Seminar - Programare Logica si Functionala

Seria 36

2025

1 Algoritmul de unificare

Teorie

O *substituţie* este o funcţie parţială de la variabile la termeni, adică $\sigma : V \to Trm_{\mathcal{L}}$. Un *unificator* pentru doi termeni t_1 şi t_2 este o substituţie θ astfel încât $\theta(t_1) = \theta(t_2)$. Un unificator ν pentru t_1 şi t_2 este un *cel mai general unificator* dacă pentru orice alt unificator ν' pentru t_1 şi t_2, există o substituţie μ astfel încât $\nu' = \nu; \mu$.

Algoritmul de unificare:

	Lista soluţie S	Lista de rezolvat R
Iniţial	$\emptyset$	$t_1 \doteq t'_1, \ldots, t_n \doteq t'_n$
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
	$x \doteq t$, $S[x \leftarrow t]$	$R'[x \leftarrow t]$
Final	S	$\emptyset$

Algoritmul *se termină normal* dacă $R = \emptyset$ (în acest caz, în S are un unificator pentru termenii din lista iniţială R).

Algoritmul este oprit cu concluzia *inexistenţei unui unificator* dacă:

(i) În R există o ecuaţie de forma $f(t_1, \ldots, t_n) \doteq g(t'_1, \ldots, t'_k)$ cu $f \neq g$. Simbolurile de constantă se consideră simboluri de funcţie de aritate 0.

(ii) În R există o ecuaţie de forma $x \doteq t$ sau $t \doteq x$ şi variabila x apare în termenul t.

(Exercitiul 1) Considerăm
- x, y, z, u, v variabile,
- a, b, c simboluri de constantă,
- $h, g, (\_)^{-1}$ simboluri de funcţie de aritate 1,
- $f, *, +$ simboluri de funcţie de aritate 2,
- p simbol de funcţie de aritate 3.

Aplicaţi algoritmul de unificare de mai sus pentru a găsi un unificator pentru termenii:

1) $p(a, x, h(g(y)))$ şi $p(z, h(z), h(u))$

2) $f(h(a), g(x))$ şi $f(y, y)$

3) $p(a, x, g(x))$ şi $p(a, y, y)$

4) $p(x, y, z)$ şi $p(u, f(v, v), u)$

5) $f(x, f(x, x))$ şi $f(g(y), f(z, g(a)))$

6) $x + (y * y)$ şi $(y * y) + z$

7) $(x * y) * z$ şi $u * u^{-1}$

8) $x * y$ şi $u * u^{-1}$

9) $x * y$ şi $x * (y * (u * v)^{-1})$

10) $x * y$ şi $y * (u * v)^{-1}$

11) $f(g(x), x)$ şi $f(y, y)$

12) $p(x, z, z)$ şi $p(y, y, b)$

13) $p(a, u, h(x))$ şi $p(y, f(y, z), z)$

14) $f(x, f(b, x))$ şi $f(f(y, a), f(b, f(z, z)))$

15) $p(x, b, x)$ şi $p(y, y, c)$

16) $f(x, y)$, $f(h(x), x)$ şi $f(x, b)$

17) $f(x, f(x, g(y)))$, $f(u, z)$ şi $f(g(y), y)$

18) $f(f(x, y), x)$, $f(g(y), z)$ şi $f(u, h(z))$

19) $f(f(x, y), x)$, $f(v, u)$ şi $f(u, h(z))$

Demonstraţie:

1.

S	R	
$\emptyset$	$p(a, x, h(g(y))) = p(z, h(z), h(u))$	DESCOMPUNE
$\emptyset$	$a \doteq z, x \doteq h(z), h(g(y)) \doteq h(u)$	REZOLVĂ
$z \doteq a$	$x \doteq h(a), h(g(y)) \doteq h(u)$	REZOLVĂ
$z \doteq a, x \doteq h(a)$	$h(g(y)) \doteq h(u)$	DESCOMPUNE
$z \doteq a, x \doteq h(a)$	$g(y) \doteq u$	REZOLVĂ
$z \doteq a, x \doteq h(a), u \doteq g(y)$	$\emptyset$	

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

2.

S	R	
$\emptyset$	$f(h(a), g(x)) \doteq f(y, y)$	DESCOMPUNE
$\emptyset$	$y \doteq h(a), y \doteq g(x)$	REZOLVĂ
$y \doteq h(a)$	$g(x) \doteq h(a)$	EŞEC

- Nu există unificator.

3.

S	R	
$\emptyset$	$p(a, x, g(x)) \doteq p(a, y, y)$	DESCOMPUNE
$\emptyset$	$a \doteq a, x \doteq y, y \doteq g(x)$	SCOATE
$\emptyset$	$x \doteq y, y \doteq g(x)$	REZOLVĂ
$x \doteq y$	$y \doteq g(y)$	EŞEC

- Nu există unificator.

