The notion of H-IFS in data modelling.

Ermir Rogova[1]
Panagiotis Chountas[1]
Krassimir T. Atanassov[2]

1 School of Electronics and Computer Science, University of Westminster
2 CLBME, Bulgarian Academy of Sciences

The Notion of H-IFS in Data Modelling

Ermir Rogova, Panagiotis Chountas, Krassimir Atanassov

***Abstract*—In this paper we revise the context of "value imprecision", as part of an knowledge–based environment. We present our approach for including value imprecision as part of a non-rigid hierarchical structures of organization. This led us to introduce the concept of closure of an Intuitionistic fuzzy set over a universe that has a hierarchical structure. Intuitively, in the closure of this Intuitionistic fuzzy set, the "kind of" relation is taken into account by propagating the degree associated with an element to its sub-elements in the hierarchy. We introduce the automatic analysis according to concepts defined as part of a knowledge hierarchy in order to guide the query answering as part of an integrated database environment with the aid of hierarchical intuitionistic fuzzy sets.**

I. Introduction

Background knowledge of data is often available, arising from a concept hierarchy, as integrity constraints, from database integration, or from knowledge possessed by domain experts. Frequently integrated DBMSs contain incomplete data which we may represent using H-IFS to declare support contained in subsets of the domain. These subsets may be represented in the database as partial values, which are derived from background knowledge using conceptual modelling to re-engineer the integrated DBMS. For example, we may know that the value of the attribute JOB-DESCRIPTION is unknown for the tuple relating to employee Natalie but also know from the attribute salary that Natalie receives an estimated salary in the range of €25K ~Salary$_{25K}$. A logic program, using a declarative language can then derive the result that Natalie is a "Junior-Staff", which we input to the attribute JOB-DESCRIPTION of tuple Natalie in the re-engineered database.

Generalised relations have been proposed to provide ways of storing and retrieving data. Data may be imprecise, hence we are not certain about the specific value of an attribute but only that it takes a value which is a member of a set of possible values. An extended relational model for assigning data to sets has been proposed by [1]. This approach may be used either to answer queries for decision making or for the extraction of patterns and knowledge discovery from relational databases. It is therefore important that appropriate functionality is provided for database systems to handle such information. A model, which is based on partial values [2], has been proposed to handle imprecise data. Partial values may be thought of as a generalisation of null values where, rather than not knowing anything about a particular attribute value, we may identify the attribute as a set of possible values. A partial value is therefore a set such that exactly one of the values in the set is the true value.

Open nulls is the main representative of the possible-unweighted–unrestricted branch. Universal nulls may also be classified under this branch assuming the OWA semantics. Inclusive disjunctive information, possible information and maybe tuples or values are indicative representatives of the possible-unweighted-restricted school.

In [3] five different types of nulls are suggested. The labels and semantics of them are defined as follows. Let V be a function, which takes a label and returns a set of possible values that the label may have.

Label (X)	V(X)
Ex-mar	D
Ma-mar	$D \cup \{\perp\}$
Pl-mar	$\{\perp\}$
Par-mar (V_s)	V_s
Pm-mar (V_s)	$V_s \cup \{\perp\}$

Fig. 1. Types of NULL and their semantics

Intuitively, V (Ex-mar) = D says that the actual value of an existential marker can be any member of the domain D. Likewise, V (Ma-mar) = $D \cup \{\perp\}$ says that the actual value of a maybe marker can be either any member of D, or the symbol $\perp$, denoting a non-existent value. Similarly, V (Par-mar (V s)) = V_s says that the actual value of a partial null marker of the form pa mar (V_s) lies in the set V_s, a subset of the domain D. An important issue is the use of $\perp$, which denotes that an attribute is inapplicable. However such an interpretation of the unknown information, is not consistent with the principles of conceptual modelling. Assuming the sample fact spouse, the individual, Tony, is a bachelor and hence, the wife field is inapplicable to him, $\perp$. Conceptually the issue can be resolved with the use of the subtypes (e.g. married, unmarried) as part of the entity class Person. A subtype is introduced only when there is at least one role recorded for that subtype. The conceptual treatment of null will permit us to reduce the table in Fig.1 using only two types of null markers.

E. Rogova is with the DKMG – HSCS, University of Westminster, Watford Road, Northwick Park, London, HA1 3TP, UK, e-mail: rogovae@wmin.ac.uk.

P. Chountas is with the DKMG – HSCS, University of Westminster, Watford Road, Northwick Park, London, HA1 3TP, UK, e-mail: chountp@wmin.ac.uk.

