TECNOLÓGICO
NACIONAL DE MÉXICO

Instituto Tecnológico de Culiacán

Carrera: Ing. Tecnologías de la información y comunicaciones

TOPICOS DE INTELIGENCIA ARTIFICIAL

DR. ZURIEL DATHAN MORA FELIX

Grupo: 12:00 – 01:00 PM

Tarea 2 control difuso en sistemas expertos

Alumno: Rodrigo Alonso Páez Gastélum

Número de control: 20170080

Introducción

Un control difuso en sistemas expertos de inteligencia artificial (IA) es un tipo de sistema de control basado en la lógica difusa, que permite manejar información incierta, imprecisa o subjetiva. Se utiliza cuando un sistema no puede modelarse con precisión mediante métodos matemáticos tradicionales o cuando las reglas del sistema son más cercanas al razonamiento humano.

¿Qué es la lógica difusa?

Es una lógica que intenta tratar las contradicciones en forma atenuada, también es multivaluada en la cual los valores de verdad de las variables pueden ser cualquier numero real comprendido entre 0 y 1. Fue propuesta por el matemático Lotfi A. Zadeh.

Se usa para estudiar la verdad parcial, o sea aquella que los valores pueden variar entre “completamente verdadero” o “completamente falso”. Tiene como base los conjuntos difusos y posee un sistema de inferencia basado en reglas de producción “SI antecedente ENTONCES consecuente”

La lógica difusa (Fuzzy Logic) es una extensión de la lógica clásica que permite manejar valores intermedios entre verdadero (1) y falso (0). En lugar de tomar decisiones binarias, usa grados de verdad.

Ejemplo:

Lógica clásica → "Si la temperatura es mayor a 30°C, entonces hace calor." (Blanco o negro)

Lógica difusa → "Si la temperatura es alrededor de 30°C, entonces hace algo de calor." (Gradual)

Esto permite que los sistemas difusos tomen decisiones más parecidas al pensamiento humano.

¿Qué es un control difuso?

Un control difuso convierte entradas numéricas en términos lingüísticos y aplica reglas para tomar decisiones. difusa se encarga de resolver una variedad de problemas estrictamente relacionados con el control de procesos industriales complejos y sistemas de decisión. Estos sistemas tratan de imitar la forma de tomar decisiones de los humanos, en este caso además de tomar decisiones más rápidamente, lo hacen según la proximidad de los valores de verdad a 0 o a 1, es decir, a "completamente falso" o "completamente verdadero".

¿Qué es un sistema experto?

Es un programa de inteligencia artificial que imita la capacidad de un experto humano para tomar decisiones y resolver problemas. Estos sistemas utilizan reglas, bases de conocimiento y técnicas de inferencia para analizar información y dar respuestas similares a las de un especialista.

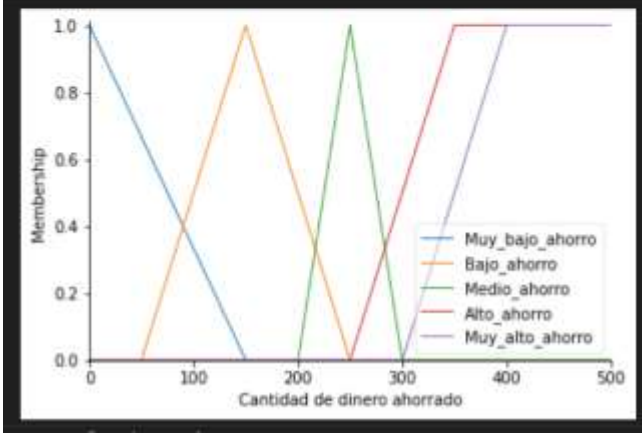
Algunas de las características de un sistema experto son

- Capacidad para inferir
- Tomar decisiones
- Dominio de una información especifica
- Permite que los usuarios interactúen con el sistema

