Cálculo de Programas

2.° ano das Licenciaturas em
Engenharia Informática (LEI) e Ciências da Computação (LCC)
da Universidade do Minho

2009/10 - Ficha nr.° 10

1. Defina como um catamorfismo a função seguinte, extraída do *Prelude* do Haskell,

$$\begin{array}{l} \mathsf{concat} :: [[a]] \to [a] \\ \mathsf{concat} = \mathit{foldr}\ (+\!\!+)\ [\,] \end{array}$$

e mostre que a propriedade

$$\mathsf{length} \cdot \mathsf{concat} = \mathsf{sum} \cdot \mathsf{map\ length} \tag{1}$$

se verifica, recorrendo às leis de *fusão*- e *absorção*-cata

$$f \cdot (\!|h|\!) = (\!|k|\!) \quad \Leftarrow \quad f \cdot h = k \cdot (\mathsf{F}\ f) \tag{2}$$

$$(\!|h|\!) \cdot \mathsf{T}\ f \quad = \quad (\!|h \cdot \mathsf{B}\ (f, id)|\!) \tag{3}$$

em que, para listas, se tem

$$\mathsf{B}\ (f, g) = id + f \times g \tag{4}$$

$$\mathsf{F}\ f = \mathsf{B}\ (id, f) \tag{5}$$

$$\mathsf{T}\ f = \mathsf{map}\ f \tag{6}$$

2. A função correspondente a concat para árvores é

$$\begin{array}{l} \mathsf{join} :: \mathsf{LTree}\ (\mathsf{LTree}\ a) \to \mathsf{LTree}\ a \\ \mathsf{join} = (\!|[id\ , \mathsf{Fork}]|\!) \end{array}$$

que junta uma *árvore de árvores* de tipo LTree numa só árvore. Conjecture a propriedade (1) para join e demonstre-a.

3. O desenho em baixo descreve aquela que é talvez a principal competência de um bom programador: a capacidade de dividir um problema complexo em partes e a de saber juntar as respectivas sub-soluções:

$$\begin{tikzcd}
\text{Problema} \arrow[r, "\mathit{divide}\ \text{(análise)}"] \arrow[d, "\mathit{solve}\ \text{(problema)}"'] & \text{Sub-problemas} \arrow[d, "\mathit{solve}\ \text{(sub-problemas)}"] \\
\text{Solução} & \text{Sub-soluções} \arrow[l, "\mathit{conquer}\ \text{(síntese)}"]
\end{tikzcd}$$

No Cálculo de Programas, o esquema desenhado acima é captado pelo diagrama de um *hilomorfismo*, cujo ingrediente principal é a fixação do padrão de organização das sub-soluções, captado pelo *functor* polinomial F:

$$\begin{aligned} solve &= [\![conquer, divide]\!] \\ &= (\!| conquer |\!) \cdot [\!(divide)\!] \end{aligned}$$

a que corresponde o diagrama

$$\begin{tikzcd} A \arrow[r, "divide"] \arrow[d, "solve"'] & \mathsf{F}A \arrow[d, "\mathsf{F}solve"] \\ B & \mathsf{F}B \arrow[l, "conquer"] \end{tikzcd}$$

No quadro que se segue mostra-se a classificação de algumas funções conhecidas de acordo com o respectivo F:

Classe	$\mathsf{F}\ \mathsf{X}$	**Serialização**	**Ordenação**	**Inversão**	**Factorial**	**Quadrado**	**Outros**
Listas	$1 + A \times X$		$iSort$	$invl$	fac	sq	$look$
BTree	$1 + A \times X^2$	*in/pré/pós*	$qSort$				$hanoi, traces$
LTree	$A + X^2$	*tips*	$mSort$	$invLTree$	$dfac$	dsq	fib

(a) Identifique a linha e coluna onde deve, do quadro acima, colocar o hilomorfismo de *bubble sorting*, identificando para ele os genes $divide$ e $conquer$:

$$\begin{aligned} & bSort :: \mathsf{Ord}\ a \Rightarrow [a] \to [a] \\ & bSort\ [\,] = [\,] \\ & bSort\ l\ = \mathbf{let}\ (x, m) = \mathsf{bubble}\ l \\ & \qquad\qquad \mathbf{in}\ \ x : bSort\ m \\ & \mathsf{bubble} :: \mathsf{Ord}\ a \Rightarrow [a] \to (a, [a]) \\ & \mathsf{bubble}\ [x] \quad = (x, [\,]) \\ & \mathsf{bubble}\ (x : l) = \mathbf{let}\ (y, m) = \mathsf{bubble}\ l \\ & \qquad\qquad\qquad \mathbf{in}\ \ \mathbf{if}\ x < y\ \mathbf{then}\ (x, y : m)\ \mathbf{else}\ (y, x : m) \end{aligned}$$

(b) Num dos outros algoritmos de ordenação do quadro acima (qual?), o passo de $divide$ é a função sep que se segue

$$\begin{aligned} & sep :: [a] \to ([a], [a]) \\ & sep = (\!| [\langle nil, nil \rangle\ , (\mathsf{cons} \times id) \cdot \mathsf{assocl} \cdot (id \times \mathsf{swap})] |\!) \\ & \quad \mathbf{where}\ nil\ \_ = [\,] \\ & \qquad \mathsf{cons}\ (a, b) = a : b \end{aligned}$$

cf. biblioteca `LTree.hs`. Mostre, usando a lei de recursividade múltipla para listas ($\mathsf{F}\ f = id + id \times f$) que sep se poderia ter escrito (ao nível *pointwise*) sob a forma de duas funções mutuamente recursivas $odds$ e $evens$,

$$\begin{aligned} & sep :: [a] \to ([a], [a]) \\ & sep = \langle odds, evens \rangle\ \mathbf{where} \\ & \quad odds\ [\,] = [\,] \\ & \quad odds\ (h : t) = h : (evens\ t) \\ & \quad evens\ [\,] = [\,] \\ & \quad evens\ (h : t) = odds\ t \end{aligned}$$

a primeira seleccionando os elementos que ocupam as posições ímpares e a segunda fazendo o mesmo para as pares.