K. Atanassov is with the CLBME, Bulgarian Academy of Sciences, Bl. 105, Sofia-1113, BULGARIA, email: krat@argo.bas.bg.

978-1-4244-1819-0/08/$25.00©2008 IEEE

Label ($_X$)	V($_X$)
V-mar (V)	{V}
P-mar (V_s)	{V_s}
Π-mar (D-V_s)	{D –V_s}

Fig. 2. The reduced set of NULL values

In the general case the algebraic issue under the use of subtypes is whether the population of the subtypes in relationship to the super type is:

- Total and Disjoint: Populations are mutually exclusive and collectively exhaustive.
- Non-Total and Disjoint: Populations are mutually exclusive but not exhaustive.
- Total and Overlapping: Common members between subtypes and collectively exhaustive, in relationship to super type.
- Non-Total and Overlapping: Common members between subtypes and not collectively exhaustive, in relationship to super type.

Conclusively it can be said that a null value is often semantically overloaded to mean either an unknown value or an inapplicable.

In the bibliography concerning the introduction of fuzzy methods for replacing unknown values with the aid of background knowledge, several issues have been dealt with but are quite distant from our proposal. We can note two main categories of papers, especially in recent research.

In studies about possibilistic ontologies [4], each term of the ontology is considered as a linguistic label and has an associated fuzzy description. Fuzzy pattern matching between different ontologies is then computed using these fuzzy descriptions. This approach is related to those concerning the introduction of fuzzy attribute values in the object relational model [5].

Also, studies about fuzzy thesauri have discussed different natures of relations between concepts. Fuzzy thesauri have been considered, for instance, in [6].

Work reported in [7, 8] in parallel to our framework is considering the problem of obtaining a family of fuzzy clusters with clear overlapping by allowing objects to fully belong to several classes. In this framework, the hesitation margin [9, 10], [11] denoting to what extent the overlapping occurs was not considered and cannot be represented directly in the fuzzy hierarchies, classes/clusters. As a result, the ordering and ranking of the query results will differ. Furthermore, we make use of different types of background knowledge in order to restrict the scope of the query and the number of clusters.

These observations led us to introduce the concept of closure of an Intuitionistic hierarchical fuzzy set, which is a developed form defined on the whole hierarchy. Intuitively, in the closure of a hierarchical Intuitionistic fuzzy set, the "kind of" relation is taken into account by propagating the degree associated with an element to its sub-elements more specific elements in the hierarchy. The rest of the paper is organized as follows:

- Section II outlines the principles of the IFS and delivers the basic definitions and properties of the H-IFS.
- Section III proposes a method for replacing null values with the aid of background knowledge as part of an integrated database environment.
- Section IV delivers the representation of a H-IFS with the aid of concept tables.
- Section V extends the definitions of traditional aggregate operators for dealing with flexible hierarchical structures of organizations.

I. Intuitionistic Fuzzy Sets and Hierarchical-IFS

A. IFS – Atanassov's Sets

Each element of an Intuitionistic fuzzy [12, 13] set has degrees of membership or truth (μ) and non-membership or falsity (ν), which don't sum up to 1.0 thus leaving a degree of hesitation margin (π).

As opposed to the classical definition of a fuzzy set given by $A' = \{< x, \mu_{A'}(x) > | x \in X\}$ where $\mu_A(x) \in [0, 1]$ is the membership function of the fuzzy set A′, an Intuitionistic fuzzy set A is given by

$$A = \{< x, \mu_A(x), \nu_A(x) > | x \in X\}$$

where: $\mu_A : X \to [0, 1]$ and $\nu_A : X \to [0, 1]$ such that $0 < \mu_A(x) + \nu_A(x) < 1$ and $\mu_A(x)\ \nu_A(x) \in [0, 1]$ denote a degree of membership and a degree of non-membership of $x \in A$, respectively.

Obviously, each fuzzy set may be represented by the following Intuitionistic fuzzy set

$$A=\{<x, \mu_A'(x), (x), 1-\mu_A'(x)>|x \in X\}$$

For each Intuitionistic fuzzy set in X, we will call $\pi_A(x) = 1 - \mu_A(x) - \nu_A(x)$ an Intuitionistic fuzzy index (or a hesitation margin) of $x \in A$ which expresses a lack of knowledge of whether x belongs to A or not. For each $x \in A$ $0<\pi_A(x)<1$.

