Cálculo de Programas

2.º Ano de MiEI+LCC (Universidade do Minho)
Ano Lectivo de 2017/18

Teste — 6 de Junho de 2018
16h00–18h00
Cantina de Gualtar

---

*Este teste consta de **8** questões que valem, cada uma, 2.5 valores. O tempo médio estimado para resolução de cada questão é de 15 min. Tome em consideração a informação que é dada em anexo.*

PROVA SEM CONSULTA (2h)

Questão 1 Considere o isomorfismo

$$\begin{tikzcd} (A+B)+C \arrow[rr, bend left, "\mathsf{coassocr}"] & \cong & A+(B+C) \arrow[ll, bend left, "\mathsf{coassocl}"] \end{tikzcd}$$

onde $\mathsf{coassocr} = [id + i_1\,, i_2 \cdot i_2]$. Calcule a sua conversa coassocl a partir da equação

$$\mathsf{coassocl} \cdot \mathsf{coassocr} \quad = \quad id$$

entregando-a em Haskell *pointwise*, sem recurso ao combinador *either*.

RESOLUÇÃO: Tem-se:

$$
\begin{array}{ll}
& \mathsf{coassocl} \cdot \mathsf{coassocr} = id \\
\equiv & \qquad \{ \ \mathsf{coassocr} = [id + i_1\,, i_2 \cdot i_2];\ \text{fusão-}+ \ \} \\
& [\mathsf{coassocl} \cdot (id + i_1)\,, \mathsf{coassocl} \cdot i_2 \cdot i_2] = id \\
\equiv & \qquad \{ \ \text{universal-}+\ ;\ \text{natural-}id \ \} \\
& \left\{ \begin{array}{l} \mathsf{coassocl} \cdot (id + i_1) = i_1 \\ \mathsf{coassocl} \cdot i_2 \cdot i_2 = i_2 \end{array} \right. \\
\equiv & \qquad \{ \ \text{definição-}+\ ;\ \text{natural-}id \ \} \\
& \left\{ \begin{array}{l} \mathsf{coassocl} \cdot [i_1\,, i_2 \cdot i_1] = i_1 \\ \mathsf{coassocl} \cdot i_2 \cdot i_2 = i_2 \end{array} \right. \\
\equiv & \qquad \{ \ \text{fusão-}+\ ;\ \text{universal-}+ \ \} \\
& \left\{ \begin{array}{l} \left\{ \begin{array}{l} \mathsf{coassocl} \cdot i_1 = i_1 \cdot i_1 \\ \mathsf{coassocl} \cdot i_2 \cdot i_1 = i_1 \cdot i_2 \end{array} \right. \\ \mathsf{coassocl} \cdot i_2 \cdot i_2 = i_2 \end{array} \right.
\end{array}
$$

Introduzindo variáveis obtém-se, em Haskell:

$$
\begin{array}{l}
\mathsf{coassocl}\ (\mathsf{Left}\ a) = \mathsf{Left}\ (\mathsf{Left}\ a) \\
\mathsf{coassocl}\ (\mathsf{Right}\ (\mathsf{Left}\ b)) = \mathsf{Left}\ (\mathsf{Right}\ b) \\
\mathsf{coassocl}\ (\mathsf{Right}\ (\mathsf{Right}\ c)) = \mathsf{Right}\ c
\end{array}
$$

□

Questão 2 Uma função $g : A \to B$ diz-se *injectiva* sempre que a igualdade $g\ x = g\ y$ é suficiente para se deduzir $x = y$. Por exemplo, $g\ x = 1 + x$ é injectiva pois $1 + x = 1 + y \;\Leftrightarrow\; x = y$. Um resultado da matemática diz-nos que, sempre que a igualdade *pointfree*

$$f \cdot g = id \tag{E1}$$

se verifica, então g é *injectiva* (e f é *sobrejectiva*). Assim, para mostrar que g (resp. f) é injectiva (resp. sobrejectiva) basta encontrar uma qualquer outra função f (resp. g) tal que (E1) se verifica.

Com base nas leis do cálculo de programas que estudou nesta disciplina mostre que as injecções i_1, i_2 são funções injectivas (como o próprio nome sugere) e que as projecções π_1, π_2 são sobrejectivas.

