AVOIDING RULE EXPLOTION AND MAKING APPROXIMATE INVERSE REASONING IN COMPUTING WITH WORDS APPLICATIONS

Oscar G. Duarte[(a)]

[(a)]Universidad Nacional de Colombia

[(a)]ogduartev@unal.edu.co

ABSTRACT

Computing with words applications are mostly built using rule based systems, which have two important lacks: first, it is not possible to deal with high dimension problems because the size of the rule base increases exponentially; and second, there are no way two compute inputs from outputs. In this paper we show an alternative kind of system that remedy those lacks in some applications. It is based on fuzzy arithmetic rather than fuzzy logic. We also show a web application tool for the environmental impact assessment.

Keywords: Computing with words, Fuzzy arithmetic, Inverse Reasoning, environmental impact assessment

1. INTRODUCTION

Fuzzy sets are an useful tool for representing linguistic concepts, as has been recognized from the earliest Zadeh's papers. The concept of linguistic variable was established by Zadeh (1975a, 1975b and 1976), and it has been the keystone of further developments, such as the computing with words (CW) paradigm shown by Zadeh (1999).

In the CW paradigm, a system compute *words from words using words*. Words involved here must be well defined in the context of a Precisiated Natural Language (PNL), that most of times is a set of Linguistic Variables, Modifiers and Semantic Rules.

Figure 1 shows the structure of a Rule Based CW system. The inputs and outputs are words (it computes words from words); the main block is the Approximate Reasoning block, a typical Mamdani inference engine whose rule base is a linguistic one (it uses words in the computation).

The Linguistic Interpretation block translates words into fuzzy sets, the Inference Engine calculates fuzzy sets, and the Linguistic Approximation block translates fuzzy sets into words.

A simple Linguistic Interpretation block just can process labels of the PNL; its output is a the fuzzy set that is related with the label in a Linguistic Variable.

A simple Linguistic Approximation block compares the output of the Inference Engine with the labels of the Linguistic Output Variable, and selects the most similar. Comparison is made with any kind of similarity measure, for example the consistency:

The consistency between two fuzzy sets x, y over de same Universe of Discourse U and with membership functions $\mu_x(u)$ and $\mu_y(u)$ respectively is:

$$\text{cons}(x,y)=\sup_u (\min(\mu_x(u),\mu_y(u))$$

A Rule Based CW system has two major lacks:

1. It is not possible to use a Rule Based CW systems in high dimension applications. The rule base has a combinatorial complexity, and as a consequence we just can manipulate a low number of variables and labels. For a simple 7 inputs and 5 labels in each input, we should define up to 5^7 = 78,125 rules. Even if we would be able to do it, the linguistic meaning of that amount of rules is unintelligible. A single rule whose antecedent has seven atomic expressions is also not easy to understand.
2. It is not possible to make inverse reasoning. By Inverse Reasoning we mean the process of computing inputs from outputs. With a Mamdani Inference Engine we cannot make it.

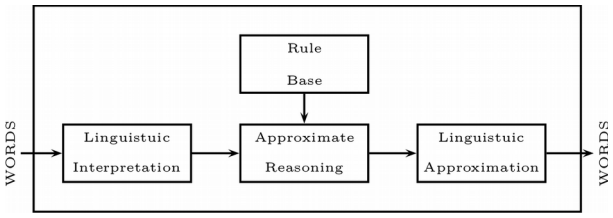

Figure 1: Rule Based CW system

2. ARITHMETIC BASED CW SYSTEMS

In this paper we propose an Arithmetic Based CW system: we propose to compute *words from words using fuzzy numbers* (In fact inputs and outputs are not restricted to words, as we will show later). Figure 2 shows the system structure: The main block is the Approximate Reasoning block that calculates fuzzy numbers from fuzzy numbers using an *approximate reasoning function* (*fra*) instead of a rule base.

In the following, we assume:

- The system has n inputs x_1, x_2,⋯, x_n and one output y. Multiple output systems are not

ISBN 978-88-97999-85-0; Affenzeller, Bruzzone, Jiménez, Longo and Piera Eds.

presented here for simplicity reasons, but can be easily built.

