A New Fuzzy Multidimensional Model

Carlos Molina, Lázaro Rodríguez-Ariza, Daniel Sánchez, and M. Amparo Vila

Abstract—As a result of the use of OLAP technology in new fields of knowledge and the merging of data from different sources, it has become necessary for models to support this technology. In this paper, we shall propose a new multidimensional model that can manage imprecision in both dimensions and facts and hide the complexity to the end user. The multidimensional structure is therefore able to model data imprecision resulting from the integration of data from different sources or even information from experts, which it does by means of fuzzy logic.

I. INTRODUCTION

EVER since the appearance of OLAP technology [1], there have been various proposals to support its special needs, and in particular, two different approaches have been documented. The first of these extends the relational model to support the structures and operations which are typical of OLAP, and the first proposal of such a type can be found in [2]. Since then, there have been other proposals [3], and most of the present relational systems include extensions to represent DataCubes and operate on them. The second approach is to develop new models using a multidimensional view of the data. Many authors have proposed models in this way [4]–[7].

In the early 1970s, the need for flexible models and query languages to manage the ill-defined nature of information in DSS was identified [8]. Nowadays, the application of OLAP technology to other knowledge fields (e.g., medical data) and the use of semistructured sources (e.g., XML) and nonstructured sources (e.g., plain text) has made these requirements on the models even more important.

It is now common for companies to require external data for strategic decisions. These data are not always compatible with the format of internal information, or if it is, it is not as reliable as internal data. Therefore, an architecture such as the one shown in Fig. 1 is desirable. Imprecision may be the result of the integration of information from different sources with different formats.

The systems now need to manage imprecision in the data, and more flexible structures are required to represent complex analysis domains and to integrate heterogenous sources.

If the dimensions cannot be flexibly defined, it is extremely difficult for certain domains to be modeled, and could result in information loss when we need to model complex domains or to merge data from different sources with incompatible formats.

In addition, the treatment of imprecision introduces complexity into the operations on the multidimensional structures. As Codd [1] mentioned, it is important to provide an intuitive way to interpret the results obtained in queries. If we consider imprecision in the model, the additional complexity which this entails must be hidden from the end user.

New models have attempted to treat this imprecision. In [9], the author proposes a model which is able to deal with DataCubes with an incomplete space (there are no data about some of the areas defined by the values in the dimensions). The model presented by Pedersen in [10] treats imprecision in dimensions by allowing the model to work with facts defined at different levels (not all the facts have the same granularity). The authors later extended the model to manage partial inclusion of elements in the hierarchies [11]. In order to answer a query, the system returns three solutions: the first considers all the related facts (to any degree), the second considers only the facts with a complete inclusion, and the third uses weights to aggregate the values. Imprecision in the facts is not treated.

Various approaches using fuzzy logic have also been explored. Laurent [12], for example, proposed a fuzzy multidimensional model which manages fuzzy facts and presents the use of fuzzy relations or fuzzy partitions in the dimensions. Details of the model and the properties of the operations can be found in [13]. In [14] and [15], the authors present a simple fuzzy multidimensional model mainly oriented towards association rule extraction in a multiagent system. Imprecision is considered by allowing fuzzy labels to be defined in dimensions (imprecise facts or other imprecise relations in dimensions are not defined). In [16], a model for spatiotemporal queries is proposed. The authors proposed the use of fuzzy logic to establish the degree of satisfaction of a space to a corresponding selection criterion. No other imprecision is considered.

Manuscript received November 15, 2004; revised June 8, 2005 and September 30, 2005.

C. Molina is with the Department of Computer Science, University of Jaén, Jaén, Spain (e-mail: carlosmo@ujaen.es).

L. Rodríguez-Ariza is with the Department of Financial Economy and Accounting, University of Granada, Granada, Spain.

D. Sánchez and M. A. Vila are with the Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain.

Digital Object Identifier 10.1109/TFUZZ.2006.879984
Although all of the models presented treat imprecision, they do not hide the added complexity from the end user, and most of them have not been developed to be used interactively. In this paper, we shall present a model bearing in mind two main goals: first, management of imprecision in the definition of data and structure, and second, the model is intended to be used by a user so all the complexity in the management of imprecision must be invisible.

With this intention, we propose a new multidimensional model which handles imprecision in hierarchies and facts by using fuzzy logic. The use of fuzzy hierarchies enables the structures of the dimensions to be defined more intuitively to the end user, thereby allowing a more intuitive use of the system. This also helps information to be merged from different sources with incompatible formats, or even information given by experts to be used in order to improve the multidimensional schema. All the complexity is hidden from the user by using a layer on the model by means of user views. A first approach of the model can be found in [17].

The paper is organized as follows. Section II contains a brief introduction to classical multidimensional models. Section II formalizes a crisp multidimensional model as the basis for introducing imprecision into the model. Section IV presents the fuzzy multidimensional model (structure and operations) as an extension of the crisp one. Properties and user views are then presented. The paper ends with an example and the main conclusions.

II. CLASSICAL MULTIDIMENSIONAL MODELS

Although there is no standard multidimensional model, we shall briefly introduce the common characteristics of the first models proposed in literature. In classical multidimensional models, we can distinguish two different types of data: on one hand, we have the facts being analyzed, and on the other, the dimensions are the context for the facts. Hierarchies may be defined in the dimensions [3]–[6]. The different levels of the dimensions allow us to access the facts at different levels of granularity. In order to do so, classical aggregation operators are needed (maximum, minimum, average, etc.). Other models, which do not define explicit hierarchies on the dimensions, use other mechanisms to change the detail level [7], [18].

The model proposed by Gray et al. [2] uses a different approach. This model defines two extensions of the relational group by (rollup and cube) that are used to group the values during aggregation.

The models that define hierarchies normally use many-to-one relations, so one element in a level can only be grouped by a single value of each upper level in the hierarchy. This makes the final structure of a DataCube rigid and well defined in the sense that given two values of the same level in a dimension, the set of facts relating to these values have an empty intersection.

The normal operations (roll-up, drill-down, dice, slice, and pivot) are defined in almost all the models. Some of these define other operations so as to provide the end user with more functionality [4], [7], [18]. In Section III, we shall present a multidimensional model with explicit hierarchies to formalize the concepts for the fuzzy extension.

III. CRISP MULTIDIMENSIONAL MODEL

As aforementioned, we shall first define a crisp multidimensional model to formalize the notation and as a basis for introducing imprecision into the model. This model shall later be extended to include imprecision management. It is based on the concepts presented in the previous section, using an explicit definition of the hierarchies in the dimension. We introduce the concept of a DataCube’s history to prevent information loss when roll-up operations are applied, and to enable drill-down operations to be applied directly. As far as the authors are aware, this concept has never been used before.

A. Crisp Multidimensional Structure

We must first define the data structure. In the multidimensional structure, we can see two different types of data (as we mentioned before). We must, therefore, define the structures for the dimensions (including hierarchies) and facts.

1) Dimensions: The dimension is the context of the facts we want to analyze. In order to access the fact at different granularities, the hierarchies are defined on the dimension levels. Each level can be regarded as a set of names or labels that define a subset of elements for the lower levels in the dimension.

