Querying Fuzzy Multidimensional Databases: Unary Operators and their Properties

Anne Laurent

International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems
Vol. -, No. - (-) 000—000

QUERYING FUZZY MULTIDIMENSIONAL DATABASES: UNARY OPERATORS AND THEIR PROPERTIES

Anne Laurent
LIP6
Université Paris 6
8, rue du capitaine Scott
F-75015 PARIS
FRANCE
Anne.Laurent@lip6.fr

In this paper, we study the properties of flexible queries in the OLAP (*On-Line Analytical Processing*) framework, focusing on unary operators. For this purpose, we consider the model we have defined for fuzzy multidimensional databases. This model provides means to handle fuzzy data and flexible queries. The operators defined in this model are closed on the set of fuzzy hypercubes (hereafter cubes), which means that the result of each operator on a fuzzy cube is a cube. Thus, these operators can be nested into expressions. In this paper, the combination of several queries is investigated in order to study the possibility for the definition of an algebra to manipulate fuzzy cubes. This would provide a framework for query rewriting and, as a result, for query optimization.

Keywords: Fuzzy Multidimensional Queries, Successive Queries, Combination of Queries.

1. Introduction

OLAP (standing for *On-Line Analytical Processing*) is a recent framework devoted to the fast analysis of multidimensional information[1]. It consists in a set of technologies to collect, store, and treat multidimensional data for analysis. It has appeared that the relational database model is not suitable for this type of applications devoted to analysis, and the multidimensional model has emerged to support OLAP[1]. The multidimensional model provides a framework to deal with data as multidimensional arrays.

However, no model is able to deal with fuzziness, neither for the representation of ill-defined data, nor for the handling of flexible queries. For this reason, a model for fuzzy multidimensional data representation and manipulation has been proposed[2]. In this work, several operations are introduced with a view to apply them to discover relevant knowledge from fuzzy multidimensional databases. For this process, called *Fuzzy-OLAP Mining*[3], successive queries (including combinations) are needed.

In this paper, we limit ourselves to unary operator definitions, and we study their properties when combining them. We aim at defining rules for query rewriting and optimization. The final goal of this study is to provide the model with an algebra to manipulate queries efficiently in a simple way. Binary operators are out of the

scope of this paper but they have to be studied in order to deal efficiently with several fuzzy hypercubes.

In this paper, properties of operation combinations are studied in the way followed in papers on the relational algebra[4]. The paper is organized as follows. Section 2 presents multidimensional databases and their fuzzy extension. Section 3 focuses on the choice of the operators in the model we propose. Sections 4 and 5 detail successive roll-up, slice, dice and projection operations. Section 6 deals with the combination of queries.

2. Multidimensional Databases

Multidimensional databases have emerged to support OLAP. They are devoted to the efficient treatment of large amounts of multidimensional data for analysis. There is no unique model for the definition of multidimensional databases[5]. The next Section presents some common properties of all existing models.

2.1. *Classical Model*

Roughly speaking, a multidimensional database is a set of hypercubes. Each hypercube is defined by means of dimensions. A dimension of particular interest is chosen to be the *measure* and its values are stored inside the hypercube, in the cells. Hierarchies may be defined on dimensions in order to deal with several levels of granularity. Fig. 1 shows a cube describing sale results by means of three dimensions $PRODUCT$, $DISTRICT$ and $MONTH$.

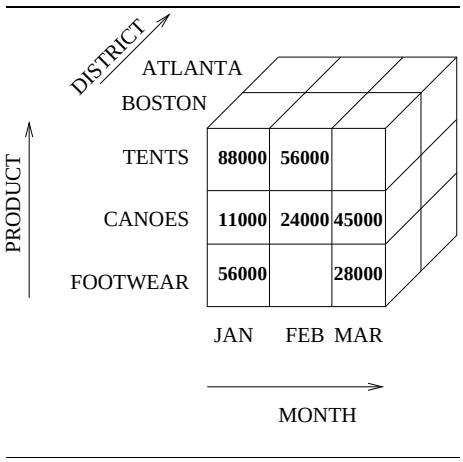

Fig. 1. Example of a classical cube

Operations are defined in order to manipulate hypercubes. The unary operators are slice and dice (selection of information), projection (selection of dimensions) and roll-up (navigation through levels of hierarchies by means of an aggregation function).

The slice operation consists in selecting slices from the cube by using a criterion on a dimension, for instance to select sales that have occurred in ATLANTA. The dice operation consists in selecting a subset of cells matching a criterion on the measure, for instance to select sales between $50,000$ and $80,000$ units. The

projection operation consists in deleting one or several dimension(s), for instance to describe sale results by means of only two dimensions instead of three. The roll-up operation consists in describing the cell values at some higher level of granularity on a dimension, for instance by rolling a cube up from the level of months to the level of quarters. The sum, mean or any other function may be considered in order to get the sale results at this new level of granularity.

However, data from the real world are often imperfect, either because they are imprecise (the age is *young*), or because they are uncertain (the age is *perhaps* 18). A model has been defined for multidimensional databases in order to represent and manipulate these data[2]. This model handles also flexible multidimensional queries. For instance, the subset of data corresponding to the age *young* may be extracted.