- Every input variable x_i is defined over U_i=[0,1] and the output variable y is defined over V=[0,1]. The sets of all fuzzy sets and fuzzy numbers over U_i are F_{Ui} and N_{Ui} respectively, and the sets of all fuzzy sets and fuzzy numbers over V are F_V and N_V respectively. We define $F_U=F_{U1}$ x FU2 x ... x F_{Un} and $N_U=N_{U1}$ x N_{U2} x ... x $N_{un.}$
- For every input variable x_i there is a linguistic variable X_i with a set of q_i linguistic labels L_{xi-1}, L_{xi-2}, ⋯, L_{xi-qi}. Every linguistic label L_{xi-j} has a related fuzzy set f_{xi-j}.
- Analogously, the output variable y has a linguistic variable Y with a set of r linguistic labels L_{y-1}, L_{y-2},⋯, L_{y-r} and every linguistic label L_{y-i} has a related fuzzy set f_{y-i}.
- Every fuzzy set f_{xi-j} and f_{y-i} has a trapezoidal shape. We will use the notation $f = T(a, b, c, d)$ in order to indicate that fuzzy set f has the membership function shown in figure 3.

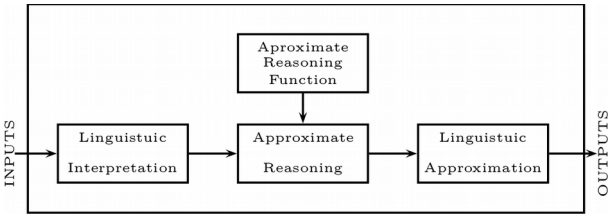

Figure 2: Arithmetic Based CW system

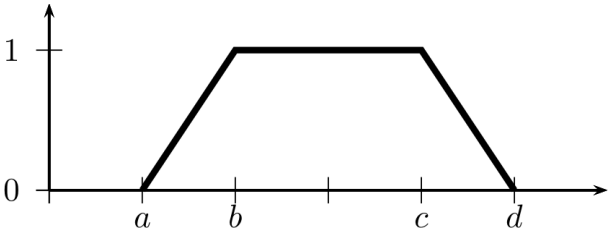

Figure 3: Trapezoidal fuzzy number $f = T(a, b, c, d)$

We will use the usual fuzzy number definition (a normal fuzzy set over R with the upper semicontinuous property). Note that $f = T(a, b, c, d)$ implies that f is a fuzzy number. Now we are ready to explain the blocks shown in figure 2.

2.1. Linguistic Interpretation

The objective of the Linguistic Interpretation block is to translate the inputs into fuzzy numbers. The inputs may be of different types and the output will be always a trapezoidal fuzzy number fn_i for every input. The following are valid types of inputs:

Crisp numbers: if the actual input x_i is the crisp number a, the output will be the singleton fuzzy number $fn_i=T(a,a,a,a)$.

Intervals: if the actual input x_i is the interval $[a,b]$, the output will be the rectangular fuzzy number $fn_i=T(a,a,b,b)$.

Trapezoidal fuzzy numbers: if the actual input x_i is the trapezoidal fuzzy number $T(a,b,c,d)$ the output will be the same input, $fn_i=T(a,b,c,d)$

Linguistic Labels: if the actual input x_i is the linguistic label L_{xi-j} the output will be its related fuzzy set f_{xi-j}.

Modified Linguistic Labels: valid modifiers are "at least" and "at most" Table 1 shows the corresponding output. In Table 1 we assume the linguistic label L_{xi-j} has a related fuzzy set $f_{xi-j}=T(a,b,c,d)$

Simple words: simple valid words are "nothing" and "anything" Table 1 shows the corresponding output.

Table 1: Valid inputs of the types "Modified Linguistic Labels" and "Simple Words"

Input	Output
"at least" L_{xi-j}	$fn_i=T(a,b,1,1)$
"at most" L_{xi-j}	$fn_i=T(0,0,c,d)$
"nothing"	$fn_i=T(0,0,0,0)$
"anything"	$fn_i=T(0,0,1,1)$

2.2. Approximate Reasoning

A typical Mamdani Inference Engine may be viewed as an application $AR: F_U \rightarrow F_V$. However, as typical inputs of this block come from a Linguistic Interpretation block (or from a fuzzyfier block in a fuzzy controller) they are really fuzzy numbers. As a result, we may view the typical Mamdani Inference Engine as an application $AR: N_U \rightarrow F_V$ (note that $N_U \subset F_U$).

