UNIVERSITÉ LUMIERE LYON 2

INSTITUTE UNIVERSITAIRE DE TECHNOLOGIE

TECHNICAL REPORT

# FUZZY LOGIC: BASICS AND USAGE

BY

OMAR LÓPEZ-ORTEGA
ENSEIGNANT INVITÉ
Universidad Autónoma del Estado de Hidalgo

ABDELIZIZ BOURAS
Directeur
Laboratoire d'Informatique pour l'Entreprise et les Systemes de Production

AÏCHA SEKHARI
Professeur IUT

APRIL – MAI 2010

# CONTENTS

# PROLOGUE

Collaboration among world-wide universities is paramount; in this sense, both students and researches can exchange ideas fluently.

The University of Lyon 2, through the University Institute of Technology (IUT) and in particular the Laboratory of Informatics for the Enterprise and Production Systems (LIESP) has launched a project called e-Link. This project is sponsored by the European Union under the Erasmus Mundus program. Although not part of the e-Link project itself, the Universiry of Lyon 2 runs its own program for Invited professors, with which the UL2 gathers scientist and researchers from abroad.

Fortunately, these two programs converged in time and space. While a great number of students of the e-Link project were still working at IUT and LIESP, prof. Omar López-Ortega came along.

This technical report is the result of this tri-party collaboration. The e-Link students attended the course on Fuzzy Logic and developed some projects. The core ideas are transcribed here.

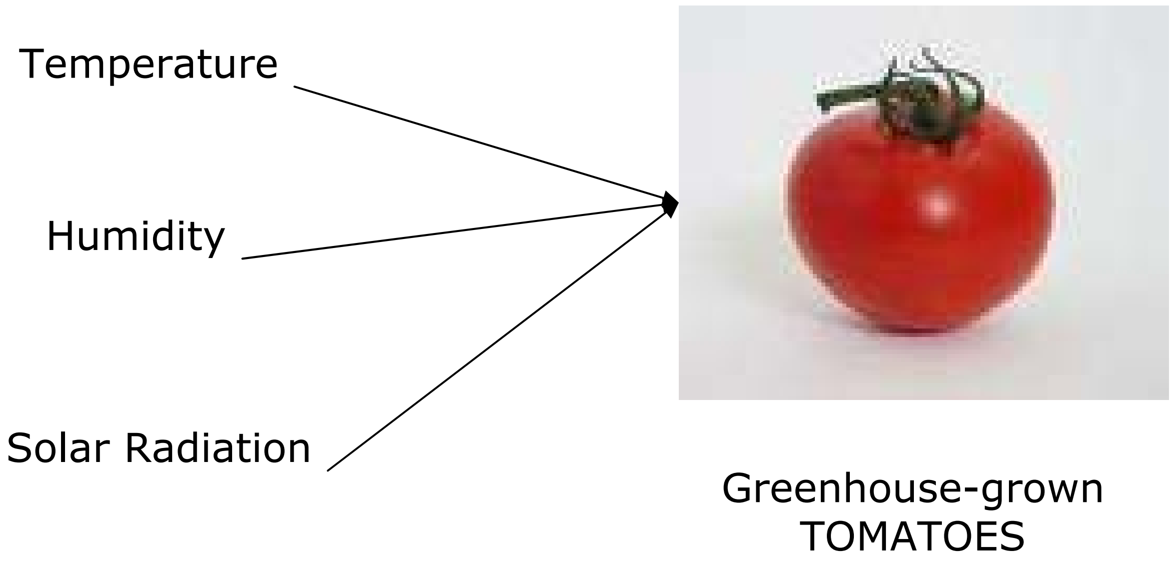
The objective of the course and, consequently, of this technical report is to present basic notions of fuzzy logic for building Mamdani rule-based systems with JFK 1.0 (java fuzzy kit). The authors will be truly satisfied if this effort echoes for long in the students' and members' that were once part of this collaboration.

The authors express their most sincere gratitude to Prof. Akhter Hossain, who organized and kept track of everything for this course to suceed, and also to the e-Link students that patiently and with passion attended the course.

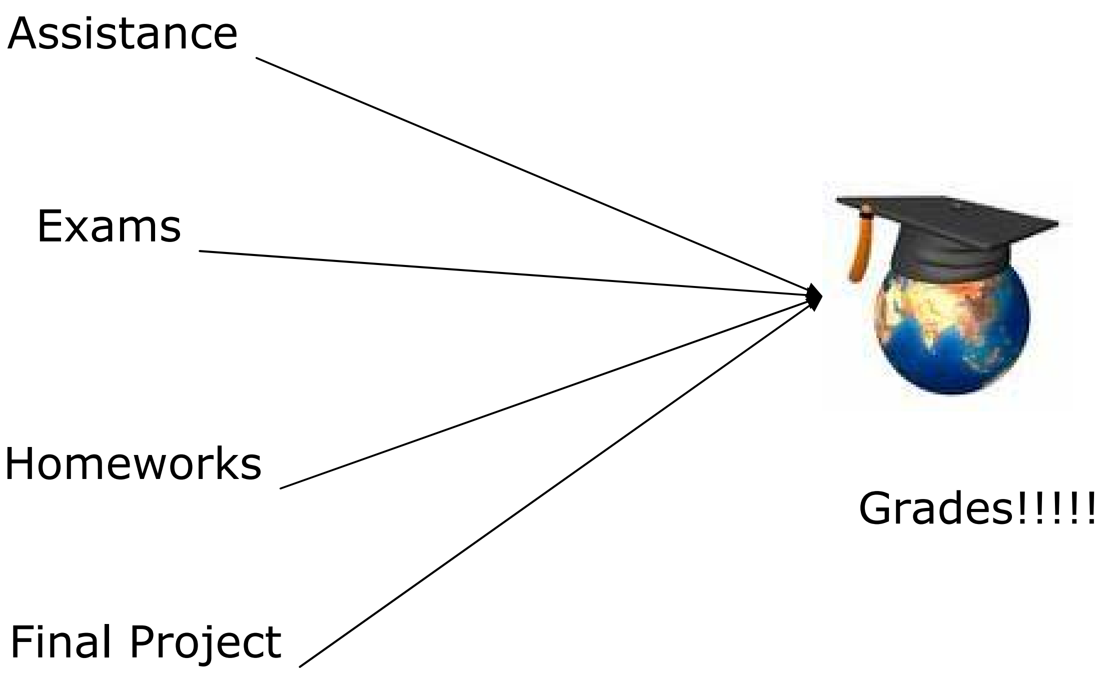
# Chapter 1: Basics of Fuzzy sets and Fuzzy Systems

**1.1 Fuzzy systems overview**

The real world presents an endless number of complex systems. Take, for example, the variables that are present to grow tomatoes in a greenhouse. Those variables, among others, interact in complex ways, which can not be modelled by equations.



Teachers also face this kind of problems when deciding the final grade for pupils, since they comply differently with the tasks assigned along the course.



Assistance

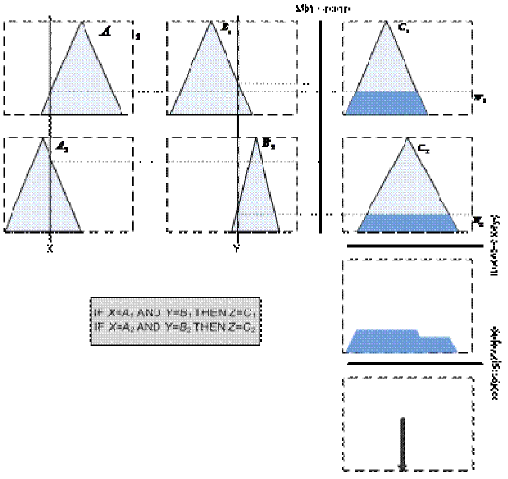
Exams

Homeworks

Final Project

Grades!!!!!

Such variables can be modeled with fuzzy numbers and such complex relations among these variables can be represented by fuzzy rules, as it is shown next:



By using fuzzy logic we can express “qualitative or “soft” relationships such as “the higher the absences the lower the grade” or “high number of absences and a good project provide a grade that is good but not necessarily excellent”.

However...

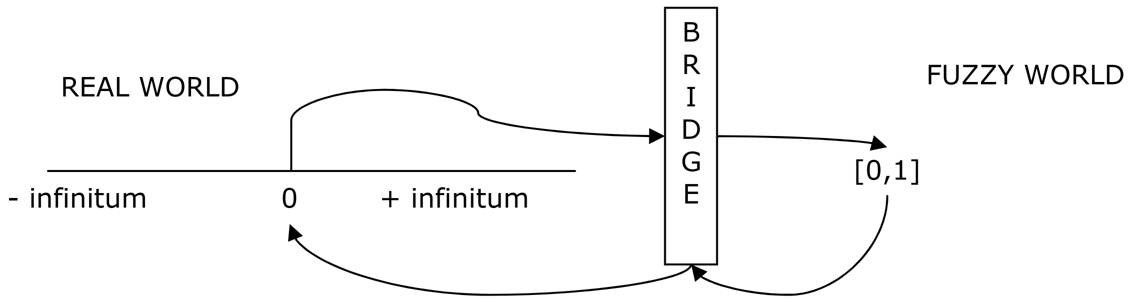
A solution based on “hard” analysis gives this….

While a “soft”qualitative solution gives this…

…not the TRUTH but acceptable

**1.2 Fuzzyfication**

Fuzzy logic is a form of multi-valued logic derived from fuzzy set theory to deal with reasoning that is “approximate rather than precise”. In contrast with "crisp logic", where binary sets have binary logic, fuzzy logic variables may have a truth value that ranges between 0 and 1 and is not constrained to the two truth values of classic propositional logic. Furthermore, when **linguistic variables** are used, these degrees may be managed by **specific functions called membership functions**. Indeed, Fuzzy logic provides tools so we can manage such complex relations in a simpler way, by mapping the real-world values into a space ranging from 0 to 1.



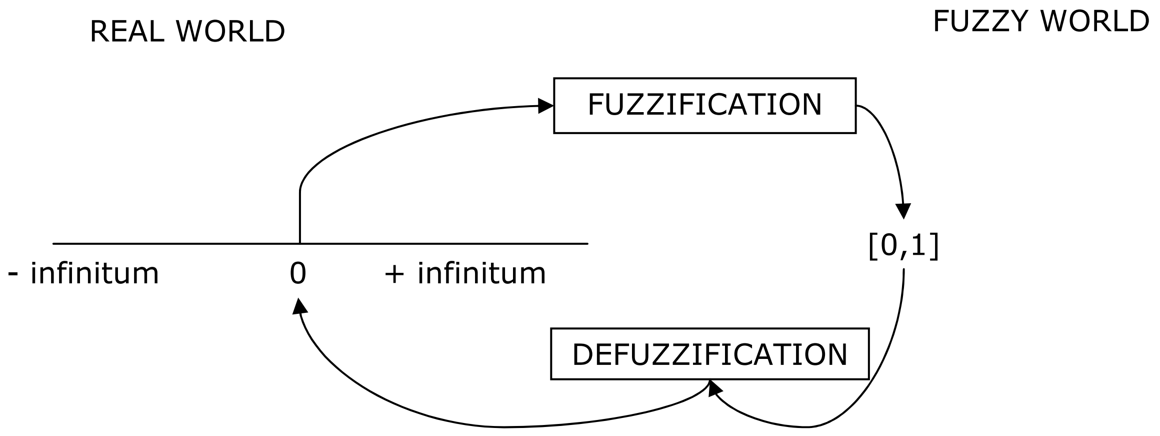
Real world means Real Numbers
(CRISP set)

Temperature
Humidity
Grades!!!!!

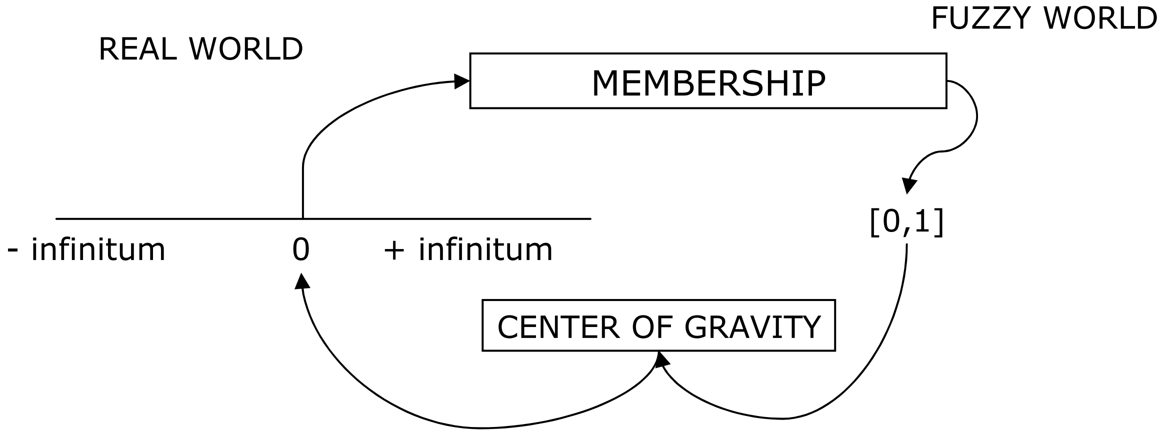
Math laws work here!!!

