The use of intuitionistic fuzzy cube and operators in treating imprecision in data repositories

Ermir Rogova

Dr.sc. Ermir ROGOVA

Abstract

Traditional data repositories introduced for the needs of business processing, typically focus on the storage and querying of crisp domains of data. As a result, current commercial data repositories have no facilities for either storing or querying imprecise/ approximate data.

No significant attempt has been made for a generic and application-independent representation of value imprecision mainly as a property of axes of analysis and also as part of dynamic environment, where potential users may wish to define their "own" axes of analysis for querying either precise or imprecise facts. In such cases, measured values and facts are characterised by descriptive values drawn from a number of dimensions, whereas values of a dimension are organised as hierarchical levels.

In this paper, an extended multidimensional model named IF-Cube is put forward, which allows the representation of imprecision in facts and dimensions and answering of queries based on imprecise hierarchical preferences.

Since the emergence of the OLAP technology, [1] different proposals have been made to give support to different types of data and application purposes. One of these is to extend the relational model (ROLAP) to support the structures and operations typical of OLAP. Further approaches [2, 3] were based on extended relational systems to represent data-cubes and operate over them. Another approach would be to develop new models using a multidimensional-cubic view of the data [4].

Nowadays, information and knowledge-based systems need to manage imprecision in the data, and more flexible structures are needed to represent the analysis domain. Models have been proposed for managing imprecision, as part of an incomplete data-cube [5], in the facts and the definition of facts using different

Iliria International Review – 2011/1

levels in the dimensions [6].

Nevertheless, these models continue to use inflexible hierarchies, thus making it difficult to merge reconcilable data from different sources with some incompatibilities in their schemata. These incompatibilities arise due to different perceptions / views about a particular modelling reality.

In addressing the problem of representing flexible hierarchies, here is proposed a new multidimensional model that is able to deal with imprecision over conceptual hierarchies utilising the concept of H-IFS.

The use of conceptual hierarchies or H-IFS enables one to:

- define the structures of a dimension in a more perceptive way to the final user, thus allowing a more perceptive use of the system.
- query information from different sources or even utilize domain preferences and enhance the description of hierarchies, thereby getting more knowledgeable query results. H-IFS is a unique way for incorporating "kind-of" relations, or conceptual hierarchies as part of a Knowledge based OLAP analysis (KNOLAP).

In the following sections, OLAP foundations are reviewed and a model aimed at resolving imprecision at the "Cube" or data level is proposed. The semantics of the Intuitionistic fuzzy cubic representation are introduced in contrast to the basic multidimensional-cubic structures. Overall, the introduced Intuitionistic Fuzzy cubic representation [7,8] allows users to deal with imprecision not only at the level of dimensions with the aid of H-IFS but also at the level of facts or data. The basic cubic operators are extended and enhanced with the aid of Intuitionistic Fuzzy Logic [9,10].

1.1 Semantics of the IF-Cube vs. Crisp Cube

In this section the semantics of Multidimensional modelling and Intuitionistic Fuzzy Logic are reviewed, and based on these a unique concept named Intuitionistic Fuzzy Cube (IF-Cube) is proposed. The IF-Cube, in conjunction with the utilisation of H-IFS, allows users to model the following cases:

- Well defined hierarchies / dimensions and imprecise data
- H-IFS based hierarchies / dimensions and imprecise data

Overview of the Cube Model

A logical model that influences both the database design and the query engines is the *multidimensional-cubic* view of data in a warehouse. In a multidimensional data model, there is a set of *numeric measures* that are the objects of analysis. Examples of such measures are total sales, available budget, etc. Each of the numeric measures depends on a set of *dimensions,* which provide the context for the measure. The attributes of a dimension may be related via a hierarchy of relationships. In the above example, the product name is related to its category and the industry attribute through a hierarchical relationship, (*see "Figure 1"*).

