

# A fuzzy expressions processor with modifiers and hedges, conceptual design and testing

Jose Antonio Martin H.

IAI-CSIC

{jamartin,tere,ricardo,gonzalez}@iai.csic.es

Teresa de Pedro

IAI-CSIC

Ricardo García Rosa

IAI-CSIC

Carlos González

IAI-CSIC

## Abstract

In this paper, we present a new approach to the design of a fuzzy expressions processor with semantic modifiers and operators. We make use of this approach to define fuzzy operations like “greater than”, “less than” and “between”. These expressions are formalized by means of membership functions. The formalization of the method is discussed, and some examples of its use are presented and justified.

**Keywords:** fuzzy logic, hedges, fuzzy operators, fuzzy concentration and dilation.

## 1 Introduction

In ordinary human language, that is, what is commonly known as natural language, a simple sentence can be transformed into another, more complex sentence by adding a number of words that in turn express shades of opinion about a particular fact. For example, there is a fine distinction between the sentences “it’s a humid day” and “it’s a [rather] humid day”. Fuzzy logic based systems can analyze linguistically expressed knowledge mathematically by automatic processing families of sentences like the above according to a process paralleling a grammar.

A modifier (hedge) is an operation that manipulates the truth value of an expression.

An introduction to the processing of fuzzy expressions from the linguistic viewpoint is given in [1], in which the concepts of modifiers or hedges appear. An extension of hedge is the use of the expressions “greater than” and “less than” as explained in [2] and defined in [5, 6]. Here, we propose a further extension with the introduction of “between”.

In the course of the ORBEX project, we simulated the design and implementation of a processor specialized in processing fuzzy operations with the idea of producing an integrated circuit at the end of the project. As a result of the falling prices of commercial processors, however, we changed our minds during the project, and the final product was the simulated version, in which only some of the components, specifically hedges such as “very” or “extremely” and their extensions “greater than” and “less than”, were integrated. This was precisely the originality of ORBEX [6] as compared with other systems like Matlab Fuzzy Logic Toolbox.

## 2 Syntax and semantics associated with hedges and their extensions

Let us start by giving a description of the expressions with which we are going to deal. These expressions are of the type of conditional clauses that are defined by an antecedent and a consequent in logic and by a clause composed of a protasis and an apodosis linked by the illative conjunction “then” in grammar [10]. We will give a formal description of the syntax as per [8]:

Sentence = “IF” protasis “THEN” apodosis

Protasis = subject predicate { (“AND” | “OR”) subject predicate }

Predicate = [ “NO” ] ( [ modifier | comparative ]

| “BETWEEN” adjective “AND” adjective )

Modifier = “NOT VERY” | “VERY” | “EXTREMELY”

Comparative = “GREATER THAN” | “LESS THAN”

Apodosis = subject adjective { “AND” subject adjective }

where the language has been simplified by removing the articles and also the verbs assumed to be the third person of the copulative “to be”, resulting in expressions resembling how American Indians are portrayed to speak in Hollywood westerns.

For information on how to process both modifiers and comparatives, see [5, 6]. Here we will dwell on the processing associated with the use of the preposition “between” defined in the 22<sup>nd</sup> edition of the DRAE (Dictionary of the Royal Academy of the Spanish Language):

*Entre, prep. - Denota: 1. la situación o estado en medio de dos o más cosas || 3. estado intermedio. Entre dulce y agrio || ...<sup>1</sup>*

Let us look at the differing degrees of semantic difficulty involved in calculating how well a given value,  $x^0$ , of the variable corresponding to the subject satisfies an expression by adding more syntactic richness as shown in [9]:

- a) “subject adjective” is  $\mu^0 = \mu(x^0)$
- b) “subject modifier adjective” can be calculated by applying the function,  $f$ , associated with the modifier –not very, very, extremely-,  $f^0 = f(\mu^0)$ .
- c) another “subject modifier adjective”, using what we have termed comparatives –greater than and less than-. In this case, the value associated with the expression has to be calculated using the function,  $\mu(x)$ , and the membership functions,  $\mu_{<}$  or  $\mu_{>}$ , should be calculated from the definitions of  $\mu(x)$ , where the most common definition of these functions associated with the concept on the left-hand side and right-hand side of  $\mu(x)$  are expressed in Figure 1.

