Cálculo de Programas

3.º Ano de LEI+MiEI (Universidade do Minho)
Ano Lectivo de 2022/23

Teste — 13 de Janeiro de 2023, 14h00–16h00
Salas E1-0.04 + E1-0.08

- *Esta prova consta de **8** questões que valem, cada uma, 2.5 valores. O tempo médio estimado para resolução de cada questão é de 15 min.*
- *Recomenda-se que os alunos leiam a prova antes de decidirem por que ordem querem responder às questões que são colocadas.*

PROVA PRESENCIAL INDIVIDUAL SEM CONSULTA (2h)

Questão 1 Recordando da biblioteca `Cp.hs` o isomorfismo $\mathsf{undistl} = [i_1 \times id, i_2 \times id]$, use diagramas para:

- descrever o tipo de $\mathsf{undistl}$;
- inferir a propriedade *natural* (ie. "grátis‘") da função distl que é inversa de $\mathsf{undistl}$. (**NB:** tem de formular essa propriedade mas não se pede para a provar analiticamente.)

RESOLUÇÃO: O tipo de $i_1 \times id$ é $A \times B \rightarrow (A + C) \times B$ e o de $i_2 \times id$ é $A' \times B' \rightarrow (C' + A') \times B'$. O "either" força as unificações $A = C'$, $C = A'$, $B = B'$, logo $[i_1 \times id, i_2 \times id]$ terá tipo

$$A \times B + C \times B \rightarrow (A + C) \times B$$

Logo $\mathsf{distl} : (A + C) \times B \rightarrow A \times B + C \times B$, o que conduz à propriedade natural

$$(f \times h + g \times h) \cdot \mathsf{distl} = \mathsf{distl} \cdot ((f + g) \times h)$$

□

Questão 2 Sabendo que a igualdade

$$(p? + p?) \cdot p? \;=\; (i_1 + i_2) \cdot p? \tag{E1}$$

se verifica, demonstre a seguinte propriedade do condicional de McCarthy:

$$p \rightarrow (p \rightarrow a\ ,\ b)\ ,\ (p \rightarrow c\ ,\ d) \;=\; p \rightarrow a\ ,\ d \tag{E2}$$

RESOLUÇÃO: Exercício das fichas práticas. □

Questão 3 Considere-se a função

$$h = \mathsf{for}\ loop\ (0,1) \tag{E3}$$

onde $loop\ (a,b) = (b, a+b)$. Sabendo que

$$\mathsf{for}\ g\ i = (\!|\, [\underline{i}, g]\, |\!) \tag{E4}$$

e recorrendo à lei de recursividade mútua, deduza as definições *pointwise* das funções f e g tal que $h = \langle f, g \rangle$.

RESOLUÇÃO: 'Calculemus' (preencher as justificações):

$$\langle f, g \rangle = \mathsf{for}\ loop\ (0,1)$$

$$\{ \quad \ldots\ldots\ldots \quad \}$$

$$\langle f, g \rangle = (\!|\, [\underline{0,1}, loop]\, |\!)$$

$$\{ \quad \ldots\ldots\ldots \quad \}$$

$$\langle f, g \rangle = (\!|\, [\langle \underline{0}, \underline{1} \rangle, \langle \pi_2, \mathsf{add} \rangle]\, |\!)$$

$$\{ \quad \ldots\ldots\ldots \quad \}$$

$$\langle f, g \rangle = (\!|\, \langle [\underline{0}, \pi_2], [\underline{1}, \mathsf{add}] \rangle\, |\!)$$

$$\{ \quad \ldots\ldots\ldots \quad \}$$

$$\left\{ \begin{array}{l} f \cdot \mathsf{in}_{\mathbb{N}_0} = [\underline{0}, \pi_2] \cdot (id + \langle f, g \rangle) \\ g \cdot \mathsf{in}_{\mathbb{N}_0} = [\underline{1}, \mathsf{add}] \cdot (id + \langle f, g \rangle) \end{array} \right.$$

$$\{ \quad \ldots\ldots\ldots \quad \}$$

$$\left\{ \begin{array}{l} f \cdot [\mathsf{zero}, \mathsf{succ}] = [\underline{0}, g] \\ g \cdot [\mathsf{zero}, \mathsf{succ}] = [\underline{1}, \mathsf{add} \cdot \langle f, g \rangle] \end{array} \right.$$

$$\{ \quad \ldots\ldots\ldots \quad \}$$

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

□

Questão 4 Mostre que o anamorfismo que calcula os sufixos de uma lista

$$suffixes = [\!(g)\!]\ \mathbf{where}\ g = (id + \langle \mathsf{cons}, \pi_2 \rangle) \cdot \mathsf{out}$$