Definition 1: A dimension is a tuple $d = (l_i, l_j, l_k, Y)$ where $l_i = l_j$, $i = 1, \ldots, n$ so that each $l_i$ is a set of values $l_i = \{c_i, \ldots, c_n\}$ and $l_i \cap l_j = \emptyset$ if $i \neq j$, and $\leq d$ is a partial order relation between the elements of $l$ so that $l_i \leq l_j$ if $\forall c_i \in l_i \Rightarrow \exists l_j \in l_k$ $c_j \subseteq c_k$. $l_k$ and $l_T$ are two elements of $l$ so that $\forall l_i \in l_k \leq l_T \leq l_i \leq l_k$.

Each element $l_i$ is called a level. In order to identify level $l$ of dimension $d$, we shall use $d_l$. An element of level $d_l$ is noted as $l_i$. The two special levels $l_L$ and $l_T$ shall be called the base level and top level, respectively. The partial order relation in a dimension gives the hierarchical relation between the levels.

Fig. 2 shows a definition of an age hierarchy. The definition of the dimension as we have presented it would be Age = (Age, Group, legal age, All, ≤Age, Age, All) and

\[
\begin{align*}
\text{Age} & \leq \text{Age} \quad \text{Age} \\
\text{Group} & \leq \text{Age} \quad \text{Group} \\
\text{Legal age} & \leq \text{Age} \quad \text{Legal age} \\
\text{All} & \leq \text{Age} \quad \text{All} \\
\text{Age} & \leq \text{Age} \quad \text{Age} \\
\text{Group} & \leq \text{Age} \quad \text{All} \\
\text{Legal age} & \leq \text{Age} \quad \text{All}
\end{align*}
\]

The values which are defined in a level are the labels for the sets of values. The value “Yes” in level “Legal age” is a set “Y ∈ S = \{18, \ldots, 110\}

In level “All,” there is a single value. This value ("All") represents the set of all ages ("All" = \{1, \ldots, 110\}), so that “Yes” ∈ “All.” The set of all the labels or names in a dimension is called the domain.

Definition 2: For each dimension $d$, the domain is $\text{dom}(d) = \bigcup l_i$
In the above example, the domain of the dimension Age is dom(Age) = \{1, \ldots, 100\}, Young, Adult, Old, Yes, No, All\}. This set represents all the values and labels that define a dimension at all the detail levels.

**Definition 3:** For each \( l_i \) the set

\[
H_{l_i} = \{ l_j \mid l_j \neq l_i \wedge l_j \leq d_i \wedge \exists k \in \mathbb{N} \: l_j \leq d_k \leq d_i \} \tag{1}
\]

and we call this the set of children in level \( l_i \).

This set defines the set of all the levels which are below a certain level \( (l_i) \) in the hierarchy. Moreover, this set gives the set of levels whose values or labels are generalized by the ones included in \( l_i \). Using the same example of the dimension on the ages, the set of children in level \( All \) is \( H_{All} = \{ Group, Legal \: age \} \). In all the dimensions which we define, for the base level, this set will always be empty (as the definition shows).

**Definition 4:** For each \( l_i \) the set

\[
P_{l_i} = \{ l_j \mid l_j \neq l_i \wedge l_j \leq d_i \wedge \exists k \in \mathbb{N} \: l_j \leq d_k \leq d_i \} \tag{2}
\]

and we call this the set of parents in level \( l_i \).

This set represents the levels on level \( l_i \) in the hierarchy. For a certain level, this set shall give all the levels that group or generalize the values of the level. On the hierarchy we have defined, the set of parents in level \( Age \) is \( P_{Age} = \{ Legal \: age, Group \} \). In the case of the top level of a dimension, this set shall always be empty.

2) **Crisp Facts:** The facts are the domain variables which we want to analyze. The dimensions set the context to these variables, giving the position of each fact in the space defined by the dimensions.

**Definition 5:** A fact is a \( m \)-tuple on the attribute domain which we want to analyze.

The dimension levels define the level of detail at which we see the facts. If we go up the hierarchy, the level of detail decreases. In this process, we need aggregation operators to adapt the granularity of the facts. In the crisp case, an aggregation operator is a function, given a bag of values, whose result is a value that characterizes the bag (e.g., average, maximum, minimum, etc.).

**Definition 6:** Let \( B(X) \) be the set of all possible bags defined using values of \( X \), and \( D_x \) a natural or numerical domain, an Aggregation Operator \( G \) is a function \( G: B(D_x) \rightarrow D_x \).

When we apply an aggregation operator, we summarize the information about a bag of values into a single value. It is not always possible to undo these operations. Should we wish to undo an operation that reduces the level of detail in a DataCube, we would therefore need something to prevent this problem, and so we define the object history.

**Definition 7:** An object of the type history is the recursive structure

\[
H^0 = \Omega \\
H^{n+1} = (A, l_h, F, G, H^n) \tag{3}
\]

where \( \Omega \) represents the end of the recursion; \( F \) is the fact set; \( l_h \) is a set of levels \( (l_{h1}, \ldots, l_{h_k}) \). \( A \) is an application from \( l_h \) to \( F(A: l_h \rightarrow F) \). \( G \) is an aggregation operator.

This structure enables detail levels of the DataCube to be restored while it is operated on so that it may be restored to a previous level of granularity. We shall examine how this structure is managed at a later stage when we present the operations.

3) **Crisp Datacube:** Once we have defined the structure of the dimensions and facts, we can introduce the DataCube structure. A DataCube can be considered to be the union of a set of facts (the variable to analyze) and a set of dimensions (the context of the analysis). In order to report the facts and dimensions, we need an application which, for each combination of values of the dimension, gives the facts related to these coordinates in the multidimensional space defined by the dimensions. In addition to these DataCube features, there are also the levels which establish the detail levels to which the facts are defined, and a history-type object to keep the aggregation states during the operations. The DataCube is therefore defined in the following way.

**Definition 8:** A DataCube is a tuple \( C = (D, l_h, F, A, H) \) such that \( D = (l_{h1}, \ldots, l_{hn}) \) is a set of dimensions, \( l_h = (l_{h1}, \ldots, l_{h_k}) \) is a set of levels such that \( l_{h_k} \) belongs to \( d_k \), \( F = R \cup \emptyset \) where \( R \) is the set of facts and \( \emptyset \) is a special symbol, \( A \) is an application defined as \( A: l_{h1} \times \cdots \times l_{hn} 
\rightarrow F \), giving the relation between the dimensions and the facts defined, \( H \) is an object of type history.

If for \( \overrightarrow{d} = (a_1, \ldots, a_n) \), \( A(\overrightarrow{d}) = \emptyset \), this means that no fact is defined for this combination of values. Normally, not all the combinations of level values have facts defined. This situation is shown by the symbol \( \emptyset \) when application \( A \) is defined.

The basis of the analysis will be a DataCube defined at the most detailed level. We shall then refine the information while operating on the DataCube. This DataCube is basic.

**Definition 9:** We say that a DataCube is basic if \( l_h = (l_{h1}, \ldots, l_{hn}) \) and \( H = \Omega \).

**B. Operations**

Having defined the crisp structure, we shall now present the normal operations on the multidimensional structure. We shall first introduce the operations that change the granularity of the DataCube: roll-up and drill-down.