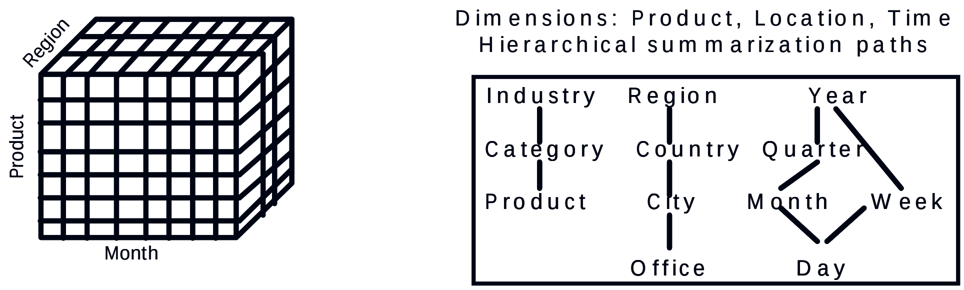


Figure 1: Cube 'Sales' - Rigid Hierarchies

A cubic structure [4] is defined as a 4-tuple <D, M, A, F> where the four components indicate the characteristics of the cube. These characteristics are:

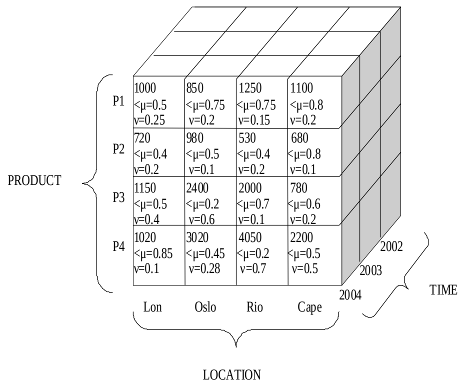
- a set of n dimensions $D = \{d_1, d_2, \ldots, d_n\}$ where each d_i is a dimension name, extracted from a domain $dom_{dim(i)}$.
- a set of k measures $M = \{m_1, m_2, \ldots, m_k\}$ where each m_i is a measure name, extracted from a domain $dom_{measure(i)}$.
- The set of dimension names and measures names are disjoint; i.e. $D \cap M = 0$.
- A set of t attributes $A = \{a_1, a_2, \ldots, a_t\}$ where each a_i is an attribute name, extracted from a domain $dom_{attr(i)}$.
- A one-to-many mapping $F: D \rightarrow A$, i.e. there exists, corresponding to each dimension, a set of attributes.

1.1.1 Semantics of the IF-Cube

In contrast, an **IF-Cube** is an abstract structure that serves as the foundation for the multidimensional data cube model. Cube C is defined as a five-tuple *(D, l, F, O, H)* where:

- *D* is a set of dimensions
- *l* is a set of levels $l_1, \ldots, l_n,$
- A dimension $d_i = (l \leq O, l_\perp, l_\top)\ dom(d_i)$ where $l = l_i\ i = 1 \ldots n.$
- l_i is a set of values and $l_i \cap l_j = \{\}$,
- $\leq O$ is a partial order between the elements of *l*.
- To identify the level *l* of a dimension, *dl* is used as part of a hierarchy.
  $l_\perp$: base level $l_\top$: top level
  for each pair of levels l_i and l_j there exist the relation:
  $$\mu_{ij} : l_i \times l_j \rightarrow [0,1] \quad v_{ij} : l_i \times l_j \rightarrow [0,1] \quad 0 < \mu_{ij} + v_{ij} < 1$$
- *F* is a set of fact instances with schema: $F = \{<x, \mu_F(x), v_F(x)> \mid x \in X\}$, where $x = <att_1, \ldots, att_n>$ is an ordered tuple belonging to a given universe *X*, $\mu_F(x)$ *and* $v_F(x)$ are the degree of membership and non-membership of *x* in the fact table *F* respectively.
- *H* is an object type history that corresponds to a cubic structure *(l, F, O, H′)* which allows the tracing back the evolution of a cubic structure after performing a set of operators i.e. aggregation.

The example below provides a sample imprecise cube *(D, l, F, O, H) i.e. sales* and a conceptual flexible hierarchy product with reference to wine consisting of $l_i,\ldots,l_n$ levels with respective levels of membership and non membership $< \mu_{ij}\ v_{ij}, >$.