We propose a different kind of Inference Engine that may be viewed as an application $ARF: N_U \rightarrow N_V$. In other words, the input of the approximate reasoning block will be the fuzzy numbers $fn_1,fn_2,...,fn_n$ and the output will be the fuzzy number O:

$$O=ARF(fn_1,fn_2,...,fn_n) \qquad (1)$$

ARF is an *Approximate Reasoning Function* whose objective is analogous to that of the rule base in a rule based system: it must capture the knowledge about the system.

We argue that in some applications the knowledge about the system is too poor and it has no sense to make a rule base with it. There are some situations in which the only knowledge available is something like:

- "Every time input i increases the output increases (or decreases)"".
- "Input i is more important than input j "

Of course, we could build a rule base with that knowledge, but a simple crisp arithmetic function can also capture that knowledge. As an example, consider the weighted average function

$$y=\sum_{i=1}^{n} w_i x_i \quad \sum_{i=1}^{n} w_i=1 \; w_i \geq 0 \qquad (2)$$

Equation (2) tell us that output y increases as well as any of the inputs x_i increases. If $w_i > w_j$ we also know that the effect of varying x_i is greater than that of varying x_j, in other words, we know that input i is more important than input j. Suppose (2) refers to a system with 7 inputs every one with 5 labels. Instead of a rule base of 78.125 rules we just need a single equation.

We propose to build (1) using simple crisp arithmetic functions that we will note as *arf* , and the extension principle.

We list some good *arf* of general application. In all cases w_i is a weight variable that let us manipulate the relative importance of every input; it is restricted by $0 \leq w_i \leq 1$ and $\sum_{i=1}^{n} w_i=1$.

We also define s_i as an auxiliary variable that define the sense of the effect of input i over the output:

- $s_i=1$ if y increases as x_i increases
- $s_i=0$ if y decreases as x_i increases

Option 1: A weighted average that includes the sense of the effect of every input:

$$arf_1=\sum_{i=1}^{n} s_i w_i x_i+\sum_{i=1}^{n} (1-s_i) w_i (1-x_i)$$

Option 2: A modified weighted average in which the importance of every input may be varying:

$$arf_2=\sum_{i=1}^{n} s_i w_i g_i(x_i)+\sum_{i=1}^{n} (1-s_i) w_i g_i(1-x_i)$$

where $g_i:[0,1]\rightarrow[0,1]$ is a monotone increasing function such that $g(0)=0$ and $g(1)=1$. As an example, suppose $g_i(x_i)=x_i^r$; if $r>1$ then the lowest values of x_i will be undervalued, and the highest will be overvalued.

Option 3: A weighted average with an offset:

$$arf_3=0.5+[1+\sum_{i=1}^{n} (-1)^{s_i+1} w_i x_i]$$

2.3. Linguistic Approximation

The objective of the Linguistic Approximation block is to translate the output of the Approximate Reasoning block (the fuzzy number O) into words. However, we propose different types of outputs for different applications:

A single word: We compute the consistence between O and every fuzzy set f_i of the linguistic variable Y

$$c_i=cons(f_i,O) \quad i=1,2,\cdots,r$$

Then we select the label with maximum consistency as the output of the system.

A descriptive sentence: We also compute c_i for $i=1,2,...,r$ and then construct a sentence such as

"Output is P_1 L_{y1}, is P_2 L_{y2}, … ,and is P_r L_{yr}"

where P_i is one of the following modifiers:

- "very possibly" if $0 \leq c_i < 1/3$
- "possibly" if $1/3 \leq c_i < 2/3$
- "low possibly" if $2/3 \leq c_i < 1$

A fuzzy number: A valid output is the fuzzy number O without any change.

