Curs 11

2020-2021

Cuprins

Logica propozițională (recap.)

Limbajul și formulele PL

- Limbajul PL
  - variabile propoziționale: $Var = \{p, q, v, \ldots\}$
  - conectori logici: $\neg$ (unar), $\rightarrow$, $\wedge$, $\vee$, $\leftrightarrow$ (binari)

- Formulele PL

$$
\begin{array}{lcl}
var & ::= & p \mid q \mid v \mid \ldots \\
form & ::= & var \mid (\neg form) \mid form \wedge form \mid form \vee form \\
 & & \mid form \rightarrow form \mid form \leftrightarrow form
\end{array}
$$

- Conectorii sunt împărțiți în conectori de bază și conectori derivați (în funcție de formalism).

- Legături între conectori:

$$
\begin{array}{rcl}
\varphi \vee \psi & := & \neg\varphi \rightarrow \psi \\
\varphi \wedge \psi & := & \neg(\varphi \rightarrow \neg\psi) \\
\varphi \leftrightarrow \psi & := & (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)
\end{array}
$$

Sintaxa și semantica

Un sistem logic are două componente:

- Sintaxa
  - noțiuni sintactice: demonstrație, teoremă
  - notăm prin $\vdash \varphi$ faptul că φ este teoremă
  - notăm prin $\Gamma \vdash \varphi$ faptul că formula φ este demonstrabilă din mulțimea de formule Γ
- Semantica
  - noțiuni semantice: adevăr, model, tautologie (formulă universal adevărată)
  - notăm prin $\models \varphi$ faptul că φ este tautologie
  - notăm prin $\Gamma \models \varphi$ faptul că formula φ este adevărată atunci când toate formulele din mulțimea Γ sunt adevărate

Completitudine: Γ-teoremele și Γ-tautologiile coincid

$$\Gamma \vdash \varphi \text{ dacă și numai dacă } \Gamma \models \varphi$$

Exemplu

Formalizați următorul raționament:

`If winter is coming and Ned is not alive then Robb is lord of Winterfell. Winter is coming. Rob is not lord of Winterfell. Then Ned is alive.`

O posibilă formalizare este următoarea:

p = `winter is coming`
q = `Ned is alive`
r = `Robb is lord of Winterfel`

$\{(p \land \neg q) \to r, p, \neg r\} \vDash q$

Logica de ordinul I (recap.)

Logica de ordinul I - sintaxa

Limbaj de ordinul I $\mathcal{L}$

- unic determinat de $\tau = (\mathbf{R}, \mathbf{F}, \mathbf{C}, ari)$

Termenii lui $\mathcal{L}$, notați $Trm_{\mathcal{L}}$, sunt definiți inductiv astfel:

- orice variabilă este un termen;
- orice simbol de constantă este un termen;
- dacă $f \in \mathbf{F}$, $ar(f) = n$ și $t_1, \ldots, t_n$ sunt termeni, atunci $f(t_1, \ldots, t_n)$ este termen.

Formulele atomice ale lui $\mathcal{L}$ sunt definite astfel:

- dacă $R \in \mathbf{R}$, $ar(R) = n$ și $t_1, \ldots, t_n$ sunt termeni, atunci $R(t_1, \ldots, t_n)$ este formulă atomică.

Formulele lui $\mathcal{L}$ sunt definite astfel:

- orice formulă atomică este o formulă
- dacă φ este o formulă, atunci $\neg\varphi$ este o formulă
- dacă φ și ψ sunt formule, atunci $\varphi \vee \psi$, $\varphi \wedge \psi$, $\varphi \rightarrow \psi$ sunt formule
- dacă φ este o formulă și x este o variabilă, atunci $\forall x\, \varphi$, $\exists x\, \varphi$ sunt formule

Logica de ordinul I - semantică

O structură este de forma $\mathcal{A} = (A, \mathbf{F}^{\mathcal{A}}, \mathbf{R}^{\mathcal{A}}, \mathbf{C}^{\mathcal{A}})$, unde

