Técnicas, Entornos y Aplicaciones de Inteligencia Artificial

Práctica 1. FUZZY-CLIPS

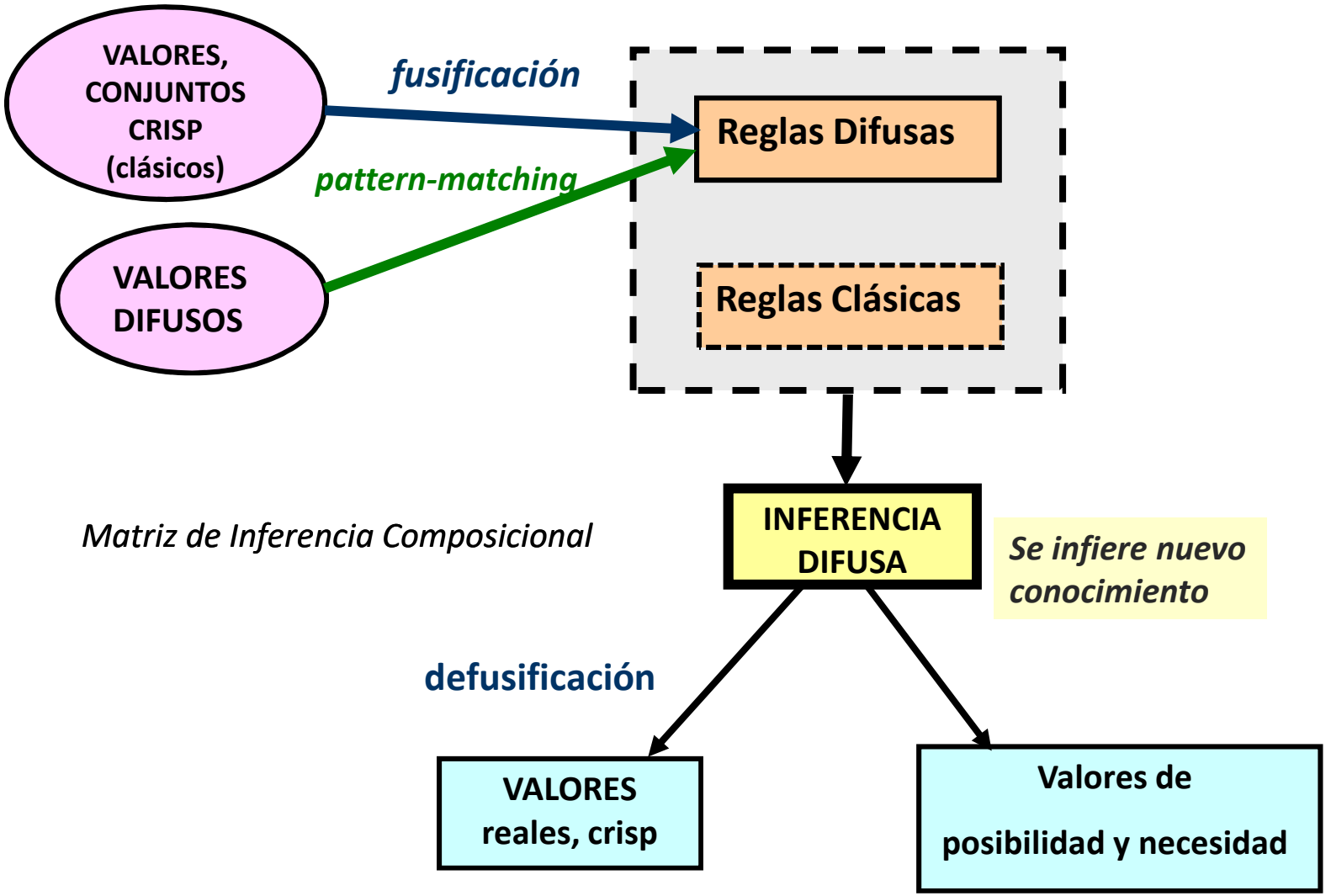
Objetivo:

utilizar FUZZY-CLIPS para la resolución de un problema donde

hay que aplicar un razonamiento difuso basado en reglas

FuzzyClips está disponible en el poliformat
Instalación: Copiar los archivos correspondientes en una misma carpeta