Set Theory
Modus Ponens
Topology…

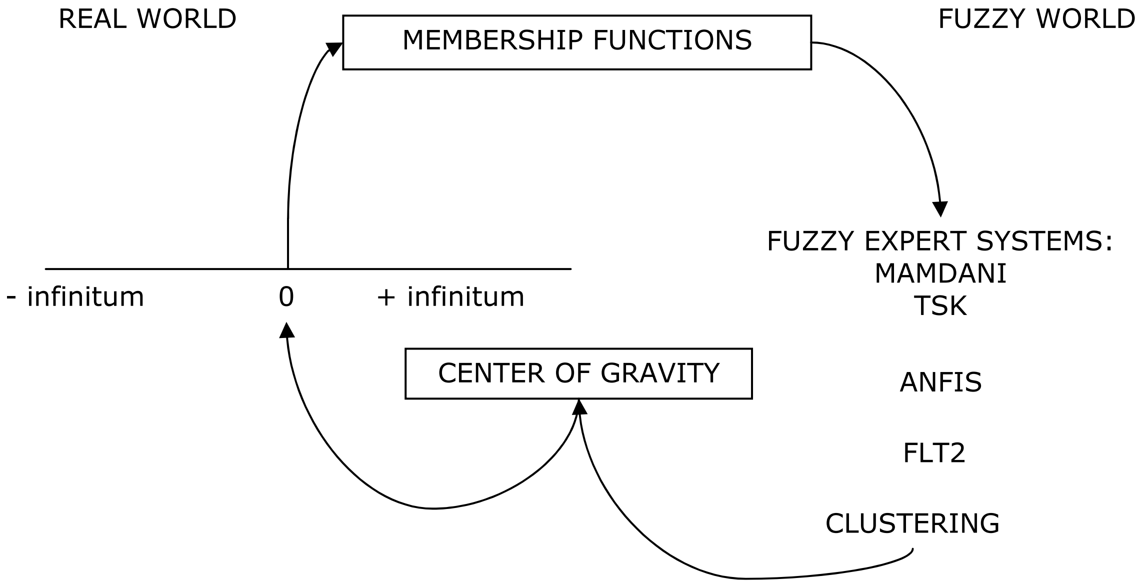
The “bridge” is called “fuzzyfication” when we go from the real world to the fuzzy world. The inverse operation is called “defuzzification”.



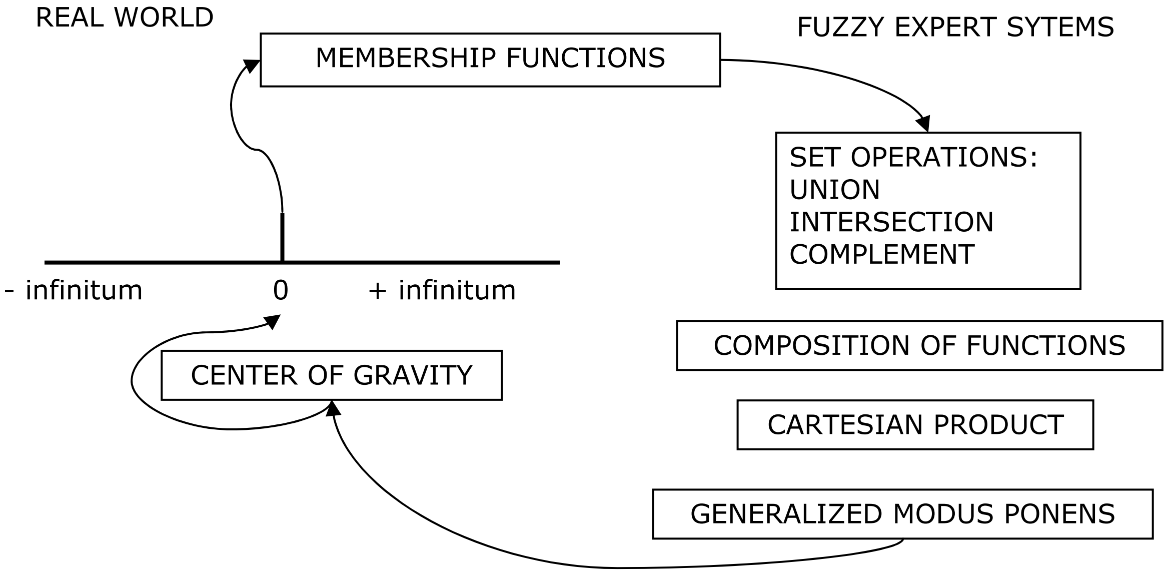
The actual fuzzyfication process is done by **membership functions,** whereas one form of defuzzifying what is fuzzy is through the **center of gravity**.



Once in the fuzzy world, a number of good things can be done. We can, for example, develop fuzzy rule bases. As for this, we can have two types: (i) The Mamdani model and (ii) the Takagi Sugeno Kan model. This latter does not require a specific deffuzification procedure, while Mamdani rulebases do.

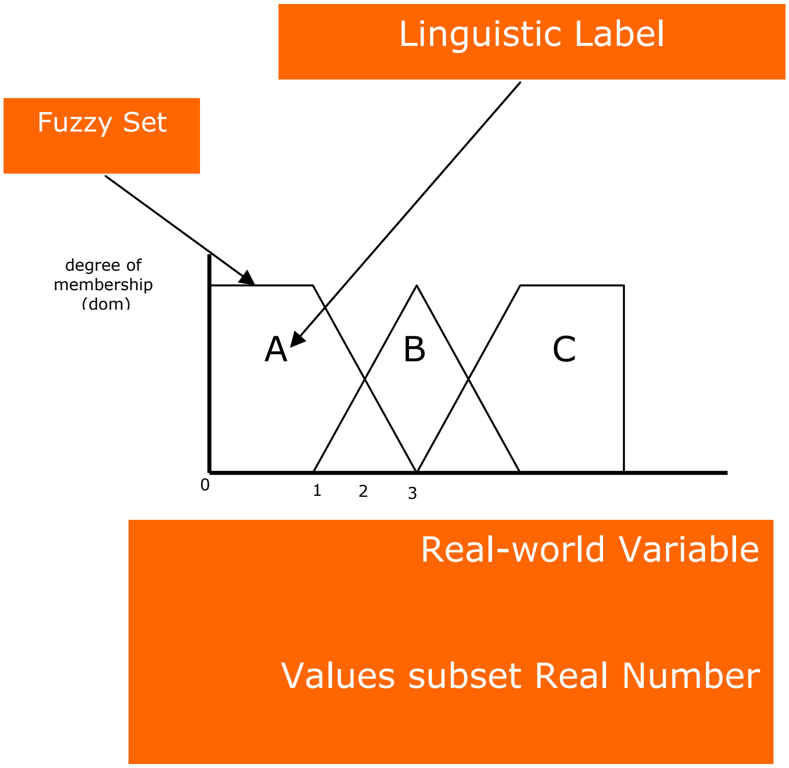


In this technical report, we center in the Mamdani model. To understand it we need to understand the basics of fuzzy mathematics:



But first, we must do the fuzzyfication process. Fuzzyfication serves to analyze analog input values in terms of logical variables that take on continuous values between 0 and ; in contrast to classical or digital logic, which operates on discrete values of either 0 or 1 (true or false). We can assimilate this to the partition of the real-world variable into a series of regions called **fuzzy sets**. We use functions for that.

So, when a real-world value is computed through one of those functions, we obtain the **degree of membership** (dom) of that value on the referred region. This is why they are called membership functions.



[0,1] = mu(Variable)

A Fuzzy Set is built!!!!!

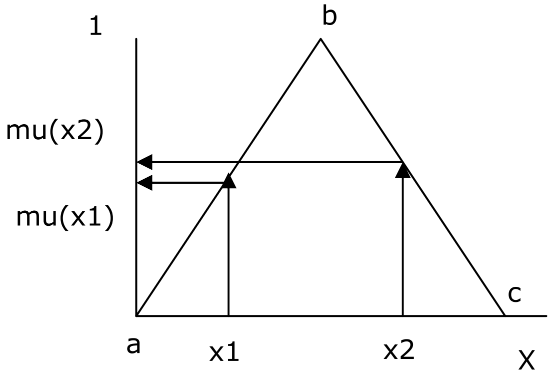
Fuzzy Set = {(var, dom)}

A = {(0,1), (1,1), (2,0.5), (3,0)}

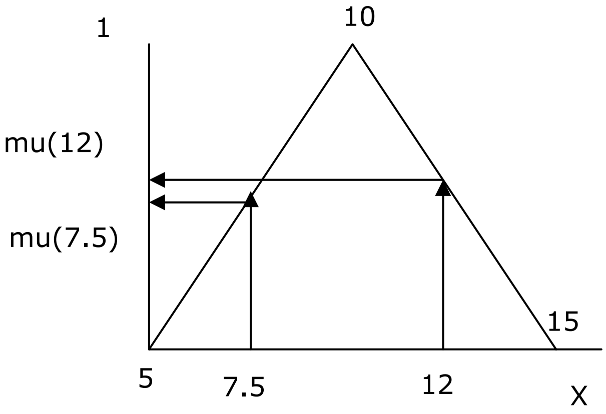
Former figure illustrates several important issues. Firstly, *crisp*, real-world values are mapped into *fuzzy* values by means of a membership function, generally called with the greek letter “mu”. Secondly, crisp values (horizontal axis) are partitioned into fuzzy sets, which in turn are **labeled with linguistic terms**[1]. Therefore, when crisp values are computed with a membership function, their degrees of membership (dom) are obtained, as represented in the vertical axis of the latter figure.

Membership functions are many. The following figure depicts a Triangular function.

[1] Linguistic terms are also known as Linguistic Labels



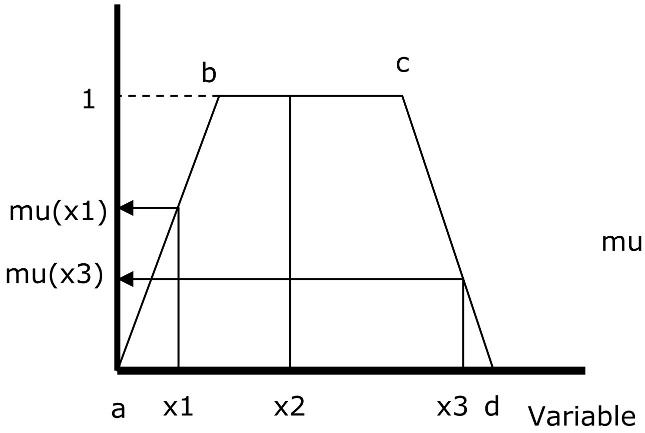
$$mu(x) = \begin{cases} (b-x)/(b-a) \text{ if } a<= x <b \\ (c-x)/(c-b) \text{ if } b <= x < c \\ 0 \text{ otherwise} \end{cases}$$



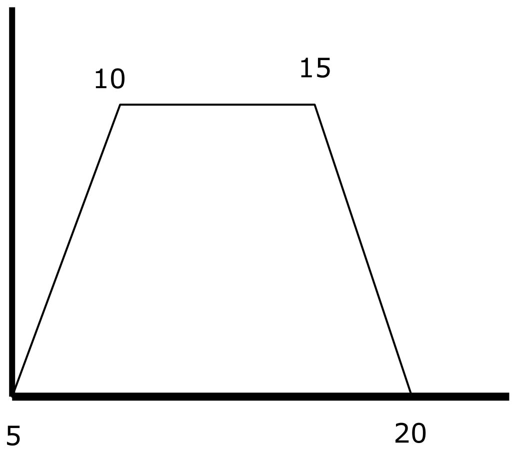
$$mu(7.5) = (10-7.5)/(10-5)$$
$$= 2.5/5$$
$$= 0.5$$

$$mu(12) = (15-12)/(15-10)$$
$$0.6$$

A Trapezoidal function takes the following form:



$$mu(x) = \begin{cases} (b-x)/(b-a) \text{ if } a<= x <b \\ 1 \text{ if } b <= x <= \\ (d-x)/(d-c) \text{ if } c < x <= d \\ 0 \text{ otherwise} \end{cases}$$



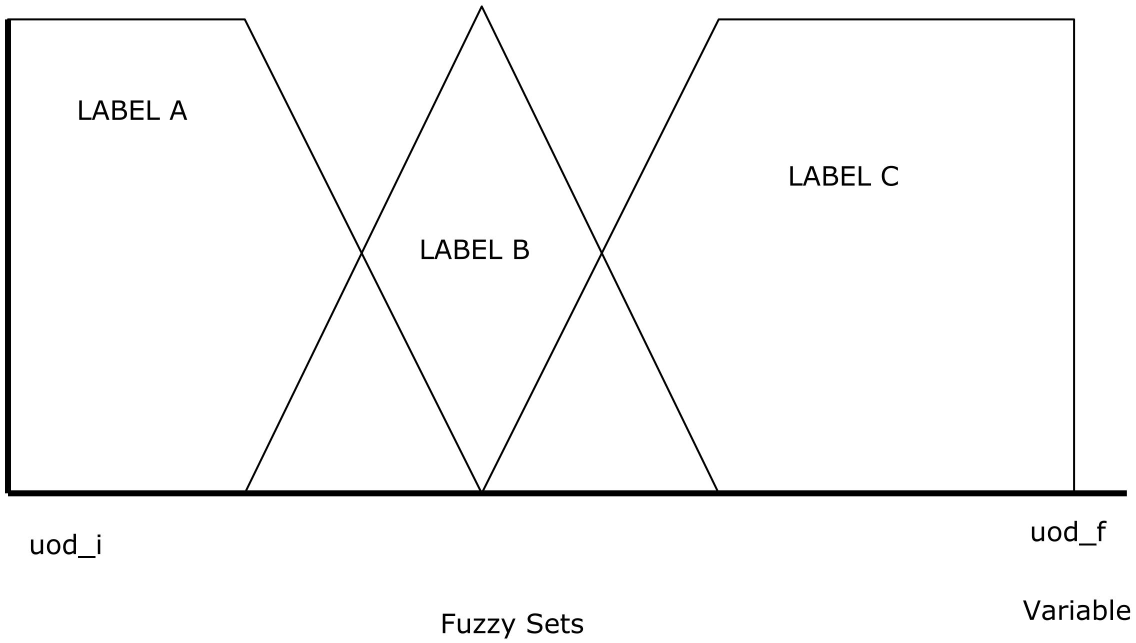
$$mu(x) = \begin{cases} (b-x)/(b-a) \text{ if } a< x <b \\ 1 \text{ if } b < x < c \\ (d-x)/(d-c) \text{ if } c < x < d \\ 0 \text{ otherwise} \end{cases}$$

Trapezoidal(7.5)?

