

? $\vdash \bar{M} : ?$

examens

12 - 15.06

19 - 25.06

$$\Gamma_M = \{ x : x \mid x \in \text{TV}(M) \}$$

$$M = x \rightarrow \bar{M} = M$$

$$M = M_1 N_1 \rightarrow \bar{M} = \bar{N}_1 \bar{M}_1$$

$$M = \lambda x. N \rightarrow \bar{M} = \lambda x. x \cdot \bar{N}$$

$\Gamma \vdash M : \tau \circ C \rightarrow \text{constrainti}$

$\{ \tau \equiv \tau' \} \quad (\text{praktisch nicht genügt } ?)$

→ example slide 6

Numeros

$$+(-2, 1)$$

$$\times(x, y)$$

2, 1 const

- fct. de aut 1

+ fct. de aut 2

NU EVALUAM !Trippel $\rightarrow (\text{Maybe Int}, \text{Bool})$

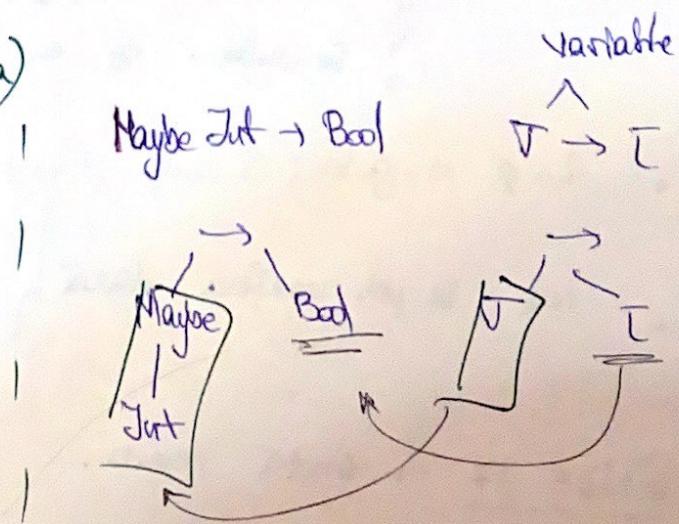
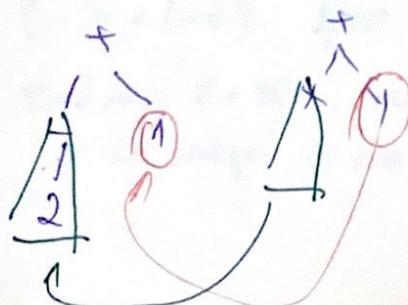
Int, Bool

Maybe fct. de aut 1

→ aut 2

$$\text{Maybe Int} \rightarrow \text{Bool}$$

Lubstetutire: {Se pot unifica}
 $\times(-2, 1) \doteq \times(x, y)$



variable
 \wedge
 $\tau \rightarrow \tau'$

Maybe Int-Bool | Maybe T
 \xrightarrow{f} |
 +(-2, 1) = -2 → Nu se poate
 $\begin{array}{c} + \\ \diagdown \quad \diagup \\ \text{Nu} \\ \diagup \quad \diagdown \\ - \end{array}$
 nu se poate
 rezolvare grea

Nu îl evaluăm,
acercăm să pythag
ortonii.



Algoritm de unificare (Scrierea de EXATEN)

S = lista sol.

R = lista de reg.

$S = \emptyset$

$R = \{t_1=t_2, \dots, t_{n-1}=t_n\}$ fiecare termen în ~~nu~~ nu
o ecuație

1. SCOPER : eliminarea entre ec. $t=t$

2. DESCOPRIMUȚE: $f(t_1, \dots, t_n) = f(t'_1, \dots, t'_n)$ se rulează
cu $t_1=t'_1, t_2=t'_2, \dots, t_n=t'_n$

3. RESOLVĂ: $x=t$ sau $t=x$
 x variabilă care nu apare în t

• SE MUTĂ în S ca $x=t$

• Se inseră x cu t în $R \setminus S$.

arități diferențiale

• $R = \emptyset \Rightarrow$ gata; S cont. cunoscute

• nu se pot unifica date

1. R cont. $f(t)=g(t)$
2. R cont. $x=t$ sau $t=x$
⇒ nu se operează cu t .

SLIDE 17 → tratată teoria.