4.

S	R	
$\emptyset$	$p(x, y, z) \doteq p(u, f(v, v), u)$	DESCOMPUNE
$\emptyset$	$x \doteq u, y \doteq f(v, v), z \doteq u$	REZOLVĂ
$x \doteq u$	$y \doteq f(v, v), z \doteq u$	REZOLVĂ
$y \doteq f(v, v), x \doteq u$	$z \doteq u$	REZOLVĂ
$z \doteq u, y \doteq f(v, v), x \doteq u$		

- $\nu = \{z/u,\ y/f(v,v),\ x/u\}$ este cgu.

5.

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

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

6.

S	R	
$\emptyset$	$x+(y*y) \doteq (y*y)+z$	DESCOMPUNE
$\emptyset$	$x \doteq y*y, z \doteq y*y$	REZOLVĂ
$x \doteq y*y$	$z \doteq y*y$	REZOLVĂ
$z \doteq y*y, x \doteq y*y$		

- $\nu = \{x/y*y,\ z/y*y)\}$ este cgu.

7.

S	R	
$\emptyset$	$(x*y)*z \doteq u*u^{-1}$	DESCOMPUNE
$\emptyset$	$u \doteq x*y, z \doteq u^{-1}$	REZOLVĂ
$u \doteq x*y$	$z \doteq (x*y)^{-1}$	REZOLVĂ
$z \doteq (x*y)^{-1}, u \doteq x*y$		

- $\nu = \{z/(x*y)^{-1},\ u/x*y,\}$ este cgu.

8.

S	R	
$\emptyset$	$x*y \doteq u*u^{-1}$	DESCOMPUNE
$\emptyset$	$x \doteq u, y = u^{-1}$	REZOLVĂ
$x \doteq u$	$y = u^{-1}$	REZOLVĂ
$y \doteq u^{-1}, x \doteq u$		

- $\nu = \{y/u^{-1},\ x/u\}$ este cgu.

9.

S	R	
$\emptyset$	$x*y \doteq x*(y*(u*v)^{-1})$	DESCOMPUNE
$\emptyset$	$x \doteq x, y \doteq y*(u*v)^{-1}$	SCOATE
$\emptyset$	$y \doteq y*(u*v)^{-1}$	EŞEC

- Nu există unificator.

10.

S	R	
$\emptyset$	$x*y \doteq y*(u*v)^{-1}$	DESCOMPUNE
$\emptyset$	$x \doteq y, y \doteq (u*v)^{-1}$	REZOLVĂ
$x \doteq y$	$y \doteq (u*v)^{-1}$	REZOLVĂ
$y \doteq (u*v)^{-1}, x \doteq (u*v)^{-1}$		REZOLVĂ

- $\nu = \{y/(u*v)^{-1},\ y/(u*v)^{-1}\}$ este cgu.

$\square$

2 Deductie naturala

Sistemul de reguli al deducţiei naturale

$$\frac{\varphi \quad \psi}{\varphi \wedge \psi}\ (\wedge i) \qquad\qquad \frac{\varphi \wedge \psi}{\varphi}\ (\wedge e_1) \qquad \frac{\varphi \wedge \psi}{\psi}\ (\wedge e_2)$$

$$\frac{\boxed{\begin{array}{c}\varphi\\ \vdots\\ \psi\end{array}}}{\varphi \to \psi}\ (\to i) \qquad\qquad \frac{\varphi \quad \varphi \to \psi}{\psi}\ (\to e)$$

$$\frac{\varphi}{\varphi \vee \psi}\ (\vee i_1) \qquad \frac{\psi}{\varphi \vee \psi}\ (\vee i_2) \qquad\qquad \frac{\varphi \vee \psi \quad \boxed{\begin{array}{c}\varphi\\ \vdots\\ \chi\end{array}} \ \boxed{\begin{array}{c}\psi\\ \vdots\\ \chi\end{array}}}{\chi}\ (\vee e)$$

$$\frac{\boxed{\begin{array}{c}\varphi\\ \vdots\\ \bot\end{array}}}{\neg\varphi}\ (\neg i) \qquad\qquad \frac{\varphi \quad \neg\varphi}{\bot}\ (\neg e)$$

$$\frac{\varphi}{\neg\neg\varphi}\ (\neg\neg i) \qquad\qquad \frac{\neg\neg\varphi}{\varphi}\ (\neg\neg e)$$

$$\frac{}{\varphi \vee \neg\varphi}\ \textbf{TND} \qquad\qquad \frac{\bot}{\varphi}\ (\bot e)$$

TND (*tertium non datur*) este regulă derivată.

Atenţie! La acest sistem se adaugă regula de copiere.