Práctica 1: FUZZY-CLIPS

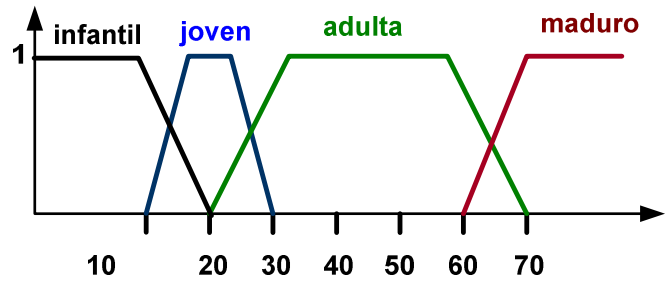


Definición de variables difusas:

Se define la variable y sus valores difusos

```
(deftemplate edad              ;Variable difusa
      0 120 años               ;Universo
    ( (infantil (12 1) (20 0))  ;Valores difusos
      (joven (10 0) (15 1) (25 1) (30 0))
      (adulta (20 0) (30 1) (60 1) (70 0))
      (mayor (60 0) (70 1)))
 )
```

Incluso con modificadores lingüísticos:
(viejo plus mayor)




```
(deftemplate edad    ;Variable difusa
     0 120 años      ;Universo
    ( (infantil (12 1) (20 0))  ;Valores difusos
      (joven (10 0) (15 1) (25 1) (30 0))
      (adulta (20 0) (30 1) (60 1) (70 0))
      (mayor (60 0) (70 1))))
```

```
(deftemplate estatura  0 250 cm
( (bajo  (0 1) (100 1) (150 0))
  (medio  (100 0) (150 1) (170 1) (180 0))
  (alto   (170 0) (180 1))))
```

```
(deftemplate numero   0 10 unit
    ( (tres (3 0) (3 1) (3 0))
      (cinco (5 0) (5 1) (5 0))))
```

Hechos Difusos: Asercion de valores difusos a variables difusas

(requiere la definición previa de las variables difusas)

(deffacts ejemplo
(edad adulta) (estatura very bajo))

```
f-1     (edad adulta) CF 1.00
f-2     (estatura very bajo) CF 1.00 0.0 0.0)  )
( (100.0 1.0) (105.0 0.81) (110.0 0.64) (115.0 0.49) (120.0 0.36)
  (125.0 0.25) (130.0 0.16) (135.0 0.09) (140.0 0.04) (145.0 0.01)
  (150.0 0.0)  )
```

(assert (edad joven))
(assert (estatura alto))

```
f-4     (estatura [ very bajo ] OR [ alto ]) CF 1.00
f-5     (edad [ [ adulta ] OR [ joven ] ] OR [ mayor OR joven ]) CF 1.00
( (10.0 0.0) (15.0 1.0) (25.0 1.0) (26.67 0.6667) (30.0 1.0) 0.01)
  (60.0 1.0) (65.0 0.5) (70.0 1.0)  )  )
```

(assert (numero tres))
(assert (edad [mayor OR joven]))

Las nuevas aserciones sobre hechos difusos se acumulan!

Definición de reglas difusas

ENTRADA: TEMPERATURA

```
(deftemplate Temp   5 50  Celsius
   ((frio (10 1) (20 0))
   (templado (10 0) (20 1) (25 1) (30 0))
   (calor (25 0) (30 1) ))))
```

SALIDA: APERTURA DE LA VALVULA

```
(deftemplate valvula 0 100 apertura
   ((poco (10 1) (20 0) )
   (medio (10 0) (30 1) (60 1) (70 0))
   (mucho (60 0) (70 1) )))
```

(defrule temperatura_frio (Temp frio) => (assert (valvula mucho)))	(defrule temperatura_buena (Temp templado) => (assert (valvula medio)))	(defrule temperatura_calor (Temp calor) => (assert (valvula poco)))

```
(deffacts  ejemplo
        (Temp very templado))
```

Facts (MAIN)

```
f-0     (initial-fact) CF 1.00
f-1     (Temp very templado) CF 1.00
( (10.0 0.0) (11.0 0.01) (12.0 0.04) (13.0 0.09) (14.0 0.16)
  (15.0 0.25) (16.0 0.36) (17.0 0.49) (18.0 0.64) (19.0 0.81)
  (20.0 1.0) (25.0 1.0) (25.5 0.81) (26.0 0.64) (26.5 0.49)
  (27.0 0.36) (27.5 0.25) (28.0 0.16) (28.5 0.09) (29.0 0.04)
  (29.ÌÌ d+ €ñ
```

Agenda (MAIN)

```
0      temperatura_frio: f-1
0      temperatura_buena: f-1
0      temperatura_calor: f-1
```