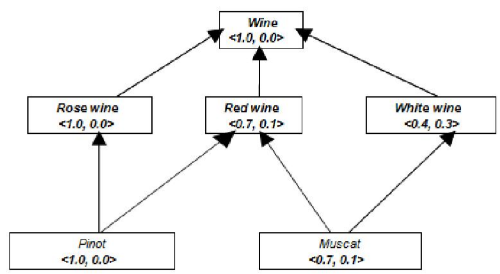


Figure 2: Imprecise cube 'Sales'   Figure 3: H- IFS Hierarchy 'Wine'

The defined IF OLAP Cube and the proposed OLAP operators make it possible to do the following:

- accommodate imprecise facts.
- utilize *conceptual hierarchies defined as H-IFS* used for aggregation purposes in the cases of roll-up and roll-down operations.
- offer a unique feature such as keeping track of the history when there is movement between different levels of a hierarchical order.

In the next section, first the current cubic operators are reviewed and then the IF-Operators are explained. These operators have been extended and redefined in order to cope with or multidimensional model.

1.2 IF-cubic operators vs. normal cubic operators

In the previous section, it was shown how the proposed IF-cube differs from the original cube and that it can be made to accommodate imprecision, both on the data level and on the conceptual level. However, the ability to store the data is only a small part of the problem. The difficulty stands with the ability to process such data, as the original cubical operators have not been designed to process imprecise information. In the subsections below, first the original operators will be shown and then the new IF-Operators presented, which will be able to deal with the new multidimensional structure.

1.2.1 Overview of the cubic operators

The cubic model proposed in [4], which is considered by many OLAP experts to be the fundamental one when it comes to the cubic model, also describes the algebraic operators necessary for the functioning of the multidimensional cube that have been adopted widely. Below is shown a brief description of these operators, the full descriptions of which can be found on [4].

Restriction (σ): This operator restricts the values on one or more dimensions. It has an *atomic predicate,* denoted by *p*, that is a logical expression involving a single dimension or a *compound predicate*, denoted by *P* the is an expression involving a set of atomic predicates.

Mathematical notation: $\sigma_p(C_i)=C_o$

Example: $\sigma_{(year=2009)}$ *(Sales)*

Aggregation (α): This operator performs aggregation on one or more dimensions. This operator is based on relational aggregate functions (e.g. SUM AVG MAX) and allows these functions to be applied to cubes with one or more dimensions specified as *grouping attributes.*

Mathematical notation: $\alpha_{h,m,S}(C_I)=C_O$

Example: $\alpha_{[SUM(amount),\{product\_name,\ year\}]}$*(Sales)*

Cartesian product (×): This is a binary operator that can be used to relate two cubes.

Mathematical notation: $C_{I1} \times C_{I2} = C_O$

Join (|×|): The join operator is a special case of the *Cartesian product* operator that is used to relate two cubes having one or more dimensions in common and having identical mapping from the common dimensions to the respective attribute sets of these dimensions.

Mathematical notation:: $C_1\ |\times|\ C_2 = \sigma p(C_1 \times C_2)$ where *p is the predicate* and $C_1$and C_2 are the two cubes.

Union (∪): This operator finds the union of two input cubes. If, for example, two cubes *Sales_Engand* and *Sales_Wales* contain the sales figures corresponding to the respective regions, and the user would like to consolidate the data for both regions into a single cube. This would be achieved by using the union operator.

Mathematical notation: $C_{I1} \cup C_{I2} = C_O$

Difference (-): This operator finds the difference of two cubes. If, for example, two cubes *Sales_England* and *Sales_London* contain sales figures corresponding to the England and London, and the user would wish to remove London figures from the England cube. This would be achieved using the difference operator.

Mathematical notation: $C_{I1} - C_{I2} = C_O$

1.2.2 The IF-cubic operators

In this section the IF-Cubic operators are defined and explained. Each operator is presented in the following format: the operator's name, symbol, textual description, input, output, mathematical description and an example of the operator.

