Mihaela (Ghindeanu) Colhon

Elemente de Logică Fuzzy

Craiova, 2012

Cuprins

Lista de figuri

Lista de tabele

Capitolul 1

Despre IA

1.1 Concepte generale. Inspiraţii

Obiectivul inteligenţei artificiale este simularea comportamentului inteligent al unui om. În inteligenţa artificială există două abordări unanim acceptate. *Abordarea conexionistă* se bazează pe ideea că inteligenţa este emergentă reţelelor neuronale. Pe de altă parte, *abordarea simbolică* bazată pe *ipoteza sistemelor simbolice fizice* a lui H. Simon şi A. Newell consideră că orice sistem inteligent (natural sau artificial) este un sistem fizic de prelucrare a simbolurilor.

În abordările simbolice de inteligenţă artificială, prelucrările sunt efectuate de un program care manipulează cunoştinţe dintr-o aşa numită *bază de cunoştinţe*. Programele de calcul sunt denumite în acest caz *sisteme bazate pe cunoştinţe*, principala lor caracteristică fiind separarea netă între baza de cunoştinţe şi mecanismul de raţionament. Una din cele mai grele probleme ale dezvoltări unui astfel de sistem este construirea bazei de cunoştinţe, aşa numita achiziţie de cunoştinţe.

1.1.1 Reprezentarea cunoaşterii

Spre deosebire de *informaţie*, noţiunea centrală în informatică (de unde şi termenul de IT - *information technology* - tehnologia informaţiei) cunoaşterea era, până de curând, specifică doar inteligenţei artificiale. În acest context, cunoaşterea are o semnificaţie bine precizată. Conform definiţiei lui Alan Newell, unul dintre pionierii inteligenţei artificiale, *cunoaşterea este ceea ce poate fi atribuit unui agent uman sau artificial astfel încât comportarea sa să poată fi catalogată drept raţională*.

Tot Newell în lucrarea sa *Nivelul cunoştinţelor* atrage atenţia asupra faptului că în universul ştinţei calculatoarelor există un nivel separat al cunoştinţelor situat deasupra nivelului programelor care la rândul lui este deasupra celor specifice hardware-ului (transferul între regiştrii, circuitele şi dispozitivele). Trecerea la un alt nivel, cel al cunoştinţelor este noutatea adusă tehnologiei informaţiei.

Capitolul 2

Elemente de Logică Fuzzy

2.1 Mulţimi fuzzy

Noţiunea de *mulţime fuzzy* a fost introdusă de Lotfi A. Zadeh începând cu anul 1965 cu scopul de a modela caracterul imprecis al apartenenţei. El a propus generalizarea conceptului de apartenenţă binară a unui element la o multime crisp deoarece teoria clasică a mulţimilor limitează posibilitatea descrierii matematice a unor situaţii reale.

Dacă pentru mulţimile crisp apartenenţa unui element la o mulţime este de tip binar (*da*/*nu*) în cazul mulţimilor fuzzy este vorba de un grad de apartenenţă. Fie X o mulţime crisp ale cărei elemente dorim să le analizăm folosind tehnici fuzzy. Atunci:

$$A = \{(x, \varphi_A(x)) \mid x \in X\}$$

defineşte o *mulţime fuzzy* a lui X unde φ_A este *functia de apartenenţă* relativă la mulţimea A, pentru $A \subseteq X$ iar $\varphi_A(x)$ reprezintă gradul de apartenenţă al lui x la mulţimea fuzzy A. De obicei $\varphi_A(x) \in [0, 1]$. Mulţimea X se mai numeste *univers de discus* sau *mulţime referenţial* [1].

Definiţia 2.1.1 *[1] O mulţime fuzzy A se numeşte normalizată sau normală dacă*

$$\sup\{\varphi_A(x) \,|x \in X\} = 1$$

şi subnormală în caz contrar.

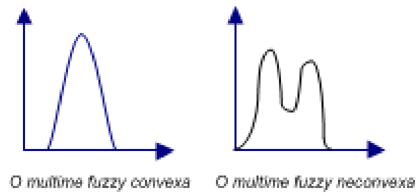

Figura 2.1: Convexitatea mulţimilor fuzzy

Orice submulţime fuzzy subnormală se poate normaliza considerând drept funcţie caracteristică pe $\frac{\varphi_A(x)}{\sup\{\varphi_A(x)\ |x \in X\}}$ în loc de $\varphi_A(x)$.

Definiţia 2.1.2 *[1] O mulţime fuzzy A a unei mulţimi univers incluse în spaţiul Euclidian se numeste convexă dacă şi numai dacă:*
$(\forall u \in X)(\forall v \in X)(\forall w \in X)(\varphi_A(w) \geq \min(\varphi_A(u), \varphi_A(v)))$
ori de câte ori $\exists \lambda \in [0, 1]$ *astfel încât:* $w = \lambda u + (1 - \lambda)v$ *sau:*
$\forall x_1, x_2, x_3 \in X : x_1 \leq x_2 \leq x_3$ *obţinem:* $\varphi_A(x_2) \geq \min(\varphi_A(x_1), \varphi_A(x_3))$

Convexitatea mulţimilor fuzzy joacă un rol important în definirea *partiţiilor fuzzy*.

Definiţia 2.1.3 *Un tuplu* $(A_1, ..., A_n)$ *cu* $A_i \in X, i = \overline{1, n}$ *se numeşte partiţie fuzzy dacă:*

$$\sum_{i=1}^{n} \varphi_{A_i}(x) = 1,\ A_i \neq \emptyset,\ A_i \neq X$$

Observaţia 2.1.1 *O partiţie fuzzy formată din mulţimi fuzzy normale şi convexe nu conţine mai mult de două mulţimi fuzzy suprapuse.*

Fie X o multime referenţiar.

Definiţia 2.1.4 *[1] Se numeşte tăietura de prag* α *sau* $\alpha -$ *tăietura mulţimea crisp:*

$$[\varphi]_\alpha = \{x \in X \mid \varphi(x) \geq \alpha\}$$

Dacă inegalitatea este strictă se spune că $\alpha -$ *tăietura* este de tip tare şi va fi notată cu $[\varphi]_{+\alpha}$.

Fie A o mulţime fuzzy de univers X.

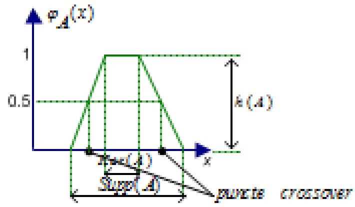

Figura 2.2: Mărimi asociate mulţimilor fuzzy

Definiţia 2.1.5 *[1] Nucleul mulţimii A notat $Ker(A)$ este tăietura de prag 1 a mulţimii:*

$$Ker(A) = \{x \in X \mid \varphi_A(x) = 1\}$$

Definiţia 2.1.6 *[1] Suportul mulţimii A notat $Supp(A)$ este tăietura strictă de nivel 0 a mulţimii A:*

$$Supp(A) = \{x \in X \mid \varphi_A(x) > 0\}$$

Definiţia 2.1.7 *[1] Frontiera mulţimii A notată $Fr(A)$ este mulţimea crisp a elementelor ce au grad de apartenenţă intermediar între 0 şi 1:*

$$Fr(A) = \{x \in X \mid \varphi_A(x) \in (0, 1)\}$$

Definiţia 2.1.8 *[1] Înalţimea mulţimii A notată $h(A)$ reprezintă cea mai mare valoare luată de funcţia sa de apartenenţă:*

$$h(A) = \sup\{\varphi_A(x) \mid x \in X\}$$

2.2 Operaţii cu mulţimi fuzzy

Fie A si B două mulţimi fuzzy de univers X. Dacă $\varphi_A = \varphi_B$ atunci se spune că A şi B sunt mulţimi fuzzy egale ($A = B$). Dacă $\varphi_A \leq \varphi_B$ atunci spunem ca mulţimea A este inclusă în mulţimea B ($A \subseteq B$).

Funcţia de apartenenţă a reuniunii $A \cup B$ este definită cu ajutorul funcţiilor φ_A si φ_B astfel [1]:

$$\varphi_{A\cup B}(x) = \max(\varphi_A(x), \varphi_B(x)),\ x \in X$$
$$\varphi_{\cup_{i\in I}A_i}(x) = \sup\{\varphi_{A_i}(x)|\ i \in I\},\ x \in X$$

Funcţia de apartenenţă a intersecţiei $A \cap B$ este definită de asemenea cu ajutorul funcţiilor φ_A şi φ_B [1]:

$$\varphi_{A\cap B}(x) = \min(\varphi_A(x), \varphi_B(x)),\ x \in X$$
$$\varphi_{\cap_{i\in I}A_i}(x) = \inf\{\varphi_{A_i}(x)|\ i \in I\},\ x \in X$$

Presupunem că A este o mulţime fuzzy normalizată. Funcţia de apartenenţă a complementarei mulţimii A, notată $C[A]$ este definită prin [1]:

$$\varphi_{C[A]}(x) = 1 - \varphi_A(x)$$

2.2.1 Principiul extensiei

De multe ori în modelarea fuzzy este necesar să fuzzificăm mulţimi crisp. Să considerăm două mulţimi crisp X şi Y iar $f : X \to Y$ o funcţie. Fie A o submulţime fuzzy de referenţial X cu funcţia de apartenenţă φ_A. Zadeh a propus o metodă pentru constuirea de mulţimi fuzzy induse de funcţii, metodă cunoscută sub numele de *principiul extensiei*.

Definiţia 2.2.1 *[1] **Principiul extensiei** Imaginea mulţimii A prin funcţia f este mulţimea fuzzy $f(A)$ de referenţial Y cu funcţia de apartenenţă definită după cum urmează:*

$$\varphi_{f(A)}(y) = \begin{cases} \sup_{x\in f^{-1}(y)} \varphi_A(x), dac\breve{a}\ \ f^{-1}(y) \neq \emptyset \\ 0,\ altfel \end{cases}$$

Acest principiu este unul din elementele importante ale teoriei mulţimilor fuzzy. De exemplu, pentru o mulţime A de univers finit, principiul extensiei ne permite extensia unei funcţii f de la punctele referenţialului X la mulţimile fuzzy ale lui X [1]:

$$f(A) = f(\varphi_A(x_1)/x_1 + \varphi_A(x_2)/x_2 + ... + \varphi_A(x_n)/x_n) =$$
$$\varphi_A(x_1)/f(x_1) + \varphi_A(x_2)/f(x_2) + ... + \varphi_A(x_n)/f(x_n)$$

Fie B o multime fuzzy de referenţial Y cu funcţia de apartenenţă φ_B. Imaginea inversă a mulţimii B prin f notată $f^{-1}(B)$ este mulţimea fuzzy de referenţial X cu funcţia de apartenenţă [1]:

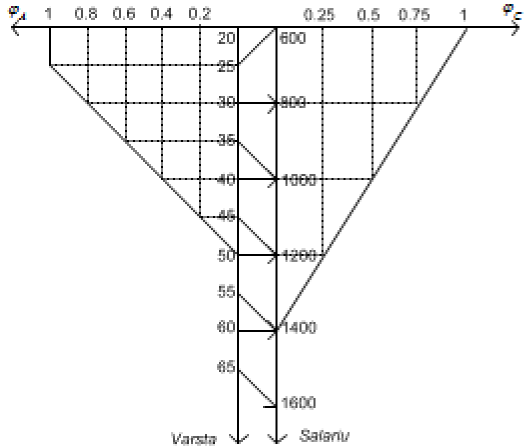

Figura 2.3: Principiul extensiei aplicat în Exemplul 1.1

$$\varphi_{f^{-1}(B)}(x) = \varphi_B(f(x))$$

Exemplul 2.2.1 *Să considerăm următorul tabel care indică salariile angajaţilor unei companii în funcţie de vârsta lor:*

varsta	20	25	30	35	40	45	50	55	60	65
salariu	600	600	800	1000	1000	1200	1200	1400	1400	1600

Ne propunem să determinăm care este salariul angajaţilor tineri folosind principiul extensiei. Pentru aceasta vom considera funcţia $f : V \to S$ unde $V = \{20, 25, 30, 35, 40, 45, 50, 55, 60, 65\}$ iar $S = \{600, 800, 1000, 1200, 1400, 1600\}$ astfel încât $f(20)= f(25)= 600$, $f(30)= 800$, $f(35)= f(40)= 1000$, $f(45)= f(50)= 1200$, $f(55)= f(60)= 1400$, $f(65)= 1600$.

Vom defini multimea A ca fiind multimea fuzzy de univers V ”angajaţi tineri”. Deoarece V este o multime discretă vom putea scrie pe A sub forma: $A = \sum_{x \in V} \varphi_A(x)/x$:

$A = 1/20 + 1/25 + 0.8/30 + 0.6/35 + 0.4/40 + 0.2/45 + 0/50 + 0/55 + 0/60 + 0/65$

Pe baza mulţimii A şi a funcţiei f construim mulţimea fuzzy B ”salariul angajaţilor tineri”. Avem că $B = f(A)$ ($\forall x \in V,\ f(x) \in S$). Obţinem:

$$\varphi_B(y) = \varphi_{f(A)}(y) = \max_{x \in V \mid f(x)=y} \varphi_A(x)$$

$B = f(A) = f(\varphi_A(20)/20 + \varphi_A(25)/25 + \varphi_A(30)/30 + \varphi_A(35)/35 + \varphi_A(40)/40 + \varphi_A(45)/45 + \varphi_A(50)/50 + \varphi_A(55)/55 + \varphi_A(60)/60 + \varphi_A(65)/65)$

$= \varphi_A(20)/f(20) + \varphi_A(25)/f(25) + \varphi_A(30)/f(30) + \varphi_A(35)/f(35) + \varphi_A(40)/f(40) + \varphi_A(45)/f(45) + \varphi_A(50)/f(50) + \varphi_A(55)/f(55) + \varphi_A(60)/f(60) + \varphi_A(65)/f(65) = 1/600 + 1/600 + 0.8/800 + 0.6/1000 + 0.4/1000 + 0.2/1200 + 0/1200 + 0/1400 + 0/1400 + 0/1600 = 1/600 + 0.8/800 + 0.6/1000 + 0.2/1200 + 0/1400 + 0/1600$

Să considerăm acum mulţimea fuzzy C "salarii mici" de univers S:

$C = 1/600 + 0.75/800 + 0.5/1000 + 0.25/1200 + 0/1400 + 0/1600$

Vrem să definim mulţimea fuzzy " vârsta angajaţilor cu salarii mici". Această mulţime este de univers V cu funcţia de apartenenţă $\varphi_{f^{-1}(C)}$ unde:

$$\varphi_{f^{-1}(C)}(x) = \varphi_C(f(x)).$$

$f^{-1}(C) = \sum_{x\in V} \varphi_{f^{-1}(C)}(x)/x = \sum_{x\in V} \varphi_C(f(x))/x = \varphi_C(f(20))/20 +$
$+\varphi_C(f(25))/25 + \varphi_C(f(30))/30 + \varphi_C(f(35))/35 + \varphi_C(f(40))/40 +$
$+\varphi_C(f(45))/45 + \varphi_C(f(50))/50 + \varphi_C(f(55))/55 + \varphi_C(f(60))/60 +$
$+\varphi_C(f(65))/65 = 1/20 + 1/25 + 0.75/30 + 0.5/35 + 0.5/40 + 0.25/45 +$
$+0.25/50 + 0/50 + 0/60 + 0/65$

2.2.2 Numere fuzzy

Definiţia 2.2.2 *Un număr fuzzy A este o mulţime fuzzy a numerelor reale care aproximează un alt numar real şi care are o funcţie de apartenenţă convexă şi continuă cu suport marginit.*

Definiţia 2.2.3 *Un număr fuzzy A se numeşte număr fuzzy triunghiular cu centrul c, lăţimea la stânga $\alpha > 0$, lăţimea la dreapta $\beta > 0$, dacă funcţia de apartenenţă are forma:*

$$\varphi_A(x) = \begin{cases} 1 - \frac{c-x}{\alpha}, & c - \alpha \le x \le c \\ 1 - \frac{x-c}{\beta}, & c < x \le c + \beta \\ 0, & altfel \end{cases}$$

Folosim pentru un număr fuzzy triunghiular notaţia $A = (c, \alpha, \beta)$ cu $supp(A) = (c - \alpha, c + \beta)$. Semnificaţia acestei mulţimi fuzzy cu centrul c este "*numărul este aproximativ egal cu c*".

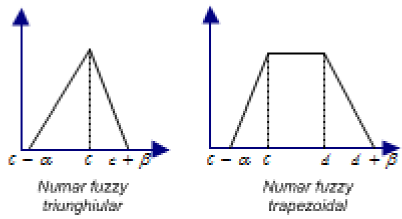

Figura 2.4: Numere fuzzy

Definiţia 2.2.4 *Un număr fuzzy A se numeşte număr fuzzy trapezoidal cu intervalul de toleranţă* $[c, d]$*, lăţimea la stânga* $\alpha > 0$*, lăţimea la dreapta* $\beta > 0$*, dacă are următoarea funcţie de apartenenţă:*

$$\varphi_A(x) = \begin{cases} 1 - \frac{c-x}{\alpha}, \ c - \alpha \le x \le c \\ 1, \ c < x \le d \\ 1 - \frac{x-d}{\beta}, \ d < x \le d + \beta \\ 0, \ altfel \end{cases}$$

Folosim următoarea notaţie pentru un număr fuzzy triunghiular $A = (c, d, \alpha, \beta)$ pentru care $supp(A) = (c - \alpha, d + \beta)$. Semnificaţia lui A este ”*numărul este aproximativ între c şi d*”.

Un număr fuzzy este întotdeauna o mulţime fuzzy însă reciproca nu este tot timpul adevărată. Extinzând noţiunea de număr fuzzy putem defini noţiunea de *interval fuzzy.*

Definiţia 2.2.5 *Un interval fuzzy este o mulţime fuzzy cu aceleaşi restricţii ca pentru numere fuzzy, cu excepţia faptului că nucleul nu mai este restricţionat la un singur punct.*

Operaţii cu numere fuzzy

Fie A si B două numere fuzzy cu funcţiile de apartenenţă φ_A respectiv φ_B. Operaţiile aritmetice elementare între două numere fuzzy se definesc astfel:

$$\varphi_{A(\star)B}(z) = \max\{\min(\varphi_A(x), \varphi_B(y)), \ z = x * y\}$$

unde prin $\star$ notăm operaţiile: + (adunare), − (scădere), · (înmulţire), /(împărţire).

Operaţiile logice $\wedge$ (*şi* logic), $\vee$ (*sau* logic) se realizează astfel:

$\varphi_{A(\wedge)B}(z) = \max\{\min(\varphi_A(x), \varphi_B(y)),\ z = \min\{x, y\}\}$

$\varphi_{A(\vee)B}(z) = \max\{\min(\varphi_A(x), \varphi_B(y)),\ z = \max\{x, y\}\}$

Exemplul 2.2.2 *Fie A_1 numărul fuzzy "în jur de 5" cu funcţia de apartenenţă*

$$\varphi_{A_1}(x) = \max(1 - \frac{1}{2}|x - 5|, 0)$$

şi A_2 numărul fuzzy "în jur de 2" cu funcţia de apartenenţă

$$\varphi_{A_2}(x) = \max(1 - |x - 2|, 0)$$

Se doreşte calculul expresiei $A_1(+)A_2$.

