Proceedings of 2013 International Conference on
Technology Innovation and Industrial Management
29-31 May 2013, Phuket, Thailand

PROVIDING SEMANTIC INTERPRETATION TO OLAP QUERIES. APPLICATION TO HOSPITAL MANAGEMENT DATA WAREHOUSES.

Carlos Molina, Dept. Informatics, University of Jaen, Spain
Email: carlosmo@ujaen.es

Belén Prados-Suarez, Dpt. Languages and Informatics Systems, University of Granada, Spain
Email: belenps@ugr.es

Miguel Prados de Reyes, San Cecilio Hospital, Granada, Spain
Email: prados@decsai.ugr.es

Carmen Peña Yañez, San Cecilio Hospital, Granada, Spain
Email: carmenpy@decsai.ugr.es

ABSTRACT

Purpose: *The aim of this paper is to present a new methodology to enrich the analysis over data cubes in an OLAP system, allowing the system to add semantic interpretation to the query results adapted to the final user.*
Methodology: *Defining a semantic translation of a given value regarding a concrete range of values can be done by means of fuzzy logic. In the case of querying data cubes, the possible combinations of dimensions, levels and aggregation functions makes this range different for each query, so a static interpretation is not valid. We propose a way to supply a semantic interpretation that adapts itself to the query considering the granularity and the semantic of the used function.*
Findings: *We propose a mechanism that allows the representation of different semantic interpretations in data-cubes, in addition to their adaptation when querying. With it users are automatically provided with personalized and elaborated results that are adapted and really meaningful for their individual requirements. In addition, the mechanism allows an individual (the meaning of a value is independent of the other values) or comparative interpretation (the semantic compare each value with the others in the query).*
Practical Implications: *The accessibility improvements achieved allow the creation of simpler interfaces and improve the query over data-cubes from mobile devices allowing the user to interpret the results without needing to review all the values for the queries.*
Originality/Value: *As far as the authors know, there are proposals to interpret values in a natural way for user, but none of them are able to adapt it to the structure of the query and the functions used, so the user is limited to the considered cases. In our approach, the system will adapt the semantic after each query, providing the user with an easy and interpretable result. Even more, the system can adapt the results to different users in a very simple way.*

INTRODUCTION

Everyday more enterprises and big organizations require advanced methods providing managers with elaborated and comprehensible information. It is especially relevant in the cases of organizations working over OLAP systems due to the immense amount of information that is stored using datacubes.

This problem is relatively recent so there are not a lot of proposals that face this problem. Most of the existing techniques are focused on presenting the information to the user in a comprehensible language for him/her; i.e. in natural language. This is the case of the linguistic summary methods, that analyze great amount of data to provide the user with results of the "Q of the X verify the property Y" structure, where Q is a quantifier. However this is not enough when the user needs the result of the query to be semantically meaningful.

An example of this situation takes place when a manager of a group of health centers has to evaluate the performance of the medical doctors. This manager may query about the number of patients that are attended by a given doctor, obtaining, as a result the number of 15 patients per day. This value doesn't show whether this doctor works a lot or, otherwise, attends to very few patients. Therefore, it would be necessary to perform the query comparing the results with the attendance values of the other medical doctors working at the same center. With it the manager may get the conclusion that all the staff at the same center have a similar productivity; however, he/she still doesn't know if it is a good productivity or not: the value obtained doesn't have the same meaning if the health center attends to a small population than if it is at a big crowded city. Hence, to know if this number is appropriated, it would be necessary to perform a query comparing this value with the ones of the medical doctors working at other health centers with similar characteristics. In other words: 15 patients/day may be a good rate in a small center (where the productivity uses to be medium) but a bad rate in a big hospital (where the average productivity uses to be high or very high).

In this example the same user has performed three different queries over the same data but with distinct purposes, each requiring a different interpretation according to the granularity of the information with which this data is compared.

The research field closer to the problem of the meaning of the queries is, as mentioned above, the linguistic summary field. According to Bouchon-Meunier, B. & Moyse [1], proposals in this scope can be categorized in two groups. On the one hand can be found the proposals using fuzzy logic quantifiers [2-8], on the other hand, proposals based on nature language generation NLG [9-14].