Regula de copiere:

- la un pas al unei demonstraţii poate fi copiată orice formulă demonstrată anterior.
- la un pas al unei demonstraţii nu pot fi copiate formule din interiorul cutiilor care sunt închise în acel moment.

Exemplu în curs sau Huth si Ryan, pg. 20, unde cartea Huth si Ryan: http://staff.ustc.edu.cn/~huangwc/book/LogicInCS.pdf

(Exercitiul 2) Demonstraţi că următorii secvenţi sunt valizi:

(1) $(p \wedge q) \wedge r, s \wedge t \vdash q \wedge s$

(2) $p, \neg\neg(q \wedge r) \vdash \neg\neg p \wedge r$

(3) $p \wedge q \to r \vdash p \to (q \to r)$

(4) $p \wedge (q \vee r) \vdash (p \wedge q) \vee (p \wedge r)$

(5) $p \to q, p \to \neg q \vdash \neg p$

Demonstraţie: (2) Huth şi Ryan, pg 8; (3) Huth şi Ryan, pg 15;

(4) Huth şi Ryan Example 1.18, pg 19; (5) Huth şi Ryan, example 1.21, pg 22 □

(Exercitiul 3) Demonstraţi că următoarele reguli pot fi derivate din regulile deducţiei naturale:

$$\frac{\varphi \to \psi \quad \neg\psi}{\neg\varphi}\ \textbf{MT} \qquad\qquad \frac{\boxed{\begin{array}{c}\neg\varphi\\ \vdots\\ \bot\end{array}}}{\varphi}\ \textbf{RAA}$$

MT =*modus tollens* RAA =*reductio ad absurdum*

Demonstraţie: Huth şi Ryan, sectiunea 1.2.2 □

(Exercitiul 4) Fie $n \geq 1$ şi $\varphi_1, \ldots, \varphi_n$, φ formule. Demonstraţi că

dacă $\vdash \varphi_1 \to (\varphi_2 \to (\cdots \to (\varphi_n \to \varphi) \cdots))$ este valid, atunci $\varphi_1, \ldots, \varphi_n \vdash \varphi$ este valid.

Demonstraţie: Huth şi Ryan, pagina 53, Step 3 □

(Exercitiul 5) Ştim că *echivalenţa logică* este definită astfel: $\varphi \leftrightarrow \psi = (\varphi \to \psi) \wedge (\psi \to \varphi)$. Găsiţi reguli de introducere şi eliminare pentru $\leftrightarrow$.

Demonstraţie: Observăm că $\leftrightarrow$ este o combinaţie între $\to$ şi $\wedge$. Regulile pentru $\leftrightarrow$ se obţin combinând regulile pentru $\to$ şi $\wedge$.

Introducerea $(\leftrightarrow i)$: pentru a introduce $\varphi \leftrightarrow \psi$ trebuie să introducem $\varphi \to \psi$ şi $\psi \to \varphi$, apoi să introducem $\wedge$; în consecinţă regula va arăta aşa

$$\frac{\boxed{\begin{array}{c}\varphi\\ \vdots\\ \psi\end{array}} \; \boxed{\begin{array}{c}\psi\\ \vdots\\ \varphi\end{array}}}{\varphi \leftrightarrow \psi}\ (\leftrightarrow i)$$

Eliminarea $(\leftrightarrow i)$: pentru a elimina $\varphi \leftrightarrow \psi$ trebuie să eliminăm $\wedge$ apoi să eliminăm o $\to$; vom avea două variante:

$$\frac{\varphi \leftrightarrow \psi \quad \psi}{\varphi} \ (\leftrightarrow e_1) \qquad \frac{\varphi \leftrightarrow \psi \quad \varphi}{\psi} \ (\leftrightarrow e_2)$$

□

(Exercitiul 6) Formalizaţi şi demonstraţi folosind deducţia naturală faptul că din ipotezele (i1)-(i5) deducem (c):

(i1) *Toţi scriitorii care înţeleg natura umană sunt înţelepţi.*
(i2) *Un scriitor care este poet adevărat poate trezi sentimente puternice.*
(i3) *Eminescu este scriitorul care a scris "Luceafărul".*
(i4) *Un poet care trezeşte sentimente puternice înţelege natura umană.*
(i5) *Numai un poet adevărat putea scrie "Luceafărul".*

(c) Eminescu este înţelept.

Demonstraţie: Formalizarea:

$S \wedge NU \rightarrow I$
$S \wedge P \rightarrow SP$
$E \rightarrow S \wedge L$
$P \wedge SP \rightarrow NU$
$L \rightarrow P$

□

3 Puncte fixe

$\boxed{\textbf{Teorie}}$

O *mulţime parţial ordonată (mpo)* este o pereche $(M, \leq)$ unde $\leq \subseteq M \times M$ este o relaţie de ordine (i.e., reflexivă, antisimetrică, tranzitivă). O mulţime parţial ordonată $(C, \leq)$ este *completă (cpo)* dacă C are prim element $\perp$ ($\perp \leq x$ oricare $x \in C$) şi $\bigvee_n x_n$ există în C pentru orice lanţ $x_1 \leq x_2 \leq x_3 \leq \ldots$.