Observăm că ambele numere A_1 şi A_2 sunt numere fuzzy triunghiulare: $A_1 = (5, 2, 2)$, $A_2 = (2, 1, 1)$.

Din expresiile funcţiilor de apartenenţă obţinem: A_1=0.5/4 + 1/5 + 0.5/6 iar A_2 = 0/1 + 1/2 + 0/3 de unde rezultă $A_1(+)A_2$ = 0.5/6 + 1/7 + 0.5/8 deci $A_1(+)A_2$ este numărul fuzzy triunghiular "în jur de 7", $A_1(+)A_2 = (7, 3, 3)$.

2.3 *t*-norme si *t*-conorme

Definiţia 2.3.1 *[1] O funcţie $T : [0, 1] \to [0, 1]$ se numeşte t-normă dacă sunt adevărate toate proprietăţile următoare:*

- $T(x, 1) = x,\ \forall x \in [0, 1]$ *condiţia la limită*
- $T(x_1, x_2) \leq T(x_3, x_4),\ \forall x_1 \leq x_3, x_2 \leq x_4$ *monotonie*
- $T(x, y) = T(y, x),\ \forall x, y \in [0, 1]$ *comutativitate*
- $T(T(x_1, x_2), x_3) = T(x_1, T(x_2, x_3)),\ \forall x_1, x_2, x_3 \in [0, 1]$ *asociativitate*

Din aceste proprietăţi mai obţinem $T(x, 0) = T(0, x) = 0,\ \forall x \in [0, 1]$. Exemple de *t*-norme [1]:

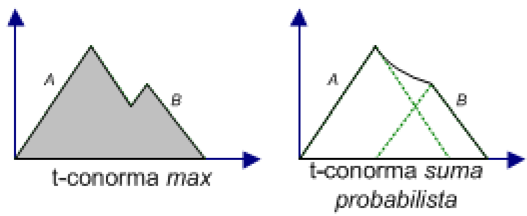

Figura 2.5: Exemple de *t*-norme

- $T_{\min}(x, y) = \min(x, y) \stackrel{not}{=} T_0(x, y)$
- $T_{Lukasiewicz}(x, y) = \max\{0, x + y - 1\} \stackrel{not}{=} T_\infty(x, y)$
- $T_{prod}(x, y) = xy \stackrel{not}{=} T_1(x, y)$

Fie A si B două mulţimi fuzzy. În cazul în care aceste mulţimi sunt legate folosind operatorul lingvistic AND atunci vom reprezenta această relaţie folosind o *t*-normă. Cele mai utilizate *t*-norme sunt:

minimum: $\varphi_{A\cap B}(x) = T(\varphi_A(x), \varphi_B(x)) = \min(\varphi_A(x), \varphi_B(x))$

produsul: $\varphi_{A\cap B}(x) = T(\varphi_A(x), \varphi_B(x)) = \varphi_A(x)\varphi_B(x)$

Definiţia 2.3.2 *[1] O funcţie* $S : [0, 1] \to [0, 1]$ *se numeşte t-conormă dacă sunt adevarate toate proprietăţile următoare:*

- $S(x, 0) = x,\ \forall x \in [0, 1]$ *condiţia la limită*
- $S(x_1, x_2) \le S(x_3, x_4),\ \forall x_1 \le x_3, x_2 \le x_4$ *monotonie*
- $S(x, y) = S(y, x),\ \forall x, y \in [0, 1]$ *comutativitate*
- $S(S(x_1, x_2), x_3) = S(x_1, S(x_2, x_3)),\ \forall x_1, x_2, x_3 \in [0, 1]$ *asociativitate*

Din proprietăţile de mai sus obţinem $S(x, 1) = S(1, x) = 1,\ \forall x \in [0, 1]$. Exemple de *t*-conorme:

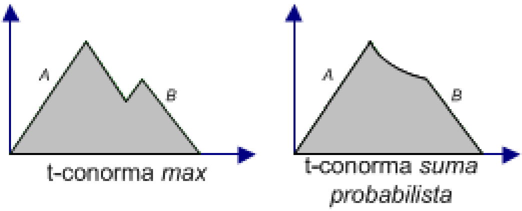

Figura 2.6: Exemple de t-conorme

- $S_{\max}(x, y) = \max(x, y) \stackrel{not}{=} S_0(x, y)$
- $S_{Lukasiewicz}(x, y) = \min\{x + y, 1\} \stackrel{not}{=} S_\infty(x, y)$
- $S_{prod}(x, y) = x + y - xy \stackrel{not}{=} S_1(x, y)$

Fie A si B două mulţimi fuzzy. În cazul în care aceste mulţimi sunt legate folosind operatorul lingvistic OR atunci vom reprezenta această relaţie folosind o t-conormă. Cele mai utilizate t-conorme sunt:

maximum: $\varphi_{A\cup B}(x) = S(\varphi_A(x), \varphi_B(x)) = \max(\varphi_A(x), \varphi_B(x))$,

suma probabilistă: $\varphi_{A\cup B}(x) = S(\varphi_A(x), \varphi_B(x)) = \varphi_A(x) + \varphi_B(x) - \varphi_A(x)\varphi_B(x)$

suma marginită: $\varphi_{A\cup B}(x) = S(\varphi_A(x), \varphi_B(x)) = \min(1, \varphi_A(x) + \varphi_B(x))$

Definiţia 2.3.3 *[1] O funcţie $n : [0, 1] \to [0, 1]$ se numeşte negaţie sau complement dacă satisface condiţiile:*

- $n(0) = 1$
- $n(1) = 0$
- *dacă $u \geq v$ atunci $n(u) \leq n(v)$ adică n este descrescătoare*

O negaţie n este *strictă* [1] dacă n este funcţie strict descrescătoare şi continuă. Uzual se utilizează negaţia standard $n(x) = 1 - x$, $\forall x \in [0, 1]$. În acest caz unei t-norme T i se poate asocia o t-conormă S prin:

$$S(x, y) = 1 - T(1 - x, 1 - y)$$

Utilitatea operaţiei *negaţie* se poate observa în propoziţia următoare:

Propoziţia 2.3.1 *[1] Fie T o t-normă şi n o negaţie strictă atunci funcţia S_n definită prin:*

$$S_n(x, y) = n(\ T(n(x), n(y))\ )$$

reprezintă o t-conormă. Dacă T este min şi $n(x) = 1 - x$ atunci S_n este max.

2.4 Relaţii fuzzy

O relaţie binară crisp *R(U,V)* poate fi definită astfel:

$$\varphi_R(u, v) = \begin{cases} 1, \ daca\ (u, v) \in R(U, V) \\ 0, \ altfel \end{cases}$$

Fiind date mulţimile univers U şi V atunci o relaţie fuzzy $R(U, V)$ este o submulţime fuzzy a lui $U \times V$:

$$R(U, V) = \{\ ((u, v), \varphi_R(u, v)) \mid (u, v) \in U \times V\},\ \varphi_R(u, v) \in [0, 1]$$

Exemplul 2.4.1 *Să presupunem că vrem să definim conceptul "vehiculul x este aproape de vehiculul y" în condiţiile în care ambele vehicule se deplasează pe un drum de 40 km. Fiecare poziţie a vehiculului este definită considerându-se distanţa dintre acesta şi origine. Dacă dorim să definim acest concept în logica clasică, trebuie să stabilim într-un mod neambiguu ce înseamnă aproape sau departe în metrica considerată. Astfel, dacă o distanţă de cel mult 30 km între cele două vehicule ne permite să spunem că vehiculele sunt aproape unul de celălalt atunci putem exprima relaţia crisp "vehiculul x este aproape de vehiculul y" prin intermediul următoarei matrici relaţionale:*

	$\mathbf{v_1 = 0}$	$\mathbf{v_2 = 20}$	$\mathbf{v_3 = 40}$
$\mathbf{u_1 = 0}$	1	1	0
$\mathbf{u_2 = 20}$	1	1	1
$\mathbf{u_3 = 40}$	0	1	1

unde prin u_1, u_2, u_3 notăm poziţiile vehiculului x iar prin v_1, v_2, v_3 poziţiile vehiculului y iar domeniul (discret) de valori pentru poziţia fiecărui vehicul este {0, 20, 40}.

Dacă în schimb alegem să definim acest concept folosind o relaţie fuzzy $R(U, V)$ unde $U = V = \{ 0, 20, 40 \}$ cu funcţia de apartenenţă: $\varphi_R(u, v) = 1 - \frac{|u-v|}{40}$, $(u, v) \in U \times V$ atunci matricea relaţională devine:

	$\mathbf{v_1 = 0}$	$\mathbf{v_2 = 20}$	$\mathbf{v_3 = 40}$
$\mathbf{u_1 = 0}$	1	1/2	0
$\mathbf{u_2 = 20}$	1/2	1	1/2
$\mathbf{u_3 = 40}$	0	1/2	1

O relaţie fuzzy este de fapt o mulţime fuzzy definită pe un spaţiu produs, ceea ce ne permite să utilizăm operaţiile de bază ale mulţimilor fuzzy şi pe mulţimea relaţiilor fuzzy. Astfel, dacă R_1 şi R_2 sunt două relaţii fuzzy definite pe acelaşi spaţiu produs $U \times V$ atunci *intersecţia* şi *reuniunea* lor pot fi definite astfel:

$$\varphi_{R_1 \cup R_2}(u, v) = S(\varphi_{R_1}(u), \varphi_{R_2}(v))$$
$$\varphi_{R_1 \cap R_2}(u, v) = T(\varphi_{R_1}(u), \varphi_{R_2}(v))$$

O relaţie fuzzy de univers $U \times V$ se numeşte [1]:

- *reflexivă* dacă $\varphi_R(u, u) = 1, \ \forall u \in U$
- *antisimetrică* dacă din $\varphi_R(u, v) > 0$ şi $\varphi_R(v, u) > 0$ avem $u = v$
- *simetrică* dacă $\varphi_R(u, v) = \varphi_R(v, u)$
- *tranzitivă* dacă $R \circ R \subseteq R$

O relaţie fuzzy reflexivă, simetrică şi tranzitivă se numeşte *relaţie fuzzy de ordine* [1].

Exemplul 2.4.2 *Să presupunem că vrem să definim conceptul "vehiculul x este aproape de vehiculul y şi vehiculul x este mai aproape de sfârşitul drumului decât y". Prima parte a propoziţiei este relaţia definită în Exemplul 2.4.1. A doua parte a propoziţiei poate fi definită folosind următoarea relaţie fuzzy de acelaşi univers ca şi precedenta:*

	$\mathbf{v_1 = 0}$	$\mathbf{v_2 = 20}$	$\mathbf{v_3 = 40}$
$\mathbf{u_1 = 0}$	0.5	0.25	0
$\mathbf{u_2 = 20}$	0.75	0.5	0.25
$\mathbf{u_3 = 40}$	1	0.75	0.5

Cum ambele părţi ale propoziţiei sunt definite pe acelaşi spaţiu produs $U \times V$, întreaga propoziţie poate fi calculată folosind o T-normă (minimum).

De fapt multe relaţii fuzzy sunt obţinute prin combinări ale mulţimilor fuzzy. Pentru a face acest lucru posibil mulţimile fuzzy sunt mai intâi convertite la relaţii fuzzy care apoi sunt operate cu o normă. Primul pas este realizat folosind *principiul extensiei cilindrice* propusă de L. A. Zadeh.

Acest principiu este aplicat atunci cand suntem interesaţi să adăugăm o nouă dimensiune unei relaţii fuzzy de n domenii pentru a obţine o relaţie fuzzy de $n+1$ domenii. În acest sens dacă avem o mulţime fuzzy X a cărei funcţie de apartenenţă e definită pe universul U, $\varphi_X : U \to [0,1]$ atunci extensia cilindrica a lui X pe $U \times V$ este relaţia fuzzy definită prin:

$$ce\ of\ X\ on\ U \times V = \{\ ((u,v), \varphi_X(u,v)) \mid (u,v) \in U \times V\}$$

Astfel, gradul de apartenenţă a lui $u \in U$ este copiat la toate perechile (u,v), $\forall v \in V$.

Operatorul complementar extensiei este cel de *proiecţie*. Definim proiecţia lui $U \times V$ pe U ca fiind mulţimea fuzzy:

$$proj\ of\ U \times V\ on\ U = \{\ (u, v_{\max\{\varphi_R(u,v)\}}) \mid u \in U\}$$

Exemplul 2.4.3 *Să presupunem că avem două mulţimi fuzzy A = ”viteza fluidului pe conductă este mare” şi B =”debitul fluidului pe conductă este mare” (vezi Figura 2.7). Cu aceste două mulţimi vrem să construim relaţia fuzzy ”viteza fluidului pe conductă este mare când debitul fluidului este mare”.*

Viteza fluidului ia valori între 0 şi 1 m/s iar debitul este definit pe o scară de la 0 la 15 m^3/s . Deoarece funcţiile de apartenenţă ale mulţimilor A şi B sunt funcţii liniare vom considera domeniile (discrete) $U = \{0, 0.5, 0.7, 1\}$ pentru mulţimea A şi $V = \{0, 7.5, 14, 15\}$ pentru mulţimea B.

Pentru început folosim principiul extensiei cilindrice pentru a construi relaţia fuzzy R_1 pornind de la mulţimea A şi relaţia fuzzy R_2 pornind de la mulţimea B:

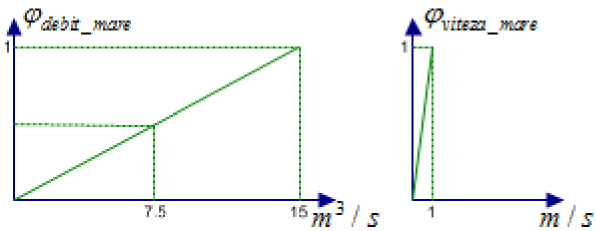

Figura 2.7: Cele două mulţimi fuzzy din Exemplul 2.4.3

viteza\debit (R_1)	**0**	**7.5**	**14**	**15**
0	0	0	0	0
0.5	0.5	0.5	0.5	0.5
0.7	0.7	0.7	0.7	0.7
1	1	1	1	1

viteza\debit (R_2)	**0**	**7.5**	**14**	**15**
0	0	0.5	0.9	1
0.5	0	0.5	0.9	1
0.7	0	0.5	0.9	1
1	0	0.5	0.9	1

Relaţia fuzzy "viteza fluidului este mare când debitul acestuia este mare" se obţine aplicând o T-normă relaţiilor R_1 şi R_2 şi anume min. Vom obţine:

viteza\debit	**0**	**7.5**	**14**	**15**
0	0	0	0	0
0.5	0	0.5	0.5	0.5
0.7	0	0.5	0.7	0.7
1	0	0.5	0.9	1

Să considerăm acum două relaţii binare definite pe spaţii produs diferite dar care au o mulţime comună: $R_1(U,V)$ şi $R_2(V,W)$. *Compunerea* acestor două relaţii este relaţia $R(U,W)$ care în logica clasică este definită astfel:

$$R(U,W) = R_1(U,V) \circ R_2(V,W) = \{\, (u,w) \in U \times W \mid \exists v \in V : (u,v) \in U \times V,\ (v,w) \in V \times W\}$$

În logica fuzzy compunerea a două relaţii binare definite pe spaţii produs diferite dar care au o mulţime comună se defineşte analog compunerii crisp:

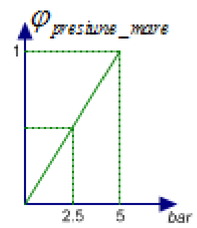

Figura 2.8: Mulţimea fuzzy *”diferenţa de presiune este mare”*

$$R(U, W) = \{\ ((u, w), \varphi_R(u, w)) \mid (u, w) \in U \times W\} \text{ unde}$$
$$\varphi_R(u, w) = S_v(\ T(\varphi_{R_1}(u, v), \varphi_{R_2}(v, w))\ )$$

Deşi se poate folosi orice T-normă şi S-normă cele mai multe compuneri de relaţii fuzzy binare folosesc *maximul* ca S-normă şi *produsul* sau *minimul* drept T-normă:

- $\varphi_R(u, w) = \max\{\min(\varphi_{R_1}(u, v), \varphi_{R_2}(v, w)) \mid v \in V\}$– compunerea *max-min*
- $\varphi_R(u, w) = \max\{\varphi_{R_1}(u, v) \bullet \varphi_{R_2}(v, w) \mid v \in V\}$– compunerea *max-produs*

De multe ori astfel de compuneri mai sunt numite şi *sup-star compuneri* unde operatorul *star* este o T-normă:

$$\varphi_R(u, w) = \sup\{\varphi_{R_1}(u, v) * \varphi_{R_2}(v, w) \mid v \in V\}$$

Exemplul 2.4.4 *Vom considera din nou relaţia fuzzy din ultimul exemplu ”viteza fluidului pe conductă e mare când debitul acestuia este mare” (R_1) şi în plus relaţia ”debitul fluidului pe conductă este mare când diferenţa de presiune este mare” (R_2) pentru a calcula: ”viteza fluidului pe conductă este mare când diferenţa de presiune pe conductă este mare” (R) folosind o cumpunere max-min:*

$$\begin{array}{r|cccc} \begin{array}{r} viteza\backslash debit \\ (R_1) \end{array} & \mathbf{0} & \mathbf{7.5} & \mathbf{14} & \mathbf{15} \\ \mathbf{0} & 0 & 0 & 0 & 0 \\ \mathbf{0.5} & 0 & 0.5 & 0.5 & 0.5 \\ \mathbf{0.7} & 0 & 0.5 & 0.7 & 0.7 \\ \mathbf{1} & 0 & 0.5 & 0.9 & 1 \end{array} \circ \begin{array}{r|cccc} \begin{array}{r} debit\backslash presiune \\ (R_2) \end{array} & \mathbf{0} & \mathbf{2.5} & \mathbf{3.5} & \mathbf{1} \\ \mathbf{0} & 0 & 0 & 0 & 0 \\ \mathbf{7.5} & 0 & 0.5 & 0.5 & 0.5 \\ \mathbf{14} & 0 & 0.5 & 0.7 & 0.9 \\ \mathbf{15} & 0 & 0.5 & 0.7 & 1 \end{array} =$$

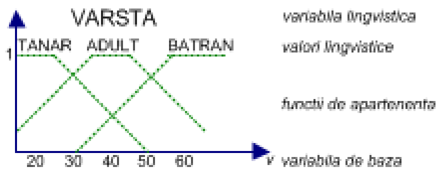

Figura 2.9: Variabila lingvistică *vârsta*

=

viteza\presiune (R)	**0**	**2.5**	**3.5**	**5**
0	0	0	0	0
0.5	0	0.5	0.5	0.5
0.7	0	0.5	0.7	0.7
1	0	0.5	0.7	1

2.5 Variabile lingvistice

Spre deosebire de variabilele algebrice ale căror valori sunt numere, variabilele lingvistice au ca valori cuvinte sau secvenţe de cuvinte în limbaj natural sau artificial. Mulţimea valorilor unei variabile lingvistice se numeşte *mulţimea termenilor* [1]. Fiecare element din mulţimea termenilor este o valoare fuzzy definită în raport cu o *variabilă de bază*. Valorile variabilei de bază definesc mulţimea univers U a tuturor valorilor fuzzy din mulţimea termenilor [1].