A crisp number: Another valid output is a crisp number representing the central value of the fuzzy number O. We use the value of fuzzy numbers defined by Delgado (1988)

A pair of crisp numbers: Another valid output is a pair of crisp numbers representing the central value and the fuzzyness of the fuzzy number O. We use the value and ambiguity of fuzzy numbers defined by Delgado (1988)

2.4. Remarks

We want to remark some important aspects of the Arithmetic Based CW systems:

1. Inputs and Outputs are not restricted to words. We can compute words from a set of heterogeneous variables (words, numbers, intervals, fuzzy numbers).
2. As fuzzy numbers are valid input and output variables, it is easy to concatenate two or more Arithmetic Based CW systems *without loosing information about uncertainty*.
3. Using Arithmetic Based CW systems we can design systems of high dimensionality because there are no rule base explosion. But we pay for it:. we must find a crisp function that captures the input-output relationships.
4. There are some applications in which knowledge is enough to construct the *arf* with simple weighted averages; This could be more simple than finding a Rule Base.
5. In low dimensionality problems we can design either Rule or Arithmetic Based CW systems; in section 3 we make a numerical comparison.
6. In section 4 we will show how to compute inputs from outputs using fuzzy arithmetic.

3. RULE BASED VS. ARITHMETIC BASED CW SYSTEMS

Our aim in this section is to make a numerical comparison between Rule and Arithmetic Based CW systems. As we can not design RBCW systems of high dimensionality we limit our experiments to 2 simple cases: a) 1 input - 1 output system and b) 2 inputs - 1 output system.

Our numerical comparison must be done (obviously) between numbers, but RBCW systems can not compute numbers, just words. As both, ABCW and RBCW systems, use the consistency to perform the Linguistic Interpretation we have selected it as the comparison variable. Two types of inputs have been proven: words and numbers. As RBCW systems can not operate with numbers, we have simulated them with words whose associated fuzzy sets are singletons.

Input and Output Linguistic Variables have been always equal: three labels (LOW, MEDIUM, HIGH) with trapezoidal fuzzy sets $T_L(0.0,0.0,0.2,0.4)$, $T_M(0.2,0.4,0.6,0.8)$ and $T_H(0.6,0.8,1.0,1.0)$

3.1. 1 input - 1 output

In this case we have an input x_1 and an output $_y$, whose relationship is increasing. The Rule Base for the RLCW system is:

- R1: IF x_1 is LOW THEN y is LOW
- R2: IF x_1 is MEDIUM THEN y is MEDIUM
- R3: IF x_1 is HIGH THEN y is HIGH

A suitable Approximate Reasoning Function for the ABCW system is

$$fra:\ y=x_1$$

Tables 2 and 3 show the consistency of the output of the systems when inputs are words. There is no difference between them, and we can conclude that the RBCW system is as good as the ABCW system.

Table 2: Consistency of the output of a RBCW system when inputs are words

Input	*cons(O,L)*	*cons(O,M)*	*cons(O,H)*
LOW	1.0	0.5	0.0
MEDIUM	0.5	1.0	0.5
HIGH	0.0	0.5	1.0

Table 3: Consistency of the output of a ABCW system when inputs are words

Input	*cons(O,L)*	*cons(O,M)*	*cons(O,H)*
LOW	1.0	0.5	0.0
MEDIUM	0.5	1.0	0.5
HIGH	0.0	0.5	1.0

Figures 4 and 5 show the consistency of the output of the systems when inputs are numbers. Note the consistency of the output when input is $x_1=0.3$: In the RBCW system the output has the same consistency (0.5) with the three labels, meaning that it can not distinguish if it is LOW, MEDIUM or HIGH; in the other hand, the ABCW system the output has consistency 0.0 with the third, meaning that it is not HIGH. We must conclude that the ABCW performs a better discrimination of the output.