Fie $(C, \leq_C)$ o mulţime parţial ordonată. Un element $a \in C$ este *punct fix* al unei funcţii $f : C \to C$ dacă $f(a) = a$. Un element $lfp \in C$ este *cel mai mic punct fix* al unei funcţii $f : C \to C$ dacă este punct fix şi pentru orice alt punct fix $a \in C$ al lui f avem $lfp \leq_C a$.

(Exercitiul 7) Care sunt punctele fixe ale următoarelor funcţii? Indicaţi cel mai punct fix.

1) $f_1 : \mathcal{P}(\{1,2,3\}) \to \mathcal{P}(\{1,2,3\})$, $f_1(Y) = Y \cup \{1\}$.

2) $f_2 : \mathcal{P}(\{1,2,3\}) \to \mathcal{P}(\{1,2,3\})$, $f_2(Y) = \begin{cases} \{1\} & \text{dacă } 1 \in Y \\ \emptyset & \text{altfel} \end{cases}$.

3) $f_3 : \mathcal{P}(\{1,2,3\}) \to \mathcal{P}(\{1,2,3\})$, $f_3(Y) = \begin{cases} \emptyset & \text{dacă } 1 \in Y \\ \{1\} & \text{altfel} \end{cases}$.

Demonstraţie:

1) Se observă că punctele fixe ale lui f_1 sunt submulţimile Y ale lui $\{1,2,3\}$ care îl conţin pe 1 (dacă $1 \notin Y$, atunci $f_1(Y) = Y \cup \{1\}$ şi evident $Y \neq Y \cup \{1\}$). Deci punctele fixe ale lui f_1 sunt $\{1\}, \{1,2\}, \{1,3\}, \{1,2,3\}$. Evident, cel mai mic punct fix este $\{1\}$.

2) Se observă că singurele puncte fixe ale lui f_2 sunt $\emptyset$ şi $\{1\}$. Evident $\emptyset$ este cel mai mic punct fix.

3) Se observă că f_3 nu are puncte fixe.

$\square$

$\boxed{\textbf{Teorie}}$

Fie $(A, \leq_A)$ şi $(B, \leq_B)$ mulţimi parţial ordonate. O funcţie $f : A \to B$ este *monotonă (crescătoare)* dacă $a_1 \leq_A a_2$ implică $f(a_1) \leq_B f(a_2)$ oricare $a_1, a_2 \in A$.

O *clauză definită propoziţională* este o formulă care poate avea una din formele:
− q (clauză unitate)
− $p_1 \wedge \ldots \wedge p_k \to q$
unde $q, p_1, \ldots, p_n$ sunt variabile propoziţionale.

Fie S o mulţime de clauze definite propoziţionale. Fie A mulţimea variabilelor propoziţionale $p_1, p_2, \ldots$ care apar în S şi $Baza = \{p_i \mid p_i \in S\}$ mulţimea clauzelor unitate din S. Definim funcţia $f_S : \mathcal{P}(A) \to \mathcal{P}(A)$ prin

$$f_S(Y) = Y \cup Baza \cup \ \{a \in A \mid (s_1 \wedge \ldots \wedge s_n \to a) \text{ este în } S, s_1 \in Y, \ldots, s_n \in Y\}$$

(Exercitiul 8) Arătaţi că funcţia f_S este monotonă.

Demonstraţie: Fie $Y_1, Y_2 \subseteq A$ astfel încât $Y_1 \subseteq Y_2$. Trebuie să arătăm că $f_S(Y_1) \subseteq f_S(Y_2)$. Fie următoarele multimi:

$$\begin{aligned} Z_1 &= \{a \in A \mid (s_1 \wedge \ldots \wedge s_n \to a) \text{ este în } S, s_1 \in Y_1, \ldots, s_n \in Y_1\}, \\ Z_2 &= \{a \in A \mid (s_1 \wedge \ldots \wedge s_n \to a) \text{ este în } S, s_1 \in Y_2, \ldots, s_n \in Y_2\}. \end{aligned}$$

Deci $f_S(Y_1) = Y_1 \cup Baza \cup Z_1$ şi $f_S(Y_2) = Y_2 \cup Baza \cup Z_2$. Cum $Y_1 \subseteq Y_2$, rămâne să arătăm doar că $Z_1 \subseteq Z_2$. Fie $a \in Z_1$. Atunci există $s_1 \wedge \ldots \wedge s_n \to a \in S$ şi $s_1, \ldots, s_n \in Y_1$. Deci $s_1, \ldots, s_n \in Y_2$, de unde rezultă că $a \in Z_2$. $\square$

$\boxed{\textbf{Teorie}}$

Fie $(A, \leq_A)$ şi $(B, \leq_B)$ mulţimi parţial ordonate complete. O funcţie $f : A \to B$ este *continuă* dacă $f(\bigvee_n a_n) = \bigvee_n f(a_n)$ pentru orice lanţ $\{a_n\}_n$ din A. Observăm că orice funcţie continuă este crescătoare.

