Cálculo de Programas

2.° ano da Licenciatura em Engenharia Informática da
Universidade do Minho

2010/11 - Ficha nr.° 11

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

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

   e mostre que a propriedade

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

   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

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

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

3. O tipo $1 + A$ ("apontador" para A) foi o primeiro exemplo de mónade apresentado nesta disciplina, em que

   $$\mu \quad = \quad [i_1\ , id] \tag{7}$$
   $$u \quad = \quad i_2 \tag{8}$$

   Mostre que μ e u satisfazem as duas propriedades que caracterizam um mónade, neste caso:

   $$\mu \cdot u = \mu \cdot (id + u) = id \tag{9}$$
   $$\mu \cdot \mu = \mu \cdot (id + \mu) \tag{10}$$

4. O tipo

   $$\mathbf{data}\ \mathsf{Error}\ a = \mathsf{Error}\ \mathsf{String} \mid \mathsf{Ok}\ a$$

que vamos querer usar para gerir a emissão de mensagens de erro em funções parciais, mostra-se facilmente ser um functor definindo

$$\mathsf{Error}\ f = inE \cdot (id + f) \cdot outE \tag{11}$$

onde $inE = [\mathsf{Error}\ , \mathsf{Ok}]$ e

$$outE\ (\mathsf{Error}\ s) = i_1\ s$$
$$outE\ (\mathsf{Ok}\ a) = i_2\ a$$

(Verifique-o como trabalho de casa.) O tipo Error forma, ainda, um mónade desde que equipado com unidade $u = \mathsf{Ok}$ e multiplicação

$$\mu :: \mathsf{Error}\ (\mathsf{Error}\ a) \to \mathsf{Error}\ a$$
$$\mu\ (\mathsf{Error}\ s) = \mathsf{Error}\ s$$
$$\mu\ (\mathsf{Ok}\ a) = a$$

(a) Complete o cálculo que se segue mais abaixo da derivação do código acima a partir da sua definição *pointfree*

$$\mathsf{Error}\ a \xleftarrow{inE} \mathsf{S} + a \xleftarrow{[i_1\ ,id]} \mathsf{S} + (\mathsf{S} + a) \xleftarrow{outE} \mathsf{Error}\ (\mathsf{S} + a) \xleftarrow{(\mathsf{Error}\ outE)} \mathsf{Error}\ (\mathsf{Error}\ a)$$

(com a seta μ de $\mathsf{Error}\ (\mathsf{Error}\ a)$ para $\mathsf{Error}\ a$)

onde S abbrevia String:

$$\mu = inE \cdot [i_1\ , id] \cdot outE \cdot (\mathsf{Error}\ outE)$$

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

$$\mu = inE \cdot [i_1\ , id] \cdot outE \cdot (inE \cdot (id + outE) \cdot outE)$$

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

$\vdots$

$\equiv\quad \{$ ... alguns passos depois ... $\}$

$\vdots$

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

$$\begin{cases} \mu \cdot \mathsf{Error} = \mathsf{Error} \\ \mu \cdot \mathsf{Ok} = id \end{cases}$$

(b) Recorra à mesma definição *pointfree* de μ para calcular definições *pointwise* para a composição monádica $f \bullet g$ e a operação de *binding*, $x \ggg f$.

(c) Considere a expressão $((divE\ 1) \bullet headE)\ [\,]$ onde

$divE :: \mathsf{Double} \to \mathsf{Double} \to \mathsf{Error\ Double}$
$divE\ n\ 0 = \mathsf{Error}$ `"tentou-se dividir por 0!"`
$divE\ n\ m = \mathsf{Ok}\ (n\ /\ m)$
$headE :: [a] \to \mathsf{Error}\ a$
$headE\ [\,] = \mathsf{Error}$ `"lista vazia!"`
$headE\ l = \mathsf{Ok}\ (\mathsf{head}\ l)$

Que espera do cálculo dessa expressão? Um valor do tipo Double? Duas ou uma mensagem de erro? Se uma, qual delas? Justifique.