Definiţia 2.5.1 *[1] Se numeşte variabilă lingvistică sistemul* $(x, T(x), U, G, M)$ *unde:*

- *x este numele variabilei*
- *$T(x)$ mulţimea termenilor lingvistici (termen lingvistic: valoare ce are asociată o funcţie de apartenenţă şi care caracterizează o variabilă lingvistică)*
- *U mulţimea universului de discurs*
- *G gramatica care produce termenii lui $T(x)$*

- *M regula semantică care mapează termenii lui $T(x)$ la mulţimi fuzzy ale lui U*

Exemplul 2.5.1 *Să considerăm variabila lingvistică x cu numele cantitate:*
$x = (cantitate, T(cantitate), U, G, M)$

- $T(cantitate) = \{mult,\ foarte\ mult,\ foarte\ foarte\ mult,\ etc\}$
- $U = [50, 100]$
- $G(cantitate) : T \rightarrow mult \mid foarte\ T$
- $M(mult) = \{\ (u, \varphi_{mult}(u)) \mid u \in [50, 100]\}$ *iar*
$$\varphi_{mult}(u) = \begin{cases} 1,\ daca\ u \in [90, 100] \\ (1 + \frac{90-u}{5})^{-2}, u \in [50, 90] \end{cases}$$

În exemplul de mai sus termenul *mult* este utilizat ca variabilă de bază în $T(cantitate)$. Astfel de termeni se numesc *termeni primari*. Adăugând modificatori la termenii primari obţinem noi termeni - *termeni fuzzy*.

În general *modificatorii lingvistici* se împart în trei categorii:

- *aproximatori de mulţimi fuzzy*. Cu astfel de modificatori dintr-un scalar putem obţine o mulţime fuzzy. Exemple de aproximatori: *aproape, în jurul, aproximativ, etc*
- *restrângere de mulţimi fuzzy.* Aceştia modifică forma funcţiei de apartenenţă. Exemple de astfel de modificatori: *sub, peste*
- *intensificare sau diminuare.* Şi aceştia modifică funcţia de apartenenţă după cum urmează:
$$\varphi(x\ modificat) = \varphi(x)^n$$
  - *intensificare (concentrare)*: *foarte* (n=2), *extrem de* (n=3), etc.
  - *diminuare (dilatare)*: *aproape* (n=1/2), *în mare măsură* (n=5/7), etc

Un termen lingvistic de cele mai multe ori este format din:

- *un termen primar: bun, prost, vechi, rar*, etc.
- *un modificator fuzzy: foarte, aproape, în jur de, probabil*, etc.

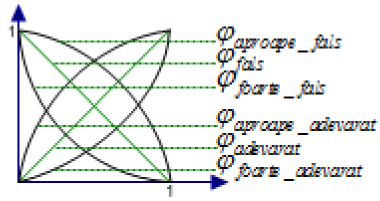

Figura 2.10: Valori de adevăr fuzzy

2.5.1 Valori de adevăr fuzzy

Folosind modificatorii lingvistici Baldwin a definit pe multimea univers $V = \{v \mid v \in [0,1]\}$ valorile de adevăr fuzzy. Mulţimea lor este:

T={*adevărat, foarte adevărat, aproape adevărat, absolut adevărat, ..., fals, absolut fals, îndeajuns de fals*}

Aceste valori de adevăr fuzzy au funcţiile de apartenenţă după cum urmează:

$$\varphi_{adevărat}(v) = v \qquad \varphi_{fals}(v) = 1 - \varphi_{adevărat}(v)$$
$$\varphi_{foarte\_adevărat}(v) = (\varphi_{adevărat}(v))^2 \qquad \varphi_{foarte\_fals}(v) = (\varphi_{fals}(v))^2$$
$$\varphi_{aproape\_adevărat}(v) = (\varphi_{adevărat}(v))^{1/2} \qquad \varphi_{aproape\_fals}(v) = (\varphi_{fals}(v))^{1/2}$$
$$\varphi_{absolut\_adevărat}(v) = \begin{cases} 1, & v = 1 \\ 0, & altfel \end{cases} \qquad \varphi_{absolut\_fals}(v) = \begin{cases} 1, & v = 0 \\ 0, & altfel \end{cases}$$

pentru $v \in [0,1]$.

Se observă că:

$\varphi_{foarte\_adevărat}(v) \leq \varphi_{adevărat}(v) \leq \varphi_{aproape\_adevărat}(v)$ şi
$\varphi_{foarte\_fals}(v) \leq \varphi_{fals}(v) \leq \varphi_{aproape\_fals}(v)$

2.6 Propoziţii şi implicaţii fuzzy

Logica fuzzy se ocupă cu stabilirea valorii de adevăr a propoziţiilor fuzzy. O *propoziţie fuzzy elementară* este de forma ”*x este A*” unde $A \in T(x)$ sau $A \in CT(x)$ (prin $CT(x)$ notăm mulţimea termenilor x care au fost modificaţi) [1].

Din motive practice trebuie ca funcţia de apartenenţă a termenului A să fie normalizată. Valoarea de adevăr a unei propoziţii elementare ”*x este A*” este definită prin intermediul lui $\varphi_A(x)$ [1].

O operaţie importantă în modelarea fuzzy este *implicaţia fuzzy*. O implicaţie între două propoziţii fuzzy defineşte tot o propoziţie fuzzy care se poate exprima condiţional prin [1]:

Dacă x este A atunci y este B

Această construcţie se obţine pornind de la [1]:

- două variabile lingvistice: $(x, T(x), U, G_x, M_x)$ şi $(y, T(y), V, G_y, M_y)$
- $A \in T(x)$ o mulţime fuzzy de univers U
- $B \in T(y)$ o mulţime fuzzy de univers V.

Practic valoarea de adevăr a implicaţiei fuzzy este definită de funcţia de apartenenţă φ_R a unei relaţii R între mulţimile U şi V. Mai precis [1]:

$$\varphi_R(u, v) = \Phi(\varphi_A(u), \varphi_B(v))$$

Definiţia implicaţiei în logica propoziţională consideră că o propoziţie este falsă dacă şi numai dacă antecedentul este fals şi concluzia adevărată. Funcţia Φ se alege astfel încât dacă A şi B sunt mulţimi crisp atunci implicaţia fuzzy să fie similară celei din logica bivalentă.

Există patru clase importante de implicaţii fuzzy:

- *S-implicaţii*: bazate pe $p \to q = \neg p \vee q = S(\, n(\varphi(p), \varphi(q))\,)$
- *QL-implicaţii*: bazate pe $p \to q = \neg p \vee (p \wedge q) = \neg(p \wedge \neg(p \wedge q)) = n(T(\varphi(p),$ $n(\, T(\varphi(p), \varphi(q)))))$

  QL-implicaţiile (*Quantum Logic*) modelează condiţii de forma:

  ”*dacă ... atunci ... altfel*”

Aceste implicaţii sunt bazate pe $p \to q = \neg p \vee (p \wedge q) = \neg(p \wedge \neg(p \wedge q)) = n(T(\varphi(p), n(T(\varphi(p), \varphi(q)))))$

În acest fel structura "*dacă x este A atunci y este B altfel y este D*" se poate interpreta:

$$(x, y) \in (A \times B) \cup (C[A] \times D)$$

deci $\varphi_R(x, y) = S(T(\varphi_A(x), \varphi_B(y)), T(\varphi_{C[A]}(x), \varphi_D(y))) = S(T(\varphi_A(x), \varphi_B(y)), T(n(\varphi_A(x)), \varphi_D(y)))$.

Dacă $T(n(\varphi_A(x)), \varphi_D(y)) = n(\varphi_A(x))$, adică dacă $\varphi_D(y) = 1$ atunci obţinem:

$$\varphi_R(x, y) = S(T(\varphi_A(x), \varphi_B(y)), n(\varphi_A(x)))) = n(T(\varphi_A(x), n(T(\varphi_A(x), \varphi_B(y)))))$$

- *R-implicaţiile* bazate pe $p \to q = \sup\{r \in [0, 1] \mid (p \wedge r) \leq q\}$ sau $p \to q = \sup\{r \in [0, 1] \mid T(\varphi(p), r) \leq \varphi(q)\}$
- *t-norm implicaţiile* bazate pe $p \to q = p \wedge q$ (falsă în logica booleană!)

Deşi implicaţiile $t - norm$, cum ar fi implicaţia *Mamdani* nu se verifică pentru mulţimi crisp ele sunt totusi cele mai utilizate în logica fuzzy.

În tabelul de mai jos sunt prezentate cele mai utilizate implicaţii fuzzy [1]:

Nume	**Functie**	**Clasa**
Reichenbach	$1 - \varphi_A(u) + \varphi_A(u)\varphi_B(v)$	**S**
Lukasiewicz	$\min(1, 1 - \varphi_A(u) + \varphi_B(v))$	**S**, **R** bazat pe: $(A \to B) = (\neg A) \vee B$
Gödel	$\begin{cases} 1, \ dac\breve{a}\ \varphi_A(u) \leq \varphi_B(v) \\ \varphi_B(v), \ altfel \end{cases}$	**R** modus ponens generalizat
Gaines	$\begin{cases} 1, \ dac\breve{a}\ \varphi_A(u) \leq \varphi_B(v) \\ 0, \ altfel \end{cases}$	**R**
Kleene-Dienes	$\max(1 - \varphi_A(u), \varphi_B(v))$	**S**, **QL** bazat pe: $(A \to B) = (\neg A) \vee B$
Zadeh	$\max(1 - \varphi_A(u), \min(\varphi_A(u), \varphi_B(v)))$	**QL** bazat pe: $(A \to B) = (\neg A) \vee (A \wedge B)$
Mamdani	$\min(\varphi_A(u), \varphi_B(v))$	**T**
Larsen	$\varphi_A(u)\varphi_B(v)$	

2.6.1 Modus ponens generalizat

Schema de raţionament modus ponens generalizat este următoarea:

Premiza: u is A^*

Regula: IF u is A THEN v is B

Consecinţa: v is B^*

Mulţimea fuzzy A^* din premiză poate să fie diferită de mulţimea A din regulă, însă ambele mulţimi A şi A^* trebuie să fie definite pe acelaşi univers. La fel pentru B şi B^*. Pentru exemplificare să considerăm următoarea regulă:

Dacă persoana X este scundă atunci aceasta nu va fi un bun jucător de basket.

şi premiza:

X are 168 cm în înălţime.

În logica crisp o regulă poate fi aplicată dacă şi numai dacă premiza ei este identică cu antecedentul regulei iar rezultatul acestei operaţii este concluzia regulei. Însă în logica fuzzy o regulă este aplicată atâta timp cât gradul de apartenenţă al valorii lingvistice din premiză la mulţimea fuzzy din antecedentul regulei este nenul.

Astfel, în această regulă de raţionament avem:

- o relaţie fuzzy $A \to B$ unde:

$$A \to B = \{((u, v), \varphi_{A\to B}(u, v)) \mid (u, v) \in U \times V\}$$

  iar U şi V sunt mulţimile univers pentru A respectiv B.

- o mulţime fuzzy A^* a cărei funcţie de apartenenţă $\varphi_{A^*}(u)$ este cunoscută, $u \in U$

- şi vrem să obţinem mulţimea fuzzy B^* adică valorile funcţiei $\varphi_{B^*}(v)$, cu $v \in V$.

Mulţimea B^* este rezultatul compunerii dintre A^* şi $A \to B$, adică:

$$B^* = A^* \circ (A \to B)$$

de unde rezultă:

$$B^* = \{(v, \sup_u(\varphi_{A^*}(u) \star \varphi_{A\to B}(u, v))) \mid v \in V\}$$

unde prin operatorul $\star$ notăm orice T-normă.

Exemplul 2.6.1 *Să considerăm urmatoarea regulă:*

Dacă presiunea este mare atunci volumul este mic.

Definim mulţimea fuzzy "presiunea este mare" prin $A = 0/1 + 0.2/3 + 0.7/6 + 1/10$ *şi mulţimea fuzzy "volumul este mic" prin* $B = 1/1 + 0.6/10 + 0/50$.

Pentru implicaţia $A \rightarrow B$ vom folosi implicaţia Mamdani $\varphi_{A\rightarrow B}(u, v) = \min(\varphi_A(u), \varphi_B(v))$. *Astfel matricea relaţiei $A \rightarrow B$ este:*

volum\presiune	1	3	6	10
1	0	0.2	0.7	1
10	0	0.2	0.6	0.6
20	0	0	0	0

Să considerăm acum premiza: "presiunea este de 3 bar", ceea ce ne permite să scriem: $A^* = 0/1 + 1/3 + 0/6 + 0/10$. *Cum A^* nu este mulţimea A vom aplica metoda modus ponens generalizat pentru a obţine concluzia B^*:*

$$\varphi_{B^*}(v) = \max_u(\min(\varphi_{A^*}(u), \varphi_{A\rightarrow B}(u, v)))$$

şi vom avea: $B^* = 0.2/1 + 0.2/10 + 0/20$.

În schimb pentru premiza "presiunea este de 10 bar", deci pentru $A^* = 0/1 + 0/3 + 0/6 + 1/10$ *vom avea* $B^* = 1/1 + 0.6/10 + 0/20$.

2.7 Sisteme fuzzy

Raţionamentele în logica fuzzy utilizează mulţimi fuzzy pentru reprezentarea şi manipularea cunoştinţelor incerte. Un *sistem de control fuzzy* utilizează o colecţie de funcţii de apartenenţă şi reguli pentru a raţiona pe baza cunoştinţelor date. Valorile crisp care reprezintă intrările unui astfel de sistem sunt transformate în valori fuzzy pentru a putea fi folosite în aplicarea regulilor sistemului, reguli ce sunt formulate prin intermediul expresiilor lingvistice. Cele mai multe sisteme fuzzy transformă şi concluzia lingvistică obţinută în valoare crisp.

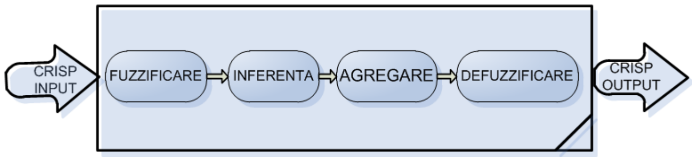

Figura 2.11: Arhitectura unui controler fuzzy

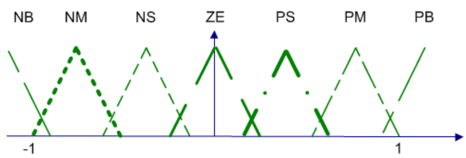

Figura 2.12: O partiţie fuzzy a intervalului [-1, 1]

Arhitectura unui sistem fuzzy pentru controlul proceselor - *controler fuzzy* este prezentată în Figura 2.11.

Aşa cum se observă în figura de mai sus componentele unui controler fuzzy sunt:

- *modulul de fuzzificare* care transformă valorile de intrare crisp în mulţimi fuzzy pentru a putea calcula valoarea de adevăr a premizei fiecărei reguli din baza de reguli pentru intrarea dată. În general intrările unui sistem fuzzy sunt valori crisp (valori *singleton*). În acest caz acest modul va transforma valoarea de intrare în *mulţime fuzzy impuls (singleton)*.

  De exemplu valoarea 10 din mulţimea univers U = {2, 5, 10, 15} va fi transformată în mulţimea fuzzy impuls A de univers U astfel:

  A = 0/2 + 0/5 + 1/10 + 0/15

  În multe aplicaţii practice se normalizează domeniul intrărilor folosindu-se următorul tip de partiţie: NB (negativ mare), NM (negativ mediu), NS (negativ mic), ZE (zero), PS (pozitiv mic), PM (pozitiv mediu), PB (pozitiv mare).

- *modulul de inferenţă* calculează valoarea de adevăr a premizelor din baza de reguli în raport cu datele de intrare şi aplică valorile obţinute concluziilor corespunzătoare obţinându-se astfel câte o mulţime fuzzy de ieşire pentru fiecare regulă.

- *modulul de agregare* combină toate mulţimile fuzzy asignate concluziilor regulilor de către modulul de inferenţă pentru a forma o singură mulţime fuzzy.
- *modulul de defuzzificare* are rolul de a converti mulţimea fuzzy obţinută de modulul de agregare la o valoare crisp care va fi valoarea de ieşire a sistemului pentru intrarea dată.

Există mai multe tipuri de sisteme fuzzy:

- sisteme fuzzy MIMO (*Multiple Input Multiple Output)*
- sisteme fuzzy MISO (*Multiple Input Single Output)*

Aşa cum am precizat, un sistem fuzzy conţine o mulţime de reguli pe baza cărora se fac raţionamentele. Regulile unui sistem fuzzy formează baza de reguli sau pe scurt *R*-baza. Fiecare regulă fuzzy este formată din două propoziţii fuzzy: premiza şi concluzia. Aceste propoziţii conţin variabile lingvistice conectate prin intermediul operatorilor lingvistici *and, or ...* . Totuşi marea majoritate a sistemelor fuzzy bazate pe reguli folosesc doar operatorul lingvistic *and* în premizele regulilor în timp ce concluzia este formată dintr-o singură valoare lingvistică (sisteme MISO). În cele ce urmează vom exemplifica componentele şi schemele de raţionament ale sistemelor fuzzy de clasă MISO şi anume ale sistemelor de tipul Two-Input/ Single-Output.

2.7.1 Two-Input/Single-Output fuzzy sistem

Forma generală a unui controler fuzzy de clasă Two-Input/Single-Output este următoarea:

Input: x is $A^{'}$ and y is $B^{'}$
$\mathbf{R}_1$ IF u is A_1 and v is B_1 THEN w is C_1
else $\mathbf{R}_2$ IF u is A_2 and v is B_2 THEN w is C_2
...
else $\mathbf{R}_p$ IF u is A_n and v is B_m THEN w is C_k

Output: w is C'

unde u şi v sunt variabile lingvistice reprezentând stările procesului modelat de sistem iar w este variabila lingvistică de control a sistemului. A_i, B_j şi C_k sunt valorile lingvistice ale variabilelor u, v respectiv w de univers U, V, W pentru $i=\overline{1,n}$ $j=\overline{1,m}$ şi $k=\overline{1,s}$.

Un astfel de sistem fuzzy necesită o bază de reguli (*R-bază*) de $n \times m$ reguli unde n reprezintă numărul de valori ale variabilei u iar m reprezinta numarul de valori ale variabilei v.

O regulă fuzzy:

$$\textbf{R}: \text{IF } u \text{ is } A_i \text{ and } v \text{ is } B_j \text{ THEN } w \text{ is } C_k$$

este implementată ca o relaţie de implicaţie fuzzy şi este definită după cum urmează:

$$\textbf{R}: (\ A_i \text{ and } B_j) \to C_k$$
$$\varphi_R(u, v, w) = \varphi_{(A_i \wedge B_j \to C_k)}(u, v, w) = [\varphi_{A_i}(u) \wedge \varphi_{B_j}(v)] \to \varphi_{C_k}(w)$$

unde A_i *and* B_j este o submulţime fuzzy $A_i \times B_j$ in $U \times V$ iar R este o relaţie de implicaţie fuzzy pe $U \times V \times W$.