Ejemplo 1

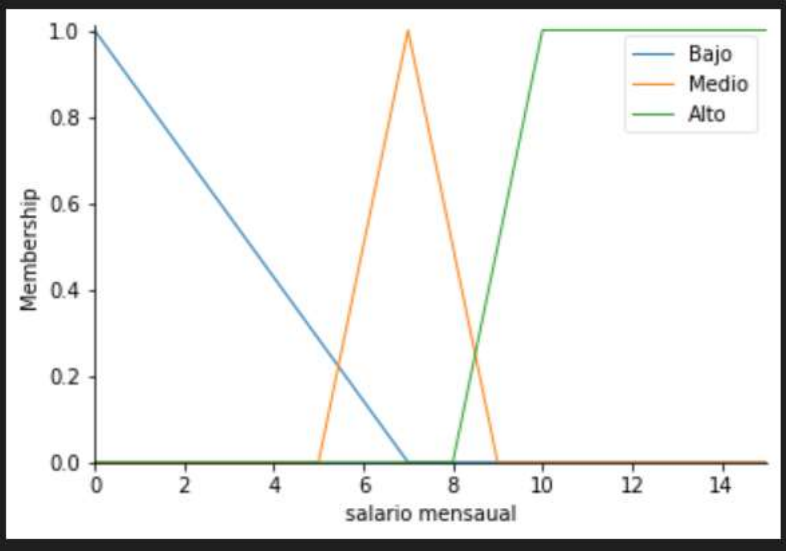
Funcion de fusificacion

```
#Funciones de membresia

# Dinero ahorrado muy bajo - bajo - medio - alto - muy alto

dinero_ahorro['Muy_bajo_ahorro'] = fuzz.trimf(dinero_ahorro.universe,[0,0,150])
dinero_ahorro['Bajo_ahorro'] = fuzz.trimf(dinero_ahorro.universe,[50,150,250])
dinero_ahorro['Medio_ahorro'] = fuzz.trimf(dinero_ahorro.universe,[200,250,300])
dinero_ahorro['Alto_ahorro'] = fuzz.trapmf(dinero_ahorro.universe,[250,350,500,500])
dinero_ahorro['Muy_alto_ahorro'] = fuzz.trapmf(dinero_ahorro.universe,[300,400,500,500])

dinero_ahorro.view()
```



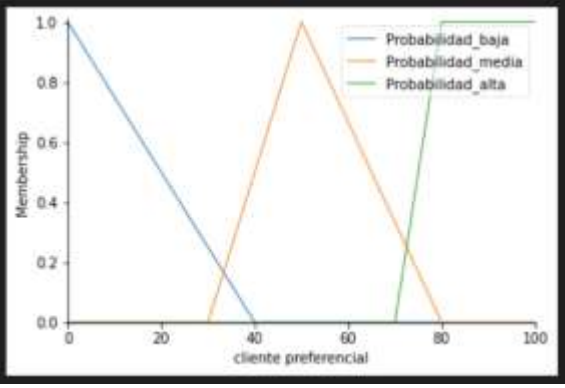
Evaluación

```
salario['Bajo'] = fuzz.trimf(salario.universe,[0,0,7])
salario['Medio'] = fuzz.trimf(salario.universe,[5,7,9])
salario['Alto'] = fuzz.trapmf(salario.universe,[8,10,15,15])

salario.view()
```



Desfusificacion

```
#Variables de salida
#Cliente preferencial baja - media - alta

cliente_preferencial['Probabilidad_baja'] = fuzz.trimf(cliente_preferencial.universe,[0,0,40])
cliente_preferencial['Probabilidad_media'] = fuzz.trimf(cliente_preferencial.universe,[30,50,80])
cliente_preferencial['Probabilidad_alta'] = fuzz.trapmf(cliente_preferencial.universe,[70,80,100,100])

cliente_preferencial.view()
```



Ejemplo 2

Reglas

```
# Las reglas
rule1 = ctrl.Rule(e_temperatura['M_frio'] & d_eT['M_positivo'], f_voltaje['M_Alto'])
rule2 = ctrl.Rule(e_temperatura['M_frio'] & d_eT['positivo'], f_voltaje['M_Alto'])
rule3 = ctrl.Rule(e_temperatura['frio'] & d_eT['M_positivo'], f_voltaje['M_Alto'])
rule4 = ctrl.Rule(e_temperatura['frio'] & d_eT['positivo'], f_voltaje['Alto'])
rule5 = ctrl.Rule(e_temperatura['frio'] & d_eT['cero'], f_voltaje['Medio'])
rule6 = ctrl.Rule(e_temperatura['ideal'] & d_eT['positivo'], f_voltaje['Medio'])
rule7 = ctrl.Rule(e_temperatura['ideal'] & d_eT['cero'], f_voltaje['Medio'])
rule8 = ctrl.Rule(e_temperatura['ideal'] & d_eT['Negativo'], f_voltaje['Medio'])
rule9 = ctrl.Rule(e_temperatura['caliente'] & d_eT['cero'], f_voltaje['Medio'])
rule10 = ctrl.Rule(e_temperatura['caliente'] & d_eT['Negativo'], f_voltaje['Medio'])
rule11 = ctrl.Rule(e_temperatura['caliente'] & d_eT['M_Negativo'], f_voltaje['Bajo'])
rule12 = ctrl.Rule(e_temperatura['M_caliente'] & d_eT['M_Negativo'], f_voltaje['M_Bajo'])
rule13 = ctrl.Rule(e_temperatura['M_caliente'] & d_eT['Negativo'], f_voltaje['M_Bajo'])
```

