Cálculo de Programas

2.º Ano de LCC+MiEI (Universidade do Minho)
Ano Lectivo de 2018/19

Exame de Recurso — 22 de Junho de 2019
09h00–11h00
Salas E2-1.01/1.03

- *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.*
- *Os alunos devem ler a prova antes de decidirem por que ordem responder às questões colocadas.*
- *Quem desejar responder à questão 7 no próprio enunciado deve preencher o número ao fundo da respectiva página antes de entregar.*

PROVA SEM CONSULTA (2h)

Questão 1 O combinador

$$\begin{aligned}&\mathsf{const} :: a \to b \to a\\&\mathsf{const}\ a\ b = a\end{aligned}$$

está disponível em Haskell para construir funções constantes, sendo habitual designarmos $\mathsf{const}\ k$ por $\underline{k}$, qualquer que seja k. Demonstre a igualdade

$$\underline{(b, a)} = \langle \underline{b}, \underline{a} \rangle$$

a partir de propriedades do cálculo de programas que constam do formulário.

Nr. do aluno: | | | | | |

Questão 2 Seja ×| uma função que se sabe satisfazer esta propriedade,

$$\langle\langle f, g\rangle, h\rangle = \times| \cdot \langle\langle f, h\rangle, g\rangle \tag{E1}$$

para quaisquer f, g e h. Calcule a definição de ×| e, a partir dela, derive a sua propriedade grátis. **Sugestão**: resolva a equação $\langle\langle f, h\rangle, g\rangle = id$ e use as soluções para obter ×| e o seu tipo mais geral.

Questão 3 Sabendo que as igualdade

$$(p? + p?) \cdot p? \ = \ (i_1 + i_2) \cdot p? \tag{E2}$$

se verifica, demonstre a seguinte propriedade combinador conhecido por condicional de McCarthy:

$$p \to (p \to a \ , \ b) \ , \ (p \to c \ , \ d) \ = \ p \to a \ , \ d \tag{E3}$$

Questão 4 A função seguinte, escrita em Haskell

$$
\begin{array}{l}
k : [A + B] \to [B] \\
k\ [\,] = [\,] \\
k\ (\mathsf{Left}\ a : x) = [\,] \\
k\ (\mathsf{Right}\ b : x) = b : k\ x
\end{array}
$$

é uma espécie de $takeWhile$: copia para a saída todos os elementos do tipo B até à ocorrência do primeiro elemento de tipo A, por exemplo: $k\ [\mathsf{Right}\ 3, \mathsf{Left}\ \texttt{'x'}, \mathsf{Right}\ 4] = [3]$.

Mostre que k é o seguinte catamorfismo de listas,

$$k = (\!|[\mathsf{nil}, \mathsf{in}] \cdot (id + \mathsf{distl})|\!) \tag{E4}$$

onde o isomorfismo $(A \times C) + (B \times C) \xleftarrow{\mathsf{distl}} (A + B) \times C$ tem como inverso $\mathsf{undistl} = [i_1 \times id, i_2 \times id]$. **Sugestão**: parta da propriedade universal-cata e anote a propriedade grátis de distl, pois vai ser-lhe útil no exercício.

Questão 5 O Arquivo Distrital de Braga é um dos mais importantes do país, tendo actualmente à sua guarda 611 fundos documentais com acesso on-line a partir de `https://bit.ly/2IVeoWJ`. O HTML dessa página foi gerado em Haskell (5 segs) a partir do respectivo catálogo, usando o hilomorfismo $untar$ que se segue e que produz uma *floresta* de expressões, a partir da qual a função $pict$ do módulo Exp[1] gera o HTML:

$$
\begin{aligned}
&untar = a \cdot (base\ id\ id\ untar) \cdot c\ \mathbf{where}\\
&\quad a = sort \cdot inExp^*\\
&\quad c = \mathsf{join} \cdot (id \times \mathsf{collect}) \cdot \mathsf{sep} \cdot o^*\\
&\quad o = (\pi_2 + \mathsf{assocr}) \cdot \mathsf{distl} \cdot (\mathsf{out}_{\mathsf{List}} \times id)\\
&\quad base\ a\ b\ y = (b + a \times y)^*
\end{aligned}
$$

Identifique, justificando, as funções α, β, γ e ω e os tipos X, Y, Z no seguinte diagrama do hilomorfismo $untar$:

$$
\begin{array}{ccccccccc}
(A^* \times B)^* & \xrightarrow{\omega} & X & \xrightarrow{\mathsf{sep}} & B^* \times (A \times (A^* \times B))^* & \xrightarrow{\beta} & Y & \xrightarrow{\mathsf{join}} & (B + A \times (A^* \times B)^*)^* \\
\downarrow{\scriptstyle untar} & & & & & & & & \downarrow{\scriptstyle \alpha} \\
(\mathsf{Exp}\ B\ A)^* & \xleftarrow[\gamma]{} & (\mathsf{Exp}\ B\ A)^* & \xleftarrow[inExp^*]{} & & & & & Z
\end{array}
$$

[1] Que faz parte do material da disciplina (`https://bit.ly/2FmSg6B`).

Nr. do aluno: | | | | | |

Questão 6 Considere a função $depth = (\!|[\mathsf{one}, \mathsf{succ} \cdot umax]|\!)$ que calcula a profundidade de árvores de tipo LTree, onde $umax\ (a, b) = max\ a\ b$.

