Inteligenta Artificiala

Universitatea Politehnica Bucuresti
Anul universitar 2010-2011

Adina Magda Florea

http://turing.cs.pub.ro/ia_10 si
curs.cs.pub.ro

Curs nr. 5

Reprezentarea cunostintelor in IA

Modelul logicii simbolice

- Reprezentarea cunostintelor in logica simbolica
- Sistem formal
- Logica propozitiilor
- Logica cu predicate de ordinul I
- Demonstrarea teoremelor prin respingere rezolutiva
- Limbajul Prolog

1. Reprezentarea cunostintelor in LS

- Logica – avantaje
- Puterea de reprezentare a diverselor logici simbolice
- Conceptualizare + exprimarea in limbaj
- Limbaj formal: sintaxa, semantica
- Reguli de inferenta

2. Sistem formal

- Un **sistem formal** este un cuadruplu $S = < A, \mathcal{F}, \mathcal{A}, \Re >$
- O ***regula de inferenta*** $R \in \Re$ de aritate *n* este o corespondenta:

$$R \subseteq \mathcal{F}^n \times \mathcal{F}, \ \overline{y} = \langle y_1, \ldots, y_n \rangle \xrightarrow{R} x, \ \ x, y_i \in \mathcal{F}, \ \ \forall i = \overline{1, n}$$

- Fie multimea de premise $\Gamma = \{y_1, \ldots, y_n\}$ $E_0 = \Gamma \cup \mathcal{A}$

$$E_1 = E_0 \bigcup_{n \geq 1} \{x | \exists \overline{y} \in E_0^n, \ \overline{y} \, \Re \, x\} \qquad E_2 = E_1 \bigcup_{n \geq 1} \{x | \exists \overline{y} \in E_1^n, \ \overline{y} \, \Re \, x\}$$

- Un element x din $E_i \ (i \geq 0)$

este o **<u>consecinta</u>** a multimii de premise Γ

Sistem formal - cont

- Daca $E_0 = \Gamma \cup A$ atunci $x \in E_i$ este **deductibil** din Γ
  $$\Gamma \vdash_S x$$
- Secventa r.i. - **deductie**

- Daca $E_0 = A$ $(\Gamma = \phi)$ atunci elementele lui E_i se numesc **teoreme**

- Fie $x \in E_i$ o teorema; se obtine prin aplicarea succesiva a r.i. asupra formulelor din E_i
- Secventa de reguli - **demonstratie** . $\vdash_{\mathcal{R}} x$

3. Logica propozitiilor

3.1 Sintaxa

- Alfabet
- O **formula bine formata** in calculul propozitional se defineste recursiv astfel:

(1) Un atom este o formula bine formata

(2) Daca P este formula bine formata, atunci $\sim P$ este formula bine formata.

(3) Daca P si Q sint formule bine formate atunci $P \wedge Q$, $P \vee Q$, $P \rightarrow Q$ si $P \leftrightarrow Q$ sint formule bine formate.

(4) Multimea formulelor bine formate este generata prin aplicarea repetata a regulilor (1)..(3) de un numar finit de ori.

3.2 Semantica

- Interpretare
- *Functia de evaluare a unei formule*
- Proprietatile fbf
  - Valida/tautologie
  - Realizabila
  - Inconsistenta
  - Formule echivalente

Semantica - cont

- O formula F este o *consecinta logica a unei formule* P daca F are valoarea adevarat in toate interpretarile in care P are valoarea adevarat.
- O formula F este *consecinta logica a unei multimi de formule* $P_1, \ldots P_n$ daca formula F este adevarata in toate interpretarile in care $P_1, \ldots P_n$ sunt adevarate.
- Consecinta logica se noteaza $P_1, \ldots P_n \Rightarrow F$.
- **Teorema**. Formula F este consecinta logica a unei multimi de formule $P_1, \ldots P_n$ daca formula $P_1, \ldots P_n \rightarrow F$ este valida.
- **Teorema**. Formula F este consecinta logica a unei multimi de formule $P_1, \ldots P_n$ daca formula $P_1 \wedge \ldots \wedge P_n \wedge \sim F$ este inconsistenta.

Legi de echivalenta

Idempotenta	$P \vee P \equiv P$	$P \wedge P \equiv P$	
Asociativitate	$(P \vee Q) \vee R \equiv P \vee (Q \vee R)$	$(P \wedge Q) \wedge R \equiv P \wedge (Q \wedge R)$	
Comutativitate	$P \vee Q \equiv Q \vee P$	$P \wedge Q \equiv Q \wedge P$	$P \leftrightarrow Q \equiv Q \leftrightarrow P$
Distributivitate	$P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$	$P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$	
De Morgan	$\sim (P \vee Q) \equiv \sim P \wedge \sim Q$	$\sim (P \wedge Q) \equiv \sim P \vee \sim Q$	
Eliminarea implicatiei	$P \rightarrow Q \equiv \sim P \vee Q$		
Eliminarea implicatiei duble	$P \leftrightarrow Q \equiv (P \rightarrow Q) \wedge (Q \rightarrow P)$		