2.2. *Fuzzy Model*

A fuzzy multidimensional database contains elements which may be imperfectly defined, imprecise and/or uncertain. Given a reference set ($\mathbb{R}$ for instance), an *element* is a pair (v, d) where v is a value which may be precise (25) or defined by a fuzzy set (for example *young*) and d ($d \in [0, 1]$) is the confidence degree associated with this value. A *domain* is a finite set of elements.

A dimension is defined on a given domain. The pair $(v(d_i), d(d_i))$ represents an element d_i of dimension D_i.

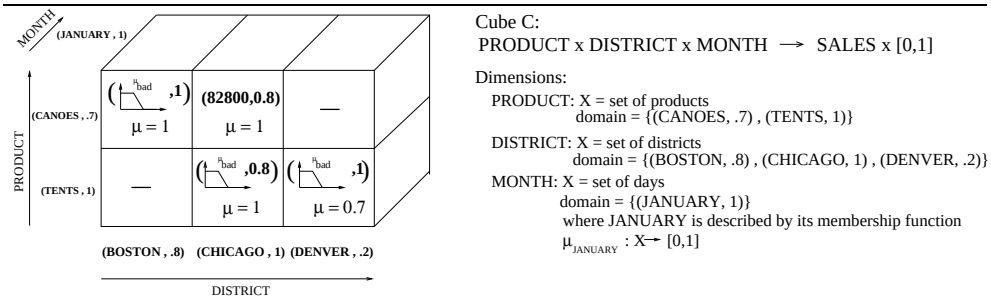

Fig. 2. Example of a fuzzy cube

A fuzzy cube is defined as a mapping:

$$D_1 \times \ldots \times D_k \rightarrow D_C \times [0, 1]$$

where $D_1, \ldots, D_k$ are dimensions, and D_C is the measure. A degree ranging from 0 to 1 is associated to each cell element to denote the extent to which the cell belongs to the cube. This degree is denoted by μ.

Thus the model handles degrees indicating to which extent entities (slices of the cube and cells) belong to the cubes. These degrees may come from the data themselves (*e.g.*, degrees of reliability of heterogeneous sources providing data) or from the fuzzy operations successively applied to the cube (selection of values matching a fuzzy criterion).

$\overrightarrow{x}$ denotes the cell identified by a set of values describing its position on all the dimensions. For instance, a cell is identified by its position $(Tents, January, Atlanta)$ on dimensions $PRODUCT$, $MONTH$, $DISTRICT$. An element in a cell $\overrightarrow{x}$ of a

cube is denoted $(v(\overrightarrow{x}), d(\overrightarrow{x}))$ where $v(\overrightarrow{x})$ is the value contained in the cell, and $d(\overrightarrow{x})$ the degree of confidence associated with this value.

Moreover, each cell $\overrightarrow{x}$ is associated with a degree μ corresponding to the extent to which this cell belongs to the cube. This degree thus applies on the whole information given by the dimension positions and the cell value. An example is given by Fig. 2.

As for relational databases, there may be two definitions of the equivalence of two cubes, based either on the structure of the cube (mapping), or on the content of the cube (element values). In the sequel of this paper, we consider two fuzzy cubes as *equivalent* if the following conditions are fulfilled:

- the 2 cubes are defined on the same dimensions,
- active domains of all the dimensions are the same (the elements taken from the cubes are the same ones -values and degrees)
- the 2 cubes contain exactly the same cells (same elements, same degrees μ).

For the sake of simplicity, a query is said to be *equivalent to another* if it produces an equivalent cube.

Operations are defined, providing tools to manipulate cubes by means of flexible OLAP queries. They are briefly described below and will be further studied in Section 3. For each operation, C denotes the cube on which the operation is performed, and C' the resulting cube.

Fuzzy Dice. The first type of query is a selection over the cells of the cube. The degrees μ of membership of cells are modified according to the satisfaction of the value $v(\overrightarrow{x})$ in the cell, the former degree $\mu_C(\overrightarrow{x})$ and the degree of confidence $d(\overrightarrow{x})$. The satisfaction of the value to the criterion O is computed by means of a so-called *measure of satisfiability* $\mathcal{C}$[6]. We denote by μ_O the membership function of criterion O. For each cell $\overrightarrow{x}$, the membership degree after selection is computed as follows:

$$\mu_{C'}(\overrightarrow{x}) = \mathcal{T}(\mathcal{C}(\mu_O, v(\overrightarrow{x})), d(\overrightarrow{x}), \mu_C(\overrightarrow{x})) \tag{1}$$

where $\mathcal{T} : [0,1] \times [0,1] \times [0,1] \to [0,1]$ is defined for instance from a t-norm $\top$ as $\mathcal{T}(x,y,z) = \top(x, \top(y,z))\ \forall (x,y,z) \in [0,1]^3$. Note that in our model, the chosen formula does not take the degree of confidence into account when comparing the value to the criterion. Another solution would be to merge first the value with its degree of confidence but we do not adopt this solution. The degree of confidence behaves as an upper bound, or a threshold.