Trapezoidal(14.9)?

Trapezoidal(18)?

The fuzzification process ends when a real-world variable is fully partitioned and the regions are labeled accordingly:



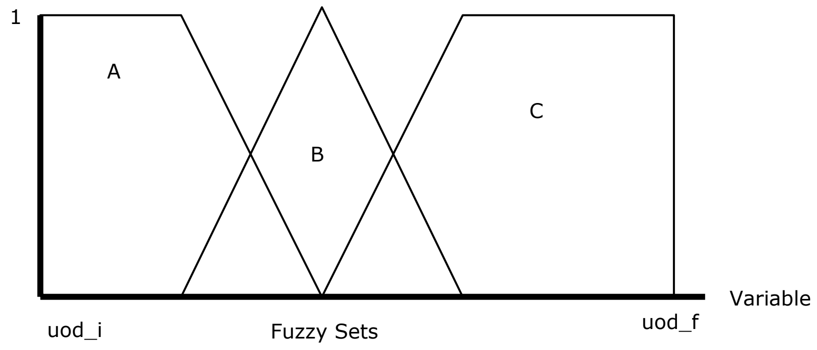
In the previous figure, **uod** means Universe of Discourse (the actual crisp values of the real-world variable). Hence, “uod_i” corresponds to the initial crisp value of the UoD, whereas “uod_f” is the final value of the UoD.

# Chapter 2. Mathematical background. Part 1.

## 2.1 Set operations

In this chapter we introduce three basic operations between fuzzy sets: (i) Intersection, (ii) Union and (iii) Complement. Then, we present a special set of operators called Triangular norms and Triangular co-norms, used as an extension of the basic intersection and union, respectively.

Remember, a real-world variable is partitioned into a series of regions called fuzzy sets:



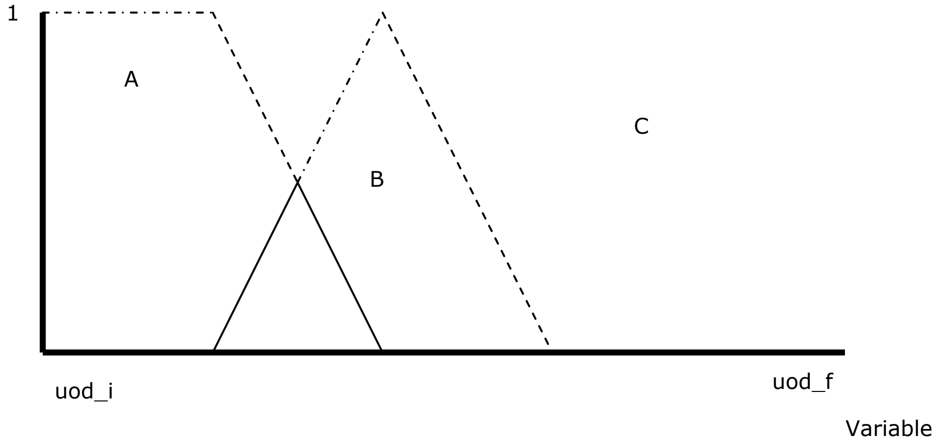
[0,1] = mu(Variable)

A Fuzzy Set is built!!!!!

Fuzzy Set = {(var, dom)}

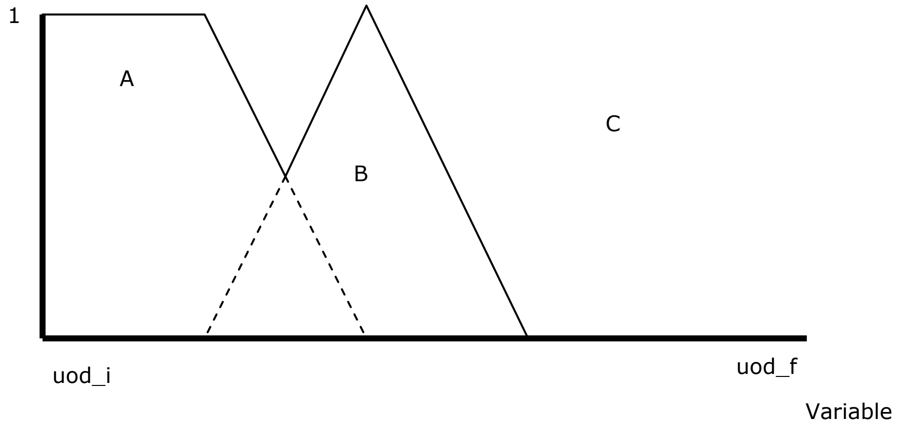
Low = {(0,1), (1,1), (2,0.5), (3,0)}

The **Intersection** of two fuzzy sets is:



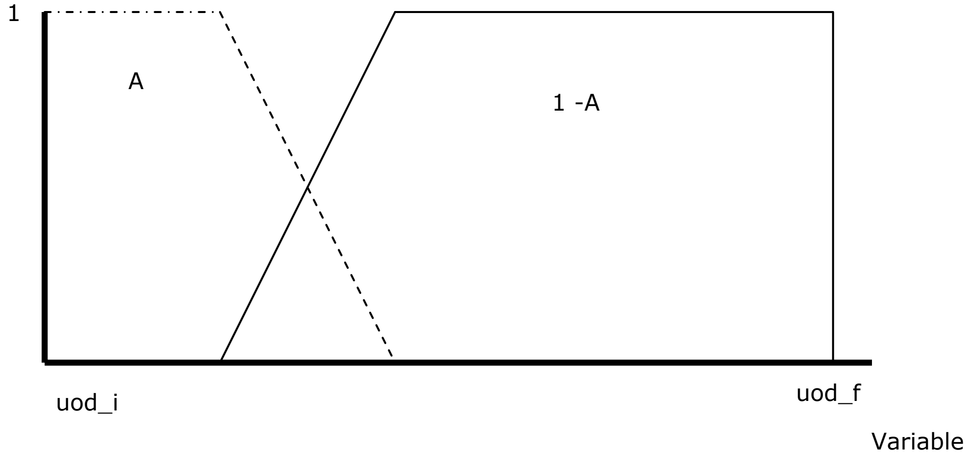
Intersection of two Fuzzy Sets :=
compute the minimum of the membership functions
INTERSECTION = min(mu(A), mu(B))

Acordingly, the **Union** of two fuzzy sets is:



Union of two Fuzzy Sets :=
compute the maximum of the membership functions
UNION = max(mu(A), mu(B))

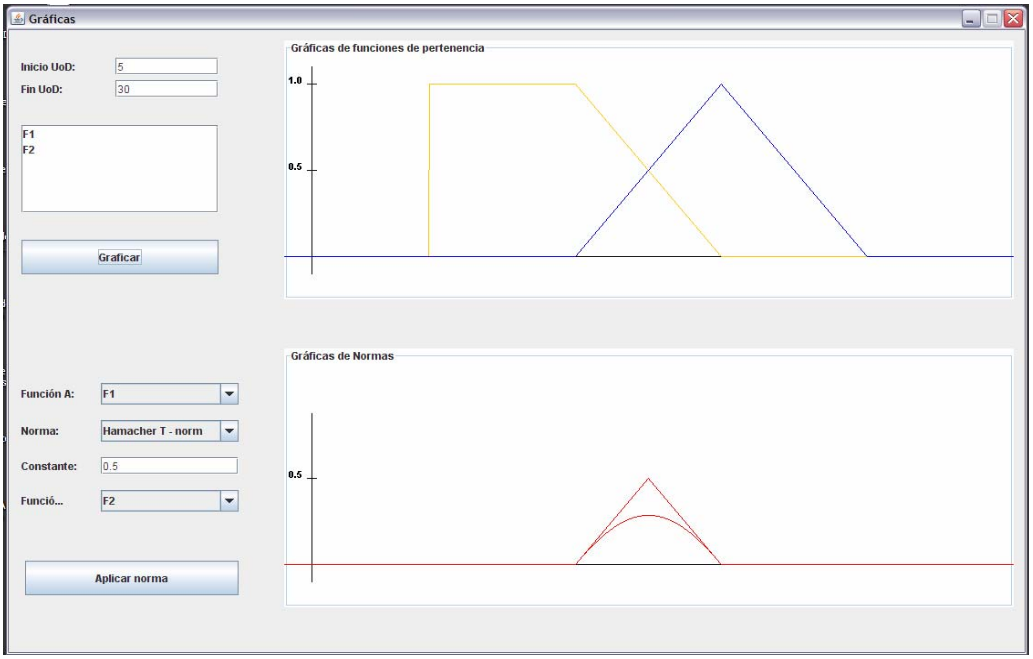
Finally, the **Complement** of a fuzzy set is defined like the following figure illustrates:



Complement of a Fuzzy Set :=
compute 1 - the membership function
COMPLEMENT = 1 - (mu(A))

### 2.2 Triangular norms and co-norms

The Intersection and the Union of sets can be computed not only with the minimin and the maximum operators. There are a number of functions that have been developed (Hammacher, Weber, Yagger, to name but a few) and used within the fuzzy logic framework. They are called Triangular Norms and Triangular Co-Norms. When we use them the results change a bit, as exemplified in the following figures.

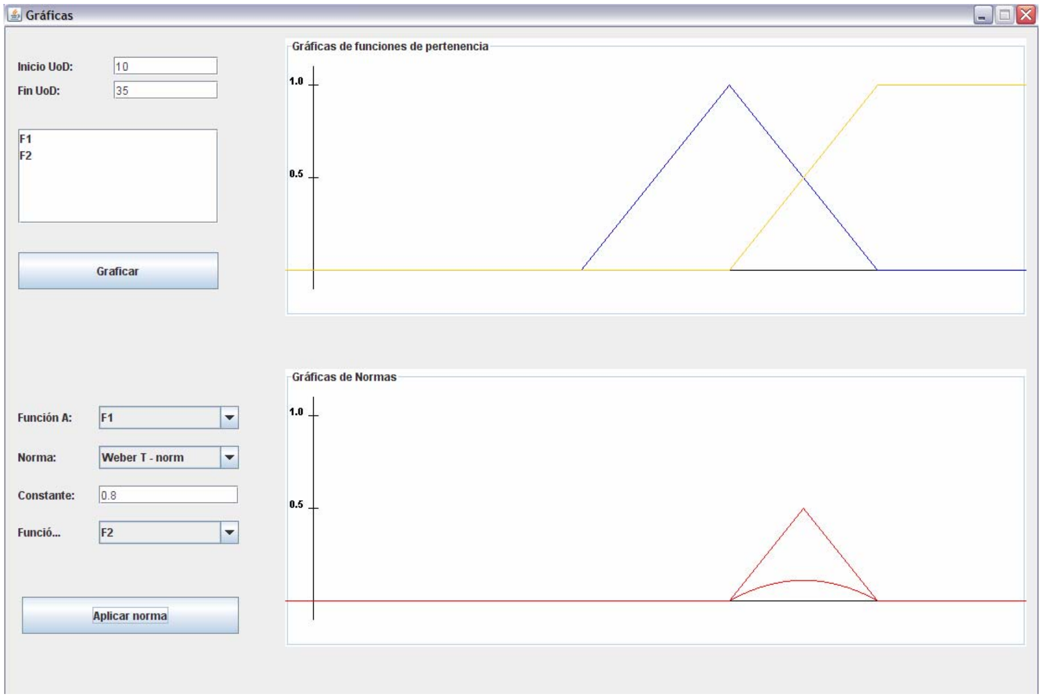


Inicio UoD: 5
Fin UoD: 30

F1
F2

Función A: F1
Norma: Hamacher T - norm
Constante: 0.5
Funció... F2

In the former figure, given two sets, we compute the basic intersection operation and the t-norm called Hamacher (the curve-shaped set in red).



