Australian Journal of Basic and Applied Sciences, 5(7): 1297-1312, 2011
ISSN 1991-8178

Fuzzy-RDR: A New Approach to Create Expert Systems by Domain Expert

[1]Hossein M. Shirazi, [2]Rodrigo Martínez-Béjar, [3]Mohammad Mozdorani and [4]Paul Compton

[1]Faculty of Information and Communication Technology, Malek-Ashtar University of Technology, Tehran, Iran.
[2]Department of Information Engineering and Communications. Faculty of Informatics. Univesity of Murcia. Spain
[3]Faculty of Engineering, School of Science, University of Tehran, Tehran, Iran.
[4]Department of AI, School of Computer Science and Engineering, The University of New South Wales, Sydney, 2052, Australia.

Abstract: Expert systems are widely used in most of area such as industry, management, and business. However, still maintenance is a subject incorporating all these knowledge-based systems, including fuzzy rule-based system. Although, the use of fuzzy concept reduces the maintenance problems partially, it introduces other problems in tuning the numbers used by the system. Ripple Down Rules (RDR) method largely overcome the maintenance problems for conventional crisp systems. The aim of this work was to apply the RDR approach to fuzzy rules to facilitate the maintenance and tuning of the numbers involved. On the other hand the use of fuzzy concept in the RDR system allows more natural expressions of concepts as the main advantage of a fuzzy approach. This paper presents a theoretical foundation for combining RDR and fuzzy concepts.

Key words:

INTRODUCTION

The experiences have shown the maintenance phase of developed expert system is an important part of an expert system's life. In fact, many systems with initial signs of success have become useless when faced with difficulties in their maintenance phase. This is exactly what RDRs are aimed at combating. In order to implement a traditional expert system, both a knowledge engineer (KE) and an expert in that field are usually involved. The knowledge engineer's task is to elicit information from the expert and encode it in some fashion in the expert system. However, in RDR, the expert himself is in charge of both creating and maintaining the expert system.

RDR is a knowledge acquisition technique developed from the maintenance experience with an expert system (Compton and Jansen, 1989). The main motivation for developing RDR has been the fact that experts can not explain how they extracted conclusions. Nevertheless, they were capable to justify the conclusions by referring to the knowledgeable context in which they had been extracted (Compton and Jansen, 1990). RDR uniquely uses the knowledge provided by experts in the context it was provided, that is, by following the sequence of evaluated rules. Moreover, if the expert did not agree with a conclusion, knowledge in the form of a new rule could be added. In this sense, rules are never removed or corrected, only added.

Multiple Classification Ripple Down Rules (MCRDR) (Dazeley and Kang, 2003) is an extension of the basic aspects of RDR for providing multiple independent conclusions. A further RDR-based approach developed recently is the so-called Ripple down rule-Oriented Conceptual Hierarchy (ROCH) (Martinez-Bejar *et al.,* 1997b; Vivancos Vicente *et al.,* 2004). The basic idea concerning ROCH-based systems is that experts themselves can construct and validate domain knowledge while they enter knowledge to the knowledge base (KB).

The purpose of this paper is to obtain a formal model to extend the knowledge acquisition representation schema underlying current ROCH-based systems so that fuzzy domain knowledge can be acquired/represented. The aim is fulfilled by exploiting the characteristics of fuzzy domains in combination with the RDR theory. Thus, by introducing a few assumptions, a model for acquiring knowledge in fuzzy context through RDR has been developed. The idea in this article will generally be illustrated together with an existing KBS project for

Corresponding Author: Hossein Shirazi, Faculty of Information and Communication Technology, Malek-Ashtar University of Technology, Tehran, Iran.
Email: shirazi@mut.ac.ir

environmental planning in Spain.

2. Expert Systems:

Since early 1960s, expert systems usages have become widespread in different area such as: manufacturing, business and management (Seker *et al.,* 2003; Ismail *et al.,* 2003; Valencia-Garcia *et al.,* 2004). The purpose of an expert system is to emulate human experts at work by attempting to reason with knowledge. In creating an expert system, the most important challenge is how to mimic human expert and how to inference from this knowledge. Knowledge acquisition is the bottleneck of creating expert systems (William Siler and James, 2005). There are two main approaches to overcome this problem. Providing some easy and useful tools for the expert to enter his/her knowledge into the knowledge base of the expert system or using some machine learning techniques to automatically capture the knowledge. Ripple Down Rules (RDR) has been introduced based on the first idea.

2.1. Ripple Down Rules (RDR):

Ripple Down Rules (RDR) is based on the idea that when a knowledge based system (KBS) makes an incorrect conclusion the new rule that is added to correct that conclusion should only be used in the same context in which the mistake was made (Compton and Jansen, 1990). In practice this means attaching the rule at the end of the sequence of rules that were evaluated leading to the wrong conclusion. Thus, this rule will only be evaluated in the same context in which the mistake was made. Rules are thus added when errors are made, allowing for an incremental, maintenance-focused approach to knowledge acquisition. Since rules are added to correct the interpretation of a specific case, the system ends up not only with a number of rules but also with cases associated with these rules: the cases that prompted the addition of the rule. This leads to validated knowledge acquisition. The new added rule should not be allowed to be satisfied by any of the cases for which rules have already been added and for which different conclusions are appropriate. These cases are already handled.

For single classification problems, where the output of the system is one set of conclusions, a binary tree structure is used. The new rule added after the last rule evaluated. It is worth noting that the tree is extremely unbalanced and resembles a decision list with decision list refinements rather than a tree (Compton and Jansen, 1990). With this structure, the only case that can reach the new conclusion is the one associated with the rule that gives the incorrect conclusion. Implicit in this outline is a non-monotonic approach in that each rule proposes a final conclusion as an output, unless this conclusion is replaced with a later correction rule that fires. Rules never to be removed or edited as all errors are to be corrected by adding new rules.

A single classification RDR approach was used to build the PEIRS system for pathology interpretation. This system was largely developed whilst in routine use. Apart from the development of the initial domain model, the knowledge base was developed entirely by an expert without knowledge engineering support or skills. The only task had the expert, was to identify features in the case which distinguished it from the case associated with the rule giving the wrong conclusion. This allowed the expert to add about 2400 rules at the rate of about 3 minutes per rule and the system ended up being in routine use for four years and covering about 25% of chemicals (Compton and Jansen, 1990). This compares very favorably with other medical expert system projects and with expert system projects in general in terms of knowledge acquisition tasks. A range of studies have also been carried out demonstrating that problems with repetition that may occur with this approach happen less in practice than might be expected (Edwards *et al.,* 1993).

The notion of repetition in the KB is an important one with respect to a trade-off between knowledge and PSM power. The use of a binary tree structure means that knowledge in one particular pathway is not accessible to a search down another pathway. However, as noted in practice the repetition is small, while the speed of knowledge acquisition with a 'chunk' added every three minutes meets the requirements of the new DARPA project on very large knowledge based systems. The requirement for this project is for KBS with between 10000 and 100000 chunks of knowledge to be built at this rate. PEIRS was only 2500 rules, but the cost of knowledge addition was close to constant as the KB grew, where the DARPA requirement is closed to linear with KB size, suggesting the DARPA requirement could be met.

2.2. Ripple down rule-Oriented Conceptual Hierarchy (ROCH):

ROCH (Martinez-Bejar *et al.,* 1997b) is the new implementation of RDR that allows experts to construct and validate domain knowledge ontology while they enter knowledge into RDR-based systems. In ROCH-based systems, for every application domain, experts construct cases by using a conceptual hierarchy (ROCH). The rules are created and maintained by the experts themselves. Such a hierarchy includes both IS-A and PART-OF relationships. Although, the expert must establish these relationships, they are also verified by the system. For instance, consider the following example (extracted from (Martinez-Bejar *et al.,* 1997b)).

Suppose that an expert wishes to create a RDR system from the following pair of ripple down rules (expressed in natural language)

$R(t_i)$ = {"*If the vegetation is very short, there is only one stratum and the seasonal variation is medium, then the area under study has got a low visual fragility (VF)*", "*If there exists a predominance of pine merged with stone outcrops then the area under study has got a high visual quality (VQ)*"} = $\{r_1, r_2\}$.

