of \mathcal{T}'

```

1  $K = \mathcal{T}'$ ;  $j=0$ ;
2 while  $K \neq \emptyset$  do
3   for  $i = j, \dots, 1$  (skip if  $j = 0$ ) do
4      $\mathcal{T}'_{j+1} = \mathcal{T}'_j$ 
5      $\mathcal{T}'_1 = \emptyset$ ;
6     foreach  $o_t \in K$  do
7       if  $\nexists o_j \in K$  s.t.  $o_t \leq_{[N^{(i)}, \Pi^{(i)}]} o_j$  then
8          $\mathcal{T}'_1 = \mathcal{T}'_1 \cup \{o_t\}$ 
9      $K = K \setminus \mathcal{T}'_1$ ;
10     $j = j + 1$ ;

```

	$[N_t^{(1)}, \Pi_t^{(1)}]_c$	$[N_t^{(2)}, \Pi_t^{(2)}]_c$	$[N_t^{(1)}, \Pi_t^{(1)}]_w$
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.

must be noticed that n is reduced to $|\mathcal{T} \setminus \mathcal{T}_0|$ 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, one can then use other propositions presenting a lower complexity where object ordering is solely based on one of the two numbers $N^{(i)}$ or $\Pi^{(i)}$ [20]. Still, by using orderings based on single numbers, one does not take full account of the imprecision in $[N^{(i)}, \Pi^{(i)}]$, and loose some of the information contained in the interval.

Example 3. Let us consider a set \mathcal{T} of six objects o_1, \dots, 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.

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

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

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

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

This second run has allowed to refine the ordering between o_1 and o_6 . Also note that the bad scores of o_5 w.r.t. constraints of rank 2 does not change its order, due to the constraints preferences and the use of a lexicographic order. Finally, a run of Algorithm 3 gives

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

Note that o_5 is not rejected, for the reason that satisfying wishes is not a requirement.

2.4. Guaranteeing some answers. In Example 3, the set \mathcal{T}_0 of rejected objects did not contain all of them (i.e., $\mathcal{T}_0 \neq \mathcal{T}$), and there were positive answers to the query. However, it may happen that a query has no satisfying answer. This may be the result of a small number of available objects (hence there are less chances of satisfying all constraints) or of the specification of unrealistic or narrow constraints. In such a case, it may be difficult for a user to know why its query has received no answers, and which constraints should be relaxed to get some.

This is why, in this section, we propose a method guaranteeing the presence of some answers, even when $\mathcal{T}_0 = \mathcal{T}$. The principle is quite simple: after having sent a message to the user (see Algorithm 1), it consists in returning the set \mathcal{T} of objects (or a subset of \mathcal{T}), ordered w.r.t. to the gap between these objects' attribute values and values they should have in order not to be rejected.

First, recall that the support of a fuzzy set F defined on \mathcal{X} , denoted by $S(F)$, is the set such that $S(F) = \{x \in \mathcal{X} | F(x) > 0\}$.

We assume that, for every attribute j , a (normalized) distance $d_j \in [0, 1]$ on \mathcal{X}_j is defined. Consider the set of supports $\{S(C_1^{i_1}), \dots, S(C_{N_c}^{i_{N_c}})\}$ obtained from the set \mathcal{C} of constraints. For an object o_i and an attribute j , the

distance between π_t^j and a constraint C^j is defined as the following value:

$$D_j(\pi_t^j, C^j) = \min_{x \in S(\pi_t^j), y \in S(C^j)} (|d_j(x, y)|).$$

This distance can be seen as the minimal modification a constraint C^j would have to undergo so that object o_t satisfies it. Now, we define the global distance of an object o_t w.r.t. constraints in \mathcal{C} as a function δ having value in $[0, N_c]$ and such that:

$$\delta(o_t, \mathcal{C}) = \sum_{k=1}^{N_c} w_{(k)} D_{i_k}(\pi_t^{i_k}, C_k^{i_k}),$$