Also note that for the fuzzy sets we consider, membership degrees are not interpreted as possibility degrees, but as degrees of truth. No disjunctive interpretation is supposed. This is the reason why theory of possibility (with necessity/possibility measures) is not carried.

Fuzzy Slice. The second type of query is a selection over some dimension. For each value d_i of the considered dimension, the new degree $d_{C'}(d_i)$ is computed according to the former degree $d_C(d_i)$ and the extent to which the value satisfies the selection criterion O. Therefore, for each d_i of the dimension, we have:

$$d_{C'}(d_i) = \top(\mathcal{C}(\mu_O, v(d_i)), d_C(d_i)) \tag{2}$$

Fuzzy Roll-Up. This operation consists in the generalization of a cube at a higher level of granularity. Choosing a dimension and a hierarchy on this dimension, rolling a cube up results in a modification of the active domain of the dimension: values are taken from the new level of granularity, and degrees of membership of each

slice are computed according to the degrees of the children of each value from the former cube. Cell elements are modified according to the function of aggregation. For instance, a *count* function may be chosen in order to summarize a cube at a high level of granularity. Computations are detailed below.

Let C be a fuzzy cube to roll-up, and dim the dimension to be generalized. Let h be the considered hierarchy, defined by a fuzzy relation R. The roll-up operation is performed from the level of granularity L_1 to the level L_2, using the function ϕ. Let C' be the resulting cube. We have:

$$C \xrightarrow[roll-up]{(dim,h,L_1,L_2,\phi)} C'$$

For each value b on dimension dim from level L_2, coefficients indicating to which extent each value a from level L_1 generalizes into b are considered. If the hierarchy is defined by means of a fuzzy relation, then these coefficients are computed using the transitive closure of R. If the hierarchy is defined by means of fuzzy partitions, then these coefficients are computed using the degree of membership of each value a to the fuzzy set describing b. $\mathbf{c(a,b)}$ denotes the coefficient of transition between values $a \in L_1$ and $b \in L_2$.

In the resulting cube C', the degrees μ in cells and the degrees on slices are modified. They are computed according to previous values and coefficient of transition between values from levels L_1 and values from level L_2.

For each $b \in L_2$, $\mu_{C'}(..., b, ...)$ is a function of

- the degrees $\mu_C(..., a, ...)$ with $a \in L_1$, and
- the values $c(a, b)$.

We have:

$$\mu_{C'}(..., b, ...) = \max_{a \in L_1} \min(c(a, b), \mu_C(..., a, ...))$$

Concerning slice degrees, the same operation is performed. Note that each element e from the domain of the dimension dim to be rolled up is like $(v(e), d(e))$. For each element e_2 in the resulting cube such that $v(e_2) = b$ and $b \in L_2$, we have:

$$d_{C'}(e_2) = \max_{\{e_1 | v(e_1) = a\}} \min(c(a, b), d(e_1))$$

Note that drill-down is not altered by the introduction of fuzziness.

Fuzzy Projection. This operation is not modified by the extension of the model to fuzziness. The projection is possible only if the domains of all dimensions to destroy (the ones that are not selected) are reduced to a singleton[7]. This means that a roll-up operation may be required before applying a projection.

3. Further Study of the Operators

This section focuses on the choice of operators for equations (1) and (2). For this purpose, both the semantic of the operators and the efficiency of computations have to be taken into account. Indeed, fast computations are required, due to the huge amount of data. For this reason, we require our model to be consistent with the classical case, in order to deal with the classical case as often as possible, letting the classical multidimensional database management system work.

3.1. *Comparing the value with the criterion*

Here, we focus on the choice of the measure $\mathcal{C}$ used to compare the value to the criterion. We need comparison measures between fuzzy subsets to determine to which extent two fuzzy values are similar. In our case, we will have to evaluate the *satisfiability* of a value to a reference description (a fuzzy criterion described by its membership function). This requires a particular kind of measure of resemblance which must enable the measurement of the inclusion of a (classical or fuzzy) description V into a reference REF and must produce a single value in order to be easily understandable. Such measures are called *measures of satisfiability* [6]. We use the notation $\mathcal{C}$ for these operations. In the proposed model, the chosen measure is the following one:

$$\mathcal{C}(V, REF) = \begin{cases} \frac{M(V \cap REF)}{M(V)} & \text{if } M(V) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

where

$$M(A) = \begin{cases} \sum_{x \in \Omega} f_A(x) & \text{if } \Omega \text{ is countable} \\ \int_\Omega f_A(x)dx & \text{otherwise} \end{cases}$$

This measure is chosen because it is consistent with the classical case. It is equivalent to the classical case when considering a precise value. This property is interesting because it enhances performances when dealing with precise data. The proof of consistency is very easy:

We denote by μ_{REF} and μ_V the membership functions describing the criterion REF and the data V. If the data is precise then it is a singleton $\{x_0\}$ such that for all $x \neq x_0$, we have $\mu_V(x) = 0$ and $\mu_V(x_0) = 1$. Thus we have $M(V \cap REF) = \mu_{REF}(x_0)$ and $M(V) = 1$. The computation is thus reduced to the computation of the membership degree of x_0 to the fuzzy set representation of the criterion: $C(V, REF) = \mu_{REF}(x_0)$

3.2. *Aggregation operator*

Most of the existing relational fuzzy querying models consider precise data and fuzzy criteria[8]. In our model, we have to take into account first the fact that both data and criteria may be fuzzy, and second the fact that all the successive operators applied are closed on the set of fuzzy cubes. Thus in order to compute the degree μ of each cell after a dice operation, we have to aggregate the degrees representing:

- the appropriateness of the value to the vague criterion,
- the degree of confidence in the value of the cell, and
- the degree μ from the former cube.