Let us also assume that there is no conceptualization yet (as the expert wants to create a KB). Then, the expert would establish the following conceptual hierarchy in two steps (corresponding to each rule):

Step 1 (for $\{r_1\}$)
vegetation_landuse (VF)
PART-OF
natural_vegetation (height, number_of_strata,seasonal_variation)
Step 2 (for $\{r_1, r_2\}$)
vegetation_landuse (VF, VQ)
PART-OF
natural_vegetation (height, number_of_strata,seasonal_variation)
IS-A
pine (merging_with_stone_outcrops, predominance)

The information into brackets after each concept stands for the properties associated to it. At this point, after the system verifies that this conceptual hierarchy is consistent, the system could compound the rules to be incorporated in the KB.

3. Fuzzy Expert Systems:

Fuzzy logic, which has widely been used for representing imprecise knowledge in expert systems (Phayung Meesad, 2001; Roubos and Setnes, 2001). It is basically a method to allow a gradual representation of likeness between two objects. It is based on the theory of fuzzy sets (Zadeh, 1965). It defines a membership function to assign a grade of membership between 0 and 1 to each element in the range of all possible elements under consideration. This grade can be thought as a measure of compatibility between the element and the concept represented by the fuzzy set. Formally, the membership function for a fuzzy set A, written $m_A(x)$, is a real valued function defined as the application m_A: X ® [0,1] for all x in a universal set X.

Fuzzy sets are used to model the vagueness and imprecision present in natural language. Thus, they can be employed to represent concepts such as *rarely* and *often*. These terms are usually known as linguistic quantifiers and are very common in natural language. The amount of vagueness can be regulated by means of the so-called modifiers or hedges, which act in combination with linguistic quantifiers. Examples of modifiers are *more* and *very*.

In order to make different applications possible, several kinds of inference should be allowed. In this work the Generalized Modus Ponens (GMP) approach, which has been used by some successful expert systems (Hwang, 1995) has been chosen. Such an approach provides a framework where different types of inference semantics can be supported.

Through the GMP technique, the proposition *y is B* can be derived from the rule "if (x is A) then (y is B)" when the proposition *x is A* is true. The GMP can also be employed when the two propositions *x is A* and *y is B* are defined imprecisely. Thus, if a proposition *x is A'*, close to *x is A*, is true, the principle of the GMP is to derive another proposition, written y is B'. This proposition is generated by taking into account both the underlying semantics of the implication of the rule and a measure of the likeness between A and A'. With all, the inference consists of defining a fuzzy set B' which is as close to B as A' is to A. More formally, Dubois and Prade (Dubois and Prade, 1988) have computed the membership function of B', written $m_{B'}$, as follows:

$\forall y \in Y, \mu_{B'}(y) = \sup_{x \in X} T(\mu_{A'}(x), (x \rightarrow y))$ where
Y is the referential of y;
X is the referential of x;
T is a triangular norm that makes the GMP compatible with the classical modus pones;
$\mu_{A'}$ is the membership function of A'; and
$(x \rightarrow y)$ represents an implication denoting the kind of causal link involved in the production rule.

According to Buckley *et al.* (1986)], a fuzzy expert system (FES) must hold the following:

- it must contain a set of rules that can deal with fuzzy tags, fuzzy sets and relations;
- it becomes feasible to process the data in a particular knowledge base (KB);
- it must provide an appropriate output, which corresponds to a particular input.

More formally, every FES can be characterized by the following:

1. it can manipulate fuzzy terms;
2. its input is comprised of imprecise attributes, which can be represented by either discrete fuzzy sets or continuous fuzzy sets.
3. its rules are defined so that they can operate with fuzzy data.
4. the final result is a fuzzy set.

A number of FESs have been successfully implemented to-date. Examples of these can be found in (Seker *et al.,* 2003; Yen and Meesad, 2000). Some of the quoted systems use Gaussian models for defining membership functions. These are particularly interesting when the usual values given to the fuzzy terms employed by experts can be represented by means of a triple (T) such as <low, medium, high>, <short, medium, high>, etc. We will term to each component of T as T_l, T_m, and T_r, respectively. Given a triple T, the formal definition of the mentioned functions is the following:

$$L(x) = \begin{cases} 1 \text{ if } x \le \alpha \\ 1 - 2\left(\dfrac{x-\alpha}{\gamma-\alpha}\right)^2 \text{ if } \alpha \le x \le \beta \\ 2\left(\dfrac{x-\gamma}{\gamma-\alpha}\right)^2 \text{ if } \beta \le x \le \gamma \\ 0 \text{ if } x \ge \gamma \end{cases} \tag{1}$$

$$T_m(x) = \begin{cases} 0 \text{ if } x \le \alpha \\ 2\left(\dfrac{x-\alpha}{\beta-\alpha}\right)^2 \text{ if } \alpha \le x \le \dfrac{\alpha+\beta}{2} \\ 1 - 2\left(\dfrac{x-\beta}{\beta-\alpha}\right)^2 \text{ if } \dfrac{\alpha+\beta}{2} \le x \le \beta \\ 1 - 2\left(\dfrac{x-\beta}{\gamma-\beta}\right)^2 \text{ if } \beta \le x \le \dfrac{\beta+\gamma}{2} \\ 2\left(\dfrac{x-\gamma}{\gamma-\beta}\right)^2 \text{ if } \dfrac{\beta+\gamma}{2} \le x \le \gamma \\ 0 \text{ if } x \ge \gamma \end{cases} \tag{2}$$

$$R(x) = \begin{cases} 0 \text{ if } x \le \alpha \\ 2\left(\dfrac{x-\alpha}{\gamma-\alpha}\right)^2 \text{ if } \alpha \le x \le \beta \\ 1 - 2\left(\dfrac{x-\gamma}{\gamma-\alpha}\right)^2 \text{ if } \beta \le x \le \gamma \\ 1 \text{ if } x \ge \gamma \end{cases} \quad \text{Where } \square = (\square + \square)/2; \tag{3}$$

For example, the fuzzy set *dense_pineland* can be represented by means of the functions $T_l(x)$, $T_m(X)$ and $T_r(x)$ where $\alpha = 70$ and $\gamma = 90$.

Usually, the fuzzy terms to be modeled are adjectives, which can be preceded by fuzzy modifiers such as "very", "not", etc. This problem is addressed in the following example. Let us consider the sentence "the vegetation density is high" where the term "high" is represented by a continuous fuzzy function. Suppose now that the modifier "very" transforms "high" into another continuous fuzzy function so that sentences such as "the vegetation density is *very* high" are allowed by the system where the expression "the vegetation density" can take values from a fuzzy term denoted as "high". In this situation, the so-called TFM function (Kandel, 1990) has demonstrated to be useful. This function works as follows:

Let *D* be a fuzzy set, and let *m* be a truth modifier. Then, the result of modifying D by means of m, written *TFM(D/m)*, is a fuzzy set, written *mD*, such as $\mu_{mD}(x) = \mu_m(D(x))$ where the $\mu_S(V)$ function stands for the membership degree of the variable V with respect to the set S.

A practical example of the use of that function is the one pointed out in Martínez-Béjar *et al.* (1997b). In this case, we have modelled the vague predicates containing modifiers by using the next TFM function:

$\mu_{mD}(x) = \mu_{m[}\ D(x)]\ \delta$

where

$0 < \delta < 1$ if the modifier m makes D to hold in a weak way. Examples are *more or less*, *almost*, *little* and *few*.
$\delta > 1$ if the modifier m makes D to hold clearly. Examples are *lot, very*, *much* and *extremely*.