RESOLUÇÃO: Tem-se: $[g\,, h] \cdot i_1 = g$, logo basta $f = [id\,, h]$, para qualquer h, para $f \cdot i_1 = id$. O mesmo processo para i_2. Logo $f = [id\,, id]$ funciona para ambos os casos. Quanto às projecções, $\pi_1 \cdot \langle id, h\rangle = id$, logo π_1 é sobrejectiva. O mesmo esquema para π_2, logo $g = \langle id, id\rangle$ funciona para ambos os casos. $\square$

Questão 3 Identifique, apoiando a sua resolução num diagrama, qual é a definição da função polimórfica α cuja propriedade natural ("grátis") é

$$(f + g) \cdot \alpha \quad = \quad \alpha \cdot (h \times g + f)$$

RESOLUÇÃO: Basta substituir f, g e h por variáveis de tipo, no diagrama (desenhar!), obtendo-se $A + B \xleftarrow{\;\alpha\;} C \times B + A$.

Reparando-se que α não pode ser uma soma, ter-se-á $\alpha = [\beta, \gamma]$, onde $A + B \xleftarrow{\;\beta\;} C \times B$ e $A + B \xleftarrow{\;\gamma\;} A$.
Logo $\gamma = i_1$ e $\beta = i_2 \cdot \pi_2$. Assim, $\alpha = [i_2 \cdot \pi_2\,, i_1] = \mathsf{coswap} \cdot (\pi_2 + id)$. $\square$

Questão 4 A profundidade de uma árvore é o tamanho do maior caminho entre a raiz e uma sua folha. Defina como um catamorfismo a função $prof : \mathsf{LTree}\ a \to \mathbb{N}_0$ que calcula a profundidade de uma árvore binária de tipo LTree (cf. anexo) e converta $prof$ para código Haskell que não recorra ao combinador *catamorfismo*.

RESOLUÇÃO: Tem-se $prof = (\!|[\underline{0}, g]|\!)$ reflectindo o facto da profundidade de uma folha ser nula, pois "raiz" e folha coincidem. Quando a $g : \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$, teremos que ir buscar a maior profundidade e incrementá-la:

$$g\ (n, m) = max\ n\ m + 1.$$

Então:

$$\begin{aligned}
& prof = (\!|[\underline{0}, g]|\!) \\
\equiv \quad & \{\ \text{universal-cata}\ \} \\
& prof \cdot \mathsf{in} = [\underline{0}, g] \cdot (id + prof^2) \\
\equiv \quad & \{\ \mathsf{in} = [Leaf\,, Fork];\ \text{fusão-+}\ ;\ \text{absorção-+}\ \} \\
& [prof \cdot Leaf\,, prof \cdot Fork] = [\underline{0}, g \cdot prof^2]
\end{aligned}$$

$$\equiv \qquad \{ \text{ Eq-+ ; introdução de variáveis } \}$$

$$\left\{ \begin{array}{l} prof\ (Leaf\ \ a) = 0 \\ prof\ (Fork\ (t, t') = g\ (prof\ \ t, prof\ \ t')) \end{array} \right.$$

$$\equiv \qquad \{ \text{ definição de } g \ \}$$

$$\left\{ \begin{array}{l} prof\ (Leaf\ \ a) = 0 \\ prof\ (Fork\ (t, t') = max\ (prof\ \ t)\ (prof\ \ t') + 1 \end{array} \right.$$

□

Questão 5 No trabalho prático mostra-se como as "quadtrees" (tipo QTree em anexo, simplificado) podem ser usadas para processamento de imagens, particionadas recursivamente em quatro quadrantes.

A função $flipQTree : \mathsf{QTree}\ \ a \to \mathsf{QTree}\ \ a$, que inverte horizontalmente uma imagem, pode ser definida em Haskell como

$$\begin{array}{l} flipQTree\ (Cell\ \ a) = Cell\ \ a \\ flipQTree\ (Block\ ((nw, ne), (sw, se))) = \\ \quad Block\ ((flipQTree\ ne, flipQTree\ nw), (flipQTree\ se, flipQTree\ sw)) \end{array}$$

1. Calcule f e g tal que: $flipQTree = (\!|\mathsf{in} \cdot (f + g)|\!) = [\!((f + g) \cdot \mathsf{out})\!]$
2. Mostre que $flipQTree \cdot flipQTree = id$.