with (k) the rank of $C_k^{i_k}$ according to \leq_c and $w_{(k)}$ a weight reflecting the constraint importance. A sum is used to translate the fact that an almost acceptable object should be (in average) close to all constraints. Objects in \mathcal{T}_0 can then be ordered according to distance δ . We propose the following formula for the weights $w_{(k)}$ for any $k \in 1, \dots, |\leq_c|$:

$$w_{(k)} = \frac{|\leq_c| + 1 - k}{|\leq_c|},$$

that is, weights are in the interval $[0, 1]$ and are linearly ordered. A possible inconvenient when using an averaging operator is that objects violating very strongly only one constraint can have the same evaluation as objects weakly violating many constraints. Two possible solutions to this problem are to provide the user with a list of the violated constraint, or to use some kind of veto level (as in [11]) to penalise objects strongly violating some constraints (this latter option meaning that the user must supply such veto levels).

Example 4. Let us consider a set $\mathcal{T} = \{o_1, \dots, o_3\}$ of three objects with two constraints expressed in the query. The intervals $[N_t^i, \Pi_t^i]_c$ ($i = \{1, 2\}$) and the minimal distances to the constraints $D_i(\pi_t^i, C^i)$ ($i = \{1, 2\}$) are given in Table 2. In this example, there is no positive answer to the query because, at least one of the constraints is not satisfied (i.e., $\mathcal{T} = \mathcal{T}_0$) for each object. According to distance δ and to the weights $w_{(1)} = 1$, $w_{(2)} = 1/2$, objects are ranked in the following order: o_1, o_3, o_2 .

There exists other methods to obtain answers from empty queries. They mainly consist of relaxing the initial query into a less constraining query using, for example, fuzzy membership function transformations [5] or case-based reasoning techniques [4]. These approaches are arguably more subtle and efficient but necessitate to specify many parameters and to apply (possibly many times) new queries, and are therefore computationally more demanding. Our aim here was to propose a simple but meaningful and efficient technique, that would not require additional parameters.

	$[N_t^1, \Pi_t^1]_c$	$[N_t^2, \Pi_t^2]_c$	$D_1(\pi_t^1, C^1)$	$D_2(\pi_t^2, C^2)$	$\delta(o_t, \mathcal{C})$
o_1	[0.0, 0.0]	[0.8, 1]	0.1	0	0.1
o_2	[0.0, 0.0]	[0.0, 0.0]	0.6	0.5	0.85
o_3	[0.3, 0]	[0.0, 0.0]	0	0.7	0.35

TABLE 2. Example 4 evaluations for two constraints and the minimal distances to the constraints.

3. A NEW DECISION SUPPORT SYSTEM FOR FOOD PACKAGING DESIGN

In this section, we present a new decision support system (DSS) for fresh fruits and vegetables 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 [26]), but it does not take into account the criteria which permit to ensure a sustainable design (a critical issue in food science). Such a sustainable design must satisfy, at least, three kinds of criteria: economical, environmental and societal. An example of economical aspect may be the cost of the packaging material. Concerning environmental aspects, the biodegradability of the package or the way to optimize the preservation of fresh fruits and vegetables at ambient temperature in order to decrease the use of the cold chain (which is energy-greedy) are important criteria. Societal aspects may concern the fact that consumers may reject the use of some additives or of nano-technology 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 its needs in terms of several criteria (e.g., conservation temperature, transparency, material cost, ...) in order to determine a list of packagings. Those packagings 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 the packages which satisfy the constraints. Starting from the users' 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 imprecise, due to the variability of engineered packaging and the biological variability of vegetables. The bipolar approach proposed in this paper takes into account this imprecision.

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, the system scans in a first step the vegetable/fruit database in order to retrieve the O_2 respiration rate (and associated parameters) of the studied vegetable/fruit. In a second step, the optimal O_2 permeance⁵ of the targeted packaging is computed thanks to a model of gas exchanges inside the package called PassiveMap (see [15] for more details about the model). In the third step, using the targeted optimal O_2 permeance and the other user's 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. A precise use case presented in the next section focuses on the DSS flexible bipolar querying engine.