Nevertheless, all of these techniques doesn't take into account the granularity of the information and just tell the user "how many of the X verify Y", when what the user really wants is to analyze the same data item from different points of view (alone, compared with a small set or with a bigger set) each with a different meaning.
This is why in this paper we introduce the concept of *Semantic Interpretation* of the results of queries on a datacube. To this purpose in section 2 we present the multidimensional model used as reference, whereas in section 3 we describe the notion of *semantic interpretation*.

MULTIDIMENSIONAL MODEL

The base for the semantic interpretation is a multidimensional model to store the data and query it. In this section we briefly present the model. A detailed definition can be found in [16,17].

1. Multidimensional Structure

In this section we present the structure of the fuzzy multidimensional model.

Definition 1. A dimension is a tuple $d=(l,\leq_d,l_\perp,l_\top)$ where $l=\{l_i, i=1,...,n\}$ so that each l_i is a set of values and $l_i \cap l_j=\emptyset$ if $i\neq j$, and $\leq_d$ is a partial order relation between the elements of l. $l_\perp$ and $l_\top$ are two elements of l so that $\forall l_i \in l \; l_\perp \leq_d l_i$ and $l_i \leq_d l_\top$.
We denote *level* to each element l_i. To identify the level l of the dimension d we will use $d.l$. The two special levels $l_\perp$ and $l_\top$ will be called *base level* and *top level* respectively. The partial order relation in a dimension is what gives the hierarchy relation between the levels.

Definition 2. For each dimension d the domain is $dom(d)=\bigcup l_i$.

Definition 3. For each l_i the set

$$H_{l_i} = \{l_j / l_j \neq l_i \wedge l_j \leq_d l_i \wedge \neg\exists l_k \quad l_j \leq_d l_k \leq_d l_i\} \tag{1}$$

We call it *set of children of the level l*.
Definition 4. For each l_i the set

$$P_{l_i} = \{l_j / l_i \neq l_j \wedge l_i \leq_d l_j \wedge \neg\exists l_k \quad l_i \leq_d l_k \leq_d l_j\} \tag{2}$$

And we call it *set of parents of the level l*.
Definition 5. For each pair of levels l_i and l_j so that $l_j \in H_{l_i}$ we have the relation

$$\mu_{ij} : l_i \times l_j \rightarrow [0,1] \tag{3}$$

We call it *kinship relation*.

The degree of inclusion of the elements of a level in the elements of their parent levels can be defined using this relation. If we use only the values 0 and 1 and we only allow an element to be included with degree 1 by a unique element of its parent levels, this relation represents a crisp hierarchy. If we relax these conditions and we allow to use values in the interval [0,1] without any other limitation, we have a fuzzy hierarchy relation. This allows the representation of several hierarchy relations in a more intuitive way. Furthermore, this fuzzy relation allows the definition of hierarchies in which there is imprecision in the relationship between elements in different levels. In this situation, the value in the interval shows the degree of confidence in the relation.

Definition 6. For each pair of levels l_i and l_j of the dimension d so that $l_j \leq_d l_i \wedge l_j \neq l_i$ we have the relation $\eta_{ij} : l_i \times l_j \rightarrow [0,1]$ defined as

$$\eta_{ij}(a,b) = \begin{cases} \mu_{ij}(a,b) \quad if \quad l_j \in H_{l_{ii}} \\ \bigoplus_{l_k \in H_{l_i}} \bigoplus_{c \in l_k} (\mu_{ik}(a,c) \otimes \eta_{kj}(c,b)) \quad in \quad other \quad case \end{cases} \tag{5}$$

Where $\oplus$ and $\otimes$ are a t-norm and a t-conorm respectively or operators from the families MOM and MAM defined by Yager ([15]), that include the t-norms and t-conorms. We call this *extended kinship relation*.
This relation gives us information about the degree of relation between two values in different levels inside the same dimension. To obtain this value, it considers all the possible paths between the elements in the hierarchy. Each one is calculated aggregating the kinship relation between elements in two consecutive levels using a t-norm. Then the final value is the aggregation of the result of each path using a t-conorm.