RESOLUÇÃO: Na primeira alínea, comecemos por desenvolver $flipQTree = (\!|\mathsf{in} \cdot (f + g)|\!)$:

$$flipQTree = (\!|\mathsf{in} \cdot (f + g)|\!)$$

$$\equiv \qquad \{ \text{ universal-cata; in} = [Cell\,, Block] \ \}$$

$$flipQTree \cdot [Cell\,, Block] = [Cell\,, Block] \cdot (f + g) \cdot (id + (flipQTree^2 \times flipQTree^2))$$

$$\equiv \qquad \{ \text{ fusão-+; absorção-+; Eq-+ } \}$$

$$\left\{ \begin{array}{l} flipQTree \cdot Cell = Cell.f \\ flipQTree \cdot Block = Block \cdot g \cdot (flipQTree^2 \times flipQTree^2) \end{array} \right.$$

$$\equiv \qquad \{ \text{ introdução de variáveis } \}$$

$$\left\{ \begin{array}{l} flipQTree\ (Cell\ \ a) = Cell\ (f\ \ a) \\ flipQTree\ (Block\ ((nw, ne), (sw, se))) = Block\ (g\ (flipQTree\ nw, flipQTree\ ne), (flipQTree\ sw, flipQTree\ se)) \end{array} \right.$$

Comparando com a definição dada, inferimos:

$$\left\{ \begin{array}{l} f\ \ a = a \\ g\ ((nw, ne), (sw, se)) = ((ne, nw), (se, sw)) \end{array} \right.$$

isto é

$$\left\{ \begin{array}{l} f = id \\ g = \mathsf{swap} \times \mathsf{swap} \end{array} \right.$$

para $\mathsf{swap}\ (a, b) = (b, a)$. Tem-se pois

$$flipQTree = (\!|\mathsf{in} \cdot (id + \mathsf{swap} \times \mathsf{swap})|\!)$$

Segunda parte:

$$
\begin{array}{ll}
& \mathit{flipQTree} \cdot \mathit{flipQTree} = \mathit{id} \\
\equiv & \quad \{ \text{ reflexão-cata } \} \\
& \mathit{flipQTree} \cdot (\!|\mathsf{in} \cdot (\mathit{id} + g)|\!) = (\!|\mathsf{in}|\!) \\
\Leftarrow & \quad \{ \text{ fusão-cata } \} \\
& \mathit{flipQTree} \cdot \mathsf{in} \cdot (\mathit{id} + g) = \mathsf{in} \cdot (\mathit{id} + \mathit{flipQTree}^2 \times \mathit{flipQTree}^2) \\
\equiv & \quad \{ \text{ cancelamento-cata } \} \\
& \mathsf{in} \cdot (\mathit{id} + g) \cdot (\mathit{id} + \mathit{flipQTree}^2 \times \mathit{flipQTree}^2) \cdot (\mathit{id} + g) = \mathsf{in} \cdot (\mathit{id} + \mathit{flipQTree}^2 \times \mathit{flipQTree}^2) \\
\equiv & \quad \{ \text{ natural-}(\mathit{id} + g) \text{ — ver (E3) abaixo } \} \\
& \mathsf{in} \cdot (\mathit{id} + \mathit{flipQTree}^2 \times \mathit{flipQTree}^2) \cdot (\mathit{id} + g) \cdot (\mathit{id} + g) = \mathsf{in} \cdot (\mathit{id} + \mathit{flipQTree}^2 \times \mathit{flipQTree}^2) \\
\equiv & \quad \{ \text{ isomorfismo } (\mathit{id} + g) \text{ — ver (E2) } \} \\
& \mathsf{in} \cdot (\mathit{id} + \mathit{flipQTree}^2 \times \mathit{flipQTree}^2) = \mathsf{in} \cdot (\mathit{id} + \mathit{flipQTree}^2 \times \mathit{flipQTree}^2) \\
\Box &
\end{array}
$$

Propriedades necessárias acima: isomorfismo

$$(\mathit{id} + g) \cdot (\mathit{id} + g) = \mathit{id} \tag{E2}$$

Propriedade grátis

$$(k + (m \times f) \times (j \times h)) \cdot (\mathit{id} + g) = (\mathit{id} + g) \cdot (k + (f \times m) \times (h \times j))$$

de onde se extrai o corolário:

$$(k + f^2 \times k^2) \cdot (\mathit{id} + g) = (\mathit{id} + g) \cdot (k + f^2 \times k^2) \tag{E3}$$

$\Box$

Questão 6 Recorra à lei de absorção-cata para demonstrar a propriedade

$$\mathit{count} \cdot (\mathsf{LTree}\ f) \quad = \quad \mathit{count} \tag{E4}$$

onde $\mathsf{LTree}\ A \xrightarrow{\mathit{count}} \mathbb{N}_0$ é o catamorfismo

$$\mathit{count} = (\!|[\underline{1}, \mathsf{add}]|\!)$$

onde $\mathsf{add}\ (a, b) = a + b$.