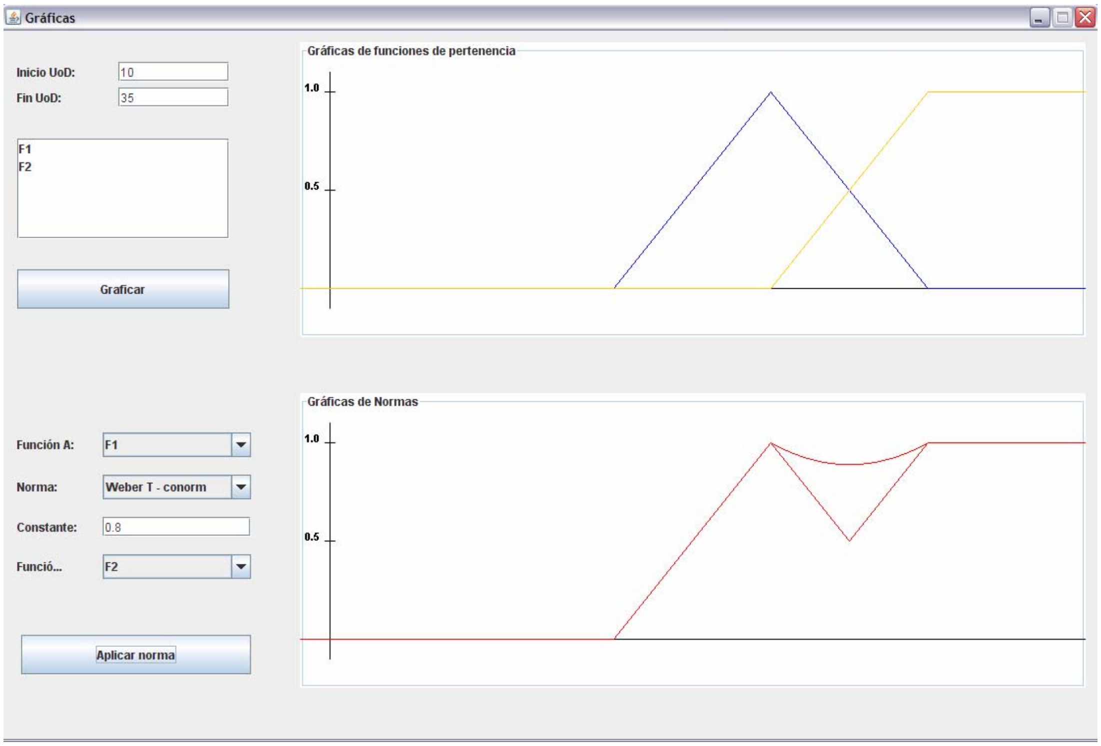
Inicio UoD: 10
Fin UoD: 35

F1
F2

Función A: F1
Norma: Weber T - norm
Constante: 0.8
Funció... F2

The former illustration shows the result of computing a Weber T-norm.

As for the computations of T-conorms we present also results.

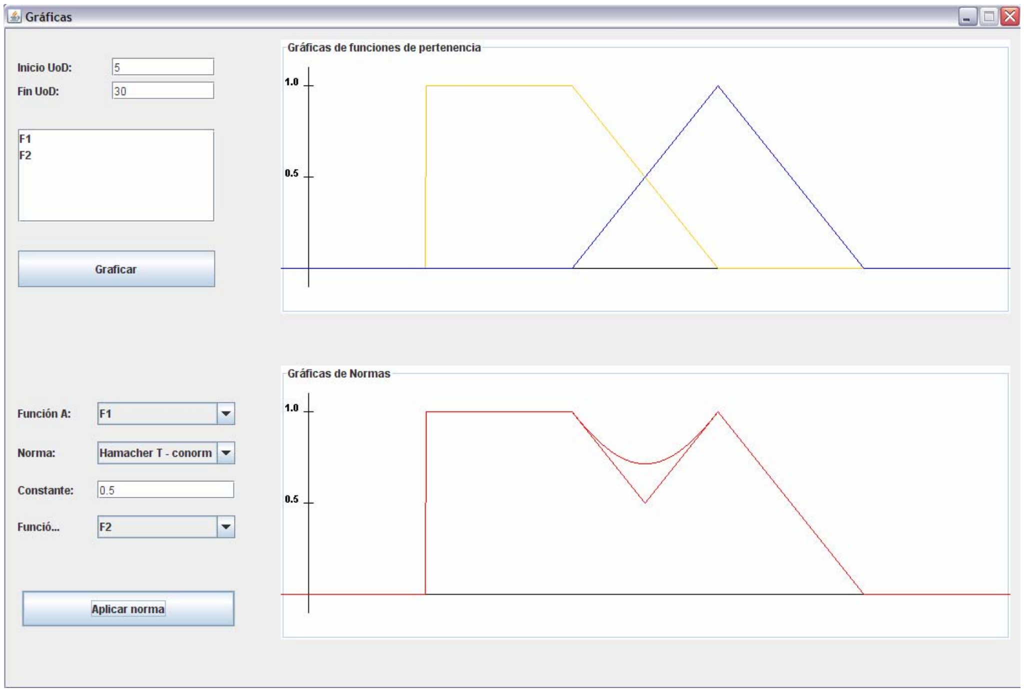


Inicio UoD: 10
Fin UoD: 35

F1
F2

Función A: F1
Norma: Weber T - conorm
Constante: 0.8
Funció... F2

The former illustrations depicts both the simple Union of fuzzy sets and the computation of the Weber T-conorm. In the next graphic we show the computation of the Hamacher T-conorm.



Inicio UoD: 5
Fin UoD: 30

F1
F2

Función A: F1
Norma: Hamacher T - conorm
Constante: 0.5
Funció... F2

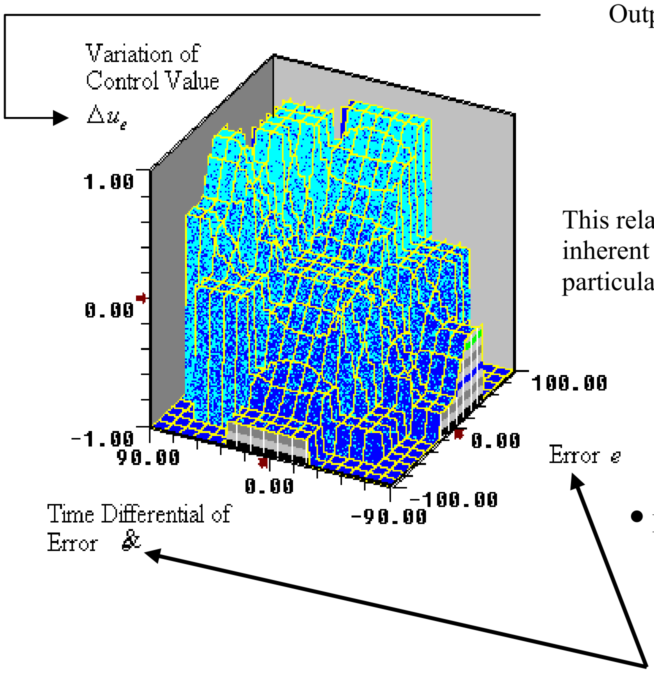
The triangular operations of T-norms and T-conorms help to compute values residing around the minimum and the maximum, respectively, but never reaching those. It is interesting to experiment the calculation of fuzzy rule bases with T-norms and T-conorms, since the results vary slighty.

# Chapter 3. Mathematical background. Part 2.

### 3.1 Cartesian Product

In this chapter we present more complex operations with fuzzy sets, namely the **Cartesian Product** and the Fuzzy **composition of functions**. With the completion of this chapter readers are ready to solve fuzzy rule-bases according to the **Mamdani model.**

We have stated previously that variables in the real world are related so change in one or more affect the behaviour or values of another. This situation defines a MISO system (Multiple Inputs Single Output). An example of such system is presented in the following illustration:



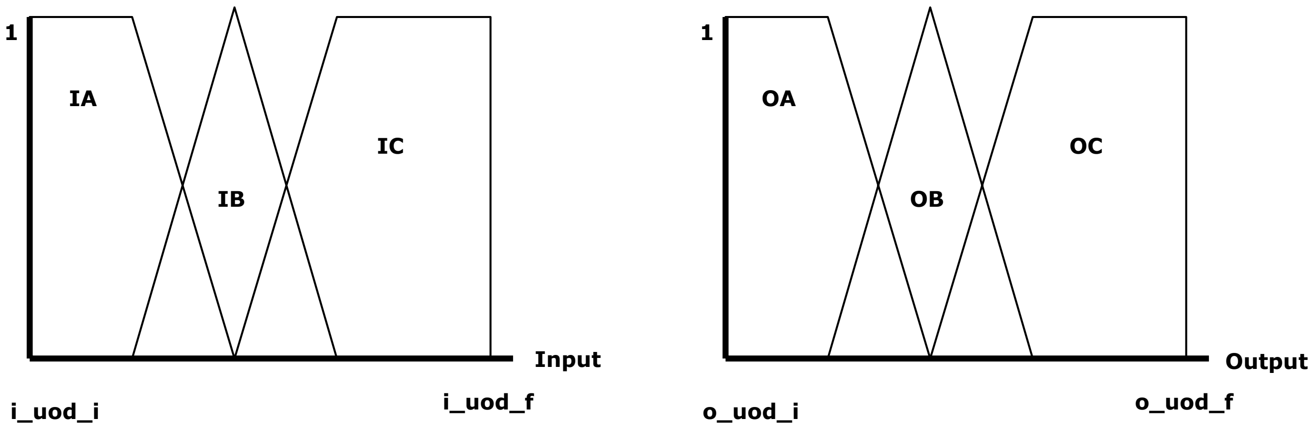
Output Variable $U_e$ is related to Input variables (***e*****,&**)

This relation ***R*** defines the knowledge inherent to the system, being independent of particular data that might come later.

Input Variables:

- Error ***e***, Time Differencial of error **&**

The mathematical manner of expressing the fuzzy relation(s) between two variables (input – output) is done through the cartesian product of two given fuzzy sets, expressed as ***R***.

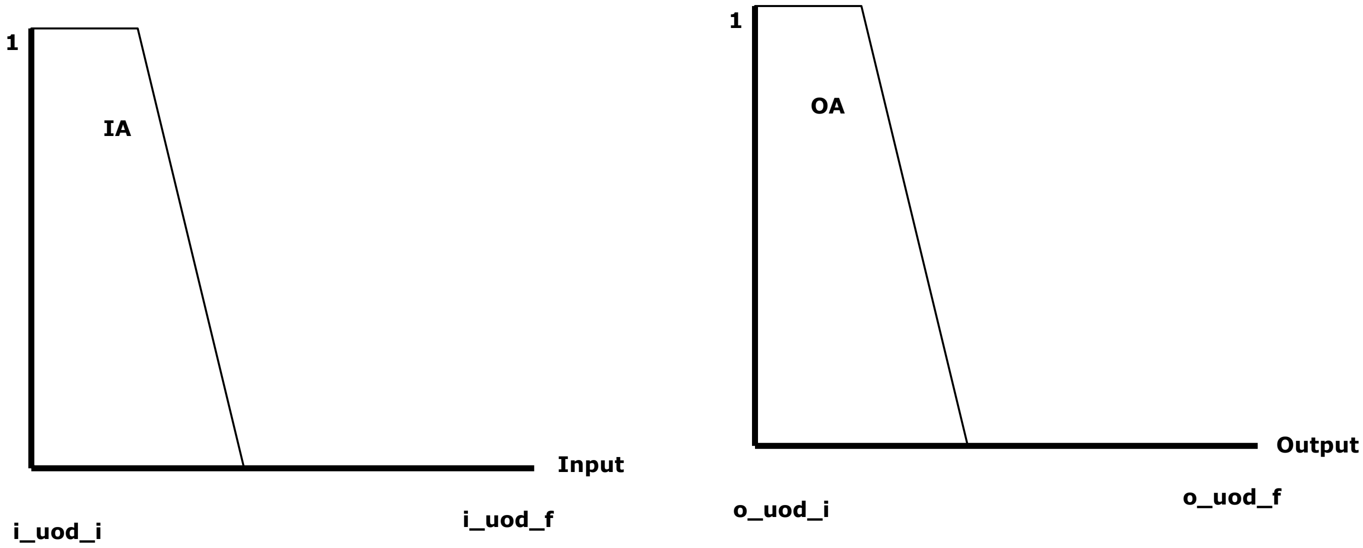


The OA region depends on the IA region

$R$ is a relation found through Cartesian Product

$R$ defines how OA is affected by IA

Let us take fuzzy sets IA (input variable) and OA (output variable) as depicted in the following illustration.



$R$ is a relationship found through Cartesian Product

IA = $\{(i_1, dom_1), (i_2, dom_{i2})\ldots(i_n, dom_{in})$

OA = $\{(o_1, dom_{o1}), (o_2, dom_{o2})\ldots(o_n, dom_{om})\}$

dom := degree of membership

We find the Cartesian Product ***R*** by accommodating a matrix in which rows define the real-worl input values and columns represent the real-world output values. However, we are only interested in the way that the actual degrees of membership for input and output are related.

