λ-calcul

Variabile libere si legate

$\lambda x.t$ - toate aparitiile lui x in t sunt aparitii legate

$\lambda x.xy$ - x este legata, y este libera
$\lambda x.zt$ - toate variabilele (z si t) sunt libere
$\lambda xy.xyx$ - toate variabilele sunt legate. Acesta este un λ-termen inchis.

Exercitiu

$FV(\lambda x.xy) = FV(xy) - \{x\} = (FV(x) \cup FV(y)) - \{x\} = (\{x\} \cup \{y\}) - \{x\} = \{y\}$

$FV(x\lambda x.xy) = FV(x) \cup FV(\lambda x.xy) = \{x\} \cup \{y\} = \{x, y\}$

$FV(x(\lambda xy.xyz)(\lambda v.yv)) = FV(x) \cup FV(\lambda xy.xyz) \cup FV(\lambda v.yv) = \{x, y, z\}$

1. $FV(x) = \{x\}$
2. $FV(\lambda xy.xyz) = FV(\lambda x.\lambda y.xyz) = FV(\lambda y.xyz) - \{x\} = (FV(xyz) - \{y\}) - \{x\} = (FV(x) \cup FV(y) \cup FV(z) - \{y\}) - \{x\} = \{x, y, z\} - \{y\} - \{x\} = \{z\}$
3. $FV(\lambda v.yv) = \{y\}$

$FV(\lambda t.((\lambda xyz.yzx)t)) = \emptyset$, deci avem un λ-termen inchis.

Substitutii

1. $[y/x]\lambda z.x = \lambda z.[y/x]x = \lambda z.y$
2. $[y/x]\lambda y.x$ - nu pot efectua substitutia

- am $x \neq y$;
- dar nu satisfac conditia $y \notin FV(y) = \{y\}$

3. $[\lambda z.z/x](\lambda x.yx)$ - nu pot efectua substitutia

- nu satisfac $x \neq x$

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

α-conversie

$\lambda x.xyz =_\alpha \lambda r.ryz$

$\lambda x.x =_\alpha \lambda z.z$

β-reductia

$(\lambda x.t)u \to_\beta [u/x]t$

```
(\x -> x + 1) 2 -beta-> 2 + 1 = 3
```

$t \to_\beta^* t_1$
$t \to_\beta^* t_2$
$t \to_\beta^* t_3$

$t_1 =_\alpha t_2 =_\alpha t_3$

Daca rescriu t intr-un t_1, si din t_1 nu mai pot aplica nicio β-reductie, atunci t_1 se numeste β-forma normala.

Pentru un λ-termen, β-forma normala este unica modulo α-conversie.

Sistemul de rescriere prin β-reductii este confluent.

$t \to_\beta^* \lambda z.z$
$t \to_\beta^* \lambda x.x$

Exercitiu

$(\lambda x.(\lambda y.yx)z)v \to_\beta [v/x]((\lambda y.yx)z) =_\alpha (\lambda y.yv)z =_\alpha [z/y](yv) =_\alpha zv$ - β forma normala

$(\lambda x.(\lambda y.yx)z)v \to_\beta (\lambda x.[z/y](yx))v =_\alpha (\lambda x.zx)v \to_\beta [v/x](zx) =_\alpha zv$ - β forma normala

Cerinte de laborator

Implementarea λ-calculului in Haskell

Fie urmatoarele tipuri de date:

Tipul pentru variabile:

```
type Variable = Char
```

Tipul de date algebric pentru expresiile din λ-calcul:

```
data Term = V Variable
    | App Term Term
    | Lam Variable Term
    deriving (Eq, Show)
```

$$l_1 = z\lambda x.xy$$

```
lambda1 = App (V 'z') (Lam 'x' (App (V 'x') (V 'y')))
```

Cerinta 1 Definiti urmatorii λ-termeni in Haskell:

1. $\lambda x.xyz$
2. $(\lambda x.xy)(y\lambda s.sz)$
3. $\lambda sz.ssz$

Cerinta 2 Definiti o functie care sa calculeze multimea variabilelor libere dintr-un λ-termen.

```
freeVars :: Term -> [Variable]
freeVars (V x) = [x]
```

```
freeVars (App t1 t2) = nub $ (freeVars t1) ++ (freeVars t2)
freeVars (Lam x t) = undefined
```

Cerinta 3 Definiti o functie care sa calculeze substitutia unei variabile cu un termen intr-un alt termen.

```
--         u        x          t      result
-- result = [u / x] t
subst :: Term -> Variable -> Term -> Term
subst u x (V y)
    | x == y = u
    | otherwise = V y
subst u x (App t1 t2) = App (subst u x t1) (subst u x t2)
subst u x (Lam y t) = undefined
```

Solutiile se trimit pe adresa *bogdan.macovei.fmi@gmail.com* cu subiectul **Grupa, Nume Prenume, Lambda calcul** pana pe 19 mai 2022, seara (23:59 sau putin dupa). Si acest laborator este notat pentru punctajul bonus.