Pentru orice mulţime de clauze definite propoziţionale S, funcţia f_S este continuă.

Teorema 1 (Knaster-Tarski)**.** *Fie $(C, \leq)$ o mulţime parţial ordonată completă şi $\mathbf{F} : C \to C$ o funcţie continuă. Atunci*

$$a = \bigvee_n \mathbf{F}^n(\perp)$$

este cel mai mic punct fix al funcţiei $\mathbf{F}$.

(Exercitiul 9) Calculaţi cel mai mic punct fix pentru functia f_{S_i}, $i \in \{1, 2, 3\}$, pentru următoarele mulţimi de clauze definite propoziţionale:

1) $S_1 = \{x_1 \wedge x_2 \to x_3,\ x_4 \wedge x_2 \to x_5,\ x_2,\ x_6,\ x_6 \to x_1\}$.

2) $S_2 = \{x_1 \wedge x_2 \to x_3,\ x_4 \to x_1,\ x_5 \to x_2,\ x_2 \to x_5,\ x_4\}$.

3) $S_3 = \{x_1 \to x_2,\ x_1 \wedge x_3 \to x_1,\ x_3\}$.

Demonstraţie:

1) Observăm că $A = \{x_1, x_2, \ldots, x_6\}$ şi $Baza = \{x_2, x_6\}$. Cum f_S este continuă, aplicăm Teorema Knaster-Tarski pentru a calcula cel mai mic punct fix:

   $f_{S_1}(\emptyset) = Baza = \{x_2, x_6\}$

   $f_{S_1}(\{x_2, x_6\}) = \{x_2, x_6, x_1\}$

   $f_{S_1}(\{x_2, x_6, x_1\}) = \{x_2, x_6, x_1, x_3\}$

   $f_{S_1}(\{x_2, x_6, x_1, x_3\}) = \{x_2, x_6, x_1, x_3\}$

   În concluzie, cel mai mic punct fix căutam este $\{x_2, x_6, x_1, x_3\}$.

2) Observăm că $A = \{x_1, x_2, \ldots, x_5\}$ şi $Baza = \{x_4\}$. Cum f_S este continuă, aplicăm Teorema Knaster-Tarski pentru a calcula cel mai mic punct fix:

   $f_{S_2}(\emptyset) = Baza = \{x_4\}$

   $f_{S_2}(\{x_4\}) = \{x_4, x_1\}$

   $f_{S_2}(\{x_4, x_1\}) = \{x_4, x_1\}$

   În concluzie, cel mai mic punct fix căutam este $\{x_4, x_1\}$.

3) Observăm că $A = \{x_1, x_2, x_3\}$ şi $Baza = \{x_3\}$. Cum f_S este continuă, aplicăm Teorema Knaster-Tarski pentru a calcula cel mai mic punct fix:

   $f_{S_3}(\emptyset) = Baza = \{x_3\}$

   $f_{S_3}(\{x_3\}) = \{x_3\}$

   În concluzie, cel mai mic punct fix căutam este $\{x_3\}$.

$\square$

4 Rezolutie SLD

Teorie

- O *clauză definită* este o formulă de forma:
  - $P(t_1, \ldots, t_n)$ (formulă atomică), unde P este un simbol de predicat, iar $t_1, \ldots, t_n$ termeni
  - $P_1 \wedge \ldots \wedge P_n \to Q$, unde toate P_i, Q sunt formule atomice.
- O regulă din Prolog $\mathtt{Q} :- \mathtt{P_1}, \ldots, \mathtt{P_n}$ este o clauză $P_1 \wedge \ldots \wedge P_n \to Q$, iar un fapt din Prolog $\mathtt{P}(\mathtt{t_1}, \ldots, \mathtt{t_n})$ este o formulă atomică $P(t_1, \ldots, t_n)$.
- O clauză definită $P_1 \wedge \ldots \wedge P_n \to Q$ poate fi gândită ca formula $Q \vee \neg P_1 \vee \ldots \vee \neg P_n$.
- Pentru o mulţime de clauze definite T, *regula rezolutiei SLD* este

