Ficha 8

Semântica das Linguagens de Programação

2023/24

1. Usando a semântica de avaliação *call-by-value*, 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 A da linguagem de programação funcional estudada:

   $$\mathsf{let}\ f \equiv \lambda\langle x, y\rangle.\,x + y\ \mathsf{in}\ f\,\langle 5, 6\rangle$$

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

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

3. Considere a seguinte expressão FACT

   $$\mathsf{letrec\ fact} \equiv \lambda n.\,\mathsf{if}\ n = 0\ \mathsf{then}\ 1\ \mathsf{else}\ n * \mathsf{fact}\,(n-1)\ \mathsf{in\ fact}$$

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

   (b) Prove que a avaliação CBV de $\mathsf{letrec\ fact} \equiv \ldots\ \mathsf{in}\ (\mathsf{fact}\,1)$ produz o valor 1.

4. Considere a seguinte expressão APP

   $$\begin{array}{l}\mathsf{letrec\ append} \equiv \lambda x.\,\lambda y.\,\mathsf{listcase}\ x\ \mathsf{of}\ (y, \lambda h.\,\lambda t.\,h :: \mathsf{append}\,t\,y)\\ \mathsf{in\ append}\,(1 :: \mathsf{nil})\,(2 :: 7 :: 8 :: \mathsf{nil})\end{array}$$

   (a) Construa uma árvore de prova do juizo $\vdash \mathsf{APP} : \mathbf{List\ Int}$

   (b) Apresente a avaliação CBV da expressão APP até à sua forma canónica.

5. Tendo definido funções de ordem superior, podemos definir novas funções utilizando definições não recursivas. Por exemplo:

   $$\begin{array}{l}\mathsf{letrec\ foldr} \equiv \lambda f.\,\lambda z.\,\lambda l.\,\mathsf{listcase}\ l\ \mathsf{of}\ (z, \lambda h.\,\lambda t.\,f\,h\,(\mathsf{foldr}\,f\,z\,t))\ \mathsf{in}\\ \mathsf{let\ append} \equiv \lambda x.\,\lambda y.\,\mathsf{foldr}\,(\lambda h.\,\lambda r.\,h :: r)\,y\,x\ \mathsf{in}\\ \mathsf{append}\,(1 :: \mathsf{nil})\,(2 :: 7 :: 8 :: \mathsf{nil})\end{array}$$

   Apresente uma definição alternativa para a função fact.

6. Use a linguagem de programação funcional que estudou para definir as seguintes funções:

   (a) Valor absoluto de um inteiro.

   (b) Testar se uma lista de inteiros está ordenada.

7. Produza uma extensão da linguagem de programação funcional estudada, por forma a incluír um tipo de árvores binárias (com números inteiros nos nós intermédios e folhas vazias).

   (a) Defina sintaxe abstracta, regras de inferência de tipos, formas canónicas e regras de avaliação CBV apropriadas.

   (b) As árvores binárias podem ser vistas como estruturas de dados construidas à custa das alternativas e dos tuplos, vendo os seus construtores e eliminadores como açucar sintáctico.

   Apresente esta definição alternativa.