1) **Roll-up:** When we progress up the hierarchies, we need to know the facts related with a value at a certain level in order to aggregate this data. The set of facts shall be obtained using the relations between values in different levels. We shall call this set \( F_{c_{ij}} \), for a value \( c_{ij} \).
Definition 10: For each value \( c_{ij} \) belonging to \( I_r \), we have the set

\[
F_{c_{ij}} = \left\{ \left. h \mid \exists k_p \in I_k \wedge h \in F \wedge \exists h' \in F' A(h') = h \right\} \text{ if } I_r \neq I_b
\]

\[
= \left\{ \left. h \mid h \in F \wedge \exists \overline{c} \cdot A(\overline{c}) = h \right\} \text{ if } I_r = I_b
\]

(4)

where \( \overline{c} = (\overline{c}_1, \ldots, \overline{c}_j, \ldots, \overline{c}_l) \).

Using this set, we can begin defining the roll-up operation.

If we want to access the data by reducing the level of detail, we must go up the hierarchies defined on the dimensions. In order to do so, we shall use the roll-up operation.

Definition 11: The result of applying roll-up on dimension \( d_i \), level \( I_{r'}(I_{r'} \neq I_{rb}) \), using the aggregation operator \( G \) on a DataCube \( C = (D, I_b, F, A_H) \) is another DataCube \( C' = (D, I_{r'}, F', A', H') \) where \( I_b' = (I_{rb}, \ldots, I_{rb}) \).

\[
A' = A(\overline{c}_1, \ldots, \overline{c}_j, \ldots, \overline{c}_l) = G \left( \left\{ h \mid h \in F_r \wedge \exists h' \in F' A(h') = h \right\} \right)
\]

(5)

is the range of \( A' \); \( F' = (A, I_{rb}, F, G, H) \).

In these operations, we use the history object so that the results may be undone. When this operation is applied, we define a new set of facts by aggregating the facts which are obtained from the original DataCube. If we go up a dimension \( d_i \) from level \( I_{rb} \) to \( I_{r'} \), each new fact is obtained by aggregating the facts with all the equal coordinates except the one relative to dimension \( d_i \). In this position, a value \( c_{ij} \) will appear from level \( I_{rb} \). The value will be the aggregation of the facts of the original DataCube defined using a value \( c_{ij} \) related to value \( c_{ij} \) in level \( I_{rb} \) and with the same values for the remaining coordinates.

If we want to undo this aggregation, we must restore the previous state of the DataCube using the history structure. In Section IV, we shall present the drill-down operation which behaves accordingly.

2) Drill-down: When we want to go down through the hierarchies, we apply drill-down. With this operation, we return to a previous state of the DataCube with a more detailed level. This operation undoes the changes made by a previous roll-up operation.

Definition 12: The result of applying drill-down on a DataCube \( C = (D, I_b, F, A_H) \) where \( H = (A', I_{rb}, F', H') \) is another DataCube \( C' = (D, I_{rb}, F', A', H') \).

If we do not use the history structure, there is not enough information to recover the aggregated values for each fact. Using this object, the operation is easy to define.

3) Dice: Basic DataCubes are defined to contain as much information as possible about a domain to be analyzed. However, most of the time we want to analyze a subset of this domain. The dice operation does this by selecting an area of multidimensional space defined in the DataCube. When this operation is applied with a condition \( \beta \) on a level, we obtain a modification of the dimension, reducing the values in all the levels to those values of the level being considered which satisfy the condition. The domain of the new dimension will therefore be a subset of the domain of the original one. For each value, we must obtain the values of the level where dice is applied which are related to it. If any of the values satisfies the condition, we consider the value for the new dimension. In any other case, the value is eliminated.

Definition 13: The result of applying dice with the condition \( \beta \) on level \( I_r \) of dimension \( d_i \) in a DataCube \( C = (D, I_b, F, A_H) \) is another DataCube \( C' = (D, I_{r'}, F', A', H') \) where \( I_{r'} = (I_{rb}, \ldots, I_{rb}) \).

\[
d_i' = \begin{cases} I_{rb} & \text{if } I_{r'} \neq I_{rb} \\ I_{rb} \cup \{d_1, \ldots, d_i, \ldots, d_n\} & \text{if } I_{r'} \subseteq I_r \end{cases}
\]

\[
A' = \left( c_{ij} \mid I_r \not\subseteq I_{rb} \right)
\]

(6)

This operation obtains a new DataCube as a restricted space of the original one, so that the history component is set to \( \Omega \) (no previous state of detail).

4) Slice: We sometimes need to reduce the dimensionality of the domain so as to reduce the context of the facts, using fewer dimensions to access them. This is done by means of a slice operation. As we reduce the dimensionality, new facts are defined in a less detailed level. In order to do so, we must aggregate the facts to obtain the new ones. If we eliminate a dimension, new facts are obtained by aggregating the facts defined at the same values for the remaining dimensions.

Definition 14: The result of applying slice on dimension \( d_i \) using the aggregation operator \( G \) in a DataCube \( C = (D, I_b, F, A_H) \) is another DataCube \( C' = (D, I_{r'}, F', A', H') \) where \( I_{r'} = \{d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_n\} \) and shown in the equation at the bottom of the page:

\[
A' = \left( c_{ij} \mid c_{ij} \in d_i \wedge \beta(c_{ij}) \right)
\]

(7)

\[
d_i' = \begin{cases} I_{rb} & \text{if } I_{r'} \neq I_{rb} \\ I_{rb} \cup \{d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_n\} & \text{if } I_{r'} \subseteq I_r \end{cases}
\]

This is the case in which this operation obtains a new DataCube as a subspace in another way. The history structure is set to \( \Omega \) for this reason (it is a new analysis space).

5) Pivot: This operation only changes the order of accessing the dimensions. Although it does not alter the facts, it does entail a redefinition of the multidimensional space.

Definition 15: The result of applying pivot on the dimensions \( d_i \) and \( d_j \) in a DataCube \( C = (D, I_b, F, A_H) \) is another DataCube \( C' = (D, I_{r'}, F, A', H') \) where \( I_{r'} = \{d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_{j-1}, d_{j+1}, \ldots, d_{n}\} \) and shown in the equation at the bottom of the page:

\[
A' = \left( c_{ij} \mid c_{ij} \in d_i \wedge \beta(c_{ij}) \right)
\]

(8)

\[
d_i' = \begin{cases} I_{rb} & \text{if } I_{r'} \neq I_{rb} \\ I_{rb} \cup \{d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_{n}\} & \text{if } I_{r'} \subseteq I_r \end{cases}
\]

C. Valid Datacube

We have defined what a DataCube is and the operation on it. We must now define when a DataCube-type structure represents a valid DataCube to operate on.

Definition 16: A DataCube is valid if it is basic or if it has been obtained by applying a finite number of operations on a basic DataCube.

IV. FUZZY MULTIDIMENSIONAL MODEL

After the formalization of crisp DataCubes, we introduce fuzzy logic into the multidimensional model in order to manage imprecision in the structure and operations. Following the same schema, we first define the fuzzy multidimensional structure and then the operations on this structure.
A. Fuzzy Multidimensional Structure

1) Fuzzy Dimension: In the crisp case, an element in a level can be grouped by a single element in the upper level. In the fuzzy case, an element can be related with more than one element in the upper level and the degree of this relation is in the interval \([0,1]\). The kinship relation defines this degree of relationship.