3.3 Obtinerea de noi cunostinte

- Conceptualizare
- Reprezentare in limbaj
- Teoria modelului

$$KB \models_S x$$

- Teoria demonstratiei

$$KB \vdash_{\mathcal{R}} x$$

- Logici monotone
- Logici nemonotone

3.4 Reguli de inferenta

- *Modus Ponens* $\dfrac{\begin{matrix} P \\ P \rightarrow Q \end{matrix}}{Q}$
- *Substitutia*
- *Regula inlantuirii* $\dfrac{\begin{matrix} P \rightarrow Q \\ Q \rightarrow R \end{matrix}}{P \rightarrow R}$
- *Regula introducerii conjunctiei* $\dfrac{\begin{matrix} P \\ Q \end{matrix}}{P \wedge Q}$
- *Regula transpozitiei* $\dfrac{P \rightarrow Q}{\sim Q \rightarrow \sim P}$

Exemplu

- *Mihai are bani*
- *Masina este alba*
- *Masina este frumoasa*
- *Daca masina este alba sau masina este frumoasa si Mihai are bani atunci Mihai pleaca in vacanta*
- B
- A
- F
- $(A \vee F) \wedge B \rightarrow C$

4. Logica cu predicate de ordinul I

4.1 Sintaxa

Fie *D* un domeniu de valori. Un *termen* se defineste astfel:

- (1) O constanta este un termen cu valoare fixa apartinand domeniului *D*.
- (2) O variabila este un termen ce poate primi valori diferite din domeniul *D*.
- (3) Daca f este o functie de n argumente si $t_1,..t_n$ sint termeni, atunci $f(t_1,..t_n)$ este termen.
- (4) Toti termenii sunt generati prin aplicarea regulilor (1)…(3).

Sintaxa LP - cont

- Predicat de aritate n
- Atom sau formula atomica.
- Literal

O *formula bine formata* in logica cu predicate de ordinul I se defineste astfel:

(1) Un atom este o formula bine formata

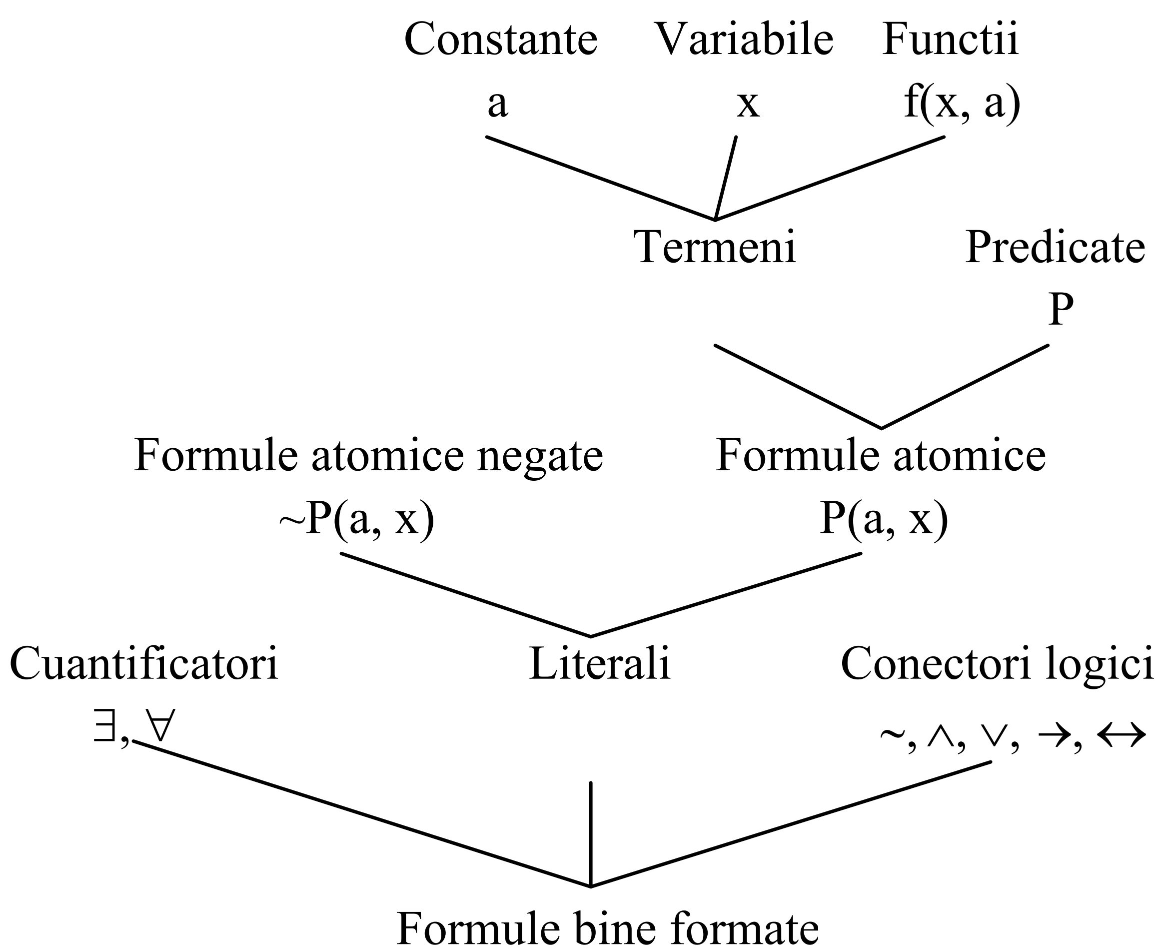
(2) Daca P[x] este fbf, atunci ~P[x] este fbf.