3.1. Fuzzy Modifiers:

Fuzzy attributes may be preceded by fuzzy modifiers, to which the TFM function (Kandel, 1990) can be applied. However, there are many different linguistic modifiers that could potentially be used by the experts when they create or maintain the system. In this sense, fuzzy modifiers used by experts independently of the application domain can be considered. Moreover, they can be grouped into several sets attending to their modification intensity. For instance, a possible fuzzy modifiers classification, which will be considered further in this work, is the following:

a) "Positive" sets of modifiers
RADICAL_POSITIVES = P1 = {"completely", "extremely", "radically", "absolutely", "totally",..};
MODERATE_POSITIVES = P2 = {"very", "much", "lot", "quite",..};

b)"Negative" sets of modifiers
RADICAL_NEGATIVES = N1 = {"almost nothing", "hardly",.. };
MODERATE_NEGATIVES = N2 = {"little", "bit", "few", "more or less", "almost",..};

c)Set of "is-not" forms
IS_NOT = {"not", "not at all", "nothing" "completely not", "practically not", "absolutely not", "practically nothing",..}.

Once the possible modifiers have been classified into sets, experts could see all these sets. It can be noticed that the modifiers belonging to N_i (P_i) are more intense that those belonging to N_j (P_j) if i < j; i, j = 1, 2. By taking this into account and by employing the TFM function, if it is assumed that fuzzy modifiers act on any of the elements of the triple $< T_l(x), T_m(x), T_r(x)>$, the following modeling criterion has been adopted.

Let $\mu_{mT}{}^{i}(x)$ be the membership degree of the variable x with respect to the set mT_i, where $m \in P \cup N \cup IS\_NOT$, $P = \bigcup_{j=1}^{n_p} P_j$, $N = \bigcup_{j=1}^{n_n} N_j$; $i \in \{l, m, r\}$; and n_p and n_n stand for the number of positive and negative non-empty sets of modifiers, respectively. Then, $m_{mT}{}^{i}(x)$ can be defined as follows:

$$(5)\ m_{mT}{}^{i}(x) = \begin{cases} (T_i(x))^{\delta}, \delta > 0, \text{if } m \notin IS\_NOT \\ 1 - T_i(x) \text{ otherwise} \end{cases}$$

In order to give an appropriate value to the parameter d, it must be taken into account that all the elements included in the negative sets of modifiers affect $T_i(x)$ in a different manner as those included in the positive ones. In particular, and by considering the equations defining $T_i(x)$, if the modifier belongs to any of the negative sets of modifiers, the values for d should imply a greater membership function than $T_i(x)$. Similarly, if the modifier belongs to any of the positive sets of modifiers, the values for δ should imply a less membership function than $T_i(x)$.

With all, d can be defined as follows: (6) $\delta = \begin{cases} \delta_p(i) \text{ if } m \in P_i \\ \delta_n(i) \text{ if } m \in N_i \end{cases}$

Where: (7) $\delta_p(i) = (2 + n_p - i)$, $i \in \{1,.., n_p\}$; (8) $\delta_n(i) = \dfrac{1}{2 + n_n - i}$, $i \in \{1,.., n_n\}$,

For example, if we consider the above sets of modifiers, (6) can be applied with $n_p = n_n = 2$ and, then, d can be defined as follows:

(7.1) $\delta_p(i) = (5 - i)$, $i \in \{1, 2\}$; (7.2) $\delta_n(i) = \dfrac{1}{5 - i}$, $i \in \{1, 2\}$

3.2. Fuzzy Frequency Quantifiers:

There are linguistic fuzzy quantifiers referred to as frequency quantifiers, like *always* or *seldom*. Hence, they have not been contemplated in any of the modifiers sets above defined. Moreover, fuzzy modifiers can be combined with the referred fuzzy modifiers in linguistic expressions. For example "A is *seldom* very high". Therefore, it is an interesting endeavor to attempt to model how these quantifiers act on a fuzzy attribute, which in turn, can be preceded by a fuzzy modifier. In this sense, and by reasoning as before for fuzzy modifiers, various sets integrating linguistic quantifiers according to their frequency intensity can be formed. A classification of such quantifiers is, for instance, the following:

HIGH_FREQUENCIES = H= {"almost always", "very often", "often",..}= $\{h_1, h_2, h_3,..\}$;
LOW_ FREQUENCIES = L = {"almost never", "very seldom", "seldom",..}= $\{l_1, l_2, l_3,..\}$;
FACTS = F = {"always"};
IMPOSIBLE_FACTS = I = {"never"}.

Notice that H and L can be disposed in such a way that they can become ordered sets. In the sense that h_i produces more frequency that h_j if $i < j$, while l_i produces less frequency that l_j if $i < j$; $i, j = 1,..,3$.

4. The Proposed RDR based Fuzzy Expert System (FROCH):

By assuming that the membership functions for fuzzy attributes respond to the model of the triple $<T_l(x), T_m(x), T_r(x)>$ as indicated earlier, and that experts can effectively be aware about the fact that they can represent their knowledge in this way, these attributes can be defined as a function of two different parameters, namely, a and g (obviously, in addition to the linguistic terms that univocally identify that attribute). We might also provide the experts with the possibility of defining these numbers with confidence intervals, as it frequently happens in real life. Moreover, the possible presence of modifiers or frequency quantifiers in their jargon can also be considered. By considering all these, FROCH_based system can be defined as follow:

Let R be a ROCH-based system. R is said to be a *fuzzy ripple down rules-oriented conceptual hierarchies (FROCH)*, if the rules can be applied to fuzzy domains by modeling fuzzy attributes, fuzzy modifiers and fuzzy quantifiers as it has been established in this Section, so that every valued attribute, written VA, is represented as follows:

In FROCH-based systems, the KA process will consist of performing the following steps:

(S1) The expert builds an input case through the FROCH.

(S2) The system runs the input case, that is, this is confronted to the rules present in the KB.

(S3) The system shows the conclusions, if any, obtained(acquired) by the system for the input case in such a way that all rule traces are stored.

(S4) If the expert disagrees with some of these conclusions or found there is a missing conclusion (by assuming that no conclusion is the default) The system retrieves information regarding the cornerstone case associated to each rule that provided a wrong conclusion together with information about the input case.

Then, the knowledge acquisition process will proceed as follows:
(S4.1) The system acquires the correct conclusions from the expert.
(S4.2) The system decides on the location of the rules to be entered by the expert.
(S4.3) The system acquires new rules from the expert and added them in order to correct the KB.

There are several issues which have to be considering in implementing FROCH such as:
- Constructing input cases in FROCH, considering some conditions could be fuzzy
- Comparing the Input case with existing rules.
- Generalizing conclusions.

Exploring all these issues completely is beyond the scope of this paper. However, brief explanations about them are provided in the following sections.

4.1. Constructing Cases:

In addition to the possibilities that ROCHs offer to construct cases from a conceptual hierarchy, in FROCHs, the expert must have a way to account for the fuzzy. Moreover, the experts can also have the necessity of considering fuzzy modifiers or fuzzy frequency quantifiers. One possible policy is that the expert could define a fuzzy attribute, through the parameters and elements explained before. This would guarantee that each input case would use the correct (exact) definition, according to the experts' criterion, for fuzzy definitions of attributes. However, it can be noticed that, if we take into account the FROCH-based definition, a fuzzy valued attribute involves, in turn, the definition of two parameters (i. e., a and g). Furthermore, the expert could also use confidence intervals for a or g, or one fuzzy frequency modifier or one fuzzy modifier. Now, if the same fuzzy attribute is supposed to take part of several input cases, what is normal in real world, the quoted policy would lead to a very inefficient system.

On the other hand, it is reasonable to think that, in the context of a maintenance session, the definitions provided by the expert for the same attribute (associated to the same concept in the FROCH) will be kept across different input cases. This assumption has been crucial for the criterion adopted here, which is exposed in the following lines.

It is proposed that, whenever the expert wishes to construct a fuzzy valued attribute not used yet, the system must prompt the expert to enter a triple <T_l, T_m, T_r>, whose elements are words different one another and corresponding to the three possible fuzzy values, respectively. Moreover, the expert must include the value, which is intended to have the fuzzy attribute under question. Furthermore, two real numbers, corresponding to the points where the membership functions T_l and T_r, respectively, reach the value 1 (that is, a and g), must be provided by the expert. However, if the fuzzy attribute corresponding to the same involved concept was already used to construct a case in the current maintenance session, the expert just needs to select amongst T_l, T_m and T_r, the fuzzy value for the current input case.