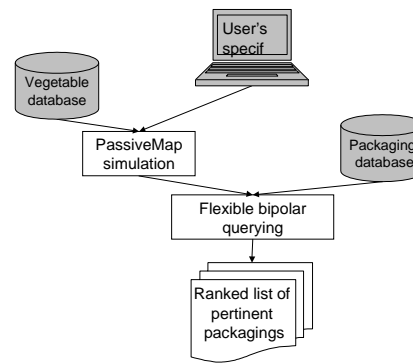


FIGURE 3. Global architecture of the DSS

3.2. Endive packaging use case. In this section, we present a use case of the DSS concerning the choice of a package for endive. 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 $5\text{e-}5 \text{ m}$), the shelf life of the vegetable (7 days) and the storage temperature ($20 \text{ }^\circ\text{C}$). Using the O_2 respiration rate (and associated parameters) retrieved from the vegetable database, an optimal O_2 permeance of $3.65\text{E-}11 \text{ mol.m}^{-2}.\text{s}^{-1}.\text{Pa}^{-1}$ is computed. The

⁵A measure of the ability of a package to conduct gas fluxes.

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

TABLE 3. Permeance at a given temperature for a excerpt of the packaging database

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 excerpt of the packaging database content is presented in Tables 3 and 4 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), for the reason that there are currently no possibilistic imprecise data in the 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 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.

In the second query, the user specifies three constraints and one wish. To design a sustainable package, the user expresses that the packaging must be biodegradable as a first constraint (rank one) and must also maximize the shelf life of the product at an ambient temperature as a second constraint (i.e. constraints of rank two on the temperature and oxygen permeance).

0687
0688
0689
0690
0691
0692
0693
0694
0695
0696
0697
0698
0699
0700
0701
0702
0703
0704
0705
0706
0707
0708
0709
0710
0711
0712
0713
0714
0715
0716
0717
0718
0719
0720
0721
0722
0723
0724
0725
0726
0727
0728
0729
0730
0731
0732
0733
0734
0735

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

TABLE 4. Transparency and biodegradability for the same excerpt of the packaging database

Then, the user expresses as its only wish that the packaging should be transparent for acceptability of this new packaging material by the consumer.

As already said in Section 2.1, user preferences for each criterion is expressed as a fuzzy set used as a general formalism which permits to represent 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 authorised, with no preference for the different values. The fuzzy sets associated with the permeance and temperature preferences are presented in Figure 4.

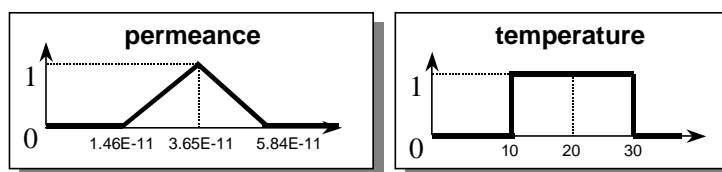


FIGURE 4. Preferences for permeance and temperature

The fuzzy set associated with the transparency (resp. biodegradability) criterion is⁶: $Pref_{transparency} = \{(transparent, 1), (translucent, 0), (opaque, 0)\}$ (resp.

⁶Here, 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 .

0736
0737
0738
0739
0740
0741
0742
0743
0744
0745
0746
0747
0748
0749
0750
0751
0752
0753
0754
0755
0756
0757
0758
0759
0760
0761
0762
0763
0764
0765
0766
0767
0768
0769
0770
0771
0772
0773
0774
0775
0776
0777
0778
0779
0780
0781
0782
0783
0784

	$[N_t^{(1)}, \Pi_t^{(1)}]_c$	$[N_t^{(1)}, \Pi_t^{(1)}]_w$
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 5. Evaluations for the constraint and the wishes of the first query.

$Pref_{biodegradability} = \{(yes, 1), (no, 0)\}$). 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.

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

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

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

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

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

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

The second query is built as follows: $\mathcal{C}_{(1)} = \{Pref_{biodegradability}\}$, $\mathcal{C}_{(2)} = \{Pref_{permeance}, Pref_{temperature}\}$ and $\mathcal{W}_{(1)} = \{Pref_{transparency}\}$. The first constraint is judged more important than the two others, which are of equal rank.