Fusificación de valores crisp:

a) Para fusificar un valor concreto podemos definir un valor difuso de tipo *singleton*:

```
(deftemplate edad 0 100 años
    ((joven (10 0) (15 1) (25 1) (30 0))
    (veinticinco  (25 0) (25 1) (25 0)))     ; singleton con valor 25
```

b) O bien, podemos utilizar la función *fuzzify* para fusificar un valor crisp **(definida en el boletín)**:

```
(fuzzify   ?fztemplate   ?value   ?delta)
```

Por ejemplo, si tenemos definida la **variable difusa edad** y la función fuzzify, podemos invocarla como:

```
(fuzzify edad  35  0.1)    y aparecerá el hecho (valor difuso):
                    (edad  (34.9  0.0)  (35  1.0)  (35.1  0.0)))
```

Lectura de valores difusos:

No se pueden asertar valores difusos leídos directamente desde consola *(punto 2.4)*

```
(defrule leerconsola    ; Aserción de un valor difuso leído desde consola
     (initial-fact)
=>
   (printout t "Introduzca la edad: joven, adulta, madura" crlf)
   (bind ?Redad  (read))
   (assert-string  (format nil "(edad %s)" ?Redad)) )
```

```
(defrule leerconsola   ; fusificación de valor crisp leído de consola y aserción del valor difuso
   (initial-fact)
=>
   (printout t "Introduzca la edad en anyos" crlf)
   (bind ?Redad  (read))
   (fuzzify edad ?Redad  0.1))
```

Esta operativa se aplica sobre las variables difusas definidas como hechos ordenados

Práctica 1: FUZZY-CLIPS

Inferencia Difusa:

Se puede elegir entre dos reglas composicionales, Max-min y Max-prod:

(set-fuzzy-inference-type <tipo>)

El lanzamiento del proceso inferencial es igual que en Clisp: (run)

Importante:

En Clips estándar si se aserta un hecho que ya existe no se duplica. Así, las reglas no se vuelven ejecutar sobre un mismo hecho.

- Sin embargo, en un sistema difuso, si se aserta un nuevo valor difuso a un slot, distinto al existente, se combinan ambos valores considerando una combinación OR ($F_{final} = F_a \cup F_b$)
- Por ello, una regla previamente ejecutada sobre este hecho volverá a ejecutarse con la nueva información

Defusificación de Variables Difusas ⇒ Valor Crisp:

Utilizaremos las funciones:

- moment-defuzzify, que aplica el algoritmo del centro de gravedad
- maximum-defuzzify, que aplica la media de máximos

Ejemplo: (bind ?variable-no-difusa (maximum-defuzzify ?variable-difusa))

```
(defrule fuzzy1
  (declare (salience -1))
     (edad ?ed)    ;valor difuso de edad
  => (bind ?e  (maximum-defuzzify ?ed ))
      (printout t  "La edad es " ?e crlf))
```