The properties below have to be fulfilled by this operator for the semantic of the operation, and for computational reasons.

Level of membership. After a selection, a cell cannot belong to the cube at a higher degree than the previous one or than the satisfaction of the criterion. This leads to consider an operator such that for all $(x, y, z) \in [0, 1]^3$, $\mathcal{T}(x, y, z) \leq min(x, y, z)$. As a consequence, 0 is an absorbent element: $\mathcal{T}(x, y, z) = 0$ if x, y or z equals 0. A t-norm $\top$ is suitable.

Monotonicity (decreasing). The result of the aggregation must decrease when decreasing one or several degrees: $\mathcal{T}(x', y', z') \leq \mathcal{T}(x, y, z)$ if $x' \leq x$ and $y' \leq y$ and $z' \leq z$.

Idempotence. Idempotence may be required in order not to alter the membership degree of a cell when the degree of appropriateness for the value is the same as the former degree μ. This would lead to the choice of the t-norm min, which is the only idempotent one, to construct $\mathcal{T}$. This choice is very interesting when considering successive selections by the same criterion, since it guarantees the stability of the result. Any other t-norm would modify the resulting degree in a counter-intuitive way. This is the reason why the current implementation of the system uses this t-norm min. We have thus $\forall x \in [0,1]$, $\mathcal{T}(x,x,x) = x$.

However, in the case of successive selections by different criteria, it would be also interesting to weaken the degree μ each time a selection is performed to data which do not completely fulfill the criterion. For this reason, a t-norm like the product may be considered.

Commutativity and associativity. These properties allow the system to enhance performances since it can optimize the order of computations. T-norms are commutative and associative, thus the t-norms min and product, among others, are suitable.

Concerning the *slice* operation, the $\mathcal{C}$ measure and the $\top$ operator have to be chosen. As previously, we argue that a slice cannot belong to the cube with a higher degree than the previous degree or than the degree of satisfaction of the criterion. This would be counter-intuitive to obtain a degree higher than the former one after a selection operation. Thus depending on whether the idempotence is required or not, the chosen t-norm is either min or product (these operators are both common and available in classical multidimensional database management systems).

3.3. *The Need for Successive Queries*

This paper is devoted to the study of successive queries. This study is motivated by several reasons.

First, Fuzzy OLAP Mining requires the application of combinations of queries. For instance, building a fuzzy decision tree requires successive slice, successive dice in order to build the successive partitions of the data set. Moreover, successive operations and combination of queries are needed when navigating through the data, especially when dealing with hierarchies for successive roll-up operations. Finally, this study is motivated from the theoretical point of view in comparison with the classical relational and multidimensional models, where the study of operator properties is a classical task, especially for query optimization[4].

4. Successive Roll-Up Operations

This section focuses on roll-up. This operator aggregates the data from a level of granularity to a higher one (for instance from the level of districts to the level of regions). An operator ϕ is thus required in order to merge cell values (*e.g.* sum, average). We study here the result of two successive roll-up operations on the same dimension, for a hierarchy defined by a fuzzy relation.
We are given:

- dim the dimension to roll up,
- R the fuzzy order relation defining the hierarchy (we denote by f_R the membership function of R and by f_{R_T} the membership function of its $max - min$ transitive closure),
- $\{L_i\}$ levels of granularity from the hierarchy (for all i, $L_i \subseteq dom(dim)$),
- and ϕ the function used to merge cell values.

We compare the results of operations (3) and (4).

$$C \xrightarrow[roll-up]{(dim,R,L_1,L_2,\phi)} C' \xrightarrow[roll-up]{(dim,R,L_2,L_3,\phi)} C_1 \tag{3}$$

$$C \xrightarrow[roll-up]{(dim,R,L_1,L_3,\phi)} C_2 \tag{4}$$

Proposition. Let C be a cube to be rolled up by function ϕ on dimension dim structured by hierarchy R. Let L_1 be the level of granularity of C, and L_2 and L_3 two higher levels of granularity. Degrees on slices and μ degrees in cells are the same by direct roll-up (from L_1 to L_3 in cube C_2) or indirect roll-up (first from L_1 to L_2 in cube c' and then from L_2 to L_3 in cube C_1):

- For all $\overrightarrow{x}$, we have $d_{C_2}(\overrightarrow{x}) = d_{C_1}(\overrightarrow{x})$ and $\mu_{C_2}(\overrightarrow{x}) = \mu_{C_1}(\overrightarrow{x})$.
- For all $c_n \in L_3$, we have $d_{C_2}(c_n) = d_{C_1}(c_n)$.