Therefore...

Given two fuzzy sets

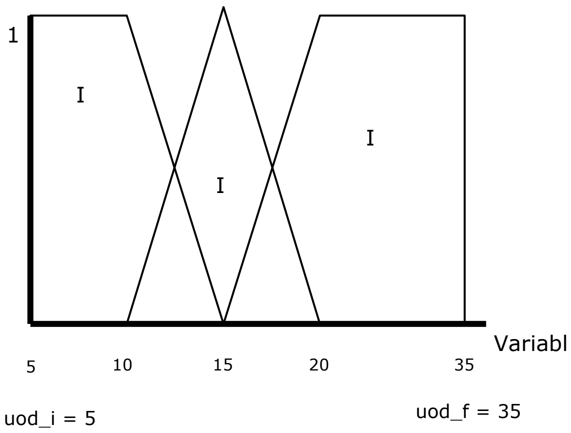
IA = $\{(i_1, dom_{i1}), (i_2, dom_{i2})...(i_n, dom_{in})\}$ and

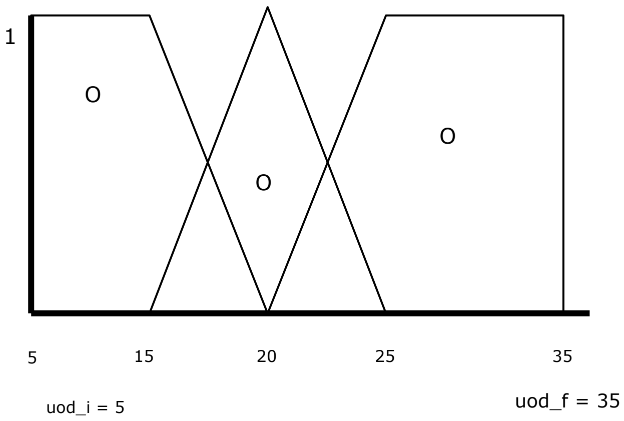
OA = $\{(o_1, dom_{o1}), (o_2, dom_{o2})...(o_n, dom_{om})\}$, ***R*** is:

$$\begin{array}{cc|ccc} & & & OA & \\ & & o_1 & o_2 & o_m \\ \hline & i_1 & \min(dom_{i1}, dom_{o1}) & & \min(dom_{i1}, dom_{om}) \\ & i_2 & & & \\ IA & \dots & & \boldsymbol{R} & \\ & i_n & & & \end{array}$$

Each entry of matix ***R*** is the minimum value of the degree of membership of input and output.

Let us present and example. The reader is encouraged to do the maths him/herself.





Calculate ***R*** = IA X OB

On the former figure,

IA = {(5,1)(6,1)(7,1)(8,1)(9,1)(10,1)(11,0.8)(12,0.6)(13,0.4)(14,0.2)(15,0)}
OA = {(5,1)(6,1)(7,1)(8,1)(9,1)(10,1)(11,1)(12,1)(13,1)(14,1)(15,1)(16,0.8)(17,0.6)(18,0.4)(19,0.2)(20,0)}

Therefore:

***R*** = IA **x** OA

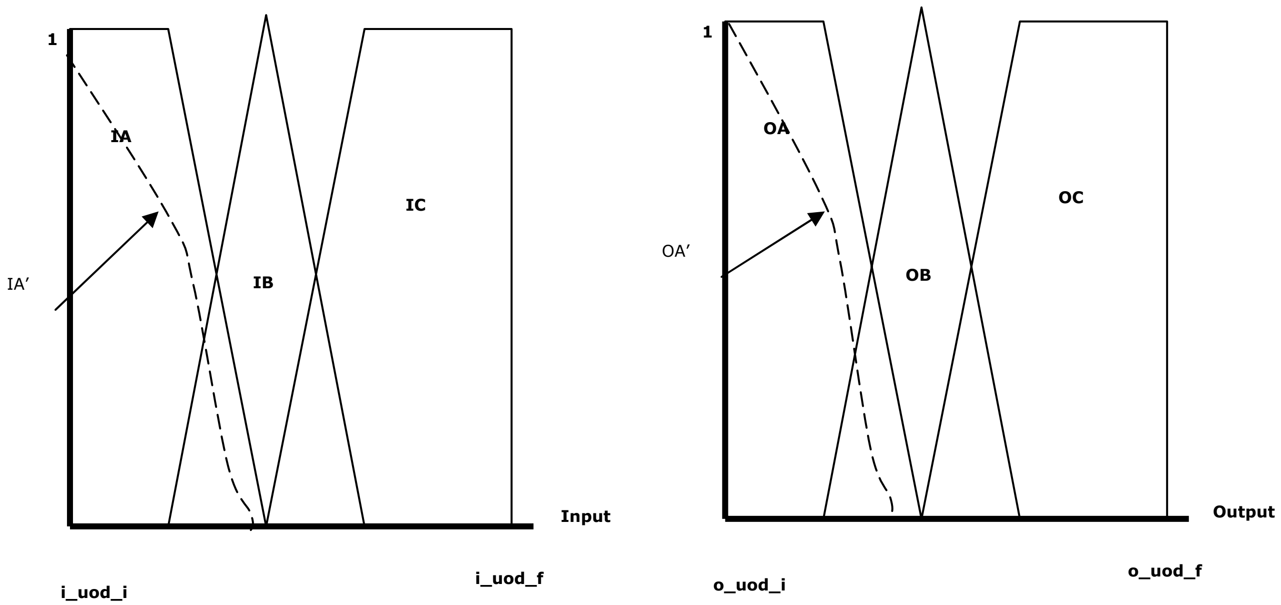
| | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0.8** | **0.6** | **0.4** | **0.2** | **0** |
| 6 | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0.8** | **0.6** | **0.4** | **0.2** | **0** |
| 7 | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0.8** | **0.6** | **0.4** | **0.2** | **0** |
| 8 | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0.8** | **0.6** | **0.4** | **0.2** | **0** |
| 9 | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0.8** | **0.6** | **0.4** | **0.2** | **0** |
| 10 | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0.8** | **0.6** | **0.4** | **0.2** | **0** |
| 11 | **0.8** | **0.8** | **0.8** | **0.8** | **0.8** | **0.8** | **0.8** | **0.8** | **0.8** | **0.8** | **0.8** | **0.8** | **0.6** | **0.4** | **0.2** | **0** |
| 12 | **0.6** | **0.6** | **0.6** | **0.6** | **0.6** | **0.6** | **0.6** | **0.6** | **0.6** | **0.6** | **0.6** | **0.6** | **0.6** | **0.4** | **0.2** | **0** |
| 13 | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.4** | **0.2** | **0** |
| 14 | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0.2** | **0** |
| 15 | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |

Take any entry of the former matrix ***R***, and you will notice that the value is the minimun of the corresponding membership functions for that chosen entry.

The most important thing about ***R*** is that it represents the knowledge of the system, even though such knowledge were adquired my taking measures and experimentation.

### 3.2 Composition of Functions

We now move to another important operation, called the min-max composition. For that, we are going to require the Cartesian Product.



We have now -> ***R*** = IA X OB

We introduce another input set similar to IA

This new set is labeled IA’

We must obtain the correspondent output set OA’

The former figure illustrates a nice situation. We have already partitioned both input and output variables into regions (IA, IB, IC, OA, OB, OC). The model is not always what appears as an actual value when the system works (but it should respond to that). The actual input value is represented by IA’, and the taks is to find the actual output value OA’, taking into account that IA and OA are related via matrix ***R*** previously calculated.

This is done with the min-max composition.

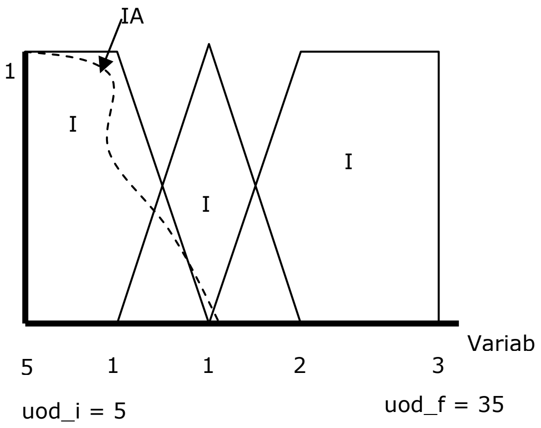
IA’ is a vector, and ***R*** is a matrix. It is straight forward to calculate, in the fuzzy world, the multiplication of vector IA’ with matrix ***R***.

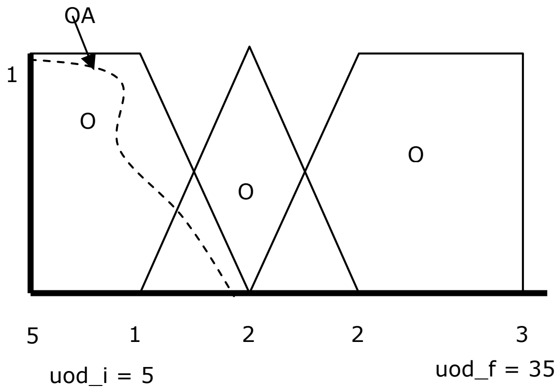
It’s just that some changes are applied to the classical operation.

1. Multiplication is now the minimun
2. Addition is replaced by the maximum

...of the degrees of membership.

We present an example. Then again, readers are encouraged to do the maths.





IA = {(5,1)(6,1)(7,1)(8,1)(9,1)(10,1)(11,0.8)(12,0.6)(13,0.4)(14,0.2)(15,0)}

IA′ = {(5,.9)(6,.9)(7,.8)(8,.8)(9,.8)(10,.7)(11,0.7)(12,0.5)(13,0.3)(14,0.1)(15,0)}

***R***

A partial calculation is:

OA′ = IA′ · ***R***

[0.9, 0.9, 0.8, 0.8, 0.8, 0.7, 0.7, 0.5, 0.3, 0.1,0]

|  | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0.8 | .6 | .4 | .2 | 0.0 |
| 6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 9 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 10 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 11 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 12 | .6 | .6 | .6 | .6 | .6 | .6 | .6 | .6 | .6 | .6 | .6 | .6 | .6 | .4 | .2 | 0.0 |
| 13 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 15 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

IA′

***R***

The entire calculation leads to finding OA':

Given the relationship ***R*** (already calculated), and given the input set IA' defined as follows,

IA' = {0.9, 0.9, 0.8, 0.8, 0.8, 0.7, 0.7, 0.5, 0.3, 0.1,0},

we proceed to calculate the first element of the composition:

OA’ [1] = [max{min(0.9,1), min(0.9,1), min(0.8,1), min(0.8,1), min(0.8,1), min(0.7,1), min(0.7,0.8), min(0.5, 0.6), min(0.3, 0.4), min(0.1, 0.2), min(0,0)}]

OA’ [1] = [max{0.9, 0.9, 0.8, 0.8, 0.8, 0.7, 0.7, 0.5, 0.3, 0.1, 0}]

OA’ [1] = [0.9]

By applying the same procedure we find the resultant output value OA’:

OA’ = {0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.8, 0.6, 0.4, 0.2, 0}

### 3.3 Generalized Modus Ponens

With the operations of chapter two and three, we have laid out the foundations to develop and solve fuzzy rulebases according to the Mamdani model.

The Mamdani model, by the way, is based on the Generalized Principle of Extension:

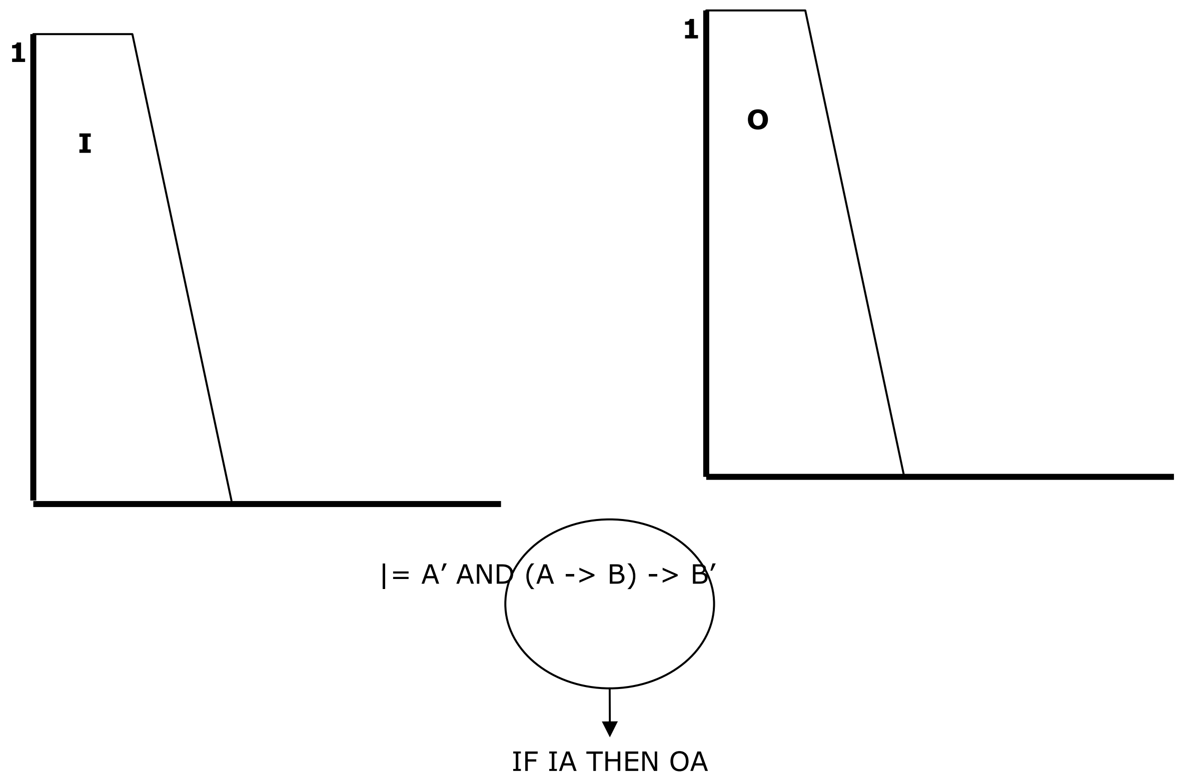
|= A’ AND (A -> B) -> B’
B’ = A’ · (A X B)

Just what he have done!!!