Definition 17: For each pair of levels \(I_i\) and \(I_j\) such that \(I_j \in H_{ik}\), we have
\[
\mu_{ij}: I_i \times I_j \rightarrow [0,1]
\]
and we call this the kinship relation.

The degree of inclusion of elements in a certain level in the elements of their parent levels can be defined using this relation. If we only use the values 0 and 1 and only one element is included with degree 1 for a single element of its parent levels, this relation represents a crisp hierarchy. Following the example, the relation between levels Legal age and Age is of this type. The parent relation in this situation is
\[
\mu_{\text{Legal age, Age}}(\text{Yes}, x) = \begin{cases} 
1 & \text{if } x \in [18, 100] \\
0 & \text{in other case}
\end{cases}
\]
\[
\mu_{\text{Legal age, Age}}(\text{No}, x) = \begin{cases} 
1 & \text{if } x \in [1, 17] \\
0 & \text{in other case}
\end{cases}
\]

If we relax these conditions and allow values to be used in the interval \([0,1]\) without any other limitation, we have a fuzzy hierarchical relation. This allows several hierarchical relations to be represented more intuitively. An example can be seen in Fig. 3 where we present the group of ages according to linguistic labels. This fuzzy relation also enables hierarchies to be defined in which there is imprecision in the relationship between elements in different levels. In such a situation, the value in the interval shows the degree of confidence in the relation.

Using the relation between elements in two consecutive levels, we can define the relation between each pair of values in different levels in a dimension.

Definition 18: For each pair of levels \(I_i\) and \(I_j\) of the dimension \(d\) such that \(I_j \subseteq d \setminus I_i\), the relation \(\eta_{ij}: I_i \times I_j \rightarrow [0,1]\) is defined as shown in (5) at the bottom of the page, where \(\otimes\) and \(\oplus\) are a t-norm and a t-conorm, respectively, or operators from the families MOM and MAM defined by Yager [19], which include the t-norms and t-conorms, respectively. This relation is called the extended kinship relation.

We consider the MAM and MOM family of operators because they establish the properties which characterize aggregations of the type and or, respectively. We shall now examine the corresponding definitions.

Definition 19: [19] The function \(H\), defined as
\[
H: \tilde{B}(I) \rightarrow I
\]
is a MAM operator if it satisfies the following properties:
- If \(A \geq B\) then \(H(A) \geq H(B)\).
- If \(D = A \cup B\) then \(H(D) \geq H(A) + H(B)\).

Definition 20: [19] The function \(G\), defined as
\[
G: \tilde{B}(I) \rightarrow I
\]
is a MOM operator if it satisfies the following properties:
- If \(A \geq B\) then \(G(A) \geq G(B)\).
- If \(D = A \cup B\) then \(G(D) \geq G(A) + G(B)\).

These families of operators include the t-norm and t-conorm functions, respectively. For a demonstration, see [19].

This relation gives us information about the degree of relation between two values in different levels in the same dimension. In order to obtain this value, it considers all the possible paths between the elements in the hierarchy. Each is calculated by aggregating the kinship relation between elements in two consecutive levels using a MAM function. The final value is then the aggregation of the result of each path using a MOM function. By way of example, we shall show how to calculate the value of \(\eta_{\text{All, 25}}\) in this situation. In this way, we have two different paths.

Let us consider each one.

- All—Legal age—Age. Fig. 4(a) shows the two ways to get to 25 from All passing through level legal age. The result of this path is \((1 \otimes 1) \oplus (1 \otimes 0)\).
- All—Group—Age. This situation is very similar to the previous one. Fig. 4(b) shows the three different paths going through level Group. The result of this path is \((1 \otimes 0.7) \oplus (1 \otimes 0.3) \oplus (1 \otimes 0)\).

We must now aggregate these two values using a MOM function in order to obtain the result. If we use the maximum as the MOM function and the minimum as the MAM function, the result is
\[
(((1 \otimes 1) \oplus (1 \otimes 0)) \oplus ((1 \otimes 0.7) \oplus (1 \otimes 0.3) \oplus (1 \otimes 0))) = (1 \otimes 0) \oplus (0.7 \otimes 0.3) \oplus (1 \otimes 0) = 1\oplus 0.7 = 1
\]
so the value of \(\eta_{\text{All, 25}}\) is 1, which means that the age 25 is grouped by All in level All with grade 1.

\[
\eta_{ij}(a, b) = \begin{cases} 
\mu_{ij}(a, b) & \text{if } I_j \in H_i \\
\otimes_{k \in H_i} \otimes_{c \in H_k} (\mu_{ik}(a, c) \otimes \eta_{kj}(c, b)) & \text{in other case}
\end{cases}
\]
2) Fuzzy Facts: In the fuzzy case, we consider that facts also have an imprecision degree in the interval \([0, 1]\). This is the same approach followed in [12].

Definition 21: We say that any pair \((h, \alpha)\) is a fact when \(h\) is an \(m\)-tuple on the attribute domain which we want to analyze, and \(\alpha \in [0, 1]\).

The value \(\alpha\) controls the influence of the fact in the analysis. Data imprecision is managed by assigning an \(\alpha\) value which represents this imprecision. When we operate with the facts, the aggregation operators must manage these values in the calculations. The arguments for the operator can be seen as a fuzzy bag since they are a set of values with a degree in the interval \([0, 1]\) that can be duplicated. Yager extends the classical bag to a fuzzy concept [20]. A crisp bag \(E\) on the domain \(D_x\) is characterized by a function \(\text{card} \_ E\) defined as

\[
\text{card} \_ E : D_x \to \mathbb{N}
\]

(8)

where for a value \(x \in D_x\) the value \(\text{card} \_ E(x)\) represents the number of occurrences of element \(x\) in the bag \(E\). In the fuzzy case, a bag \(\tilde{E}\) is characterized using a crisp bag on the domain \(D_x \times [0, 1]\). By way of example, if we consider the fuzzy bag \(\tilde{E} = \{0.2/\alpha_a, 0.5/\alpha_b, 0.1/b, 0.2/a, 0.3/a, 0.7/d\}\), then \(\text{card} \_ (\alpha_a, 0.2) = 2\), which means that the element \(a\) appears twice in the fuzzy bag with membership 0.2. A more complex characterization of fuzzy bags can be found in [21].

The result of aggregation must also be a fact. The result must then present the same structure of a fact: a value of a domain and a value in the interval \([0, 1]\). In the fuzzy case, aggregation operators may, therefore, be defined in the following way.

Definition 22: If \(\tilde{B}(X)\) is all the possible fuzzy bags defined using elements in \(X\), \(\tilde{P}(X)\) the fuzzy power set of \(X\), and \(D_x\) a numerical or natural domain, we define an aggregation operator \(G\) as a function \(G : \tilde{B}(D_x) \to \tilde{P}(D_x) \times [0, 1]\).