Función de fusificacion

```
#Se declaran los rangos de la entradas y la salida
e_temperatura = ctrl.Antecedent(np.arange(0, 60, 1),'e_temperatura')
d_eT = ctrl.Antecedent(np.arange(-0.1,0.1, 0.001), 'd_eT')
f_voltaje=ctrl.Consequent(np.arange(0.1,110, 1), 'f_voltaje')

e_temperatura['M_frio'] = fuzz.trapmf(e_temperatura.universe, [0, 0, 37, 38])
e_temperatura['frio'] = fuzz.trimf(e_temperatura.universe, [37, 39, 40])
e_temperatura['ideal'] = fuzz.trimf(e_temperatura.universe, [39, 40, 41])
e_temperatura['caliente']= fuzz.trimf(e_temperatura.universe, [40, 41, 42.07])
e_temperatura['M_caliente']= fuzz.trapmf(e_temperatura.universe, [41.1,42.12, 59.1,59.1])
```

```
# Generate fuzzy membership functions
Temperatura_mf = fuzz.trapmf(x_e_temperatura,[0, 0, 37, 38])
Temperatura_f = fuzz.trimf(x_e_temperatura ,[37, 39, 40])
Temperatura_ideal = fuzz.trimf(x_e_temperatura , [39, 40, 41])
Temperatura_c = fuzz.trimf(x_e_temperatura , [40, 41, 42.07])
Temperatura_mc = fuzz.trapmf(x_e_temperatura , [41.1,42.12, 59.1,59.1])
#Para Derivada del error
eT_Mn= fuzz.trapmf(x_d_eT, [-0.1,-0.1,-0.06,-0.04])
eT_n = fuzz.trapmf(x_d_eT, [-0.06,-0.04,-0.003333,0])
eT_cero = fuzz.trimf(x_d_eT, [-0.003333,0,0.003333])
eT_p = fuzz.trapmf(x_d_eT, [0, 0.003333, 0.04,0.06])
eT_Mp=fuzz.trapmf(x_d_eT,[0.04,0.06,0.1,0.1])

#Para la voltaje
voltaje_M_Bajo= fuzz.trimf(x_Voltaje, [0,0,5])
voltaje_Bajo= fuzz.trapmf(x_Voltaje, [0,10,65,70])
voltaje_Medio = fuzz.trimf(x_Voltaje, [60,70,80])
voltaje_Alto = fuzz.trapmf(x_Voltaje,[70, 75, 107,109])
voltaje_M_Alto = fuzz.trapmf(x_Voltaje, [108,109.4,110,110])
```