Regula compoziţională de inferenţă

Să considerăm o regulă fuzzy din R-baza sistemului fuzzy considerat şi inferenţa sa (GMP - modus ponens generalizat):

Input: u is A' and v is B'

R: IF u is A and v is B THEN w is C

Output: w is C'

$A \subset U, B \subset V, C \subset W, u \in U, v \in V, w \in W$. Vom interpreta această regulă ca o implicaţie $A \times B \to C$ care este definită pe mulţimea produs $U \times V \times W$.

$$\text{R}: A \times B \to C \text{ sau } \text{R} = A \times B \times C$$

Când sistemul de inferenţă primeşte intrarea (A', B') atunci ieşirea C' este obţinută printr-o inferenţă notată cu operatorul de compunere $\circ$ astfel:

$$C' = (A', B') \circ \text{R}$$

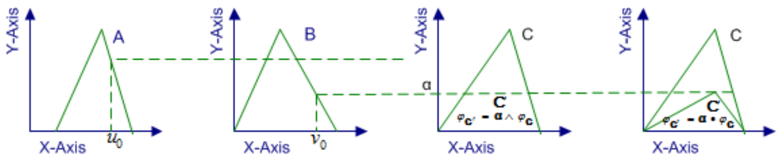

Figura 2.13: Mulţimea de ieşire $C^{'}$ corespunzătoare intrărilor crisp u_0 şi v_0

Zadeh a numit acest proces de inferenţă prin *regula compoziţională de inferenţă*. Inferenţa în sistemele fuzzy este determinată de doi factori: “operatorul de implicaţie” şi “operatorul de compunere”. Pentru implicaţie cei mai des folosiţi operatori sunt:

- *implicaţia Mamdani*: operatorul *min*
- *implicaţia Larsen*: operatorul *produs*

  Pentru compunere de asemenea avem doi operatori foarte utilizati:

- *compunerea Mamdani: max-min*
- *compunerea Larsen: max-produs*

Propoziţia 2.7.1 *Pentru inputuri singleton mulţimea de ieşire $C^{'}$ este determinată de minimul gradelor de potrivire ale mulţimilor fuzzy A şi B pentru intrările date. Dacă intrările sunt valori singleton $A^{'} = u_0$ şi $B^{'} = v_0$ şi notăm $\alpha = \min\{\varphi_A(u_0), \varphi_B(v_0)\}$ atunci rezultatul relaţiei de implicaţie fuzzy R este:*

- $\varphi_{C'}(w) = \alpha \wedge \varphi_C(w)$ *dacă folosim implicaţia Mamdani*
- $\varphi_{C'}(w) = \alpha \bullet \varphi_C(w)$ *dacă folosim implicaţia Larsen*

Demonstraţie. Cum $C^{'} = (A^{'},B^{'}) \circ R = (A^{'},B^{'}) \circ (A \text{ and } B \to C) = [A^{'} \circ (A \to C)] \cap [B^{'} \circ (B \to C)]$ rezultă că avem:

$$\varphi_{C'}(w) = \min\{u_0 \circ (\varphi_A(u) \to \varphi_C(w)), v_0 \circ (\varphi_B(v) \to \varphi_C(w))\} =$$
$$= \min\{\varphi_A(u_0) \to \varphi_C(w), \varphi_B(v_0) \to \varphi_C(w)\}$$

1. Dacă utilizăm operatorul de implicaţie fuzzy Mamdani:

   $\varphi_{C'}(w) = \min\{\min\{\varphi_A(u_0), \varphi_C(w)\}, \min\{\varphi_B(v_0), \varphi_C(w)\}\} = \min\{\min\{\varphi_A(u_0), \varphi_B(v_0)\}, \varphi_C(w)\} = \min\{\alpha\ , \varphi_C(w)\} = \alpha \wedge \varphi_C(w)$

2. Dacă utilizăm operatorul de implicaţie fuzzy Larsen:

   $\varphi_{C'}(w) = \min\{\varphi_A(u_0) \bullet \varphi_C(w), \varphi_B(v_0) \bullet \varphi_C(w)\} = \min\{\varphi_A(u_0), \varphi_B(v_0)\} \bullet \varphi_C(w) = \alpha \bullet \varphi_C(w)$ □

Propoziţia 2.7.2 *Pentru intrarea fuzzy ($A^{'}, B^{'}$) mulţimea de ieşire $C^{'}$ este determinată de minimul gradelor de potrivire ale lui ($A^{'}, A$) şi ($B^{'}, B$). Astfel, dacă intrarea este perechea de mulţimi fuzzy ($A^{'}, B^{'}$) atunci gradul de potrivire α este dat de minimul dintre ($A^{'}, A$) şi ($B^{'}, B$), deci:*

$$\alpha = \min\{\max_u(\varphi_{A'}(u) \wedge \varphi_A(u)), \max_v(\varphi_{B'}(v) \wedge \varphi_B(v))\}$$

iar mulţimea de ieşire $C^{'}$ este definită astfel:

1. $\varphi_{C'}(w) = \alpha \wedge \varphi_C(w)$ dacă folosim implicaţia Mamdani

2. $\varphi_{C'}(w) = \alpha \bullet \varphi_C(w)$ dacă folosim implicaţia Larsen

Demonstraţie. Cum $C^{'} = [A^{'} \circ (A \to C)] \cap [B^{'} \circ (B \to C)]$ obţinem:

$$\varphi_{C'}(w) = \min\{\varphi_{A'}(u) \circ (\varphi_A(u) \to \varphi_C(w)), \varphi_{B'}(v) \circ (\varphi_B(v) \to \varphi_C(w))\}$$

1. Dacă folosim operatorii Mamdani *min* pentru implicaţie şi *max-min* pentru compunere avem:

   $\varphi_{C'}(w) = \min\{\varphi_{A'}(u) \circ \min_u\{\varphi_A(u), \varphi_C(w)\}, \varphi_{B'}(v) \circ \min_v \{\varphi_B(v), \varphi_C(w)\}\} = \min \{\varphi_{A'}(u) \circ (\varphi_A(u) \wedge \varphi_C(w)), \varphi_{B'}(v) \circ (\varphi_B(v) \wedge \varphi_C(w))\} = \min\{\max_u \min_u \{\varphi_{A'}(u), \varphi_A(u) \wedge \varphi_C(w)\}, \max_v \min_v\{\varphi_{B'}(v), \varphi_B(v) \wedge \varphi_C(w)\}\} = \min\{\max_u(\varphi_{A'}(u) \wedge \varphi_A(u)) \wedge \varphi_C(w), \max_v(\varphi_{B'}(v) \wedge \varphi_B(v)) \wedge \varphi_C(w)\} = \min\{\max_u(\varphi_{A'}(u) \wedge \varphi_A(u)), \max_v(\varphi_{B'}(v) \wedge \varphi_B(v))\} \wedge \varphi_C(w) = \alpha \wedge \varphi_C(w)$

2. Dacă folosim operatorii Larsen *produs* pentru implicaţie şi *max-produs* pentru compunere avem:

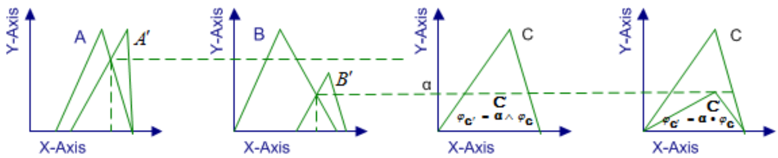

Figura 2.14: Mulţimea de ieşire $C^{'}$ corespunzătoare intrărilor fuzzy $A^{'}$ şi $B^{'}$

$\varphi_{C'}(w) = \min\{\varphi_{A'}(u) \circ (\varphi_A(u) \bullet \varphi_C(w)),\ \varphi_{B'}(v) \circ (\varphi_B(v) \bullet \varphi_C(w))\} = \min\{\max_u \min_u(\varphi_{A'}(u), \varphi_A(u)) \bullet \varphi_C(w), \max_v \min_v(\varphi_{B'}(v) \bullet \varphi_B(v)) \bullet \varphi_C(w)\} = \min\{\max_u(\varphi_{A'}(u) \wedge \varphi_A(u)) \bullet \varphi_C(w), \max_v(\varphi_{B'}(v) \wedge \varphi_B(v)) \bullet \varphi_C(w)\} = \min\{\max_u(\varphi_{A'}(u) \wedge \varphi_A(u)), \max_v (\varphi_{B'}(v) \wedge \varphi_B(v))\} \bullet \varphi_C(w) = = \alpha \bullet \varphi_C(w)$

2.7.2 Inferenţa fuzzy într-o *R*-bază a unui sistem fuzzy Two-Input/ Single-Output

În această secţiune vom extinde regula compoziţională de inferenţă la un set de p reguli. Astfel vom considera:

Input: $(A^{'}, B^{'})$

R = $\bigcup_{l=1}^{p} R_l$ unde R_l : $(A_i$ and $B_j) \rightarrow C_k$

Output: $C^{'}$

Propoziţia 2.7.3 *Mul'timea de ie'sire $C^{'}$ este rezultatul procesului de agregare a tuturor concluziilor $C_l^{'} = (A^{'}, B^{'}) \circ R_l$, $l=\overline{1,p}$ ob'tinute în urma aplicării celor p reguli din R-bază pentru intrarea $(A^{'}, B^{'})$.*

$$C^{'} = (A^{'}, B^{'}) \circ R = (A^{'}, B^{'}) \circ \bigcup_{l=1}^{p} R_l = \bigcup_{l=1}^{p} (A^{'}, B^{'}) \circ R_l = \bigcup_{l=1}^{p} C_l^{'}$$

Demonstraţie. Într-adevăr relaţia $C^{'} = (A^{'}, B^{'}) \circ \bigcup_{l=1}^{p} R_l$ este echivalentă cu:

$$\varphi_{C'}(w) = [\varphi_{A'}(u), \varphi_{B'}(v)] \circ \max_{u,v,w}\{\varphi_{R_1}(u, v, w), ..., \varphi_{R_p}(u, v, w)\}$$

Înlocuind operatorul de compunere $\circ$ cu operatorul *max-min* obţinem:

$\varphi_{C'}(w) = \max_{u,v} \min_{u,v}\{[\varphi_{A'}(u), \varphi_{B'}(v)], \max_{u,v,w}\{\varphi_{R_1}(u, v, w), ...,$

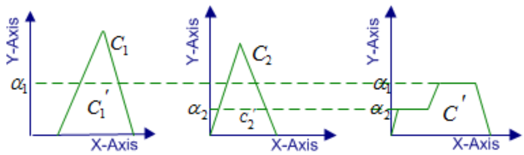

Figura 2.15: Reprezentarea grafică a metodei de inferenţă Mamdani

$\varphi_{R_p}(u, v, w)\}\}$= max$_{u,v}$ max$_{u,v,w}$$\{min_{u,v}$$\{[\varphi_{A'}(u), \varphi_{B'}(v)], \varphi_{R_1}(u, v, w)\}, ...,$
min$_{u,v}$$\{[\varphi_{A'}(u), \varphi_{B'}(v)], \varphi_{R_p}(u, v, w)\}\}$

Dar max$_{u,v}$ min$_{u,v}$$\{[\varphi_{A'}(u), \varphi_{B'}(v)], \varphi_{R_l}(u, v, w)\} = (\varphi_{A'}(u), \varphi_{B'}(v)) \circ \varphi_{R_l}(u, v, w) = \varphi_{C_l'}(w)$ cu l=$\overline{1, p}$ aşa încât vom avea:

$\varphi_{C'}(w)$ = max$_w$$\{\varphi_{C_1'}(w), , ..., \varphi_{C_p'}(w)\}$ deci $C' = \bigcup_{l=1}^{p} C_l'$. □

Pentru simplicitate vom considera în cele ce urmează că sistemul conţine numai două reguli în bază şi anume:

R$_1$ IF u is A_1 and v is B_1 THEN w is C_1

else R$_2$ IF u is A_2 and v is B_2 THEN w is C_2

Notăm cu α_1, α_2 gradul de potrivire al intrărilor în raport cu premiza primei, respectiv a celei de-a doua reguli din bază. Motorul de inferenţă al unui controler fuzzy poate fi implementat folosindu-se diverse metode de inferenţă. Cele mai cunoscute sunt: *Mamdani*, *Larsen*, *Tsukamoto* şi *TSK* (Takagi-Sugeno-Kang). Ele vor fi prezentate în cele ce urmează.

Metoda Mamdani

Această metodă foloseşte operatorul *min* pentru implicaţie şi operatorul *min-max* pentru compunere. Aşa cum s-a arătat

$$\varphi_{C_i'}(w) = \alpha_i \wedge \varphi_{C_i}(w) \text{ cu } i = \overline{1, 2}$$

iar mulţimea de ieşire a sistemului C' este reuniunea mulţimilor C_1' şi C_2'. Grafic, ieşirea unui sistem cu două reguli ce foloseşte metoda de inferenţă Mamdani este prezentată în Figura 2.15.

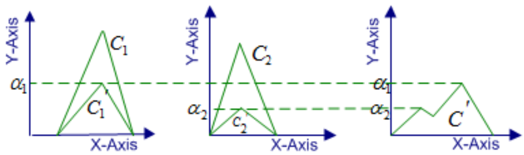

Figura 2.16: Reprezentarea grafică a metodei de inferenţă Larsen

Mulţimea rezultat $C^{'}$ este o mulţime fuzzy şi drept consecinţă dacă dorim să obţinem un control deterministic trebuie să defuzzificăm această mulţime.

Metoda Larsen

Această metodă foloseşte operatorul *produs* pentru implicaţie şi operatorul *max-produs* pentru compunere. După cum s-a demonstrat mai sus

$$\varphi_{C_i'}(w) = \alpha_i \bullet \varphi_{C_i}(w) \text{ cu } i = \overline{1,2}$$

iar mulţimea de ieşire a sistemului $C^{'}$ este reuniunea mulţimilor $C_1^{'}$ şi $C_2^{'}$. Grafic, ieşirea unui sistem cu două reguli ce foloseşte metoda de inferenţă Larsen este prezentată în Figura 2.16.

Metoda Tsukamoto

În această metodă concluzia fiecărei reguli este reprezentată printr-o mulţime fuzzy care are ca funcţie de apartenenţă $\varphi_{C_i}(w)$ o functie monotonă. De aceea ieşirea fiecărei reguli este o valoare crisp w_i acestă valoare calculându-se în funcţie de gradele de potrivire α_i ale intrărilor în raport cu valorile lingvistice din premizele R-bazei.

$$w_i = \varphi_{C_i}^{-1}(\alpha_i) \text{ cu } i = \overline{1,2}$$

Rezultatul final este media ponderată a ieşirilor fiecărei reguli.

$$w_0 = \frac{\alpha_1 w_1 + \alpha_2 w_2}{\alpha_1 + \alpha_2}$$

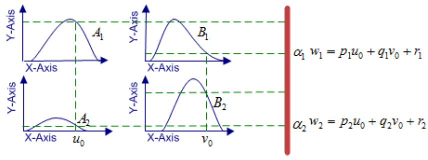

Figura 2.17: Reprezentarea grafică a metodei de inferenţă TSK

Cum rezultatul aplicării fiecărei reguli din bază este o valoare crisp, modelul Tsukamoto combină aceste ieşiri crisp într-o medie ponderată obţinându-se pentru concluzia finală tot o valoare crisp. În consecinţă, folosindu-se această metodă se elimină consumul de timp necesar defuzzificării.

Metoda TSK

Această metodă a fost propusă de Takagi, Sugeno şi Kang. O regulă fuzzy în acest model are forma:

$$\text{IF } u \text{ is } A \text{ and } v \text{ is } B \text{ THEN } w = f(u, v)$$

unde A şi B sunt mulţimi fuzzy iar $f(u, v)$ este o funcţie crisp. De obicei $f(u, v)$ este un polinim de variabile u şi v de aceea aceasta metodă necesită intrări singleton.

Să presupunem că cele două reguli ale R-bazei au următoarea formă:

R_1 :IF u is A_1 and v is B_1 THEN $w_1 = f_1(u, v) = p_1u+q_1v+r_1$

R_2 :IF u is A_2 and v is B_2 THEN $w_2 = f_2(u, v) = p_2u+q_2v+r_2$

unde p_i, q_i, r_i sunt constante.

Valoarea obţinută în urma aplicării regulii R_i pentru intrarea singleton u_0 şi v_0 este valoarea crisp $f_i(u_0, v_0)$ cu gradul de potrivire α_i. Rezultatul agregării acestor ieşiri este dat de media ponderată:

$$w_0 = \frac{\alpha_1 f_1(u_0,v_0)+\alpha_2 f_2(u_0,v_0)}{\alpha_1+\alpha_2} = \frac{\alpha_1 w_1+\alpha_2 w_2}{\alpha_1+\alpha_2}$$

La fel ca şi metoda Tsukamoto ieşirea w_0 nu necesită defuzzificare fiind o valoare crisp.

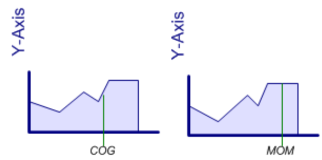

Figura 2.18: Defuzzificare

2.7.3 Metode de defuzzificare

Pentru sistemele fuzzy care obţin concluzia finală sub forma unei mulţimi fuzzy se pune de multe ori problema conversiei acesteia la o valoare crisp. Această conversie fuzzy-crisp este realizată de modulul de defuzzificare.

Două din cele mai utilizate tehnici de defuzzificare sunt *metoda centrului de greutate* (center of gravity:COG) şi *media maximului* (mean of maxima: MOM).

În metoda centrului de greutate valoarea crisp de ieşire este variabila a cărei apartenenţă la mulţimea fuzzy de ieşire se află în centrul de greutate al valorilor funcţiei de apartenenţă.

$$y_{COG} = \frac{\int x\varphi(x)dx}{\int \varphi(x)dx}$$

Folosind media maximului valoarea crisp de ieşire se alege din mulţimea valorilor care au apartenenţă maximă la mulţimea fuzzy care este supusă defuzzificării.

Exemplificare:

Considerăm sistemul fuzzy cu două intrări şi o singură ieşire care modelează relaţia:

$$\frac{presiune \cdot volum}{temperatura} = const$$

care pe baza valorilor de intrare pentru temperatură şi volum calculează valoarea corespunzătoare presiunii.