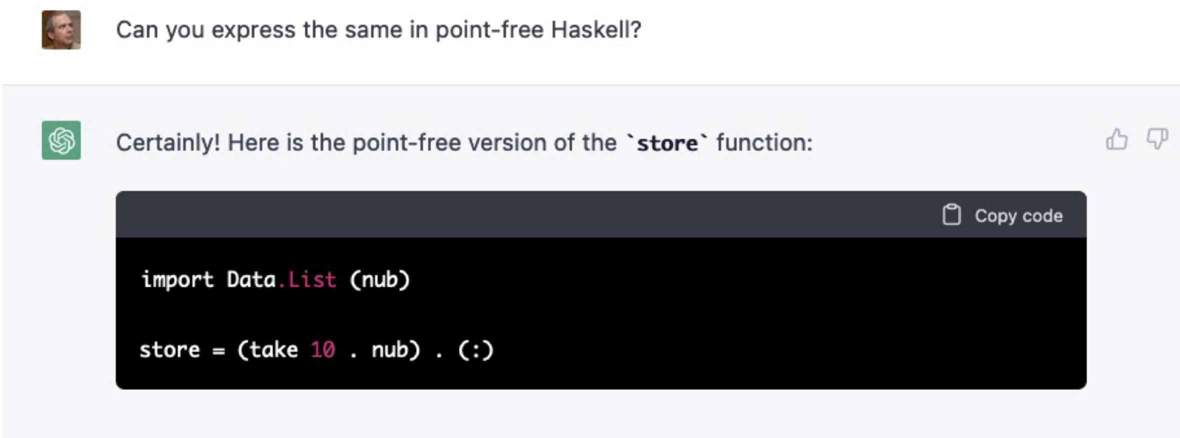
é a função:

$$\begin{array}{l} suffixes\ [\,] = [\,] \\ suffixes\ (h : t) = (h : t) : suffixes\ t \end{array}$$

Questão 5 Recorde o *problema do telemóvel antigo* que foi abordado na primeira ficha das aulas práticas desta disciplina:

*(...) For each **list of calls** stored in the mobile phone (eg. numbers dialed, SMS messages, lost calls), the **store** operation should work in a way such that **(a)** the more recently a **call** is made the more accessible it is; **(b)** no number appears twice in a list; **(c)** only the most recent 10 entries in each list are stored.*

Tendo-se pedido ao CHATGPT uma solução **pointfree** para estes requisitos, a resposta foi esta, para $store :: Eq\ a \Rightarrow a \to [a] \to [a]$:



Apesar de impressionante, a resposta tem um erro (apenas!). Identifique-o e diga como se pode corrigir.[1]

RESOLUÇÃO: Erro: $(:) : A \to {A^*}^{A^*}$ está curried, logo não compõe com $(\mathsf{take}\ 10 \cdot nub) : A^* \leftarrow A^*$. Ranking the hipóteses:

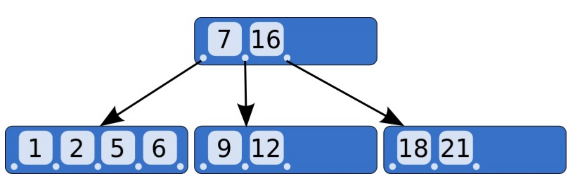
1. $\mathsf{exp}\ (\mathsf{take}\ 10 \cdot nub) \cdot (:)$ — ou *fmap* em lugar de exp (a que revela que entenderam o functor exponencial).
2. $\overline{\mathsf{take}\ 10 \cdot nub \cdot \mathsf{cons}}$
3. $store\ c = \mathsf{take}\ 10 \cdot nub \cdot (c:)$

$\square$

Questão 6 Uma "B-tree" é uma generalização das árvores binárias do módulo BTree a mais do que duas sub-árvores por nó:

$$\mathbf{data}\ \mathsf{B\_tree}\ a = Nil \mid Block\ \{ leftmost :: \mathsf{B\_tree}\ a, block :: [(a, \mathsf{B\_tree}\ a)] \}$$

Por exemplo, a B-tree[2]



é representada no tipo acima por:

$t = Block\ \{$
$\quad leftmost = Block\ \{$

[1] CHATGPT usa a função nub, para a qual dá a seguinte explicação: *"In Haskell, the nub function is used to remove duplicate elements from a list. It returns a new list containing only the unique elements from the original list, in the order in which they first appear. For example, $nub\ [1, 2, 3, 2, 1]$ would return $[1, 2, 3]$"*.

[2] Créditos: figura extraída de `https://en.wikipedia.org/wiki/B-tree`.