One example of aggregation operators which are valid for the fuzzy bags are those proposed by Rundensteiner and Bic [22]. We consider these operators because they are consistent in the sense that when they are applied on data without imprecision (in our case, all the kinship relations and \(\alpha\)-values are equal to 1), they behave as classical ones. In order to use this proposal in our model, we must adapt the definition.

Definition 23: Let \(R\) be an operator defined by Rundensteiner and Bic [22], and \(\tilde{F}\) a fuzzy bag on the facts. We define the operator \(G_R\) as \(G_R(\tilde{F}) = (R(\tilde{F}), 1)\). To illustrate this fact, let \(\tilde{F}' = \{\alpha/h\}\) such that \((h, \alpha) \in \tilde{F}\).
B. Operations

In this section, we shall only present the operations that must be redefined to support imprecision. We must first redefine the set \( F'_{c_{ij}} \) so that the kinship relation is considered. A fact belongs to \( F'_{c_{ij}} \) if there is a path in the dimension that connects the value \( c_{ij} \) and the fact with all the kinship relations greater than 0.

Definition 25: For each value \( c_{ij} \) belonging to \( I_r \), we have the set shown in

\[
F_{c_{ij}} = \left\{ \frac{h_{k'}}{c_{ij}} \in F_{k'} \wedge \mu_{k'}(c_{ij}, c_{ij}) > 0 \quad \text{if} \quad l_r \neq h_r
\right\}
\]

where \( c' = (c_1, \ldots, c_{ij}, \ldots, c_n) \).

1) Roll-up: The definition of the roll-up operation is similar to the one previously proposed but also considers the kinship relation and the \( \alpha \) value of the facts when aggregating values.

Definition 26: The result of applying roll-up on dimension \( d_r \), level \( l_r \) (\( l_r \neq l_1 \)), using the aggregation operator \( G \) on a DataCube \( C' = (D, l_r, F', A, H) \) is another DataCube \( C'' = (D, l'_r, F', A', H') \) where \( l'_r = (l_{1r}, \ldots, l_{r-1}, l_{r+1}, \ldots, l_n) \);

\[
A'(c_1', \ldots, c_{ij}', \ldots, c_n') = G(\{(b, \alpha) \otimes \eta_{k'}(c_1', c_2') \mid (b, \alpha) \in F_{c_{ij}} \wedge A(c_1, \ldots, c_{ij}, \ldots, c_n) = (b, \alpha)\}); F' is the range of \( A', H' = (A, l_r, F, G, H) \).

This definition extends the crisp definition because we must consider the imprecision of values in the dimensions involved. Imprecision is measured using the extended kinship relation and is combined with the \( \alpha \)-value of the facts. The result of this aggregation gives a single measure of the imprecision of each fact for a set of coordinates. The selection of function \( \otimes \) will depend on the nature of the imprecision managed and the analysis domain, and it will, therefore, be a design decision.

If the extended kinship relations involved are very low, the influence of the fact on the analysis will also be very low. In this situation, the result will not be unduly influenced by whether a fact is aggregated or not, although there will be an increase in the complexity of the calculation process. We can therefore define a threshold for the extended kinship relation to consider the most influential facts. Following this approach, we defined the set \( F^\alpha_{c_{ij}} \).

Definition 27: Let \( \alpha \in [0,1] \). For each value \( c_{ij} \) belonging to \( l_r \) we have the set shown in (10) at the bottom of the page where \( c' = (c_1, \ldots, c_{ij}, \ldots, c_n) \).

The \( F^\alpha_{c_{ij}} \) groups all the facts related to a value in a dimension that have at least one relation of degree \( \alpha \). If we use a value \( \alpha = 1 \), then \( F^\alpha_{c_{ij}} \) will only include the facts that are related to the value \( c_{ij} \) without imprecision. In such a case, we reduce the model to a crisp approach (the fact is completely related or not). Formally, we obtain a simplification of the multidimensional schema using the threshold so that what is analyzed is a subset of the data. It must, therefore, be considered as a simplification in order to reduce complexity. If the global imprecision of the model is low (most of the hierarchical relations have high values as well as the facts), we can use a higher threshold value than if the global imprecision is high. In the second case, a high threshold implies that the subset of data considered could not be representative of the entire dataset since few facts are considered in the operations. It is, therefore, essential to select the threshold value carefully.

We can now define the roll-up\( ^\alpha \) in order to reduce the number of facts to aggregate to those with a greater relationship than \( \alpha \).

Definition 28: The result of applying roll-up\( ^\alpha \) on dimension \( d_r \), level \( l_r \) (\( l_r \neq l_1 \)), using the aggregation operator \( G \) on a DataCube \( C' = (D, l_r, F', A, H) \) is another DataCube \( C'' = (D, l'_r, F', A', H') \) where \( l'_r = (l_{1r}, \ldots, l_{r-1}, l_{r+1}, \ldots, l_n) \);

\[
A'(c_1', \ldots, c_{ij}', \ldots, c_n') = G(\{(b, \alpha) \otimes \eta^\alpha_k(c_1', c_2') \mid (b, \alpha) \in F^\alpha_{c_{ij}} \wedge A(c_1, \ldots, c_{ij}, \ldots, c_n) = (b, \alpha)\}); F' is the range of \( A', H' = (A, l_r, F, G, H) \).

It can clearly be seen that the definition is very similar to the previous one. The difference is that now we only consider the
2) Dice: When selecting values, we must also consider the kinship relation. We must, therefore, redefine the dice operation. The difference between this and crisp operation is that we reduce the space to those values with any relation (a kinship relation higher than 0) to the values that satisfy a condition $\beta$.

**Definition 29:** The result of applying dice with the condition $\beta$ on level $L_r$ of dimension $d_i$ in a DataCube $C = (D, h, F, A, H)$ is another DataCube $C' = (D', h', F', A', \Omega)$ where

$$D' = d_1, \ldots, d_n'$$

with $d_i' = (l_{j_i}) \leq d_i \leq l_{j_i}$ and

$$d_{i,j'} = \begin{cases} c_{j,k} \mid c_{j,k} \in l_j \land \beta(c_{j,k}), & \text{if } l' = l_r \\ c_{j,k} \mid c_{j,k} \in d_{i} \land \delta_{j'}(c_{j,k}), & \text{if } l' \leq d \leq l_j \\ c_{j,k} \mid c_{j,k} \in d_{i} \land \delta_{j'}(c_{j,k}), & \text{if } l_r \leq l' \leq l_j \\ \end{cases}$$

where $\delta_{j'}(c) = \exists x_r \in l_r \beta(c_r) \land \eta_{j'0}(c_{r \ast} c) > 0$; $A'(c_{j_1}', \ldots, c_{j_n}') = (h_r, \alpha)/c_1' \cdots d_{i} \land \delta_{j'}(c_{j,k}) = (h_r, \alpha)$; $F'$ is the range of $A'$.

As in the case of roll-up, in order to restring the space to those values with a relation over a certain threshold so as to reduce complexity, we define dice $\alpha$.

**Definition 30:** If $\alpha \in [0, 1]$, the result of applying dice$^\alpha$ with the condition $\beta$ on level $L_r$ of dimension $d_i$ in a DataCube $C = (D, h, F, A, H)$ is another DataCube...
There is no loss of information when applying \( \text{with} \) applied does not change the result.

