Ficha 9

Semântica das Linguagens de Programação

2021/22

1. Usando a semântica de avaliação *call-by-name*, calcule o valor de cada uma das seguintes expressões:

   (a) $(\lambda u.\lambda l.\,\mathsf{listcase}\ l\ \mathsf{of}\ (\mathsf{nil}, \lambda h.\lambda t.\,u :: t))\,(7{+}2)\,(1 :: 2 :: \mathsf{nil})$

   (b) $\langle \mathsf{True}, \lambda x.\,x + x, 5 * 3\rangle.2\,((\lambda y.\,y + 1)\,3)$

   (c) $(\lambda f.\lambda u.\,\mathsf{sumcase}\ (f\,u)\ \mathsf{of}\ (\lambda x.\,x, \lambda x.\,x * 2, \lambda x.\,10))\,(\lambda x.\,@2\,x)\,((\lambda y.\,y * y)\,3)$

2. Considere a seguinte expressão B da linguagem de programação funcional estudada:

   $$(\lambda\langle x, y\rangle.\ \mathsf{if}\ x \leq y\ \mathsf{then}\ x\ \mathsf{else}\ y)\,\langle\,(\lambda x.x + 1)\,3,\ (\lambda x.x - 1)\,4\,\rangle$$

   (a) Construa uma árvore de prova do juízo $\vdash B : \mathsf{Int}$.

   (b) Calcule o valor de B, usando a semântica de avaliação *call-by-name* da linguagem (deve começar por traduzir o açúcar sintáctico utilizado).

3. Considere a seguinte expressão da linguagem de programação funcional estudada:

   $$\begin{array}{l}\mathsf{let}\ \ \mathsf{posroot} \equiv \lambda a.\,\mathsf{sumcase}\ a\ \mathsf{of}\ (\lambda x.\mathsf{False}, \lambda y.\,(y.1) > 0)\\ \mathsf{in}\ \ \ \ \mathsf{posroot}\,(@2\,\langle\,7 + 3,\ @1\langle\rangle,\ @1\langle\rangle\,\rangle)\end{array}$$

   (a) Calcule o seu valor, usando a semântica de avaliação *call-by-name*.

   (b) Construa uma árvore de prova do juízo

   $$a : \mathsf{Unit} + \mathsf{Int} \times \mathsf{A} \times \mathsf{A} \vdash \mathsf{sumcase}\ a\ \mathsf{of}\ (\lambda x.\mathsf{False}, \lambda y.\,(y.1) > 0) : \mathsf{Bool}$$

4. Usando a semântica de avaliação *call-by-name* da linguagem funcional que estudou, calcule o valor da seguinte expressão:

   $$\begin{array}{l}\mathsf{letrec}\ \ \mathsf{comp} \equiv \lambda l.\,\mathsf{listcase}\ l\ \mathsf{of}\ (0, \lambda h.\lambda t.1 + \mathsf{comp}\,t)\\ \mathsf{in}\ \ \ \ \ \mathsf{comp}\,((3 * 4) :: \mathsf{nil})\end{array}$$

   Construa também a árvore de tipificação desta expressão.

5. Considere a seguinte definição na linguagem de programação funcional estudada:

   $$\mathsf{letrec}\ \mathsf{map} \equiv \lambda f.\,\lambda l.\,\mathsf{listcase}\ l\ \mathsf{of}\ (\mathsf{nil}, \lambda h.\,\lambda t.\,f\,h :: \mathsf{map}\,f\,t)\ \ \ldots$$

   (a) Apresente a avaliação CBN da expressão $\mathsf{letrec}\ \mathsf{map} \equiv \ldots\ \mathsf{in}\ \mathsf{map}\,(\lambda x.\,2{*}x)\,(7 :: \mathsf{nil})$ até à sua forma canónica.

(b) No ambito da semântica CBN que estudou, a lista infinita de números naturais pode ser codificada por $\mathsf{letrec\ map} \equiv \ldots\ \mathsf{in}\ \ \mathsf{rec}\,(\lambda l.\, 0 :: \mathsf{map}\,(\lambda x.x+1)\,l)$

Apresente uma definição alternativa para a lista de naturais chamada natlist.

6. Considere a seguinte função que testa se duas listas são iguais

$$\mathsf{eqlist} : \mathbf{List\ Int} \to \mathbf{List\ Int} \to \mathbf{Bool}$$

e compare a avaliação CBN e CBV do seguinte programa

$$\begin{array}{l}
\mathsf{letrec\ eqlist} \equiv \lambda l_1.\, \lambda l_2. \\
\quad \mathsf{listcase}\ l_1\ \mathsf{of}\ (\\
\qquad \mathsf{listcase}\ l_2\ \mathsf{of}\ (\mathsf{True}, \lambda h_2.\lambda t_2.\, \mathsf{False}), \\
\qquad\ \lambda h_1.\lambda t_1.\, \mathsf{listcase}\ l_2\ \mathsf{of}\ (\mathsf{False}, \lambda h_2.\lambda t_2.\ h_1 = h_2\ \wedge\ \mathsf{eqlist}\, t_1\, t_2) \\
\quad) \\
\mathsf{in}\quad \mathsf{eqlist}\ (1 :: 2 :: 3 :: \mathsf{nil})\ (3 :: 2 :: 1 :: \mathsf{nil})
\end{array}$$

7. Construa uma extensão da linguagem de programação funcional, por forma a incluír um tipo de árvores binárias com informação nos nodos intermédios e nas folhas (*"full trees"*).

Defina a sintaxe abstracta das novas expressões e do novo tipo, as novas regras de inferência de tipo, as novas formas canónicas da linguagem e as novas regras de avaliação *"call-by-name"*.