(3) Daca P[x] si Q [x] sunt fbf atunci $P[x] \wedge Q[x]$, $P[x] \vee Q[x]$, $P \rightarrow Q$ si $P \leftrightarrow Q$ sunt fbf.

(4) Daca P[x] este fbf atunci $\forall x\ P[x]$, $\exists x\ P[x]$ sunt fbf.

(5) Multimea formulelor bine formate este generata prin aplicarea repetata a regulilor (1)..(4) de un numar finit de ori.

Sintaxa pe scurt



FNC, FND

- O formula bine formata este in *forma normala conjunctiva*, pe scurt FNC, daca formula are forma

$$F_1 \wedge \ldots \wedge F_n,$$

unde este F_i , i=1,n sunt formule formate dintr-o disjunctie de literali ($L_{i1} \vee \ldots \vee L_{im}$).

- O formula bine formata este in *forma normala disjunctiva*, pe scurt FND, daca formula are forma ,

$$F_1 \vee \ldots \vee F_n,$$

unde F_i , i=1,n sunt formule formate dintr-o conjunctie de literali ($L_{i1} \wedge \ldots \wedge L_{im}$)

4.2 Semantica LP

- *Interpretarea* unei formule F in logica cu predicate de ordinul I consta in fixarea unui domeniu de valori nevid D si a unei asignari de valori pentru fiecare constanta, functie si predicat ce apar in F astfel:
- (1) Fiecarei constante i se asociaza un element din D.
- (2) Fiecarei functii f, de aritate n, i se asociaza o corespondenta $D^n \to D$, unde

$$D^n = \{(x_1,\ldots,x_n) | x_1 \in D,\ldots,x_n \in D\}$$

- (3) Fiecarui predicat de aritate n, i se asociaza o corespondenta $P: D^n \to \{\mathbf{a},\mathbf{f}\}$

Interpretare I

$$(\forall x)(((A(a,x) \vee B(f(x))) \wedge C(x)) \rightarrow D(x))$$

D={1,2}

a
2

f(1)	f(2)
2	1

A(2,1)	A(2,2)	B(1)	B(2)	C(1)	C(2)	D(1)	D(2)
a	**f**	**a**	**f**	**a**	**f**	**f**	**a**

X=1 $((\mathbf{a} \vee \mathbf{f}) \wedge \mathbf{a}) \rightarrow \mathbf{f}$

X=2 $((\mathbf{f} \vee \mathbf{a}) \wedge \mathbf{f}) \rightarrow \mathbf{a}$

4.3 Proprietatile fbf in LP

- Valida/tautologie
- Realizabila
- Inconsistenta
- Echivalente

- F - *consecinta logica a unei formule* P
- F - *consecinta logica a unei multimi de formule* $P_1,\ldots P_n$
- **Teorema**. Formula F este consecinta logica a unei multimi de formule $P_1,\ldots P_n$ daca formula $P_1,\ldots P_n \rightarrow F$ este valida.
- **Teorema**. Formula F este consecinta logica a unei multimi de formule $P_1,\ldots P_n$ daca formula $P_1 \wedge \ldots \wedge P_n \wedge \sim F$ este inconsistenta.

Echivalenta cuantificatorilor

$(Qx)F[x] \vee G \equiv (Qx)(F[x] \vee G)$	$(Qx)F[x] \wedge G \equiv (Qx)(F[x] \wedge G)$
$\sim ((\forall x)F[x]) \equiv (\exists x)(\sim F[x])$	$\sim ((\exists x)F[x]) \equiv (\forall x)(\sim F[x])$
$(\forall x)F[x] \wedge (\forall x)H[x] \equiv (\forall x)(F[x] \wedge H[x])$	$(\exists x)F[x] \vee (\exists x)H[x] \equiv (\exists x)(F[x] \vee H[x])$
$(Q_1x)F[x] \wedge (Q_2x)H[x] \equiv (Q_1x)(Q_2z)(F[x] \wedge H[z])$	$(Q_1x)F[x] \vee (Q_2x)H[x] \equiv (Q_1x)(Q_2z)(F[x] \vee H[z])$

Exemple

- Toate merele sunt rosii
- Toate obiectele sunt mere rosii
- Exista un mar rosu
- Toate pachetele din camera 27 sunt mai mici decat orice pachet din camera 28

Toate ciupercile purpurii sunt otravitoare

- $\forall x$ (Purpuriu(x) $\wedge$ Ciuperca(x)) $\Rightarrow$ Otravitor(x)
- $\forall x$ Purpuriu(x) $\Rightarrow$ (Ciuperca(x) $\Rightarrow$ Otravitor(x))
- $\forall x$ Ciuperca (x) $\Rightarrow$ (Purpuriu (x) $\Rightarrow$ Otravitor(x))

$(\forall x)(\exists y)$ iubeste(x,y)
$(\exists y)(\forall x)$ iubeste(x,y)

4.4. Reguli de inferenta in LP

- Modus Ponens (MP)

$$\frac{\begin{array}{c} P(a) \\ (\forall x)(P(x) \rightarrow Q(x)) \end{array}}{Q(a)}$$