Basic operators

Selection (Σ): The selection operator selects a set of fact-instances from a cubic structure that satisfy a predicate (θ). A predicate (θ) involves a set of atomic predicates ($\theta_1, \ldots, \theta_n$) associated with the aid of logical operators p (i.e. $\wedge$, $\vee$, etc.). Only the cells that satisfy the predicate p are captured into the result cube. If θ' is an Intuitionistic fuzzy predicate, then the set of possible facts that satisfy the θ should carry a degree of membership μ and non-membership v expressed as follows:

$$F = \{<x, \min(\mu_F(x), \mu(\theta(x))), \max(v_F(x), v(\theta(x)))> \mid x \in X\}$$

Thus the resulting cube populated with fact instances that either satisfy the predicate (θ) completely or to some degree of certainty. Where $\pi = 1 - (\mu + v)$ and acts as an index of the uncertainty, i.e. the higher the value of π, the more uncertain the fact instance is, even though it may entail the same level of membership μ.

Input: $C_i = (D, l, F, O, H)$ and the predicate θ.

Output: $C_o = (D, l, F_o, O, H)$, where

$F_o \subseteq F$ and $F_o = \{f \mid (f \in F) \wedge (f \text{ satisfies } \theta)\}$.

Mathematical notation: $\sum_\theta (C_i) = C_o$.

Example: Find the sales amount of 1000 with membership of greater than 0.4 and non-membership of less than 0.3 for all products in all cities during 2004:

$$\Sigma_{(\text{amount}=1000 \wedge (\mu>0.4 \wedge v<0.3) \wedge \text{year}=2004)}(\text{Sales}) = C_{\text{Result}}$$

Cubic Projection (Π): In cubic instances that hold non-deterministic facts, there can be no projecting-out of any of individual domains. The reason behind this statement is that unlike deterministic cubes, in non-deterministic ones the membership and non-membership of a fact instance determines the likelihood of all domains involved in that cube/fact instance. Hence, projecting out a domain, would result in loss of information.

Input: $C_i = (D, l, F, O, H)$.

Output: $C_o = (D, l, F, O, H)$.

Mathematical notation: $\Pi_F(C_i) = C_o$.

Example: Project the cube from the previous example:

$$\Pi_{(\text{Sales})}(\Sigma_{(\text{amount}=1000 \wedge (\mu=0.4 \wedge v=0.3) \wedge \text{year}=2004)}(\text{Sales})) = C_{\text{Result}}$$

Basic Cubic Product (⊗): This is a binary operator $C_{i1} \otimes C_{i2}$. It is used to relate two cubes C_{i1} and C_{i2} assuming that $D_1 \subseteq D_2$ and O_1, O_2 are reconcilable partial orders. Thus, l_1, l_2 could lead to 1_o being a ragged hierarchy.

Input: $C_{i1} = (D_1, l_1, F_1, O_1, H_1)$ and $C_{i2} = (D_2, l_2, F_2, O_2, H_2)$.

Output: $C_o = (D_o, l_o, F_o, O_o, H_o)$, where

$D_o = D_1 \cup D_2$, $l_o = l_1 \cup l_2$, $O_o = O_1 \cup O_2$, $H_o = H_1 \cup H_2$,

$F_o = F_1 \times F_2 = \{<<x, y>, \min(\mu_{f1}(x), \mu_{f2}(y)), \max(\nu_{f1}(x), \nu_{f2}(y))> \mid <x, y> \in X \times Y\}$.

Mathematical notation: $C_{i1} \otimes C_{i2} = C_o$.

Example: Consider the two cubes one wants to relate,

C_{i1}: C_{Sales} and C_{i2}: $C_{\text{Discounts}}$.

