Semânticas de Linguagens de Programação - CC3033

Exercícios 6

Introdução ao λ-calculus e Conceitos de linguagens funcionais

Conceitos básicos de λ-calculus

1. Para cada uma das expressões seguintes:

   - determina as que correspondem a um λ-termo.
   - para cada uma que for um λ-termo e indica o conjunto das variáveis livres e o conjunto das variáveis ligadas.

   1. $xyxx$
   2. $\lambda x.\lambda x.x$
   3. $y\lambda x.$
   4. $x\lambda x.\lambda y.xy$
   5. $xy\lambda x.x\lambda.x.yxx$
   6. $\lambda x.xy(\lambda y.xy)$

2. Sendo M e N dois λ-termo s, $M \equiv_\alpha N$ se são o mesmo λ-termo ou se podem ser obtidos um do outro por mudança de variáveis ligadas em M ou em N. Por exemplo $(\lambda x.x)z \equiv_\alpha (\lambda y.y)z$ e $(\lambda x.x)z \not\equiv_\alpha (\lambda x.y)z$. Prova usando indução na estrutura dum λ-termo M que se $x \neq y$ e se $x \notin FV(L)$ então:
   $$M[N/x][L/y] \equiv_\alpha M[L/y][N[L/y]/x]$$

3. Para cada um dos seguintes λ-termo s reduz a forma normal β se possível:

   1. xy
   2. $(\lambda x.x)(yz)$
   3. $(\lambda x.xy)(\lambda z.z)$
   4. $(\lambda xy.xy)$
   5. $(\lambda x.x\lambda x.xy)(yy)$
   6. $(\lambda x.xx)(\lambda x.xx)$
   7. $(\lambda x.xxx)(\lambda x.xxx)$
   8. $(\lambda xy.y)((\lambda x.xx)(\lambda x.xx))b$
   9. $M = AAx$ onde $A = \lambda axz.z(aax)$

4. O que está errado nas seguintes reduções?

   1. $(\lambda xy.yx)y \to_\beta \lambda y.yy$
   2. $(\lambda xx.xx)y \to_\beta \lambda x.yy$
   3. $(\lambda x.xx)(\lambda y.ay)k \to_\beta (\lambda x.xx)ak \to_\beta akak$

5. Obtem reduções por ordem de aplicação e por ordem normal paracadaum dos seguintes λ-termo:
   (a) $(\lambda x.xxx)(\lambda yz.yz)$
   (b) $A(AA)$ onde $A = \lambda x.xKSK$, $K = \lambda xy.x$ e $S = \lambda xyz.xz(yz)$.

6. Considerando os numerais de Church, para $n \geq 0$, $c_n = \lambda fx.\overbrace{f(f(f\cdots(fx)\cdots))}^{n\ f\text{'s}}$. Mostra que

i. sendo $\texttt{suc} = \lambda nfx.f(nfx)$, tem-se que $\texttt{suc}\, c_n \equiv_\beta c_{n+1}$.

ii. sendo $A_+ = \lambda mn.m\, \texttt{suc}\, n$,tem-se que $A_+ c_m c_n \equiv_\beta c_{m+n}$.

iii. sendo $A_* = \lambda nmf.n(mf)$, tem-se que $A_* c_n c_m \equiv_\beta c_{n*m}$.

7. Sendo

$$\begin{aligned} \texttt{true} &= \lambda xy.x \\ \texttt{false} &= \lambda xy.y \\ \texttt{pair} &= \lambda xyf.fxy \\ \texttt{fst} &= \lambda p.p\,\texttt{true} \\ \texttt{snd} &= \lambda p.p\,\texttt{false} \end{aligned}$$

   Mostra que

$$\begin{aligned} \texttt{fst}(\texttt{pair}\, M\, N) &\equiv_\beta M \\ \texttt{snd}(\texttt{pair}\, M\, N) &\equiv_\beta N \end{aligned}$$

8. Considerando ainda $\texttt{if} = \lambda bxy.bxy$ define λ-termos que correspondam às as operações Booleanas $\neg$, $\wedge$ e $\vee$.

9. Utilizando um operador de ponto fixo, escreve um λ-termo que corresponde à definição recursiva da soma:

$$\begin{aligned} sum(n, 0) &= n \\ sum(n, m+1) &= sum(n, m) + 1 \end{aligned}$$

10. Encontra λ-termo s fechados F tal que:

    1. $Fx \;=_\beta\; xI$
    2. $Fxy \;=_\beta\; xIy$
    3. $Fx \;=_\beta\; F$
    4. $Fx \;=_\beta\; xF$

11. Considera o `The Programming Languages Zoo`: `http://plzoo.andrej.com`.

    (a) Analiza os termos definidos no ficheiro `example.lambda` e compara com os definidos nos exercícios anteriores.

    (b) Implementa os exemplos dados anteriormente (e nas aulas teóricas).

12. Considera as regras do sistema de tipos simples $\lambda \to$ dado:

$$\Gamma \vdash x : \alpha \qquad \text{se } x : \alpha \in \Gamma \qquad (Axioma)$$

$$\frac{\Gamma \vdash M : \alpha \to \beta \qquad \Gamma \vdash N : \alpha}{\Gamma \vdash (MN) : \beta}\ (APP)$$

$$\frac{\Gamma, x : \alpha \vdash M : \beta}{\Gamma \vdash (\lambda x.M) : (\alpha \to \beta)} \ (ABS)$$

Usando essas regras constrói as inferências para as seguintes asserções:

(a) $\vdash \lambda xyz.x(yz) : (a \to b) \to (c \to a) \to (c \to b)$

(b) $\vdash \lambda x.x : (a \to a)$

(c) $\vdash \lambda x.x : (a \to b) \to a \to b$

(d) $\vdash \lambda xy.x : (a \to b \to a)$

(e) $\vdash \lambda xyz.xzy : (a \to b \to c) \to b \to a \to c$

(f) $\vdash \lambda xy.xyy : (a \to a \to b) \to a \to b$

Conceitos de Linguagens Funcionais

9. Identifica algumas características das linguagens funcionais na linguagem Haskell, em particular:

    1. definição de funções (recursivas ou não)
    2. sistema de tipos
    3. encaixe de padrões
    4. mecanismos de passagem de parâmetros
    5. funções locais
    6. composição de funções

10. Resolve os problemas de implementação em Haskell das semânticas operacionais sugeridos na folha 2 (Exercício 12 e 19) e folha 4 (Exercício 7).

11. Considera o `The Programming Languages Zoo: http://plzoo.andrej.com`. Compara a linguagem `miniml` com a `minihaskel` e indica algumas diferenças.