- Substitutia
- Regula inlantuirii
- Transpozitia
- Eliminarea conjunctiei (ElimC)
- Introducerea conjunctiei (IntrC)
- Instantierea universala (InstU)
- Instantierea existentiala (InstE)
- Rezolutia

Exemplu

- Caii sunt mai rapizi decat cainii si exista un ogar care este mai rapid decat orice iepure. Se stie ca Harry este un cal si ca Ralph este un iepure. Sa se demonstreze faptul ca Harry este mai rapid decat Ralph.
- Cal(x) Ogar(y)
- Caine(y) Iepure(z)
- MaiRapid(y,z)

$$\forall x\ \forall y\ Cal(x) \wedge Caine(y) \Rightarrow MaiRapid(x,y)$$

$$\exists y\ Ogar(y) \wedge (\forall z\ Iepure(z) \Rightarrow MaiRapid(y,z))$$

$$Cal(Harry)$$

$$Iepure(Ralph)$$

$$\forall y\ Ogar(y) \Rightarrow Caine(y)$$

$$\forall x\ \forall y\ \forall z\ MaiRapid(x,y) \wedge MaiRapid(y,z) \Rightarrow MaiRapid(x,z)$$

Exemplu de demonstrare

- **Teorema: MaiRapid(Harry, Ralph) ?**
- **Demonstrare folosind reguli de inferenta**

1. $\forall x\ \forall y\ Cal(x) \wedge Caine(y) \Rightarrow MaiRapid(x,y)$
2. $\exists y\ Ogar(y) \wedge (\forall z\ Iepure(z) \Rightarrow MaiRapid(y,z))$
3. $\forall y\ Ogar(y) \Rightarrow Caine(y)$
4. $\forall x \forall y \forall z\ MaiRapid(x,y) \wedge MaiRapid(y,z) \Rightarrow MaiRapid(x,z)$
5. Cal(Harry)
6. Iepure(Ralph)

---

7. $Ogar(Greg) \wedge (\forall z\ Iepure(z) \Rightarrow MaiRapid(Greg,z))$ — 2, InstE
8. Ogar(Greg) — 7, ElimC
9. $\forall z\ Iepure(z) \Rightarrow MaiRapid(Greg,z))$ — 7, ElimC

Exemplu de demonstrare - cont

10.	Iepure(Ralph) $\Rightarrow$ MaiRapid(Greg,Ralph)	9, InstU
11.	MaiRapid(Greg,Ralph)	6,10, MP
12.	Ogar(Greg) $\Rightarrow$ Caine(Greg)	3, InstU
13.	Caine(Greg)	12, 8, MP
14.	Cal(Harry) $\wedge$ Caine(Greg) $\Rightarrow$ MaiRapid(Harry, Greg)	1, InstU
15.	Cal(Harry) $\wedge$ Caine(Greg)	5, 13, IntrC
16.	MaiRapid(Harry, Greg)	14, 15, MP
17.	MaiRapid(Harry, Greg) $\wedge$ MaiRapid(Greg, Ralph) $\Rightarrow$ MaiRapid(Harry,Ralph)	4, InstU
18.	MaiRapid(Harry, Greg) $\wedge$ MaiRapid(Greg, Ralph)	16, 11, IntrC
19.	MaiRapid(Harry,Ralph)	17, 18, MP

5. Demonstrarea teoremelor prin respingere rezolutiva

5.1 Forma standard

- Clauza
- Clauza Horn
- Clauza Horn definita
- Clauza vida
- Transformare in forma clauzala
- Forme de notare
    - Forma clauzala $\sim A(x) \vee B(x)$
    - Forma clauzala multime $\{\sim A(x), B(x)\}$
    - Forma consecinta (Gentzer) $A(x) \Rightarrow B(x)$
    - Forma Prolog B(x) :- A(x)

5.2 Unificarea expresiilor

- Substitutie α
- Unificator
- Cel mai general unificator β
- Expresie
- Algoritm de unificare

Algoritm Unifica(E1,E2): Unificarea expresiilor

1. **daca** E1 si E2 sunt constante
   atunci
   1.1. **daca** E1=E2
        atunci intoarce { }
   1.2. **intoarce** INSUCCES
2. **daca** E1 este variabila **atunci**
   2.1 **daca** E1 apare in E2 **atunci intoarce** INSUCCES
        altfel intoarce {E1/E2}
3. **daca** E2 este variabila **atunci**
   3.1 **daca** E2 apare in E1 **atunci intoarce** INSUCCES
        altfel intoarce {E2/E1}
4. **daca** E1= { } sau E2 = { } **atunci intoarce** INSUCCES

5. **daca** E1=simb(t_{11},…,t_{1n}) **si** E2=simb(t_{21},…t_{2n})
atunci
5.1 X $\leftarrow t_{11}$, Y $\leftarrow t_{21}$
5.2 $Rest_1 \leftarrow t_{12},\ldots,t_{1n}$, $Rest_2 \leftarrow t_{22},\ldots t_{2n}$
5.3 $\alpha 1 \leftarrow$ Unifica(X,Y)
5.4 **daca** α1= INSUCCES
atunci intoarce INSUCCES
5.5 $G_1 \leftarrow$ aplica(α1,$Rest_1$), $G_2 \leftarrow$ aplica(α1,$Rest_2$)
5.6 $\alpha 2 \leftarrow$ Unifica(G_1, $G2_2$)
5.7 **daca** α2= INSUCCES
atunci intoarce INSUCCES
altfel intoarce (α1, α2)
6. **intoarce** INSUCCES
sfarsit.

