Cálculo de Programas

2.º Ano de LCC+LEI (Universidade do Minho)
Ano Lectivo de 2014/15

Teste — 16 de Junho de 2015
20h00
Salas CPII 201, 202, 203

---

*Este teste consta de **10** questões que valem, cada uma, 2 valores. O tempo médio estimado para resolução de cada questão é de 15 min.*

PROVA SEM CONSULTA (2h30m)

Questão 1 Recorra às leis que conhece dos produtos, coprodutos e funções constantes para demonstrar a igualdade:

$$[\langle f, \underline{k}\rangle , \langle g, \underline{k}\rangle] \quad = \quad \langle [f \ , g], \underline{k}\rangle \tag{E1}$$

Questão 2 Suponha que apenas sabe a seguinte propriedade de uma dada função α,

$$\alpha \cdot \langle f, \langle g, h\rangle\rangle = \langle h, f\rangle \tag{E2}$$

válida para quaisquer f, g e h que a tipem correctamente.

Deduza a definição de α e, a partir do seu tipo mais geral, a respectiva propriedade *natural* (também chamada *grátis*) usando o habitual diagrama.

Questão 3 Considere a função $\alpha = [\overline{i_1} \ , \overline{i_2}]$.

- Calcule o tipo mais geral de α, representando-o através de um diagrama.
- $\widehat{\alpha}$ é um isomorfismo que conhece. Identifique-o através da inferência do respectivo tipo.

NB: $\overline{f}$ e $\widehat{f}$ abreviam $curry\ f$ e $uncurry\ f$, respectivamente.

Questão 4 Demonstre a seguinte propriedade do combinador condicional de McCarthy

$$(p \rightarrow g, h) \times f \quad = \quad p \cdot \pi_1 \rightarrow g \times f, h \times f \tag{E3}$$

sabendo que

$$\langle f, (p \rightarrow g, h)\rangle \quad = \quad p \rightarrow \langle f, g\rangle, \langle f, h\rangle \tag{E4}$$

se verifica.

Questão 5 Suponha dado o catamorfismo $f = (\![\mathsf{zero}, \mathit{least}]\!)$ definido sobre listas de números naturais ($\mathbb{N}_0$), para $\mathsf{zero}\ \_ = 0$ e $\mathit{least}\ (x, y) = \mathbf{if}\ x \leqslant y\ \mathbf{then}\ x\ \mathbf{else}\ y$. Demonstre que f é uma função constante $\underline{k}$, identificando o valor de k.

Questão 6 Pretendendo-se uma função que conte o número de folhas de uma LTree apareceram duas soluções: uma é o catamorfismo

$$count = (\![\mathsf{one}, \mathsf{add}]\!) \tag{E5}$$

onde $\mathsf{one} = \underline{1}$ e $\mathsf{add}\ (x, y) = x + y$; a outra,

$$count = \mathsf{length} \cdot tips \tag{E6}$$

baseia-se em duas funções que conhece das bibliotecas e trabalho prático da disciplina.

Recorrendo à lei de fusão dos catamorfismos, entre outras, mostre que as duas propostas (E5) e (E6) são a mesma função. **NB:** recorda-se que o functor de base do tipo LTree é $\mathsf{B}\ (f, g) = f + g \times g$; não precisa de provar a propriedade $\mathsf{length}\ (x +\!\!+\ y) = (\mathsf{length}\ x) + (\mathsf{length}\ y)$, se dela precisar.

Questão 7 Considere o tipo

$$\mathbf{data}\ \mathsf{TLTree}\ a = \mathsf{T}\ a \mid N\ (\mathsf{TLTree}\ a)\ (\mathsf{TLTree}\ a)\ (\mathsf{TLTree}\ a)$$

semelhante ao que foi usado no trabalho prático desta disciplina para manipular triângulos de Sierpinski.

- Defina $\mathsf{in}_{\mathsf{TLTree}}$ e $\mathsf{out}_{\mathsf{TLTree}}$ por forma a que o functor de base deste tipo seja $\mathsf{B}_{\mathsf{TLTree}}\ (f, g) = f + ((g \times g) \times g)$
- Defina o functor de tipo $\mathsf{TLTree}\,f$ [1] sob a forma de um anamorfismo e derive a sua implementação em Haskell com variáveis.