$C_{\text{Discounts}}$ has the same dimensions as C_{Sales} except the measure amount is not sale but is a discount. In that case the cubic product of these two, would be:

$C_{\text{Sales}} \otimes C_{\text{Discounts}} = C_{\text{Result}}$

ProdID	StoreID	Amount	<μ, ν>
P1	S1	10	0.7, 0.2
P2	S2	15	0.5, 0.5

⊗

ProdID	StoreID	Discount	<μ, ν>
P2	S1	2	0.5, 0.5
P3	S3	5	0.3, 0.3

⇓

S.Prod ID	S.Store ID	S.Amou nt	D.Prod ID	D.Store ID	Discoun t	<μ, ν>
P1	S1	10	P2	S1	2	0.5, 0.5
P1	S1	10	P3	S3	5	0.3, 0.3
P2	S2	15	P2	S1	2	0.5, 0.5
P2	S2	15	P3	S3	5	0.3, 0.5

Table 1: Cubic product

Union (∪): The union operator is a binary operator that finds the union of two cubes. C_{i1} and C_{i2} have to be union compatible. The operator also coalesces the value-equivalent facts using the minimum membership and maximum non-membership.

Input: $C_{i1} = (D_1, 1_1, F_1, O_1, H_1)$ and $C_{i2} = (D_2, l_2, F_2, O_2, H_2)$.

Output: $C_o = (D_o, l_o, F_o, O_o, H_o)$, where

$D_o = D_1 = D_2$, $l_o = l_1 = l_2$,

$O_o = O_1 = O_2$, $H_o = H_1 = H_2$,

$F_o = F_1 \cup F_2 =$

$= \{<x, \max(\mu F_1(x), \mu F_2(x)), \min(\nu F_1(x), \nu F_2(x))> \mid x \in X\}$.

Mathematical notation: $C_{i1} \cup C_{i2} = C_o$.

Example: Consider the two cubes one want to relate,

C_{i1}: $C_{\text{Sales\_North}}$ and C_{i2}: $C_{\text{Sales\_South}}$,

in that case the union of these two cubes would be:

$C_{\text{Sales\_North}} \cup C_{\text{Sales\_South}} = C_{\text{Result}}$

ProdID	StoreID	Amount	<μ, ν>
P1	S1	10	0.7, 0.2
P2	S2	15	0.5, 0.5

∪

ProdID	StoreID	Amount	<μ, ν>
P1	S1	10	0.5, 0.5
P3	S3	5	0.3, 0.3

⇓

S.ProdID	S.StoreID	S.Amount	<μ, ν>
P1	S1	10	0.7, 0.2
P2	S2	15	0.5, 0.5
P1	S1	10	0.5, 0.5
P3	S3	5	0.3, 0.3

⇓

S.ProdID	S.StoreID	S.Amount	<μ, ν>
P1	S1	10	0.7, 0.2
P2	S2	15	0.5, 0.5
P3	S3	5	0.3, 0.3

Table 2: Union operator example

Difference (-): The difference operator is a binary operator that the difference of two cubes. It is similar to the difference operator in relational algebra. C_{i1} and C_{i2} have to be union compatible. The difference operator removes the portion of the cube C_{i1} that is common to both cubes.

Input: $C_{i1} = (D_1, l_1, F_1, O_1, H_1)$ and $C_{i2} = (D_2, l_2, F_2, O_2, H_2)$.

Output: $C_o = (D_o, l_o, F_o, O_o, H_o)$, where

$D_o = D_1 = D_2$, $l_o = l_1 = l_2$, $O_o = O_1 = O_2$,

$H_o = H_1 = H_2$,

$F_o = F_1 \cap F_2 = \{<x, \min(\mu F_1(x), \mu F_2(x)), \max(\nu F_1(x), \nu F_2(x))> \mid x \in X\}$.

Mathematical notation: $C_{i1} - C_{i2} = C_o$.

