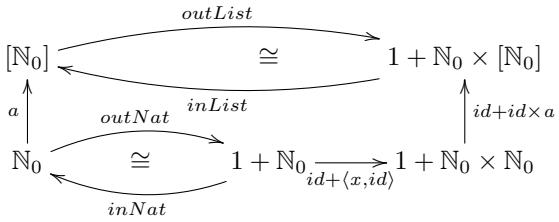
Cálculo de Programas

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

2012/13 - Ficha nr.° 10

1. Considere o diagrama



que capta a seguinte propriedade da função a,

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

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

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

$$a \cdot \mathsf{in} \quad = \quad inList \cdot (id + \langle x, a \rangle)$$

   (b) Diga o que faz a função a para $x = \mathsf{succ}$.

2. Considere os anamorfismos

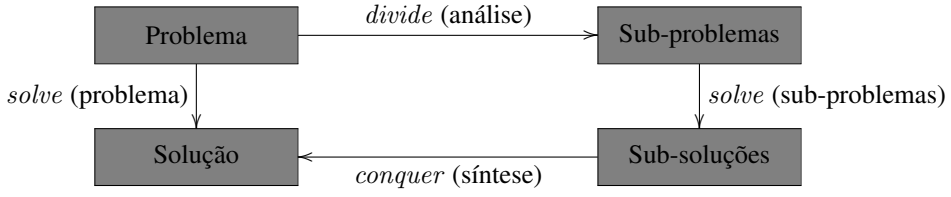
$$odds = [\![odd]\!] \ \mathbf{where}$$
$$odd = (id + \langle impar, id \rangle) \cdot outNat$$
$$impar\ n = 2 * n + 1$$

e

$$suffixes = [\![g]\!]$$
$$\mathbf{where}\ g\ [\,] = i_1\ [\,]$$
$$g\ (h : t) = i_2\ (h : t, t)$$

Desenhe os diagramas destes anamorfismos e derive as correspondentes versões em Haskell com váriáveis.

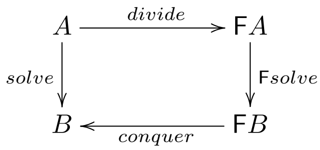
3. O desenho que se segue



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 para assim resolver o problema inicial.

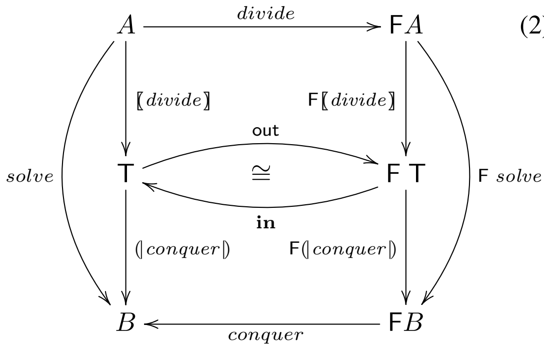
No Cálculo de Programas, o esquema desenhado acima é captado pelo conceito de *hilomorfismo*,

$$solve = conquer \cdot (\mathsf{F}\ solve) \cdot divide \qquad (1)$$



que normalmente se escreve $solve = \llbracket conquer, divide \rrbracket$ e se factoriza na composição do anamorfismo $[\!(divide)\!]$ com o catamorfismo $(\!| conquer |\!)$,

$$\llbracket conquer, divide \rrbracket = (\!| conquer |\!) \cdot [\!(divide)\!] \qquad (2)$$



mediados por uma estrutura de dados do tipo T associado ao functor F.[1]

(a) Apresente justificações para os passos seguintes da demonstração do princípio da *hilo-factorização*, isto é, da equivalência entre (1) e (2):

$$
\begin{array}{ll}
& solve = \llbracket conquer, divide \rrbracket \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& solve = (\!| conquer |\!) \cdot [\!(divide)\!] \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& solve = (conquer \cdot \mathsf{F}\ (\!| conquer |\!) \cdot \mathsf{out}) \cdot (\mathsf{in} \cdot \mathsf{F}\ [\!(divide)\!] \cdot divide) \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& solve = conquer \cdot \mathsf{F}\ (\!| conquer |\!) \cdot \mathsf{F}\ [\!(divide)\!] \cdot divide \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& solve = conquer \cdot \mathsf{F}\ ((\!| conquer |\!) \cdot [\!(divide)\!]) \cdot divide \\
\equiv & \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& solve = conquer \cdot \mathsf{F}\ solve \cdot divide
\end{array}
$$

(b) Desenhe o diagrama (2) correspondente ao hilomorfismo $f = \llbracket [g\ , h], p \rightarrow i_1, (i_2 \cdot k) \rrbracket$, identificando o tipo T. Mostre ainda que f satisfaz a propriedade seguinte:

$$f = p \rightarrow g, (h \cdot f \cdot k)$$

[1] Esta estrutura intermédia é designada normalmente por estrutura de dados **virtual** por "não ser ver" quando o algoritmo executa, ficando escondida no 'heap' do sistema de 'run-time' da linguagem recursiva que está a ser utilizada.