Definition 1. Let A and B be two fuzzy sets defined on a domain X. A is included in B (denoted $A \subseteq B$) if and only if their membership functions and non-membership functions satisfy the condition: $(\forall \chi \in X)\ (\mu A(x) \le \mu B(x)\ \&\ \nu A(x) \ge \nu B(x))$

Two scalar measures are classically used in classical fuzzy pattern matching to evaluate the compatibility between an ill-known datum and a flexible query, known as

- a possibility degree of matching, **Π(Q; D)**
- a necessity degree of matching, **N(Q/ D)**

Definition 2. Let Q and D be two Intuitionistic fuzzy sets defined on a domain X and representing, respectively, a flexible query Q and an ill-known datum D:

- The possibility degree of matching between Q and D, denoted Π(Q; D), is an "optimistic" degree of overlapping that measures the maximum compatibility between Q and D, and is defined by: $\Pi(Q / D) = \sup_{x \in X} \min (< 1-\nu_Q(x), \nu_Q(x) >, < 1-\nu_D(x), \nu_D(x) >)$
- The necessity degree of matching between Q and D, denoted N(Q; D), is a "pessimistic" degree of inclu-

sion that estimates the extent to which it is certain that D is compatible with Q, and is defined by:

$N(Q/D) = \inf_{x \in X} \max(<\mu_Q(x), 1-\mu_Q(x)>, <\mu_D(x), 1-\mu_D(x)>)$

B. H-IFS

The definition domains of the hierarchical fuzzy sets [14, 15, 16] that we propose below are subsets of hierarchies composed of elements partially ordered by the "kind of" relation. An element l_i is more general than an element l_j (denoted $l_i \sim l_j$), if l_i is a predecessor of l_j in the partial order induced by the "kind of" relation of the hierarchy. An example of such a hierarchy is given in Fig. 3. A hierarchical intuitionistic fuzzy set is then defined as follows.

Definition 3. Let F be a H-IFS defined on a subset D of the elements of a hierarchy L. It degree is denoted as $<\mu, \nu>$. The closure of F, denoted clos(F), is a H-IFS defined on the whole set of elements of L and its degree $<\mu, \nu>_{clos(F)}$ is defined as follows.

For each element **l** of L, let $S_L = \{l_1, \ldots, l_n\}$ be the set of the smallest super-elements in D.

If S_L is not empty,

$<\mu, \nu>_{clos(F)}(S_L) = <\max_{1\leq i\leq n}(\mu(L_i)), \min_{1\leq i\leq n}(\nu(L_i))>$

else, $<\mu, \nu>_{clos(F)}(S_L) = <0, 0>$

In other words, the closure of a H-IFS F is built according to the following rules. For each element l_I of L:

- If l_I belongs to F, then l_I keeps the same degree in the closure of F (case where $S_L = \{ l_I \}$).
- If l_I has a unique smallest super-element l_I in F, then the degree associated with l_I is propagated to L in the closure of F, $S_L = \{ l_I \}$ with $l_I > l_I$)

If L has several smallest super-elements $\{l_1, \ldots, l_n\}$ in F, with different degrees, a choice has to be made concerning the degree that will be associated with l_I in the closure. The proposition put forward in definition 3, consists of choosing the maximum degree of validity μ and minimum degree of non validity v associated with $\{l_1, \ldots, l_n\}$.

We focus on the fact that two different H-IFSs, defined on the same hierarchy, can have the same closure, as in the following example.

Example. The H-IFSs Q={Wine<1,0>, Red Wine<0.7,0.1>, Brown Wine<1,0>, White Wine <0.4,0.3>} and

R ={Wine<1,0>, Red Wine<0.7,0.1>, Brown Wine<1,0>, Pinot Noir <0.4,0.3>} have the same closure, represented Fig.3 below.

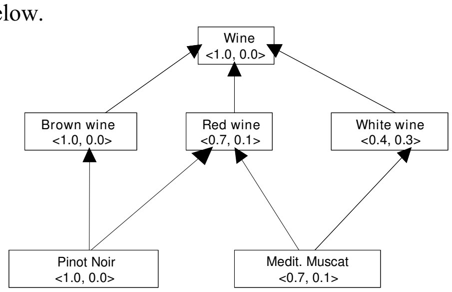

Fig. 3. Common closure of the H-IFSs Q and R

Such H-IFSs form equivalence classes with respect to their closures.

Definition 4. Two H-IFSs Q and R, defined on the same hierarchy, are said to be equivalent Q≡R if and only if they have the same closure

Property Let Q and R be two equivalent Intuitionistic hierarchical fuzzy sets. If $l_I \in$ dom(Q) $\cap$ dom(R), then $<\mu,\nu>(Q.l_I) = <\mu,\nu>(R.l_I)$

Proof According to the definition of the closure of a H-IFS F, definition 3, the closure of F preserves the degrees that are specified in F. As Q and R have the same closure (by definition of the equivalence), an element that belongs to Q and R necessarily has the same degree $<\mu,\nu>$ in both.

We can note that R contains the same element as Q with the same $<\mu,\nu>$, and also one more element Pinot Noir<1,0>. The $<\mu,\nu>$ associated with this additional element is the same as in the closure of Q. Then it can be said that the element, Pinot Noir<1,0> is derivable in R through Q.

