**FMI, Info, Anul II, 2020-2021**
**FLP**

# Exemplu examen

**(S1.1)** **Unificare.** Considerăm

- $x, y, z, u$ variabile,
- $a$ simbol de constantă,
- $h, g$ simboluri de funcție de aritate 1,
- $p$ simbol de funcție de aritate 3.

1 Aplicați algoritmul de unificare din curs pentru a găsi un unificator pentru termenii:

$$p(a, x, h(g(y))) \text{ și } p(z, h(z), h(u))$$

**Soluție.**

| Regula aplicată | Lista soluție<br>S | Lista de rezolvat<br>R |
|---|---|---|
| Inițial | $\emptyset$ | $p(a, x, h(g(y))) \doteq p(z, h(z), h(u))$ |
| 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 | $z \doteq a, x \doteq h(a), u \doteq g(y)$ | $\emptyset$ |

2 Aplicați algoritmul de unificare descris din curs pentru a găsi un unificator pentru termenii:

$$\{g(y), x, p(x, h(y), y), p(g(z), b, z)\}$$

Problema revine la gasirea unui unificator pentru perechile

$$\{g(y) \doteq x,\ p(x, h(y), y) \doteq p(g(z), b, z)\}$$

| | $S$ | $R$ |
|---|---|---|
| REZOLVĂ | $\emptyset$ | $g(y) \doteq x,\ p(x, h(y), y) \doteq p(g(z), b, z)$ |
| DESCOMPUNE | $x \doteq g(y)$ | $p(g(y), h(y), y) \doteq p(g(z), b, z)$ |
| - EŞEC - | $x \doteq g(y)$ | $g(y) \doteq g(z), h(y) \doteq b,\ y \doteq z$ |

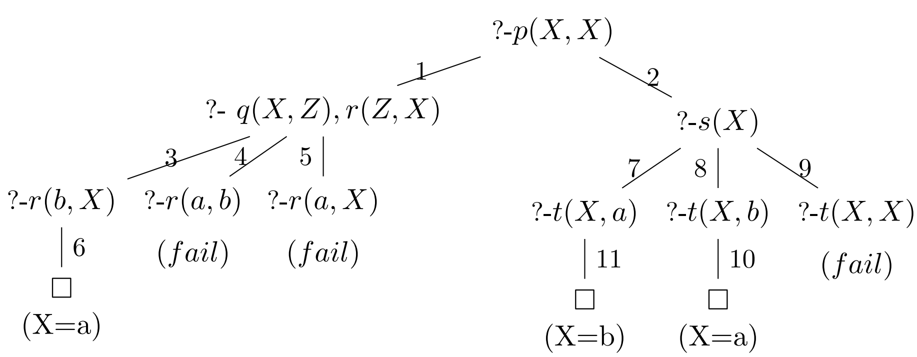
- $h$ şi $b$ sunt simboluri de operaţii diferite!
- Nu există unificator pentru ecuaţiile din $U$.

**(S1.2)** $\boxed{\textbf{Execuția programelor în Prolog}}$ Considerăm următorul program Prolog

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).`

Să se descrie arborele de execuție al query-ului $p(X,X)$.

**Soluție 1.**




**Soluție 2.** Pentru a rezolva $p(X,X)$ putem aplica regula (1) sau (2)

(i) Aplicăm regula (1).

- redenumim regula (1) în `p(X1,Y1) :- q(X1,Z1), r(Z1,Y1).`
- unificăm $p(X,X)$ cu $p(X1,Y1)$ — obținem $\theta_1 = X1 \doteq X, Z1 \doteq X$
- înlocuim $p(X,X)$ cu $q(X1,Z1), r(Z1,Y1)$ și aplicăm substituția $\theta_1$
- obținem noua țintă $q(X,Z1), r(Z1,X)$

- Trebuie să rezolvăm $q(X, Z1)$. Putem aplica regulile (3), (4), sau (5)
  (a) Aplicăm regula (3)
    - redenumim regula (3) în `q(X2,b).`
    - unificăm $q(X, Z1)$ cu $q(X2, b)$ — obținem $\theta_2 = X2 \doteq X, Z1 \doteq b$
    - înlocuim $q(X, Z1)$ cu $\emptyset$ și aplicăm substituția $\theta_2$
    - obținem noua țintă $r(b, X)$
    - Putem aplica doar regula (6). Nu avem ce redenumi
      * unificăm $r(b, X)$ cu $r(b, a)$ — obținem $\theta_3 = X \doteq a$
      * înlocuim $r(b, X)$ cu $\emptyset$
      * obținem ținta vidă. Avem soluția $\theta = X \doteq a$