# Chapter 4. Fuzzy rule bases

### 4.1 Fuzzy rule bases

We have previously seen that real-world variables are related, that is to say, what happens in one region of the input reflects a corresponding value in the output.



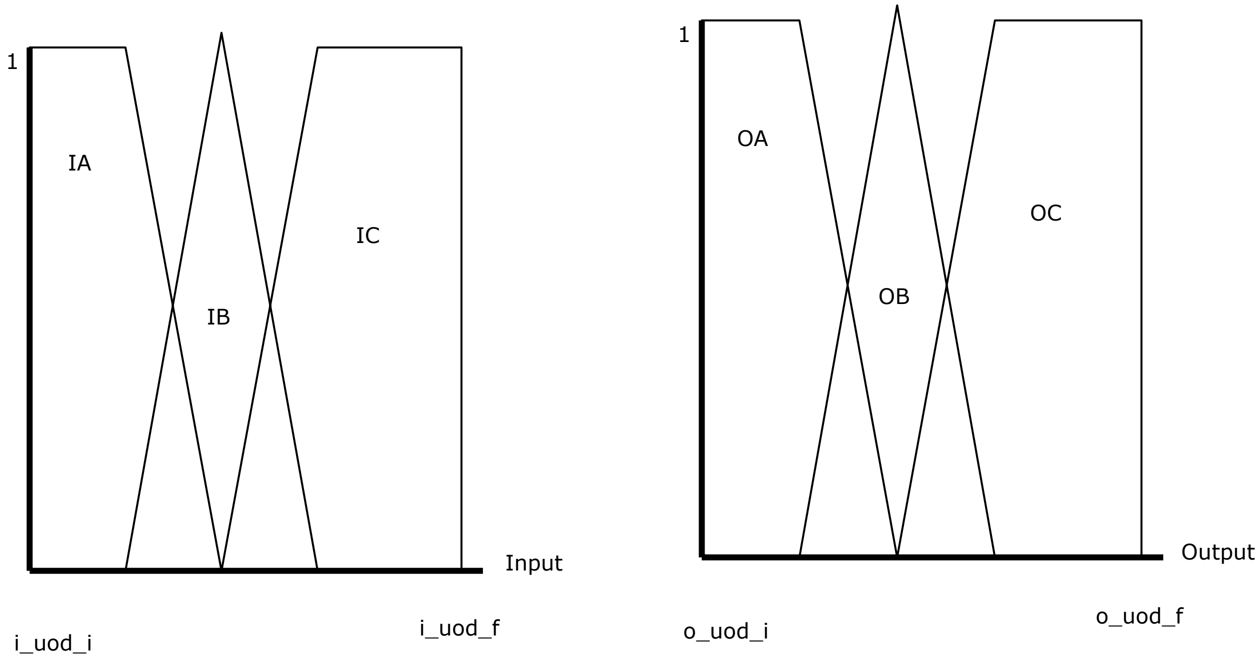
One Input one Output

one Input Region one Output Region

### 4.2 The Mamdani model

According to Mamdani, this relation can be expressed in terms of rules, like it is shown in the above figure.

However, things are more complicated, because every variable has more than one region, as exemplified next:

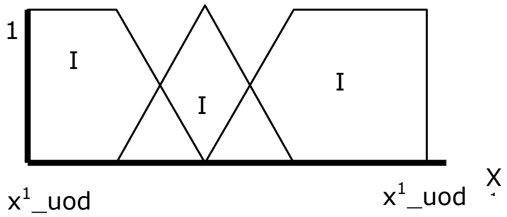


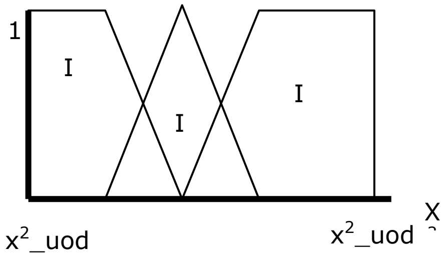
IF IA THEN OA
IF IB THEN OB
IF IC THEN OC

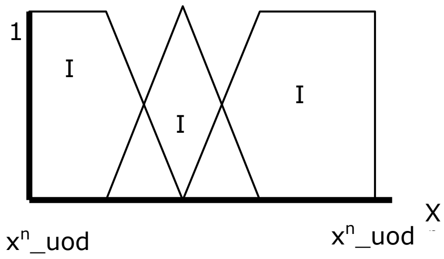
Simple Fuzzy Rule-Base

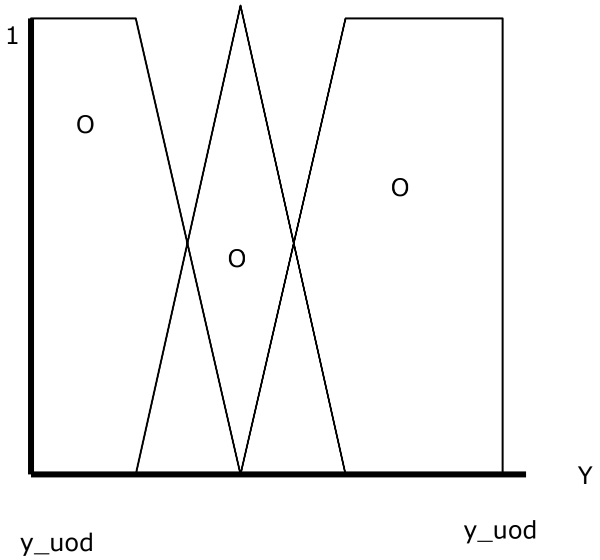
A simple fuzzy rule base represents the relations that are formed between input and output regions.

Things, nevertheless, can get more complicated, because a given output variable depends on more than one input variable. This type of systems are called MISO (Multiple Inputs Single Output), as it is depicted next.









IF IA$^1$ AND IA$^2$··· AND IA$^n$ THEN OA
IF IB$^1$ AND IB$^2$··· AND IB$^n$ THEN OB
IF IC$^1$ AND IC$^2$ ··· AND Ic$^n$ THEN OC

Therefore, given a rule base as follows:

Rule1: IF x1 is IA1 AND x2 is IA2··· AND xn is IAn THEN OA
Rule2: IF x1 is IB1 AND x2 is IB2··· AND xn is IBm THEN OB
...
Rulen: IF x1 is IC1 AND x2 is IC2 ··· AND xn is Icm THEN OC

It is necessary to split ever rule to solve the puzzle:

***R***11 = IA1 x OA
***R***12 = IA2 x OA
***R***1m = IAm x OA
***R***21 = IB1 x OB
***R***22 = IB2 x OB
***R***2m = IBm x OB
***R***n1 = IC1 x OC
***R***n2 = IC2 x OC
***R***nm = ICm x OC

Once we have calculated all ***R***’s, then we must solve the rule base by using the min-max composition on every ***R*** for every rule. This causes a trememdous burden.

Instead, mathematicians have come up with a simple algoritm to solve fuzzy rulebases according to the Mamdani model.

### 4.3 Algorithm to resolve Mamdani fuzzy rulebases

**Mamdani algorithm:**

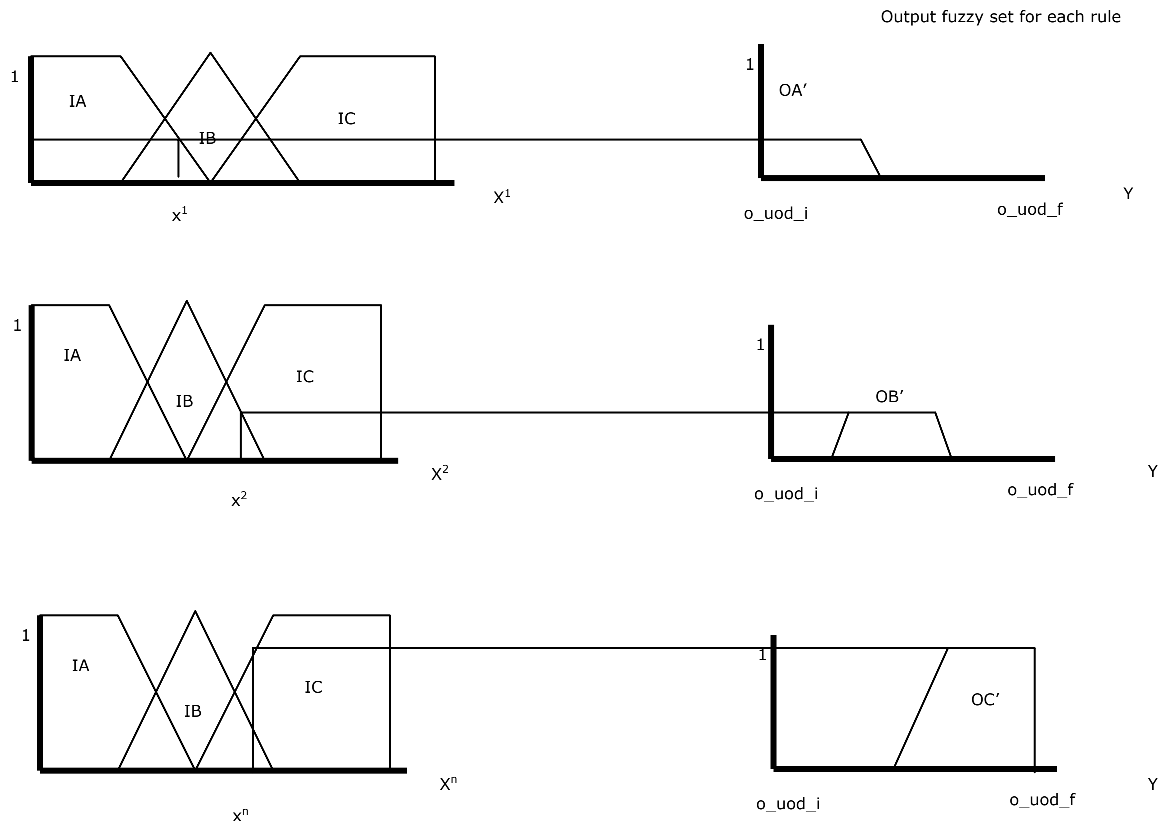
1. For each rule ***j***, compute the degree of fulfilment ***BETAj***:
***BETAj*** = ***mu***(***x1***) AND ***mu***(***x2***) ··· AND ***mu***(***xn***)

2. For each rule ***j***, derive the output fuzzy set ***B’j***:
***B’j*** = ***BETAj*** AND **mu*BJ***(***y***) for all ***y*** in output

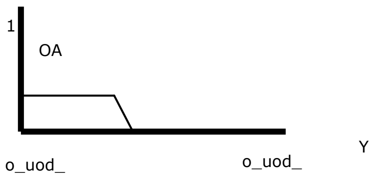
3. Compute the aggregated fuzzy set ***B’***:
***B’*** = UNION(***B’j***) for all ***y*** in output

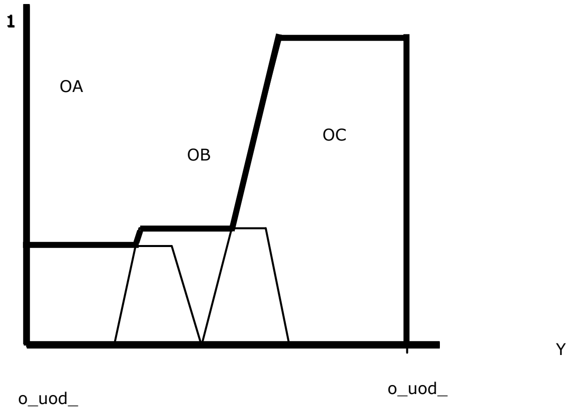
The Mamdani algorithm can be represented graphically:

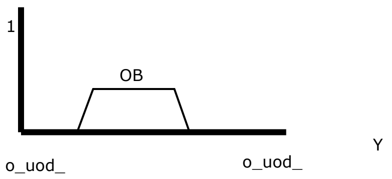
1,2. The degree of fulfilment will give a partial output set:



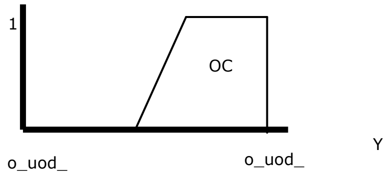
3. All the partial output sets are aggregated by using the UNION operator.