4.2. Running Cases:

To compare an input case and a rule belonging to the KB, some explicit policies should be adopted. These policies are about how to match conditions that have fuzzy valued attributes defined earlier. It is also clear that we can not just use some simple string comparison technique to compare two fuzzy conditions. Each expert might has his or her particular definition of certain fuzzy attribute. Furthermore, depending on the (input) case, the same expert could use different definitions for the same fuzzy valued attribute. He or she could also define the underlying attribute as a fuzzy valued or as a crisp valued indistinctly. So, the pattern recognition solution can lead to a very inefficient, incomprehensible system, since many definitions with minimal differences among them might be given for the same fuzzy valued attribute.

For example, suppose the KB contains a rule(R), defined as "if vegetation.density(1,80) is seldom very high then C". Also, assume the expert who is in charge of maintaining the system(E), is different from the one who entered the cornerstone case containing the condition expressed in R(E). In this case, by assuming that E wishes to use a fuzzy format for the attribute under question, he or she must define *vegetation.density* in a hypothetical input case exactly as E' did in R in order to allow an eventual matching between R and the input case.

Therefore, the same attribute for a given concept can be defined as fuzzy or as crisp (depending on the input case) in FROCH-based systems. This has been taken into account for defining a criterion on which the matching process will be based. In particular, we will first define the matching criterion involving two fuzzy valued attributes corresponding to one of the conditions of the input case and to one of the rules of the KB, respectively. Then, by applying some results obtained in the definition of this criterion, we will deal with the problem of having a crisp valued attribute against a fuzzy valued one in the case running process.

•Fuzzy Valued Attributes Versus Fuzzy Valued Attributes:

Attending to the FROCH-based system definition, some compatibility criterion among two whatever definitions of a fuzzy valued attribute could be defined. More precisely, this criterion can be established by taking into account both the information about the parameters defining the underlying membership functions(i.e., a , g and their respective confidence intervals) and the elements of the attribute definition(i.e, the quantifier, the modifier, and the fuzzy value) involving the two fuzzy valued attributes under question. Therefore, the semantic distance among the elements or parameters implied in a comparison has been considered as a basic factor.

Formally, the referred criterion can be stated in a step-by-step process as it is explicated in the following definitions.

First, we can group the values associated to each membership function integrating the triple referenced before into semantic compatibility classes. It can be done by taking into account, for example, the linguistic tags associated to $T_l(x)$ and $T_r(x)$, respectively, should be different. They should not overlap from the semantic point of view. Formally, the referred classification can be established as definition 1-12 and principle 1-4 in appendixes.

•Fuzzy Valued Attributes Versus Crisp Valued Attributes:

When a case is entered to a FROCH-based system, several situations can be distinguished. This depends on the nature of the attributes implied in the matching process, namely, either fuzzy or crisp. Thus, the crisp valued attributes can respond to various formats. Here, we will consider the character string-based one and the numeric interval-based one. All these possibilities are analyzed in the following sections.

•Fuzzy Valued Attributes Versus Character String-like Valued Attributes:

A representative example of this situation can be "*vegetation.height(5,40) is high* versus *vegetation.height = medium*". By assuming that the same pair (concept, attribute) is referenced in both conditions. The criterion to be followed will consist of checking whether the values (and, eventually, the modifiers and the quantifiers) are compatible or not. Moreover, if they found to be compatible and the rule involved in the comparison process possesses siblings in the MCRDR structure, the crisp valued attribute under question will be replaced by the fuzzy valued attribute in further comparisons for the current input case. In this way, given an input case, the matching process involving sibling rules will take place in a context of a more precise knowledge.

Formally, the compatibility criterion can be stated as follows

Definition: Compatible Non-numerical Crisp Valued Attributes:

Let FVA $_1$ and CSVA $_2$ be a fuzzy valued attribute and a character string-like valued attribute in a FROCH-based system and represented, respectively, as follows :

FVA $_1$ = "concept$_1$.attribute$_1$(p $_1$) is [$q_{1]}$ [$m_{1]}$ v_1"; CSVA $_2$ = "concept$_2$.attribute$_2$ = [$q_{2]}$ [$m_{2]}$ v_2"
Where p_1 = (a $_1$[Ia $_{1]}$, g $_1$[Ig $_{1]}$)
FVA $_1$ and CSVA $_2$ are said to be s-compatible, written *s-compatible(FVA $_1$, CSVA $_2$),* if (concept$_1$.attribute$_1$ = concept$_2$.attribute$_2$) and (compatible_values(v_1, v_2)) and (compatible_modifiers(m_1, m_2)) and (compatible_quantifiers(q_1, q_2)) holds.

•Fuzzy Valued Attributes Versus Numeric Interval-like Valued Attributes:

A typical example of this case is "*height(5,40) is high* versus *height $\in$ [5,12]* ". By assuming that the same pair (concept, attribute) is referenced in both conditions, the system will check whether the interval underlying the crisp valued attribute under question is included in the definition of the fuzzy attribute. If so, the system will determine whether the linguistic tag referenced in the fuzzy attribute corresponds to the interval into which the interval associated to the crisp attribute can be allocated. Finally, the modifiers and the frequency quantifiers, if any, will be confronted to check compatibility. This entire process can be expressed in a formal manner in a step-by-step process as definition 14-16 in appendixes.

4.3. Acquiring Conclusions:

To group the conclusions into compatibility classes, we can make use of the possibilities that fuzzy logic offers to help the experts to decide which conclusion concerning a certain feature should be considered. We can exploit the fact that fuzzy membership functions can be propagated from antecedents to consequents in the rules.

To clarify how to proceed with the propagation of membership function values from the antecedent to the consequent part of a fired rule. There are several methods**,** including the algebraic product-sum, the Hamacher product-sum, the Einstein product-sum, and the Bounded difference-sum. In FROCH, the algebraic product has been chosen. This method provides a good alternative and is computationally parsimonious. Through this method, the membership function value of the consequent of a rule is calculated as the product of the membership function values corresponding to the each of the sub conditions forming the antecedent of the rule under question. Therefore, the problem will be reduced to how to calculate the membership function value for each sub condition of the antecedent.

To calculate the membership function value (MFV) associated to a sub condition, it must be considered the fact that rules as well as input cases can include crisp or fuzzy valued attributes. If a rule condition is crisp, an MFV equals to 1 will be assumed. Otherwise, the MFV will be dependent of the nature of the input case condition that matches the rule condition under question. The definitions, lemmas, principals and the algorithms used in this section can be found in the appendixes.

Conclusions:

Ripple down rules(RDR) is a knowledge acquisition technique whose aim is to use the knowledge only in the context provided by the expert. This context is the sequence of rules evaluated to give a certain conclusion. With this approach, rules are never removed or corrected, only added. This addition only occurs when the expert does not agree with a conclusion supplied by the system and, then, he or she wishes to correct such a conclusion.

Fuzzy terms are widely used in real-life problems to which current RDR-based systems may be applied. Although fuzzy logic is normally used to deal with fuzzy domains, those systems only allow to assign crisp values to fuzzy-by-nature attributes. Moreover, current developments of RDR do not allow experts to use fuzzy terminology in a natural, compact manner. The knowledge that can be acquired and represented in such systems is, hence, restrictive and non-natural from the experts' point of view. In this paper, a formal approach that can be used to acquire and represent fuzzy domain knowledge through the most recent versions of RDR-based systems has been presented.

We have defined a model that allows experts to use fuzzy modifiers as well as fuzzy frequency quantifiers when they construct cases for a further processing by the system. Moreover, based on some well-defined assumptions about the kind of fuzzy terms that can be represented in our model, a new methodology for running cases, which can include both fuzzy and crisp values, has been proposed.