5.3 Rezolutia in logica propozitiilor

- Rezolvent
- Clauze care rezolva
- Principiul demonstratiei
- Arbore de demonstrare

Algoritm: Respingerea prin rezolutie in logica propozitionala.

1. Converteste setul de axiome A in forma clauzala si obtine multimea de clauze S
2. Neaga teorema, transforma teorema negata in forma clauzala si adauga rezultatul la S

   3. **repeta**

   3.1. Selecteaza o pereche de clauze C_1 si C_2 din S

   3.2. Determina $R = Res(C_1,C_2)$

   3.3. **daca** $R \neq \square$

   atunci adauga R la S

   pana $R = \square$ **sau** nu mai exista nici o pereche de clauze care rezolva

4. **daca** s-a obtinut clauza vida

   atunci teorema este adevarata (este demonstrata)

5. **altfel** teorema este falsa

sfarsit.

- Comentarii
  - strategie
  - decidabilitate

Exemplu

- *Mihai are bani*
- *Masina este alba*
- *Masina este frumoasa*
- *Daca masina este alba sau masina este frumoasa si Mihai are bani atunci Mihai pleaca in vacanta*
- B
- A
- F
- $(A \vee F) \wedge B \rightarrow C$

5.4 Rezolutia in logica predicatelor

- Rezolvent binar
- Clauze care rezolva
- Factor al unei clauze
- Rezolvent

Algoritm: Respingerea prin rezolutie in logica predicatelor.

1. Converteste setul de axiome A in forma clauzala si obtine multimea de clauze S
2. Neaga teorema, transforma teorema negata in forma clauzala si adauga rezultatul la S
3. **repeta**
   3.1. Selecteaza o pereche de clauze C_1 si C_2 din S
   3.2. Determina $R = \{Res(C_1,C_2)\}$
   3.3. **daca** $\square \notin R$
   atunci reuneste R cu S

   pana s-a obtinut $\square$ **sau** nu mai exista nici o pereche de clauze care rezolva **sau** un efort a fost epuizat
4. **daca** s-a obtinut clauza vida
   atunci teorema este adevarata (este demonstrata)
5. **altfel daca** nu mai exista nici o pereche de clauze care rezolva
   atunci teorema este falsa
   altfel nu se stie

sfarsit.

- Comentarii
  - strategie
  - decidabilitate
  - completitudine

Exemplu

- Horses are faster than dogs and there is a greyhound that is faster than every rabbit. We know that Harry is a horse and that Ralph is a rabbit. Derive that Harry is faster than Ralph.

- Horse(x) Greyhound(y)
- Dog(y) Rabbit(z)
- Faster(y,z))

$$\forall x\ \forall y\ \mathbf{Horse(x)} \wedge \mathbf{Dog(y)} \Rightarrow \mathbf{Faster(x,y)}$$

$$\exists y\ \mathbf{Greyhound(y)} \wedge (\forall z\ \mathbf{Rabbit(z)} \Rightarrow \mathbf{Faster(y,z)})$$

Horse(Harry)

Rabbit(Ralph)

$$\forall y\ \mathbf{Greyhound(y)} \Rightarrow \mathbf{Dog(y)}$$

$$\forall x\ \forall y\ \forall z\ \mathbf{Faster(x,y)} \wedge \mathbf{Faster(y,z)} \Rightarrow \mathbf{Faster(x,z)}$$

A1. $\forall x\ \forall y\ Horse(x) \wedge Dog(y) \Rightarrow Faster(x,y)$

A2. $\exists y\ Greyhound(y) \wedge (\forall z\ Rabbit(z) \Rightarrow Faster(y,z))$

A3. Horse(Harry)

A4. Rabbit(Ralph)

A5. $\forall y\ Greyhound(y) \Rightarrow Dog(y)$

A6. $\forall x\ \forall y\ \forall z\ Faster(x,y) \wedge Faster(y,z) \Rightarrow Faster(x,z)$

T Faster(Harry,Ralph)

C1. $\sim Horse(x) \vee \sim Dog(y) \vee Faster(x,y)$

C2. Greyhound(Greg)

C2’ $\sim Rabbit(z) \vee Faster(Greg,z)$

C3. Horse(Harry)

C4. Rabbit(Ralph)