- A este o mulțime nevidă
- $\mathbf{F}^{\mathcal{A}} = \{f^{\mathcal{A}} \mid f \in \mathbf{F}\}$ este o mulțime de operații pe A; dacă f are aritatea n, atunci $f^{\mathcal{A}} : A^n \to A$.
- $\mathbf{R}^{\mathcal{A}} = \{R^{\mathcal{A}} \mid R \in \mathbf{R}\}$ este o mulțime de relații pe A; dacă R are aritatea n, atunci $R^{\mathcal{A}} \subseteq A^n$.
- $\mathbf{C}^{\mathcal{A}} = \{c^{\mathcal{A}} \in A \mid c \in \mathbf{C}\}$.

O interpretare a variabilelor lui $\mathcal{L}$ în $\mathcal{A}$ ($\mathcal{A}$-interpretare) este o funcție $I : V \to A$.

Inductiv, definim interpretarea termenului t în $\mathcal{A}$ sub I notat $t_I^{\mathcal{A}}$.

Inductiv, definim când o formulă este adevărată în $\mathcal{A}$ în interpretarea I notat $\mathcal{A}, I \vDash \varphi$. În acest caz spunem că $(\mathcal{A}, I)$ este model pentru φ.

O formulă φ este adevărată într-o structură $\mathcal{A}$, notat $\mathcal{A} \vDash \varphi$, dacă este adevărată în $\mathcal{A}$ sub orice interpretare. Spunem că $\mathcal{A}$ este model al lui φ.

O formulă φ este adevărată în logica de ordinul I, notat $\vDash \varphi$, dacă este adevărată în orice structură. O formulă φ este validă dacă $\vDash \varphi$.

O formulă φ este satisfiabilă dacă există o structură $\mathcal{A}$ și o $\mathcal{A}$-interpretare I astfel încât $\mathcal{A}, I \vDash \varphi$.

Algoritmul de unificare

Unificare

- O subtituție σ este o funcție (parțială) de la variabile la termeni,

$$\sigma : V \to Trm_{\mathcal{L}}$$

- Doi termeni t_1 și t_2 se unifică dacă există o substituție θ astfel încât

$$\theta(t_1) = \theta(t_2).$$

- În acest caz, θ se numește unificatorul termenilor t_1 și t_2.

- Un unificator ν pentru t_1 și t_2 este un cel mai general unificator (cgu,mgu) dacă pentru orice alt unificator ν' pentru t_1 și t_2, există o substituție μ astfel încât

$$\nu' = \nu; \mu.$$

Algoritmul de unificare

- Pentru o mulțime finită de termeni $\{t_1, \ldots, t_n\}$, $n \geq 2$, algoritmul de unificare stabilește dacă există un cgu.

- Algoritmul lucrează cu două liste:
  - Lista soluție: S
  - Lista de rezolvat: R

- Inițial:
  - Lista soluție: $S = \emptyset$
  - Lista de rezolvat: $R = \{t_1 \doteq t_2, \ldots, t_{n-1} \doteq t_n\}$

- $\doteq$ este un simbol nou care ne ajută sa formăm perechi de termeni (ecuații).

Algoritmul de unificare

Algoritmul constă în aplicarea regulilor de mai jos:

- SCOATE
  - orice ecuație de forma $t \doteq t$ din R este eliminată.
- DESCOMPUNE
  - orice ecuație de forma $f(t_1, \ldots, t_n) \doteq f(t'_1, \ldots, t'_n)$ din R este înlocuită cu ecuațiile $t_1 \doteq t'_1, \ldots, t_n \doteq t'_n$.
- REZOLVĂ
  - orice ecuație de forma $x \doteq t$ sau $t \doteq x$ din R, unde variabila x nu apare în termenul t, este mutată sub forma $x \doteq t$ în S.
    În toate celelalte ecuații (din R și S), x este înlocuit cu t.

Algoritmul de unificare

Algoritmul se termină normal dacă $R = \emptyset$. În acest caz, S dă cgu.