Example: Consider the two cubes one wants to relate,

C_{i1}: $C_{\text{Sales\_North}}$ and C_{i2}: $C_{\text{Sales\_South}}$,

in that case the difference between North and South sale cubes would be:

$$C_{\text{Sales\_North}} - C_{\text{Sales\_South}} = C_{\text{Result}}$$

ProdID	StoreID	Amount	<μ, ν>
P1	S1	10	0.7, 0.2
P2	S2	15	0.5, 0.5

–

ProdID	StoreID	Amount	<μ, ν>
P1	S1	10	0.5, 0.5
P3	S3	5	0.3, 0.3

⇓

S.ProdID	S.StoreID	S.Amount	<μ, ν>
P1	S1	10	0.7, 0.2
P2	S2	15	0.5, 0.5
P1	S1	10	0.5, 0.5

⇓

S.ProdID	S.StoreID	S.Amount	<μ, ν>
P1	S1	10	0.5, 0.5
P2	S2	15	0.5, 0.5

Table 3: Difference operator example

Extended Operators

Join (Θ): The join operator relates two cubes having one or more dimensions in common, and having identical mappings from common dimensions to the respective attribute sets of these dimensions. This operation can be expressed using Cubic Product operation.

$C_{i1} = (D_1, l_1, F_1, O_1, H_1)$ and $C_{i2} = (D_2, l_2, F_2, O_2, H_2)$ are candidates to join if $D_1 \cap D_2 \neq 0$.

Input: $C_{i1} = (D_1, 1_1, F_1, O_1, H_1)$ and $C_{i2} = (D_2, l_2, F_2, O_2, H_2)$.

Output: $C_o = (D_o, l_o, F_o, O_o, H_o)$.

Mathematical notation: $C_{i1}\ \Theta\ C_{i2} = \sigma_p(C_{i1} \otimes C_{i2})$.

Example: Consider the two cubes one wants to relate, C_{i1}: C_{Sales} and C_{i2}: $C_{\text{Discounts}}$. $C_{\text{Discounts}}$ has the same dimensions as C_{Sales} except the measure amount is not sale but is a discount.

Also there is a predicate $p = (S.ProdID = D.ProdID \wedge S.StoreID = D.StoreID)$. In that case the join of these two, would be:

$C_{\text{Sales}}\ \Theta\ C_{\text{Discounts}} = C_{\text{Result}}$.

ProdID	StoreID	Amount	<μ, ν>
P1	S1	10	0.7, 0.2
P2	S2	15	0.5, 0.5

Θ

ProdID	StoreID	Discount	<μ, ν>
P1	S1	2	0.5, 0.5
P3	S3	5	0.3, 0.3

$\Downarrow$

S.ProdID	S.StoreID	S.Amount	D.Discount	<μ, ν>
P1	S1	10	2	0.5, 0.5

Table 4: Join operator example

Aggregation (A): The aggregation operator performs aggregation on one or more dimensional attributes utilizing Intuitionistic Fuzzy functions such as IFS_{SUM}, IFS_{AVG}, IFS_{MIN}, IFS_{MAX}. An aggregation operator A is a function $A(G)$, where $G = \{<x, \mu_F(x), \nu_F(x)> \mid x \in X\}$, where $x=<att_1, \ldots, att_n>$ is an ordered tuple, belonging to a given universe X, $\{att_1, \ldots, att_n\}$ is the set of attributes of the elements of X, $\mu_F(x)$ and $\nu_F(x)$ are the degree of membership and non-membership of x.

The result is a bag of the type $\{<x', \mu_F(x'), \nu_F(x')> \mid x' \in X\}$.To this extent, the bag is a group of elements that can be duplicated and each one has a degree of μ and ν.

Input: $C_i = (D, l, F, O, H)$ and the function $A(G)$.

Output: $C_o = (D, l_o, F_o, O_o, H_o)$.

The definition of aggregation operator points to the need of defining the IFS extensions for traditional group operators such as SUM, AVG, MIN and MAX.

Group Operations & Operators

In this section an investigation is made on how traditional group operations can be redefined to cope with the IFS representation of data. Note that the introduction of the IF facts influence the evaluation of aggregates at different levels:

- Will the result over which the aggregate is performed be either crisp or Intuitionistic Fuzzy?
- What is the meaning of the result after the IF aggregation is performed?

Using the standard definitions for the group operators (SUM, AVG, MIN and MAX) as foundations, their IF extensions and meaning is provided.