$$\text{SLD} \quad \boxed{\frac{\neg P_1 \vee \cdots \vee \neg P_i \vee \cdots \vee \neg P_n}{(\neg P_1 \vee \cdots \vee \neg Q_1 \vee \cdots \vee \neg Q_m \vee \cdots \vee \neg P_n)\theta}}$$

  unde $Q \vee \neg Q_1 \vee \cdots \vee \neg Q_m$ este o clauză definită din T (în care toate variabilele au fost redenumite) şi θ este c.g.u pentru P_i şi Q.
- Fie T o mulţime de clauze definite şi $P_1 \wedge \ldots \wedge P_m$ o ţintă, unde P_i sunt formule atomice. O derivare din T prin rezoluţie SLD este o secvenţă $G_0 := \neg P_1 \vee \ldots \vee \neg P_m$, G_1, …, G_k, … în care G_{i+1} se obţine din G_i prin regula SLD. Dacă există un k cu $G_k = \square$ (clauza vidă), atunci derivarea se numeşte *SLD-respingere*.

Teorema 2 (Completitudinea SLD-rezoluţiei)**.** *Sunt echivalente:*

(i) există o SLD-respingere a lui $P_1 \wedge \ldots \wedge P_m$ din T,

(ii) $T \vDash P_1 \wedge \cdots \wedge P_m$.

(Exercitiul 10) Găsiţi o SLD-respingere pentru următoarele programe Prolog şi ţinte:

(a)
```
1. r :- p,q.     5. t.     ?- w.
2. s :- p,q.     6. q.
3. v :- t,u.     7. u.
4. w :- v,s.     8. p.
```

(b)
```
1. q(X,Y) :- q(Y,X), q(Y,f(f(Y))).      ?- q(f(Z),a).
2. q(a,f(f(X))).
```

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

Demonstraţie:

(a)

$G_0 = \neg w$	
$G_1 = \neg v \vee \neg s$	(4)
$G_2 = \neg t \vee \neg u \vee \neg s$	(3)
$G_3 = \neg u \vee \neg s$	(5)
$G_4 = \neg s$	(7)
$G_5 = \neg p \vee \neg q$	(2)
$G_6 = \neg q$	(8)
$G_7 = \square$	(6)

(b)

$G_0 = \neg q(f(Z), a)$	
$G_1 = \neg q(a, f(Z)) \vee \neg q(a, f(f(a)))$	(1 cu $\theta(X) = f(Z)$ şi $\theta(Y) = a$)
$G_2 = \neg q(a, f(Z))$	(2 cu $\theta(X) = a$)
$G_3 = \square$	(2 cu $\theta(Z) = f(X)$)

(c)

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

$\square$

Teorie

Fie T o mulţime de clauze definite şi o ţintă $G_0 = \neg P_1 \vee \ldots \vee \neg P_m$. Un *arbore SLD* este definit astfel:

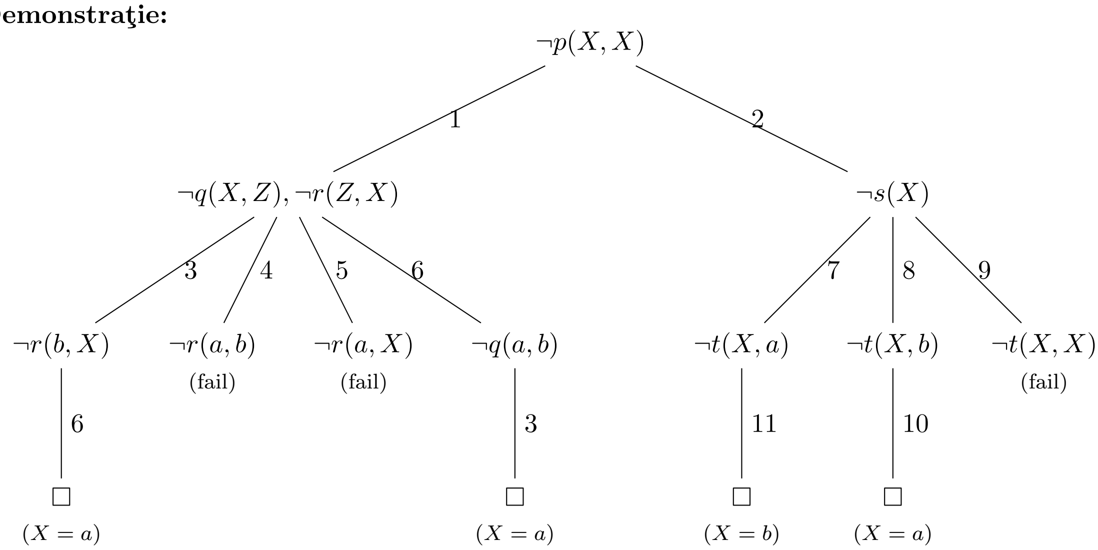
- Fiecare nod al arborelui este o ţintă (posibil vidă)
- Rădăcina este G_0
- Dacă arborele are un nod G_i, iar G_{i+1} se obţine din G_i folosind regula SLD folosind o clauză $C_i \in T$, atunci nodul G_i are copilul G_{i+1}. Muchia dintre G_i şi G_{i+1} este etichetată cu C_i.