Exemplu:

R

$$\left\{ \begin{array}{l} g(y) = x, f(x, h(x), y) = f(g(z), w, z) \end{array} \right\} \text{ AESC}$$

1. descompune

$$g(y) = x, x = g(z), h(x) = w \quad \text{REZ.} \quad \text{z, y, z, w variab.}$$

$$y = z \quad g(z) = x, \quad \cancel{x = g(z)}, \quad h(x) = w \quad \text{REZ.} \quad f \text{ fet. de out-3}$$

$$x = g(z) \quad g(z) = g(z), \quad \cancel{g(z)} = h(\cancel{z}) = w \quad \text{ELIM.}$$

$$y = z$$

$$x = g(z)$$

$$y = z$$

$$w = h(g(z))$$

$$x = g(z)$$

$$y = z$$

$$\theta(w) = h(g(z))$$

$$\theta(x) = g(z)$$

$$\theta(y) = z$$

✓

$$\begin{cases} x \\ y = g(z) \\ z = z \end{cases}$$

$$g(z) = g(z) \quad \checkmark$$

$$f(g(z), h(g(z)), g(z)) = f(g(z), h(g(z)), z) \quad \checkmark$$

Exemplu 2	
<u>S</u>	<u>R</u> $g(y) = x, f(x, h(y), y) = f(g(z), h, z)$ Regr.
<u>$x = g(y)$</u>	$f(g(z), h(y), y) = f(g(z), h, z)$ Desc.
<u>$x = g(y)$</u>	$g(y) = g(z), [h(y) = b], z = z$ autore differente
	Nu se poate unitrea h, b au alt. dif.
$h(y) = \text{const}$ $h(y) = f(x)$ $b = c (\text{const})$	oprire
	\therefore peste tot

$b = y$ (const = var) \Rightarrow DA, merge

Exemplu 3	
<u>S</u>	<u>R</u> $g(y) = x, f(x, h(x), y) = f(z, w, z)$ Regr.
<u>$x = g(y)$</u>	$f(g(y), h(g(y)), y) = f(y, w, z)$ Desc.
<u>$x = g(y)$</u>	$g(y) = z, h(g(y)) = w, z = z$ Reg.
<u>$z = g(z)$</u>	$h(g(y)) = w$
<u>$z = g(z)$</u>	$y = g(y)$ $g(z) = z$ nu \exists cupt!
	z apare în g y apare în g

Afg. de wuy. pt $M = XX$.

③

Evidență

1.

S	R
	$p(a, x, h(g(y))) = p(z, h(z), h(u))$ DESC
$\cancel{z=a}$	$x = h(z), \neg h(g(y)) = h(u)$ leg
$\cancel{z=a}$ $x = h(a)$	$\cancel{x = h(a)}, h(g(y)) = h(u)$ leg DESC
$\cancel{z=a}$ $x = h(a)$	$h(g(y)) = h(u)$ DESC
$\cancel{g(y)=u}$ $\cancel{z=a}$ $\cancel{x = h(a)}$	$\cancel{g(y) = u}$
	✓
	✗
	$f(h(a), g(x)) = f(y, y)$ DESC
$\cancel{h(a) = y}$	$h(a) = y, g(x) = y$ leg
$h(a) = y$	$g(x) = y$ leg
$\cancel{h(a) = g(x)}$	✗ Nu are l

2.

	$p(a, x, g(x)) = p(a, z, y)$ DESC
	$a = a, x = z, g(x) = y$ elim
	$x = z, g(x) = y$ leg
$x = z$	$g(z) = y \rightarrow$ Nu se poate, y apare în conjuncție

3.