Algoritmul este oprit cu concluzia inexistenței unui cgu dacă:

1. În R există o ecuație de forma
$$f(t_1, \ldots, t_n) \doteq g(t'_1, \ldots, t'_k) \text{ cu } f \neq g.$$
2. În R există o ecuație de forma $x \doteq t$ sau $t \doteq x$ și variabila x apare în termenul t.

Algoritmul de unificare - schemă

	Lista soluție S	Lista de rezolvat R
Inițial	$\emptyset$	$t_1 \doteq t'_1, \ldots, t_n \doteq t'_n$
SCOATE	S	R', $t \doteq t$
	S	R'
DESCOMPUNE	S	R', $f(t_1, \ldots, t_n) \doteq f(t'_1, \ldots, t'_n)$
	S	R', $t_1 \doteq t'_1, \ldots t_n \doteq t'_n$
REZOLVĂ	S	R', $x \doteq t$ sau $t \doteq x$, x nu apare în t
	$x \doteq t$, $S[x/t]$	$R'[x/t]$
Final	S	$\emptyset$

$S[x/t]$: în toate ecuațiile din S, x este înlocuit cu t

Exemplu

Exemplu

- Ecuațiile $\{g(y) \doteq x,\ f(x, h(x), y) \doteq f(g(z), w, z)\}$ au gcu?

S	R	
$\emptyset$	$g(y) \doteq x,\ f(x, h(x), y) \doteq f(g(z), w, z)$	REZOLVĂ
$x \doteq g(y)$	$f(g(y), h(g(y)), y) \doteq f(g(z), w, z)$	DESCOMPUNE
$x \doteq g(y)$	$g(y) \doteq g(z),\ h(g(y)) \doteq w,\ y \doteq z$	REZOLVĂ
$w \doteq h(g(y))$, $x \doteq g(y)$	$g(y) \doteq g(z),\ y \doteq z$	REZOLVĂ
$y \doteq z, x \doteq g(z)$, $w \doteq h(g(z))$	$g(z) \doteq g(z)$	SCOATE
$y \doteq z, x \doteq g(z)$, $w \doteq h(g(z))$	$\emptyset$	

- $\nu = \{y/z,\ x/g(z),\ w/h(g(z))\}$ este cgu.

Exemplu

Exemplu

- Ecuațiile $\{g(y) \doteq x,\ f(x, h(y), y) \doteq f(g(z), b, z)\}$ au gcu?

S	R	
$\emptyset$	$g(y) \doteq x,\ f(x, h(y), y) \doteq f(g(z), b, z)$	REZOLVĂ
$x \doteq g(y)$	$f(g(y), h(y), y) \doteq f(g(z), b, z)$	DESCOMPUNE
$x \doteq g(y)$	$g(y) \doteq g(z),\ h(y) \doteq b,\ y \doteq z$	- EȘEC -

- h și b sunt simboluri de operații diferite!
- Nu există unificator pentru ecuațiile din U.

Exemplu

Exemplu

- Ecuațiile $\{g(y) \doteq x,\ f(x, h(x), y) \doteq f(y, w, z)\}$ au gcu?

S	R	
$\emptyset$	$g(y) \doteq x,\ f(x, h(x), y) \doteq f(y, w, z)$	REZOLVĂ
$x \doteq g(y)$	$f(g(y), h(g(y)), y) \doteq f(y, w, z)$	DESCOMPUNE
$x \doteq g(y)$	$g(y) \doteq y,\ h(g(y)) \doteq w,\ y \doteq z$	- EȘEC -

- În ecuația $g(y) \doteq y$, variabila y apare în termenul $g(y)$.
- Nu există unificator pentru ecuațiile din U.

Validitate și satisfiabilitate

Propoziție

Dacă φ este o formulă atunci

$$\varphi \text{ este validă} \quad \text{dacă și numai dacă} \quad \neg\varphi \text{ nu este satisfiabilă.}$$

Formă clauzală. Rezoluție

Literali. FNC

- În logica propozițională un literal este o variabilă sau negația unei variabile.

$$literal := \; p \mid \neg p \qquad \text{unde } p \text{ este variabilă propozițională}$$

- În logica de ordinul I un literal este o formulă atomică sau negația unei formule atomice.

$$literal := \; P(t_1, \ldots, t_n) \mid \neg P(t_1, \ldots, t_n)$$

unde $P \in \mathbf{R}, ari(P) = n,$ și $t_1, \ldots, t_n$ sunt termeni.