Proof: By definition, drill-down only needs to be applied on a DataCube whose history structure is different from \( \Omega \); slice, pivot and dice set the history to this value. Roll-up is the only operation that sets this structure to another value.

4) Property 4: Pivot is associative with all the operations except drill-down.

Proof: If we apply pivot, the resulting DataCube has \( \Omega \) as its history structure. Drill-down cannot, therefore, be applied in this situation (Property 3). Pivot only changes the structure rather than the values, changing the order of the dimensions. This change is not important for the remaining operations, including itself. The order of application is therefore not important, and the result will be the same.

5) Property 5: Applying two dice operations with conditions \( \beta \) and \( \gamma \) is equivalent to applying only one with condition \( \beta \land \gamma \).

Proof: The result of applying a dice operation is to restrict the values to those that satisfy a condition. If we apply another dice operation, the resulting values are those that also satisfy the second condition. Since the final set of values will satisfy both conditions, it is equivalent to applying only one dice with the condition \( \beta \land \gamma \).

6) Property 6: The order in which two dice operations are applied does not change the result.

Proof: We prove this property by using the previous one. The and operator is commutative, so \( \beta \land \gamma \) is equivalent to \( \gamma \land \beta \). Applying the first dice with \( \beta \) and then with \( \gamma \) will, therefore, have the same effect as applying \( \beta \) after \( \gamma \).
VI. USER VIEW

We have presented a structure that manages imprecision by means of fuzzy logic. We need to use aggregation operators on fuzzy bags in order to apply some of the operations presented. Most of the methods previously documented give a fuzzy set as a result. As this situation can make the result difficult to understand and use in a decision process, we propose a two-layer model: one of the layers is the structure presented in the previous section; and the other is defined on this, the main objective of which is to hide the complexity of the model and provide the user with a more understandable result.

In order to do so, we propose the use of a fuzzy summary operator that gives a more intuitive result but which retains as much information as possible. As an example of this type of operator, we can use the one proposed in [23]. This operator proposes the use of the fuzzy number that best fits (as regards fuzziness) the fuzzy set or fuzzy bag. We can use simpler operators as the weighted average. By way of example, we shall apply both operators to the fuzzy bag \{1/1, 1/2, 0.2/0.5, 0.8/2.3, 0.2/0.3, 0.1/2.5\}.

- Linguistic summary. Using this operator, the result is \(1, 2, 1, 0.5\), the associated linguistic expression of which is “more or less between 1 and 2.”
- Weighted average. In this situation, the value shown to the user is 1.4.

It can be seen that in both cases the user obtains a more intuitive access to the results, with more information being lost in the second one due to its simplicity. Using this type of operator, we shall define the user view.

**Definition 31:** Given a summary operator \(M\), we define the user view of a DataCube \(C = (D, \mathbf{b}, F, A, H)\) using \(M\) as the structure \(C_M = (D, \mathbf{b}, F_M, A_M)\) where \(A_M(a_1, \ldots, a_n) = M(A(a_1, \ldots, a_n))\) and \(F_M\) is the range of \(A_M\).

Since we can define as many user views of a DataCube as the number of summary operators used, each user can therefore have their own user view with the most intuitive view of data according to their preferences by using a DataCube. Consequently,
TABLE IV
EXAMPLE FACT ON THE MULTIDIMENSIONAL SCHEMA FROM A SOURCE WITH UNCERTAINTY

<table>
<thead>
<tr>
<th>No.</th>
<th>Date</th>
<th>Age</th>
<th>Cause</th>
<th>Product</th>
<th>Amount</th>
<th>α</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>06/01/2003</td>
<td>19</td>
<td>Breaking</td>
<td>Video</td>
<td>2</td>
<td>0.7</td>
</tr>
<tr>
<td>2</td>
<td>22/01/2003</td>
<td>80</td>
<td>Missing spare</td>
<td>TV</td>
<td>1</td>
<td>0.7</td>
</tr>
<tr>
<td>3</td>
<td>07/02/2003</td>
<td>19</td>
<td>Missing spare</td>
<td>Video</td>
<td>4</td>
<td>0.7</td>
</tr>
<tr>
<td>4</td>
<td>19/02/2003</td>
<td>72</td>
<td>Missing spare</td>
<td>Video</td>
<td>3</td>
<td>0.7</td>
</tr>
<tr>
<td>5</td>
<td>28/02/2004</td>
<td>32</td>
<td>Breaking</td>
<td>TV</td>
<td>2</td>
<td>0.7</td>
</tr>
<tr>
<td>6</td>
<td>07/03/2004</td>
<td>19</td>
<td>Missing spare</td>
<td>CD player</td>
<td>2</td>
<td>0.7</td>
</tr>
<tr>
<td>7</td>
<td>11/03/2004</td>
<td>40</td>
<td>Slight damage</td>
<td>CD player</td>
<td>3</td>
<td>0.7</td>
</tr>
<tr>
<td>8</td>
<td>27/03/2004</td>
<td>23</td>
<td>Breaking</td>
<td>CD player</td>
<td>1</td>
<td>0.7</td>
</tr>
<tr>
<td>9</td>
<td>28/04/2004</td>
<td>23</td>
<td>Breaking</td>
<td>TV</td>
<td>5</td>
<td>0.7</td>
</tr>
<tr>
<td>10</td>
<td>30/04/2004</td>
<td>32</td>
<td>Slight damage</td>
<td>CD player</td>
<td>1</td>
<td>0.7</td>
</tr>
<tr>
<td>11</td>
<td>12/05/2004</td>
<td>40</td>
<td>Breaking</td>
<td>TV</td>
<td>2</td>
<td>0.7</td>
</tr>
<tr>
<td>12</td>
<td>28/05/2004</td>
<td>23</td>
<td>Slight damage</td>
<td>Radio</td>
<td>2</td>
<td>0.7</td>
</tr>
<tr>
<td>13</td>
<td>11/06/2004</td>
<td>40</td>
<td>Electric problem</td>
<td>TV</td>
<td>1</td>
<td>0.7</td>
</tr>
<tr>
<td>14</td>
<td>27/06/2004</td>
<td>23</td>
<td>Electric problem</td>
<td>CD player</td>
<td>1</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Fig. 11. Result of Query 1 in the crisp case.

users do not see the results as fuzzy sets because they work with their own user views (Fig. 5).

As Codd et al. showed in the 11th OLAP product evaluation rule [1], it is important to provide an intuitive way to interpret the result. Most of the time, users will find it easier to understand a diagram than a table with the results. Present systems use charts to show the result to the decision-maker. In our model, providing a graphical representation is even more important due to the fact that interpreting fuzzy values is complicated even for fuzzy logic experts.

We propose two methods to represent fuzzy numbers graphically as a user view. Both approaches are shown in Fig. 6. In Fig. 6(a) the approach followed is to use a color gradient to represent the membership degree of the values. The other approach [Fig. 6(b)] consists in changing the width of a bar to represent membership.

Both can be used to construct charts, and one example is shown in Fig. 7. This example represents fuzzy values related to crisp ones (the labels). In certain situations, representing fuzzy values related to fuzzy labels can prove interesting. By following the first approach, we can achieve this. The procedure followed, therefore, is to aggregate the membership values in both axes, using a t-norm, and use the result to build the color gradient. Fig. 8 shows an example of a chart where the labels are defined using linguistic labels.