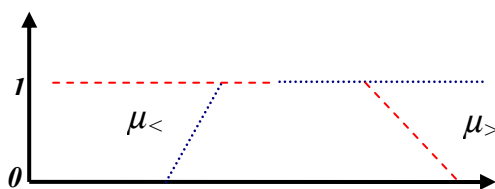


Figure 1. Right-hand side and left-hand side

## 2.1 Example

Going back to the sentence “it’s a humid day”, the semantics associated with “humid day” is provided by the membership function  $\mu$ , where  $\mu$  evaluates the truthfulness of a given perception or measurement of humidity by any of the linguistic

<sup>1</sup> *between, prep. Indicating 1. situation or state intermediate to two or more things || 3. intermediate state. Between sweet and bitter || ...*

labels defined for this concept. Accordingly, the meaning of adding adverbs is realized by applying functions on  $\mu$ . The most commonly used functions are processed in a standard manner as per the following examples:

$$\text{“very”} = \mu^2 \text{ and “more or less”} = \mu^{1/2}.$$

If “humid day” refers to atmospheric “humidity”, which is described using the mathematical variable (h) and can have any of the following linguistic values:

$$\{ \text{zero, low, medium, high} \}$$

where each one has an associated membership function, which (although there are several possible forms, like triangle, trapezium, Gaussian, ramps, etc.) takes the form of the trapezium illustrated by the function in Figure 2. for high humidity, defined by axes  $x_0, x_1, x_2, x_3$ .

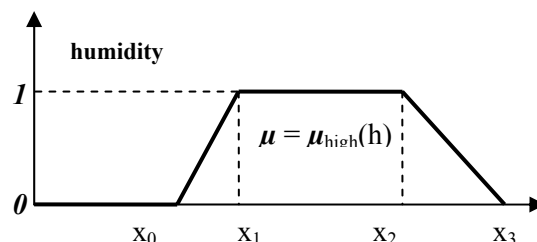


Figure 2. Membership function of the linguistic label high humidity.

This membership function can be modified by means of whichever operation is required. Hence, “very high humidity” will be represented by:

$$\begin{aligned} \mu_{\text{very high}} &= (\mu_{\text{high}}(h))^2 \\ \mu_{\text{more or less high}} &= (\mu_{\text{high}}(h))^{1/2} \end{aligned} \tag{1}$$

Other variations on the original sentence can be formed from comparative judgments like “greater than”, “less than” or “between”, which will also call for a mapping or reinterpretation of the original membership function. The membership functions associated with “greater than” and “less than” are derived from the complement of the original membership function, which, in the case of “greater than” turns out to be the right-hand complement and, in the case of “less than” translates to the left-hand complement, as illustrated in Figure 3.

This complementarity means that these mappings can be written as follows:

a) greater than high:  $\mu_{>high} = 1 - \mu_{high}$  for  $x^0 > x_2$ , otherwise = 0

b) less than high:  $\mu_{<high} = 1 - \mu_{high}$  for  $x^0 < x_1$ , otherwise = 0

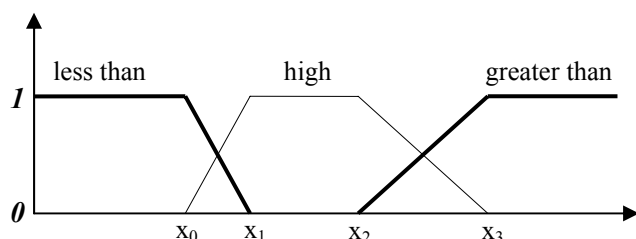


Figure 3. Membership functions of the linguistic labels “greater than” and “less than”

The comparative judgement “between a and b” is a different case, as this preposition must be associated with two adjectives, for example “humidity between medium and high”, resembling, for example, the membership function shown in Figure 4.