The approach presented is concerned with the acquisition/representation of fuzzy domain knowledge in RDR-based systems. Therefore, the formal model we have exposed can be viewed as an extension of current RDR-based systems to deal with fuzzy domains in a direct, natural way. In this sense, our approach is complementary to work on MCRDR systems and the ROCH approach, as these provide methods to apply RDR-based systems to other more complex tasks and to construct RDR ontological frameworks, respectively.

REFERENCES

Compton, P. and R. Jansen, 1989. Knowledge in context: a strategy for expert system maintenance, In C. J. Barter and M. J. Brooks (Eds.), *AI'88,* 292-306, Berlin, Springer-Verlag.

Compton, P. and R. Jansen, 1990. A philosophical basis for knowledge acquisition, *Knowledge Acquisition*, 2: 241-257.

Dazeley, R. and B.H. Kang, 2003. Weighted MCRDR: Deriving Information about Relationships between Classifications in MCRDR, The 16th Australian Joint Conference on Artificial Intelligence. 2003. Perth, Australia: Springer-Verlag, Berlin Heidelberg New York.

Martínez-Béjar, R., V.R. Benjamins, P. Compton, P. Preston and F. Martín-Rubio, 1997b. A formal framework to build domain knowledge ontology for ripple down rules-based systems, In Press.

Vivancos Vicente, P.J., R. Valencia-Garcia, J.T. Fernandez Breis, R. Martinez Bejar, 2004. An approach for multi relational ontology modeling. Lecture Notes in Artificial Intelligence: Proceedings of 8th Pacific Rim International Conference on Artificial Intelligence, 3157: 997-998.

Seker, H., M. Odetayo, D. Petrovic and R.N.G. Naguib, 2003. "A Fuzzy Logic Based Method for Prognostic Decision Making in Breast and Prostate Cancers", *IEEE Trans. On* Information Technology in Biomedicine

Ismail SARITAS, Novruz ALLAHVERDI and Ibrahim Unal SERT, 2003. A Fuzzy Expert System Design for Diagnosis of Prostate Cancer, International Conference on Computer Systems and Technologies, Sofia, Bulgaria.

Valencia-García, R., J.M. Ruiz-Sánchez, P.J. Vivancos-Vicente, J.T. Fernández-Breis, R. Martínez-Béjar, 2004. An incremental approach for discovering medical knowledge from texts. Expert Systems With Applications, 26(3): 291-299.

William Siler, James J. Buckley, 2005. Fuzzy Expert Systems and Fuzzy Reasoning, ISBN: 978-0-471-38859-3.

Edwards, G., P. Compton, R. Malor, A. Srinivasan and Lazarus, 1993. PEIRS: a pathologist maintained expert system for the interpretation of chemical pathology reports, Pathology, 25: 27-34.

Phayung Meesad, 2001. Quantitative Measures of a Fuzzy Expert System, IEEE NNC Studen Summer Research.

Roubos, H. and M. Setnes, 2001. "Compact and transparent fuzzy models and classifiers through iterative complexity reduction," IEEE Trans. Fuzzy Syst., pp: 9-4.

Zadeh, L.A., 1965. Fuzzy sets, Information and Control, 8: 338-353.

Hwang, G-J., 1995. Knowledge acquisition for fuzzy expert systems, International Journal of Intelligent Systems, 10: 541-560.

Dubois, D. and H. Prade, 1988. Possibility Theory, an Approach to Computerised Processing of Uncertainty, New York, Plenum Press.

Buckley, J.J., W. Siler and D. Tucker, 1986. A fuzzy expert system, Fuzzy Sets and Systems, 20: 1-16.

Seker, H., M. Odetayo, D. Petrovic and R.N.G. Naguib, 2003. "A Fuzzy Logic Based Method for Prognostic Decision Making in Breast and Prostate Cancers", IEEE Trans. On Information Technology in Biomedicine.

Yen, G. and P. Meesad, 2000. "Constructing a fuzzy expert system using the ILFN network and the genetic algorithm," in Proc. SMC'00, pp: 3.

Kandel, A., 1990. Fuzzy mathematical techniques with applications, Reading, MA, Addison-Wesley.

Appendix:

Definition 2: compatible/incompatible fuzzy values:

Let < T_1, T_2, T_3> be a triple containing three different linguistic tags associated to the elements of the triple < $T_l(x)$, $T_m(x)$, $T_r(x)$> and defined according to (1), (2) and (3), respectively; and let V_1 and V_2 be two elements belonging to the union set $\bigcup_{i=1}^{3} T_i$. V_1 and V_2 are said to be two *compatible fuzzy values*, written *compatible_values* (V_1, V_2), if the following holds:

$$\bigvee_{k=1}^{2} \left(V_{12} \subseteq (\{T_k\} \cup \{T_{k+1}\})\right) \quad \text{where } V_{12} = \{V_1\} \cup \{V_2\}.$$

Otherwise, that is, if the above condition does not hold, p_1 and p_2 are said to be two *incompatible fuzzy values*, written *incompatible_values* (V_1, V_2).

Definition 3: left/right -side membership functions:

Let (α_f, γ_f) be the parameters corresponding to the definition of a fuzzy attribute, written FVA, in a FROCH-based system. Consider also the triple < $T_l(x)$, $T_m(x)$, $T_r(x)$>, defined according to (1), (2) and (3), respectively. The *left-side membership functions associated to* (α_f, γ_f), written *left_side_mf* (α_f, γ_f), is defined as the set {$T_l(x)$, $T_m(x)$}in the interval [α_f, β_f].

Similarly, the *right-side membership functions associated to* (α_f, γ_f), written *right_side_mf* (α_f, γ_f), is defined as the set {$T_m(x)$, $T_r(x)$}in the interval [β_f, γ_f].

Definition 4: maximum parameter-based compatibility distance:

Let (α_f, γ_f) be the parameters corresponding to the definition of a fuzzy attribute, written FA, in a FROCH-based system; let L = (l_1, l_2) be the set of linguistic tags associated to those of left-side_mf (α_f, γ_f); and let R = (r_1, r_2) be the set of linguistic tags associated to those of right-side_mf (α_f, γ_f). The maximum parameter-based compatibility distance for FA, written *MPCD*, is defined as a real number such that [LEFT_COMPATIBILITY(α_f, γ_f, MPCD) and RIGHT_COMPATIBILITY (α_f, γ_f MPCD)] holds where
LEFT_COMPATIBILITY (α_f, γ_f MPCD) = [$\forall$ x $\in$ [β_f - MPCD, β_f] $\exists$ l_i $\in$ L such that $\mu_1(x) \geq \mu_r(x)$ where (r $\in$ R) and (r $\in$ L);
RIGHT_COMPATIBILITY(α_f, γ_f, MPCD) = [$\forall$ x $\in$ [β_f, β_f + MPCD] $\exists r_i$ $\in$ R such that (x) μ_r $\mu_l(x)$ where (l $\in$ L) and (l $\notin$ R); $\beta_f = \left(\frac{\alpha_f + \gamma_f}{2}\right)$

Lemma:

Let (α_f, γ_f) be the parameters corresponding to the definition of a fuzzy attribute in a FROCH-based system. Then, the following equality holds: MPCD = $\frac{\gamma_f - \alpha_f}{2}$

By applying the above lemma to the characteristics of the fuzzy memberships functions underlying a FROCH-based system, the following definitions can be written.

Definition 5: parameter-based compatibility range:

Let p be a parameter defining a fuzzy membership function associated to a fuzzy valued attribute, written FVA, in a FROCH-based system; and let MPCD be the maximum parameter-based compatibility distance for FVA. The *parameter-based compatibility range for p*, written *PCR(p)*, is defined as the interval [p - MPCD, p + MPCD].

Definition 6: parameter-based compatibility:

Let p, p' be belonging to the set {α_f, β_f , γ_f}, and whose elements are the parameters that define the fuzzy memberships functions $T_l(x)$, $T_m(x)$ and $T_r(x)$, respectively for certain fuzzy attribute in a FROCH-based system; and let PCR(p) the parameter-based compatibility range for p. p' is said to be compatible with respect to p, written *compatible_parameters (p, p')*, if p' $\in$ PCR(p).