***IFS_{SUM}* :** The IFS_{sum} aggregate, like its standard counterpart, is only defined for numeric domains. Given a fact F defined on the schema $X(att_1, \ldots, att_n)$, let att_{n-1} defined on the domain $U=\{u_1, \ldots, u_n\}$. The fact F consists of fact instances f_i

with $1 \le i \le m$. The fact instances f_i are assumed to take Intuitionistic fuzzy values for the attribute att_{n-1} for i = 1 to m

$$f_i\,[att_{n-1}] = \{<\mu_i(u_{ki}),\, v_i(u_{ki})>/\, u_{ki} \mid 1 \le k_i \le n\}.$$

The IFS_{sum} of the attribute att_{n-1} of the fact table F is defined by:

$$IFS_{SUM}((att_{n-1})(F)) =$$

$$= \left\{ < u > / \, y \mid ((u = \min_{i=1}^{m}(\mu_i(u_{ki}), \nu_i(u_{ki})) \wedge \left(y = \sum_{k_i = k_1}^{km} u_{ki} \right) (\forall k_1, \ldots, k_m : 1 \le k_1, \ldots, k_m \le n)) \right\}$$

***IFS**$_{AVG}$* **:** The IFS_{AVG} aggregate, like its standard counterpart, is only defined for numeric domains. This aggregate makes use of the IFS_{SUM} that was discussed previously and the standard *COUNT*. The IFS_{AVG} can be defined as:

$$IFS_{AVG}((att_{n-1})(F) = IFS_{SUM}((att_{n-1})(F)) \,/\, COUNT((att_{n-1})(F)).$$

***IFS**$_{MAX}$* **:** The IFS_{MAX} aggregate, like its standard counterpart, is only defined for numeric domains. Given a fact F defined on the schema $X(att_1, \ldots, att_n)$, let at_{n-1} defined on the domain $U=\{u_1, \ldots, u_n)$. The fact F consists of fact instances f_i *with* $1 \le i \le m$. The fact instances f_i are assumed to take intuitionistic fuzzy values for the attribute att_{n-1} for i = 1 to m

$$f_i[att_{n-1}] = \{<\mu_i(u_{ki}),\, v_i(u_{ki})>/\, u_{ki} \mid 1 \le k_i \le n\}.$$

The IFS_{sum} of the attribute att_{n-1} of the fact table F is defined by:

$$IFS_{MAX}((att_{n-1})(F)) +$$

$$= \left\{ < u > / \, y \mid ((u = \min_{i=1}^{m}(\mu_i(u_{ki}), \nu_i(u_{ki})) \wedge \left(y = \max_{i=1}^{m}(\mu_i(u_{ki}), \nu_i(u_{ki})) \right) (\forall k_1, \ldots, k_m : 1 \le k_1, \ldots, k_m \le n)) \right\}$$

IFS**$_{MIN}$:** The IFS_{MIN} aggregate, like its standard counterpart, is only defined for numeric domains. Given a fact F defined on the schema $X(att_1, \ldots, att_n)$, let att_{n-1} defined on the domain $U=\{u_1, \ldots, u_n)$. The fact F consists of fact instances f_i *with* $1 \le i \le m$. The fact instances f_i are assumed to take intuitionistic fuzzy values for the attribute att_{n-1} for i = 1 to m therefore $f_i[att_{n-1}] = \{<\mu_i(u_{ki}), v_i(u_{ki})>/ u_{ki} \mid 1 \le k_i \le n\}$. The IFS_{sum} of the attribute att_{n-1} of the fact table F is defined by:

$$IFS_{MIN}((att_{n-1})(F)) =$$

$$= \left\{ < u > / \, y \mid ((u = \min_{i=1}^{m}(\mu_i(u_{ki}), \nu_i(u_{ki})) \wedge \left(y = \min_{i=1}^{m}(\mu_i(u_{ki}), \nu_i(u_{ki})) \right) (\forall k_1, \ldots, k_m : 1 \le k_1, \ldots, k_m \le n)) \right\}$$

It can be observed that the IFS_{MIN} is extended in the same manner as IFS_{MAX} aggregate except for replacing the symbol **max** in the IFS_{MAX} definition with **min**.