⑤

<u>S</u>	<u>R</u>	
	$\varphi(x, y, z) = \varphi(u, \varphi(v, v), u)$	base
$x = u$	$y = f(v, v), z = u$	Reg
$x = u$	$y = f(v, v), z = u$	Reg
$z = u$	$y = f(v, v)$	Reg
$x = z$		
$y = f(v, v)$		
$z = u$	\emptyset	
$x = z$		

lectura

FLP
curs 8 sept. '11

08.05.2023

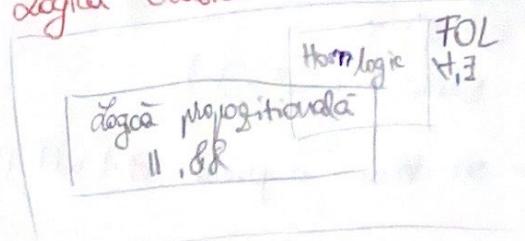
- curs 03 - programare logică
programul lui Ernster

PROGRAM = LOGICĂ + CONTROL

FOL = first-order logic

- example:
 $\text{oslo} \rightarrow \text{windy}$ ~~oslo :- windy~~
 $\text{oslo} \rightarrow \text{norway}$
 $\text{norway} \rightarrow \text{cold}$
 $\text{cold} \wedge \text{windy} \rightarrow \text{winterIsComing}$
 oslo
—
 $\text{winterIsComing}?$

Logică clasica



19.06.2023

1 si
v sau

windy :- oslo.
oslo.

- Swish .swi - prodg .org
- const. cu literă mare / minuscule / "Bcd" / * → Casa = val.
- ravitate family
Griffon

casa = atunci
casa = val.

ane are mere, ~~✓~~

- griff(x) :- gfather(y,x), griff(y).

pred. de avatare
gfather/2

- griff(x) → regulat
griff(y) → ce valori poate avea y
variabila

- Head :- body
reguli form body = fapte

Head :-
} { Sunt multe de

Operatori:

= unificare

= compară (nu le și unifică)

\= different

• Ex: verificare dacă este lista

isList([]),

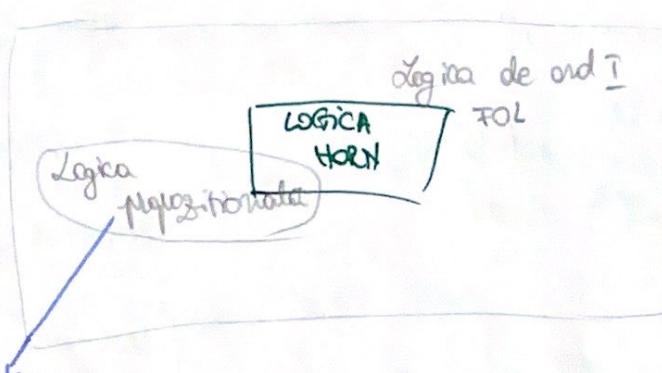
isList([-1 -])

- un termen e parțial el | ult | coada listei!

~~isList([*])~~

last([x], x)

last([-1 T], y) :- last(T, y)

FLP
curs 10

Există o infinitate de logici de ord. 1 și fol. de alfabet.

Variabile: $x, y, t \dots$ $x, \exists x, \forall x \rightarrow y,$ $\exists x \forall y \rightarrow z$ $(\exists x) \wedge (\forall x \rightarrow y)$

Alfabet: $\rightarrow R: simboluri de relații$ (fiecare cu o aritate aritale)

$\rightarrow f: simboluri de funcție de aritățile $f: FUR \rightarrow M^*$$

$\rightarrow c: simboluri de constante$

$\models ar(R), ar(f), ar(c)$

termenii compusi = predicabile
operatori = funcții (\wedge, \vee) $(father(x, y))$

relație \rightarrow simboluri
funcție \rightarrow simboluri

$$\text{ex: } R = \{R, P\} \\ f = \{f\}$$

FOL

$$R = V_P \\ ar(R) = 0$$

$$C = \{c\}$$

$f(x_1, c) \rightsquigarrow$ termen

nu pot fi scris; DAR pot fi scris $f(x_1, c) \rightsquigarrow x_1, M_L$
 $P(f(x_1, c)) \mapsto P(x_1)$

$$\forall x_1 \exists y_1 \dots$$

$\exists f(x_1, c) \rightsquigarrow$ formula atomică

În FOL, sunte luate fără consecuri rezolvare și
aceșia cu simbol de relație.