0785
0786
0787
0788
0789
0790
0791
0792
0793
0794
0795
0796
0797
0798
0799
0800
0801
0802
0803
0804
0805
0806
0807
0808
0809
0810
0811
0812
0813
0814
0815
0816
0817
0818
0819
0820
0821
0822
0823
0824
0825
0826
0827
0828
0829
0830
0831
0832
0833

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

TABLE 6. Evaluations for the constraints and the wish of the second query.

Let us consider the set \mathcal{T} of the twelve packages o_1, \dots, o_{12} whose characteristics are given in tables 3 and 4 and whose evaluations for the constraints and the wish of query 2 are given in table 6. After the run of Algorithm 1, we obtain $\mathcal{T}_0 = \{o_1, o_2, o_3, o_4, o_5, o_6, o_7, o_8, o_9, o_{10}, o_{12}\}$. After the first run of Algorithm 2 with $\mathcal{C}_{(1)}$, we obtain the following partition:

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

Since only one object is not rejected, the second run of Algorithm 2 with $\mathcal{C}_{(2)}$ ($[N_t^{(2)}, \Pi_t^{(2)}]_c$) and the run of Algorithm 3 with $\mathcal{W}_{(1)}$ keeps the partition unchanged.

We can see with the result obtained for the second query, from which only one result is retrieved, that the constraints may be very restrictive compared to the content of the database. We consider in the following a third query, derived from query 2 and defined as follows: $\mathcal{C}_{(1)} = \{Pref_{biodegradability}\}$, $\mathcal{C}_{(2)} = \{Pref_{permeance_restricted}, Pref_{temperature}\}$ and $\mathcal{W}_{(1)} = \{Pref_{transparency}\}$. In this query, we restrict the variation around the permeance optimal value ($3.65E-11 \text{ mol.m}^{-2}.\text{s}^{-1}.\text{Pa}^{-1}$) from 60% to only 50%. Consequently, the support of the restricted fuzzy set $Pref_{permeance_restricted}$ is equal to the interval $[1.83E-11, 5.48E-11] \text{ mol.m}^{-2}.\text{s}^{-1}.\text{Pa}^{-1}$. Query 3 being more restrictive than query 2 and considering the permeance value associated with package o_{11} in Table 3, we can see that the result of query 3 is empty.

In this case, our method will provide guaranteed results which are defined as the set of packages ordered w.r.t. to the gap between the packaging attribute values and the values they should have in order not to be rejected.

	$D_1(\pi_t^1, C^1)$	$D_2(\pi_t^2, C^2)$	$D_3(\pi_t^3, C^3)$	$\delta(o_t, \mathcal{C})$	rank	
0834						
0835						
0836	o_1	1	0.6828	0	1,3414	10
0837	o_2	1	0	0	1	6
0838	o_3	0	0.7545	0	0.3772	5
0839	o_4	1	0	0	1	6
0840	o_5	0	0.7545	0	0.3772	5
0841	o_6	0	0.3813	0	0.1906	3
0842	o_7	1	0.0813	0	1,0406	7
0843	o_8	0	0.6746	0	0.3373	4
0844	o_9	1	0.5555	0	1,2777	9
0845	o_{10}	0	0.3205	0	0.1602	2
0846	o_{11}	0	0.0406	0	0.0203	1
0847	o_{12}	1	0.1187	0	1,0593	8
0848						
0849						
0850						

TABLE 7. The minimal distances to the constraints of query 3.

As introduced in section 2.4, we need to define for every object attribute j , a normalized distance $d_j \in [0, 1]$ on \mathcal{X}_j . In the following, we will consider, for the attribute *biodegradability* defined on $\mathcal{X}_{biodegradability} = \{yes, no\}$, the distance $d_1(x, y)$ with $(x, y) \in \mathcal{X}_{biodegradability}^2$, which is equal to 1 if $x = y$ and equal to 0 else. For the attributes *permeance* (resp. *temperature*) defined on \mathbb{R} , we consider the distance $d_2(x, y)$ (resp. $d_3(x, y)$) with $(x, y) \in \mathbb{R}^2$ and defined as follows:

$$d_2(x, y) = d_3(x, y) = 1 - \frac{1}{1 + |x - y|}.$$

Note that, given the variation of permeance compared to, e.g. temperature, we have applied a log transformation to it before computing the distance.