Înainte de a proiecta regulile sistemului trebuie să definim variabilele fuzzy: *temperatura, volum* şi *presiune*. Astfel definim variabila lingvistică *temperatura* prin sistemul (t, $T(t)$, U, G_t, M_t) unde:

- U=[0,120] măsurat în grade Celsius

- T(t)= {*mică, sub medie, medie, peste medie, mare*} - mulţimea termenilor variabilei t

- $G_t(t)$: $\begin{array}{l} t \rightarrow mică|medie|mare \\ t \rightarrow sub\ t|peste\ t, dacă\ t = medie \end{array}$

- $M_t(mică) = \{\ (u, \varphi_{mică}(u))\ |\ u \in [0, 120]\}$

$$\varphi_{mică}(u) = \begin{cases} 1,\ dacă\ u \in [0, 20] \\ \frac{40-u}{20},\ dacă\ u \in [20, 40] \\ 0,\ altfel \end{cases}$$

$M_t(medie) = \{\ (u, \varphi_{medie}(u))\ |\ u \in [0, 120]\}$

$$\varphi_{medie}(u) = \begin{cases} \frac{u-40}{20},\ dacă\ u \in [40, 60] \\ \frac{80-u}{20},\ dacă\ u \in [60, 80] \\ 0,\ altfel \end{cases}$$

$M_t(mare) = \{\ (u, \varphi_{mare}(u))\ |\ u \in [0, 120]\}$

$$\varphi_{mare}(u) = \begin{cases} \frac{u-80}{20},\ dacă\ u \in [80, 100] \\ 1,\ dacă\ u \in [100, 120] \\ 0,\ altfel \end{cases}$$

$M_t(sub\ medie) = \{\ (u, \varphi_{sub\_medie}(u))\ |\ u \in [0, 120]\}$

$$\varphi_{sub\_medie}(u) = \begin{cases} \varphi_{medie}(u + 20) + 1,\ dacă\ u \in [20, 40] \\ \varphi_{medie}(u + 20) - 1,\ dacă\ u \in [40, 60] \\ 0,\ altfel \end{cases}$$

$M_t(peste\ medie) = \{\ (u, \varphi_{peste\_medie}(u))\ |\ u \in [0, 120]\}$

$$\varphi_{peste\_medie}(u) = \begin{cases} \varphi_{medie}(u - 20) - 1,\ dacă\ u \in [60, 80] \\ \varphi_{medie}(u - 20) + 1,\ dacă\ u \in [80, 100] \\ 0,\ altfel \end{cases}$$

Variabila lingvistică *volum* este definită prin sistemul: (v, $T(v)$, V, G_v, M_v) unde:

- V={10,15,30}

- T(v)= {*mic, mediu, mare*}

- $G_v(v) : v \rightarrow mic\ |mediu\ |mare$

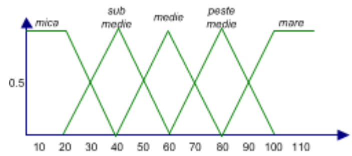

Figura 2.19: Variabila lingvistică *temperatura*

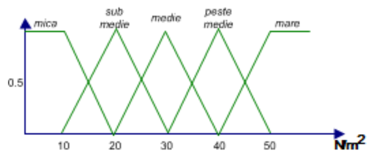

Figura 2.20: Variabila lingvistică *presiunea*

- $M_v(mic) = 1/10 + 0.6/15 + 0/30$

  $M_v(mediu) = 0.6/10 + 1/15 + 0/30$

  $M_v(mare) = 0/10 + 0.4/15 + 1/30$

Variabila lingvistică *presiunea* se defineşte de o manieră asemănătoare celei folosite în cazul variabilei *temperatura*: $(p, T(p), W, G_w, M_w)$ unde:

- W=[0,60] măsurat în N/m^2

- T(p)={*mică, sub medie, medie, peste medie, mare*}

Deoarece intrările *temperatura* şi *volum* au mulţimea termenilor de 5, respectiv 3 elemente obţinem că R-baza sistemului trebuie să aibă 5×3 relaţii câte una pentru fiecare posibilă combinaţie de intrări. Considerând următoarele notaţii pentru valorile lingvistice folosite de sistem: S(mică), BM(sub medie), M(mediu), AM(peste medie), B(mare) atunci setul complet de reguli al sistemului este:

volum\temp	**S**	**BM**	**M**	**AM**	**B**
S	*S*	*S*	*M*	*AM*	*B*
M	*S*	*S*	*BM*	*M*	*M*
B	*S*	*S*	*S*	*S*	*BM*

Vom prezenta inferenţa în acest sistem considerând intrările crisp: $t_0 = 77°C$ şi $v_0 = 10m^3$. Observăm că:

$\varphi_{medie}(t_0) = 0.15, \varphi_{peste\_medie}(t_0) = 0.85$

$\varphi_{mic}(v_0) = 1, \varphi_{mediu}(v_0) = 0.6$

Rezultă că următoarele reguli vor fi aplicate:

R_{13} : IF t este *medie* AND v este *mic* THEN p este *medie* $\Rightarrow \alpha_{13} = \min(\varphi_{medie}(t_0), \varphi_{mic}(v_0)) = \min(0.15, 1) = 0.15$

R_{14} : IF t este *peste medie* AND v este *mic* THEN p este *peste medie* $\Rightarrow \alpha_{14} = \min(\varphi_{peste\_medie}(t_0), \varphi_{mic}(v_0)) = \min(0.85, 1) = 0.85$

R_{23} : IF t este *medie* AND v este *mediu* THEN p este *sub medie* $\Rightarrow \alpha_{23} = \min(\varphi_{medie}(t_0), \varphi_{mediu}(v_0)) = \min(0.15, 0.6) = 0.15$

R_{24} : IF t este *peste medie* AND v este *mediu* THEN p este *medie* $\Rightarrow \alpha_{24} = \min(\varphi_{peste\_medie}(t_0), \varphi_{mediu}(v_0)) = \min(0.85, 0.6) = 0.6$

Aşa cum am spus există două tipuri de implicaţie foarte des folosite: implicaţia *Mamdani* şi implicaţia *Larsen*.

Implicaţia Mamdani	*Implicaţia Larsen*
$\varphi_{OUTPUT\ R13}(p) = 0.15 \wedge \varphi_{medie}(p)$	$\varphi_{OUTPUT\ R13}(p) = 0.15 \bullet \varphi_{medie}(p)$
$\varphi_{OUTPUT\ R14}(p) = 0.85 \wedge \varphi_{peste\_medie}(p)$	$\varphi_{OUTPUT\ R14}(p) = 0.85 \bullet \varphi_{peste\_medie}(p)$
$\varphi_{OUTPUT\ R23}(p) = 0.15 \wedge \varphi_{sub\_medie}(p)$	$\varphi_{OUTPUT\ R23}(p) = 0.15 \bullet \varphi_{sub\_medie}(p)$
$\varphi_{OUTPUT\ R24}(p) = 0.6 \wedge \varphi_{medie}(p)$	$\varphi_{OUTPUT\ R24}(p) = 0.6 \bullet \varphi_{medie}(p)$

pentru $p \in [0, 60]$.

Aşa cum s-a precizat concluzia finală şi anume concluzia corespunzătoare celor patru reguli care au fost aplicate este reuniunea concluziilor parţiale:

$$\varphi_{OUTPUT}(p) = \varphi_{OUTPUT\ R13}(p) \cup \varphi_{OUTPUT\ R14}(p) \cup \varphi_{OUTPUT\ R23}(p) \cup \varphi_{OUTPUT\ R24}(p)$$

Aceste mulţimi fuzzy de ieşire pot fi defuzificate în cazul în care se doreşte ca ieşirea sistemului să fie de aceeaşi natura cu intrarea acestuia - şi anume o valoare crisp.

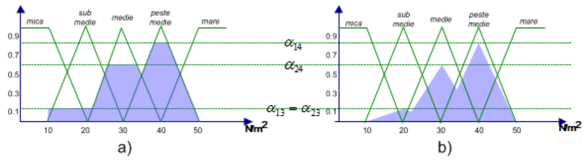

Figura 2.21: Ieşirea sistemului folosind implicaţia *Mamdani* (a) şi *Larsen* (b)

Vom aplica metoda de defuzificare COG pentru mulţimea fuzzy obţinută cu implicaţia Mamdani şi avem:

$$y_{COG} = \frac{\int x\varphi(x)dx}{\int \varphi(x)dx}$$

$\int \varphi(x)dx = \int_{10}^{11.5}(\frac{x}{10} - 1)dx + \int_{11.5}^{21.5} 0.15dx + \int_{21.5}^{26}(\frac{x}{10} - 2)dx + \int_{26}^{34} 0.6dx + \int_{34}^{35}(4 - \frac{x}{10})dx + \int_{35}^{38.5}(\frac{x}{10} - 3)dx + \int_{38.5}^{41.5} 0.85dx + \int_{41.5}^{50}(5 - \frac{x}{10})dx = 20.175$

$\int x\varphi(x)dx = \int_{10}^{11.5}(\frac{x^2}{10} - x)dx + \int_{11.5}^{21.5} 0.15xdx + \int_{21.5}^{26}(\frac{x^2}{10} - 2x)dx + \int_{26}^{34} 0.6xdx + \int_{34}^{35}(4x - \frac{x^2}{10})dx + \int_{35}^{38.5}(\frac{x^2}{10} - 3x)dx + \int_{38.5}^{41.5} 0.85xdx + \int_{41.5}^{50}(5x - \frac{x^2}{10})dx = 383.725$

Deci obţinem: $y_{COG} = 19.01$

Bibliografie

[1] **Ion Vaduva, Grigore Albeanu** *Introducere in Modelarea Fuzzy*, Editura Universităţii din Bucureşti, 2004

[2] **L.A. Zadeh** *Fuzzy Sets* Information and control, vol. 8, pp. 338 - 353, 1965

[3] **R. Yager** *Fuzzy logic in control*, Ph. D. dissertation, Technical University of Delft, 1995

[4] **Carles Garigga Berga** *A new approach to the synthesis of intelligible fuzzy models from input-output data*, Ph. D. dissertation, Universitat Ramon Llull, 2005

[5] **R. Jang, C. T. Sun, E. Mizutani** *Neuro-Fuzzy and Soft-Computing*, First Edition, Prentice Hall, 1997

[6] **F. Chevrie, F. Guely** *Fuzzy Logic*, Cahier technique no. 191, 1998

[7] **D. Dubois, H. M. Prade**, *Fuzyy sets and systems: theory and applications*, New-York Accademic Press, 1980

[8] Depatament of Computing Science, Umea University, Sweden, *Fuzzy Systems - Lecture notes*

Capitolul 3

Sisteme fuzzy pentru control inteligent

3.1 Mulţimi vagi

Noţiunea de *mulţime vagă* a fost introdusă în matematică şi teoria sistemelor de L.A. Zadeh în 1965 sub denumirea de *mulţime fuzzy* care în traducere înseamnă mulţime neclară, estompată şi se foloseşte cu sensul de *vag, imprecis*. Acestui domeniu îi sunt consacrate deja lucrări fundamentale recunoscute.

Pornind de la concepţia clasică cu privire la mulţime şi element al unei mulţimi, se poate susţine că noţiunea de *mulţime fuzzy* reprezintă o abordare dintr-un unghi diferit a conceptului de mulţime, mai precis, între apartenenţa unui element la o mulţime şi nonapartenenţă există o serie de situaţii tranzitorii, de natura continuă, caracterizate de aşa-numitele *grade de apartenenţă*.

Definiţia 3.1.1 *Fie X o mulţime oarecare. Se numeşte mulţime fuzzy (în X) rezultatul unei aplicaţii $F : X \to [0,1]$.*

Mulţimea fuzzy F este caracterizată de funcţia de apartenenţă: $m_F : X \to [0,1]$.

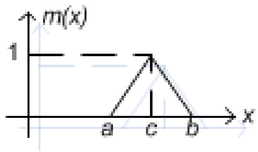

Figura 3.1: Funcţia de apartenenţă triunghiulară

3.1.1 Funcţii de apartenenţă tipice

Pentru descrierea fuzzy a unor fenomene şi procese, aplicaţiile $m_F(x)$ pot admite diferite exprimări analitice. Câteva dintre acestea sunt consacrate în aplicaţii datorită unor facilităţi legate de calculabilitate şi uşurinţa implementării hardware/software. Pentru exemplificare vom considera că funcţiile sunt definite pe intervalul $[a, b]$ şi $c = (a + b)/2$.

Funcţia de apartenenţă triunghiulară

Expresia analitică a acestei funcţii de apartenenţă este identică cu ecuaţia dreptei ce trece prin punctele de coordonate $(a, 0)$ şi $(c, 1)$ pentru $x \in [a, c]$ şi cu ecuaţia dreptei ce trece prin punctele de coordonate $(c, 1)$ şi $(b, 0)$, pentru $x \in (c, b]$. Deci:

$$m(x) = \begin{cases} (x - a)/(c - a), a \leq x \leq c \\ 1 - (x - c)/(b - c), c < x \leq b \end{cases}$$

În cazul în care $c = (a + b)/2$ relaţia se poate scrie compact sub forma:

$$m(x) = 1 - 2\frac{|x-c|}{b-a}$$

Funcţia de apartenenţă trapezoidală

Expresia analitică a acestei funcţii de apartenenţă se obţine simplu observând că trapezul rezultă prin intersecţia unui triunghi de forma celui din Figura 3.1, având înălţimea $h_t > 1$ şi dreapta de ecuaţie $m(x) = 1$. Astfel:

$$m(x) = \min\left[1, h_t(1 - 2\tfrac{|x-c|}{b-a})\right], \text{ cu } h_t > 1.$$

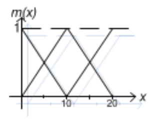

Figura 3.2: Funcţia de apartenenţă trapezoidală

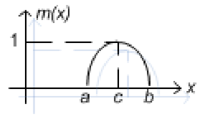

Figura 3.3: Funcţia de apartenenţă parabolică

Funcţia de apartenenţă parabolică

O parabolă având axa de simetrie verticală şi vârful în punctul (x_0, m_0) de pe această axă este descrisă de ecuaţia:

$$m(x) - m_0 = -2p(x - x_0)^2$$

Din condiţiile ca parabola să aibă vârful în punctul $(c, 1)$ şi să intersecteze axa abciselor în punctele de coordonate $(a, 0)$ şi $(b, 0)$ rezultă expresia analitică a funcţiei de apartenenţă parabolică din Figura 3.3. Aceasta este:

$$m(x) = 1 - 4\left(\frac{x-c}{b-a}\right)^2$$

Funcţia de apartenenţă tip armonic

O variantă a acesteia se poate construi cu ajutorul funcţiei sinusoidale de amplitudine 1 şi fază 0:

$$m(x) = \sin(x), \text{ cu } x \in [0, \pi],$$

pentru care, punând condiţiile $m(c) = 1$ şi $m(a) = m(b) = 0$ rezultă expresia analitică:

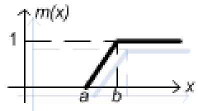

Figura 3.4: Funcţia de apartenenţă de tip *clopot*

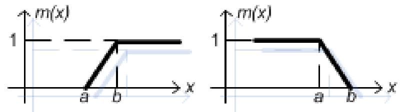

Figura 3.5: Funcţii de apartenenţă de tip saturaţie

$$m(x) = \sin\left(\pi \frac{x-a}{a-b}\right), \text{ cu } x \in [a, b]$$

având reprezentarea grafică similara cu cea din Figura 3.3.

Funcţia de apartenenţă de tip clopot

Expresia analitică a funcţiei de tip "*clopotul lui Gauss*" caracterizată prin dispersia σ, amplitudinea A şi centrată pe dreapta de ecuaţie $x = x_0$ este:

$$m(x) = Ae^{-\frac{(x-x_0)^2}{2\sigma^2}}$$

Pentru funcţia de apartenenta din Figura 3.4 cu amplitudinea $A = 1$, dispersia $\sigma = b-a$ şi centrată pe dreapta de ecuaţie $x = c$, se obţine expresia:

$$m(x) = e^{-\frac{(x-c)^2}{2(b-a)^2}}$$

Funcţii de apartenenţă de tip saturaţie

Acestea se definesc ca funcţii de tip rampă la dreapta sau la stânga, cuplate în partea extremă cu porţiunea de saturaţie. Exprimarea analitică a acestor funcţii de apartenenţă se face pe intervale, după cum urmează:

$$m(x) = \begin{cases} 0, x < a \\ \frac{x-a}{b-a}, a \leq x \leq b \\ 1, x > b \end{cases} \text{ sau } m(x) = \begin{cases} 1, x < a \\ -\frac{x-a}{b-a}, a \leq x \leq b \\ 0, x > b \end{cases}$$

Dependenţa liniară de pe intervalul $[a, b]$ poate fi înlocuită cu oricare alt tip de funcţie (polinomială, exponenţială, etc.) asigurându-se trecerea (cu derivata continuă) de la valorile 0 şi/sau 1 la valorile apropiate.

3.1.2 Operaţii cu mulţimi fuzzy

Operaţiile uzuale din teoria clasică a mulţimilor se pot reface în cazul mulţimilor fuzzy în termenii funcţiei de apartenenţă:

- *mulţimea vidă* $\emptyset \subseteq X$ este caracterizată de $m_{\emptyset}(x) = 0$, $x \in X$
- *mulţimea totală* X este caracterizată de $m_X(x) = 1$, $x \in X$
- două mulţimi fuzzy sunt egale, dacă funcţiile lor de apartenenţă sunt egale, adică: $M = N \Leftrightarrow m_M = m_N$
- mulţimea fuzzy M este conţinută în mulţimea fuzzy N, adică: $M \subseteq N \Leftrightarrow m_M \leq m_N$ (relaţia de ordine punctuală între funcţii)
- între mulţimile M şi N se definesc operaţiile:
  reuniune: $M \cup N$ cu $m_{M \cup N}(x) = \max(m_M(x), m_N(x))$
  intersecţie: $M \cap N$ cu $m_{M \cap N}(x) = \min(m_M(x), m_N(x))$
  complementară: C_M, cu $m_{CM}(x) = 1 - m_M(x)$
- *produsul algebric* al mulţimilor M şi N, notat $M \cdot N$ este caracterizat de funcţia de apartenenţă: $m_{M \cdot N} = m_M \cdot m_N$
- *suma algebrică* a mulţimilor M şi N, notată $M + N$ este caracterizată de funcţia de apartenenţă: $m_{M+N} = m_M + m_N$

3.1.3 Evenimente fuzzy, probabilitate şi entropie

Se numeşte *eveniment fuzzy* în spaţiul euclidian n-dimensional R^n, o mulţime fuzzy $E \subseteq R^n$, a cărei funcţie de apartenenţă, m_E, este măsurabilă Borel.

Se numeşte *probabilitate* a unui eveniment fuzzy E, funcţia definită astfel:

$$P(E) = \int_R m_E(x) dP(x)$$

Două evenimente fuzzy se numesc *independente*, dacă: $P(E \cdot F) = P(E) \cdot P(F)$

Se numeşte *entropie* a mulţimii fuzzy $A \subset X = \{x_1, ..., x_n\}$ mărimea:

$$I^P(A) = -\sum m_A(x_i) \cdot p_i \cdot \ln p_i$$

unde $P = \{p_1, ..., p_n\}$ reprezintă distribuţia de probabilităţi.

Se numeşte *entropie totală* expresia:

$$I_t = -\sum p_i \cdot \ln p_i + \sum p_i S(m_A(x_i)),$$

unde $S(m_A(x_i))$ reprezintă o măsură a gradului de incertitudine.

Observaţie: I_t se poate interpreta drept media totală de incertitudine de a prevedea elementul lui X, care apare dintr-o experienţă (proces) aleator şi de a lua o decizie (notată cu 0 sau 1) asupra acestui element.

