Cálculo de Programas

2.° ano
Lic. Ciências da Computação e Mestrado Integrado em Engenharia Informática
UNIVERSIDADE DO MINHO

2019/20 - Ficha nr.° 5

1. Considere o isomorfismo de ordem superior *flip* definido pela composição de isomorfismos seguinte:

$$\begin{array}{ccccccc} (C^B)^A & \cong & C^{A\times B} & \cong & C^{B\times A} & \cong & (C^A)^B \\ f & \mapsto & \widehat{f} & \mapsto & \widehat{f}.\mathsf{swap} & \mapsto & \overline{\widehat{f}\cdot\mathsf{swap}} = \mathit{flip}\ f \end{array}$$

   Mostre que

$$\mathit{flip}\ f\ x\ y = f\ y\ x \tag{F1}$$

   se verifica.

2. O diagrama seguinte representa o combinador *catamorfismo* (de naturais) que se começou a estudar na última aula teórica, onde a notação $(\!|\, g \,|\!)$ abrevia $cata\ g$ então usada:

$$\begin{tikzcd} \mathbb{N}_0 \arrow[d, "(\!|\, g \,|\!)"'] & 1+\mathbb{N}_0 \arrow[l, "\mathsf{in}"'] \arrow[d, "id+(\!|\, g \,|\!)"] \\ B & 1+B \arrow[l, "g"] \end{tikzcd} \quad \text{onde} \quad \left\{ \begin{array}{l} \mathsf{in} = [\mathsf{zero}\ , \mathsf{succ}] \\ \mathsf{zero}\ \_ = 0 \\ \mathsf{succ}\ n = n+1 \end{array} \right.$$

   Assumindo a seguinte propriedade universal desse combinador,

$$k = (\!|\, g \,|\!) \equiv k\cdot\mathsf{in} = g\cdot(id+k) \tag{F2}$$

   mostre que o combinador "ciclo-for"definido por

$$\mathsf{for}\ b\ i = (\!|\, [\underline{i}\ , b] \,|\!)$$

   se converte na seguinte versão "pointwise":

$$\begin{array}{l} \mathsf{for}\ b\ i\ 0 = i \\ \mathsf{for}\ b\ i\ (n+1) = b\ (\mathsf{for}\ b\ i\ n) \end{array}$$

3. Mostre, usando (F2), que o catamorfismo de naturais $(a+) = \mathsf{for}\ \mathsf{succ}\ a = (\!|\, [\underline{a}\ , \mathsf{succ}] \,|\!)$ se converte na definição[1]

$$\begin{array}{l} a+0 = a \\ a+(n+1) = 1+(a+n) \end{array}$$

[1] Repare que esta função mais não faz do que usar duas propriedades da adição de números – quais?

4. Considere agora a operação $a \ominus n$ de subtracção entre um inteiro a e um número natural n:

$$a \ominus 0 = a$$
$$a \ominus (n+1) = (a \ominus n) - 1$$

Encontre k e g tal que $(a\ominus) = \llparenthesis\, [\underline{k}\ , g]\, \rrparenthesis$. **Sugestão**: apoie a sua resolução num diagrama.

5. Codifique, em Haskell

$$\llparenthesis\, g\, \rrparenthesis = g \cdot (id + \llparenthesis\, g\, \rrparenthesis) \cdot \mathsf{out}$$
$$\mathsf{for}\ b\ i = \llparenthesis\, [\underline{i}\ , b]\, \rrparenthesis$$

em que out foi calculada numa ficha anterior. De seguida, codifique

$$f = \pi_2 \cdot aux\ \mathbf{where}\ aux = \mathsf{for}\ \langle \mathsf{succ} \cdot \pi_1, \mathsf{mul} \rangle\ (1,1)$$

e inspeccione o comportamento de f, bem como o de outras funções que definiu acima usando o combinador $\llparenthesis\, \_\, \rrparenthesis$

6. Deduza os isomorfismos $\mathsf{in_T}$ e $\mathsf{out_T}$ em

$$\begin{tikzcd} \mathsf{T} \arrow[rr, bend left, "\mathsf{out_T}"] \arrow[rr, phantom, "\cong"] & & \mathsf{F\ T} \arrow[ll, bend left, "\mathsf{in_T}"] \end{tikzcd} \tag{F3}$$

para cada tipo T (e respectivo $\mathsf{F\ T}$) cuja codificação em Haskell vem dada a seguir:[2]

(a) Listas finitas de elementos em A:

$\mathsf{T} = A^*$ $\qquad$ $\mathsf{F\ T} = 1 + A \times \mathsf{T}$

Haskell: $[\, a\,]$

(b) Árvores com informação de tipo A nos nós:

$\mathsf{T} = \mathsf{BTree}\ A$ $\qquad$ $\mathsf{F\ T} = 1 + A \times \mathsf{T}^2$

Haskell: $\mathbf{data}\ \mathsf{BTree}\ a = \mathit{Empty} \mid \mathit{Node}\ (a, (\mathsf{BTree}\ a, \mathsf{BTree}\ a))$

(c) Árvores com informação de tipo A nas folhas:

$\mathsf{T} = \mathsf{LTree}\ A$ $\qquad$ $\mathsf{F\ T} = A + \mathsf{T}^2$

Haskell: $\mathbf{data}\ \mathsf{LTree}\ a = \mathit{Leaf}\ a \mid \mathit{Fork}\ (\mathsf{LTree}\ a, \mathsf{LTree}\ a)$

(d) Árvores com informação nos nós e nas folhas:

$\mathsf{T} = \mathsf{FTree}\ B\ A$ $\qquad$ $\mathsf{F\ T} = B + A \times \mathsf{T}^2$

Haskell: $\mathbf{data}\ \mathsf{FTree}\ b\ a = \mathit{Unit}\ b \mid \mathit{Comp}\ (a, (\mathsf{FTree}\ b\ a, \mathsf{FTree}\ b\ a))$

(e) "Rose trees":

$\mathsf{T}\ X = \mathsf{Rose}\ X$ $\qquad$ $\mathsf{F\ T} = X \times \mathsf{T}^*$

Haskell: $\mathbf{data}\ \mathsf{Rose}\ a = \mathit{Ros}\ a\ [\mathsf{Rose}\ a]$

(f) Árvores de expressão:

$\mathsf{T} = \mathsf{Expr}\ V\ O$ $\qquad$ $\mathsf{F\ T} = V + O \times \mathsf{T}^*$

Haskell: $\mathbf{data}\ \mathsf{Expr}\ v\ o = \mathit{Var}\ v \mid \mathit{Op}\ (o, [\mathsf{Expr}\ v\ o])$

Codifique $\mathsf{in_T}$ e $\mathsf{out_T}$ na linguagem Haskell, para cada caso, e faça testes às igualdades $\mathsf{in_T} \cdot \mathsf{out_T} = id$ e $\mathsf{out_T} \cdot \mathsf{in_T} = id$.

[2] A^* é a notação adoptada para sequências finitas de As.