Mostre, por absorção-cata, que a profundidade de uma árvore t não é alterada quando aplica uma função f a todas as suas folhas:

$$depth \cdot \mathsf{LTree}\ f = depth \tag{E5}$$

Questão 7 Considere a seguinte lei de recursividade mútua válida para hilomorfismos que partilham o mesmo anamorfismo:

$$\langle f, g\rangle = (\!|\langle h, k\rangle|\!) \cdot [\!(q)\!] \quad \equiv \quad \begin{cases} f = h \cdot \mathsf{F}\ \langle f, g\rangle \cdot q \\ g = k \cdot \mathsf{F}\ \langle f, g\rangle \cdot q \end{cases} \tag{E6}$$

- Derive a lei de recursividade mútua do formulário a partir de (E6). (Sugestão: procure a lei de reflexão-ana do formulário.)
- Apresente as justificações em falta no seguinte cálculo da lei (E6):

$$\begin{array}{ll} & \langle f, g\rangle = (\!|\langle h, k\rangle|\!) \cdot [\!(q)\!] \\ \equiv & \quad \{\ \ldots\ \} \\ & \langle f, g\rangle = \langle \alpha \cdot [\!(q)\!], \beta \cdot [\!(q)\!]\rangle \\ \equiv & \quad \{\ \ldots\ \} \\ & \begin{cases} f = \alpha \cdot [\!(q)\!] \\ g = \beta \cdot [\!(q)\!] \end{cases} \\ \equiv & \quad \{\ \ldots\ \} \\ & \begin{cases} f = h \cdot \mathsf{F}\ \langle \alpha, \beta\rangle \cdot \mathsf{out} \cdot [\!(q)\!] \\ g = k \cdot \mathsf{F}\ \langle \alpha, \beta\rangle \cdot \mathsf{out} \cdot [\!(q)\!] \end{cases} \\ \equiv & \quad \{\ \ldots\ \} \\ & \begin{cases} f = h \cdot \mathsf{F}\ \langle \alpha, \beta\rangle \cdot \mathsf{F}\ [\!(q)\!] \cdot q \\ g = k \cdot \mathsf{F}\ \langle \alpha, \beta\rangle \cdot \mathsf{F}\ [\!(q)\!] \cdot q \end{cases} \\ \equiv & \quad \{\ \ldots\ \} \end{array}$$

$$\left\{ \begin{array}{l} f = h \cdot \mathsf{F}\ \langle \alpha \cdot [\![q]\!], \beta \cdot [\![q]\!] \rangle \cdot q \\ g = k \cdot \mathsf{F}\ \langle \alpha \cdot [\![q]\!], \beta \cdot [\![q]\!] \rangle \cdot q \end{array} \right.$$

$\equiv$ { ..... }

$$\left\{ \begin{array}{l} f = h \cdot \mathsf{F}\ \langle f, g \rangle \cdot q \\ g = k \cdot \mathsf{F}\ \langle f, g \rangle \cdot q \end{array} \right.$$

□

Questão 8 Pode mostrar-se que a seguinte variante do tipo "rose tree"

$$\mathbf{data}\ \mathsf{Rose}\ a = L\ a \mid R\ [\mathsf{Rose}\ a]$$

que tem por base $\mathsf{B}\ (f, g) = f + g^*$, forma um mónade

$$X \xrightarrow{u} \mathsf{Rose}\ X \xleftarrow{\mu} \mathsf{Rose}\ (\mathsf{Rose}\ X)$$

onde

$$u = L \tag{E9}$$
$$\mu = (\!|[id, \mathsf{in} \cdot i_2]|\!) \tag{E10}$$

Construa as funções in / out para este tipo e desenhe o diagrama dos seus catamorfismos. Com base nesse diagrama,

- Converta para Haskell com variáveis a componente μ do referido mónade.
- Mostre que a lei monádica $\mu \cdot u = id$ se verifica.

Nr. do aluno: | | | | | |

ANEXO — Catálogo de alguns tipos de dados estudados na disciplina.

1. Números naturais:

$$\mathsf{T} = \mathbb{N}_0 \qquad \begin{cases} \mathsf{F}\ X = 1 + X \\ \mathsf{F}\ f = id + f \end{cases} \qquad \mathsf{in} = [\underline{0}, \mathsf{succ}] \tag{E11}$$

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

2. Listas de elementos em A:

$$\mathsf{T} = A^* \qquad \begin{cases} \mathsf{F}\ X = 1 + A \times X \\ \mathsf{F}\ f = id + id \times f \end{cases} \qquad \mathsf{in} = [\mathsf{nil}, \mathsf{cons}] \tag{E12}$$

Haskell: $[a]$.

3. Árvores com informação de tipo A nos nós:

$$\mathsf{T} = \mathsf{BTree}\ A \qquad \begin{cases} \mathsf{F}\ X = 1 + A \times X^2 \\ \mathsf{F}\ f = id + id \times f^2 \end{cases} \qquad \mathsf{in} = [\underline{Empty}, Node] \tag{E13}$$

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:

$$\mathsf{T} = \mathsf{LTree}\ A \qquad \begin{cases} \mathsf{F}\ X = A + X^2 \\ \mathsf{F}\ f = id + f^2 \end{cases} \qquad \mathsf{in} = [Leaf, Fork] \tag{E14}$$

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

5. Árvores com informação nos nós e nas folhas:

$$\mathsf{T} = \mathsf{FTree}\ B\ A \qquad \begin{cases} \mathsf{F}\ X = B + A \times X^2 \\ \mathsf{F}\ f = id + id \times f^2 \end{cases} \qquad \mathsf{in} = [Unit, Comp] \tag{E15}$$

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

6. Árvores de expressão:

$$\mathsf{T} = Expr\ V\ O \qquad \begin{cases} \mathsf{F}\ X = V + O \times X^* \\ \mathsf{F}\ f = id + id \times f^* \end{cases} \qquad \mathsf{in} = [Var, Op] \tag{E16}$$

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