181
✓ Sem

~~P CERT~~

Ecuțiu : $\begin{cases} x = 4 \in \mathbb{Z}^2 \\ f = \{x_1 + x_2\} \\ c = 403 \end{cases}$ relații
functii

SCHEMĂ :

termeni : $0 + s(0)$ $0 + x$
 $x + y$ $s(0) + y$

notam numerele x, y
cu semne +

formule atomice : $s(0) < 0$
 $x + y < 0 + x$; $0 + x < s(0 + y)$
 $s(0) + y < x + y$

termenii cu

formule :

$(x + y < 0 + x) \vee (0 + x < s(0 + y))$
 $(s(0) + y < x + y) \wedge (x + y < 0 + x)$

formule atomice
cu II, III,
7, 8

$\neg(x + y < 0 + x)$

$\neg \Psi \equiv \neg \Psi \vee \Psi$

Literei : $p(\text{term}) \neg p(\text{term})$ (formule atomice / negație a unei
formule atomice)

$\wedge (\vee \neg)$

"pacete de literale"
disjuncție de literale

= CLAUZA

CLAUZA TRIVIALĂ

$\boxed{p(x) \vee \neg p(x)} \vee q(x, c)$

$\{p(x), \neg p(x)\}$

$\forall x \Rightarrow$ clauza vidă \square

literali
 $\{L_1, \dots, L_m\}$
 clauză

- punem grupa neg. în fapt: $\{TQ_1 \rightarrow \neg Q_m, P_1, \dots, P_n\} \rightarrow Q_1 \dots P_k =$ formulele astfelite
- adăugăm $\vdash x_1 \rightarrow \vdash_{\text{ex}}(TQ_1 \vee \dots \neg \neg TQ_m \vee P_1 \vee P_2 \dots \vee P_k)$

\Rightarrow O clauză e de fapt $Q_1 \dots \neg \neg Q_m \rightarrow P_1 \vee \dots \vee P_k$,

unde Q este cele care apăr cu T și
 P cele care apar fără.

$\begin{cases} k=1: n>0 & : Q_1, \dots \neg \neg Q_m \rightarrow P \\ n=0 & : T \rightarrow P \end{cases}$ (unitate, mereu ADEV) - clauza program def.

La HOLN:

$k=0 \rightarrow$ clauză scop (inșehări)

$Q_1 \dots \neg \neg Q_m \mapsto \perp$ formule monice false

$n=k=0 \rightarrow$ clauză vidă

\Rightarrow def: O clauză HOLN e o clauză program definită

$\begin{cases} (k>1): n>0: Q_1 \dots \neg \neg Q_m \rightarrow P \\ n=0: T \rightarrow P \end{cases}$ sau o clauză scop, în care

$k=0: Q_1 \dots \neg \neg Q_m \rightarrow \perp$

$k=1 \Rightarrow$ ~~clauzele din prolog~~; $k=0 \Rightarrow$ adrefări

$n=0 \Rightarrow$ fapte

Prolog

$\sum Q_1, Q_2 \quad \exists z (Q_1 \wedge Q_2(z))$

Neg:

$$\begin{aligned} \neg \exists z (Q_1 \wedge Q_2(z)) &= \forall z (\neg (Q_1(z) \vee Q_2(z))) \\ &= \forall z (\neg Q_1(z) \wedge \neg Q_2(z)) \end{aligned}$$

Knowledge base: $K_B \vdash Q_1 \wedge Q_n \rightarrow$
 cuant. universal cuant. existențial
 $\Delta_{\text{SLD}} \models Q_2 = 0$
 \rightarrow derivarea se înșează
 SLD - rezolvare

$$\text{arc}(k_{xy}, z) \doteq \text{arc}(x, y)$$

$$\Theta \Rightarrow h_{k_{xy}} \rightarrow k_{xy}, f^{+z}$$

$$\text{arc}(k_{xy}, z) \doteq \text{dom}(x, y) \rightarrow \text{nu se pot verifica}$$

$$\nexists \text{cang}$$

Axiomele vorbă ca fct. în art. de dezvoltare

Reguli de plasaj: - metr. cele mai importante principii
 (cel de către de sus în jos)