Definition 7. We call *fact* to any pair (h,α) where h is a m-tuple over the domain of the attributes we want to analyze, and $\alpha \in [0,1]$.

The management of uncertainty in the facts is carried out using a degree of certainty with each one. This degree of certainty allows us to use values in analysis that can be interesting to the decider but imply imprecision.

Definition 8. An object of type *history* is the recursive structure

$$H = \begin{cases} \text{Ø} \\ (\text{A}, \text{l}_\text{b}, F, G, H') \end{cases} \quad (7)$$

Where
- Ø is a special symbol.
- F is a set of facts.
- l_b is a set of levels $(l_{1b},\ldots,l_{nb})$.
- A is an application from l_b to F
- G is an aggregation operator.
- H' is a structure of type *history*.

Definition 9. A datacube is a tuple $C=(D,l_b,F,H,A)$ so that $D=(d_1,\ldots,d_n)$ is a set of dimensions, $l_b=(l_{1b},\ldots,l_{nb})$ is a set of levels so that l_{ib} belongs to d_i, $F = RU\text{Ø}$ where R is the set of facts and Ø is a special symbol, H is an object of type *history*, and A is an application defined as $A: l_{1b} \times \ldots \times l_{nb} \rightarrow F$.

If for a $\vec{a} = (a_1,\ldots,a_n)$ we have $A(\vec{a}) = \text{Ø}$, this means that there isn't defined a fact for this combination of values.

Definition 10. We say a datacube is *basic* if $l_b = (l_{1\perp},\ldots,l_{n\perp})$ and H= Ø.

2. Operations

Once we have the structure of the multidimensional model, we need the operations to analyse the data in the datacube. In this section we present the normal operations (roll-up, drill-down, slice, dice and pivot) over the structure proposed. In section 3 we present an example of the application of these operations over a multidimensional schema.

Definition 11. An aggregation operator G is a function $G(B)$ where $B = \{(h,\alpha)/(h,\alpha) \in F\}$ and the result is a tuple (h', α').

The parameter that operator needs can be seen as a fuzzy bag ([6]). In this structure there is a group of elements that can be duplicated, and each one has a degree of membership.

Definition 12. For each value *a* belonging to d_i we have the set

$$F_a = \begin{cases} \bigcup_{l_j \in H_{l_i}} F_b / b \in l_j \wedge \mu_{ij}(a,b) > 0 \quad if \quad l_i \neq l_b \\ \{h / h \in H \wedge \exists a_1,...,a_n A(a_1,..,a,...,a_n) = h\} \quad if \quad l_i = l_b \end{cases} \tag{11}$$

The set F_a represents all the facts that are related to the value *a*.
With this structure, the basic operations over datacubes are defined: *roll-up, drill-down, dice, slice* and *pivot* (see [16] for definition and properties).

SEMANTIC INTERPRETATION

In this section we present the inclusion of semantic interpretations in the fuzzy multidimensional model and the query process using them. Next section presents the structure of the semantic interpretations. Section 3.2 studies the aggregation functions related to the semantic of the results. The last section presents the process of the query.

1. Structure

A *Semantic Interpretation* (SI) is a structure associated to each fact. Its elements are:

- $L=\{L_1,...,L_m\}$: a set of linguistic labels over the basic domain. The set doesn't have to be a partition but this characteristic is desirable.
- $f_a(L,c)$: a function to adapt the labels in *L* to a cardinality *c*. As *c* can be a fuzzy set the function has to be able to work with this kind of data. The function f_a has to be continuous and monotone.
- $G=\{G_1,...,G_n\}$: a set indicating the aggregation functions that keep the semantic interpretation.