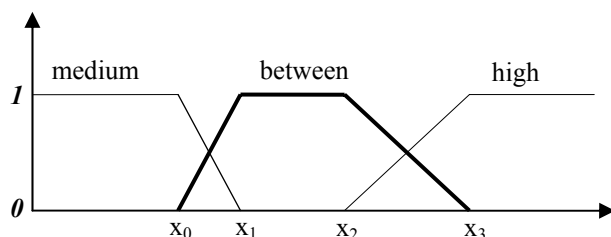


Figure 4. Membership function of the linguistic label “between medium and high”

### 3 Formalizing the mapping procedure by means of hedges

To formalize the use of the above-mentioned hedges, we will start with a definition assuming that the fuzzy partitions are modeled by means of trapeziums, as is the case in ORBEX and CBU[5, 6].

Let V be a fuzzy variable whose crisp value is a real number  $x^0$  and whose linguistic labels or fuzzy partitions are  $\{ P_1, P_2, P_3, \dots, P_n \}$ , where each  $P_i$  is a trapezium. We will denote by  $\mu_i(x^0)$  the value of

the membership function of the linguistic label or partition  $P_i$  when V has the crisp value  $x^0$ .

$$\mu^0 = \mu_i(x^0) \tag{2}$$

So, for simple mappings like “very” and “more or less”, a mathematical function will be directly applied to the number  $\mu^0$  to get the value of the modifier. This is done as follows:

$$\mu_{very} = \mu_i(x^0)^2$$

$$\mu_{more\ or\ less} = \mu_i(x^0)^{1/2} \tag{3}$$

, and, generally,

$$\mu^0 = f(\mu_i(x^0)) \tag{4}$$

, where  $f(\cdot)$  can be any real function of a variable with domain  $[0,1]$ . The most common functions are  $f(x) = x^2$  or  $f(x) = x^{1/2}$ , as already illustrated for the cases “very” and “more or less”, respectively.

The extension of this form of functional composition is what will allow us to model another kind of more useful and at the same time more versatile mappings, such as “greater than”, “less than” and “between”, among many other possible options, because, as we shall see, this same type of functional mapping can be extended not only to functions of a variable but also to expressions whose parameters are actually other membership functions or fuzzy partitions. These mappings or operations will be denoted by  $\Omega$ .

Therefore, for the case of “greater than” and “less than” the resulting  $\Omega$  expression is as follows:

$$\mu^0 = \Omega_{>}(x^0, \mu_i) : 1 - \mu_i \text{ for } x^0 > x_2, \text{ otherwise } = 0.$$

$$\mu^0 = \Omega_{<}(x^0, \mu_i) : 1 - \mu_i \text{ for } x^0 < x_1, \text{ otherwise } = 0.$$

where  $x^0$  is the crisp value of the variable V in a given measurement and  $\mu_i$  is the membership function of the linguistic label or i-th fuzzy partition of the variable V and  $\Omega$  is the type of mapping or operation.

Hence, generally:

$$\mu^0 = \Omega_{mod}(x^0, \mu_i) \tag{5}$$

where  $x^0$  is the crisp value of the variable V and  $\mu_i$  is the membership function of the linguistic label or fuzzy partition  $P_i$ , which, in this particular case, is a trapezium, and {mod} refers to the type of mapping.

An extension to the multifunctional case is illustrated in Equation 6.

$$\mu^0 = \Omega_{\text{mod}}(x^0, [\mu_i \dots \mu_n]) \quad (6)$$

where  $[\mu_i \dots \mu_n]$  is a list of membership functions of the variable  $V$  and  $\Omega_{\text{mod}}$  represents the modifier or hedge.

