Cálculo de Programas

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

2010/11 - Ficha nr.° 12

1. Considere o hilomorfismo "quick fold" que se segue:

$$\begin{aligned}&qFold :: (\mathsf{Ord}\ a) \Rightarrow (a \to a \to a) \to a \to [a] \to a\\ &qFold\ f\ b = [\![[\underline{b}\ , g], qsep]\!]\\ &\quad \mathbf{where}\ g\ (a,(x,y)) = a\ `f`\ (x\ `f`\ y)\end{aligned}$$

   (a) Escreva a função auxiliar g em notação *pointfree*.

   (b) Qual o tipo da árvore intermédia do hilomorfismo dado? Faça um diagrama explicativo em que se explicitem todos os tipos envolvidos e, em particular, o da função $qsep$.

   (c) A definição de $qFold$ mostra como combinar "material genético" de diferentes hilomorfismos para produzir novos hilomorfismos. Qual é a função que conhece onde $qsep$ é usada como gene da correspondente parte "ana"?

2. Em contraste com a definição matemática de *mónade*, que oferece um functor F equipado com os operadores *unidade* e *multiplicação*,

$$A \xrightarrow{u} \mathsf{F}A \xleftarrow{\mu} \mathsf{F}^2 A \tag{1}$$

   a class Monad em Haskell exibe a unidade (renomeada para return) mas substitui a multiplicação pela operação de "binding":

$$\begin{aligned}&\mathbf{class}\ \mathsf{Monad}\ m\ \mathbf{where}\\ &\quad \mathsf{return} :: a \to m\ a\\ &\quad (\gg\!=) :: m\ a \to (a \to m\ b) \to m\ b\end{aligned}$$

   (a) Mostre que, tendo $\gg\!=$, pode sempre obter μ através de

$$\mu\ x \quad = \quad x \gg\!= id \tag{2}$$

   qualquer que seja o mónade subjacente.

   (b) Mostre ainda que, num mónade F,

$$id \bullet id \quad = \quad \mu \tag{3}$$

   se verifica e que $\mathsf{F}\ f$ se pode obter também a partir de $\gg\!=$:

$$\mathsf{F}\ f\ x \quad = \quad x \gg\!= (\mathsf{return} \cdot f) \tag{4}$$

3. Aplique as regras para "monadificação" de programas Haskell (escritos ao nível *pointwise*) apresentadas nas aulas teóricas aos combinadores

$$\begin{aligned}&foldr\ f\ b\ [\,] = b\\ &foldr\ f\ b\ (a : x) = f\ a\ (foldr\ f\ b\ x)\end{aligned}$$

e

$$\begin{aligned}&\mathsf{map}\ f\ [\,] = [\,]\\&\mathsf{map}\ f\ (a : x) = (f\ a) : \mathsf{map}\ f\ x\end{aligned}$$

por forma a obter as correspondentes versões monádicas, com tipos

$$mmap :: (\mathsf{Monad}\ m) \Rightarrow (a \to m\ b) \to [a] \to m\ [b]$$

e

$$mfoldr :: (\mathsf{Monad}\ m) \Rightarrow (a \to b \to m\ b) \to b \to [a] \to m\ b$$

respectivamente.

4. O functor de tipo LTree forma um mónade cuja unidade u é o construtor Leaf e cuja multiplicação μ é a função

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

Recorra às leis de cálculo de catamorfismos que conhece para mostrar que join satisfaz as duas leis (**Multiplicação** e **Unidade** no formulário) que definem um mónade.

5. O quadro seguinte tabula os mónades estudados nesta disciplina,

Mónade	**Tipo**	μ	u (return)	$x \gg\!= f$
Apontadores	$1 + \mathsf{A}$	$[i_1\ , id]$	i_2	
Mensagens de erro	Error A	(ver Ficha 11)	Ok	
Listas	A^*	concat	$singl$	
Árvores	LTree A	join (ver questão 4)	Leaf	
Acções sobre um estado	$(A \times S)^S$	exp ap	$\overline{id}$	$\widehat{f} \cdot x$

onde (recordar Ficha 5) exp f designa o mesmo que $\overline{(f \cdot \mathsf{ap})}$, isto é, $\mathsf{exp}\ f\ g = f \cdot g$. Calcule as definições da operação de *binding* que faltam na tabela.

6. Considere as seguintes acções que habitam o mónade de estado $\mathsf{St}\ A = (A \times S)^S$, para um dado espaço de estados S:

$$\begin{aligned}modify\ g &= \langle !, g\rangle &(5)\\query\ f &= \langle f, id\rangle &(6)\end{aligned}$$

A acção $query\ f$ interroga o estado aplicando-lhe a observação f e deixando-o inalterado; já a acção $modify\ g$ recorre a g para actualizar o valor corrente do estado, dando como resultado um mero "acknowledgement" da accção realizada.

Mostre que as igualdades

$$\begin{aligned}&\mathbf{do}\ \{modify\ id; query\ g\} = query\ g &(7)\\&\mathbf{do}\ \{modify\ f; modify\ g\} = modify\ (g \cdot f) &(8)\end{aligned}$$

se verificam.