The same conclusions can be drawn in the case of Medit. Muscat <0.7, 0.1>

Definition 5.. Let F be a hierarchical fuzzy set, with dom(F) = $\{l_1, \ldots, l_n\}$, and F_{-k} the H-IFS resulting from the restriction of F to the domain dom(F) \ $\{l_k\}$. l_k is deducible in F if:

$$<\mu, \nu>clos_{(F-k)}(l_k) = <\mu, \nu>clos_{(F)}(l_k)$$

As a first intuition, it can be said that removing a derivable element from a hierarchical fuzzy set allows one to eliminate redundant information. But, an element being derivable in F does not necessarily mean that removing it from F will have no consequence on the closure: removing k from F will not impact the degree associated with k itself in the closure, but it may impact the degrees of the sub-elements of k in the closure.

For instance, if the element Brown Wine is derivable in Q, according to definition 5, removing Brown Wine <1,0> from Q would not modify the degree of Brown Wine itself in the resulting closure, but it could modify the degree of its sub-element Pinot Noir. Thus, Brown Wine <1,0> cannot be derived or removed. This remark leads us to the following definition of a minimal hierarchical fuzzy set.

Definition 6. In a given equivalence class (that is, for a given closure C), a hierarchical fuzzy set is said to be **minimal** if its closure is C and if none of the elements of its domain is derivable**.**

Obtaining the Minimal H-IFS

Step 1: Assign Min-H-IFS ← ∅. Establish an order so that the sub-elements $\{l_1, \ldots, l_n\}$ of the hierarchy L are examined after its super-elements.

Step 2: Let l_1 be the first element and **$(l_1)/<\mu, \nu> \neq (l_1)/<0, 0>$** then add **$l_1$** to Min-H-IFS and **$<\mu, \nu>_{clos(Min-HIFS)}(l_1) = (l_1)/<\mu, \nu>$**.

Step 3: Let us assume that K elements of the hierarchy L satisfy the condition **$<\mu, \nu>_{clos(Min-HIFS)}(l_i) = (l_i)/<\mu, \nu>$.** In this case the Min-H-IFS do not change. Otherwise go to next element **l_{k+1}** and execute Step 4.

Step 4: The $\mathbf{l_{k+1}}$/<$\boldsymbol{\mu}_{\mathbf{k+1}}$, $\boldsymbol{\nu}_{\mathbf{k+1}}$> associated with $\mathbf{l_{k+1}}$**.** In this case $\mathbf{l_{k+1}}$ is added to Min-H-IFS with the corresponding <$\boldsymbol{\mu}_{\mathbf{k+1}}$, $\boldsymbol{\nu}_{\mathbf{k+1}}$>**.**

Step 5: Repeat steps three and four until $\mathbf{clos_{(Min\text{-}HIFS)}=C}$.

For instance the H-IFSs S_1 and S_2 are **minimal** (none of their elements is derivable). They cannot be reduced further.

S_1= Wine<1,0>

S_2= {Wine<1,0>, Red Wine<0.7,0.1>, Pinot Noir<1,0>, White Wine <0.4, 0.3>}

II. NULL VALUES & BACKGROUND KNOWLEDGE IN DBMS

In a generalised relational database we consider an attribute A and a tuple t_i of a relation R in which n attribute value $t_i[A]$ may be a partial value. A partial value is formally defined as follows.

Definition 7. A partial value is determined by a set of possible attribute values of tuple t of attribute A of which one and only one is the true value. We denote a partial value by $P = [a_1,...,a_n]$ corresponding to a set of P possible values $\{a_1,... , a_n\}$ of the same domain, in which exactly one of these values is the true value of . Here, P is the cardinality of $\{a_1,... , a_n\}$ is a subset of the domain set $\{a_0,... , a_{n+1}\}$ of attribute A of relation R, and $P \leq n+1$.

Queries may require operations to be performed on partial values; this can result in a query being answered by means of bags, where the tuples have partial attribute values [17].

Definition 8. An Intuitionistic Fuzzy partial value relation R, is a relation based on partial values for domains $D_1, D_2,..., D_n$ of attributes $A_1, A_2,..., A_n$ where $R \subseteq P_1 \times P_2 \times \times P_n$ and P_i is the set of all the partial values on power set of domain D_i. A pruned value of attribute A_i of the relation R corresponds then to a **H-IFS** which is a subset of the domain D_i. An example of a partial value relation is presented in Table 1 below:

Let l be an element defined by a structured domain D_i. $U(e)$ is the set of higher level concepts:

$U(e) = \{n|n \in D_i \wedge n$ is an ancestor of $l\}$, and

$L(e)$ is the set of lower concepts $L(e) = \{n|n \in D_i \wedge n$ is a descendent of $l\}$.