Taking the Equation 6. as a framework, we find that it is the natural extension of this same principle that allows us to model the comparative judgment “between a and b” and other more complex types of judgments where the complex modifiers or hedges will be somehow parameterized and whose semantic meaning will depend on which membership functions are instantiated in a particular sentence or argument. Then, the case of the complex modifier “between a and b” can be represented and computed by means of the following expression:

$$\mu^0 = \Omega_{\text{between}(a,b)}(x^0, \Omega_{>}(x^0, \mu_a), \Omega_{<}(x^0, \mu_b)) \quad (7)$$

where  $\mu_a$  and  $\mu_b$  are the partitions that act as the lower bound and upper bound. This complex modifier  $\Omega_{\text{between}(a,b)}$  can be computed as follows:

$$\Omega_{\text{between}(a,b)} = \text{t-norm}(\Omega_{>}(x^0, \mu_a), \Omega_{<}(x^0, \mu_b)) \quad (8)$$

or, simply,

$$\mu^0 = \Omega_{\text{between}(a,b)} = \Omega_{>}(x^0, \mu_a) \times \Omega_{<}(x^0, \mu_b) \quad (9)$$

which is equivalent in ordinary language to the sentence:

$\text{Between}_{(a,b)} = \text{GREATER THAN (a) AND LESS THAN (b)}$

Having formalized this framework, the possibility of combining simple and more complex hedges is clear. For example, the sentence “temperature more or less between a and b” would translate as follows:

$$\mu^0 = \mu_{\text{more or less}}(\Omega_{\text{between}(a,b)}) \quad (10)$$

which, generally, would be written as the following meta-expression:

$$\mu^0 = f(\Omega_{\text{mod}}(x^0, \mu_i \dots \mu_n)) \quad (11)$$

where  $f(\cdot)$  can be any real function of a variable with domain  $[0,1]$  and  $\Omega_{\text{mod}}$  is any of the defined complex modifiers.

So, this is how the operations of “not between a and b”, “strictly between a and b” or “more or less between a and b” are defined.

## 4 Implementation

### 4.1 Motivation and background

The ORBEX system was designed to implement any kind of fuzzy systems applications. However, our primary applications fell within the control area, and it became the universal fuzzy controller (CBU), which combines a great expressiveness with a special-purpose architecture for real-time [3] applications, see Figure 5.



Figure 5. A Clavileño vehicle demo in Madrid.

A new application [4] within the simulation field (Figure 6) led us to think of using the preposition “between” to extend the expressiveness of the sentences used. This new version is called CBUX.



Figure 6. SIAMI Project Coal Extraction

Here we describe what the benefits of its use and the theoretical and practical problems involved in its implementation are. We also give some preliminary solutions and examples.

## 4.2 Tests and Results

We developed a fuzzy controller according to the outlined design, which was built to easily implement and interpret the hedges and mappings mentioned in section 2. To this end, our controller operates with fuzzy partitions defined by means of trapeziums. This controller allows the definition of a fuzzy logic program written in a text file following a special grammar defined in section 2 according to the standard in [8]. This grammar allows the system input variables to be defined by declaring a name for each variable followed by a set of linguistic labels or fuzzy partitions that are defined by four numbers to represent a trapezium for each fuzzy partition.

The system uses rules to make inferences. The defuzzification procedure then gets the output variables, which are defined by means of a singleton-type function.

This can be illustrated by an example taken from the SIAMI project [4], the aim of which is to model performance in terms of coal extracted by coal mining machinery.

A small part of the fuzzy program is listed below.

Inputs:

```
Thickness { low 0 0 2.4 4 high 2.4 4 5 5 }
Dip { low 0 0 15 30 high 15 30 35 35 }
Inclination { low 0 0 5 10 high 5 10 20 20 }
Fractures { low 0 0 1 2 medium 1 2 2 3
            high 2 3 5 5 }
Barren rocks { medium 0 50 50 100 }
Hardness { coal 0 0 30 40 slate 30 40 60 100
           sandstone 50 80 100 100 }
```

Outputs:

```
Progress { slow 0 medium 0.5 fast 1 }
```

Rules

1. **IF** Dip low AND Thickness high AND Inclination low **THEN** Progress fast
2. **IF** Thickness high AND Dip high AND Inclination high **THEN** Progress slow
3. **IF** Hardness sandstone **THEN** Progress slow
4. **IF** Hardness **LESSTHAN** slate **THEN** Progress fast
5. **IF** Fractures **BETWEEN** low AND high **THEN** Progress medium

As we can see, this program uses the LESSTHAN and BETWEEN hedges/modifiers.

