Mathware & Soft Computing 5 (1998) 167-174

Modeling a Fuzzy Coprocessor and its Programming Language

Ricardo García Rosa and Teresa de Pedro Lucio
Instituto de Automática Industrial, CSIC
La Poveda, 28500 Arganda del Rey, Madrid, España
e-mail: {*ricardo,tere*}*@iai.csic.es*

Abstract

A computational model for a fuzzy coprocessor (types and structures of data and the set of instructions) is proposed. The coprocessor will be charged only of the typical operations of fuzzy logic as calculating membership degrees, unions and intersectíons of fuzzy sets, fuzzy inferences, defuzzifications and so on. One main novelty is that the programming language admits fuzzy rules conditions in which there would be linguistic edges preceding the predicates and the coprocessor is designed to deal with them directly. In order to that, the compiler codifies as the predicates as the edges into the instructions. Using this coprocessor the number of linguistic values of the variables and the number of rules can be reduced and the executions of systems based on fuzzy rules will be speeded up.

1 Introduction

The first step to solve a problem with fuzzy techniques is to build its linguistic model by describing the variables involved and the fuzzy rules. The syntax of the description language can be independent of the problem domain but the semantics, the meaning of the linguistic values of the variables and the edges, depends on the applications. Even more, the variables can be real or logical and their actual values can be measured by sensors or provided by the users. Taking this fact into account in our model of fuzzy coprocessor we distinguish two parts: the problem description language, or programming language, and the model proper, that is the types of data and the set of machine instructions.

Associated with the description language there is a formal grammar, to form well formed language sentences, and a compiler, which translates the language sentences in machine instructions. Resuming, the sequence of sentences defining a concrete problem become in a set instructions, ones of them will be conventional sentences, for the host processor, and other will be fuzzy instructions, for the fuzzy coprocessor.

It is convenient to consider that some machine instructions can be context dependent, due to the different nature of the systems based on fuzzy rules. For

instance, in the case of fuzzy controllers, the more extended fuzzy systems, the conclusions have to provide crisp numerical values for the action variables, while in other fuzzy expert systems the conclusions will change the linguistic values of the some variables in the base of knowledge.

2 Syntactical description of the programming language

The programming language contains sentences to define fuzzy variables and rules and, as usual, the compiler translates them into the internal memory representation [4]. In the proposed fuzzy coprocessor model, the data and the instructions, are stored separately in different memory structures, a data base and an instruction base. These data base and instruction base are related respectively with the fuzzy variables and rules describing the problem. In other words, the variables involved in the system, named inputs and outputs if the system is a controller or, in general, conditions and conclusions, are converted in elements of the data base, while the rules, keeping the expert knowledge about the problem management, are converted in instructions of the instruction base.

2.1 Formal grammar

The user language is of big worth in rule based systems, namely in fuzzy systems. Taking into account that our purpose is to achieve a tool for Spanish people, we have employed the Spanish as a base for the programming language, so the key words of the formal grammar, written between " " are Spanish words. Nevertheless, the coprocessor model proper is independent of the programming language.

Formal grammar in a Backus normal form

```
Texto = "entradas:" variables "salidas:" variables "reglas:" oraciones "fin"
    variables = variable {variable}
    oraciones = oración {oración}
      variable = nombre real (etiquetas | "grupo" "3"|"5"|"7"|"9") real real) "fin"
        etiquetas = etiqueta {etiqueta}
          etiqueta = nombre (("trapecio" | "m") real real real real —
                             ("triángulo" | "t") real real real |
                             ("ascenso"|"a" |"descenso"|"d"|"campana"|"c") real
                             real )
    oración = ("SI" | "si") prótasis ("ENTONCES" | "entonces" ) apódosis
      prótasis = nominal {("Y" | "y" | "O" | "o") nominal}
        nominal = sujeto (afirmativa | negativas )
          afirmativa = verbo [ modificador ] adjetivo
            verbo = "ES" | "ESTA" | "es" | "está"
              modificador = "MUY" | "BASTANTE" | "EXTRA" |
                            "ALGO" | "POCO" | "CASI" | "APENAS" |
                            "MAYORQUE" | "MENORQUE" |
```

```
                    "muy" | "bastante" | "extra" | "algo" | "poco" |
                    "casi" | "apenas" | "mayorque" | "menorque"
    negativas = ("NO" | "no") afirmativa { ("NI" | "ni" ) sujeto afirmativa }
apódosis = sujeto verbo adjetivo { ("Y" | "y" ) apódosis }
  sujeto = [artículo] nombre
  adjetivo = nombre
```