Proof. Let us consider query (3), we have:
$\forall c_n \in L_3,\ d_{C_1}(c_n) = \max\limits_{b_l \in L_2} \min(\max\limits_{a_k \in L_1} \min(d(a_k), f_{R_T}(a_k, b_l)), f_{R_T}(b_l, c_n))$

Considering query (4), we have:
$\forall c_n \in L_3,\ d_{C_2}(c_n) = \max\limits_{a_k \in L_1} \min(d(a_k), f_{R_T}(a_k, c_n))$

Besides,
$f_{R_T}(a_k, c_n) = \max\limits_{b_l \in L_2} \min(f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n))$

Thus $d'(c_n) =$
$\max\limits_{a_k \in L_1} \min(d(a_k), \max\limits_{b_l \in L_2} \min(f_{R_T}(a_k, b_l), f_{R_T}(b_l, c_n)))$

Hence we have $d_{C_2}(c_n) = d_{C_1}(c_n)$

The same holds for degrees μ.
□

Aggregate functions*used for the construction of cubes and the computation of super-aggregates (aggregates of aggregates) are of several kinds[9]. Aggregate functions can be distributive (*e.g. count*, *min*, *max*), algebraic (*e.g. average*), or holistic (*e.g. median*). The two first categories of functions allow optimizations while the third one requires a direct computation from the whole database. For instance, the *minimum* value can be easily computed using a lower level, since it is associative. For counting, a *sum* has to be applied on the intermediate level, while *average* requires two values to be handled (*count* and *sum*). However in this paper, neither the construction of the cubes nor the semantic of the measures are studied. Thus this question is out of the scope of this paper. Note that the problem described here is not due to the introduction of fuzziness and that is the

*Note that *aggregate* refers here to the function defined to compute a summary value of cell values, as used in the classical relational model. This must not be confused with the aggregation of degrees in the Fuzzy Logic framework.

same in classical multidimensional databases. Only associative aggregate functions (which are parts of distributive ones) result in the equality of equations (3) and (4). Associativity guarantees that values obtained directly by computing the resulting cube from level L_1 to level L_3 are the same as using an indirect computation by first rolling the cube up to level L_2.

5. Successive Slice, Dice and Projection Operations

This section focuses on the methods for query rewriting and studies the rules that are applicable in our framework. These properties are quite the same as in the relational framework and guarantee that some optimization may be done[4].

5.1. *Successive Dice Operations*

Due to the commutativity property of t-norms, the result of two successive selections on the cells is independent of the order:

$$C \xrightarrow[dice]{\gamma_1} C' \xrightarrow[dice]{\gamma_2} C_1 \tag{5}$$

is equivalent to:

$$C \xrightarrow[dice]{\gamma_2} C'' \xrightarrow[dice]{\gamma_1} C_2 \tag{6}$$

Proposition. Given C_1 the cube obtained by applying a selection on the cells of the cube by the criterion γ_1 followed by a selection by the criterion γ_2, and C_2 the cube obtained by applying a selection on the cells of the cube by the criterion γ_2 followed by a selection by the criterion γ_1, we have $C_1 = C_2$.

Proof. This amounts to show that each cell contains the same degree μ since this degree is the only one that is modified by a dice operation. μ_{γ_1} (resp. μ_{γ_2}) denotes the membership function of γ_1 (resp. γ_2). Due to associativity and commutativity properties of t-norms, we have:

$$\begin{aligned}
&\forall \overrightarrow{x}, \mu_{C_1}(\overrightarrow{x}) \\
&= \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), d(\overrightarrow{x}), \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))) \\
&= \mathcal{T}(\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x})), \mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}), d(\overrightarrow{x})) \\
&= \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}), d(\overrightarrow{x}), \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))) \\
&= \mu_{C_2}(\overrightarrow{x})
\end{aligned}$$

□

Remark. Note that the only way to get $C_1 = C'(= C'' = C_2)$ by applying twice the same selection ($\gamma_1 = \gamma_2$) is to consider an idempotent t-norm.

Indeed, if for each cell $\overrightarrow{x}$, $\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_\gamma), d(\overrightarrow{x}), \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_\gamma), d(\overrightarrow{x}), \mu_C(\overrightarrow{x})))$ is equal to $\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_\gamma), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$ then $\mathcal{T}$ is necessarily built using the t-norm min.

5.2. *Successive Slice Operations*

The proof of the commutativity is very simple, and it is quite the same as the previous one for the dice operation. When applying a selection on a dimension by criteria γ_1 and γ_2, the order has no effect on the resulting cube:

$$C \xrightarrow[slice]{(D,\gamma_1)} C' \xrightarrow[slice]{(D,\gamma_2)} C_1 \tag{7}$$

is equivalent to:

$$C \xrightarrow[slice]{(D,\gamma_2)} C'' \xrightarrow[slice]{(D,\gamma_1)} C_2 \tag{8}$$

Proposition. Denoting C the first cube, D the dimension on which the selection is applied, $dom(D)$ its domain, C_1 (resp. C_2) the cube obtained by selection first by criterion γ_1 (resp. γ_2) and then by criterion γ_2 (resp. γ_1).