3.2. 2 input2 - 1 output

In this case we have two inputs x_1, x_2 and an output y. The relationship between y and x_1 is increasing, but between y and x_2 is decreasing. The Rule Base for the RLCW system is described in table 4, where as a suitable Approximate Reasoning Function for the ABCW system we design

$$fra:\ y=0.5(1+x_1-x_2)$$

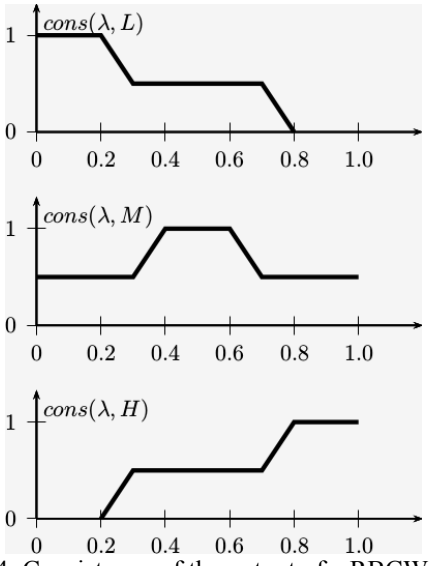

Figure 4: Consistency of the output of a RBCW system when inputs are numbers

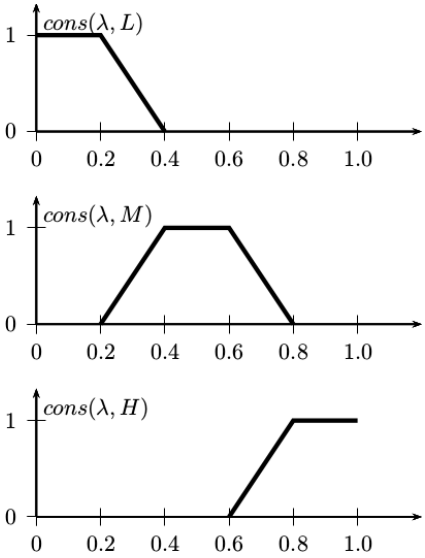

Figure 5: Consistency of the output of a ABCW system when inputs are numbers

Table 4: Rule Base for a RBCW system with 2 inputs

	x_2		
x_1	LOW	MEDIUM	HIGH
LOW	MEDIUM	HIGH	HIGH
MEDIUM	LOW	MEDIUM	HIGH
HIGH	LOW	LOW	MEDIUM

Tables 5 and 6 show the consistency of the output of the systems when inputs are words. Every cell in those tables has three numbers, the consistency of the output with every one of the three labels LOW, MEDIUM and HIGH. Note that every label that has maximum consistency in the RBCW system also has maximum consistency in the ABCW system.

ISBN 978-88-97999-85-0; Affenzeller, Bruzzone, Jiménez, Longo and Piera Eds.

Table 5: Consistency of the output of a 2 inputs RBCW system when inputs are words

	x_2		
x_1	LOW	MEDIUM	HIGH
LOW	0.5/1.0/0.5	0.5/0.5/1.0	0.0/0.5/1.0
MEDIUM	1.0/0.5/0.5	0.5/1.0/0.5	0.5/0.5/1.0
HIGH	1.0/0.5/0.0	1.0/0.5/0.5	0.5/1.0/0.5

Table 6: Consistency of the output of a 2 inputs ABCW system when inputs are words

	x_2		
x_1	LOW	MEDIUM	HIGH
LOW	0.25/1.0/0.25	0.0/1.0/0.0	0.0/0.5/1.0
MEDIUM	1.0/0.5/0.5	0.5/1.0/0.5	0.0/1.0/1.0
HIGH	1.0/0.5/0.0	1.0/1.0/0.0	0.25/1.0/0.25

Figures 6 and 7 show the consistency of the output of the systems when inputs are numbers. We have assumed a monotone relationship between every input and the output; however, the RBCW system produces non-monotone surfaces of the consistencies while thos of ABCW systems are monotone. We argue that ABCW systems represent better than RBCW system the previous knowledge.

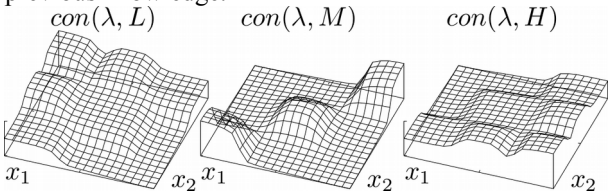

Figure 6: Consistency of the output of a 2-inputs RBCW system when inputs are numbers

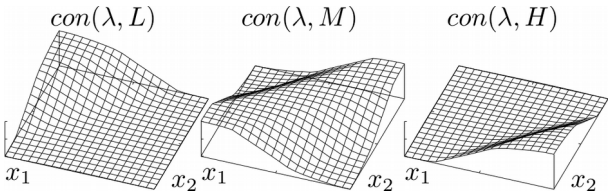

Figure 7: Consistency of the output of a 2-inputs ABCW system when inputs are numbers

4. INVERSE REASONING

CW systems as those of figures 1 and 2 are designed to compute outputs from inputs; however, sometimes we need to perform the inverse calculus: to compute one or more inputs from the remaining inputs and the outputs.