Let us consider the set \mathcal{T} of the twelve packages o_1, \dots, o_{12} whose characteristics are given in tables 3 and 4, the normalized distances given above and the supports of the fuzzy sets $Pref_{biodegradability}$, $Pref_{permeance_restricted}$, $Pref_{temperature}$ being respectively the singleton $\{yes\}$, the interval $[1.83E-11, 5.48E-11] \text{ mol.m}^{-2}.\text{s}^{-1}.\text{Pa}^{-1}$ and the interval $[10, 30] \text{ }^\circ\text{C}$, the minimal distances to the constraints of query 3 and a rank based on the global distance are given in table 7. We can see that packaging o_{11} is at the first rank which is not surprising because it is the only one which satisfies the constraints of query 2. Hence, in the above case, the user would be able to see that package o_{11} almost answers his/her needs.

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

Indeed, the fuzzy set framework has been shown to be a sound scientific choice to model flexible queries ([6]). It is a natural way of representing the notion of preference using a gradual scale. In [8], 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 ([12, 24, 16]). 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 [21, 22]. 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 ([9, 10]) or to propose a framework to deal with bipolarity in regular relational databases ([31]).

The second proposition is to use possibility distributions (whose formalism is mathematically equivalent to that of fuzzy set) to represent imprecise values ([34]). Several authors have developed this approach in the context of databases ([27, 28, 7, 3, 29, 13]).

To the best of our knowledge, the only other works dealing with the concept of bipolarity in flexible querying of databases including imprecise values, outside some research perspectives in [22], is the one of G. De tr *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) and data are uncertain. 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) consider orderings between constraints or wishes (2)

(pre-)order the results according to the bipolar preferences, thus presenting a list of equivalence classes to the user and (3) return potential answers when no object satisfy simultaneously all specified constraints.

The proposed approach is currently 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 [18, 17] that could be included in the database, as well as methods that would allow to extract automatically information concerning packages from the web, since manually entering this information is time-consuming and can only be done by an expert.

Concerning the support decision tool, we are planing to link it with a version of PassiveMap able to perform uncertainty analysis and to provide interval-valued or possibilistic valued optimal permeance. The uncertainty analysis will also be able to cope with ill-known values of vegetables parameters, using classical uncertainty propagation techniques [1].

REFERENCES

- [1] C. Baudrit and D. Dubois. Comparing methods for joint objective and subjective uncertainty propagation with an example in a risk assessment. In *Proc. Fourth International Symposium on Imprecise Probabilities and Their Application (ISIPTA'05)*, pages 31–40, Pittsburg (USA, Pennsylvanie), 2005.
- [2] 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.
- [3] 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.
- [4] P. Bosc, C. Brando, A. HadjAli, H. Jaudoin, and O. Pivert. Semantic proximity between queries and the empty answer problem. In *IFSA/EUSFLAT Conf.*, pages 259–264, 2009.
- [5] P. Bosc, A. HadjAli, and O. Pivert. Incremental controlled relaxation of failing flexible queries. *J. Intell. Inf. Syst.*, 33(3):261–283, 2009.
- [6] 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.
- [7] P. Bosc, L. Lietard, and O. Pivert. *Fuzziness in Database Management Systems*, chapter Fuzzy theory techniques and applications in database management systems, pages 666–671. Academic Press, 1999.