3 Data

From an analysis of the formal grammar it is possible to realize that the types of data required to model the base of knowledge of a system based on fuzzy rules, are the following: variables, labels -named "etiquetas" in the formal grammar- and parameters [3]. These data are stored in three chained lists, one for the variables, other for the fuzzy sets and the last one for the parameters. Schematically we can write:

"variable $\rightarrow$ labels $\rightarrow$ parameters"

meaning that a variable points to the fuzzy set representing its first linguistic value and a label, that is a fuzzy set, points to its first parameter. Besides the data required to represent the problem description we have introduced a register, named weight, which purpose will be understood after.

3.1 Variables

The variables represent the state of the system, in the case of fuzzy systems they can take linguistic values defined by fuzzy sets and crisp numerical values, to store the current values measured or inferred trough sensors or provided by users, their memory structure is:

variable = real, pointer

where the field "real" contains a real numerical value and the field "pointer" contains the address from which the list of their linguistic values and their associated fuzzy sets are stored in memory.

3.2 Fuzzy sets

They are the data storing the meaning of the linguistic values, labels, assigned to the variables; their structure is the following:

fuzzy-set = type, pointer

where the field "type" contains a code referent to the kind of membership function and the field "pointer" contains the address from which the list of their parameters is stored in memory. In our model we have included the more common kinds of membership functions: slope-up, slope-down, bell, peak -or triangle- and trapeze that we can abbreviate as "u", "d", "b", "p" and "t" respectively.

3.3 Parameters

This kind of data refer to the numerical parameters fixing the range of definition of the membership functions, they are stored in an array, so a fuzzy set is fixed by its type and its array of parameters. As it is explicit in the formal grammar, there are two parameters associated with the fuzzy sets of types slope-up, slope-down and bell, three parameters associated with the fuzzy sets of type peak and four parameters associated with the fuzzy sets of type trapeze.

3.4 Registers

We have designed our coprocessor in such a way that a special register, name weight, is needed; its function will be clear later, because it is related with the execution of the instructions and not with the representation of the data extracted by the compiler.

4 Instructions

The basic set of instructions of the proposed fuzzy coprocessor contains only four operations: AND-IF, OR-IF, THEN and DEFUZZIFY. The general format of the instructions is a field of code and, except for the DEFUZZIFY instruction, several fields of parameters, whose number depends on the type of the instruction.

4.1 AND-IF and OR-IF instructions

These instructions are very similar, they allow representing a condition part of a rule, or antecedent. They are the more complex instructions, their structure is respectively:

AND-IF variable verb edge label
OR-IF variable verb edge label

Where the meaning of the different fields, according with the formal grammar, is the following:

Variable =	a condition variable
Verb =	(IS | IS_NOT) an (affirmative | negative) condition
Edge =	(NULL | VERY | LITTLE | EXTRA | MORE-THAN | LESS-THAN)
Label =	a linguistic value associated with the variable

The processing of each parameter causes a sub-instruction, detailed below, and the final result is obtained in several steps after executing all the following sub-instructions:

1. Access to the variable
2. Access to the label
3. Apply the verb (calculate the membership degree or the complementary)
4. Apply the edge is any (modify the membership degree if needed)
5. Apply the conjunction (apply the T-norm or the S-norm)
6. Store the result in the weight register

The first and second steps get the current values of the condition variable and the fuzzy set label respectively. The sub-instruction of the third step calculates the membership degree of the current value of the variable to the fuzzy set label, if the verb is affirmative this degree is the result of the sub-instruction, but if the verb is negative the result is the complementary of this membership degree. The fourth step has effect if the field edge is not null, in this case the sub-instruction is to apply the operator of modification chosen to the membership function. (In this point it is worth to note that the inclusion of the new edges MORE-THAN and LESS-THAN represent a relevant novelty in the processing of fuzzy systems [2]). The fifth step is to apply a norm, a T-norm is the instruction is AND-IF and a S-norm if the instruction is an OR-If, to the result of the sequence of sub-instructions and the current contents of the weight register. The result is stored in the weight register.

4.2 THEN instruction

This kind of instruction represents a conclusion of the right part of a rule, or the consequent. Its structure is the following:

THEN label

being label the linguistic value of an output or conclusion variable, in which the result of the instruction is stored. This result is obtained by applying a T-norm between the actual value of the weight register and the value stored in label. The purpose of this operation is to aggregate in label the conclusions of all the rules in which this label appears.