In order to precise the above paragraph, suppose a CW system with three inputs x_1, x_2, x_3 and an output y. Suppose also we know the actual values (words or numbers) of the first two inputs, and we want y to get some desired value; the question is ¿which value must have x_3?

If the CW system is Rule Based, we do not have a procedure to compute x_3. If it is Arithmetic Based, we must deal we the lack of invertibility of fuzzy arithmetic (see Yager (1980) and Bouchon (1997)). We propose to use the Algorithm that computes the Necessary Extension of Inverse Functions (see Duarte (2003) and Duarte (2000)).

In few words, the algorithm verifies if exists a suitable value of x_3 (verifies the invertibility). If it does not exist, the algorithm modifies the desired output y in such a way that the inverse exists. The algorithm is valid for monotone decreasing or increasing functions. *arf* listed in section 2.2 are of this type.

Figure 8 shows the proposed system: We want to compute x_k; Inputs are X_k and y, where X_k is the vector of known x inputs; outputs are $x^{nec}{}_k$, the necessary value of x_k, and $\widetilde{y}$, the value of y that we can get with X_k and $x^{nec}{}_k$. The inverse reasoning is performed with the Algorithm that computes the Necessary Extension of Inverse Functions. Linguistic Interpretation and Approximation blocks are analogous to those of ABCW systems. We call such system an Arithmetic Based Computing with Words Inverse (ABCWI) system.

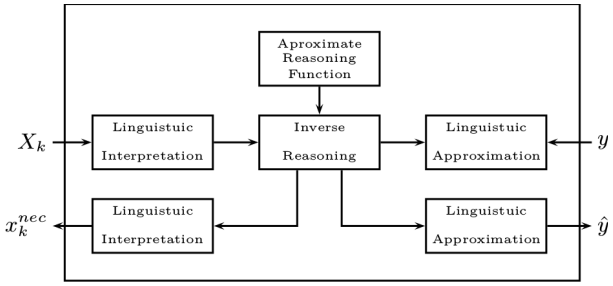

Figure 8: Arithmetic Based Computing with Words Inverse (ABCWI) system

5. AN APPLICATION EXAMPLE

We show in this section an example of risk analysis. It is an ABCW system that computes the risk of damage caused by an atmospheric electrical discharge (lightning) in a specific building (figure 9). A more detailed use of fuzzy techniques for this problem can be found in Gallego (2003); simplification is made here for ilustration proposes. Inputs and outputs are:

- I: it is the *Lightning Intensity*, the peak current of the ligthning.
- D: it is the *Lightning Density*, the number of ligthning per square kilometer that are expected in a year where the building is placed.
- P: it is the *Level of Protection* defined by the type of protection apparatus in the electrical network of the building
- M: it is the *Importance Index* of the facilities in the building.
- R: it is the *Risk of Damage* caused by lightning in the building.

ISBN 978-88-97999-85-0; Affenzeller, Bruzzone, Jiménez, Longo and Piera Eds.

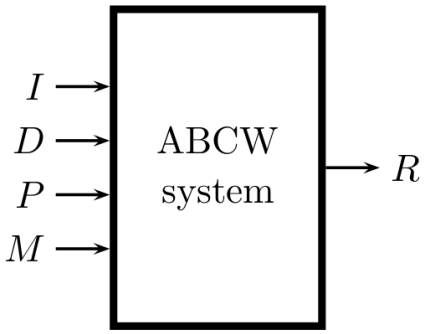

Figure 9: ABCW system of the application example

Note that the relationship between the output R and the inputs *I*, *D* and M is increasing, whereas between *R* and *P* is decreasing (more Level of Protection implies less Risk of Damage). We should study the electrical effects of direct and indirect lightning over the specific electrical network, but this is just an example for illustration, so we omit it. We will use as inputs: numbers for *I* and *D* and words for *P* and *M*. All variables are defined over [0,1] whose linguistic variables are as those of examples in section 3; *I* is the most important variable, so we weight it the double of the others.