- 0981 [8] P. Bosc and O. Pivert. SQLf: a relational database language for fuzzy
0982 querying. *IEEE Transactions on fuzzy systems*, 3(1):1–17, February
0983 1995.
0984
0985 [9] P. Bosc and O. Pivert. About bipolar division operators. In *Flexible*
0986 *Query Answering Systems, 8th International Conference, FQAS 2009,*
0987 *Roskilde, Denmark, October 26-28, 2009. Proceedings*, volume 5822
0988 of *Lecture Notes in Computer Science*, pages 572–582. Springer, 2009.
0989 [10] P. Bosc, O. Pivert, A. Mokhtari, and L. Lietard. Extending relational
0990 algebra to handle bipolarity. In *Proceedings of the 2010 ACM Sympo-*
0991 *sium on Applied Computing (SAC), Sierre, Switzerland, March 22-26,*
0992 *2010*, pages 1718–1722. ACM, 2010.
0993 [11] P. Bosc, O. Pivert, and G. Smits. A model based on outranking for
0994 database preference queries. In *IPMU*, pages 95–104, 2010.
0995 [12] N. Bruno, S. Chaudhuri, and L. Gravano. Top-k selection queries over
0996 relational databases: Mapping strategies and performance evaluation.
0997 *ACM Trans. Database Syst.*, 27(2):153–187, 2002.
0998 [13] P. Buche and O. Haemmerlé. Towards a unified querying system of
1000 both structured and semi-structured imprecise data using fuzzy views.
1001 In *Proceedings of the 8th International Conference on Conceptual*
1002 *Structures, Lecture Notes in Artificial Intelligence #1867*, pages 207–
1003 220, Darmstadt, Germany, August 2000. Springer-Verlag.
1004 [14] J. T. Cacioppo, W. L. Gardner, and G. G. Berntson. Beyond bipolar
1005 conceptualizations and measures: The case of attitudes & evaluative
1006 space. *Personality & Social Psychology Review*, 1:3–25, 1997.
1007 [15] F. Charles, J. Sanchez, and N. Gontard. Modeling of active modified
1008 atmosphere packaging of endives exposed to several postharvest tem-
1009 peratures. *Journal of Food Science*, 8:443–448, 2005.
1010 [16] J. Chomicki. Preference formulas in relational queries. *ACM Trans.*
1011 *Database Syst.*, 28(4):427–466, 2003.
1012 [17] S. Destercke, D. Dubois, and E. Chojnacki. Unifying practical uncer-
1013 tainty representations: II clouds. *Int. J. of Approximate Reasoning (in*
1014 *press)*, 2007.
1015 [18] S. Destercke, D. Dubois, and E. Chojnacki. Unifying practical uncer-
1016 tainty representations: I generalized p-boxes. *Int. J. of Approximate*
1017 *Reasoning (In press)*, 2008.
1018 [19] D. Dubois and H. Prade. *Possibility Theory: An Approach to Comput-*
1019 *erized Processing of Uncertainty*. Plenum Press, New York, 1988.
1020 [20] D. Dubois and H. Prade. *Fuzziness in Database Management Systems*,
1021 chapter Tolerant fuzzy pattern matching: An introduction. Physica-
1022 Verlag, 1995.
1023 [21] D. Dubois and H. Prade. Bipolarity in flexible querying. In *Flexible*
1024 *Query Answering Systems, 5th International Conference, FQAS 2002,*
1025
1026
1027
1028
1029

- 1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
- Copenhagen, Denmark, October 27-29, 2002, Proceedings*, volume 2522 of *Lecture Notes in Computer Science*, pages 174–182. Springer, 2002.
- [22] D. Dubois and H. Prade. An overview of the asymmetric bipolar representation of positive and negative information in possibility theory. *Fuzzy Sets and System*, 160:1355–1366, 2008.
- [23] D. Dubois and H. Prade. An overview of the asymmetric bipolar representation of positive and negative information in possibility theory. *Fuzzy Sets Syst.*, 160:1355–1366, 2009.
- [24] 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.
- [25] E. Klement, R. Mesiar, and E. Pap. *Triangular Norms*. Kluwer Academic Publisher, Dordrecht, 2000.
- [26] 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.
- [27] 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.
- [28] 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.
- [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.

1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127

- [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.

UMR IATE

E-mail address: sdestercke@gmail.com

UMR IATE

E-mail address: patrice.buche@supagro.inra.fr

UMR IATE

E-mail address: guillard@univ-montp2.fr