- Pentru un literal L vom nota cu L^c literalul complement.

O formulă este în formă normală conjunctivă (FNC) dacă este o conjuncție de disjuncții de literali.

Forma clauzală în logica propozițională

- Pentru orice formulă α există o FNC α^{fc} astfel încât $\alpha \mathrel{\text{⫤}\!\text{⊨}} \alpha^{fc}$.
- Pentru o formulă din logica propozițională determinăm FNC corespunzătoare prin următoarele transformări:

1. înlocuirea implicațiilor și echivalențelor

$$\begin{aligned} \varphi \to \psi &\quad \mathrel{\text{⫤}\!\text{⊨}} \quad \neg\varphi \vee \psi \\ \varphi \leftrightarrow \psi &\quad \mathrel{\text{⫤}\!\text{⊨}} \quad (\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi) \end{aligned}$$

2. regulile De Morgan

$$\begin{aligned} \neg(\varphi \vee \psi) &\quad \mathrel{\text{⫤}\!\text{⊨}} \quad \neg\varphi \wedge \neg\psi \\ \neg(\varphi \wedge \psi) &\quad \mathrel{\text{⫤}\!\text{⊨}} \quad \neg\varphi \vee \neg\psi \end{aligned}$$

3. principiului dublei negații

$$\neg\neg\psi \quad \mathrel{\text{⫤}\!\text{⊨}} \quad \psi$$

4. distributivitatea

$$\begin{aligned} \varphi \vee (\psi \wedge \chi) &\quad \mathrel{\text{⫤}\!\text{⊨}} \quad (\varphi \vee \psi) \wedge (\varphi \vee \chi) \\ (\psi \wedge \chi) \vee \varphi &\quad \mathrel{\text{⫤}\!\text{⊨}} \quad (\psi \vee \varphi) \wedge (\chi \vee \varphi) \end{aligned}$$

Forma clauzală în logica de ordinul I

- O formulă este formă normală conjunctivă prenex (FNCP) dacă
  - este în formă prenex $Q_1 x_1 \ldots Q_n x_n \psi$ ($Q_i \in \{\forall, \exists\}$ oricare i)
  - ψ este FNC

- O formulă este formă clauzală dacă este enunț universal și FNCP:

$$\forall x_1 \ldots \forall x_n \psi \text{ unde } \psi \text{ este FNC}$$

- Pentru orice formulă φ din logica de ordinul I există o formă clauzală φ^{fc} astfel încât

$$\varphi \text{ este satisfiabilă dacă și numai dacă } \varphi^{fc} \text{ este satisfiabilă}$$

- Pentru o formulă φ, forma clauzală φ^{fc} se poate calcula astfel:
  1. se determină forma rectificată
  2. se cuantifică universal variabilele libere
  3. se determină forma prenex
  4. se determină forma Skolem

     în acest moment am obținut o formă Skolem $\forall x_1 \ldots \forall x_n \psi$
  5. se determină o FNC ψ' astfel încât $\psi \mathrel{\text{⫤}\!\!\vDash} \psi'$
  6. φ^{fc} este $\forall x_1 \ldots \forall x_n \psi'$

Clauze

- O clauză este o disjuncție de literali.
- Dacă $L_1, \ldots, L_n$ sunt literali atunci clauza $L_1 \vee \ldots \vee L_n$ o vom scrie ca mulțimea $\{L_1, \ldots, L_n\}$

  clauză = mulțime de literali

- Clauza $C = \{L_1, \ldots, L_n\}$ este satisfiabilă dacă $L_1 \vee \ldots \vee L_n$ este satisfiabilă.
- O clauză C este trivială dacă conține un literal și complementul lui.
- Când $n = 0$ obținem clauza vidă, care se notează $\square$
- Prin definiție, clauza $\square$ nu este satisfiabilă.