$\quad\quad \mathit{leftmost} = \mathit{Nil},$
$\quad\quad \mathit{block} = [(1, \mathit{Nil}), (2, \mathit{Nil}), (5, \mathit{Nil}), (6, \mathit{Nil})]\},$
$\quad \mathit{block} = [$
$\quad\quad (7, \mathit{Block}\ \{$
$\quad\quad\quad \mathit{leftmost} = \mathit{Nil},$
$\quad\quad\quad \mathit{block} = [(9, \mathit{Nil}), (12, \mathit{Nil})]\}),$
$\quad\quad (16, \mathit{Block}\ \{$
$\quad\quad\quad \mathit{leftmost} = \mathit{Nil},$
$\quad\quad\quad \mathit{block} = [(18, \mathit{Nil}), (21, \mathit{Nil})]\})$
$\quad\quad]\}$

Identifique, justificando, o functor de base

$$\left\{ \begin{array}{l} \mathsf{B}\ (X, Y) = \ldots \\ \mathsf{B}\ (f, g) = \ldots. \end{array} \right.$$

que capta o padrão de recursividade da declaração de B_tree dada acima, em Haskell, bem como o isomorfismo:

$$\mathsf{in} : \mathsf{B}\ (A, \mathsf{B\_tree}\ A) \to \mathsf{B\_tree}\ A.$$

Resolução:

$$\left\{ \begin{array}{l} \mathsf{B}\ (X, Y) = 1 + Y \times (X \times Y)^* \\ \mathsf{B}\ (f, g) = id + g \times (f \times g)^* \end{array} \right.$$

$$\mathit{iin} = [\underline{\mathit{Nil}}, \widehat{\mathit{Block}}]$$
$$\mathsf{out}\ \mathit{Nil} = i_1\ (); \mathsf{out}\ (\mathit{Block}\ l\ b) = i_2\ (l, b)$$

□

Questão 7 Considere a seguinte definição

$$\begin{array}{l} \mathit{dups} :: (\mathit{Eq}\ a) \Rightarrow [a] \to \mathbb{B} \\ \mathit{dups}\ [\,] = \text{FALSE} \\ \mathit{dups}\ (h : t) = h \in t \vee (\mathit{dups}\ t) \end{array}$$

de uma função que testa se uma lista contém elementos repetidos. Defina-a como um hilomorfismo identificando B e os genes f e g do diagrama seguinte:

$$\begin{tikzcd}
A^* \arrow[r, bend left, "g"] \arrow[d, "[\![g]\!]"'] \arrow[dd, bend right=60, "\mathit{dups}"'] & \mathsf{B}\ (\mathbb{B}, A^*) \arrow[d, "\mathsf{B}\ (id, [\![g]\!])"] \\
\mathbb{B}^* \arrow[r, bend left, "\mathsf{out}"] \arrow[d, "(\!| f |\!)"'] \arrow[r, phantom, "\cong"] & \mathsf{B}\ (\mathbb{B}, \mathbb{B}^*) \arrow[l, bend left, "\mathsf{in}"] \arrow[d, "\mathsf{B}\ (id, (\!| f |\!))"] \\
\mathbb{B} & \mathsf{B}\ (\mathbb{B}, \mathbb{B}) \arrow[l, bend left, "f"]
\end{tikzcd}$$

Resolução:

$$h = [\![f, g]\!]$$
$$f = [\mathit{false}, \widehat{\vee}]$$
$$g\ [\,] = i_1\ ()$$
$$g\ (h : t) = i_2\ (h \in t, t)$$

□

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 $\mathsf{B}\ (f, g) = f + \mathsf{map}\ g$, forma um mónade

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

onde

$$u = L \tag{E5}$$
$$\mu = (\!|\, [id, \mathsf{in} \cdot i_2]\, |\!) \tag{E6}$$

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.

RESOLUÇÃO: Tem-se

$$\mathsf{in} = [L, R]$$
$$\mathsf{out} \cdot L = i_1$$
$$\mathsf{out} \cdot R = i_2$$

e, como

$$\mathsf{F}\ g = \mathsf{B}\ (id, g) = id + \mathsf{map}\ g$$

o diagrama correspondente a $k \cdot \mathsf{in} = g \cdot (id + \mathsf{map}\ k)$ iff $k = (\!|\, g\, |\!)$. Logo:

$$\mu = (\!|\, [id, \mathsf{in} \cdot i_2]\, |\!)$$
$$\equiv \quad \{ \text{ universal-cata (??) } \}$$
$$\mu \cdot \mathsf{in} = [id, \mathsf{in} \cdot i_2] \cdot (id + \mathsf{map}\ \mu)$$
$$\equiv \quad \{ \ \mathsf{in} = [L, R] \text{ , logo } \mathsf{in} \cdot i_2 = R \text{ ; absorção-+ (??) } \}$$
$$\mu \cdot [L, R] = [id, R \cdot (\mathsf{map}\ \mu)]$$
$$\equiv \quad \{ \text{ fusão-+ (??) ; Eq-+ (??) } \}$$
$$\begin{cases} \mu \cdot L = id \\ \mu\ R = R \cdot (\mathsf{map}\ \mu) \end{cases}$$
$$\equiv \quad \{ \text{ introdução de variáveis } \}$$
$$\begin{cases} \mu\ (L\ a) = a \\ \mu\ (R\ x) = R\ ((\mathsf{map}\ \mu\ x)) \end{cases}$$

□

A cláusula $\mu \cdot L = id$ acima é a lei $\mu \cdot u = id$ que se pede para provar. □