Proof.
$\forall d_i \in dom(D)$,

$$\begin{aligned} d_{C_1}(d_i) &= \top(\mathcal{C}(v(d_i), \mu_{\gamma_2}), \top(\mathcal{C}(v(d_i), \mu_{\gamma_1}), d_C(d_i))) \\ &= \top(\top(\mathcal{C}(v(d_i), \mu_{\gamma_2}), \mathcal{C}(v(d_i), \mu_{\gamma_1}), d_C(d_i))) \\ &= \top(\mathcal{C}(v(d_i), \mu_{\gamma_1}), \top(\mathcal{C}(v(d_i), \mu_{\gamma_2}), d_C(d_i))) \\ &= d_{C_2}(d_i) \end{aligned}$$

□

Remark. Note that, as mentioned above, the only way to get the same cube by applying twice the same selection ($\gamma_1 = \gamma_2$) is to consider an idempotent t-norm. If for each slice $d_i \in dom(D)$, $\top(\mathcal{C}(v(d_i), \mu_\gamma), \top(\mathcal{C}(v(d_i), \mu_\gamma), d(d_i)))$ is equal to $\top(\mathcal{C}(v(d_i), \mu_\gamma), d(d_i))$ then $\top$ is the t-norm min.

5.3. *Cascade of Projections*

Considering a set of dimensions $\{A_1, ..., A_n, B_1, ..., B_m\}$ such that $\{A_1, ..., A_n\} \subseteq \{B_1, ..., B_m\}$, we know that the two queries (9) and (10) are equivalent. This is analogous to the classical model by Agrawal and al.[7]:

$$C \xrightarrow[Proj]{\{B_1,...,B_m\}} C' \xrightarrow[Proj]{\{A_1,...,A_n\}} C_1 \tag{9}$$

$$C \xrightarrow[Proj]{\{A_1,...,A_n\}} C_2 \tag{10}$$

Indeed, the projection is the operation consisting in selecting a set of dimensions by destroying the other ones (whose active domains have to be reduced to singletons). In this case, since the set of A_i's is among the set of B_i's, limiting directly the dimensions to the A_i's leads to the same result as limiting first to the B_i's and then to the A_i's.

5.4. *Splitting Criteria*

This section studies if there are equivalences between successive queries using two criteria and a single query using the conjunction of these criteria.

5.4.1. *Cascade of Dice Selections*

We study here if query (11) is equivalent to query (12), where $\wedge$ stands for the conjunction of criteria.

$$C \xrightarrow[dice]{\gamma_1} C' \xrightarrow[dice]{\gamma_2} C_1 \tag{11}$$

$$C \xrightarrow[dice]{\gamma_1 \wedge \gamma_2} C_2 \tag{12}$$

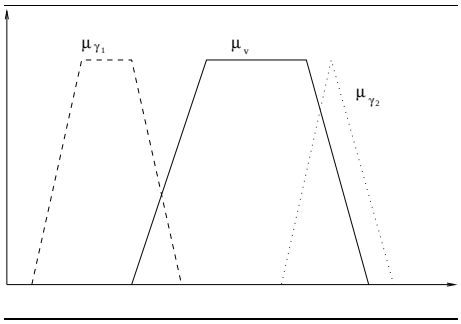

Fig. 3. A Counter Example

We show below that μ_{C_1} is not equal to μ_{C_2} in any case (see Fig. 3). A counter-example may be considered. In the case the following conditions are true, μ_{C_1} is different from μ_{C_2}:

- $\gamma_1 \cap \gamma_2 = \emptyset$,
- $v \cap \gamma_1 \neq \emptyset$,
- $v \cap \gamma_2 \neq \emptyset$,
- $\mathcal{T}$ is built using a strict t-norm $\top$ (such that $\top(a, b) > 0$ if $a > 0$ and $b > 0$).

For each $\overrightarrow{x}$, we have:
$\mu_{C_2}(\overrightarrow{x}) = \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1 \wedge \gamma_2}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$
and $\mu_{C_1}(\overrightarrow{x}) = \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), d(\overrightarrow{x}), \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}), d(\overrightarrow{x}), \mu_C))$
$= \mathcal{T}(\top(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}), \mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2})), d(\overrightarrow{x}), \mu(\overrightarrow{x}))$
Thus $\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}) \neq 0$ and $\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}) \neq 0$.

Besides, $C(v(\overrightarrow{x}), \mu_{\gamma_1 \wedge \gamma_2}) = 0$ since $v(\overrightarrow{x}) \cap \gamma_1 \cap \gamma_2 = \emptyset$.