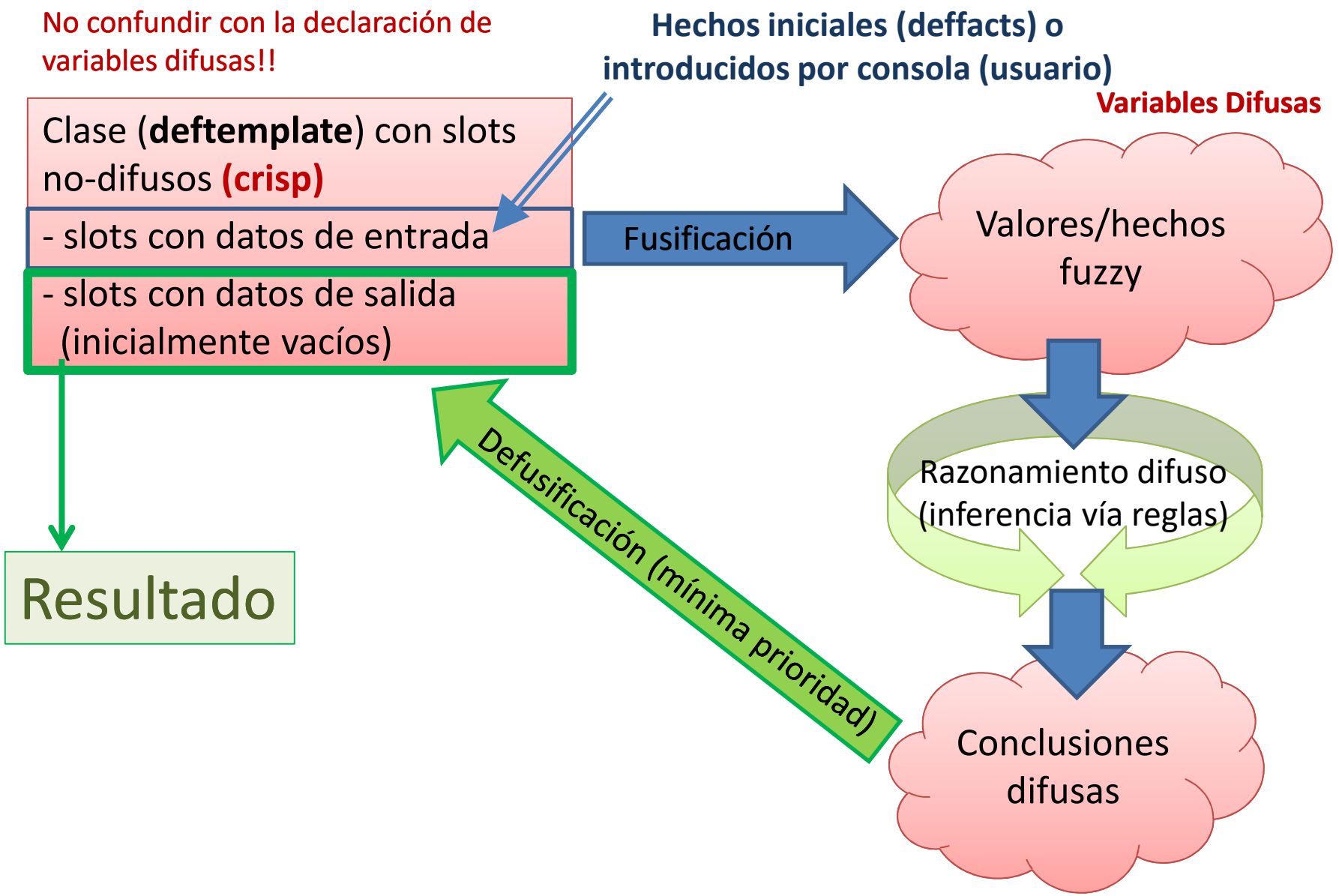
Debe tener la mínima prioridad para ser la última regla en aplicarse, una vez realizado todo el proceso inferencial.

Se defusifica el valor difuso de la variable 'edad', obteniendo su valor crisp

NOTAS: No olvidad el uso de (clear), (reset) y (run).

Ver ejemplos finales en el boletín y manual

Esquema típico



Introducción/inicialización de datos

```
(deffunction proceso ()
   (reset)
   (printout t "Introduzca temperatura del agua: fria, tibia" crlf)
   (bind ?Redad  (read))
   (assert-string  (format nil "(agua %s)" ?Redad))        ;leemos y asertamos un valor difuso

   ;Alternativamente…
   (printout t "Introduzca temperatura en grados" crlf)   ;leemos un valor crisp  y se fusifica (y aserta)
   (bind ?Redad  (read))
   (fuzzify agua ?Redad  0.1)

   (run)
```

```
FuzzyCLIPS> (proceso)
Introduzca temperatura del agua: fria, tibia
tibia
```

Facts (MAIN)

```
(initial-fact) CF 1.00
(agua tibia) CF 1.00
(apertura ???) CF 1.00 .0 1.0) (25.0 0.0)  )
(crisp valvula-maximum 1.333333333333334) CF 1.00
(crisp valvula-moment 1.688888888888889) CF 1.00
```

```
FuzzyCLIPS> (proceso)
Introduzca temperatyura en grados
15
<Fact-1>
```

Facts (MAIN)

```
(initial-fact) CF 1.00
(agua ???) CF 1.00
(apertura ???) CF 1.00 .1 0.0)  )
(crisp valvula-maximum 1.495049504950495) CF 1.00
(crisp valvula-moment 1.759683617086921) CF 1.00
```

En general…. Definición de variables y Reglas difusas. Fusificación y defusificación

```
(deftemplate agua      ;Variable difusa
 0 25 grados           ;Universo
 ((fria (0 1) (10 1) (20 0))
  (tibia (0 0) (10 0) (15 1) (20 1) (25 0))))
```

```
(deftemplate apertura          ;Variable difusa
 0 4 unidades                  ;Universo
 ((poca (0 1) (2 1) (4 0))))
```

```
(deffunction fuzzify (?fztemplate ?value ?delta)
……)
```

```
(defrule regla1   ; entre otras reglas
            (agua fria)
            =>
            (assert (apertura poca)))
```

```
(defrule defuzzificar     ; Regla final para determinar valor exacto de la válvula
(declare (salience -1))  ; mínima prioridad (última regla)
 (apertura ?val)
=>
 (assert (crisp valvula-maximum (maximum-defuzzify ?val)))
 (assert (crisp valvula-moment  (moment-defuzzify  ?val))))
```