It must be noted that, in the case of the THEN sentence, the formal grammar of the description language limits the value of the verb to the affirmative form IS and avoid edges. We think that these limitations are coherent with the common way in which conclusions are established by experts.

4.3 Defuzzify instruction

This instruction has not parameters because it is supposed it has to be done over all the output variables, though there could be a defuzzify instruction for each output variable. Its function is to calculate the crisp values resulting for every output variables.

5 Coding

The binary and octal codes chosen for the basic instructions are the following:

AND IF	00	0
OR IF	01	1
THEN	10	2
DEFUZZIFY	11	3

Similarly the codes chosen for the verb are the following:

IS	0
IS NOT	1

And the codes chosen for the edges are the following:

NULL	000	0
LITTLE	001	1
VERY	010	2
SUPER	011	3
MORE-THAN	100	4
LESS-THAN	101	5

Finally the codes chosen for the types of fuzzy sets are the following:

SLOPE-UP	00	0
SLOPE-DOWN	001	1
BELL	010	2
PEAK	011	3
TRAPEZE	100	4

5.1 Examples

In order to fix ideas we codify some conditions for the variable temperature and its linguistic value high. It is supposed that the variable temperature is stored in the address VVV and its label high is stored in the address position LLL.

AND IF temperature IS high	00 VVV 0 000 LLL
AND IF temperature IS LITTLE high	00 VVV 0 001 LLL
AND IF temperature IS NOT high	00 VVV 1 000 LLL
OR IF temperature IS NOT VERY high	01 VVV 1 010 LLL

Other example shows a sequence of sub-instructions correspondent to the conditional sentence:

IF the pressure IS VERY high AND the volume IS constant ....

Step	operation
1.	access to the current value of pressure p
2.	access to fuzzy set high
3.	calculate the membership degree: degree = mhigh(p)
4.	modify the membership degree according to operator VERY: degree = VERY (mhigh(p))
5.	calculate weight = T-norm (weight, degree)
6.	access to the current value of volume v
7.	access to the fuzzy set constant
8.	no operation
9.	calculate the membership degree: degree = mconstant(v)
10.	calculate weight = T-norm (weight, degree)

These steps are suitable of being executed in a pipe-line mode according to the following diagram, thus the ten clock cycles can be reduced to the half.

clock cycle	steps
1	1, 2, 6, 7
2	3, 8
3	4, 9
4	5
5	10

6 Conclusions

We have explained a first approach to model a fuzzy coprocessor of general purpose and its programming language. With the topic general purpose we want to mean that the applications to be run in it can be fuzzy controllers or any other kind of fuzzy rules based systems. Until now the generality of our model has not been achieved, it performs right for controllers but probably some improvements have to be done to process other systems. The main elements of the coprocessor model are the data types and structures and the machine instructions. By other hand our aim is that the programming language was as ciose as possible to the Spanish language, in this way it will be meaningful and friendly. Evidently we can not expect that users describe problems in a complete Spanish, but we think possible to admit true Spanish sentences as programming sentences, though we had to limit the allowed syntax.

Note.- This work has been taken as a base for modeling a fuzzy coprocessor integrated in a chip, being developed now in the frame of the project TIC- 96-1393 CO6-03 founded by CICYT.

References

[1] M. Sugeno, T. Yasukawa (1993) A Fuzzy Logic Based Approach to Qualitative Modeling, *IEEE Transactíons on Fuzzy Systems*, vol. 1 n. l: 7-31.

[2] L. de Salvador, J. Gutierrez (1995) Modifiers on Fuzzy Hardware, *Proc. of the Sixth Intemational Fuzzy Systems Association World Congress.*

[3] R. García Rosa, T. de Pedro, M.T. de Andrade (1995) Parallel Inference Engine for Fuzzy Controllers, *Cybemetícs and Systems,* vol. 25, n. 2: 30-40.

[4] M.T. de Andrade, T. de Pedro, R. García Rosa (1 994) Software Tools to Obtain a Distributed Fuzzy Controller, *Proc. of the ISIE'94 International Symposium on Industrial Electronics.*

[5] N. Wirth (1997) What can we do about the unnecessary diversity of notations for syntactic definitions?, *Communications of the ACM*, vol. 20, n.11.

[6] Real Academia Española (1983) *Esbozo de una nueva gramática de la lengua española,* Espasa Calpe.