Aggregated



### 4.4 Defuzzification: Center of Gravity

Finally, the output crisp value is obtained via, among other defuzzitication methods, the center of gravity.

Center of Gravity.
cog = [SUM{***y·mu***(y)} / SUM {***mu***(***y***)}

For all ***y*** in **Y**

What we get, as stated previously, is:

Solution:

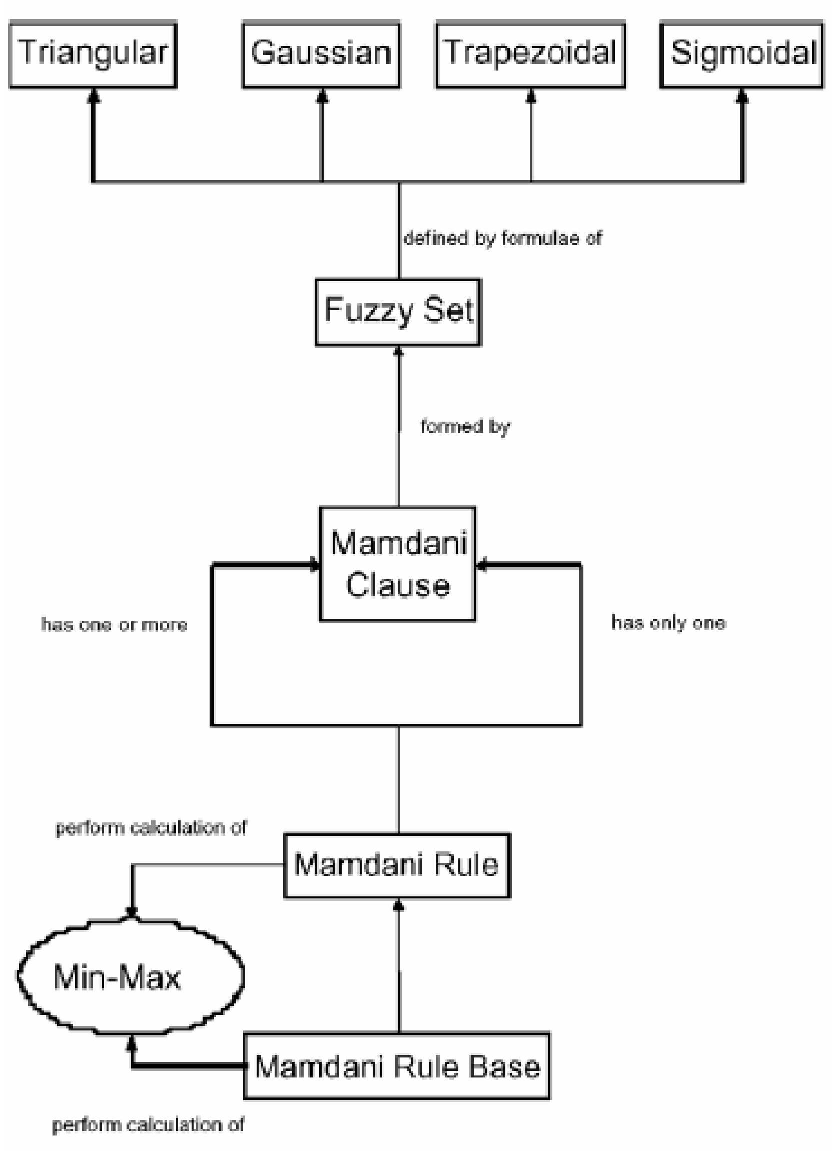
Center of Gravity of the Aggregated Set

# Chapter 5. Constructing fuzzy rule bases with JFK

### 5.1 General structure of JFK

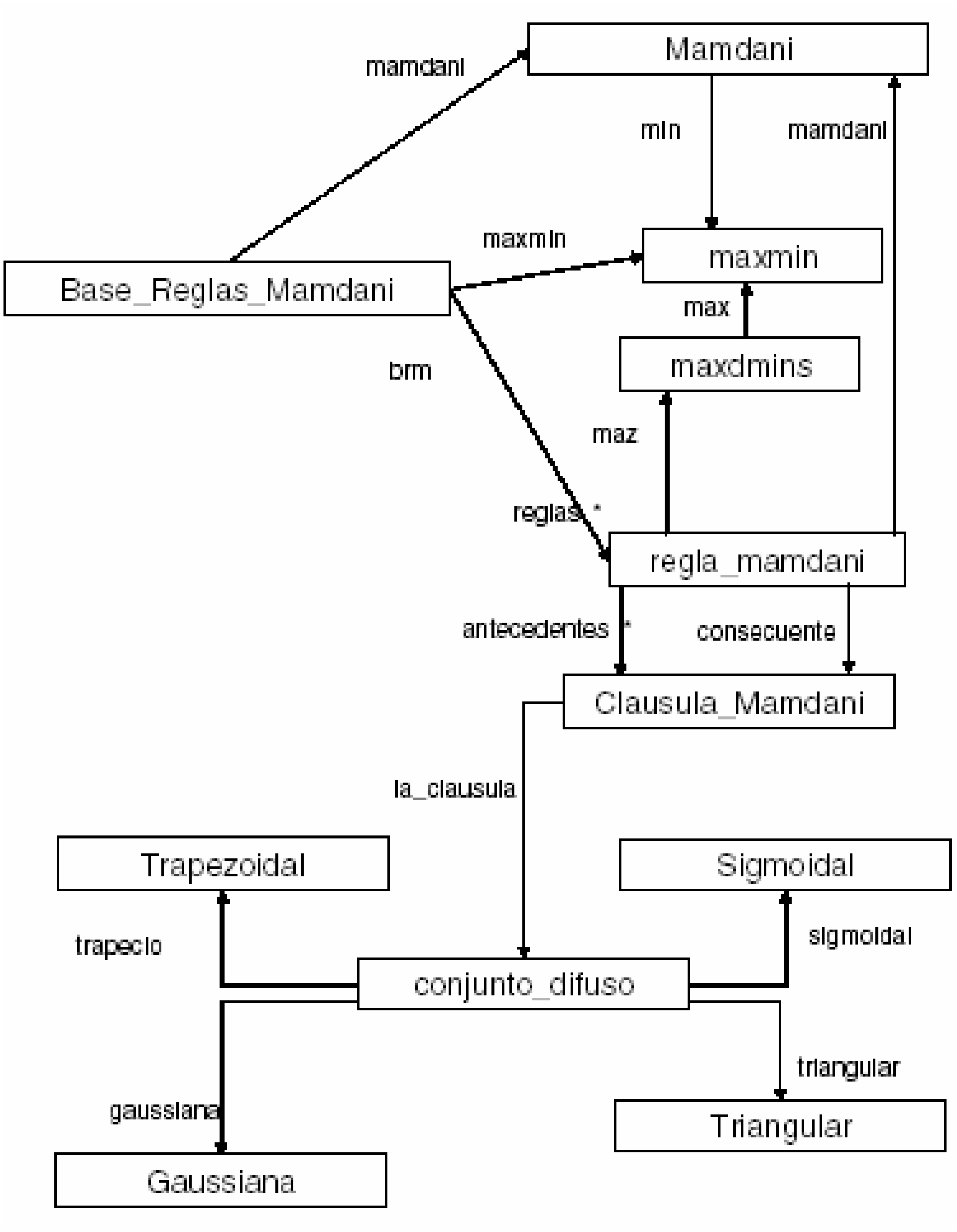
Fuzzy logic is a poweful tool, yet it requires many operations. Thus, it is well suited to be automated on software tools. On this line of reasoning, Prof. Omar López has come up with a shell to build fuzzy rulebases on the Mamdani model. This tool is called Java Fuzzy Kit[2], as it is written in the Java programming language.

The general structure of JFK is as follows:



[2] Omar López-Ortega, Java Fuzzy Kit (JFK): A shell to build fuzzy inference systems according to the generalized principle of extension; Expert Systems with Applications; vol. 34, p. 796 – 804, January 2008.

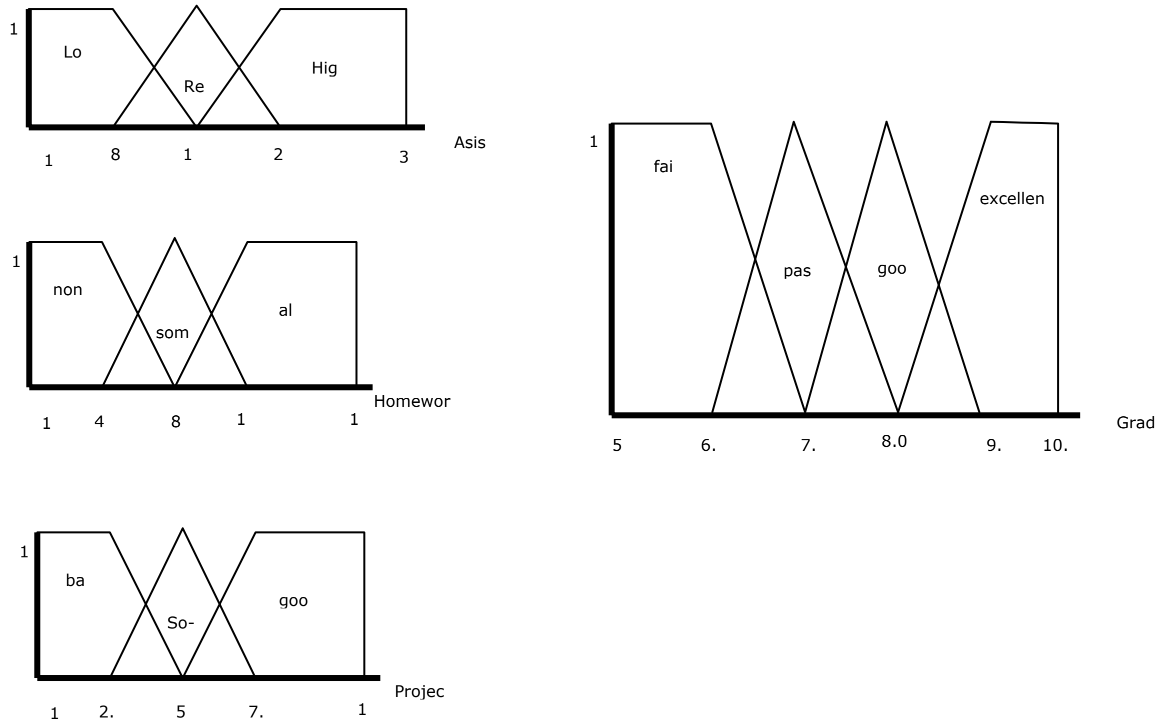
The class diagram of JFK is presented next:



We present an example in order to see the implementation of fuzzy rule bases. In the example we wish to calculate students's final grades according to three variables: Number of assistances in class, number of completed and correct homeworks, and final project.