Dacă un arbore SLD cu rădăcina G_0 are o frunză $\square$ (clauza vidă), atunci există o SLD-respingere a lui G_0 din T.

(Exercitiul 11) Desenaţi arborele SLD pentru programul Prolog de mai jos şi ţinta `?- p(X,X).`

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

Demonstraţie:



$\square$

5 Lambda-calcul

Teorie

Lambda-calculul (sau λ-calculul) a fost dezvoltat între 1929-1932 de Alonzo Church și a fost propus ca sistem formal pentru logica matematică. În 1935, a demonstrat că orice funcție calculabilă peste numerele naturale, poate fi calculată în λ-calcul. Tot în 1935, independent de Church, Alan Turing a dezvoltat mecanismul numit astăzi mașina Turing, iar în 1936 a argumentat și el că orice funcție calculabilă peste numerele naturale poate fi calculată de o mașină Turing și, în plus, a arătat echivalența celor două modele de calcul (λ-calcul și mașini Turing), ducând la ceea e numim astăzi *Teza Church-Turing.*

Sintaxa λ-calculului:

$$t = x \mid \lambda x.t \mid t\ t$$

λ**-termeni**. Fie $Var = \{x, y, z, ...\}$ o mulțime infinită de variabile. Mulțimea λ-termenilor ΛT este definită inductiv, astfel:

$$\begin{aligned}&[\text{Variabilă}]\ Var \subseteq \Lambda T\\ &[\text{Aplicare}]\ \text{dacă } t_1, t_2 \in \Lambda T \text{ atunci } (t_1\ t_2) \in \Lambda T\\ &[\text{Abstractizare}]\ \text{dacă } x \in Var \text{ și } t \in \Lambda T,\ \text{atunci } (\lambda x.t) \in \Lambda T\end{aligned}$$

Convenții de scriere.

- în scrierea λ-termenilor vom elimina parantezele exterioare și vom scrie, de exemplu, xy în loc de (xy), sau $\lambda x.xy$ în loc de $(\lambda x.(xy))$;
- aplicarea este asociativă la stânga: $t_1t_2t_3$ este $(t_1t_2)t_3$;
- corpul abstractizării este extins la dreapta: $\lambda x.t_1t_2$ este $\lambda x.(t_1t_2)$;
- scriem $\lambda xyz.t$, în loc de $\lambda x.\lambda y.\lambda z.t$.

5.1 Variabile libere și legate

Variabile libere și legate. Pentru un termen $\lambda x.t$ spunem că:

- aparițiile variabilei x în t sunt legate *(bound)*;
- λx este legătura *(binder)*, iar t este domeniul *(scope)* legării;
- o apariție a unei variabile este liberă *(free)* dacă apare într-o poziție în care nu este legată.

Un termen fără variabile libere se numește **închis** *(closed)*.

Mulțimea variabilelor libere FV(t). Pentru un λ-termen t, mulțimea variabilelor libere este definită astfel:

[Variabilă] $FV(x) = \{x\}$

[Aplicare] $FV(t_1t_2) = FV(t_1) \cup FV(t_2)$

[Abstractizare] $FV(\lambda x.t) = FV(t) - \{x\}$

Exercițiul 1. *Calculați mulțimea variabilelor libere pentru următorii λ-termeni:*

a. $\lambda x.xy$

b. $x\lambda x.xy$

c. $x(\lambda xy.xyz)(\lambda v.yv)$

d. $\lambda t.((\lambda xyz.yzx)t)$

5.2 Substituții

Fie t un λ-termen și $x \in Var$. Pentru un λ-termen u vom nota prin $[u/x]t$ rezultatul înlocuirii tuturor aparițiilor libere ale lui x cu u în t.

[Variabilă] $[u/x]x = u$

[Variabilă] $[u/x]y = y$ dacă $x \neq y$

[Aplicare] $[u/x](t_1 t_2) = [u/x]t_1[u/x]t_2$

[Abstractizare] $[u/x]\lambda y.t = \lambda y.[u/x]t$ unde $x \neq y$ și $y \notin FV(u)$

Variabilele legate pot fi redenumite.