It can be noticed that, attending to the previous corollaries, one could argue that if one wants to know the parameters, which are compatible with respect to b_f, some redundancy would be produced as there are two statements asserting this compatibility. To avoid this kind of notation redundancies, we can write the compatibility respect to b_f by using a new notation.

Definition 7: relative parameter-based compatibility:

Let p, p_i , 1 $\leq$ i $\leq$ n, be two parameters defining a fuzzy membership function for a fuzzy valued attribute in a FROCH-based system such that the following holds: compatible_parameters(p, p_1) and compatible_parameters(p, p_2) and...and compatible_parameters(p, p_n). The parameter-based compatibility relative to p, written *relative_compatibility(p)*, is defined as the set { p_1 , p_2,.., p_n}.

So far, we have formally established the definitions and use of compatibility classes into a same fuzzy attribute according to its definition parameters (that is, α and γ). However, the objective pursued in this subsection is more ambitious in that we attempt to establish a compatibility criterion between two fuzzy attributes in a FROCH-based system. To reach this goal, we can establish some principles, which will constitute the foundation for a further definition of the referred criterion.

Firstly, we can extend the maximum parameter-based compatibility distance concept to the case of two (in general) different fuzzy attributes. For it, we will take into account the following principle.

Principle 1: maximum distance:

Let f_1 and f_2 be two fuzzy valued attributes belonging to a FROCH-based system, and defined, respectively, as "concept.attribute(α_i, γ_i)". A necessary condition for f_1 and f_2 to be semantically compatible is that p_j $\in$ PCR(p_k), where p_j $\in$ {α_j, β_j, γ_j); p_k $\in$ {α_k, β_k, γ_k}; $\beta_i = \left(\frac{\alpha_i + \gamma_i}{2}\right)$; i, j, k = 1, 2.

Secondly, by considering what definition 2 states, we can restrict the parameters that could potentially be semantically inter-compatible to those whose values belong to the same class.

Principle 2: non-incompatible values:

Let f_1 and f_2 be two fuzzy valued attributes belonging to a FROCH-based system, and defined, respectively, as "concept.attribute (α $_i$, γ $_i$) is value$_i$", i = 1, 2. A necessary condition for f_1 and f_2 to be semantically compatible is that compatible_values (value$_1$, value$_2$).

By analysing the two compatibility classes defined above, it is easy to note that the two geometrical regions where such classes exist have interesting properties respect to the values that the respective membership functions involved in them take. This fact can be expressed in a formal step-by-step process as follows.

From what has bee considered so far, someone could argue that a criterion about the semantic compatibility between two fuzzy valued attributes, where the difference is only in the parameters or in the values, can be derived from the logic conjunction of the two principles indicated earlier. This idea can be founded on the fact that all the so-obtained results can be applied to many situations. Consider, for example, the distributions ((a) - (e)) shown in Figure 1.

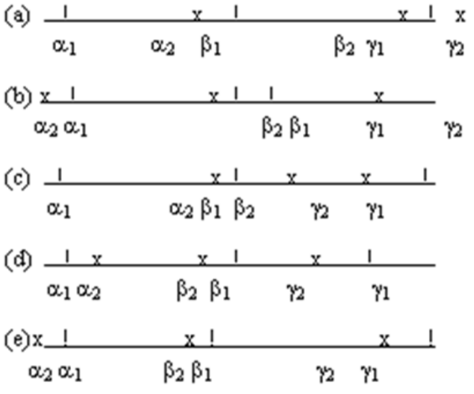

Fig. 1: Example of parameters distributions

It can be noted that all the cases (a) - (e) in Figure 2 hold the two principles mentioned previously. Moreover, by applying Definition 6 to each of the above cases, the following is obtained:

(a) relative_comp(a $_1$) = {a $_2$, b $_1$}; relative_comp(g $_1$) = {b $_1$, b $_2$, g $_2$};

(b) relative_comp(a $_1$) = {b $_1$, a $_2$, b $_2$}; relative_comp(g $_1$) = {b $_1$, g $_2$};

(c) relative_comp(a $_1$) = {b $_1$, a $_2$}; relative_comp(g $_1$) = {b $_1$, b $_2$, g $_2$};

(d) relative_comp(a $_1$) = {b $_1$, a $_2$, b $_2$}; relative_comp(g $_1$) = {b $_1$, g $_2$};

(e) relative_comp(a $_1$) = {b $_1$, a $_2$, b $_2$}; relative_comp(g $_1$) = {b $_1$, g $_2$}.

However, problems still remain when one of those fuzzy valued attributes is composed by parameters very close each other and, in turn, all of them being very far from one of the pairs (a , g). Figure 2 illustrates this phenomenon. Besides, when this occurs, although the two necessary conditions underlying Principle 1 and Principle 2, respectively, hold, it seems not to be reasonable to consider the possibility of the two fuzzy valued attributes under question to be compatible.

Figure 2. Accumulation of parameters.

Formally, this situation can be expressed in the following principle:

Principle 3: maximum distance:

Let f_1 and f_2 be two fuzzy valued attributes belonging to a FROCH-based system, and defined, respectively, as "concept.attribute (α $_i$, γ $_i$) is value$_i$"; and let MPCD$_i$ be the maximum parametric compatibility distance for f_i, i = 1, 2. A necessary condition for f_1 and f_2 to be semantically compatible is that the following holds:

{[(g $_j$ £ g $_k$) and (a $_j$ ³ a $_k$)] ® (MPCD$_j$ ³ K $_j$* MPCD$_k$)}, j¹ k = 1,2

where

K $_j$ = min{β $_j$ - MAXa $_j$, MINg $_j$ - β$_j$};

$$\mathrm{MAXa}_{j} = \begin{cases}(1+I_{\alpha_1})\alpha_j \text{ if } I_{\alpha_1} \text{ is specified}\\ 0.5 \qquad\qquad \text{otherwise}\end{cases} \qquad \mathrm{MINg}_{j} = \begin{cases}(1-I_{\gamma_1})\gamma_j \text{ if } I_{\gamma_1} \text{ is specified}\\ 0.5 \qquad\qquad \text{otherwise}\end{cases}$$

Based on the three principles exposed before, the following definition can be derived:

Definition 8: compatible/ incompatible definition parameters:

Let f_1 and f_2 be two fuzzy valued attributes belonging to a FROCH-based system, and defined, respectively, as "concept.attribute(α_{k}[I $\alpha_{k]}$, $\gamma_{k[}$ I$\gamma_{k]}$) is value$_k$"; let p_k be (α_{k}, γ_{k}); let K_j be in accordance with Principle 3; and let $MPCD_k$ be the maximum parametric compatibility distance for f_k, k = 1, 2. p_1 and p_2 are said to be *compatible definition parameters*, written *compatible_parameters (p_1, p_2)* if (compatible_values(value$_1$, value$_2$)) and (within_compatible_ranges(p_i, p_j)) and (sufficient_distance(p_i, p_j)) holds, where

$$\text{within\_compatible\_ranges}(p_i,\ p_j) = \bigvee_{i\neq j=1}^{2}\left[(\alpha_i \in \mathrm{PCR}(\alpha_j)) \text{ and } (\gamma_i \in \mathrm{PCR}(\gamma_j))\right] \quad ;$$

$$\text{sufficient\_distance}(p_i,\ p_j) = \bigvee_{i\neq j=1}^{2}\left[(\alpha_j \geq \alpha_i) \text{ and } (\gamma_j \geq \gamma_i) \rightarrow (\mathrm{MPCD}_j \geq K_j * \mathrm{MPCD}_i)\right]$$

Now, if we go one step beyond, in the sense of considering also fuzzy frequency quantifiers, another designing principle, operating on the frequency sets pointed out in Section 3, can be established.

Principle 4**: **frequency compatibility:

A fuzzy frequency quantifier belonging to the I È L must not be considered as semantically compatible to another one not belonging to such a union set.