Forma clauzală

- Observăm că o FNC este o conjuncție de clauze.
- Dacă $C_1, \ldots, C_k$ sunt clauze atunci $C_1 \wedge \ldots \wedge C_k$ o vom scrie ca mulțimea $\{C_1, \ldots, C_k\}$

$$\text{FNC} = \text{mulțime de clauze}$$

- O mulțime de clauze $\mathcal{C} = \{C_1, \ldots, C_k\}$ este satisfiabilă dacă $C_1 \wedge \ldots \wedge C_k$ este satisfiabilă
- Când $k = 0$ obținem mulțimea de clauze vidă, pe care o notăm $\{\}$
- Prin definiție, mulțimea de clauze vidă $\{\}$ este satisfiabilă.

$$\{\} \text{ este satisfiabilă, dar } \{\square\} \text{ nu este satisfiabilă}$$

Forma clauzală

- Dacă φ este o formulă în calculul propozițional, atunci
$$\varphi^{fc} = \bigwedge_{i=1}^{k} \bigvee_{j=1}^{n_i} L_{ij} \text{ unde } L_{ij} \text{ sunt literali}$$

- Dacă φ o formulă în logica de ordinul I, atunci
$$\varphi^{fc} = \forall x_1 \ldots \forall x_n \left(\bigwedge_{i=1}^{k} \bigvee_{j=1}^{n_i} L_{ij} \right) \text{ unde } L_{ij} \text{ sunt literali}$$

φ este satisfiabilă dacă și numai dacă

φ^{fc} este satisfiabilă dacă și numai dacă

$\{\{L_{11}, \ldots, L_{1n_1}\}, \ldots, \{L_{k1}, \ldots, L_{kn_k}\}\}$ este satisfiabilă

Rezoluția în logica propozițională

Regula rezoluției

$$Rez \quad \frac{C_1 \cup \{p\}, C_2 \cup \{\neg p\}}{C_1 \cup C_2}$$

unde C_1, C_2 clauze, iar p este variabila propozițională astfel încât $\{p, \neg p\} \cap C_1 = \emptyset$ și $\{p, \neg p\} \cap C_2 = \emptyset$.

Fie $\mathcal{C}$ o mulțime de clauze. O derivare prin rezoluție din $\mathcal{C}$ este o secvență finită de clauze astfel încât fiecare clauză este din $\mathcal{C}$ sau rezultă din clauzele anterioare prin rezoluție (este rezolvent).

Derivare prin rezoluție

Fie $\mathcal{C}$ o mulțime de clauze. O derivare prin rezoluție din $\mathcal{C}$ este o secvență finită de clauze astfel încât fiecare clauză este din $\mathcal{C}$ sau rezultă din clauzele anterioare prin rezoluție (este rezolvent).

Exemplu

Fie $\mathcal{C} = \{\{\neg q, \neg p\}, \{q\}, \{p\}\}$ o mulțime de clauze. O derivare prin rezoluție pentru $\square$ din $\mathcal{C}$ este

$$
\begin{array}{lll}
C_1 = & \{\neg q, \neg p\} & \\
C_2 = & \{q\} & \\
C_3 = & \{\neg p\} & (Rez, C_1, C_2) \\
C_4 = & \{p\} & \\
C_5 = & \square & (Rez, C_3, C_4)
\end{array}
$$

Teorema de completitudine

$\models \varphi$ dacă și numai dacă există o derivare prin rezoluție a lui $\square$ din $(\neg\varphi)^{fc}$.

Procedura Davis-Putnam DPP (informal)

Intrare: o mulțime $\mathcal{C}$ de clauze

Se repetă următorii pași:

- se elimină clauzele triviale
- se alege o variabilă p
- se adaugă la mulțimea de clauze toți rezolvenții obținuti prin aplicarea Rez pe variabila p
- se șterg toate clauzele care conțin p sau $\neg p$

Ieșire: dacă la un pas s-a obținut $\square$, mulțimea $\mathcal{C}$ nu este satisfiabilă; altfel $\mathcal{C}$ este satisfiabilă.