Thus $\top(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}), \mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2})) \neq C(v(\overrightarrow{x}), \mu_{\gamma_1 \wedge \gamma_2})$

However, successive selections on the cells are often performed in order to refine a criterion, and there is no need to apply two successive criteria having an empty intersection. We focus thus on the cases where $\gamma_1 \cap \gamma_2 \neq \emptyset$. Considering successive selections for the refinement of the selection criterion, we have $\gamma_2 \subseteq \gamma_1$. If the used t-norm to aggregate the degrees is not idempotent then there exist some examples where the result is different when applying successively the two criteria or by applying simultaneously $\gamma_1 \wedge \gamma_2 = \gamma_2$. If $\top$ is the t-norm min then queries (11) and (12) are equivalent:

Proposition. We consider $\mathcal{T}$ built using the idempotent t-norm $\top = \min$. Denoting C_1 the cube obtained by applying a selection on the cells of the cube by the criterion γ_1 followed by a selection by the criterion γ_2, and C_2 the cube obtained by applying a selection on the cells of the cube by the criterion $\gamma_1 \wedge \gamma_2$, C_1 is equal to C_2 if and only if $\gamma_2 \subseteq \gamma_1$.

Proof.
only if see above the counter-example
if

$$\mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), d(\overrightarrow{x}), \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x})))$$
$$= \mathcal{T}(\top(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), \mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1})), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$$

Besides, $\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}) \leq \mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1})$ since $\gamma_2 \subseteq \gamma_1$ and $\mathcal{C}$ is monotonous, being a measure of satisfiability. Thus we have:

$$\mu_{C_1}$$
$$= \mathcal{T}(\top(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), \mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1})), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$$
$$= \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_2}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$$
$$= \mathcal{T}(\mathcal{C}(v(\overrightarrow{x}), \mu_{\gamma_1 \wedge \gamma_2}), d(\overrightarrow{x}), \mu_C(\overrightarrow{x}))$$
$$= \mu_{C_2}$$
□

5.4.2. *Cascade of Slice Selections*

The problem is the same as previously, checking whether query (13) is equivalent to query (14), where $\wedge$ stands for the conjunction of criteria. The slice operation, applied twice on the same dimension, either by two successive criteria, or by their conjunction, results in the same cube only if the second criterion is included within the first one.

$$C \xrightarrow[slice]{(D,\gamma_1)} C' \xrightarrow[slice]{(D,\gamma_2)} C_1 \tag{13}$$

$$C \xrightarrow[slice]{(D,\gamma_1 \wedge \gamma_2)} C_2 \tag{14}$$

Proposition. Denoting C_1 the cube obtained by applying a selection on the cells of the cube by the criterion γ_1 followed by a selection by the criterion γ_2, and C_2 the cube obtained by applying a selection on the cells of the cube by the criterion $\gamma_1 \wedge \gamma_2$, C_1 is equal to C_2 iff:

- $\gamma_1 \subseteq \gamma_2$,
- $\top$ is idempotent ($\top = minimum$)

The proof is the same as previously.

6. Combining Operators

Next subsections focus on projection, slice and dice combination properties. Problems appear when dealing with roll-up. However, we insist on the fact that this is not due to the introduction of fuzziness. The points described here are of the same kind in the classical models.

6.1. *Commuting Slice and Dice*

The slice and dice operations do not modify the same entities in the cube. Thus queries (16) and (15) are equivalent.

$$C \xrightarrow[dice]{\gamma_1} C' \xrightarrow[slice]{(D,\gamma_2)} C_1 \tag{15}$$

$$C \xrightarrow[slice]{(D,\gamma_2)} C' \xrightarrow[dice]{\gamma_1} C_1 \tag{16}$$

Proposition. Slice and Dice operations can be applied successively without any effect of the order.

The proof of this property is obvious since these operations do not interact. The first operation modifies degrees $d(d_i)$ on slices d_i from the domain of dimension D while the second one modifies degrees μ inside cells.

6.2. *Dealing with projections when combining queries*

A projection operation results in the destruction of one or several dimensions. Moreover, note that the projection can only be performed if the dimension to destroy has an active domain containing a single value. Otherwise, a slice would have to be automatically chosen, which is not possible. Thus a slice operation reducing the active domain to a singleton may be required (or a roll-up). In this framework, the following queries are not *applicable*:

$$C \xrightarrow[proj]{\mathcal{D}\backslash\{D\}} C' \xrightarrow[slice]{(D,\gamma)} C_1$$

$$C \xrightarrow[proj]{\mathcal{D}\backslash\{D\}} C' \xrightarrow[roll-up]{(D,h,L_1,L_2,\phi)} C_1$$

where $\mathcal{D}$ stands for a set of dimensions, and $\mathcal{D}\backslash D$ stands for a set of dimensions that does not include dimension D.

6.2.1. *Commuting Selections and Projections*

$$C \xrightarrow[proj]{\mathcal{D}} C' \xrightarrow[slice]{(dim,\gamma)} C_1 \tag{17}$$

$$C \xrightarrow[slice]{(dim,\gamma)} C' \xrightarrow[proj]{\mathcal{D}} C_1 \tag{18}$$

A slice operation is not relevant if the dimension has been destroyed by a projection. If the projection and the slice do not apply on the same dimension, the order of the two successive operations has no effect on the resulting cube.