The above principle can be used to formally define the compatibility it alludes to as follows:

Definition 9: compatible/ incompatible fuzzy frequency quantifiers:

Let q_1 and q_2 be two fuzzy frequency quantifiers such that q_k belongs to the union set $H \cup L \cup F \cup I \cup UN_q$, k = 1, 2; where H, L, F and I respond to the definition given in (9), while UN_q stands for the fact that some quantifier is unspecified. So, UN_q can be expressed as the set {"unspecified_quantifier"}. Moreover, we will assume that an unspecified quantifier is a synonym concept of "always". Under these conditions, q_1 and q_2 are said to be two *compatible fuzzy frequency quantifiers*, written *compatible_quantifiers(q_1, q_2),* if the following logic condition holds:

[($q_{12} \subseteq$ INFERIOR_FREQS) or ($q_{12} \subseteq$ SUPERIOR_FREQS)]

where

$q_{12} = \{q_1\} \cup \{q_2\}$; INFERIOR_FREQS = ($I \cup L$); SUPERIOR_FREQS = ($F \cup H \cup UN_q$).

Otherwise, that is, if the above logic equation does not hold, q_1 and q_2 are said to be two *incompatible fuzzy frequency quantifiers*, written *incompatible_quantifiers(q_1, q_2).*

The anterior principle can also be used when fuzzy modifiers are present in a fuzzy valued attribute. The nuance to be considered in this case is that a modifier can belong to several compatibility classes

Formally, the compatibility concerning fuzzy modifiers has been established by means of the following definition.

Definition 10: compatible/incompatible fuzzy modifiers:

Let m_1 and m_2 be two fuzzy modifiers such that q_k belongs to the union set $P \cup N \cup$ IS_NOT $\cup\ UN_m$, k = 1, 2; where P, N, and IS_NOT respond to the definition given in (5), while UN_m accounts for the fact that some modifier is unspecified. So, UN_q can be expressed as the set {"unspecified_modifier"}. Moreover, we will assume that the presence of an unspecified modifier means that the parameter d takes the value 1. Under these conditions, m_1 and m_2 are said to be two *compatible modifiers*, written *compatible_modifiers(m_1, m_2),* if the logic condition $\left(\bigvee_{l=1}^{4} M_l\right)$ holds, where

$$M_1 = \bigvee_{i \neq j=1}^{2} \left((m_i \in N_1) \text{ and } (m_j \in (\text{IS\_NOT} \cup N_1 \cup N_2))\right) \; ; \; M_2 = \bigvee_{i \neq j=1}^{2} \left((m_i \in N_2) \text{ and } (m_j \in (N_1 \cup N_2 \cup UN_m))\right) \; ;$$

$$M_3 = \bigvee_{i \neq j=1}^{2} \left((m_i \in UN_m) \text{ and } (m_j \in (N_2 \cup UN_m \cup P_2))\right) \; ; \; M_4 = \bigvee_{i \neq j=1}^{2} \left((m_i \in P_2) \text{ and } (m_j \in (UN_m \cup P_2 \cup P_1))\right) \; ;$$

Otherwise, that is, if the above logic equation does not hold, q_1 and q_2 are said to be two *incompatible fuzzy modifiers*, written *conflicitve_modifiers(q_1, q_2).*

By considering the previous definitions, a compatibility/conflictivity criterion to be applied to eventually fuzzy valued attributes in the context of a FROCH-based system can be derived as follows.

Definition 11: compatible fuzzy valued attributes:

Let FVA_1 and FVA_2 be two fuzzy valued attributes represented, according to definition 1, as "$c_i.a_i(p_i)$ is $[q_{i]}\ _{[}\ m_{i]}\ v_i$" respectively, where p_i = ($\alpha\ _i[I\ \alpha\ _{i]}$, $\gamma\ _i[I\gamma\ _{i]}$), i = 1, 2. FVA_1 and FVA_2 are said to be two *compatible fuzzy valued attributes*, written *compatible_val_atts (FVA_1, FVA_2)*, if [($EQUAL_{12}$) or ((EQUAL_ca (FVA_1, FVA_2) and (COMPATIBLE_vpmq (FVA_1, FVA_2))] holds, where
$EQUAL_{12}$ = (FVA_1 = FVA_2);
EQUAL_ca (FVA_1, FVA_2) = ($c_1.a_1$ = $c_2.a_2$); and COMPATIBLE_vpmq (FVA_1, FVA_2) = [(compatible_parameters(p_1, p_2)) and (compatible_modifiers(m_1, m_2)) and (compatible_quantifiers(q_1, q_2))].
Similarly, the following definition can be established:

Definition 12: incompatible fuzzy valued attributes:

Let FVA_1 and FVA_2 be two fuzzy valued attributes represented, according to definition 1, as "$c_i.a_i(p_i)$ is $q_i m_i v_i$" respectively, where p_i = ($\alpha\ _i[I\ \alpha\ _{i]}$, $\gamma\ _i[I\gamma\ _{i]}$), i = 1, 2. FVA_1 and FVA_2 are said to be two *incompatible fuzzy valued attributes*, written *incompatible_val_atts (FVA_1, FVA_2)*, if [EQUAL_ca (FVA_1, FVA_2) and (NOT(COMPATIBLE_vpmq(FVA_1, FVA_2))] holds, where EQUAL_ca (FVA_1, FVA_2) and COMPATIBLE_vpmq (FVA_1, FVA_2) are defined as in previous definition.

Definition 13: compatible non-numerical crisp valued attributes:

Let FVA $_1$ and CSVA $_2$ be a fuzzy valued attribute and a character string-like valued attribute in a FROCH-based system and represented, respectively, as follows:

FVA $_1$ = "$concept_1.attribute_1$(p $_1$) is [$q_{1]}$ [$m_{1]}$ v_1"; CSVA $_2$ = "$concept_2.attribute_2$ = [$q_{2]}$ [$m_{2]}$ v_2"
Where p_1 = ($\alpha\ _1[I\alpha\ _{1]}$, $\gamma\ _1[I\ \gamma\ _{1]}$)

FVA $_1$ and CSVA $_2$ are said to be s-compatible, written *s-compatible(FVA $_1$, CSVA $_2$),* if ($concept_1.attribute_1$ = $concept_2.attribute_2$) and (compatible_values(v_1, v_2)) and (compatible_modifiers(m_1, m_2)) and (compatible_quantifiers(q_1, q_2)) holds.

Definition 14: active interval:

Let (α [I α], γ [I γ]) be a pair containing the parameters and their respective confidence intervals corresponding to the definition of a fuzzy attribute, written FVA, in a FROCH-based system; let < T_1, T_2, T_3> be a triple containing three different linguistic tags associated to the respective elements of the triple < $T_l(x)$, $T_m(x)$, $T_r(x)$>, defined according to (1), (2) and (3), respectively; let C, A be the concept and the attribute, respectively, underlying FVA.; and let v be the value defined in FVA such that v ∈ $\bigcup_{i=1}^{3} T_i$. The *active interval* for C.A in FVA, written *active_interval (C.A, FVA)* is defined as follows:

$$\text{active\_interval (C.A, FVA)} = \begin{cases} \left(-\infty, \dfrac{\text{MAX}\alpha + \text{MAX}\gamma}{2}\right] & \text{if } v = T_1; \\ [\text{MIN}\alpha, \text{MAX}\gamma] & \text{if } v = T_2; \\ [\text{MIN}\gamma, \infty) & \text{if } v = T_3 \end{cases}$$

where

$$\text{MAXa} = \begin{cases}(1+I_\alpha)\text{ if } I_\alpha \text{ is specified}\\ \alpha \quad \text{otherwise}\end{cases} \qquad \text{MAXg} = \begin{cases}(1+I_\gamma)\text{ if } I_\gamma \text{ is specified}\\ \gamma \quad \text{otherwise}\end{cases}$$

$$\text{MINa} = \begin{cases}(1-I_\alpha)\text{ if } I_\alpha \text{ is specified}\\ \alpha \quad \text{otherwise}\end{cases} \qquad \text{MINg} = \begin{cases}(1-I_\gamma)\text{ if } I_\gamma \text{ is specified}\\ \gamma \quad \text{otherwise}\end{cases}$$