__**Rule-1:**__ If $(|U(e)| > 1 \wedge L(e) = \varnothing)$, then it is simply declared that a child or base concept has many parents. Therefore a child or base concept acting as a selection predicate can claim any tuple (parent) containing elements found in $U(e)$, as its ancestor. If both arguments are high level concepts or low level concepts then $B((l_1),(l_2)) = \varnothing$.

__**Rule-2:**__ If $B((l_1),(l_2)$ is defined and $|B((l_1),(l_2))| > 1$, then it is simply declared that multiple parents, high level concepts, are receiving a base concept as their own child. Therefore a parent or high level concept acting as a selection predicate can claim any tuple (child) containing elements found in $(L(l_1) \wedge (l_2))$, as its descendant, but with variants level of certainty.

A. *Replacing Unknown Attribute Values with H-IFS*

Concept hierarchies have previously been used for attribute-induced knowledge discovery [18]. However the proposed use of background knowledge in this context is unique.

We assume that original attribute values may be given either as singleton sets, or subsets of the domain, or as concepts, which correspond to subsets of an attribute domain. In the last case the values may be defined in terms of a concept hierarchy. In addition there are rules describing the domain, and these may be formed in a number of ways: they may take the form of integrity constraints, where we have certain restrictions on domain values; functional dependencies and also rules specified by a domain expert.

All descendents of an instance of a high-level concept are replaced with a minimal H-IFS has these descendents as members. A null value is regarded as a partial value with all base domain values as members. We refer to the resultant partial value, obtained as a result of this process, as a primal partial value. The replacement process is thus performed by the following procedure:

TABLE I
REPLACEMENT PROCEDURE

Procedure: Replacement
Input: A concept table R consisting of partial values, or nulls.
Output: A re-engineered partial value table U.
Method: For each attribute value of R recursively replace the cell value by a **primal** partial value. For each cell of R replace, the primal partial value by a pruned prime-partial –value, until a minimal partial value is reached.

If a particular member of a partial value violates the domain constraint (rule) then it is pruned from the minimal H-IFS primal partial value. This process is continued until all partial values have been pruned by the constraints as much as possible. We refer to the resultant partial value, obtained as a result of this process, as a minimal partial value.

B. *Forms of Background Knowledge*

Background knowledge may be specified as arising from a hierarchical Intuitionistic Fuzzy hierarchy, as integrity constraints, from the integration of conflicting databases, or from knowledge selected by domain experts. Using such information we offer to re-engineer the database by replacing missing, conflicting or unacceptable data by sets of the attribute domain.

In an integrated DBMS environment it will be also useful not to query all sources, but only those that contain information relevant to our request. This is quite critical for achieving better query performance. For this reason we equip our integrated architecture with a repository that contains various constraints (i.e. Intuitionistic Fuzzy Range Constraints, Intuitionistic Fuzzy Functional Dependencies, etc) that are related to the information sources that participate in the Integrated Architecture.

Range constraints: such as "The average income per person is estimated to be in the range of €50K". Considering a finite universe of discourse, say X whose cardinality is N. Let us suppose that $X=\{X_1, X_2, \ldots, X_n\}$ and the Intuitionistic fuzzy number ~a given by ~a $=\{(x_i, \mu_i, \nu_i): x_i \in X, I = 1,2\ldots N\}$ We can express the above constraint as follows **~Income50K {(49, .8, .1), (50, .9, .02) (51, .7, .15)}**

Classical data integrity constraints such as "All persons stored at a source have a unique identifier".

Functional Dependencies: for instance, a source relation S1(Name, lives, income, Occupation) has a functional dependency Name→(Lives, ~Income).These constraints are very useful to compute answers to queries.

There are several reasons we want to consider constraints separately from the query language. Describing constraints separately from the query language can allow us to do reasoning about the usefulness of a data source with respect to a valid user request.

Some of source constraints can be naturally represented as local constraints. Each **local constraint** is defined on one data source only. These constraints carry a rich set of semantics, which can be utilized in query processing. Any projected database instance of source, these conditions must be satisfied by the tuples in the database.

Definition 9. Let si,...,sl be l sources in a data-integrated system. Let P = {pi,..., pn } be a set of global predicates, on which the contents of each source s are defined. A general global constraint is a condition that should be satisfied by any database instance of the global predicates P.

General global constraints can be introduced during the design phase of such a data-integration system. That is, even if new sources join or existing ones leave the system, it is assumed that these constraints should be satisfied by any database instance of the global predicates. Given the global predicate **Income**, if a query asks for citizens with an average income above **~Income60K**, without checking the source contents and constraints, the integrated system can immediately know that the answer is empty.