RESOLUÇÃO: Tem-se:

$$
\begin{array}{ll}
& \mathit{count} \cdot (\mathsf{LTree}\ f) \\
= & \quad \{ \ \mathit{count} = (\!|[\underline{1}, \mathsf{add}]|\!) \ \} \\
& (\!|[\underline{1}, \mathsf{add}]|\!) \cdot \mathsf{LTree}\ f \\
= & \quad \{ \text{ absorção-cata para } \mathsf{B}\ (f, g) = f + g^2 \ \} \\
& (\!|[\underline{1}, \mathsf{add}] \cdot (f + \mathit{id})|\!) \\
= & \quad \{ \text{ absorção-+ ; função constante } \underline{1}\text{; natural-}\mathit{id} \ \} \\
& (\!|[\underline{1}, \mathsf{add}]|\!) \\
= & \quad \{ \ \mathit{count} = (\!|[\underline{1}, \mathsf{add}]|\!) \ \} \\
& \mathit{count} \\
\Box &
\end{array}
$$

$\square$

Questão 7 Considere-se a função $h = \mathsf{for}\ \mathsf{swap}\ (0, 1)$. Sabendo que $\mathsf{for}\ g\ i = (\![\underline{i}, g]\!)$ e recorrendo à lei de recursividade mútua, deduza as definições *pointwise* das funções f e g tal que $h = \langle f, g \rangle$.

RESOLUÇÃO: Tem-se:

$$
\begin{array}{cl}
& h = \mathsf{for}\ \mathsf{swap}\ (0,1) \\
\equiv & \qquad \{\ \mathsf{for}\ g\ i = (\![\underline{i}, g]\!)\ \} \\
& h = (\![\underline{(0,1)}, \mathsf{swap}]\!) \\
\equiv & \qquad \{\ \text{split de funções constantes, cf. exercício de uma ficha; definição de swap}\ \} \\
& h = (\![\langle \underline{0}, \underline{1} \rangle, \langle \pi_2, \pi_1 \rangle]\!) \\
\equiv & \qquad \{\ h = \langle f, g \rangle\ ;\ \text{lei da troca}\ \} \\
& \langle f, g \rangle = (\!\langle [\underline{0}, \pi_2], [\underline{1}, \pi_1] \rangle\!) \\
\equiv & \qquad \{\ \text{recursividade mútua}\ \} \\
& \left\{ \begin{array}{l} f \cdot \mathsf{in} = [\underline{0}, \pi_2] \cdot (id + \langle f, g \rangle) \\ g \cdot \mathsf{in} = [\underline{1}, \pi_1] \cdot (id + \langle f, g \rangle) \end{array} \right. \\
\equiv & \qquad \{\ \mathsf{in} = [\underline{0}, \mathsf{succ}];\ \text{simplificação}\ \} \\
& \left\{ \begin{array}{l} f \cdot [\underline{0}, \mathsf{succ}] = [\underline{0}, g] \\ g \cdot [\underline{0}, \mathsf{succ}] = [\underline{1}, f] \end{array} \right. \\
\equiv & \qquad \{\ \text{fusão e Eq-+}\ \} \\
& \left\{ \begin{array}{l} \left\{ \begin{array}{l} f \cdot \underline{0} = 0 \\ f \cdot \mathsf{succ} = g \end{array} \right. \\ \left\{ \begin{array}{l} g \cdot \underline{0} = 1 \\ g \cdot \mathsf{succ} = f \end{array} \right. \end{array} \right.
\end{array}
$$

isto é, com variáveis:

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

$\square$

Questão 8 Pode provar-se que um tipo indutivo $\mathsf{T}\ X$ cuja base é o bifunctor

$$
\begin{array}{l}
\mathsf{B}\ (X, Y) = X + \mathsf{F}\ Y \\
\mathsf{B}\ (f, g) = f + \mathsf{F}\ g
\end{array}
$$

onde F é um outro qualquer functor, é um mónade, tendo-se:

$$\mu = (\![id, \mathsf{in} \cdot i_2]\!) \tag{E5}$$

$$u = \mathsf{in} \cdot i_1 \tag{E6}$$

Calcule a versão de μ para o caso $\mathsf{F}\ X = 1$, $\mathsf{F}\ f = id$. Admitindo $\mathsf{in} = [Ok\,, \underline{Ko}]$, defina o tipo T em sintaxe Haskell e escreva μ em Haskell *pointwise*.