If we use the second approach, we can construct fuzzy bar charts representing fuzzy values (Fig. 9).

VII. EXAMPLE

In this section, we shall present an example of multidimensional schema to show how to build a DataCube using our approach. We shall also resolve queries on the structure to show how they are applied. A crisp model of the schema is used to compare the results obtained.

Fig. 10 shows the multidimensional schema. It has been designed to analyze customer complaints about the company’s products. We have four dimensions: Customer, Product, Time and Cause. In order to improve the schema, we have used information given by experts. The level Severity in dimension Cause and the Quality of the provider in dimension Product represent this information.
The DataCube structure would be

$$C_{\text{Compaine}} = (\{\text{Customer, Product, Time, Cause}\},\{\text{Amount}\} \cup \emptyset, \Omega, A)$$

where dimensions are defined as

Customer = (\{\text{Age, Legal Age, Group, All}\}, \leq_{\text{Customer}}, \text{Age, All})

Product = (\{\text{Product, Provider, Quality, All}\}, \leq_{\text{Product, Product, All}})

Time = (\{\text{Date, Month, Year, All}\}, \leq_{\text{Time, Date, All}})

Cause = (\{\text{Cause, Severity, All}\}, \leq_{\text{Cause, All}})

The only element we need to complete the definition of the DataCube is application $A$. In this situation, its definition would be

$$A : \text{Age} \times \text{Product} \times \text{Date} \times \text{Cause} \rightarrow \{\text{Amount}\} \cup \emptyset$$

The structures of the dimensions common to both models are as follows.

- **Cause dimension:**
  - Cause breaking electric~problem, slight~damage
  - Severity = \{ light, serious \}
  - All = \{ all \} = \mathbb{I}_T

- **Time dimension:**
  - Date = \{Dec-01, Mar-01, \ldots, Feb-28-04\} = \mathbb{I}_L
  - Month = \{Dec-02, Jan-04, Feb-04\}
  - Year = \{2003, 2004\}
  - All = \{ all \} = \mathbb{I}_T
In the crisp case, we must complete the model obtaining the values grouped by each label in the levels:

- **Cause dimension:**
  - Severity: light = breaking, light-damage; serious = electric-problem missing spare
  - All: all = light | serious = breaking slight-damage, electric-problem missing spare

- **Time dimension:**
  - Month: Dec-03 = {Dec-01-03, ..., Dec-31-03}, Jan-04 = {Jan-01-04, ..., Jan-31-04}, Feb-04 = {Feb-01-04, ..., Feb-28-04}
  - Year: 2002 = Dec-03 = {Dec-01-03, ..., Dec-31-03}, 2004 = Jan-03 = {Jan-01-04, ..., Feb-28-04}
  - All: all = 2002 | 2003 = {Dec-01-02, ..., Feb-28-03}

- **Product dimension:**
  - Provider: Provider 1 = {radio, CD player}, Provider 2 = {TV, video}
  - Quality: good = Provider 1 = {radio, CD player}, bad = Provider 2 = {TV, video}
  - All: all = good | bad = {radio, CD player, TV, video}

- **Customer dimension:**
  - Legal age: yes = {18, ..., 100}, No = {1, ..., 17}
  - Group: young = {1, ..., 25}, adult = {26, ..., 64}, old = {65, ..., 100}
  - All: all = yes | No = young | adult | old = {1, ..., 100}

In this model, we shall use the classical aggregation operator (e.g., maximum, minimum, etc.).

This schema modeled using the fuzzy multidimensional model is different when we define three of the kinship relations:

- between the levels Severity and Cause (Table I) of the Cause dimension;
- Quality and Provider (Table II) of the Product dimension;
- and Group and Age (Fig. 3) in the Customer dimension.

In the example queries, we shall use the minimum as t-norm and the maximum as t-conorm to calculate the extended kinship relation. When we need to aggregate fuzzy values, we shall use the adapted operators defined by Rundensteiner and Bic presented in Definition 23.

In order to populate the DataCube, we use the data obtained within the company and the data provided by a distribution company about our products. The data obtained from the external source cannot be considered as reliable as the data from our company due to the fact that we do not know how the data is obtained and what criteria are applied in the process.

In the crisp case, we have two options. The first option is to disregard data from the external source, although this would mean that we would run the risk of making decisions based on a partial set of data. The other option is make the analysis using all the data, but here we would run the opposite risk of basing the decision on data that might be false. By using the fuzzy approach, however, we can choose another option, which is to use the data from the external source in the analysis but control its influence in the final result. The data obtained in our company shall be considered with an alpha of 1. While the information from the distribution company shall be considered, it is less important, and so we assign a value of 0.7 to the facts that use this information. The data used in this example are collected in Tables III and IV.

### A. Query 1

We want to know

“number of complaints according to the severity and the quality of the product provider”. 

In the example queries, we shall use the minimum as t-norm and the maximum as t-conorm to calculate the extended kinship relation. When we need to aggregate fuzzy values, we shall use the adapted operators defined by Rundensteiner and Bic presented in Definition 23.

In order to populate the DataCube, we use the data obtained within the company and the data provided by a distribution company about our products. The data obtained from the external source cannot be considered as reliable as the data from our company due to the fact that we do not know how the data is obtained and what criteria are applied in the process.

In the crisp case, we have two options. The first option is to disregard data from the external source, although this would mean that we would run the risk of making decisions based on a partial set of data. The other option is make the analysis using all the data, but here we would run the opposite risk of basing the decision on data that might be false. By using the fuzzy approach, however, we can choose another option, which is to use the data from the external source in the analysis but control its influence in the final result. The data obtained in our company shall be considered with an alpha of 1. While the information from the distribution company shall be considered, it is less important, and so we assign a value of 0.7 to the facts that use this information. The data used in this example are collected in Tables III and IV.

### A. Query 1

We want to know

“number of complaints according to the severity and the quality of the product provider”. 

In this model, we shall use the classical aggregation operator (e.g., maximum, minimum, etc.).

This schema modeled using the fuzzy multidimensional model is different when we define three of the kinship relations:

- between the levels Severity and Cause (Table I) of the Cause dimension;
- Quality and Provider (Table II) of the Product dimension;
- and Group and Age (Fig. 3) in the Customer dimension.

In the example queries, we shall use the minimum as t-norm and the maximum as t-conorm to calculate the extended kinship relation. When we need to aggregate fuzzy values, we shall use the adapted operators defined by Rundensteiner and Bic presented in Definition 23.

In order to populate the DataCube, we use the data obtained within the company and the data provided by a distribution company about our products. The data obtained from the external source cannot be considered as reliable as the data from our company due to the fact that we do not know how the data is obtained and what criteria are applied in the process.

In the crisp case, we have two options. The first option is to disregard data from the external source, although this would mean that we would run the risk of making decisions based on a partial set of data. The other option is make the analysis using all the data, but here we would run the opposite risk of basing the decision on data that might be false. By using the fuzzy approach, however, we can choose another option, which is to use the data from the external source in the analysis but control its influence in the final result. The data obtained in our company shall be considered with an alpha of 1. While the information from the distribution company shall be considered, it is less important, and so we assign a value of 0.7 to the facts that use this information. The data used in this example are collected in Tables III and IV.