The definition of the extended group operations makes it possible to define the extended group operators ***Roll up (Δ), and Roll Down (Ω).***

Roll up (Δ): The result of applying *Roll up* over dimension d_i at level dl_r *using the* aggregation operator A over a datacube $C_i = (D_i, l_i, F_i, O, H_i)$ *is another datacube* $C_o = (D_o, l_o, F_o, O, H_o)$.

Input: $C_i = (D_i, l_i, F_i, O, H_i)$.

Output: $C_o = (D_o, l_o, F_o, O, H_o)$.

An object of type history is a recursive structure:

$$H = \begin{cases} \omega \text{ is the initial state of the cube} \\ (l, D, A, H') \text{ is the state of the cube after performing an operation on the cube} \end{cases}$$

The structured history of the datacube allows the storing of all the information when applying *Roll up* and the recall of it back when *Roll Down* is performed. In order to be able to apply the operation of *Roll Up* the IFS_{SUM} aggregation operator needs to be put to use.

Roll Down (Ω): This operator performs the opposite function of the *Roll Up* operator. It is used to roll down from the higher levels of the hierarchy with a greater degree of generalization, to the leaves with the greater degree of precision. The result of applying *Roll Down* over a datacube $C_i = (D, l, F, O, H)$ having $H = (l', D', A', H')$ is another datacube $C_o = (D', l', F', O, H')$.

Input: $C_i = (D, l, F, O, H)$.

Output: $C_o = (D', l', F', O, H')$ where F' à *set of fact instances defined by operator A*.

To this extent, the *Roll Down* operative makes use of the recursive history structure previously created after performing the *Roll Up* operator.

1.3 Conclusions

In this paper the context of value imprecision was revised, as part of an MOLAP based environment. A new approach for extending the MOLAP model was presented, so that it can include treatment of value uncertainty as part of a multidimensional model, inhabited by concepts and flexible hierarchical structures of organization. A new multidimensional-cubic model named the IF-Cube was introduced, which is able to operate over data with imprecision either in the facts or in the dimensional hierarchies.

The main contribution of this new multidimensional-cubic model is that is able to operate over data with imprecision in the facts and the summarisation hierarchies. Classical models imposed a rigid structure that made the models present difficulties when merging information from different but still reconcilable sources.

These features are inexistent in current OLAP tools. Furthermore, it has been noticed that the IF-Cube can be used for the representation of Intuitionistic fuzzy linguistic terms.

References:

1. R. Kimball, The Data Warehouse Toolkit. New York, John Wiley & Sons, 1996.
2. S. Chaudhuri, U. Dayal, V. Ganti Database Technology for Decision Support Systems. In: Computer, Vol. 34, p. 48-55, 2001
3. M. Jarke, Fundamentals of data warehouses. Springer, London, 2002
4. H. Thomas & A. Datta, A Conceptual Model and Algebra for On-Line Analytical Processing in Decision Support Databases. Information Systems Research 12: pp.83-102, 2001
5. C. Dyreson, Information retrieval from an incomplete data cube, VLDB, Morgan Kaufman Publishers, pp. 532-543, 1996.
6. T. Pedersen, C. Jensen, and C. Dyreson, A foundation for capturing and querying complex multidimensional data, Information Systems, vol. 26, pp. 383-483, 2001.
7. E. Rogova, P. Chountas, On Imprecision Intuitionistic Fuzzy Sets & OLAP – The Case for KNOLAP, IFSA'07, Springer-Verlag GmBh , Theoretical advances and application of fuzzy logic and soft computing, pp. 11-20
8. E. Rogova, P. Chountas, B. Kolev, Intuitionistic Fyzzy Knowledge-based OLAP, Notes on Intuitionistic Fuzzy sets, Vol. 13, No.2, ISSN 1310-4926, 2007, pp.88-100
9. K. Atanassov (1999). Intuitionistic Fuzzy Sets, Springer-Verlag, Heidelberg
10. K. Atanassov Remarks on the Intuitionistic fuzzy sets. – Fuzzy Sets and Systems, Vol. 51, 1992, No 1, pp.117-118.