To this extent we can interrogate the constraints repository to find out if a particular source contains relevant information with respect to particular request. We now consider the problem of aggregation for the partial value data model. In what follows we are concerned with symbolic attributes, which are typically described by counts and summarised by aggregated tables. The objective is to provide an aggregation operator which allows us to aggregate individual tuples to form summary tables.

III. Background Knowledge-Based Table

The structure of any H-IFS can be described by a domain concept relation DCR = (Concept, Element), where each tuple describes a relation between elements of the domain on different levels. The DCR can be used in calculating recursively [19] the different summarisation or selection paths as follows:

$$PATH \leftarrow DCR_{\{x=1\ldots(n-2)\,|\,n>2\}} \bowtie DCR_x$$

If $n \leq 2$, then DCR becomes the Path table as it describes all summarisation and selection paths.

These are entries to a knowledge table that holds the metadata on parent-child relationships. An example is presented below:

TABLE II
Domain Concept Relation

DCR	
Concept	**Element**
Wine <1.0, 0.0>	Brown Wine <1.0, 0.0>
Wine <1.0, 0.0>	Red Wine <0.7, 0.1>
Wine <1.0, 0.0>	White Wine <0.4, 0.3>
Brown Wine <1.0, 0.0>	Pinot Noir <1.0, 0.0>
Red Wine <0.7, 0.1>	Pinot Noir <1.0, 0.0>
Red Wine <0.7, 0.1>	Medit. Muscat <0.7, 0.1>
White Wine <0.4, 0.3>	Medit. Muscat <0.7, 0.1>

Table II shows how our Wine hierarchy knowledge table is kept. Paths are created by running a recursive query that reflects the 'PATH' algebraic statement. The hierarchical IFS used as example throughout this paper comprises of 3 levels, thus calling for the SQL-like query as below:

SELECT A.Concept as Grand-concept, b.concept, b.element
FROM DCR as A, DCR as B
WHERE A.child=B.parent;

This query will produce the following paths:

TABLE III
Path Table

Path			
Grand-concept	**Concept**	**Element**	**Path Colour**
Wine <1.0, 0.0>	Brown Wine <1.0, 0.0>	Pinot Noir <1.0, 0.0>	Red
Wine <1.0, 0.0>	Red Wine <0.7, 0.1>	Pinot Noir <1.0, 0.0>	Blue
Wine <1.0, 0.0>	Red Wine <0.7, 0.1>	Medit. Muscat <0.7, 0.1>	Green
Wine <1.0, 0.0>	White Wine <1.0, 0.0>	Medit. Muscat <0.7, 0.1>	Brown

Fig. 4 presents a pictorial view of the four distinct summarisation and selection paths.

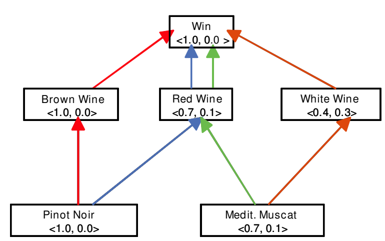

Fig. **4.** Pictorial representation of paths

These paths will be used in fuzzy queries to extract answers that could be either definite or possible. This will be realised with the aid of the 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. ∧,

$\vee$, etc.). Consider a predicate θ that takes the value "Red Wine", θ = "Red Wine".

After utilizing the IFS hierarchy presented in Fig.7, this predicate can be reconstructed as follows:

$\theta = \theta_1 \vee \theta_2 \vee ... \vee \theta_n$

In our example, θ_1="Red Wine", θ_2="Pinot Noir" and θ_n="Medit. Muscat".

The reconstructed predicate θ = (Red Wine $\vee$ Pinot Noir $\vee$ Medit. Muscat) allows the query mechanism to not only definite answers, but also possible answers [20].

In terms a query retrieving data from a summary table, the output contains not only records that match the initial condition, but also those that satisfy the reconstructed predicate. Consider the case where no records satisfy the initial condition (Red Wine). Traditional aggregation query would have returned no answer, however, based on our approach, the extended query would even in this case, return an answer, though only a possible one, with a specific belief and disbelief $<\mu, \nu>$. It will point to those records that satisfy the reconstructed predicate θ, more specifically, "Pinot Noir and Medit. Muscat".

Following the representation of H-IFS as concept relations and the definition of summarisation paths, there is still a need to extend the traditional aggregation operators in order to cope with flexible hierarchies of data organisations.