Logica Horn

Clauze definite. Programe logice. Clauze Horn

- clauză:

$$\{\neg Q_1, \ldots, \neg Q_n, P_1, \ldots, P_k\} \quad \text{sau} \quad Q_1 \wedge \ldots \wedge Q_n \to P_1 \vee \ldots \vee P_k$$

  unde $n, k \geq 0$ și $Q_1, \ldots, Q_n, P_1, \ldots, P_k$ sunt formule atomice.

- clauză program definită: $k = 1$
  - cazul $n > 0$: $Q_1 \wedge \ldots \wedge Q_n \to P$
  - cazul $n = 0$: $\top \to P$ (clauză unitate, fapt)

  Program logic definit = mulțime finită de clauze definite

- scop definit (țintă, întrebare): k=0
  - $Q_1 \wedge \ldots \wedge Q_n \to \bot$

- clauza vidă $\square$: $n = k = 0$

Clauza Horn = clauză program definită sau clauză scop ($k \leq 1$)

Programare logica

- Logica clauzelor definite/Logica Horn: un fragment al logicii de ordinul I în care singurele formule admise sunt clauze Horn
  - formule atomice: $P(t_1, \ldots, t_n)$
  - $Q_1 \wedge \ldots \wedge Q_n \to P$
    unde toate Q_i, P sunt formule atomice, $\top$ sau $\bot$

- Problema programării logice: reprezentăm cunoștințele ca o mulțime de clauze definite KB și suntem interesați să aflăm răspunsul la o întrebare de forma $Q_1 \wedge \ldots \wedge Q_n$, unde toate Q_i sunt formule atomice

$$KB \models Q_1 \wedge \ldots \wedge Q_n$$

  - Variabilele din KB sunt cuantificate universal.
  - Variabilele din $Q_1, \ldots, Q_n$ sunt cuantificate existențial.

Limbajul PROLOG are la bază logica clauzelor Horn.

Sistem de deducție *backchain*

Sistem de deducție pentru clauze Horn

Pentru un program logic definit KB avem

- Axiome: orice clauză din KB
- Regula de deducție: regula *backchain*

$$\frac{\theta(Q_1) \quad \theta(Q_2) \quad \ldots \quad \theta(Q_n) \quad (Q_1 \land Q_2 \land \ldots \land Q_n \to P)}{\theta(Q)}$$

unde $Q_1 \land Q_2 \land \ldots \land Q_n \to P \in KB$, iar θ este cgu pentru Q și P.

Sistem de deducție

$$\frac{\theta(Q_1) \quad \theta(Q_2) \quad \ldots \quad \theta(Q_n) \quad (Q_1 \wedge Q_2 \wedge \ldots \wedge Q_n \rightarrow P)}{\theta(Q)}$$

unde $Q_1 \wedge Q_2 \wedge \ldots \wedge Q_n \rightarrow P \in KB$, iar θ este cgu pentru Q și P.

Exemplu

KB conține următoarele clauze definite:

$father(jon, ken)$. $father(ken, liz)$.
$father(X, Y) \rightarrow ancestor(X, Y)$
$daugther(X, Y) \rightarrow ancestor(Y, X)$
$ancestor(X, Y) \wedge ancestor(Y, Z) \rightarrow ancestor(X, Z)$

atunci

$$\frac{\dfrac{father(ken, liz)}{father(ken, Z)} \quad (father(Y, X) \rightarrow ancestor(Y, X))}{ancestor(ken, Z)}$$

Rezoluția SLD

Rezoluția SLD

Fie T o mulțime de clauze definite.

$$\text{SLD}\quad \boxed{\frac{\neg Q_1 \vee \cdots \vee \neg Q_i \vee \cdots \vee \neg Q_n}{\theta(\neg Q_1 \vee \cdots \vee \neg P_1 \vee \cdots \vee \neg P_m \vee \cdots \vee \neg Q_n)}}$$

unde

- $Q \vee \neg P_1 \vee \cdots \vee \neg P_m$ este o clauză definită din KB (în care toate variabilele au fost redenumite) și
- variabilele din $Q \vee \neg P_1 \vee \cdots \vee \neg P_m$ și Q_i se redenumesc
- θ este c.g.u pentru Q_i și Q

Rezoluția SLD

Fie KB o mulțime de clauze definite și $Q_1 \land \ldots \land Q_m$ o întrebare, unde Q_i sunt formule atomice.