### A. Query 1

We want to know

“number of complaints according to the severity and the quality of the product provider”.
In order to execute this query, a roll-up operation must be applied. We need to change the granularity using the levels Severity in the Cause dimension, and Quality in the Product dimension, using the Sum as the aggregation operator.

Before discussing the query results, we shall outline the main differences between using both approaches. In the crisp case, we must decide whether to include data from an external source in the analysis. We shall show the query result when this data is included and when it is not so that the results may be compared.

The answer to the query in both approaches is presented in Table V and Fig. 11 for the crisp model, and in Table VI and Fig. 12 for the fuzzy one.

The diagrams clearly show that, in the crisp case, different answers are obtained according to whether the external source is considered or not, with two results for the user to analyze. In the fuzzy model, however, we have a single answer that gives an overall view of the resulting data and which controls the influence of each source.

**B. Query 2**

The second query we want to answer is

"number of complaints made by young people each month"

In order to resolve this query, we must apply the following.

- **Dice** on the *Customer* dimension with condition \( \beta(x) = "x \text{ is Young}" \) in the level Group.
- **Roll-up** on the dimensions *Time*, level *Month*, and *Customer*, level *Group*, using the *Sum* aggregation operator.

In this query, the main difference between the crisp and fuzzy approach to the multidimensional model is the definition of the age group (Fig. 13). In the crisp case, the label *young* represents the interval of ages \([0, 25]\), while in the fuzzy model we use a fuzzy number with support \([0, 35]\) and kernel \([0, 20]\). The first approach is less intuitive for a user: a 25-year-old person is young but s/he can be considered more or less adult. The same problem occurs with values which are very close to the interval.
TABLE VIII
RESULT OF QUERY 2 IN THE FUZZY CASE

<table>
<thead>
<tr>
<th>Month</th>
<th>Number of complaints</th>
<th>$C_{LS}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>January</td>
<td>${1.0/1.0, 0.8/2.0, 0.7/4.0, 0.2/6.0},1$</td>
<td>(1,1,0,4.8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Slightly greater than 1&quot;</td>
</tr>
<tr>
<td>February</td>
<td>${1.0/2.0, 0.8/5.0, 0.7/9.0, 0.2/11.0},1$</td>
<td>(2,2,0,8.8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Slightly greater than 2&quot;</td>
</tr>
<tr>
<td>March</td>
<td>${1.0/5.0, 0.7/8.0, 0.2/11.0},1$</td>
<td>(5,5,0,6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Slightly greater than 5&quot;</td>
</tr>
<tr>
<td>April</td>
<td>${1.0/5.0, 0.7/10.0, 0.2/11.0},1$</td>
<td>(5,5,0,4.6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Slightly greater than 5&quot;</td>
</tr>
<tr>
<td>May</td>
<td>${1.0/6.0, 0.7/8.0, 0.2/12.0},1$</td>
<td>(6,6,0,5.6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Slightly greater than 6&quot;</td>
</tr>
<tr>
<td>June</td>
<td>${1.0/7.0, 0.7/8.0, 0.2/12.0},1$</td>
<td>(7,7,0,4.3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Slightly greater than 7&quot;</td>
</tr>
</tbody>
</table>

bounds: a 26-year-old person may be adult, but if 25 is young then only one year cannot be considered different enough to be included in a different group. A user will find it more intuitive to consider a 26-year-old person a bit older than 25 (which results in a membership which is lower in group young and higher in adult).

The results of this query are collected in Table VII for the crisp case, and Table VIII for the fuzzy model. The graphs for these tables are Figs. 14 and 15, respectively. Account should be taken of the differences in the crisp model according to whether the external source is considered or not. The user obtains different analyses for each situation and is faced with the problem of which one is correct. In the fuzzy modeling, we have an integrated view of the data. While we consider both sources, the influence of the external data is controlled. The user can now analyze the result and have an integrated answer to the query, viewing the tendency in the data (Fig. 16).

VIII. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a new multidimensional model. The main contribution of this new model is that it operates on data with imprecise facts and hierarchies, hiding the complexity from the end user. Classical models impose a rigid structure that makes it difficult for information from different sources to be merged if there are incompatibilities in the schemata. Our model, on the other hand, handles these problems by means of fuzzy logic which allows our proposal to help in the integration, relaxing the schemata in order to obtain a new one that covers the others and attempting to retain as much information as possible. In addition, the flexible structure can be used to incorporate information given by experts which is often imprecise. These data can be used to improve the multidimensional schema so that it may be used by the end user in the decision-making process. The use of fuzzy logic results in a complexity which is difficult for the end user to understand. In order to avoid this situation, we propose a layer on the multidimensional structure that hides this complexity from the end user and returns simpler and more intuitive results.

This model has been implemented within an OLAP Server prototype. The system is built using Java language and implements both crisp (using ROLAP and MOLAP approaches) and the fuzzy (pure MOLAP) multidimensional model presented. Oracle (9i and 10g) and PostgreSQL 7.4 have been tested to work with the system. We plan to apply the fuzzy multidimensional model with medical data. The next step is to study data mining techniques on the multidimensional model proposed. The integration of data mining and OLAP have been studied [24]–[26], resulting in what is called OLAP Mining or on-line analytical mining (OLAM). We would like to complete the model by adapting data mining techniques (e.g., [27], [28]) which would help in the extraction of useful information from
fuzzy DataCubes. Some efforts have already been made in this regard using fuzzy logic [14], [15], [29], [30].

REFERENCES


Carlos Molina was born in Granada, Spain, on October 15, 1979. He received the M.S. and Ph.D. degrees, both in computer science, in 2002 and 2005, respectively, from the University of Granada. He is an Assistant Professor with the Department of Computer Science, University of Jaén, Jaén, Spain, since 2004. His current main research interests are in the fields of multidimensional model, data mining, and soft computing.

Lázaro Rodríguez-Ariza received the M.S. and Ph.D. degrees, both in management and economic sciences, in 1982 and 1985, respectively, from the University of Malaga, Spain. He is a Full Professor with the Department of Financial Economy and Accounting, University of Granada, Granada, Spain, and Director of the Andalucía Business School Foundation.

Daniel Sánchez was born in Almería, Spain, on March 11, 1972. He received the M.S. and Ph.D. degrees, both in computer science, in 1995 and 1999, respectively, from the University of Granada, Granada, Spain. He has been an Associate Professor with the Department of Computer Science and Artificial Intelligence, University of Granada, since 2001. He has participated and is currently a member of the teams of several projects, and has published more than 30 papers in international journals and conferences. His current main research interests are in the fields of knowledge discovery and data mining, relational databases, information retrieval, fuzzy sets theory, and soft computing.

M. Amparo Vila received the Ph.D. degree in mathematics from the University of Granada, Granada, Spain, in 1978. She is Full Professor with the Department of Computer Science and Artificial Intelligence, University of Granada, where she leads the Intelligent Databases and Information Systems Research Group. Her main research interests are in the fields of database design, data mining, and mathematical theory.