IV. Aggregation Operators

Aggregation (A): An aggregation operator A is a function $A(G)$ where $G = \{<x, \mu_F(x), \nu_F(x)> | x \in X\}$ where $x=<att_1, ..., att_n>$ is an ordered tuple belonging to a given universe X, $\{att_1, ..., att_n\}$ 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 $\{<x', \mu_F(x'), \nu_F(x')> | x' \in X\}$. 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: $R_i = (l, F, H)$ and the function $A(G)$

Output: $R_o = (l_o, F_o, H_o)$ *where*

- l is a set of levels $l_1, ..., l_n$ that belong to a partial order $\leq O$ To identify the level l as part of a hierarchy we use dl.
  $l_\perp$: base level $l_\top$: top level
  for each pair of levels l_i and l_j we have the relation
  $\mu_{ij} : l_i \times l_j \rightarrow [0,1]$ $\nu_{ij} : l_i \times l_j \rightarrow [0,1]$, $0 < \mu_{ij} + \nu_{ij} < 1$
- F is a set of fact instances with schema $F = \{<x, \mu_F(x), \nu_F(x)> | x \in X\}$, where $x=<att_1, ..., att_n>$ is an ordered tuple belonging to a given universe X, $\mu_F(x)$ and $\nu_F(x)$ 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 structure(l, F, H') which allows us to trace back the evolution of a structure after performing a set of operators i.e. aggregation

The definition of the extended group operators allows us 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 relation $R_i=(l_i, F_i, H_i)$ is another relation $R_o=(l_o, F_o, H_o)$

Input: $R_i = (l_i, F_i, H_i)$

Output: $R_o = (l_o, F_o, H_o)$

An object of type history is a recursive structure:

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

The structured history of the relation allows us to keep all the information when applying *Roll up* and get it all back when *Roll Down* is performed. To be able to apply the operation of *Roll Up* we need to make use of the IF_{SUM} aggregation operator.

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 relation $R_i = (l, F, H)$ having $H=(l', A', H')$ is another relation $R_o=(l', F', H')$.

Input: $R_i=(l, F, H)$

Output: $R_o=(l', F', H')$ where $F' \rightarrow$ *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.

The definition of aggregation operator points to the need of defining the IF extensions for traditional group operators [20], such as *SUM, AVG, MIN and MAX.* Based on the standard group operators, we provide their IF extensions and meaning.

IF_{SUM}: The IF_{sum} aggregate, like its standard counterpart, is only defined for numeric domains. The relation R consists of tuples R_i *with* $1 \leq i \leq m$. The tuples R_i are assumed to take Intuitionistic Fuzzy values for the attribute att_{n-1} for $i = 1$ *to* m we have $R_i[att_{n-1}] = \{<\mu_i(u_{ki}), \nu_i(u_{ki})>/ u_{ki} \mid 1 \leq k_i \leq n\}$. The IF_{sum} of the attribute att_{n-1} of the relation R is defined by:

$$IF_{SUM}((att_{n-1})(R)) =$$

$$\{<u>/ y \mid ((u = \min_{i=1}^{m}(\mu_i(u_{ki}), \nu_i(u_{ki})) \wedge (y = \sum_{ki=k1}^{km} u_{ki}) \; (\forall_{k1, ...km} : 1 \leq k1, ...km \leq n))\}$$

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

$$IF_{AVG}((att_{n-1})(R)) = IF_{SUM}((att_{n-1})(R)) / COUNT((att_{n-1})(R))$$

IF_{MAX}: The IF_{MAX} aggregate, like its standard counterpart, is only defined for numeric domains. The IF_{sum} of the attribute att_{n-1} of the relation R is defined by:

$IF_{MAX}((att_{n-1})(R)) =$
$\{<u>y|((u=\min_{i=1}^{m}(\mu_i(u_{ki}),v_i(u_{ki}))\wedge$
$(y=\max_{i=1}^{m}(\mu_i(u_{ki}),v_i(u_{ki})))(\forall_{k1,\ldots km}:1\leq k1,\ldots km\leq n))\}$

***IF_{MIN}*:** The IF_{MIN} aggregate, like its standard counterpart, is only defined for numeric domains. Given a relation R 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 relation R consists of tuples R_i *with* $1 \leq i \leq m$. Tuples R_i are assumed to take Intuitionistic Fuzzy values for the attribute att_{n-1} for $i = 1$ *to* m we have $R_i[att_{n-1}] = \{<\mu_i(u_{ki}), v_i(u_{ki})>/ u_{ki} \mid 1 \leq k_i \leq n\}$. The IF_{sum} of the attribute att_{n-1} of the relation R is defined by:
$IF_{MIN}((att_{n-1})(R)) = \{<u>/ y|((u=\min_{i=1}^{m}(\mu_i(u_{ki}),v_i(u_{ki}))\wedge$
$(y=\min_{i=1}^{m}(\mu_i(u_{ki}),v_i(u_{ki})))(\forall_{k1,\ldots km}:1\leq k1,\ldots km\leq n))\}$

We can observe that the IF_{MIN} is extended in the same manner as IF_{MAX} aggregate except for replacing the symbol **max** in the IF_{MAX} definition with **min.**

V. CONCLUSION

We provide a means of using background knowledge to re-engineer the data representation into a partial value representation with the aid of H-IFS and Intuitionistic Fuzzy relational representation.