- O derivare din KB prin rezoluție SLD este o secvență

$$G_0 := \neg Q_1 \lor \ldots \lor \neg Q_m, \quad G_1, \quad \ldots, \quad G_k, \ldots$$

  în care G_{i+1} se obține din G_i prin regula SLD.
- Dacă există un k cu $G_k = \square$ (clauza vidă), atunci derivarea se numește SLD-respingere.

Rezoluția SLD

Exercițiu

Găsiți o SLD-respingere pentru următorul program Prolog și ținta:

```
1.  p(X) :- q(X,f(Y)), r(a).       ?- p(X), q(Y,Z).
2.  p(X) :- r(X).
3.  q(X,Y) :- p(Y).
4.  r(X) :- q(X,Y).
5.  r(f(b)).
```

Soluție

$G_0 = \neg p(X) \vee \neg q(Y,Z)$	
$G_1 = \neg r(X_1) \vee \neg q(Y,Z)$	(2 cu $\theta(X) = X_1$)
$G_2 = \neg q(Y,Z)$	(5 cu $\theta(X_1) = f(b)$)
$G_3 = \neg p(Z_1)$	(3 cu $\theta(X) = Y_1$ și $\theta(Y) = Z_1$)
$G_4 = \neg r(X)$	(2 cu $\theta(Z_1) = X$)
$G_5 = \square$	(5 cu $\theta(X) = f(b)$)

Rezoluția SLD - arbori de căutare

Arbori SLD

- Presupunem că avem o mulțime de clauze definite KB și o țintă $G_0 = \neg Q_1 \vee \ldots \vee \neg Q_m$
- Construim un arbore de căutare (arbore SLD) astfel:
  - Fiecare nod al arborelui este o țintă (posibil vidă)
  - Rădăcina este G_0
  - Dacă arborele are un nod G_i, iar G_{i+1} se obține din G_i folosind regula SLD folosind o clauză $C_i \in KB$, atunci nodul G_i are copilul G_{i+1}. Muchia dintre G_i și G_{i+1} este etichetată cu C_i.
- Dacă un arbore SLD cu rădăcina G_0 are o frunză $\square$ (clauza vidă), atunci există o SLD-respingere a lui G_0 din KB.

Rezoluția SLD - arbore de căutare complet

Exercițiu

Desenați arborele SLD pentru programul Prolog de mai jos și ținta
`?- p(X,X).`

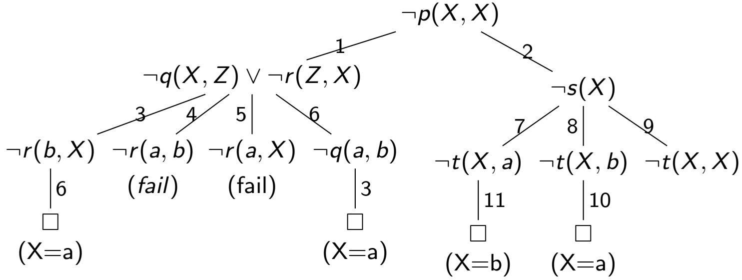
```
1.  p(X,Y) :- q(X,Z), r(Z,Y).
2.  p(X,X) :- s(X).
3.  q(X,b).
4.  q(b,a).
5.  q(X,a) :- r(a,X).
6.  r(b,a).
```

```
7.  s(X) :- t(X,a).
8.  s(X) :- t(X,b).
9.  s(X) :- t(X,X).
10.  t(a,b).
11.  t(b,a).
```