C5. $\sim Greyhound(y) \vee Dog(y)$

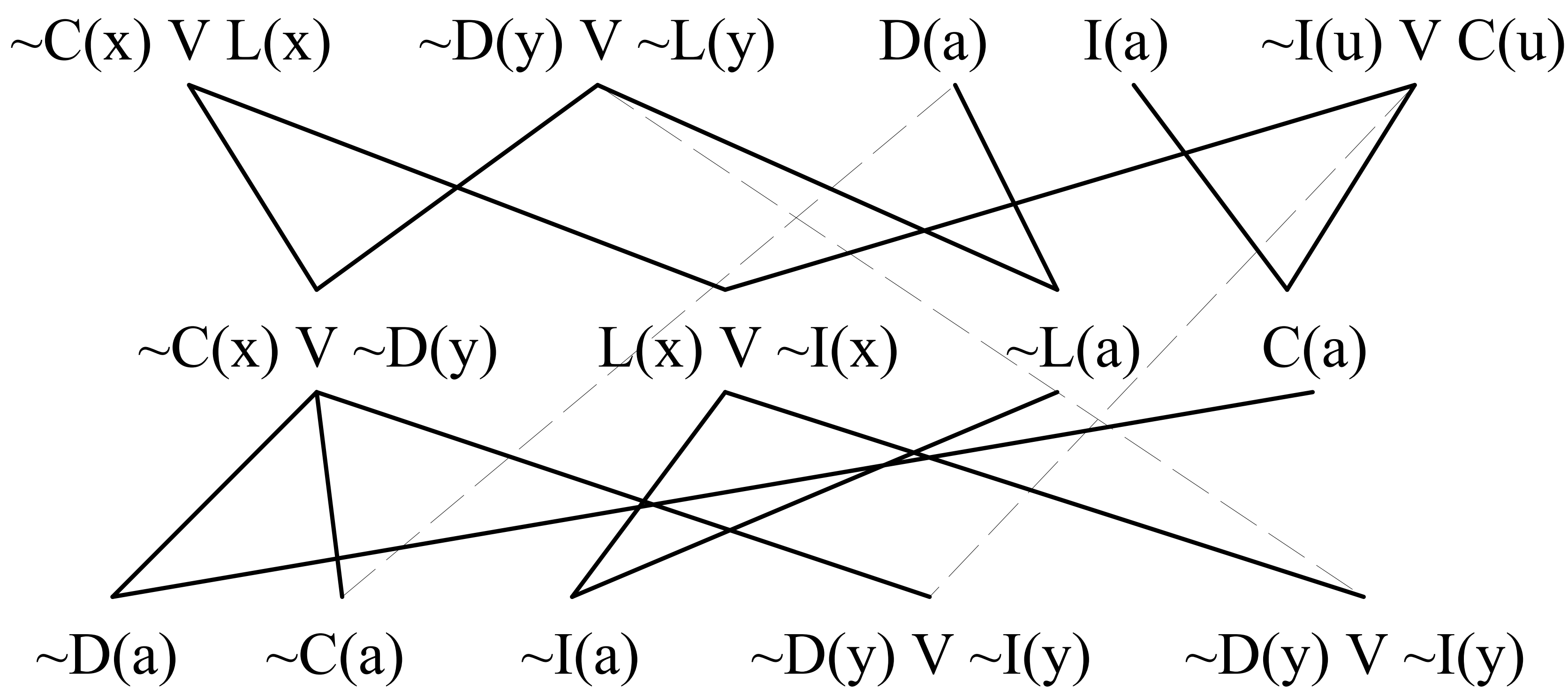
C6. $\sim Faster(x,y) \vee \sim Faster(y,z) \vee Faster(x,z)$

C7. ~Faster(Harry,Ralph)

5.5 Strategii rezolutive

- *Strategia dezvoltarii pe latime*
- *Strategia multimii suport*
- *Strategia rezolutiei liniare*
- *Strategia rezolutiei liniare de intrare*
- *Strategia rezolutiei unitare*
- *Strategia eliminarii*

Strategia dezvoltarii pe latime



Strategia dezvoltarii pe latime

- S_0, L_0
- $L_{k+1} \leftarrow \{Res(C_i, C_j) | C_i \in L_k, C_j \in S_k\}$
- $S_{k+1} \leftarrow S_k \cup L_{k+1}$

  $k = 0,1,2,\ldots$

Strategia multimii suport

- S, T $\qquad S_0 = S \cup T,\ S \cap T = \phi$
- $S \leftarrow S \cup \{Res(C_i, C_j) | C_i \in S, C_j \in T\}$

Strategia rezolutiei liniare

- $S, \qquad C_0 \in S$
- $C_1 \leftarrow \{Res(C_0, C_i) \mid C_{0,} C_i \in S\}$
- $C_{k+1} \leftarrow \{Res(C_k, C_i) \mid C_i \in \{C_{k-1}, C_{k-2, ..}\} \cup S\}$

  $k=1, 2, 3, \ldots$

Strategia rezolutiei liniare de intrare

- $S, \qquad C_0 \in S$
- $C_1 \leftarrow \{Res(C_0, C_i) \mid C_{0,} C_i \in S\}$
- $C_{k+1} \leftarrow \{Res(C_k, C_i) \mid C_i \in S\}$

  $k=1, 2, 3, \ldots$

Strategia rezolutiei unitare

Clauze subsumate

- O clauza C subsumeaza o clauza D daca si numai daca exista o substitutie α astfel incat $C\alpha \subseteq D$. D se numeste clauza subsumata.
- $C=P(x)$ $\quad\quad$ $D=P(a) \vee Q(a)$

Strategia eliminarii

Verificare C subsumeaza D

$D = L_1 \vee \ldots \vee L_m$, $\alpha = \{x_1/a_1, \ldots x_n/a_n\}$

$D\alpha = L_1\alpha \vee \ldots \vee L_m\alpha$, $\sim D\alpha = \sim L_1\alpha \wedge \ldots \wedge \sim L_m\alpha$