## 5.2 **Modelling a fuzzy rule base**

The model is presented next:



An excerpt of the rule base is given in the following lines:

```
Rule1: IF asist is LOW AND hw is NONE AND project is BAD THEN grade is FAIL
Rule2: IF asist is REG AND hm is SOME AND project is SO-SO THEN grade is PASS
Rule3: IF asist is HIGH AND hm is all AND project is SO-SO THEN grade is A-
EXCELLENT
Rule4:  IF  asist  is  HIGH  AND  hm  is  ALL  and  project  is  GOOD  THEN  grade  is
EXCELLENT
```

## **5.3 Usage of JFK1.0**

The entire rule base, codified on JFK is presented in the following source code:

1. Imports and Declaration of JFK variables

```
import mx.edu.uaeh.jfk.BaseReglasMamdani;
import mx.edu.uaeh.jfk.VariableMamdani;
import mx.edu.uaeh.jfk.funciones.*;
```

```
public BaseReglasMamdani brmGrades;//
public VariableMamdani asist; //asistencies
public VariableMamdani homework;//
public VariableMamdani project;//
public VariableMamdani grade;//
```

2. An Instance of BaseReglasMamdani is constructed:

```
brmGrades = new BaseReglasMamdani("Grades");
```

3. Each variable is declared

```
asist = new VariableMamdani("asist", 1, 32);//UoD asist
//method agregarClausula is employed
asist.agregarClausula(new Trapezoidal(1, 1, 8, 16), "low");
asist.agregarClausula(new Triangular(8, 16, 24), "reg");
asist.agregarClausula(new Trapezoidal(16, 24, 32, 32),"high");
```

4. Variables are added to the rule-base

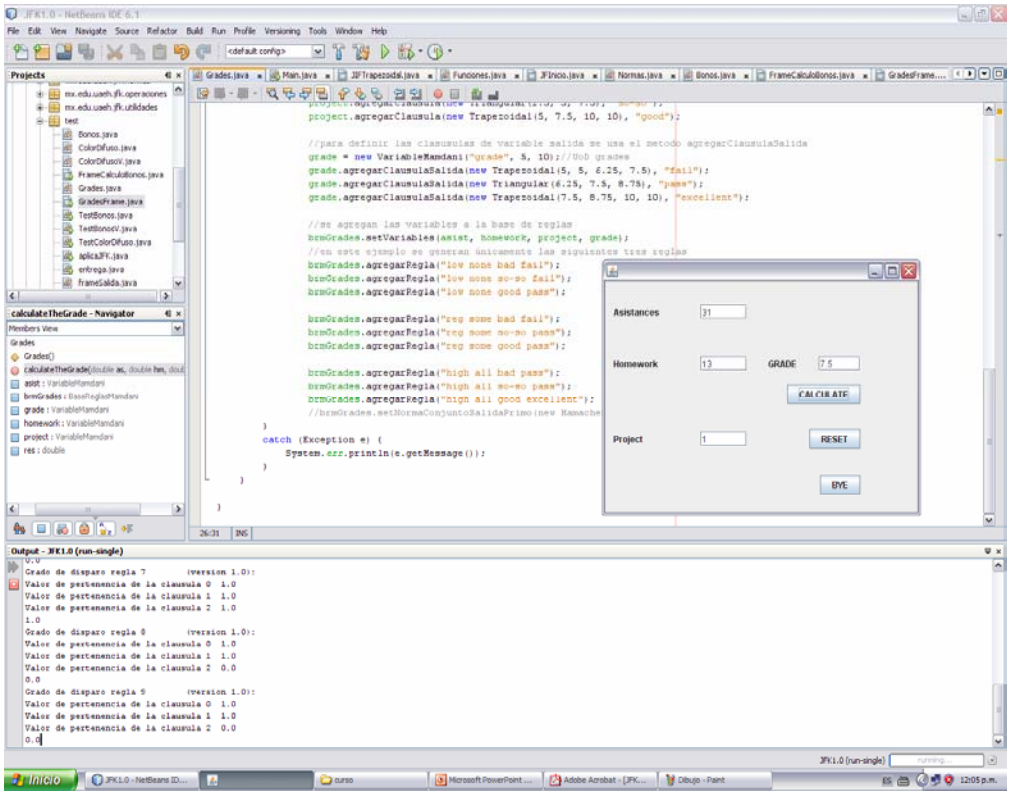
```
brmGrades.setVariables(asist, homework, project, grade);

5. Rules are constructed
brmGrades.agregarRegla("low none bad fail");
brmGrades.agregarRegla("low none so-so fail");
brmGrades.agregarRegla("low none good pass");

brmGrades.agregarRegla("low some bad fail");
brmGrades.agregarRegla("low some so-so pass");
brmGrades.agregarRegla("low some good pass");

brmGrades.agregarRegla("low all bad fail");
brmGrades.agregarRegla("low all so-so pass");
brmGrades.agregarRegla("low all good pass");

brmGrades.agregarRegla("reg none bad fail");
brmGrades.agregarRegla("reg none so-so fail");
brmGrades.agregarRegla("reg none good pass");

brmGrades.agregarRegla("reg some bad fail");
brmGrades.agregarRegla("reg some so-so pass");
brmGrades.agregarRegla("reg some good pass");

brmGrades.agregarRegla("reg all bad pass");
brmGrades.agregarRegla("reg all so-so pass");
brmGrades.agregarRegla("reg all good pass");

brmGrades.agregarRegla("high none bad fail");
brmGrades.agregarRegla("high none so-so pass");
brmGrades.agregarRegla("high none good pass");
brmGrades.agregarRegla("high some bad pass");
brmGrades.agregarRegla("high some so-so pass");
brmGrades.agregarRegla("high some good pass");
brmGrades.agregarRegla("high all bad pass");
brmGrades.agregarRegla("high all so-so almost-excellent");
brmGrades.agregarRegla("high all good excellent");
```

5. Result is catched

```
double res = 0;
res = brmGrades.getResultadoNew();
```

## 5.4 A Java example

The following screen-shots show an application running on JFK.


JFK1.0 - NetBeans IDE 6.1
File Edit View Navigate Source Refactor Build Run Profile Versioning Tools Window Help
Grades.java
Main.java
JIFTrapezoidal.java
Funciones.java
JFInicio.java
Normas.java
Bonos.java
FrameCalculoBonos.java
GradesFrame....
calculateTheGrade - Navigator
Output - JFK1.0 (run-single)
Inicio

```
        project.agregarClausula(new Trapezoidal(5, 7.5, 10, 10), "good");

        //para definir las clausulas de variable salida se usa el metodo agregarClausulaSalida
        grade = new VariableMamdani("grade", 5, 10);//UoD grades
        grade.agregarClausulaSalida(new Trapezoidal(5, 5, 6.25, 7.5), "fail");
        grade.agregarClausulaSalida(new Triangular(6.25, 7.5, 8.75), "pass");
        grade.agregarClausulaSalida(new Trapezoidal(7.5, 8.75, 10, 10), "excellent");

        //se agregan las variables a la base de reglas
        brmGrades.setVariables(asist, homework, project, grade);
        //en este ejemplo se generan únicamente las siguientes tres reglas
        brmGrades.agregarRegla("low none bad fail");
        brmGrades.agregarRegla("low none so-so fail");
        brmGrades.agregarRegla("low none good pass");

        brmGrades.agregarRegla("reg some bad fail");
        brmGrades.agregarRegla("reg some so-so pass");
        brmGrades.agregarRegla("reg some good pass");

        brmGrades.agregarRegla("high all bad pass");
        brmGrades.agregarRegla("high all so-so pass");
        brmGrades.agregarRegla("high all good excellent");
        //brmGrades.setNormaConjuntoSalidaPrimo(new Hamache
    }
    catch (Exception e) {
        System.err.println(e.getMessage());
    }
  }

}
```

Asistances 31

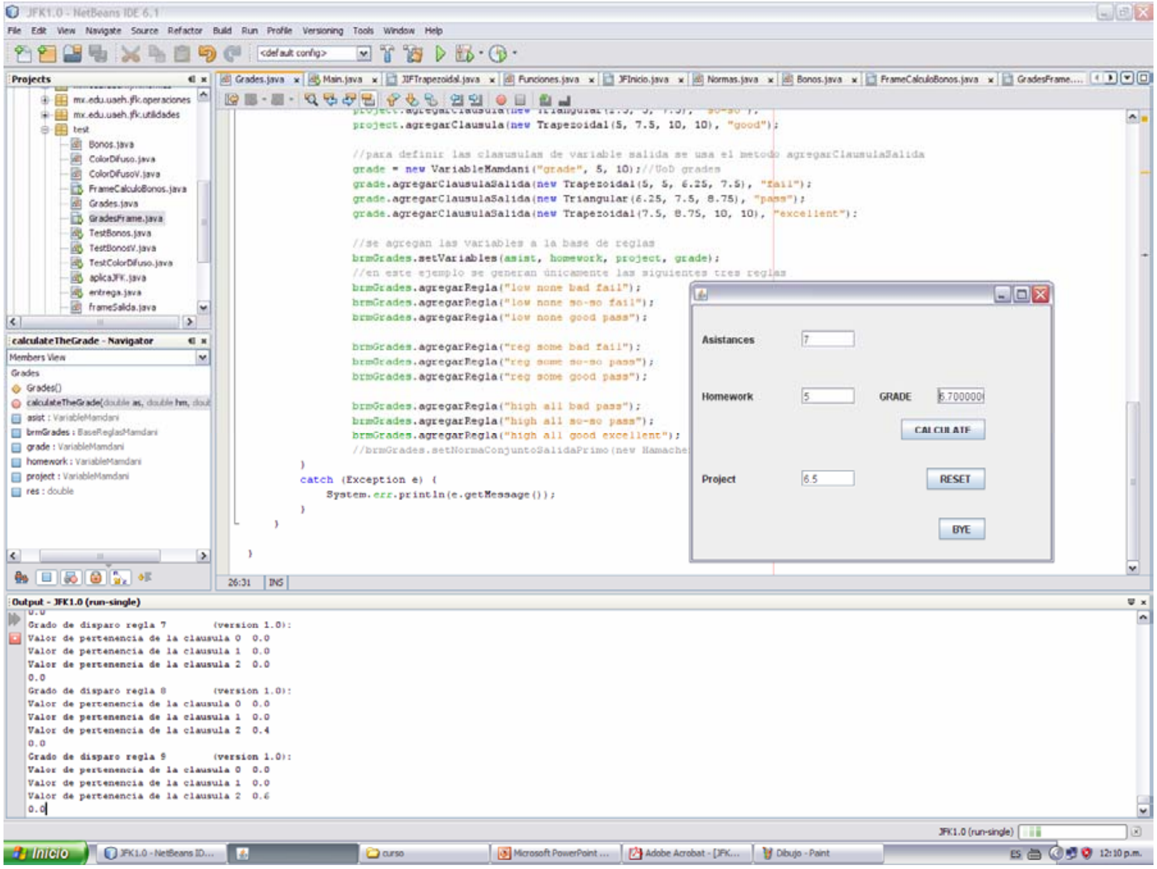
Homework 13 GRADE 7.5

CALCULATE

Project 1 RESET

BYE

```
Grado de disparo regla 7        (version 1.0):
Valor de pertenencia de la clausula 0  1.0
Valor de pertenencia de la clausula 1  1.0
Valor de pertenencia de la clausula 2  1.0
1.0
Grado de disparo regla 8        (version 1.0):
Valor de pertenencia de la clausula 0  1.0
Valor de pertenencia de la clausula 1  1.0
Valor de pertenencia de la clausula 2  0.0
0.0
Grado de disparo regla 5        (version 1.0):
Valor de pertenencia de la clausula 0  1.0
Valor de pertenencia de la clausula 1  1.0
Valor de pertenencia de la clausula 2  0.0
0.0
```



```
            project.agregarClausula(new Trapezoidal(5, 7.5, 10, 10), "good");

            //para definir las clasusulas de variable salida se usa el metodo agregarClausulaSalida
            grade = new VariableMamdani("grade", 5, 10);//UoD grades
            grade.agregarClausulaSalida(new Trapezoidal(5, 5, 6.25, 7.5), "fail");
            grade.agregarClausulaSalida(new Triangular(6.25, 7.5, 8.75), "pass");
            grade.agregarClausulaSalida(new Trapezoidal(7.5, 8.75, 10, 10), "excellent");

            //se agregan las variables a la base de reglas
            brmGrades.setVariables(asist, homework, project, grade);
            //en este ejemplo se generan únicamente las siguientes tres reglas
            brmGrades.agregarRegla("low none bad fail");
            brmGrades.agregarRegla("low none so-so fail");
            brmGrades.agregarRegla("low none good pass");

            brmGrades.agregarRegla("reg some bad fail");
            brmGrades.agregarRegla("reg some so-so pass");
            brmGrades.agregarRegla("reg some good pass");

            brmGrades.agregarRegla("high all bad pass");
            brmGrades.agregarRegla("high all so-so pass");
            brmGrades.agregarRegla("high all good excellent");
            //brmGrades.setNormaConjuntoSalidaPrimo(new Hamache
        }
        catch (Exception e) {
            System.err.println(e.getMessage());
        }
    }

}
```

Asistances 7

Homework 5

GRADE 6.700000

Project 6.5

Output - JFK1.0 (run-single)

```
0.0
Grado de disparo regla 7        (version 1.0):
Valor de pertenencia de la clausula 0  0.0
Valor de pertenencia de la clausula 1  0.0
Valor de pertenencia de la clausula 2  0.0
0.0
Grado de disparo regla 8        (version 1.0):
Valor de pertenencia de la clausula 0  0.0
Valor de pertenencia de la clausula 1  0.0
Valor de pertenencia de la clausula 2  0.4
0.0
Grado de disparo regla 9        (version 1.0):
Valor de pertenencia de la clausula 0  0.0
Valor de pertenencia de la clausula 1  0.0
Valor de pertenencia de la clausula 2  0.6
0.0
```

The GUI is intended so a given user enters the values for the referred variables, and JFK performs the calculation to obtain a final grade.

Examples like this are many, but it suffices for the purposes of this technical report.

# Conclusions

Fuzzy logic is a powerful tool to compute complex relations among variables ocurring in the real world. Even more so, the mathematical procedures and algorithms are not thought to be solved hands-on-paper but by using programming languages. In this sense, Fuzzy Logic is of great use to those interested in modelling and solve complex systems.

The present technical report is intended to provide explanations about the foundations of fuzzy logic. Also, it explains how to use a tool called JFK where fuzzy rule bases are constructed.

Topics ranging from membership functions, operations on fuzzy sets, triangular norms and co-norms, cartesian product, fuzzy composition, are the set of core mathematical background needed to understand the “logic” part of “fuzzy” systems.

The Generalized Modus Ponens was also covered. It is the link between maths and logic. Soon after, fuzzy rule bases are easy to grasp. With JFK, implementation is carried fluently.