;Posteriormente, Introducción (consola, lectura, aserción, deffacts) de los hechos iniciales. (reset), (run)

Definición de clases (templates en Fuzzy-Clips, con valores crisp):

```
(deftemplate persona
  (slot nombre (type SYMBOL))
  (slot edad (type INTEGER))
  (slot vive (type SYMBOL))) ; o tipo INTEGER o FLOAT
```

No utilizar slots difusos

Aserción hecho (instancia del template):

```
(assert     (persona (nombre david) (edad 30)  (vive Valencia)))
```

También se puede asertar como hecho inicial con:

```
(deffacts ejemplo
     (persona (nombre david) (edad 30) (vive Valencia))))
```

Aserción hechos difusos a partir de slots crisp

(deftemplate persona
(slot nombre (type SYMBOL))
(slot edad (type INTEGER))
(slot vive (type SYMBOL))) ; o tipo INTEGER o FLOAT

(deffacts ejemplo
(persona (nombre david) (edad 30) (vive Valencia))))

(dftemplate edad-difusa
0 120 años
((infantil (12 1) (20 0)) *;Valores difusos*
(joven (10 0) (15 1) (25 1) (30 0))
(adulta (20 0) (30 1) (60 1) (70 0))
(mayor (60 0) (70 1))))

Regla:

```
(defrule selección      ;Crea hecho difuso a partir valor slot crisp
?f <-   (persona (nombre ?n) (edad ?e) (vive ?v)
        (test (> ?e 20))
=>
  (printout  t “La persona " ?n “ vive en “ ?v crlf)
  (fuzzify edad-DIFUSA ?e  0.1))))
```

Aserción valor a slot crisp a partir de hechos difusos

(deftemplate edad-difusa
0 120 años
((infantil (12 1) (20 0)) *;Valores difusos*
(joven (10 0) (15 1) (25 1) (30 0))
(adulta (20 0) (30 1) (60 1) (70 0))
(mayor (60 0) (70 1))))

Dato difuso: **(assert (edad-difusa joven))**

(deftemplate persona
(slot nombre (type SYMBOL))
(slot edad-maximum (type FLOAT))
(slot edad-moment (type FLOAT))
)

(assert (persona (nombre Pedro)))

```
(defrule defuzzificar-final      ; Pone un  valor a slot crisp a partir hecho difuso
        (declare (salience -200))
        (edad-difusa  ?ed)
?p <- (persona (nombre ?n))
=>
   (retract ?p)
   (assert (persona  (nombre ?n)  (edad-maximum (maximum-defuzzify ?ed))
                                  (edad-moment (moment-defuzzify ?ed))))
    (printout t  "Persona " ?n " - Edad por moment:  " (moment-defuzzify ?ed) crlf)
    (printout t  "Persona " ?n " - Edad por maximum:  " (maximum-defuzzify ?ed) crlf)
   (halt)
)
```

Problema a resolver

Datos entrada (crisp)
- Categoria
- Edad-Aparente

fusificación

Variables Difusas

Datos entrada (difusos)
- Categoria
- Edad-Aparente

Inferencia difusa

Tabla > Reglas

Salida (difusos)
- Valor catastral/superficie

defusificación

Salida (crisp)
- Valor catastral/superficie (VUE)

Nº de ventanas

Valor catastral modificado

Categoria-Vivienda	Edad_Aparente	⇒	VUE-DIF
Alta	*Reciente*	⇒	*muyAlto*
Alta	*niNuevo_niMedio*	⇒	*Alto*
Alta	*niMedio_niViejo*	⇒	*Medio*
Intermedio	*Nuevo*	⇒	*niMedio_niAlto*
Intermedio	*niMedio_niViejo*	⇒	*Bajo*
Estándar	*Nuevo*	⇒	*Medio*
Estándar	*Viejo*	⇒	*muyBajo*
Económica	*Nuevo*	⇒	*niBajo_niMedio*
Económica	*niMedio_niViejo*	⇒	*muyBajo*

Práctica 1: FUZZY-CLIPS

Tarea:

- Realizar el ejercicio propuesto (se necesitará para el día de la evaluación, en el que se planteará una breve ampliación o modificación)
- La evaluación de las prácticas presupone que se hayan realizado las mismas y obtenido el código correspondiente al problema planteado en cada práctica

Calendario:

Sem	LABORATORIO	Evaluación
29-IX	Fuzzy-CLIPS	
6-X	Fuzzy-CLIPS	
13-X		P1 *Evaluación FuzzyClips*

Razonamiento Aproximado (15%) P1