Mulţimi fuzzy temporale

Mulţimile fuzzy variabile în timp reprezintă un concept cu aplicare în formularea criteriilor pentru luarea decizilor şi reflectă caracterul *dinamic* al acestor procese. Caracterul temporal al mulţimilor fuzzy este exploatat practic în cadrul sistemelor fuzzy adaptive.

Definiţia 3.1.2 *Fie $\{X(t)\}_{t \in T}$ o familie de mulţimi şi $\{A(t)\}_{t \in T}$ submulţimi fuzzy, $A(t) \subseteq X(t)$. Numim familia $\{A(t)\}_{t \in T}$ mulţime fuzzy dependentă de timp.*

Funcţia de apartenenţă se defineşte: $m_{A(t)} : X(t) \to [0, 1]$, $t \in T$ iar scopul principal al introducerii variabilei de timp este de a analiza situaţii în care parţi ale problemei se modifică în timp.

3.2 Logica fuzzy de tip lingvistic

Logica fuzzy este o generalizare a logicii clasice bivalente, înlocuind caracterul discret al acesteia (în 0 şi 1) cu unul de natură continuă. Fundamentul logicii fuzzy îl constituie aşa-numitele *logici polivalente* introduse şi studiate de J. Lukasiewicz, în urma unor cercetări legate de studiul modalităţilor.

Presupunând că $V_1, V_2, ..., V_n$ sunt variabile logice, în logica fuzzy acestea iau valori în intervalul [0,1].

Definiţia 3.2.1 *Orice variabilă V_i este o formulă fuzzy.*

Dacă $P, Q, ...$ sunt formule în logica fuzzy, valorile logice (de adevăr) ale compuselor $P \vee Q, P \wedge Q, \overline{P}$ se stabilesc astfel:

$A(P \vee Q) = \max(A(P), A(Q))$

$A(P \wedge Q) = \min(A(P), A(Q))$

$A(\overline{P}) = 1 - A(P)$

$A(P \to Q) = \min(1 - A(P) + A(Q), 1)$

Observaţie. Este evident că acest mod de a vedea lucrurile coincide cu cel din logica bivalentă, unde $A(P) \in \{0, 1\}$, oricare ar fi propoziţia P.

Logica fuzzy este de tip logică continuă, deoarece variabilele logice iau valori de adevăr în intervalul [0,1]. Acest fapt atrage existenţa unor elemente particulare cu privire la *variabilele lingvistice*: relaţie de *implicaţie fuzzy* şi noţiunea de *inferenţă fuzzy*.

Un alt tip de logică continuă a fost propusă de H. Reichenbach şi se defineşte astfel:

$A(P \vee Q) = A(P) + A(Q) - A(P) \cdot A(Q)$

$A(P \wedge Q) = A(P) \cdot A(Q)$

$A(\overline{P}) = 1 - A(P)$

$A(P \to Q) = 1 - A(P) + A(P) \cdot A(Q)$

3.2.1 Variabile lingvistice

Variabilele fuzzy sunt mărimi fuzzy asociate celor deterministe. Echivalentul valorii scalare în sens determinist este pentru o variabilă fuzzy *gradul lingvistic* (*eticheta,atributul*) asociat acesteia. Astfel, aşa cum pentru logica bivalentă, valorii deterministe ”1” i se asociază atributul ADEVARAT, iar lui ”0” eticheta FALS, în logica fuzzy, pentru variabila deterministă *număr real pozitiv* variabila lingvistică asociata poate fi, de exemplu, *distanţa* dintre două puncte, care poate avea gradele lingvistice MICĂ, MEDIE, MARE sau FOARTE MICĂ, MICĂ, MEDIE, MARE, FOARTE MARE. Domeniul de valori ale mărimii deterministe se numeşte *univers de discurs*.

Fiecărui atribut al unei variabile lingvistice îi este asociat o funcţie de apartenenţă, a cărei valoare (în sens determinist) indică nivelul de încredere cu ajutorul căruia unei valori deterministe i se poate asocia acel atribut al variabilei lingvistice. De exemplu, considerând pentru variabila lingvistică *distanţa* trei grade lingvistice: MICĂ, MEDIE, MARE, asociem acestora funcţii de apartenenţă tipice, aşa cum se arata în Figura 3.6. Se observă că funcţia de apartenenţă pentru gradul lingvistic MICĂ este o dreaptă, care indică faptul că între 0 şi 10 km distanţele sunt considerate mici, cu diferite niveluri de încredere situate între 0 şi 1: cu cât valoarea deterministă este mai aproape de 0, cu atat gradul de apartenenţă al acesteia la eticheta respectivă este mai mare.

Similar, pentru gradul lingvistic MEDIE se consideră nivelul maxim de apartenenţă la această categorie, corespunzător valorii 10 km. Valorile mai mici sau mai mari decât aceasta duc la scăderea nivelului de încredere relativ la atributul MEDIE. Gradul lingvistic MARE se aplică distaţelor mai mari de 10 km, cu niveluri de încredere din ce în ce mai ridicate, pe masură ce valoarea creşte până la 20 de km, iar pentru orice valoare peste aceasta, distanţa este considerată MARE cu nivelul de încredere maxim.

Pentru funcţiile de apartenenţă din Figura 3.6 se poate afirma, de exemplu, că o distanţă având valoarea 5 km este MICĂ cu nivelul de încredere (valoarea funcţiei de apartenenţă) $m_{MICA}(5) = 0.50$ dar este şi MEDIE cu nivelul de încredere $m_{MEDIE}(5) = 0.44$ şi este şi MARE cu nivelul de încredere 0. Numărul atributelor unei variabile lingvistice şi funcţiile lor de apartenenţă depind de natura aplicaţiei.

3.2.2 Implicaţii pentru logica fuzzy

În logica fuzzy implicaţia este o operaţie de compunere a formulelor (variabilelor) fuzzy, în sensul corelării a două categorii de evenimente, denumite *premise*, respectiv *consecinţe*. Implicaţia fuzzy este similară, dar nu corespunde pe deplin compunerii funcţiilor din cazul determinist şi se referă la evaluarea gradelor lingvistice ale unei submulţimi fuzzy Q, care este consecinţa logică sau funcţională a unei submulţimi

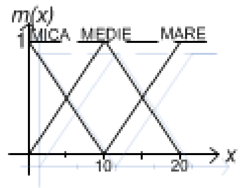

Figura 3.6: Variabila lingvistică *distanţa* cu trei valori lingvistice

fuzzy P. Rezultatul unei implicaţii fuzzy este, de asemenea, o submulţime fuzzy notată $Q^{'} \equiv P \rightarrow Q$, care are aceleaşi grade lingvistice ca şi Q, dar funcţiile ei de apartenenţă, ce exprimă gradul de adevăr $A(Q^{'}) \equiv A(P \rightarrow Q)$, rezultă în urma unor calcule algebrice efectuate asupra valorilor funcţiilor de apartenenţă corespunzătoare gradelor lingvistice ce compun implicaţia fuzzy. Prin urmare, considerând formulele fuzzy:

$P : x\ este\ MARE$

$Q : y\ este\ MIC$

unde x, respectiv y, reprezintă variabila deterministă aparţinând universului de discurs al submulţimii P, respectiv Q, se exprimă implicaţia fuzzy:

$Q^{'} \equiv P \rightarrow Q \Leftrightarrow DAC\breve{A}\ x\ este\ MARE, ATUNCI\ y\ este\ MIC$

Considerând $m_P(x)$, respectiv $m_Q(y)$, funcţiile de apartenenţă ce caracterizează mulţimile fuzzy P şi Q, se pune problema determinării funcţiei de aparteneţă $m_{Q'}(x, y) = m_{P\rightarrow Q}(x, y)$. Cele mai utilizate definiţii ale implicaţiei fuzzy sunt:

a) implicaţia în sens Mamdani:

$$m_{P\rightarrow Q}(x, y) = MIN[m_P(x), m_Q(y)]$$

b) implicaţia booleana:

$$m_{P\rightarrow Q}(x, y) = MAX[1 - m_P(x), m_Q(y)]$$

c) implicaţia Zadeh I:

$$m_{P\rightarrow Q}(x, y) = MIN[1, 1 - m_P(x) + m_Q(y)]$$

d) implicaţia Zadeh II:

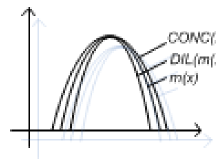

Figura 3.7: Efectul operatorilor "concentrator" şi "dilatator"

$$m_{P\to Q}(x,y) = MIN\{MAX[m_P(x), m_Q(y)], 1 - m_P(x)\}$$

e) implicaţia Larsen:

$$m_{P\to Q}(x,y) = m_P(x) \times m_Q(y)$$

3.2.3 Operatori pentru logica fuzzy

Combinarea mai multor variabile fuzzy, în conformitate cu o anumită logică, conduce la apariţia unor expresii fuzzy cu mai mulţi termeni, legati prin operaţii logice elementare (AND, OR, NOT).

Aşa cum s-a aratat deja, gradele de apartenenţă fuzzy pot fi combinate cu ajutorul *operatorilor de compunere fuzzy* propuşi de Zadeh:

$$m_A \; AND \; m_B = \min(m_A, m_B)$$
$$m_A \; OR \; m_B = \max(m_A, m_B)$$
$$NOT \; m_A = 1 - m_A$$

Dezvoltarea impetuoasa a aplicaţiilor practice în domeniul sistemelor fuzzy a condus la găsirea altor operatori, similari celor utilizaţi la combinarea probabilităţilor:

- operatorul "produs": $m_A \; AND \; m_B = m_A \cdot m_B$
- operatorul "suma": $m_A \; OR \; m_B = m_A + m_B - m_A \cdot m_B$

Pentru a facilita modificările care se impun asupra funcţiilor de apartenenţă în diferite aplicaţii, au fost creaţi doi operatori singulari, unici în logica fuzzy, denumiţi “concentrator”, respectiv “dilatator”. Un operator de concentrare comprimă funcţia de apartenenţă, iar un operator de dilatare realizează o expandare a acesteia. Majoritatea lucrărilor în domeniu definesc principalii operatori de acest tip sub forma:

$$CONC(m) = m^2$$
$$DIL(m) = m^{1/2}$$

Efectul acestor operatori asupra unei funcţii de apartenenţă de referinţă $m(x)$ este ilustrat în Figura 3.7.

3.3 Metodologia de sinteză a automatelor fuzzy

Etapele metodei de sinteză a automatului (controlerului) fuzzy pot fi structurate astfel:

- identificarea **relaţiilor funcţionale** pe structura dată a unui sistem de control
- stabilirea variabilelor (parametrilor) de lucru
- completarea structurii cu elemente impuse de modelul fuzzy

Deşi până acum abordarea fuzzy a proceselor de control nu este pe deplin fundamentată ca teorie de sine stătătoare, numeroase aplicaţii constituie fie o alternativă la problemele de automatizare clasice, fie soluţii practice inedite în diverse domenii.

Majoritatea lucrărilor în domeniu adoptă următoarele faze tipice ale algoritmului de modelare fuzzy:

1. descrierea bazei euristice a problemei
2. alegerea variabilelor de intrare-ieşire
3. stabilirea mulţimilor fuzzy şi a valorilor lingvistice asociate acestora
4. întocmirea bazelor de reguli pentru inferenţe fuzzy
5. stabilirea procedeelor de fuzificare, realizare a inferenţelor logice şi de defuzificare a ieşirilor

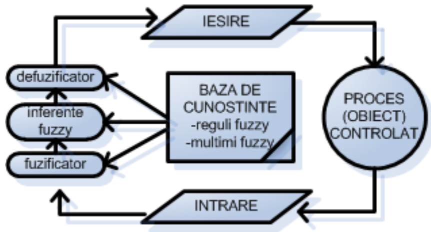

Figura 3.8: Arhitectura unui controler fuzzy

6. adoptarea mecanismelor de defuzificare
7. descrierea mecanismelor de adaptare şi a schemelor de învăţare
8. în forma finală, sistemele fuzzy pot fi implementate sub formă cablată (hardware) în structuri dedicate sau programabile de uz general

3.3.1 Baza euristică a algoritmului

În cadrul acestei faze se urmăreşte o descriere cât mai completă şi fidelă a procesului. Este evident că volumul informaţiei ce trebuie furnizată este direct proporţional cu complexitatea problemei. Orice încercare de formalizare a descrierii proceselor conduce implicit la neglijarea unor aspecte, departând astfel modelul de realitate. Prin urmare apare natural să se pornească cu descrierea procesului în *termeni lingvistici* ca fiind forma primară, cea mai flexibilă şi cu grad mare de completitudine în modelarea evenimentelor. Prin operaţii de structurare a descrierii lingvistice şi aplicarea logicii predicatelor de ordinul întâi, modelele lingvistice constituie baza sistemelor cu IA.

Sistemele fuzzy sunt în esenţă, modele numerice care exploatează cunoştinţe structurate sub forma unor aserţiuni logice (cu valoare de *regulă*). Aşadar, procesul admite în prima fază o descriere lingvistică "grosieră" care ulterior trece prin anumite filtre logice, sub formă de raţionamente sau judecăţi, rezultând un pachet de cunoştinţe cu grad mai mare de structurare, ce concentrează cvasitotalitatea informaţiilor despre proces. Iniţial, rolul de judecător îl are expertul uman (sau grupul de specialişti) şi numai acesta hotărăşte asupra aspectelor ce pot fi minimalizate sau omise în descrierea problemei.

În cadul acestei etape, descrierea lingvistică nu poate fi exhaustivă, dar este esenţială pentru trecerea la celelalte faze ale algoritmului de sinteză a sistemului fuzzy. Tabloul euristic al întregii probleme se va completa în faza întocmirii bazelor de reguli fuzzy, precum şi la adoptarea mecanismelor de funcţionare adaptivă.

3.3.2 Definirea variabilelor de lucru

Pe baza fluxului informaţional şi a schemei sistemului se urmăreşte alegerea ca variabile de lucru a acelor parametri fizici care sunt direct accesibili pe baza măsurătorilor cu ajutorul senzorilor sistemului. Aceşti parametrii pot fi: coordonate de poziţie, viteze de deplasare, forţe, cupluri de forţe, presiuni, temperaturi, curenţi electrici, etc. constituind *variabilele de stare* ale sistemului de control. Aceste mărimi fizice vor fi tratate ca variabile fuzzy. Din mulţimea variabilelor de stare se selecţionează submulţimea *variabilelor de comandă* acestea fiind considerate variabile de ieşire. Complementara acestei submulţimi rezultă drept mulţimea *variabilelor de intrare*. Se presupune că, în general, parametrii de intrare pot fi accesibili sistemului pe trei căi: măsurători directe, estimarea prin calcul, furnizarea ca valori apriorice.

După cum s-a aratat, un atribut major al modelelor fuzzy este acela că nu necesită cunoaşterea explicită a modului în care variabilele de ieşire depind de cele de intrare. Acest lucru este esenţial pentru fenomenele complexe deoarece relaţiile de interacţiune sunt, în general, nedeterministe şi neliniare, şi ca atare mai greu de modelat cu instrumente matematice clasice.

Un alt aspect este acela că variabilele de lucru, privite ca semnale informaţionale, pot fi *continue* sau *discrete* în funcţie de modul furnizării lor de către senzorii sistemului de urmărire. Vom face observaţia ca modul de prelucrare a informaţiei de către algoritmul fuzzy impune tratarea secvenţiala a mărimilor ca în orice structură de calcul numeric, în ciuda faptului că în etapa operării cu reguli, modelele fuzzy se comportă ca procesoare paralele. Eşantionarea variabilelor de lucru se produce fie pe baza perioadei $\triangle T$ de achiziţie a informaţiei de la senzori, fie datorită modului secvenţial de lucru al algoritmului fuzzy cu perioadă de răspuns (cvasiconstantă) δt care depinde de complexitatea calculelor (model & context). De acest aspect se va ţine seama la pregătirea programului şi a datelor de simulare. În cele ce urmează vom considera dependenţa funcţională în domeniul timp ca fiind discretă, de forma:

$$t_k = t_{k-1} + \delta t, k = 1, ..., N$$

Universul de discurs al variabilelor fuzzy alese este mulţimea numerelor reale **R** dar din considerente tehnice aceasta se restricţionează la intervale continue finite. În continuare problema se completează din punct de vedere euristic prin definirea mulţimilor fuzzy asociate variabilelor de lucru, cu ajutorul unor *valori lingvistice* alese convenabil.

Se menţionează că nu este esenţială codificarea în sine a nivelurilor respective, ci doar numărul şi suportul lor intuitiv, care se bazează pe schema de modelare a procesului.

Stabilirea mulţimilor fuzzy asociate variabilelor de lucru se realizează prin adoptarea *funcţiilor de apartenenţă.* În general, adoptarea formei funcţiilor de apartenenţă precum şi repartizarea nivelurilor fuzzy exprimate prin valori lingvistice, pe domeniile de definiţie ale variabilelor de lucru constitue o problemă ce ţine de experienţa şi intuiţia analistului de proces. S-a arătat că funcţiile de apartenenţă pot fi alese dintr-o gamă largă de forme fără ca aceasta să influenţeze semnificativ răspunsul sistemului fuzzy. Din acest punct de vedere se aleg de obicei forme liniare, definind triunghiuri sau trapeze, care necesită calcule mai simple. Algoritmii fuzzy sunt flexibili în ceea ce priveşte alegerea limitelor nivelurilor fuzzy asociate valorilor lingvistice şi implicit a gradului lor de *suprapunere*. Cu alte cuvinte, este important modul de distribuţie (repartiţie) a nivelurilor lingvistice pe universul de discurs al variabilei fuzzy. În general, această problemă se rezolvă prin încercări, în scopul obţinerii celui mai avantajos răspuns al sistemului fuzzy în raport cu un criteriu impus. Problema este abordată în cadrul mecanismelor cu funcţionare *adaptivă*. În acest caz avem de a face cu mulţimi fuzzy dependente de timp. Prin urmare, funcţiile de apartenenţă vor fi definite într-o formă generală, flexibilă, pentru a permite particularizarea lor în funcţie de context. În acest sens, operatorii CONcentrator şi DILatator işi dovedesc utilitatea. Ma mult decât atât, cei doi operatori sunt o particularizare a principiului de obţinere a noilor funcţii de apartenenţă pornind de la o formă dată, numită *funcţie de apartenenţă de bază*. Astfel, considerând, de exemplu, că valoarea fuzzy MICĂ este definită de funcţia de apartenenţa $m_{MICA}(x) = f(x)$ putem aplica un nou subnivel fuzzy - FOARTE ca o funcţie oarecare H_{FOARTE} de funcţia de bază $H_{FOARTE}\,(f(x))$. Atunci, în mod recursiv se pot defini niveluri fuzzy suplimentare:

$$m_{FOARTE\_MICA}(x) = H_{FOARTE}(m_{MICA}(x))$$
$$m_{FOARTE\_FOARTE\_MICA}(x) = H_{FOARTE}(H_{FOARTE}(m_{MICA}(x)))$$

3.3.3 Baza de reguli pentru inferenţe fuzzy

Un *sistem fuzzy* se defineşte în general ca fiind o relaţie funcţională trasată între două spaţii multi-dimensionale unitare:

$$S : I^n \rightarrow I^p$$

în care spaţiul n-dimensional unitar I^n cuprinde toate cele n submulţimi fuzzy ale variabilelor de intrare, iar spatiul p-dimensional I^p conţine toate cele p submulţimi fuzzy ale variabilelor de ieşire. Această asociere între spaţii de mulţimi fuzzy este consacrată sub denumirea *Fuzzy Associative Memories* (FAMs) însa în această lucrare vom folosi denumirea de *Baze de Reguli Fuzzy* (BRF).