Now, the following definition can be established:

Definition 15: compatible numeric crisp valued attribute:

Let FVA $_1$ and NIVA $_2$ be a fuzzy valued attribute and a numeric interval-like valued attribute in a FROCH-based system and represented, respectively, as follows :
FVA $_1$ = "concept$_1$.attribute$_1$(p $_1$) is [q$_{1]}$ [m$_{1]}$ v$_1$";
CSVA $_2$ = "concept$_2$.attribute$_2$ ∈ (v$_{inf}$, v$_{sup}$)"
where

$p_1 = (\alpha\ _1[I\ \alpha\ _{1]}\ ,\ \gamma\ _1[I\gamma\ _{1]}\ )$

FVA $_1$ and NIVA $_2$ are said to be n-compatible, written *n-compatible(FVA $_1$, NIVA $_2$),* if (concept$_1$.attribute$_1$ = concept$_2$.attribute$_2$) and (compatible_values(v$_1$, v$_2$)) and (compatible_modifiers(m$_1$, m$_2$)) and (compatible_quantifiers(q$_1$, q$_2$)) and [(v$_{inf}$, v$_{sup}$) Î active_interval(concept$_1$.attribute$_1$, FVA $_1$)] holds.

By using previous definitions, the condition for a given rule in a FROCH-based system to by fired by an input case, can be established as follows:

Definition 16: fired fuzzy ripple down rule:

Let CR$_i$ be a condition of a rule, written R; let CI$_j$ be a condition for an input case, written I; and let CR$_i$ and CI$_j$ being respectively represented (according to the representation format adopted here for FROCH-based system) as follows:

$$C_k = \begin{cases} c_k.a_k(p_k) \text{ is } [m_k][q_k]v_k \text{ if } a_k \text{ is a fuzzy valued attribute;}\\ c_k.a_k R(v'_1, v'_2, .., v'_{n_k}) \quad \text{otherwise}\end{cases}$$

Where: C$_k$ ∈ {CR$_k$, CI$_k$}; k ∈ {i, j}; p$_k$ = (α $_k$[I α $_{k]}$, γ $_k$[I γ $_{k]}$); n$_k$ = number of values involved in c$_k$.a$_k$
R is said to be fired by I, written *fired (R, I)* if ∀ CR$_i$ ∈ R, ∃ CI$_j$ ∈ I, such that compatible_valued_attributes(CR$_i$, CI$_j$) holds Where:

$$\text{compatible\_valued\_attributes }(CR_i, CI_j)= \begin{cases}\text{compatible\_val\_atts}(CR_i, CI_j) \text{ if both } CR_i \text{ and } CI_j \text{ are fuzzy valued attributes;}\\ \text{s-compatible}(CR_i, CI_j) \text{ if one of the arguments is a fuzzy valued attribute and}\\ \qquad \text{the other is a non-numerical crisp valued one;}\\ \text{n-compatible}(CR_i, CI_j) \text{ if one of the arguments is a fuzzy valued attribute and}\\ \qquad \text{the other is a numerical crisp valued one;}\\ \text{match}(CR_i, CI_j) \text{ if both } CR_i \text{ and } CI_j \text{ are crisp valued attributes;}\end{cases}$$

$$\text{match}(CR_i, CI_j) = \begin{cases} \textit{true} \text{ if } CR_i \text{ and } CI_j \text{ are in accordance with their values;}\\ \textit{false} \text{ otherwise}\end{cases}$$

$$R = \bigcup_{i=1}^{Card(R)} CR_i \quad ; \; I = \bigcup_{j=1}^{Card(I)} CI_j$$

Conclusion Acquisition (CA) algorithm:

1. *A set of conclusions (C) corresponding to the consequents of the rules that match I is generated;*
1. *the system classifies the conclusions in C according to their pair (concept,attribute) referenced in each of them. The sets C$_1$, C$_2$, .., C$_n$ will be obtained;*
1. *for every C$_i$ do { C$_i$ = (concept $_i$.attribute $_i$)*
   L$_i$ = φ {List of ordered conclusions (according with their certainty) associated to C$_i$ } do

if Card(C$_i$) > 1 then

group the conclusions in C_i into **compatibility classes** such that, for each compatibility class, the conclusion having the greatest depth in the MCRDR structure is selected as the "representative" one for the compatibility class under question; the representative conclusions C_1(concept$_i$.attribute$_i$), C_2(concept$_i$.attribute$_i$), ., C_m(concept$_i$.attribute$_i$) will be obtained;
classify the so-obtained conclusions according to the depth of their location in the MCRDR structure in a decreasing order; the sets of representative conclusions C^1(concept$_i$.attribute$_i$),C^2(concept$_i$.attribute$_i$),.., C^s(concept$_i$.attribute$_i$) will be obtained;
for every C^j(concept$_i$.attribute$_i$) do
if Card(C^j(concept$_i$.attribute$_i$)) > 1 then
group the conclusions in C^j(concept$_i$.attribute$_i$) into rule antecedent-based membership function values in a decreasing order; record each value together with the conclusion to which that value is associated;
$L_i = L_i \cup C^j$(concept$_i$.attribute$_i$);
else $L_i = L_i \cup C_i$
where the compatible conclusions are obtained from the following definition:

Definition 17: compatible/incompatible conclusions:

Let C_1 and C_2 be two conclusions involving the valued attributes VA_1 and VA_2, respectively. C_1 and C_2 are said to be two *compatible conclusions*, written *compatible_conclusions (C_1, C_2)*, if compatible_valued_attributes (VA_1, VA_2) holds. In an analogous way, they are said to be two incompatible conclusions, written *incompatible_conclusions (C_1, C_2)*, if not(compatible_valued_attributes(VA_1, VA_2)) holds. Once the conclusions have been grouped, the following algorithm will be carried out in this approach:

Conclusion Show (CS) algorithm:

```
For i=1 to n do {n = number of pairs (concept,attribute)}
  for j=1 to $m_i$ do { $m_i$ = number of elements in $L_i$}
    show $L_i^j$ to the expert;
    if he or she does not agree with $L_i^j$ then
      ask the expert to enter a new conclusion, NC, such that it will only affect the fired rule(s) containing $L_i^j$ or another compatible conclusion(s) situated at the same depth in the MCRDR structure than $L_i^j$;
      if j < $m_i$ then
        for k = j + 1 to $m_i$ do
          if compatible_conclusions(NC, $L_i^k$) then
            $L_i = L_i \setminus \{ L_i^k \}$; $m_i = m_i - 1$
```

Definition 18: F-non-membership:

Let A be a rule condition represented as follows:

$$A = \begin{cases} c.a(\alpha[I_\alpha], \beta[I_\beta]) \text{ is } [m][q]v & \text{if A is a fuzzy valued attribute;} \\ c.a\mathbf{R}(v_1, v_2, .., v_n) & \text{otherwise} \end{cases}$$

and let C be a case in a FROCH-based system to which A is confronted. A is said to be a *non-member of C* in the mentioned context, written *F-non-member (A, C)*, if not (∃ $C_j \in C$ s. t. compatible_valued_attributes (C_j, A)) where C = $\bigcup_{j=1}^{n} C_j$

By taking into account last definition and the classic method to construct difference lists in RDR-based systems, a new definition for difference list in the context of FROCH-based systems can be expressed as follows:

Definition 19: F-difference list:

Let I and C be an input case and a cornerstone case composed respectively by n and m conditions; n, m ³ 1. The *difference list between I and C* in the context of a FROCH-based system, written *F-difference list (I, C)*, is defined as the union set FD_{input} È $FD_{cornerstone}$
Where FD_{input} = {I_i | F-non-member(I_i, C)}; $FD_{cornerstone}$ = {NOT(C_j)| F-non-member (C_j, I)};

I = $\bigcup_{i=1}^{n} I_i$; C = $\bigcup_{j=1}^{m} C_j$