Exercițiul 2. *Aplicați substituțiile indicate în următorii λ-termeni:*

a. $[y/x]\lambda z.x$

b. $[y/x]\lambda y.x$

c. $[\lambda z.z/x](\lambda x.yx)$

d. $[\lambda z.z/x](\lambda y.yx)$

5.3 α-conversie (α-echivalență)

α-conversia $=_\alpha$ este relația binară care satisface următoarele proprietăți:

[Reflexivitate] $t =_\alpha t$

[Simetrie] $t_1 =_\alpha t_2$ implică $t_2 =_\alpha t_1$

[Tranzitivitate] $t_1 =_\alpha t_2$ și $t_2 =_\alpha t_3$ implică $t_1 =_\alpha t_3$

[Redenumire] $\lambda x.t =_\alpha \lambda y.[y/x]t$ dacă $y \notin FV(t)$

[Compatibilitate] $t_1 =_\alpha t_2$ implică $tt_1 =_\alpha tt_2$, $t_1 t =_\alpha t_2 t$ și $\lambda x.t_1 =_\alpha \lambda x.t_2$

Compatibilitatea cu substituția.

$$t_1 =_\alpha t_2 \text{ și } u_1 =_\alpha u_2 \text{ implică } [u_1/x]t_1 =_\alpha [u_2/x]t_2$$

Exercițiul 3. *Verificați care dintre α-conversiile următoare sunt adevărate:*

a. $\lambda x.x =_\alpha \lambda y.y$

b. $\lambda x.y =_\alpha \lambda y.x$

c. $\lambda x.xy =_\alpha \lambda x.xz$

d. $\lambda x.xy =_\alpha \lambda y.yy$

e. $x\lambda x.xy =_\alpha x\lambda z.zy$

f. $x\lambda x.xy =_\alpha y\lambda x.xy$

5.4 β-reducția

β-reducția este o relație definită pe mulțimea α-termenilor, $\beta \subseteq \Lambda T \times \Lambda T$, unde

[Aplicarea] $(\lambda x.y)u \rightarrow_\beta [u/x]t$

[Compatibilitatea] $t_1 \rightarrow_\beta t_2$ implică $tt_1 \rightarrow_\beta tt_2$, $t_1 t \rightarrow_\beta t_2 t$ și $\lambda x.t_1 \rightarrow_\beta \lambda x.t_2$

Închiderea reflexivă și tranzitivă a acestei relații se notează $\rightarrow_\beta^* \subseteq \Lambda T \times \Lambda T$, iar $t_1 \rightarrow_\beta^* t_2$ dacă există $n \geq 0$ și $u_0, ..., u_n$ asfel încât $t_1 =_\alpha u_0 \rightarrow_\beta u_1 \rightarrow_\beta ... \rightarrow_\beta u_n =_\alpha t_2$.

Exercițiul 4. *Să se aplice două β-reducții succesive asupra λ-termenului $(\lambda x.(\lambda y.yx)z)v$. Este o singură variantă de aplicare?*

Un termen poate fi β-redus în mai multe moduri, iar proprietatea de confluență ne asigură că vom ajunge întotdeauna la același rezultat. (forma normală este unică, modulo α-echivalență)

5.5 Numeralii Church

Fie numerele naturale definite prin:

- $0 := \lambda fx.x$
- $1 := \lambda fx.fx$
- $2 := \lambda fx.f(fx)$
- ...
- $m := \lambda fx.f^m x$
- ...

Fie functiile

- **succ** $:= \lambda nfx.f(nfx)$
- **add** $:= \lambda mnfx.mf(nfx)$

Sa se arate ca

- **succ** $2 =_\alpha 3$

Demonstratie.

$$\begin{aligned} \textbf{succ}\ 2 &:= (\lambda nfx.f(nfx))(\lambda f'x'.f'(f'x')) \to_\beta \\ &\to_\beta [(\lambda f'x'.f'(f'x')/n]\lambda fx.f(nfx) =_\alpha \\ &=_\alpha \lambda fx.f((\lambda f'x'.f'(f'x')fx) \to_\beta^* \\ &\to_\beta^* \lambda fx.f(f(fx)) =: 3 \end{aligned}$$

- **add** $3\ 1 =_\alpha 4$

Demonstratie.

$$\begin{aligned} \textbf{add}\ 3\ 1 &:= (\lambda mnfx.mf(nfx))(\lambda f'x'.f'(f'(f'x)))(\lambda f''x''.f''x'') \to_\beta^* \\ &\to_\beta^* [\lambda f'x'.f'(f'(f'x))/m][\lambda f''x''.f''x''/n](\lambda fx.mf(nfx)) =_\alpha \\ &=_\alpha \lambda fx.(\lambda f'x'.f'(f'(f'x)))f((\lambda f''x''.f''x'')fx) \to_\beta^* \\ &\to_\beta^* \lambda fx.(\lambda f'x'.f'(f'(f'x)))f(fx) =_\alpha \\ &=_\alpha \lambda fx.f(f(f(fx))) =: 4 \end{aligned}$$