Practic, orice sistem poate fi considerat drept o relaţie intrare-iesire al cărei răspuns se prezintă ca o funcţie generică $iesiri = f(intrari)$. Această funcţie constituie fundamentul modelului de bază al sistemelor, care în general reprezintă o exprimare matematică a modului de lucru al acestora. Un sistem bazat pe reguli poate aproxima orice funcţie continuă cu oricâte variabile, cu o precizie oricât de bună.

Consistenţa unei BRF poate fi dedusă pe baza datelor privind punctele de funcţionare a sistemului. Aceste date sunt accesibile prin monitorizarea sistemului existent în funcţiune, al unuia similar sau pe baza unui model analitic. Sistemele fuzzy prezintă o strânsă interdependenţă între funcţiile de apartenenţă ale variabilelor din sistem şi BRF.

Astfel, o BRF se construieste prin punerea în legătură (corelare) logică a mulţimilor fuzzy asociate variabilelor de ieşire cu mulţimile fuzzy ale variabilelor de intrare. În acest scop se porneşte de la strategia generală de descriere a procesului, care eventual se defalcă în substrategii specifice anumitor etape de control. Un rol important îl au aici metodele euristice şi tehnicile ingineriei de cunoştinţe.

Strategia de control se exprimă în termeni lingvistici pe baza cărora se formează *inferenţe logice* care vor constitui regulile BRF (asocieri logice). Inferenţa este operaţia logică care permite trecerea de la premisă la concluzie pe baza raţionamentelor formale. Schema generală a inferenţei este urmatoarea:

$$\boxed{\text{PREMISA}} \xrightarrow{\text{INFERENTA}} \boxed{\text{CONCLUZIE}}$$

Practic, o regulă apare când există o *premisă* cu privire la un eveniment, care implică sau atrage o anumită *consecinţă logică* (*concluzie*). În general, orice proces fizic poate fi modelat pe baza descrierii sale prin reguli. Aceasta presupune stabilirea unui set de premise şi identificarea mulţimii consecinţelor (practic se realizează o identificare a unei mulţimi de relaţii de tip cauză-efect). Prin urmare, o regulă se exprimă prin compunerea premiselor P_i ($i = \overline{1, ..., n}$) cu ajutorul operatorilor logici consacraţi (notaţi generic cu simbolul $\otimes$) şi echivalarea rezultatului cu consecinţa C, adică:

$$P_1 \otimes P_2 \otimes ... \otimes P_n = C$$

3.4 Implementarea software a algoritmilor fuzzy

În momentul de faţă înţelegem prin algoritm o metodă generală de rezolvare a unui anumit tip de problemă, care se poate implementa pe calculator. În acest context un algoritm este esenţa absolută a unei rutine. Algoritmul care stă la baza automatului fuzzy cuprinde, pe lângă partea logică, şi o parte algebrică. Este deci un algoritm de tip mixt, organizat ca o succesiune finită de paşi, cuprinzând mai multe operaţii specifice. Acestea îndeplinesc în totalitate condiţiile de bază pentru a fi implementate pe calculator, adică sunt *definite* şi *efective*. Conform celebrei ecuaţii a lui N. Wirth:

$$\textit{algoritmi} + \textit{structuri de date} = \textit{programe}$$

un program este format dintr-un ansamblu de proceduri, privite ca şi cutii negre şi un ansamblu aferent de date asupra cărora operează procedurile.

Forma pe care o capătă algoritmul într-o implementare pe calculator este subordonată *stilului de programare* şi depinde în special de tipul limbajului. Acceptând standardul de programare structurată, se prezintă în Figura 3.9 schema logică a algoritmului fuzzy pentru un sistem de control. La ieşire programul afişează valori numerice din proces la momentul curent sau, opţional, reprezentări grafice.

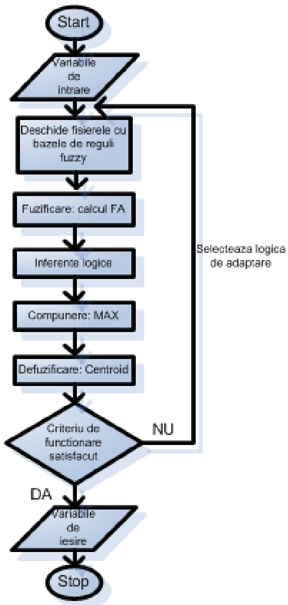

Figura 3.9: Schema logică a unui controler fuzzy

Implementarea software a automatului fuzzy poate fi realizată şi în baza unui *algoritm paralel*. Calculul paralel conferă o dimensiune nouă construcţiei de algoritmi şi programe. Se subliniază că programarea paralelă nu este o simplă extensie a programării seriale şi ca nu toţi algoritmii secvenţiali pot fi paralelizaţi.

3.4.1 Structura unui automat fuzzy

În cazul implementărilor de tip software sinteza automatelor (controlerelor) fuzzy este suficient de flexibilă, fiind practic o problemă de emulare a fazelor tipice ale algoritmului, pentru modelul problemei date. Abilitatea de a programa o problemă este în acest caz foarte importantă. O facilitatate deosebită o oferă, în acest scop, pachetele de programe dedicate (tools-urile) pentru proiectarea sistemelor fuzzy. Totuşi sunt necesare câteva consideraţii, de care trebuie să se ţină seama la structurarea unui sistem de control fuzzy, indiferent de forma sub care va fi implementat.

Configurarea unui controler fuzzy, destinat conducerii unui proces, ţine seama de descompunerea convenţională a dinamicii acestuia, corespunzatoare strategiilor de evoluţie adoptate în etapa de modelare.

Din punct de vedere al comenzii, controlerul fuzzy va fi structurat pe canale de comandă în funcţie de tipul sistemului dinamic controlat: *single-input/single-output* (SISO), *multiple-input/multiple-output* (MIMO). Problema independenţei canalelor de comandă se analizează în contextul existenţei cuplajului informaţional. Interdependenţa canalelor este impusă de metodologia conducerii procesului de control ţinându-se seama de ea la descrierea bazei euristice a problemei şi în etapa de alcătuire a bazei de reguli. Acest lucru este important, mai ales la implementarea software, deoarece modul de procesare a informaţiei este tributar principiului secvenţial de operare şi a posibilităţilor limitate de paralelizare a calculelor.

Posibilităţile de implementare direct hardware a sistemelor fuzzy sunt în prezent o realitate datorită apariţiei circuitelor logice fuzzy (CLF), a maşinilor elementare pentru efectuarea de inferenţe fuzzy (MEIF) şi a circuitelor pentru generarea funcţiilor caracteristice cu grad de apartenenţă comandat. Aceste sisteme microelectronice se regăsesc în structura procesoarelor fuzzy dedicate sau de uz general.

Componentele software asociate procesoarelor fuzzy sunt:

- editorul pentru definirea variabilelor I/O, a funcţilor de apartenenţă şi pentru editarea regulilor; de asemenea editorul permite utilizatorului să specifice universul de discurs, rezoluţia datelor de intrare şi dimensiunea datelor

- compilatorul pentru generarea codului obiect al procesorului

- depanatorul (debuggerul) ce permite execuţia pas cu pas şi monitorizarea programelor

- simulatorul pentru simularea sistemului fuzzy în buclă închisă şi afişarea grafică a rezultatelor

- instrumente de export pentru aplicaţii fuzzy generate cu procesorul, către alte medii de simulare

Producţia de procesoare fuzzy destinate unor controlere de uz general nu exclude însă implementarea software a algoritmilor fuzzy pe sistemele de calcul de uz comun, ca etapă de cercetare-proiectare în realizarea modelelor fizice.

3.4.2 Proiectarea sistemelor fuzzy cu MATLAB

În mediul software de tip Matlab există toolbox-uri specializate pentru analiza şi proiectarea unor sisteme de control inteligent, din care face parte şi Fuzzy Logic Toolbox (FLT). Acest toolbox este format dintr-un ansamblu de fisiere Matlab (pe doua subdirectoare: FUZZY şi FUZZYDEMOS) cu ajutorul cărora se abordează etapele caracteristice sintezei unui sistem bazat pe inferenţe fuzzy (FIS).

Subdirectorul FUZZY conţine fişiere de tip function, grupate pe următoarele categorii de funcţii şi operaţii specifice:

- Funcţii pentru interfaţa grafică cu utilizatorul (GUI); funcţii pentru editarea sistemului de inferenţe fuzzy (FIS), a funcţiilor de apartenenţă (FA) şi a regulilor utilizate, a diagramelor de inferenţe şi a suprafeţelor de control asociate; funcţii pentru comanda de generare FIS (prin fuzificare - definirea unei FA pentru fiecare variabilă implicată în reguli de tip fuzzy, prin defuzificare - pe baza estimării inferenţelor fuzzy, precum şi prin transferul de parametri între funcţii şi variabile, respectiv prin evaluarea generică a FA şi vizualizarea suprafeţei de control); funcţii pentru implementarea altor rutine (FIS de tip Sugeno, clustere de tip C-means etc.)

- Operaţii referitoare la diferenţa a două FA cu diferite forme (sigmoidală, Gauss, pi, trapezoidală, triunghiulară, Z, S, inclusiv combinaţii ale lor), la concatenarea matricelor, la discretizarea FIS-ului, la evaluarea de FA multiple, la editarea liniilor de text, inclusiv help-uri active

Subdirectorul FUZZYDEMOS (FUZZY LOGIC TOOLBOX DEMOS) conţine diferite aplicaţii demostrative pentru funcţii şi operaţii fuzzy de bază, precum şi modele fuzzy complete ale unor sisteme de control inteligent.

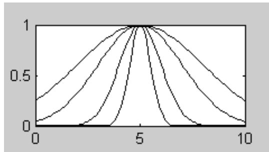

Figura 3.10: Funcţia GAUSSMF

Exemple de funcţii şi operaţii

Funcţii de comandă pentru interfaţa grafică cu utilizatorul

FUZZY pentru ediatrea FIS-ului

MFEDIT pentru editarea funcţiilor de apartenenţă

RULEEDIT pentru editarea regulilor

SURFVIEW vizualiazarea suprafeţelor de control

RULEVIEW vizualizarea regulilor (RULEVIEW(FIS)) şi a diagramelor de inferenţă fuzzy pentru o matrice FIS - RULEVIEW('FILENAME')

Funcţii de apartenenţă (FA)

GAUSSMF(x, PARAMS) - FA de tip Gauss cu flancuri simetrice, unde PARAMS= [c d] este un vector ce determină forma şi poziţia FA prin parametrul de dispersie - s şi de medie - c:

$$GAUSSMF(x, [s\ c]) = \exp[-\frac{(x-c)^2}{2s^2}]$$

Exemplu:

```
x = (0:0.1:10)';
y1=gaussmf(x,[0.5 5]);
y2=gaussmf(x, [1 5]);
y3=gaussmf(x, [2 5]);
y4=gaussmf(x, [3 5]);
subplot(321); plot(x, [y1 y2 y3 y4], 'k');
```

TRIMF(x, PARAMS) - FA de tip triunghi, unde PARAMS = [A B C] este un vector cu trei elemente, care precizează punctele de discontinuitate:

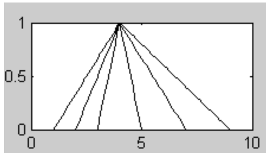

Figura 3.11: Funcţia TRIMF

$$TRIMF(x,[A\ B\ C]) = \begin{cases} 0, x \leq A \\ \frac{x-A}{B-A}, A < x \leq B \\ \frac{C-x}{C-B}, B < x \leq C \\ 0, C < x \end{cases}$$

De remarcat că această FA are întotdeauna o înălţime unitară. Pentru a genera o FA triunghiulară cu dimensiune subunitară se va utiliza comanda TRIMF.

Exemplu:

```
x=(0:0.1:10);
y1=trimf(x,[3 4 5]);
y2=trimf(x,[2 4 7]);
y3=trimf(x,[1 4 9]);
subplot(321), plot(x, [y1 y2 y3], 'k');
```

TRAPMF(x, PARAMS) - FA de tip trapez, unde PARAMS = $[A\ B\ C\ D]$ este un vector de patru elemente, care precizează punctele de discontinuitate ale funcţiei:

$$TRAPMF(x,[A\ B\ C\ D]) = \begin{cases} 0, x \leq A \\ \frac{x-A}{B-A}, A < x \leq B \\ 1, B < x \leq C \\ \frac{D-x}{D-C}, C < x \leq D \\ 0, D < x \end{cases}$$

Dacă avem $A < B < C < D$ funcţia de apartenenţă este trapezoidală. Dacă $A < D$ şi $B \geqslant C$, această FA va avea o formă triunghiulară cu baza AD şi vârful definit de coordonatele:

$$\left(\frac{BD-AC}{D-C+B-A}, 1 - \frac{B-C}{D-C+B-A}\right)$$

Exemplu:

```
x=(0:0.1:10);
```

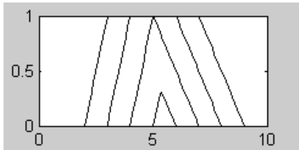

Figura 3.12: Funcţia TRAPMF

```
y1=trapmf(x, [2 3 7 9]);
y2=trapmf(x, [3 4 6 8]);
y3=trapmf(x, [4 5 5 7]);
y4=trapmf(x, [5 6 4 6]);
subplot(321),plot(x, [y1 y2 y3 y4], 'k');
```

Funcţii de comanda pentru FIS

NEWFIS - creează o nouă matrice FIS (Fuzzy Inference System)

Moduri de apelare:

- FIS=NEWFIS(*FisName*) - crează un FIS de tip Mamdani cu numele FisName
- FIS=NEWFIS(*FisName, FisType, andMethod, orMethod, impMethod, aggMethod, defuzzMethod*). Implicit parametrii sunt setaţi astfel: *FisType* la tipul 'Mamdani', *andMethod - 'min', orMethod - 'max', impMethod - 'min', aggMethod - 'max', defuzzMethod - 'centroid'.*

ADDVAR - adaugă o variabilă lingvistică la FIS. Variabilelor li se aplică indici în ordinea în care sunt adăugate. Variabilele de intrare şi de ieşire sunt numărate independent.

Modul de apelare:

FIS2=ADDVAR(*FIS1, varType, varName, varRange*)

Efectul apelării: adaugă variabila specificată prin *varName* la variabilele lingvistice din FIS1 unde:

- *varType* specifică tipul variabilei: 'input' sau 'output'
- *varRange* precizează domeniul variabilei

Exemplu:

```
a=newfis('control');
```

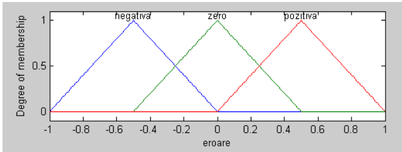

Figura 3.13: Variabila lingvistică "eroare"

```
a=addvar(a, 'input', 'eroare', [-1 1]);
getfis(a, 'input', 1);
```

Rulând secvenţa de program de mai sus se obţine:

```
Name=eroare
NumMFs=0
MFLabels=
Range=[-1 1]
```

ADDMF adaugă o funcţie de apartenenţă la FIS.
Modul de apelare:
FIS2=ADDMF(*FIS1, varType, varIndex, mfName, mfType, mfParams*)
Efectul apelării: adaugă o FA la FIS1 specificând:

- *mfName*: numele funcţiei de apartenenţă
- *varType*: tipul variabilei ('input' sau 'output')
- *varIndex*: indicele variabilei
- *mfType*: forma FA
- *mfParams*: parametrii funcţiei de apartenenţă

Exemplu:

```
a=newfis('control');
a=addvar(a, 'input', 'eroare', [-1 1]);
a=addmf(a, 'input', 1, 'negativa', 'trimf', [-1 -0.5 0]);
a=addmf(a, 'input', 1, 'zero', 'trimf', [-0.5 0 0.5]);
a=addmf(a, 'input', 1, 'pozitiva', 'trimf', [0 0.5 1]);
subplot(211),plotmf(a, 'input', 1);
```

Rezultatul rulării acestui fisier Matlab este prezentat în Figura 3.13:

ADDRULE - adaugă reguli la FIS.

Modul de apelare:

FIS2=ADDRULE(*FIS1,ruleList*)

Efectul apelării: adaugă regulile date de variabila *ruleList* la FIS1, unde *ruleList* este o listă cu mai multe rânduri, fiecare dintre acestea reprezentând o regulă dată.

Dacă sistemul are M intrări şi N ieşiri, trebuie definite în lista de reguli M+N+2 coloane. Primele M coloane se refera la intrările sistemului, iar următoarele N coloane se refera la ieşirile sistemului. Fiecare coloană conţine un număr care se referă la indexul FA al variabilelor de intrare şi ieşire. Coloana M+N+1 conţine ponderea care este aplicată la regula respectivă şi este un număr între 0 şi 1. Coloana M+N+2 conţine 1 dacă operatorul fuzzy pentru compunerea regulilor este AND şi 2 dacă acesta este OR.

Exemplu:

```
a=newfis('control');
a=addvar(a, 'input', 'eroare', [-1 1]);
a=addmf(a, 'input', 1, 'negativa', 'trimf', [-1 -0.5 0]);
a=addmf(a, 'input', 1, 'zero', 'trimf', [-0.5 0 0.5]);
a=addmf(a, 'input', 1, 'pozitiva', 'trimf', [0 0.5 1]);
a=addvar(a, 'input', 'var\_eroare', [-1 1]);
a=addmf(a, 'input', 2, 'negativa', 'trimf', [-1 -0.5 0]);
a=addmf(a, 'input', 2, 'zero', 'trimf', [-0.5 0 0.5]);
a=addmf(a, 'input', 2, 'pozitiva', 'trimf', [0 0.5 1]);
a=addvar(a, 'output', 'comanda', [0 2]);
a=addmf(a, 'output', 1, 'mica', 'trimf', [0 0.5 1]);
a=addmf(a, 'output', 1, 'medie', 'trimf', [0.5 1 1.5]);
a=addmf(a, 'output', 1, 'mare', 'trimf', [1 1.5 2]);
ruleList=[1 1 3 1 2;2 1 3 1 2;3 1 2 1 2;1 2 3 1 2;2 2 2 1 2;
3 2 1 1 2;1 3 2 1 2;3 3 1 1 2];
a=addrule(a, ruleList);
writefis(a,'control');
showrule(a);
```

Se obţine setul de reguli (ponderate unitar):

1. If (eroare is negativa) or (var_eroare is negativa) then (comanda is mare) (1)
2. If (eroare is zero) or (var_eroare is negativa) then (comanda is mare) (1)
3. If (eroare is pozitiva) or (var_eroare is negativa) then (comanda is medie) (1)
4. If (eroare is negativa) or (var_eroare is zero) then (comanda is mare) (1)

5. If (eroare is zero) or (var_eroare is zero) then (comanda is medie) (1)
6. If (eroare is pozitiva) or (var_eroare is zero) then (comanda is mica) (1)
7. If (eroare is negativa) or (var_eroare is pozitiva) then (comanda is medie) (1)
8. If (eroare is pozitiva) or (var_eroare is pozitiva) then (comanda is mica) (1)

DEFUZZ - realizează defuzificarea pentru un FIS.