Proposition. If $\mathcal{D} \cap \{dim\} = \emptyset$ then a slice operation on dimension dim can be mixed with a projection on dimensions $\mathcal{D}$ whatever the order may be without modifying the result.

The proof of this property is obvious since the two operations (projection and slice) do not modify the same entities of the cube. Obviously, the same holds for dicing, without any restriction on the dimension(s) to be deleted by projection:

Proposition. A dice operation on dimension can be mixed with a projection whatever the order may be without modifying the result.

6.2.2. *Commuting Roll-Up and Projection*

Note that most of the time, a roll-up is necessary before a projection in order to reduce the active domain. If the dimension to roll-up is among the ones to be destroyed, then the roll-up must be performed first to the upper level where only one value is contained in the active domain. This level is usually called level ALL.

Proposition. If $\mathcal{D} \cap \{dim\} = \emptyset$ then a roll-up operation on dimension dim can be mixed with a projection on dimensions $\mathcal{D}$ whatever the order may be without modifying the result.

The proof of this property is obvious, since the projection operation does not modify the set of cells to be merged in order to compute the cube on the higher level of granularity.

6.3. *Commuting Roll-Up and Selection (slice or dice)*

Rolling a cube up modifies the active domain of a dimension to a higher level of granularity. It modifies the content of the cells (elements and μ degrees) as well. The compatibility of the criterion regarding the data has to be studied very carefully. For instance, selecting sales results by a criterion describing *medium* sales does not have the same effect depending on the level of granularity. This level of granularity depends on the roll-up operations previously applied. If the roll-up operation is applied using the aggregation function *count* or *sum*, the selection criterion is not compatible anymore with data after roll-up whereas it was compatible before.

For instance, we consider the cube describing cell results, Fig. 4 shows that the description of sales depends on the level of granularity. The sum of all units sold within a region is supposed to be greater than the number of units sold within a district. However, this definition depends on the function used to determine the cell values. If the average (or minimum, maximum, median ...) value is considered, then the definition of low, medium, and high sales does not depend on the level of granularity.

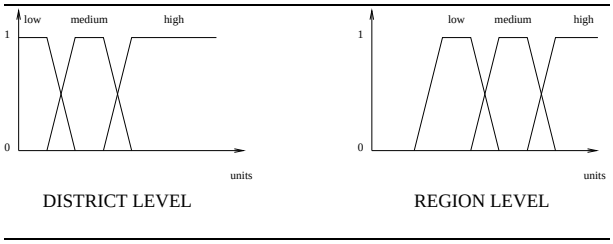

Fig. 4. Criteria Depending on the Granularity Level

7. Conclusion

In this paper, we study the properties of unary operators over fuzzy cubes. This work is twofold. First it aims at studying operators of aggregation in order to deal with degrees of confidence and degrees of membership. Second, this work aims at studying the combination of unary operators in order to define an algebra.

In most cases, properties from the classical relational and multidimensional models are preserved after the introduction of fuzziness. Moreover if a property is not preserved, it appears for particular cases that do not occur in the framework of data mining, which is the goal of the fuzzy multidimensional database model.

Current and future work concern the study of binary operators (union, intersection, difference, product, join). It will provide foundations to manipulate complex queries in an algebraic way for query rewriting. The system will thus be able to perform query optimization. The enhancement of the implementation of the operators introduced in this paper is also planned. The existing system *FUB* (standing for FUzzy cUBe) is implemented using `Oracle Express Server`, `Oracle Express Objects`, `Java` and `C++`[2].

References

1. E.F. Codd, S.B. Codd and C.T. Salley. Providing OLAP (On-Line Analytical Processing) to User-Analysts: An IT Mandate, *White Paper, Arbor Software Corp.*, 1993.
2. A. Laurent. Bases de données multidimensionnelles floues. in *Proc. Journées Bases de Données Avancées*, Cépaduès Editions, pp. 107–117, 2001.
3. A. Laurent, B. Bouchon-Meunier, A. Doucet. Towards Fuzzy-OLAP Mining. in *Proc. Work. PKDD "Database Support for KDD"*, pp. 51–62, 2001.
4. J.D. Ullman *Principles of Database Systems*, Pitman, 1980.
5. P. Vassiliadis and T. Sellis. A survey of logical Models for OLAP Databases, *SIGMOD Record*, Vol. 28, No. 4, 1999.
6. B. Bouchon-Meunier, M. Rifqi, and S. Bothorel. Towards general measures of comparison of objects. *Fuzzy Sets and Systems*, 84:143–153, 1996.
7. R. Agrawal, A. Gupta, and S. Sarawagi. Modeling Multidimensional Databases, in *Proc. of ICDE*, 1997.
8. P. Bosc and J. Kacprzyk (Ed.) *Fuzziness in Database Management Systems, Studies in fuzziness.* Springer-Verlag, 1995.
9. J. Gray, A. Bosworth, A. Layman and H. Pirahesh. Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tabs, and Sub-Totals. *Tech. Report MSRTR-95-22,* Microsoft, 1995.