Cálculo de Programas

2.° ano das Licenciaturas em
Engenharia Informática e Ciências da Computação
UNIVERSIDADE DO MINHO

2012/13 - Ficha nr.° 11

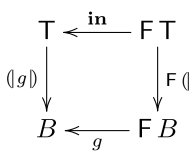
1. O algoritmo da divisão inteira,

$$
\begin{array}{l}
m \div n \\
\quad \mid m < n = 0 \\
\quad \mid \mathit{otherwise} = 1 + (m - n) \div n
\end{array}
$$

corresponde ao anamorfismo $(\div n) = [\![g\ n]\!]$ em que $g\ n = (<n) \rightarrow (i_1 \cdot !), (i_2 \cdot (-n))$. É de esperar que o algoritmo dado satisfaça a propriedade $(m * n) \div n = m$, isto é, $(\div n) \cdot (*n) = id$. Complete a prova seguinte, por fusão-ana, dessa propriedade:

$$
\begin{array}{ll}
 & (\div n) \cdot (*n) = id \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & [\![g\ n]\!] \cdot (*n) = [\![\mathsf{out}]\!] \\
\Leftarrow & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & (g\ n) \cdot (*n) = (id + (*n)) \cdot \mathsf{out} \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & ((<n) \cdot (*n)) \rightarrow (i_1 \cdot !), (i_2 \cdot (-n) \cdot (*n)) = (id + (*n)) \cdot \mathsf{out} \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & (= 0) \rightarrow (i_1 \cdot !), (i_2 \cdot (*n) \cdot \mathit{pred}) = (id + (*n)) \cdot \mathsf{out} \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & (id + (*n)) \cdot \mathsf{out} = (id + (*n)) \cdot \mathsf{out} \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
 & \mathit{true}
\end{array}
$$

2. Recorde o diagrama genérico de um catamorfismo de gene g sobre o tipo T e a sua propriedade universal:



$$k = (\!| g |\!) \quad \equiv \quad k \cdot \mathbf{in} = g \cdot \mathsf{F}\ k$$

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

$\mathbf{data}\ \mathsf{LTree}\ a = \mathsf{Leaf}\ a \mid \mathsf{Fork}\ (\mathsf{LTree}\ a, \mathsf{LTree}\ a)$
$\mathbf{data}\ \mathsf{BTree}\ a = \mathsf{Empty} \mid \mathsf{Node}\ (a, (\mathsf{BTree}\ a, \mathsf{BTree}\ a))$

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

$\mathbf{data}\ \mathsf{NEList}\ a = \mathsf{Sing}\ a \mid \mathsf{Add}\ (a, \mathsf{NEList}\ a)$

e o das chamadas "rose trees":

$\mathbf{data}\ \mathsf{Rose}\ a = \mathsf{Rose}\ a\ [\mathsf{Rose}\ a]$

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

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

$\mathsf{concat} :: [[a]] \to [a]$
$\mathsf{concat} = foldr\ (+\!\!+)\ [\,]$

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 $\mathsf{B}\ (f, g) = id + f \times g$, $\mathsf{F}\ f = \mathsf{B}\ (id, f)$ e $\mathsf{T}\ f = \mathsf{map}\ f$.

4. A função correspondente a concat para árvores é

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

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

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

T	F X	**Serialização**	**Ordenação**	**Inversão**	**Factorial**	**Quadrado**	**Outros**
$\mathbb{N}_0$	$1 + X$						$(n*), (\div n)$
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

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

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