  (b) Aplicăm regula (4). Nu avem ce redenumi.
    - unificăm $q(X, Z1)$ cu $q(b, a)$ — obținem $\theta_2 = X \doteq b, Z1 \doteq a$
    - înlocuim $q(X, Z1)$ cu $\emptyset$ și aplicăm substituția $\theta_2$
    - obținem noua țintă $r(a, b)$
    - Putem aplica doar regula (6). Nu avem ce redenumi
      * unificăm $r(a, b)$ cu $r(b, a)$ — unificare eșuată, backtrack

  (c) Aplicăm regula (5)
    - redenumim regula (3) în `q(X2,a) :- r(a, X2).`
    - unificăm $q(X, Z1)$ cu $q(X2, a)$ — obținem $\theta_2 = X2 \doteq X, Z1 \doteq a$
    - înlocuim $q(X, Z1)$ cu $r(a, X2)$ și aplicăm substituția $\theta_2$
    - obținem noua țintă $r(a, X), r(b, X)$
    - Putem aplica doar regula (6). Nu avem ce redenumi
      * unificăm $r(a, X)$ cu $r(b, a)$ — unificare esuată, backtrack

(ii) Aplicăm regula (2)

- redenumim regula (2) în `p(X1,X1) :- s(X1).`
- unificăm $p(X, X)$ cu $p(X1, X1)$ — obținem $\theta_1 = X1 \doteq X$
- înlocuim $p(X, X)$ cu $s(X1)$ și aplicăm substituția $\theta_1$
- obținem noua țintă $s(X)$. Putem aplica regulile (7), (8), sau (9)
  (a) Aplicăm regula (7)
    - redenumim regula (7) în `s(X2) :- t(X2, a).`
    - unificăm $s(X)$ cu $s(X2)$ — obținem $\theta_2 = X2 \doteq X$
    - înlocuim $s(X)$ cu $t(X2, a)$ și aplicăm substituția $\theta_2$

- obținem noua țintă $t(X, a)$
- Putem aplica regula (10) sau (11)

(1) Aplicăm regula (10). Nu avem ce redenumi
  * unificăm $t(X, a)$ cu $t(a, b)$ — unificare eșuată, backtrack

(2) Aplicăm regula (11). Nu avem ce redenumi
  * unificăm $t(X, a)$ cu $t(b, a)$ — obținem $\theta_3 = X \doteq b$
  * înlocuim $t(X, a)$ cu $\emptyset$
  * obținem ținta vidă. Avem soluția $\theta = X \doteq b$

(b) Aplicăm regula (8)
- redenumim regula (8) în `s(X2) :- t(X2, b).`
- unificăm $s(X)$ cu $s(X2)$ — obținem $\theta_2 = X2 \doteq X$
- înlocuim $s(X)$ cu $t(X2, b)$ și aplicăm substituția $\theta_2$
- obținem noua țintă $t(X, b)$
- Putem aplica regula (10) sau (11)

(1) Aplicăm regula (10). Nu avem ce redenumi
  * unificăm $t(X, b)$ cu $t(a, b)$ — obținem $\theta_3 = X \doteq a$
  * înlocuim $t(X, b)$ cu $\emptyset$
  * obținem ținta vidă. Avem soluția $\theta = X \doteq a$

(2) Aplicăm regula (11). Nu avem ce redenumi
  * unificăm $t(X, b)$ cu $t(b, a)$ — unificare eșuată, backtrack

(c) Aplicăm regula (9)
- redenumim regula (9) în `s(X2) :- t(X2, X2).`
- unificăm $s(X)$ cu $s(X2)$ — obținem $\theta_2 = X2 \doteq X$
- înlocuim $s(X)$ cu $t(X2, X2)$ și aplicăm substituția $\theta_2$
- obținem noua țintă $t(X, X)$
- Putem aplica regula (10) sau (11)

(1) Aplicăm regula (10). Nu avem ce redenumi
  * unificăm $t(X, X)$ cu $t(a, b)$ — unificare eșuată, backtrack

(2) Aplicăm regula (11). Nu avem ce redenumi
  * unificăm $t(X, X)$ cu $t(b, a)$ — unificare eșuată, backtrack

**(S1.3)** $\boxed{\textbf{Variabile libere / variabile legate.}}$ Considerăm $\lambda$-expresia

$$\lambda x.(\lambda y.(\lambda x.x + z)\ ((\lambda x.x\ x)\ (x + y) + x))\ (\lambda z.x\ y\ z)$$

**Soluție.** Adnotăm fiecare apariție a unei variabile cu poziția sa în formula pentru a putea exprima mai ușor relațiile de legătură:

$$\lambda x^1.(\lambda y^2.(\lambda x^3.x^4 + z^5)\ ((\lambda x^6.x^7\ x^8)\ (x^9 + y^{10}) + x^{11}))\ (\lambda z^{12}.x^{13}\ y^{14}\ z^{15})$$

Aparițiile variabilelor îndeplinesc următoarele roluri:

$x^1$ este variabilă de legătură

$y^2$ este variabilă de legătură

$x^3$ este variabilă de legătură

$x^4$ este legată de $x^3$

$z^5$ este liberă

$x^6$ este variabilă de legătură

$x^7$ este legată de $x^6$

$x^8$ este legată de $x^6$

$x^9$ este legată de $x^1$

$y^{10}$ este legată de $y^2$

$x^{11}$ este legată de $x^1$

$z^{12}$ este variabilă de legatură

$x^{13}$ este legată de $x^1$

$y^{14}$ este liberă

$z^{15}$ este legată de $z^{12}$

**(S1.4)** $\boxed{\textbf{Reguli de tipuri.}}$ Vrem să adăugăm liste la limbajul LAMBDA prezentat la curs. Pentru aceasta avem nevoie e o modalitate de a construi liste și o modalitate de a de ”deconstrui”. Ca să realizăm acest lucru, adăugăm la LAMBDA următoarele expresii:

**empty** care desemnează lista vidă

**cons** funcție care primește ca argumente un element și o listă de elemente de același tip și produce o listă

**uncons** care primește trei argumente, `l`, `ifEmpty` și `ifCons` cu următoarele semnificații:

- **l** lista care trebuie procesată
- **ifEmpty** constanta care să fie folosită dacă lista **l** e vidă
- **ifCons** funcția care să fie folosită dacă lista **l** e nevidă: ea va lua ca argument capul și coada listei și va produce un rezultat de același tip cu **ifEmpty**.

De exemplu, folosind **uncons** putem să definim următoarele funcții noi

**null** `let(null, l -> uncons $ l $ true $ (a -> b -> false),`
`null $ (cons $ 4 $ empty))`

**tail** `let(tail, l -> uncons $ l $ empty $ (h -> t -> t),`
`tail $ (cons $ 4 $ empty))`

**head** `let(head, default -> l -> uncons $ l $ default $ (h -> t -> h),`
`head $ 0 $ (cons $ 4 $ empty))`
(pentru head am adăugat o constantă de eroare)

Extindeți sistemul de tipuri pentru limbajul LAMBDA prezentat la curs pentru a încorpora această extensie.

**Soluție.** Adăugăm un nou constructor de tipuri, `[a]` pentru liste cu elemente de tipul `a`, și următoarele reguli:

```
type(_, empty, [_]).
type(_, cons, T -> [T] -> [T]).
type(_,uncons, [TL] -> T -> (TL -> [TL] -> T) -> T).
```

**(S1.5)** $\boxed{\textbf{Semantica operaţională.}}$ Pentru acest exercitiu folosim limbajul IMP definit în curs. Folosind regulile semanticii operationale ”small step” definite in curs justificati pasii de tranzitie intre configuratia

$$\langle \texttt{if}\ (0 \texttt{ <= } i\ ,\ i \texttt{ = } i \texttt{ + } -4\ \texttt{;while}\ (0 \texttt{ <= } i, \{\ i \texttt{ = } i \texttt{ + } -4\ \})\ \texttt{,skip})\ ,\ i \mapsto 3\rangle$$

şi configuraţia

$$\langle i \texttt{ = } i \texttt{ + } -4\ \texttt{;while}\ (0 \texttt{ <= } i, \{\ i \texttt{ = } i \texttt{ + } -4\ \})\ ,\ i \mapsto 3\rangle$$

indicând termenul prelucrat de regulă.

**Soluție.** Tranziţiile sunt:

$$\begin{aligned}
&\langle \texttt{if}\ (0 \texttt{ <= } \underline{i}\ ,\ i \texttt{ = } i \texttt{ + } -4\ \texttt{;while}\ (0 \texttt{ <= } i, \{\ i \texttt{ = } i \texttt{ + } -4\ \})\ \texttt{,skip})\ ,\ i \mapsto 3\rangle \xrightarrow{\textsc{Id}}\\
&\langle \texttt{if}\ (\underline{0 \texttt{ <= } 3}\ ,\ i \texttt{ = } i \texttt{ + } -4\ \texttt{;while}\ (0 \texttt{ <= } i, \{\ i \texttt{ = } i \texttt{ + } -4\ \})\ \texttt{,skip})\ ,\ i \mapsto 3\rangle \xrightarrow{\textsc{Leq-true}}\\
&\langle \underline{\texttt{if}\ (\texttt{true},\ i \texttt{ = } i \texttt{ + } -4\ \texttt{;while}\ (0 \texttt{ <= } i, \{\ i \texttt{ = } i \texttt{ + } -4\ \})\ \texttt{,skip})}\ ,\ i \mapsto 3\rangle \xrightarrow{\textsc{If-true}}\\
&\langle i \texttt{ = } \underline{i} \texttt{ + } -4\ \texttt{;while}\ (0 \texttt{ <= } i, \{\ i \texttt{ = } i \texttt{ + } -4\ \})\ ,\ i \mapsto 3\rangle
\end{aligned}$$