Rezoluția SLD - arbore SLD complet

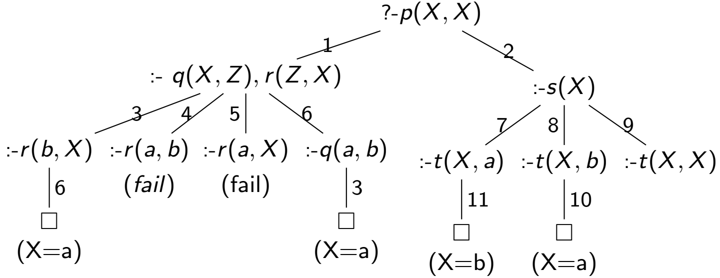
```
1.  p(X,Y) :- q(X,Z), r(Z,Y).
2.  p(X,X) :- s(X).
3.  q(X,b).
4.  q(b,a).
5.  q(X,a) :- r(a,X).
6.  r(b,a).
7.  s(X) :- t(X,a).
8.  s(X) :- t(X,b).
9.  s(X) :- t(X,X).
10. t(a,b).
11. t(b,a).
```

$p(X, Y) \vee \neg q(X, Z) \vee \neg r(Z, Y)$
$p(X, X) \vee \neg s(X)$
$q(X, b)$
$q(b, a)$
$q(X, a) \vee \neg r(a, X)$
$r(b, a)$
$s(X) \vee \neg t(X, a)$
$s(X) \vee \neg t(X, b)$
$s(X) \vee \neg t(X, X)$
$t(a, b)$
$t(b, a)$



Rezoluția SLD - arbori de execuție

```
1.  p(X,Y) :- q(X,Z), r(Z,Y).
2.  p(X,X) :- s(X).
3.  q(X,b).
4.  q(b,a).
5.  q(X,a) :- r(a,X).
6.  r(b,a).
7.  s(X) :- t(X,a).
8.  s(X) :- t(X,b).
9.  s(X) :- t(X,X).
10. t(a,b).
11. t(b,a).
```



Rezoluția SLD

Exercițiu

Fie KB următoarea bază de cunoștințe definită în Prolog:

1. `r(a, a)`
2. `q(X, a)`
3. `p(X, Y) :- q(X, Z), r(Z, Y)`

(a)Desenați arborele SLD și arborele de execuție pentru întrebarea
`?- p(X, Z)`

(b) Exprimați KB ca o mulțime de formule în logica de ordinul I
demonstrați folosind rezoluția că din KB se deduce `p(X,Z)`, adică
KB $\vdash \exists x \exists z\, p(x, z)$.

(a) Soluție:

1. `r(a, a).`
2. `q(X, a).`
3. `p(X, Y) :- q(X, Z), r(Z, Y)`

Arborele SLD:

$$\begin{array}{c} \neg p(X,Z) \\ \downarrow 3 \\ \neg q(X1,Z1) \vee \neg r(Z1,Y1) \\ \downarrow 2 \\ \neg r(a,Y1) \\ \downarrow 1 \\ \square \end{array}$$

Arborele de execuție:

```
?- p(X,Z)
   ↓ 3
:- q(X1,Z1), r(Z1,Y1)
   ↓ 2
:- r(a,Y1)
   ↓ 1
   □
```

(cont.)
Fie KB următoarea bază de cunoștințe definită în Prolog:

1. `r(a, a).` 2. `q(X, a).` 3. `p(X, Y) :- q(X, Z), r(Z, Y)`

(b) Soluție:

KB$=\{r(a,a), \forall x\, q(x,a), \forall x \forall y \forall z\, (\neg q(x,y) \vee \neg r(z,y) \vee p(x,y))\}$

KB $\vdash \exists x \exists z\ p(x,z)$ dacă și numai dacă există o derivare prin rezoluție pentru $\square$ din forma clauzală a mulțimii $KB \cup \{\neg(\exists x \exists z\ p(x,z))\}$.

Forma clauzală a mulțimii $KB \cup \{\neg(\exists x \exists z\ p(x,z))\}$ este $\mathcal{C} = \{\{r(a,a)\}, \{q(x,a)\}, \{\neg q(x,y), \neg r(z,y), p(x,y)\}, \{\neg p(x,z)\}\}$. Se face derivarea direct sau se construiește arborele SLD.

Pe săptămâna viitoare!