

Cálculo de Programas

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

2009/10 - Ficha nr.º 8

1. A lei universal

$$\begin{array}{ccc}
 T & \xleftarrow{\mathbf{in}} & \mathbf{F} T \\
 \downarrow \langle g \rangle & & \downarrow \mathbf{F} \langle g \rangle \\
 B & \xleftarrow{g} & \mathbf{F} B
 \end{array}
 \quad k = \langle g \rangle \equiv k \cdot \mathbf{in} = g \cdot \mathbf{F} k
 \quad (1)$$

dá-nos meios para raciocinarmos sobre uma função k sempre que esta se pode escrever como um catamorfismo de gene g sobre o tipo indutivo T .

Nesta disciplina vimos vários exemplos de T , por exemplo os números naturais (\mathbb{N}_0), as listas ($[A]$) e dois tipos de árvores binárias,

data LTree $a = \text{Leaf } a \mid \text{Fork } (\text{LTree } a, \text{LTree } a)$

e

data BTree $a = \text{Empty} \mid \text{Node } (a, (\text{BTree } a, \text{BTree } a))$

A estes tipos podemos acrescentar outros como, por exemplo, o das listas não vazias

data Nelist $a = \text{Sing } a \mid \text{Add } (a, \text{Nelist } a)$

e o das chamadas “rose trees”:

data Rose $a = \text{Rose } a \text{ [Rose } a]$

Preencha o quadro seguinte, em que a coluna da esquerda identifica funções sobre o tipo da coluna T , funções essas que conhece ou cujo significado facilmente identifica:

k	g	$\mathbf{F} X$	$\mathbf{F} f$	T	\mathbf{in}	B
length		$1 + A \times X$		$[A]$	$[\text{nil}, \text{cons}]$	\mathbb{N}_0
length			$id + id \times f$	Nelist A		\mathbb{N}_0
count					$[\text{Leaf}, \text{Fork}]$	\mathbb{N}_0
listify	$[\text{singl}, +]$			LTree A		$[A]$
reverse					$[\text{nil}, \text{cons}]$	
sum		$1 + A \times X^2$				
sum					$[\text{Sing}, \text{Add}]$	
mirror	$\mathbf{in} \cdot (id + \text{swap})$				$[\text{Leaf}, \text{Fork}]$	
filter p			$id + id \times f$	$[A]$		$[A]$
$gmax$	$[id, max]$	$A + A \times X$				A
$gmax$	$[id, max]$				$[\text{Leaf}, \text{Fork}]$	A

2. Considere a função

$$\begin{aligned} \text{mirror}(\text{Leaf } a) &= \text{Leaf } a \\ \text{mirror}(\text{Fork } (x, y)) &= \text{Fork}(\text{mirror } y, \text{mirror } x) \end{aligned}$$

que “espelha” uma árvore binária de tipo $LTree$ e que, como se viu na questão anterior, é o catamorfismo

$$\text{mirror} = (\text{inLTree} \cdot (id + \text{swap})) \tag{2}$$

onde

$$\text{inLTree} = [\text{Leaf}, \text{Fork}] \tag{3}$$

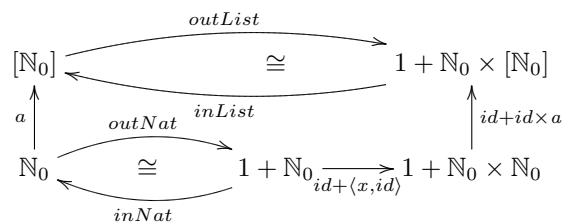
- (a) Comece por desenhar o digrama que representa o ctamorfismo mirror .
- (b) É fácil provar que mirror é um isomorfismo de árvores mostrando que a função é a sua própria inversa:

$$\text{mirror} \cdot \text{mirror} = id \tag{4}$$

Complete a seguinte demonstração desta propriedade:

$$\begin{aligned} & \text{mirror} \cdot \text{mirror} = id \\ \equiv & \quad \{ \dots \} \\ & \text{mirror} \cdot (\text{inLTree} \cdot (id + \text{swap})) = (\text{inLTree}) \\ \Leftarrow & \quad \{ \dots \} \\ & \dots \\ \dots & \quad \{ \dots \} \\ & \dots \end{aligned}$$

3. Considere o diagrama



que capta a seguinte propriedade da função a ,

$$a = \text{inList} \cdot (id + id \times a) \cdot (id + \langle x, id \rangle) \cdot \text{outNat}$$

para $\text{inNat} = [\underline{0}, \text{succ}]$ e $\text{inList} = [\text{nil}, \text{cons}]$, onde $\text{nil} = []$ e $\text{cons}(h, t) = h : t$.

Funções com esta estrutura dizem-se *anamorfismos* do seu tipo de saída (listas de naturais neste caso).

- (a) Explique por que é que a propriedade dada se pode escrever, alternativamente, sob a forma

$$a \cdot \text{inNat} = \text{inList} \cdot (id + \langle x, a \rangle) \tag{5}$$

- (b) Diga o que faz a função a para as situações seguintes: $x = \text{succ}$ e $x = \text{succ} \cdot (2^*)$.