PROLOG ESTE INCOMPLET,
 baza la logica SLD

$$\begin{array}{c}
 \text{stg. la dr} \\
 \overline{Q_1 \vee \neg \exists Q_2 \rightarrow Q_m} \\
 \Theta \vdash Q_1 \vee \neg P_1 \vee \neg P_m \vee Q_2
 \end{array}$$

Teorema \rightarrow \nexists sld res p a lui $Q_1 \wedge \neg Q_m$ din K_B

$$\begin{array}{c}
 (\Rightarrow) \\
 K_B \vdash Q_1 \wedge Q_m
 \end{array}$$

$$: \doteq \Rightarrow \vee \neg \exists$$

Ex. pe slide 31 // se ajunge la clauza următoare

ex. de examen 36

FLP

curs săpt. 13

Limbaj de programare : Sintaxă = rul., cuv. cheie
 Practic = cum e folosit
 Semantica \rightsquigarrow înțelegerea limbajului
 \rightsquigarrow precizarea unei noi limbaj
 \rightsquigarrow baza pentru demonstrarea corectitudinii

Problema corectitudinii programelor :

- în raport cu ce?
- metode ce arată "celitatea", în găsirea greșalilor

exemplu :

int main (void) {

 int $x=0;$

 return ($x=1 \rightarrow x=2;$) ; \rightarrow 3 sau 4? definire de cauza erorii

}

Limbajul reprezentativ IMP

- expr \rightarrow citim $x+3$
 \rightarrow boala $x>7$

- instr \rightarrow atrib.
 \rightarrow if (cond, if, else)
 \rightarrow while ($x>7$, $x=x-1$)

- compoziție instr.
- bloaci de instrucțiuni

Program = (Struct, n)

Cum ne definim sintaxa BNF a limbajului IMP:

$E ::= m | x$ (expresii)

$| E + E | E \cdot E$ (bulevi)

$B ::= \text{true} / \text{false}$

$| E \geq E | E >= E | E == E$
 $| \text{not}(B) | \text{and}(B, B) | \text{or}(B, B)$

$C ::= \text{skip}$ (instrucțiuni)

$| x = E$
 $| \text{if}(B, C_1, C_2)$
 $| \text{while}(B, C)$

$| h C_1 | C ; C$

$P ::= h C_1 E$ (forma progr., expresii fi val. în care
introduce)

Semantica:

operanță = cum se exec. un progr. pe o mașină, blestă.
cel mai mult pas de
small-step: $\frac{(cool, \sigma) \rightarrow (cool, \sigma'))}{(\text{un sg. pas mai mare})}$, slujite

Starea executării: $\sigma: \text{Var} \rightarrow \text{Int}$ (st. parțială). $\sigma_{x \leftarrow v} (y) \rightarrow \begin{cases} \sigma(y), & y \neq x \\ v, & y = x \end{cases}$

exemplu:
 $x = 0, \sigma = \{x = 0\}, \perp \rightarrow \{x = 0+1, \sigma \mapsto 0\} \rightarrow x = 0+1, \sigma \mapsto 0$
 $\text{dom}(f) = \emptyset \rightarrow \{x = 1, \sigma \mapsto 0\} \rightarrow \{x = 1, \sigma \mapsto 1\}$
nu e def. mutabil.

$\text{if } (0 <= 5 + 7 \text{ } \textcircled{x}), n = 1, r = 0$

$\text{Lif} (\text{true}, b_1, b_2), \sigma \rightarrow \{b_1, \sigma\}$

\exists K framework \rightarrow definește un lanțaj
semantice pentru un lanțaj nu definit

Lemantica axiomatice: valoarea rest. pe baza unor reguli
logice ; $\{P\} \subseteq \{Q\}$

$$\begin{array}{c} \{x=1\} \\ P \end{array} \quad \begin{array}{c} x=x+1 \\ C \end{array} \quad \begin{array}{c} \{x=2\} \\ Q \end{array}$$

corect dăv. progr. și exec. într-o
stare ce satgl. Q și progr. și

$\{T\} \{if(x > y) z = x; \text{else } z = y\} \{z^{out}_{(x,y)}\}$ termină (corect - parțial)
corect total = și cum se termină?

$$x=0 \rightarrow x \neq 0$$