1. $W \leftarrow \{\sim L_1\alpha, \ldots, \sim L_m\alpha\}$
2. $k \leftarrow 0$, $U^0 \leftarrow \{C\}$
3. **daca** $\square \in U^k$ **atunci intoarce** SUCCES
4. $U^{k+1} \leftarrow \{rez(C_1,C_2) \mid C_1 \in U^k, C_2 \in W\}$
5. **daca** $U^{k+1} = \phi$ **atunci intoarce** INSUCCES
6. $k \leftarrow k+1$
7. **repeta** de la 3

sfarsit

5.6 Obtinerea raspunsurilor

$$(\forall x)(EsteLa(mihai, x) \rightarrow EsteLa(grivei, x))$$

$$EsteLa(mihai, scoala)$$

$$(\exists y) EsteLa(grivei, y)$$

C1 $\sim EsteLa(mihai, x) \vee EsteLa(grivei, x)$

C2 $EsteLa(mihai, scoala)$

C3 $\sim EsteLa(grivei, y)$

(C3) ~EsteLa(grivei,y) ~EsteLa(mihai,x) V EsteLa(grivei,x) (C1)

{x/y}

(C4) ~EsteLa(mihai,x) EsteLa(mihai,scoala) (C2)

{scoala/x}

$\square$

(a)

(C3) ~EsteLa(grivei,y) V **EsteLa(grivei,y)** ~EsteLa(mihai,x) V EsteLa(grivei,x) (C1)

{x/y}

(C4) ~EsteLa(mihai,x) V **EsteLa(grivei,x)** EsteLa(mihai,scoala) (C2)

{scoala/x}

EsteLa(grivei,scoala)

(b)

6. Prolog

- R. Kowalski, A. Colmerauer - începutul anilor '70

- Corespondenta Logica cu Predicate si limbajul Prolog – ***clauze Horn definite***

- Structura logica vs structura de control si executie

6.1 Structura logica a limbajului Prolog

- Un scop **S** este **adevarat** intr-un program Prolog, adica *poate fi satisfacut* sau *derivă logic din program*, daca si numai daca:

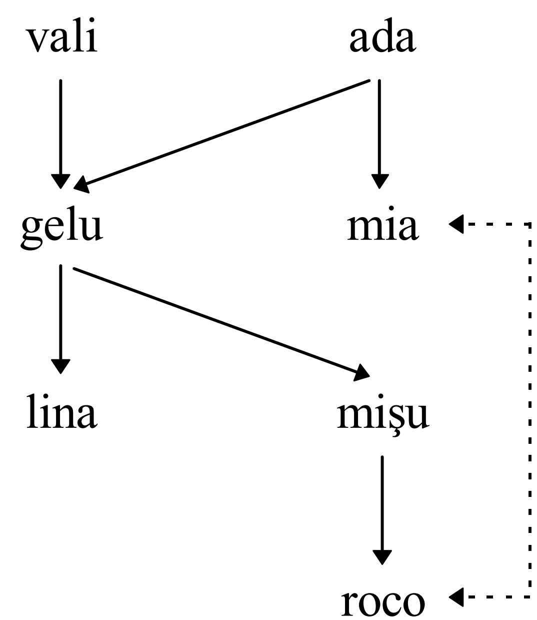
1. exista o clauza **C** a programului;
2. exista o instanta **I** a clauzei **C** astfel incat:
   - antetul lui **I** sa fie identic cu cel al lui **S**;
   - toate scopurile din corpul lui **I** sunt adevarate, deci pot fi satisfacute

6.2 Structura de control si executie a limbajului Prolog

- Rezolutie liniara de intrare
- Ordinea faptelor si clauzelor
- Ordinea scopurilor in corpul clauzelor
- Backtracking

6.3 Prolog – exemplu

```
% parinte(IndividX, IndividY)
% stramos(IndividX, IndividZ)
parinte(vali, gelu).
parinte(ada, gelu).
parinte(ada, mia).
parinte(gelu, lina).
parinte(gelu, misu).
parinte(misu, roco).
```



str1(X, Z) :- parinte(X, Z).
str1(X, Z) :- parinte(X, Y), str1(Y, Z).

Prolog – exemplu

% Se schimba ordinea regulilor:

str2(X, Z) :- parinte(X, Y), str2(Y, Z).

str2(X, Z) :- parinte(X, Z).

% Se schimba ordinea scopurilor in prima varianta:

str3(X, Z) :- parinte(X, Z).

str3(X, Z) :- str3(X, Y), parinte(Y, Z).

% Se schimba atat ordinea regulilor, cat si ordinea scopurilor:

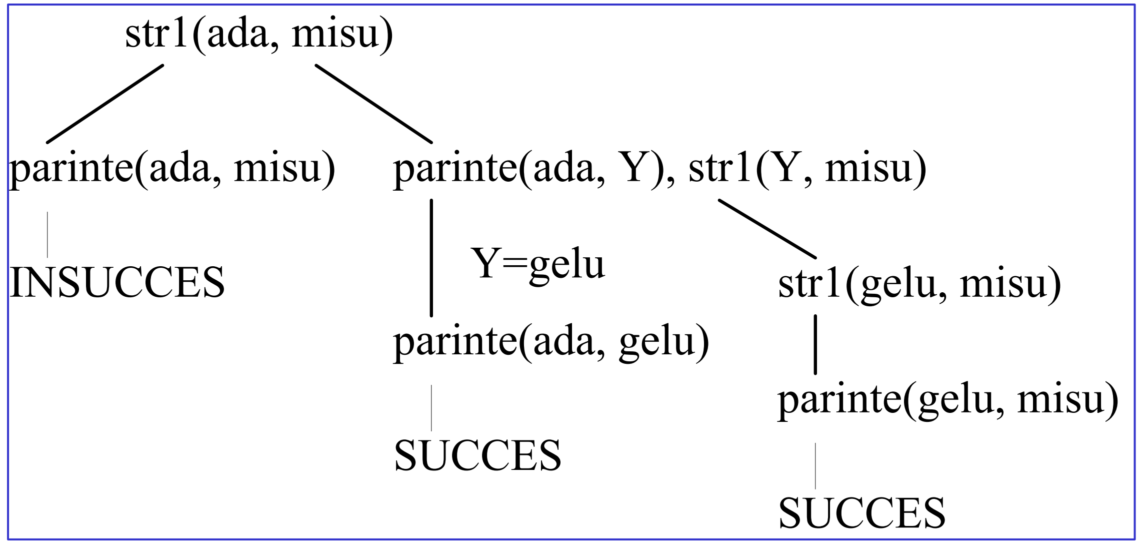
str4(X, Z) **:-** str4(X, Y), parinte(Y, Z).

str4(X, Z) **:-** parinte(X,Z).

Prolog - exemplu

?- str1(ada, misu).

yes

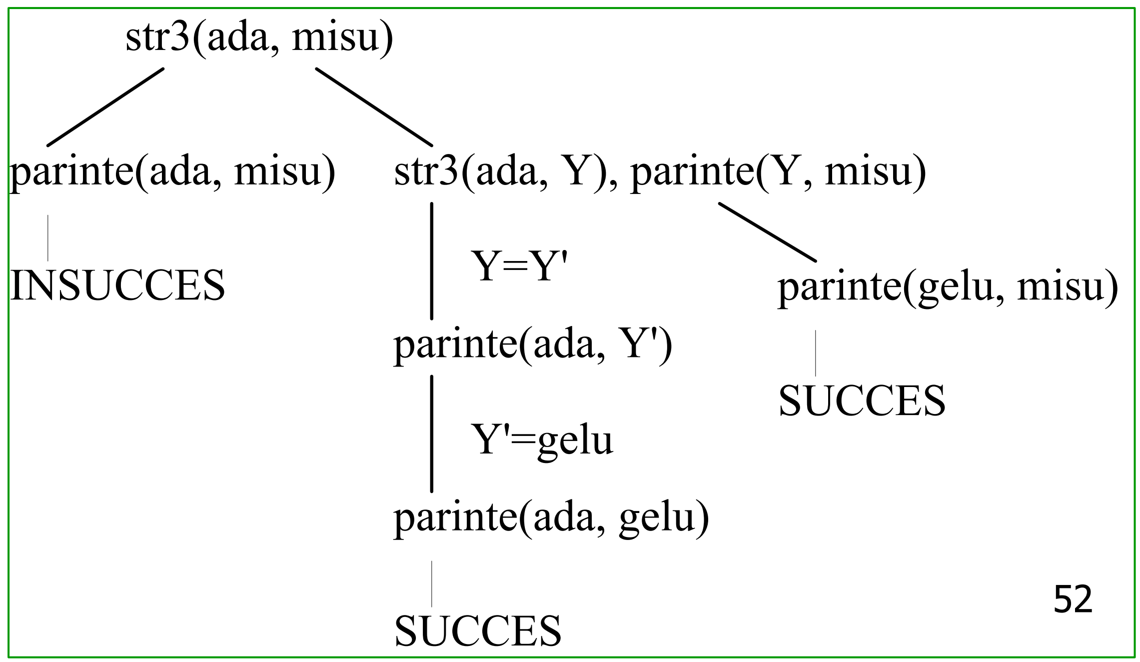


?- str2(ada, misu).

yes

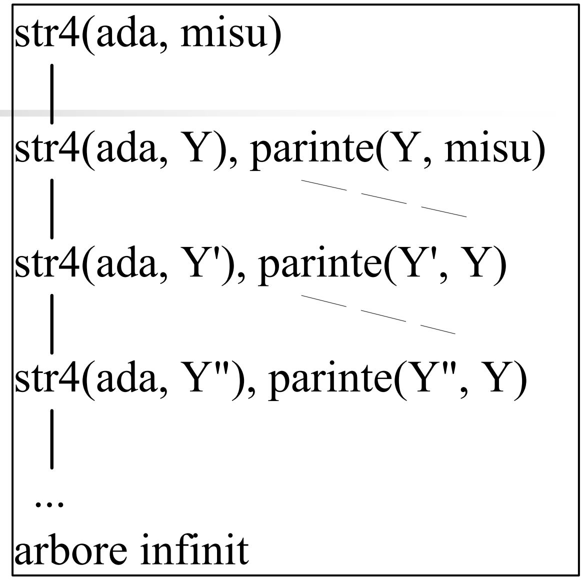
?- str3(ada, misu).

yes



Prolog – exemplu

?- str4(ada, misu).



?- str3(mia, roco).