Multiple SI can be associated to each measure. On each fact we have to store as a metadata the cardinality associated to the value. This cardinality means the number of values that were aggregated to obtain this value.
When a value is going to be shown, the system applies the semantic interpretation to translate the value into a label. In this process we can differentiate two different approaches:

1. *Independent interpretation.* In this situation each value is studied without considering the context (the rest of the values) so we obtain an independent interpretation of the value. In this case, the cardinality to adapt the labels is the one stored in the value.
2. *Relative interpretation.* In this case, the values are compared with the other facts in the query so the interpretation is relative to the complete query. Hence the cardinality to adapt the labels depends on the complete set of values. In this case, the system calculates the average cardinality of all the values and uses this cardinality to adapt the labels.

2. Aggregation functions

Aggregation functions have an important role in the query process and in the semantic of the results. In this section we will study the different aggregation function types we can find according to the cardinality of the results and if a change of semantic occurs.

Let be a set of value $V=\{v_1,...,v_m\}$, of which set of cardinality is $C=\{n_1,...,n_m\}$, considering these two factors, we can classify the functions in three categories:

1. *Aggregators.* These functions aggregate the values and the cardinality is the sum of the cardinalities of each value:
$$c = \sum_{i=1}^{m} n_i$$
The only aggregation function that satisfies this behaviour is the SUM.
2. *Summaries.* In that case, the functions take a set of values and obtain a value that summarizes the complete set. Then the cardinality has to represent the average cardinality of the values.
$$c = \frac{\sum_{i=1}^{m} n_i}{m}$$
Most of the statistic indicators are in this category (maximum, minimum, average, median, percentiles, etc.).
3. *Others.* These functions represent a complete change of the semantic of the values so the result has to be considered in a new domain. In that case, the cardinality should be established to 1.
$$c=1$$
In this category we find functions like the variance or the count.

Once we have studied the aggregation functions we have all the elements to show the query process with the semantic interpretations.

3. Query process

In this section we present the query process considering the use of the SI. We can differentiate two phases on the query process: the OLAP query over the datacube and the report with the results. In both phases the SI are involved in a different way. Let show the process on each one:

1. *Query over the datacube*. In this phase is where the values/cardinalities are calculated. Inside this process we have to calculate the metatada of each one so, in the next phase, the values can be shown using the SI. The cardinality is calculated over each value considering the aggregation function used as shown in section 3.3.
In this process the system has to control whether the semantic has changed or not. On each value the system checks if the aggregation function used is in the set *G* of each *SI*. If the function is not included, then this *SI* is deleted. In next phase (the report) the user can only use the *SIs* that satisfy this restriction.
2. *Report*. Once the query has finished the result is shown to the user in a report. In this process the user has to choose the way to represent the values (the *SI* to use) and the interpretation (independent or relative as shown in section 3.1). After these steps, the

system adapts the labels of each value using the f_a functions and the right cardinality (the absolute or the average).

LEARNING THE f_a FUNCTIONS

In the previous section we have presented the query process using SI. One of the phases adapts the labels in L so they are fitted to the new cardinality. This process is carried out using the f_a function. The quality of the result will depend on this function, so an important point is the process to define it.
Asking the user for that function is not always possible because most of the times the user is not able to use a mathematical expression to define his/her interpretations. This is why we propose to learn the functions, following the next steps:

1. First we ask the user for an interpretation over the basic domain so we can define the set L of labels over it.
2. To learn the function now the system runs some queries over the datacube showing the results.
3. For each query the system asks the user to associate a label of L to the value. The system then stores the associations and the cardinalities of each value.
4. With these associations the system tries to fit a function that satisfies the interpretation with the corresponding cardinality. In this process, the system will try continuous and monotone functions to adapt the labels. If the fitted function has good quality (the adapted labels correspond to the labels associated by the user) the process ends. In other case, the process goes on but to point 2 to show more queries so the system gets more data to fit the function.

EXAMPLE

In this section we will present a small example to show in detail the proposed method. Let us suppose we have a simple datacube with only two dimensions (time and centre) and only one measure (number of patients). The hierarchies for both dimensions are shown in Figure 1.