RESOLUÇÃO: Tem-se:

$$
\begin{array}{ll}
& \mu = (\![[id\,, \mathsf{in} \cdot i_2]]\!) \\
\equiv & \quad \{ \text{ universal-cata } \} \\
& \mu \cdot \mathsf{in} = ([id\,, \mathsf{in} \cdot i_2] \cdot (id + \mathsf{F}\ \mu) \\
\equiv & \quad \{ \ \mathsf{F}\ f = id,\ id + id = id, \text{ natural-}id \ \} \\
& \mu \cdot \mathsf{in} = [id\,, \mathsf{in} \cdot i_2] \\
\equiv & \quad \{ \ \mathbf{in} = [Ok\,, \underline{Ko}] \text{ duas vezes, cancelamento via } i_2 \ \} \\
& \mu \cdot [Ok\,, \underline{Ko}] = [id\,, \underline{Ko}] \\
\equiv & \quad \{ \text{ Eq-+ } \} \\
& \left\{ \begin{array}{l} \mu \cdot Ok = id \\ \mu \cdot \underline{Ko} = \underline{Ko} \end{array} \right. \\
\equiv & \quad \{ \text{ variáveis } \} \\
& \left\{ \begin{array}{l} \mu\ (Ok\ x) = x \\ \mu\ Ko = Ko \end{array} \right.
\end{array}
$$

$\mathsf{T}\ A$ é isomorfo a $Maybe\ A$.
□

ANEXO — Catálogo de tipos de dados estudados na disciplina.

1. Números naturais:

$$\mathsf{T} = \mathbb{N}_0 \qquad \left\{ \begin{array}{l} \mathsf{F}\ X = 1 + X \\ \mathsf{F}\ f = id + f \end{array} \right. \qquad \mathsf{in} = [\underline{0}, \mathsf{succ}] \tag{E7}$$

Haskell: Int inclui $\mathbb{N}_0$.

2. Listas de elementos em A:

$$\mathsf{T} = A^* \qquad \left\{ \begin{array}{l} \mathsf{F}\ X = 1 + A \times X \\ \mathsf{F}\ f = id + id \times f \end{array} \right. \qquad \mathsf{in} = [\mathsf{nil}, \mathsf{cons}] \tag{E8}$$

Haskell: $[\,a\,]$.

3. Árvores com informação de tipo A nos nós:

$$\mathsf{T} = \mathsf{BTree}\ A \qquad \left\{ \begin{array}{l} \mathsf{F}\ X = 1 + A \times X^2 \\ \mathsf{F}\ f = id + id \times f^2 \end{array} \right. \qquad \mathsf{in} = [\underline{Empty}, Node] \tag{E9}$$

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

4. Árvores com informação de tipo A nas folhas:

$$\mathsf{T} = \mathsf{LTree}\ A \qquad \left\{ \begin{array}{l} \mathsf{F}\ X = A + X^2 \\ \mathsf{F}\ f = id + f^2 \end{array} \right. \qquad \mathsf{in} = [Leaf, Fork] \tag{E10}$$

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

5. Árvores quaternárias com informação de tipo A nas folhas:

$$\mathsf{T} = \mathsf{QTree}\ A \qquad \left\{ \begin{array}{l} \mathsf{F}\ X = A + X^2 \times X^2 \\ \mathsf{F}\ f = id + f^2 \times f^2 \end{array} \right. \qquad \mathsf{in} = [Cell, Block] \tag{E11}$$

Haskell: $\mathbf{data}\ \mathsf{QTree}\ a = Cell\ \ a \mid Block\ ((\mathsf{QTree}\ a, \mathsf{QTree}\ a), (\mathsf{QTree}\ a, \mathsf{QTree}\ a))$.

6. Árvores com informação nos nós e nas folhas:

$$\mathsf{T} = \mathsf{FTree}\ B\ A \qquad \left\{ \begin{array}{l} \mathsf{F}\ X = B + A \times X^2 \\ \mathsf{F}\ f = id + id \times f^2 \end{array} \right. \qquad \mathsf{in} = [Unit, Comp] \tag{E12}$$

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