An example of the Input/Output of this program is given below:

Inputs:

```
Dip: 0.0
Inclination: 0.0
Hardness: 35.0
Barren rocks: 25.0
Fractures: 2.5
Thickness 4.0
```

The level of satisfaction of each rule is shown in Table 1, where (w) represents the level of satisfaction or weight of each rule.

Table 1: Rule satisfaction

Rule No.	Level of satisfaction
1	w = 1
2	w = 0
3	w = 0
4	w = 0.5
5	w = 0.5

We find from Table 1 that rules 4 and 5, which use the modifiers LESSTHAN and BETWEEN, correctly modulate the analysis space without the need to previously define partitions or trapeziums to get a finer grained coverage of the input space. In this way, the fuzzy reasoning application is easier, more intuitive and simpler to design, as modifiers that are very often used in human reasoning can be employed.

## 5 Conclusions

We presented a theoretical and methodological approach to the design of a fuzzy logic expressions processor that provides for the use and definition of different kinds of hedges and modifiers.

One of the problems with analyzing linguistic expressions algorithmically is that there is a wide variety of grammatical structures (heterogeneity), which is an obstacle to the construction of robust and flexible algorithms. Therefore, the framework presented here has been defined with the intention of implementing a method that can consistently (homogeneously) process linguistic expressions with hedges and complex modifiers, such as, for example, the preposition "BETWEEN" and its extensions like "not between a and b", "strictly between a and b" and "more or less between a and b" in the same way as simpler expressions like "very", "not very" and negation are processed in [5,6].

We now have an extended preliminary version of a linguistic expressions and fuzzy variables processor (CBUX) that can process simple and complex linguistic expressions alike.

## Acknowledgments

This paper is part of the technological results achieved within the SIAMI project (Carbonar S.A.), funded by the Centre for Technological Industrial Development (CDTI 20040116), Ministry of Industry.

## References

- [1] Alejandro Sobrino, "Lógica borrosa y lingüística" in *Aplicaciones de la lógica borrosa*, pp 89,105. CSIC. ISBN 84-00-07271-5.
- [2] J. Gutiérrez, F. Fernández, "Procedimiento para la aplicación de modificadores lingüísticos basado en particiones borrosas". Estylf 2002.
- [3] J. Eugenio Naranjo "Sistema de conducción automática de vehículos basada en lógica borrosa y Sistemas Globales de Posicionamiento por Satélite: Programa Autopía". Fac. Informática, UPM, 2005
- [4] Luís Argüelles, Jose M. Rivas, Javier Toraño, Carlos González, R. García, Teresa de Pedro, Jose Antonio Martín H., *Fuzzy Modelling for Coal Seams A Case Study for a Hard-Coal Mine*, in proceedings of the Tenth International Conference on Computer Aided Systems Theory Eurocast 2005, Las Palmas de Gran Canaria, Spain, 2005. pp. 13
- [5] R. García Rosa and T de Pedro "Modelling a Fuzzy Coprocessor and its Programming Language", *Mathware & Soft Computing* 2,3 (1988), 167-174.
- [6] R. García Rosa and T de Pedro "First Applications of the ORBEX Coprocessor: Control of Unmanned Vehicles", *Mathware & Soft Computing* 7 (2000), 265-273.
- [7] Ulrich Bodenhofer, "Fuzzy "between" Operators in the Framework of Fuzzy Orderings". In *Intelligent Systems for Information Processing: From Representation to Applications*, B. Bouchon, L. Foulloy and R.R. Yager editors. Elsevier, Amsterdam 2003, pp 59-70. ISBN 0-444-51379-5.
- [8] Wirth, Niklaus "What Can We Do about the Unnecessary Diversity of Notations for Syntactic Definitions? *Communications of the ACM*, November 1977, vol 20, no. 11, pp 822-823
- [9] Soto, Adolfo and Trillas, Enric, *Antónimos y Modificadores Lingüísticos*, VII Congreso Español sobre Tecnologías y Lógica Fuzzy (ESTYLF'97).
- [10] DRAE, 22<sup>nd</sup> edition, 2001