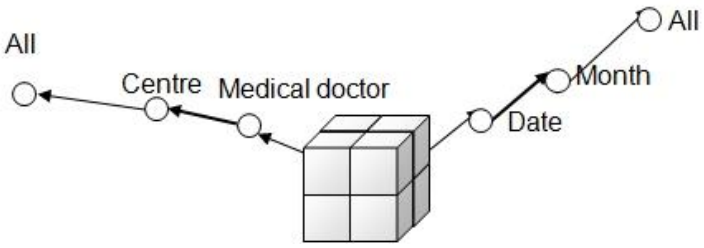

Figure 1: Example datacube

For the measure we define a *SI* indicating if the number of patients attended by a doctor is *Low*, *Normal* or *High*. At base level (doctor and day) the fuzzy partition is shown in Figure 2.

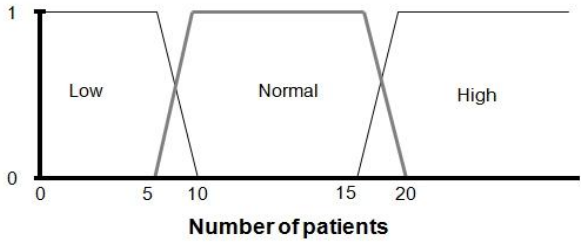

Figure 2: Fuzzy partition over the measure Number of Patients

The *SI* is valid for aggregations like sum and average. The last aspect to define is the f_a function. In this example we suppose that it is lineal and just multiply the points of the fuzzy label by the new cardinality (e.g. if Low is defined as (0,0,5,10) for one doctor in a day, for two doctors the label will be (2x0,2x0,2x5,2x10)=(0,0,10,20)).

Let us suppose that we have two centres with different size. One (*C1*) is placed in a city and there are 500 medical doctors in the staff. The second one (*C2*) is placed in a small village and only 10 doctors are working in that centre. If a manager asked the system to calculate the number of patients attended by both centres each month we can get a table like that

Table 1: Query result

Centre	Month	Patients
C1	January	125.000
C1	February	130.000
...	...	...
C2	January	4.200
C2	February	5.000
...	...	...

The table shows very different results for each centre and it is not easily interpretable due to the differences in the size of each one. Applying our proposal we obtain the label that best represents each value.

For centre C1 we have to calculate the cardinality of the result so we can adjust the labels. We have the data defined to doctor and day so, for each month we have to aggregate the values for 500 medical doctors and 20 working days for month. Therefore the cardinality is 500x20=10.000. We adjust the fuzzy partition for this new cardinality as shown in Figure 3. In the case of centre *C2* then the cardinality is 10x20=200 (Figure 4).

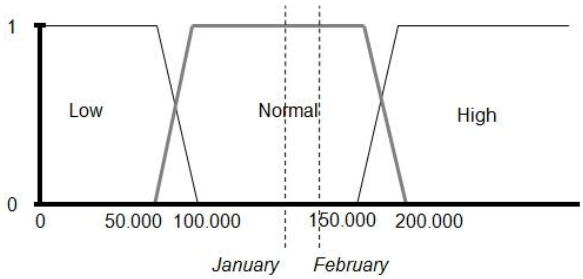

Figure 3: Labels adaptation for query for centre C1

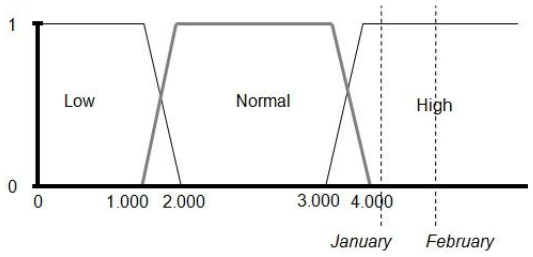

Figure 4: Labels adaptation for query for centre C2

In the figures 3 and 4 we have indicated the values for the Table 1, so we have the labels associated to each result. In Table 2 we have added the label associated to each value.

Table 2: Query results using Semantic Interpretation

Centre	Month	Patients	*Label*
C1	January	125.000	*Normal*
C1	February	130.000	*Normal*
...	...	...	...
C2	January	4.200	*High*
C2	February	5.000	*High*
...	...	...	...