Questão 8 Recorde do trabalho prático a função $depth = (\![\mathsf{one}, \mathsf{succ} \cdot \widehat{max}]\!)$ que calcula a profundidade de árvores de tipo LTree.

Mostre que a profundidade de uma árvore t não é alterada quando aplica uma função f a todas as folhas de uma árvore t; isto é, use as leis dos catamorfismos para provar a propriedade:

$$depth \cdot \mathsf{LTree}\,f = depth \tag{E7}$$

Questão 9 O apuramento do valor médio das folhas (valores numéricos) de uma árvore binária t

$$avg\ t = \frac{\mathsf{sum}\ t}{count\ t}$$

de tipo LTree mostra a necessidade de duas travessias de t, uma feita por $\mathsf{sum} = (\![id, \mathsf{add}]\!)$ e a outra por $count = (\![\underline{1}, \mathsf{add}]\!)$, onde $\mathsf{add} = \widehat{+}$. A lei de "banana-split"

$$\langle (\!|i|\!), (\!|j|\!) \rangle = (\!|h|\!) \quad \Leftarrow \quad h = \langle i \cdot \mathsf{F}\,\pi_1, j \cdot \mathsf{F}\,\pi_2 \rangle \tag{E8}$$

permite fazer o mesmo apuramento com uma só travessia, obtendo-se:

[1] Isto é, $fmap\ f$ em Haskell, após instância deste tipo na classe Functor.

$$
\begin{array}{l}
avg\ t = n \mathbin{/} d\ \mathbf{where} \\
\quad (n, d) = aux\ t \\
\qquad aux\ (Leaf\ a) = (a, 1) \\
\qquad aux\ (Fork\ (x, y)) = (n1 + n2, d1 + d2)\ \mathbf{where} \\
\qquad\quad (n1, d1) = aux\ (Fork\ x) \\
\qquad\quad (n2, d2) = aux\ (Fork\ y)
\end{array}
$$

Complete as reticências nos seguintes passos que já se deram no processo de conversão do par ("split") de catamorfismos sum e $count$ num único catamorfismo, de que o algoritmo acima deriva:

$$
\begin{array}{ll}
& \langle \mathsf{sum}, count \rangle = (\!|h|\!) \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& \langle (\!|[id\ , \mathsf{add}]|\!), (\!|[\underline{1}\ , \mathsf{add}]|\!) \rangle = (\!|h|\!) \\
\Leftarrow & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& h = \langle [id\ , \mathsf{add}] \cdot \mathsf{F}\,\pi_1, [\underline{1}\ , \mathsf{add}] \cdot \mathsf{F}\,\pi_2 \rangle \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& h = \langle [id\ , \mathsf{add}] \cdot (id + \pi_1 \times \pi_1), [\underline{1}\ , \mathsf{add}] \cdot (id + \pi_2 \times \pi_2) \rangle \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
& h = [\langle id, \underline{1} \rangle\ , h2]\ \mathbf{where}\ h2\ ((n1, d1), (n2, d2)) = (n1 + n2, d1 + d2)
\end{array}
$$

Questão 10 O functor

$$
\begin{array}{l}
\mathsf{T}\ X = X \times X \\
\mathsf{T}\ f = f \times f
\end{array}
$$

oferece um mónade que nos permite trabalhar com pares encarados como vectores (y, x) a duas dimensões. Por exemplo, neste mónade a expressão

$$\mathbf{do}\ \{ x \leftarrow (2, 3); y \leftarrow (4, 5); \mathsf{return}\ (x + y) \}$$

dá $(6, 8)$ como resultado — a soma dos vectores $(2, 3)$ e $(4, 5)$. Definindo

$$\mu = \pi_1 \times \pi_2 \tag{E9}$$

$$u = \langle id, id \rangle \tag{E10}$$

para este functor T, demonstre que μ e u satisfazem as propriedades (58) e (57) do formulário, essenciais à evidência de que

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

é, de facto, um mónade.