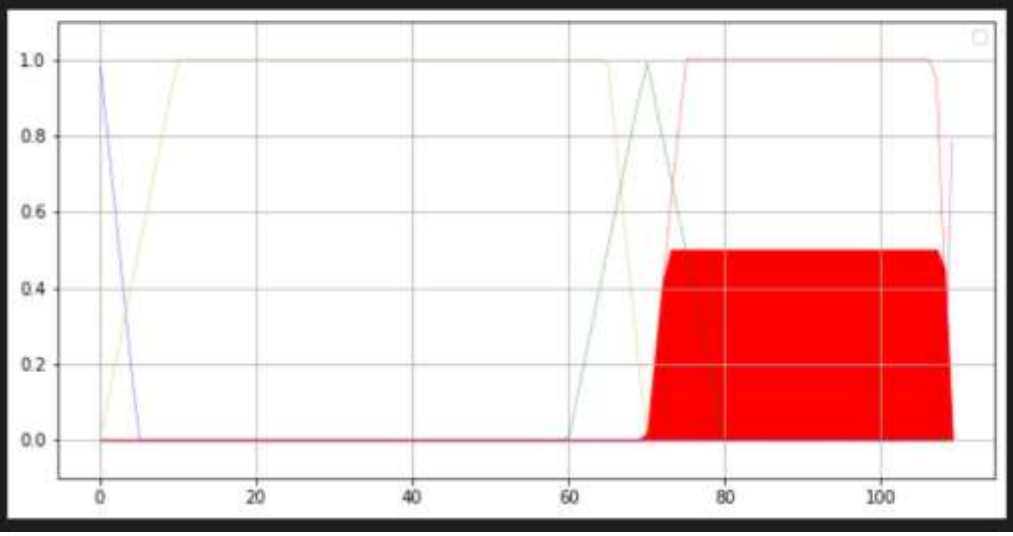
```
plt.rcParams["figure.figsize"] = 10, 20

plt.subplot(5,1,2), plt.plot(x_e_temperatura , Temperatura_mf, 'b', linewidth=1.5, label='Muy frio')
plt.subplot(5,1,2), plt.plot(x_e_temperatura , Temperatura_f, 'g', linewidth=1.5, label='frio')
plt.subplot(5,1,2), plt.plot(x_e_temperatura , Temperatura_ideal, 'r', linewidth=1.5, label='ideal')
plt.subplot(5,1,2), plt.plot(x_e_temperatura , Temperatura_c, 'm', linewidth=1.5, label='caliente')
plt.subplot(5,1,2), plt.plot(x_e_temperatura , Temperatura_mc, 'y', linewidth=1.5, label=' muy caliente'), plt.title('Error temperatura')
plt.legend()

plt.subplot(5,1,3), plt.plot(x_d_eT, eT_Mn, 'y', linewidth=1, label='muy negativo')
plt.subplot(5,1,3), plt.plot(x_d_eT, eT_n, 'b', linewidth=1, label='Negativo')
plt.subplot(5,1,3), plt.plot(x_d_eT, eT_cero, 'g', linewidth=1, label='cero')
plt.subplot(5,1,3), plt.plot(x_d_eT, eT_p, 'r', linewidth=1, label='Positivo')
plt.subplot(5,1,3), plt.plot(x_d_eT, eT_Mp, 'r', linewidth=1, label='muy positivo'), plt.title('derivada deError')
plt.legend()

plt.subplot(5,1,4), plt.plot(x_Voltaje, voltaje_M_Bajo, 'b', linewidth=1.5, label='V_bajo')
plt.subplot(5,1,4), plt.plot(x_Voltaje, voltaje_Bajo, 'b', linewidth=1.5, label='bajo')
plt.subplot(5,1,4), plt.plot(x_Voltaje, voltaje_Medio, 'g', linewidth=1.5, label='V_medio')
plt.subplot(5,1,4), plt.plot(x_Voltaje,voltaje_Alto, 'r', linewidth=1.5, label='Alto'),plt.title('A_Voltaje')
plt.subplot(5,1,4), plt.plot(x_Voltaje,voltaje_M_Alto, 'r', linewidth=1.5, label='Alto'),plt.title('MA_Voltaje')
```

Desfusificacion

```
step=or_rule(voltaje_act_medio1, voltaje_act_medio2, voltaje_act_medio3)
step1=or_rule(step,voltaje_act_medio4,voltaje_act_medio5)
voltaje_act_Medio=np.fmax(step1,voltaje_act_medio5)

step= or_rule(voltaje_act_Mbajo,voltaje_act_bajo1,voltaje_act_Medio )
voltaje= or_rule(step,voltaje_act_alto1, voltaje_act_MAlto)

plt.rcParams["figure.figsize"] = 10, 5
plt.plot(x_Voltaje, voltaje_M_Bajo, 'b', linewidth=0.5, linestyle='--', )
plt.plot(x_Voltaje, voltaje_Bajo, 'y', linewidth=0.5, linestyle='--', )
plt.plot(x_Voltaje,voltaje_Medio, 'g', linewidth=0.5, linestyle='--')
plt.plot(x_Voltaje, voltaje_Alto, 'r', linewidth=0.5, linestyle='--')
plt.plot(x_Voltaje, voltaje_M_Alto, 'm', linewidth=0.5, linestyle='--')
plt.legend()

plt.fill_between(x_Voltaje, voltaje, color='r')
plt.ylim(-0.1, 1.1)
plt.grid(True)
plt.show()



defuzz_voltaje =(fuzz.defuzz(x_Voltaje,voltaje,'centroid'))
Voltaje_act = fuzz.interp_membership(x_Voltaje, voltaje, defuzz_voltaje)  # for plot
voltaje0 = np.zeros_like(x_Voltaje)
```

```
# Turn off top/right axes
for ax in (ax0,):
    ax.spines['top'].set_visible(False)
    ax.spines['right'].set_visible(False)
    ax.get_xaxis().tick_bottom()
    ax.get_yaxis().tick_left()
    plt.tight_layout()
    print('El valor de centroide es: ', defuzz_voltaje)
return voltaje


mperatura_A= float(input("Digite la t_entrada [30-60]"))
ltaje_eval = apply_T_rules(temperatura_A)
```

```
Digite la t_entrada [30-60]38
No handles with labels found to put in legend.
La derivada de error es:  0.02
La derivada de error es:  0.02
```



Referencias

https://es.wikipedia.org/wiki/L%C3%B3gica_difusa