As can be seen using the *Semantic Interpretations* in the example it is appreciated how the values are adapted so the user has the interpretation (the meaning) of the values directly.

CONCLUSIONS

In this paper we have introduced the new concept of *Semantic Interpretation*, that provides the OLAP systems with the new capability of querying about the same given item with different purposes obtaining in each case a result with a different meaning. With our proposal, the semantic of the results of the query can be distinct and adapted to the needs of the user, by taking into account the granularity of the information considered.

We also have formalized the concept of *SI* and provide the formulation to apply over a multidimensional model using the normal OLAP operations.

REFERENCES

1. Bouchon-Meunier, B. & Moyse, G. Fuzzy linguistic summaries: Where are we, where can we go? Computational Intelligence for Financial Engineering Economics (CIFEr), 2012 IEEE Conference on, 2012, 1-8
2. R. R. Yager, "A new approach to the summarization of data," Information Sciences, vol. 28, no. 1, pp. 69-86, Oct. 1982.
3. L. Zadeh, "A computational approach to fuzzy quantifiers in natural languages," Computers & Mathematics with Applications, vol. 9, no. 1, pp. 149-184, 1983.
4. R. R. Yager, "Fuzzy summaries in database mining," in Proceedings the 11th Conference on Artificial Intelligence for Applications, 1995, pp. 265-269.

5. J. Kacprzyk, A. Wilbik, and S. Zadrozny, "Linguistic summarization of time series using a fuzzy quantifier driven aggregation," Fuzzy Sets and Systems, vol. 159, no. 12, pp. 1485-1499, Jun. 2008.
6. L. Liétard, "A new definition for linguistic summaries of data," in 2008 IEEE International Conference on Fuzzy Systems (IEEE World Congress on Computational Intelligence), 2008, pp. 506-511.
7. P. Bosc, L. Liétard, and O. Pivert, "Extended functional dependencies as a basis for linguistic summaries," Lecture notes in computer science, pp. 255-263, 1998.
8. D. Rasmussen and R. R. Yager, "Finding fuzzy and gradual functional dependencies with SummarySQL," Fuzzy Sets and Systems, vol. 106, no. 2, pp. 131-142, Sep. 1999.
9. Yseop, "Faire parler les chiffres automatiquement," 2011. [Online]. Available: http://www.yseop.com/demo/diagFinance/FR/.
10. S. Sripada, E. Reiter, and I. Davy, "SUMTIME-MOUSAM: Configurable Marine Weather Forecast Generator." 2003.
11. J. Yu, E. Reiter, J. Hunter, and S. Sripada, "SumTime-Turbine: A Knowledge-Based System to Communicate Gas Turbine Time-Series Data," in The 16th International Conference on Industrial & Engineering Applications of Artificial Intelligence and Expert Systems, 2003, p. 23-26.
12. L. Danlos, F. Meunier, and V. Combet, "EasyText: an Operational NLG System," in ENLG 2011, 13th European Workshop on Natural Language Generation, 2011.
13. E. Goldberg, N. Driedger, and R. . I. Kittredge, "Using natural-language processing to produce weather forecasts," IEEE Expert, vol. 9, no. 2, pp. 45-53, Apr. 1994.
14. F. Portet, E. Reiter, J. Hunter, and S. Sripada, "Automatic generation of textual summaries from neonatal intensive care data," in 11th Conference on Artificial Intelligence in Medicine (AIME '07), 2007, p. 227-236.
15. Yager, R.R.: Aggregation Operators and Fuzzy Systems Modeling. Fuzzy Sets and Systems 67 (1994) 129-145
16. Molina C, Rodriguez-Ariza L, Sanchez D, Vila A. A New Fuzzy Multidimensional Model. Fuzzy Systems, IEEE Transactions on. 2006;14:897 -912
17. Delgado M, Molina C, Rodríguez Ariza L, Sanchez D, Vila A. F-Cube Factory: a fuzzy olap system for supporting imprecision. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems [Internet]. 2007;15:59-81