The hierarchical links are defined by the “kind of, ≤” relation. The membership of an element in a H-IFS has consequences on the membership and non-membership of its sub elements in this set.

The proposed methodology aims at expanding the user preferences expressed when defining a query, in order to obtain related and complementary answers.

This is likely to be a useful tool for decision support and knowledge discovery in, for example, data mediators, data warehouses, where the data are often subject to such imperfections. Furthermore we notice that our approach can be used for the representation of Intuitionistic Fuzzy Linguistic terms.

REFERENCES

[1] D. Bell, J. Guan, S. Lee “*Generalized union and project operations for pooling uncertain and imprecise information*”. DKE 18 (1996) 89-117

[2] A. Chen, F. Tseng “*Evaluating aggregate operations over imprecise data”*. IEEE Trans. on Knowledge and Data Engineering, 8 (1996) 273-284

[3] C. Zaniolo “*Database relations with null values”*. J. Comput. Syst. Sci. Vol.28 (1984) 142–166.

[4] “*Evaluation of Term-Based Queries Using Possibilistic Ontologies*,” Soft Computing for Information Retrieval on the Web, E. Herrera-Viedma, G. Pasi, and F. Crestani, eds. Springer-Verlag,2005.

[5] M. Koyuncu, A. Yazici “IFOOD: *An Intelligent Fuzzy Object-Oriented Database Architecture”*. IEEE Trans. Knowl. Data Eng. 15(5): pp. 1137-1154, 2003

[6] G. Bordogna, G. Pasi “*Modeling Vagueness in Information Retrieval”*. ESSIR 2000: 207-241

[7] R. Thomopoulos, P. Buche, O. Haemmerlé: *“Fuzzy Sets Defined on a Hierarchical Domain”*. IEEE Trans. Knowl. Data Eng. 18(10): 1397-1410 (2006)

[8] A. Amo, J. Montero, G. Biging, V. Cutello: “*Fuzzy classification systems”* European Journal of Operational Research (2004) Volume 156, Issue 2, pp.495-507

[9] E. Szmidt, J. Baldwin “*Intuitionistic Fuzzy Set Functions, Mass Assignment Theory, Possibility Theory and Histograms”*. 2006 IEEE World Congress on Computational Intelligence 237—243.

[10] E. Szmidt, J. Kacprzyk “*Intuitionistic fuzzy sets in group decision making”,* Notes on IFS, 2, 15—32. (1996)

[11] E. Szmidt, J. Kacprzyk “*Remarks on some applications of intuitionistic fuzzy sets in decision making”*, Notes on IFS, 2(3), (1996) 22—31.

[12] K. Atanassov „*Intuitionistic Fuzzy Sets“*, Heidelberg, Springer-Verlag, 1999.

[13] K. Atanassov “*Remarks on the Intuitionistic fuzzy sets”*. – Fuzzy Sets and Systems, Vol. 51, 1992, No 1, 117-118.

[14] E. Rogova, P. Chountas, K. Atanassov “*Flexible hierarchies and fuzzy knowledge-based OLAP”*. – FSKD 2007, IEEE Computer Society Press Vol.2, pp. 7-11

[15] P. Chountas “*On intuitionistic fuzzy sets over universes with hierarchical structures”*. – Notes on Intuitionistic Fuzzy Sets, Vol. 13, 2007, No 1, 52-56.

[16] E. Rogova, P. Chountas “*On imprecision intuitionistic fuzzy sets & OLAP – The case for KNOLAP*”, IFSA 2007, Springer-Verlag ISBN: 3540724338, 2007, pp.11-23

[17] V. Dhar, A. Tuzhilin “*Abstract-driven pattern discovery in databases”*. IEEE Trans. on Knowledge and Data Engineering 6 (1993) 926-938

[18] J. Han, Y. Fu “*Attribute-oriented induction in data mining”*. Advances in Knowledge Discovery. AAAI Press/MIT Press, Cambridge, MA (1996) 399-421

[19] A. Silberschatz, H. Korth, S. Sudarshan *“ Database System Concepts”*. McGraw-Hill

[20] S. Parsons “*Current approaches to handling imperfect information in data and knowledge bases”*. IEEE Trans. Knowl. Data Eng. 8(3) (1996) 353-372