In such a situation, we can define *arf* as

arf: R= 0.4I+0.2D+0.2(1-P)+0.2M

Now suppose the inputs are: *I=T(0.5,0.6,0.7,0.8)*, *D=T(0.2,0.3,0.4,0.5)*, P=MEDIUM, M=LOW. The output of the Approximate Reasoning block is a trapezoidal fuzzy number *O=T(0.28,0.38,0.52,0.66)*.

Consistency between *O* and the three labels are:

cons(O,L)=0.40
cons(O,M)=1.00
cons(O,H)=0.18

Meaning that the risk of damage is MEDIUM. Now we can compute what kind of Level of Protection we need if we want to have a LOW Risk of Damage. The ABCWI system calculates $P^{nec}=T(1.0,1.0,1.0,1.0)$ and the Risk of Damage with that Level of Protection would be

$$\hat{R}=T\left(0.24,0.30,0.40,50\right)$$

whose consistency with the three labels is

cons(O,L)=0.62
cons(O,M)=1.00
cons(O,H)=0.00

We must conclude that even with the best Level of Protection P^{nec} we cannot obtain a LOW Risk of Damage.

6. CONCLUSIONS

We can use Fuzzy Arithmetic to design CW systems. Those systems avoid rule explosion for high dimensionality problems; in low dimensionality problems they also have performance as good as the performance of the RBCW systems or even better. Moreover, we can compute inputs from outputs in ABCWI systems that are implemented using the Algorithm that computes the Necessary Extension of Inverse Functions.

ACKNOWLEDGMENTS

If the paper requires an acknowledgements section it can be placed after the main body of the text.

REFERENCES

Bouchon-Meunier B., Kosheleva O., Kreinovich V., and Nguyen H.T.. 1997. Fuzzy numbers are the only fuzzy sets that keep invertible operations invertible. Fuzzy Sets and Systems, pages 155-164, 1997.

Delgado M., Vila M.A., and Voxman W., 1988. On a canonical representation of fuzzy numbers. Fuzzy Sets and Systems, (93):205–216.

Duarte O. G., 2000. Técnicas Difusas En la Evaluación de Impacto Ambiental. PhD thesis, Universidad de Granada.

Duarte O. G., Requena I., and Delgado M., 2003 Algorithms to extend crisp functions and their inverse functions to fuzzy numbers. International Journal of Intelligent Systems, 18: 855–876. doi:10.1002/int.10121

Gallego L. E., Duarte O. G., Torres H., et. al. 2004 Lightning risk assessment using fuzzy logic. Journal of Electrostatics, 60:233–239, March 2004.

Yager R.R., 1980. On the lack of inverses in fuzzy arithmetic. Fuzzy Sets and Systems, pages 73-82.

Zadeh L.A., 1975a. The concept of linguistic variable and its applications to approximate reasoning, part. Information Sciences, 8:199–249.

Zadeh L.A., 1975b. The concept of linguistic variable and its applications to approximate reasoning, part II. Information Sciences, 8:301–357.

Zadeh L.A., 1976. The concept of linguistic variable and its applications to approximate reasoning, part III. Information Sciences, 9:43–80.

Zadeh L.A., 1999. Computing with Words in Information/Intelligent Systems. What is Computing with Words? Physica Verlag.

AUTHORS BIOGRAPHY

Oscar Duarte was born in Bogotá, Colombia. He received the Electrical Engineering degree and the M.Sc. In Industrial Automation from Universidad Nacional de Colombia and the Ph.D. in Computer Science from Universidad de Granada (Spain). He joined the Department of Electrical and Electronics Engineering, Universidad Nacional de Colombia, as a lecturer in 1994. His current research interests include soft computing, modeling and simulation of dynamic systems, control, and engineering education. He is the author of 3 books and more than 40 papers of conference and journals. He was also the Academic Vicedean of the School of Engineering since 2012 to 2016.