Modul de apelare:

$valDefuzz$=DEFUZZ($x, mf, type$)

Efectul apelării: returnează în variabila *valDefuzz* valoarea defuzificată a unei FA de tip mf, utilizând strategia de defuzificare precizată de *type*. Valoarea obţinută este abcisa:

- centrului de greutate al ariei pentru *type*='centroid' (tipul implicit)
- bisectoarea ariei pentru *type*='bisector'
- media maximelor pentru *type*='mom'
- celui mai mic dintre maxime, pentru *type*='som'
- celui mai mare dintre maxime, pentru *type*='lom'

Exemplu:

```
x=-10:0.1:10
mf=trapmf(x, [-10 -8 -2 7]);
plot(x, mf, 'b');
axis([min(x) max(x) 0 1.2]);
hold on;
type='centroid';
xx=defuzz(x, mf, type);
plot(xx, 0.0, 'b*');
text(xx, 0.1, type, 'hor', 'center', 'ver', 'top', 'color', 'b');
type='bisector';
xx=defuzz(x, mf, type);
plot(xx, 0.0, 'r*');
text(xx, 0.2, type, 'hor', 'center', 'ver', 'top','color','r');
type='mom';
xx=defuzz(x, mf, type);
plot(xx, 0.0, 'm*');
text(xx, 0.1, type, 'hor', 'center', 'ver', 'top','color','m');
type='som';
```

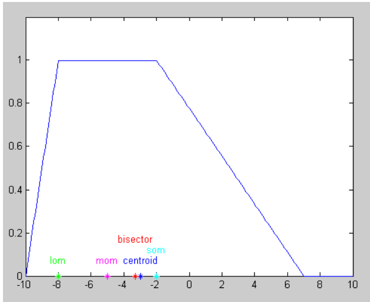

Figura 3.14: Efectul funcţiei DEFUZZ

```
xx=defuzz(x, mf, type);
plot(xx, 0.0, 'c*');
text(xx, 0.15, type, 'hor', 'center', 'ver', 'top', 'color', 'c');
type='lom';
xx=defuzz(x, mf, type);
plot(xx, 0.0, 'g*');
text(xx, 0.1, type, 'hor', 'center', 'ver', 'top', 'color', 'g');
hold off;
```

În Figura 3.14 se prezintă rezultatele obţinute la defuzificarea unei FA de tip TRAPMF cu metodele: *centroid*, *bisector*, *lom*, *som* şi *mom*.

3.4.3 Aplicaţie. Sistem fuzzy pentru analiză spaţială

Una din cele mai importante caracteristici ale sistemelor de informare geografică (SIG) este interpretarea şi rezolvarea problemelor spaţiale. În general, cercetătorii folosesc pachete de statistici de interfaţă SIG pentru analiza spatială. Alte căi ale analizei datelor spaţiale s-au bazat pe logică, automate celulare, reţelele neuronale şi tehnicile inteligenţei artificiale (precum sistemele bazate pe informaţii/ cunoştinţe).

Desi câteva din tehnicile IA au fost folosite pentru analiza datelor spaţiale, este foarte dificil ca un non-expert în tehnicile IA să dezvolte şi să folosească un sistem hibrid între tehnicile SIG şi IA. Motivul este acela că majoritatea tehnicilor IA necesită abilităţi de programare ale computerului pentru a folosi cunoştinţele în sistem. Totuşi, un sistem bazat pe reguli este cea mai folosită tehnică în IA pentru un sistem hibrid cu SIG deoarece are o structură optimă pentru derivarea unui răspuns.

Programele timpurii, precum DENDRAL şi MYCIN, au demonstrat posibilitatea simulării modalităţii în care experţii, precum chimiştii sau fizicienii, îşi rezolvă problemele. După aceste succese, multe sisteme expert au fost dezvoltate pentru aplicaţii tehnice din diverse domenii.

Sistemul de control fuzzy pe care l-am implementat utilizând mediul de programare MATLAB poate fi folosit pentru recunoaşterea obiectelor dintr-o fotografie spaţială. În funcţie de caracteristicile pe care utilizatorul le introduce pentru un anumit detaliu din fotografie sistemul stabileşte dacă acesta este:

- lac
- râu
- teren arabil
- parc
- cartier rezidenţial
- stradă
- corp industrial

Există patru elemente de bază pentru interpretarea fotografiilor aeriene:

- *textura* este frecvenţa schimbărilor de ton în cadrul fotografiilor
- *valoarea spectrală* reprezintă densitatea strălucirii. Este o înregistrare de reflectare a luminii de la suprafaţa pământului pe fotografie
- *aria* se stabileşte în raport cu a celorlaltor obiecte din fotografie.
- *conturul* reflectă cât de regulat sau neregulat este detaliul în cauză

Folosind aceste patru elemente au fost construite regulile din cadrul sistemului. Mai intâi *conturul* şi *valoarea spectrală* sunt folosite pentru a separa utilităţile urbane de alte elemente precum lacuri sau terenuri. *Textura* este folosită pentru a separa râurile de terenuri, corpurile industriale de cele de locuinţe iar *aria* ne permite sa stabilim mărimea obiectului vizat în raport cu celelalte.

Baza de reguli a sistemului

Aşa cum am spus, pe baza celor patru caracteristici ale detaliilor ce pot apărea într-o fotografie spaţială au fost construite regulile controlerului fuzzy. Acestea sunt:

1. If (**contour** is regular) and (**texture** is very_umsmooth) and (**spectral** is medium) and (**area** is medium) then (**Structure** is Residential)

2. If (**contour** is regular) and (**texture** is v_umsmooth) and (**spectral** is medium) and (**area** is large) then (**Structure** is Residential)

3. If (**contour** is regular) and (**texture** is unsmooth) and (**spectral** is deep) and (**area** is medium) then (**Structure** is Park)

4. If (**contour** is regular) and (**texture** is unsmooth) and (**spectral** is deep) and (**area** is small) then (**Structure** is Park)

5. If (**contour** is regular) and (**texture** is smooth) and (**spectral** is light) and (**area** is medium) then (**Structure** is Industrial)

6. If (**contour** is regular) and (**texture** is smooth) and (**spectral** is light) and (**area** is large) then (**Structure** is Industrial)

7. If (**contour** is very_regular) and (**texture** is smooth) and (**spectral** is deep) and (**area** is large) then (**Structure** is Arable Field)

8. If (**contour** is very_regular) and (**texture** is smooth) and (**spectral** is deep) and (**area** is very_large) then (**Structure** is Arable Field)

9. If (**contour** is regular) and (**texture** is very_smooth) and (**spectral** is medium) and (**area** is large) then (**Structure** is Car Road)

10. If (**contour** is irregular) and (**texture** is very_smooth) and (**spectral** is deep) and (**area** is medium) then (**Structure** is River)

11. If (**contour** is very_irregular) and (**texture** is very_smooth) and (**spectral** is deep) and (**area** is small) then (**Structure** is Lake)

După cum se observa *valoarea spectrală* separă unităţile urbane de cele naturale (parcuri, terenuri, râuri). Dacă avem structuri artificiale, ca de exemplu cartiere rezidenţiale sau zone industriale atunci aceastea sunt diferenţiate în primul rând după *textură*. Structurile liniare care pot fi râuri sau strazi sunt diferenţiate după *contur*.

Un exemplu de figură spaţială care este folosită în această versiune a sistemului fuzzy pentru analiza spaţială şi pe baza căreia se pot face raţionamentele implementate în regulile de mai sus este cea din Figura **??**.

****

Sunt câteva limitari în aplicarea acestui sistem bazat pe reguli. Mai întâi, daca un diagnostic tipic sau o sarcina complexa trebuie executata, sunt necesare câteva sute de reguli. Daca regulile necesita mai multe cunostinte în domenii variate, s-ar putea ca regulile sa fie dificil de construit datorita diverselor aspecte din cadrul domeniilor.

În al doilea rând, un sistem expert bazat pe reguli nu poate folosi experientele ce au fost folosite în rezolvarea problemelor similare. Desi o problema este aceeasi sau similara cu cea veche, utilizatorii trebuie sa raspunda din nou la toate întrebarile în acest sistem bazat pe reguli. Nu exista nici o modalitate de a salva un caz similar în sistemul bazat pe reguli. Mentinerea cunostintelor într-un sistem bazat pe reguli este în mod normal un proces manual ce necesita achizitionarea unor cunostinte suplimentare.

În al treilea rând, nu exista exceptii în ceea ce priveste regulile. Regulile trebuie sa explice cunostintele, întrebarile exacte, si raspunsurile exacte. Nu este permis ca utilizatorii sa faca greseli în raspunsuri. Pentru a depasi aceste limitari, exista si alte cai de acces în inteligenta artificiala: rationamentul bazat pe model (RBM) (Davis si Hamscher, 1986) sau rationamentul bazat pe caz (RBC) (Slade, 1991).

Totusi, sistemele bazate pe reguli sunt mult mai indicate pentru rezolvarea problemelor atunci când este dificil sa aduni date despre caz (Althoff si altii 1994). Într-un sistem expert, cunostintele pot fi mai usor implementate în sistemele bazate pe reguli decât în sistemele RBC deoarece regulile pot fi construite printr-o structura de rationament foarte simpla (structura daca-atunci) si într-un limbaj natural.

3.4.4 Sistem fuzzy pentru analiza spatiala - Interfata

Ca si intreg controlerul fuzzy si interfata acestuia este proiectata folosind mediul de programare Matlab. Au fost implementate cinci ferestre de dialog dupa cum urmeaza:

Prima fereastra de dialog sau fereastra care este afisata atunci cand lansam in executie programul are rolul:

- de a afisa fotografia pe baza careia se vor face rationamente
- de a porni setul de intrebari pe baza carora se vor obtine valorile (crisp) pentru variabilele lingvistice utilizate in premisele regulilor
- dupa ce setul de intrebari a fost epuizat si s-au obtinut valorile necesare rationamentului fuzzy prin intermediul acestei ferestre putem afla rezultatul (concluzia)

Odata ce am apasat pe butonul ”Start Reasoning” va fi afisat dialogul ce ne permite selectarea valorii pentru variabila lingvistica *arie*.

Aceasta fereastra afiseaza functiile de apartenenta corespunzatoare valorilor acestei variabile: *small, medium, large, very large* si un control de tip listbox ce permite selectarea valorii numerice corespunzatoare ariei detaliului pentru care dorim sa se efectueze rationamentul. Prin acest control de tip listbox utilizatorul va alege o valoare numerica cat mai apropiata de valoarea ariei obiectului vizat.

Astfel, obiectele sunt considerate ”mici” daca au aria pana in 10000m^2, obiectele ”medii” se presupune ca au aria cuprinsa intre 8000m^2 si 40000m^2, etc.

De asemenea se afiseaza fotografia initiala insa filtrata astfel incat utilizatorul sa poata compara mai usor dimensiunea obiectului vizat cu a celorlaltora. Filtrarea consta in afisarea tuturor detaliilor in nunate ale aceleiasi culori si estomparea neregularitatilor acestora.

Urmatoarea fereastra afisata corespunde variabilei ”contour”. Functiile de apartenenta ale acestei variabile corespund valorilor: *very irregular, irregular, regular, very regular*.

Spre deosebire de dialogul anterior, in acesta fotogafia initiala este afisata astfel incat neregularitatile in conturul detaliilor sunt puse in evidenta:

Cea de-a treia fereastra afisata corespunde variabilei ”texture”. Functiile de apartenenta ale acestei variabile corespund valorilor: *very unsmooth, unsmooth, smooth, very smooth*.

In fereastra corespunzatoare variabilei ”texture” fotografia este modificata astfel incat toate neregularitatile sunt evidentiate: si anume nu numai conturul in plan cat si conturul spatial (tridimensional):

Ultima fereastra corespunde variabilei ”spectral value” continand functiile de apartenenta ale celor trei valori ale acesteia: *deep, medium* si *light*.

Fotografia de aceasta data este modificata astfel incat petele de culoare sunt intensificate sau intunecate mai tare dupa caz.

Pentru exemplificare vom considera urmatorul detaliu in fiecare din cele 5 figuri afisate in cadrul programului (prima fiind cea initiala si celelalte patru corespunzand fiecarei caracteristici in parte):

Astfel daca vom alege pentru variabila *area* valoarea numerica crisp 40000, pentru variabila *contour* valoarea numerica 0.9, pentru variabila *texture* valoarea 1 iar pentru *spectral value* valoarea crisp 0.7 atunci se va aplica urmatoarea regula:

IF AREA is large, CONTOUR is very regular, TEXTURE is very smooth, SPECTRAL VALUE is medium THEN CAR ROAD

deci se va confirma ca detaliul in cauza este o strada.

3.4.5 Sistem fuzzy pentru analiza spatiala - Implementarea

Fisierul de reguli *spatial.m*:

```
a=newfis('spatial');
a.input(1).name='contour'
a.input(1).range=[0 1];
a.input(1).mf(1).name='v_irreg';
```

```
a.input(1).mf(1).type='trapmf';
a.input(1).mf(1).params=[0 0 0.2 0.4];
a.input(1).mf(2).name='irreg';
a.input(1).mf(2).type='trapmf';
a.input(1).mf(2).params=[0.2 0.4 0.4 0.6];
a.input(1).mf(3).name='reg';
a.input(1).mf(3).type='trapmf';
a.input(1).mf(3).params=[0.4 0.6 0.6 0.8];
a.input(1).mf(4).name='v_reg';
a.input(1).mf(4).type='trapmf';
a.input(1).mf(4).params=[0.6 0.8 1 1];
a.input(2).name='texture'
a.input(2).range=[0 1];
a.input(2).mf(1).name='v_unsmooth';
a.input(2).mf(1).type='trapmf';
a.input(2).mf(1).params=[0 0 0.2 0.4];
a.input(2).mf(2).name='unsmooth';
a.input(2).mf(2).type='trapmf';
a.input(2).mf(2).params=[0.2 0.4 0.4 0.6];
a.input(2).mf(3).name='smooth';
a.input(2).mf(3).type='trapmf';
a.input(2).mf(3).params=[0.4 0.6 0.6 0.8];
a.input(2).mf(4).name='v_smooth';
a.input(2).mf(4).type='trapmf';
a.input(2).mf(4).params=[0.6 0.8 1 1];
a.input(3).name='spectral';
a.input(3).range=[0 1];
a.input(3).mf(1).name='light';
a.input(3).mf(1).type='trapmf';
a.input(3).mf(1).params=[0 0 0.25 0.5];
a.input(3).mf(2).name='medium';
a.input(3).mf(2).type='trapmf';
a.input(3).mf(2).params=[0.25 0.5 0.5 0.75];
a.input(3).mf(3).name='deep';
a.input(3).mf(3).type='trapmf';
```

```
a.input(3).mf(3).params=[0.5 0.75 1 1];
a.input(4).name='area'
a.input(4).range=[1000 110000];
a.input(4).mf(1).name='small';
a.input(4).mf(1).type='trapmf';
a.input(4).mf(1).params=[1000 1000 5000 10000];
a.input(4).mf(2).name='medium';
a.input(4).mf(2).type='trapmf';
a.input(4).mf(2).params=[8000 20000 30000 40000];
a.input(4).mf(3).name='large';
a.input(4).mf(3).type='trapmf';
a.input(4).mf(3).params=[30000 50000 80000 100000];
a.input(4).mf(4).name='v_large';
a.input(4).mf(4).type='trapmf';
a.input(4).mf(4).params=[80000 100000 110000 110000];
a.output(1).name='Structure';
a.output(1).range=[0 7];
a.output(1).mf(1).name='Lake';
a.output(1).mf(1).type='trapmf';
a.output(1).mf(1).params=[0 0.5 0.5 1];
a.output(1).mf(2).name='River';
a.output(1).mf(2).type='trapmf';
a.output(1).mf(2).params=[1 1.5 1.5 2];
a.output(1).mf(3).name='Arable Field';
a.output(1).mf(3).type='trapmf';
a.output(1).mf(3).params=[2 2.5 2.5 3];
a.output(1).mf(4).name='Park';
a.output(1).mf(4).type='trapmf';
a.output(1).mf(4).params=[3 3.5 3.5 4];
a.output(1).mf(5).name='Residential';
a.output(1).mf(5).type='trapmf';
a.output(1).mf(5).params=[4 4.5 4.5 5];
a.output(1).mf(6).name='Car Road';
a.output(1).mf(6).type='trapmf';
a.output(1).mf(6).params=[5 5.5 5.5 6];
```

```
a.output(1).mf(7).name='Industrial';
a.output(1).mf(7).type='trapmf';
a.output(1).mf(7).params=[6 6.5 6.5 7];
a.rule(1).antecedent=[3 1 2 2];
a.rule(1).consequent=[5];
a.rule(1).weight=1;
a.rule(1).connection=1;
a.rule(2).antecedent=[3 1 2 3];
a.rule(2).consequent=[5];
a.rule(2).weight=1;
a.rule(2).connection=1;
a.rule(3).antecedent=[3 2 3 2];
a.rule(3).consequent=[4];
a.rule(3).weight=1;
a.rule(3).connection=1;
a.rule(4).antecedent=[3 2 3 1];
a.rule(4).consequent=[4];
a.rule(4).weight=1;
a.rule(4).connection=1;
a.rule(5).antecedent=[3 3 1 2];
a.rule(5).consequent=[7];
a.rule(5).weight=1;
a.rule(5).connection=1;
a.rule(6).antecedent=[3 3 1 3];
a.rule(6).consequent=[7];
a.rule(6).weight=1;
a.rule(6).connection=1;
a.rule(7).antecedent=[4 3 3 3];
a.rule(7).consequent=[3];
a.rule(7).weight=1;
a.rule(7).connection=1;
a.rule(8).antecedent=[4 3 3 4];
a.rule(8).consequent=[3];
a.rule(8).weight=1;
a.rule(8).connection=1;
```

```
a.rule(9).antecedent=[3 4 2 3];
a.rule(9).consequent=[6];
a.rule(9).weight=1;
a.rule(9).connection=1;
a.rule(10).antecedent=[2 4 3 2];
a.rule(10).consequent=[2];
a.rule(10).weight=1;
a.rule(10).connection=1;
a.rule(11).antecedent=[1 4 3 1];
a.rule(11).consequent=[1];
a.rule(11).weight=1;
a.rule(11).connection=1;
writefis(a,'spatial');
```

Bibliografie

[1] **Ion Iancu** *Rules-Based Reasoning Under Uncertainty and Imprecision*, Subseries of Research Notes in Computer Science, vol. 304, Editura Universitaria Craiova, 2010

[2] **Emil Sofron, Nicu Bizon, Silviu Ionita, Radian Raducu** *Sisteme de control fuzzy*, Editura ALL, 1998

[3] **Ion Vaduva, Grigore Albeanu** *Introducere in Modelarea Fuzzy*, Editura Universitatii din Bucuresti, 2004