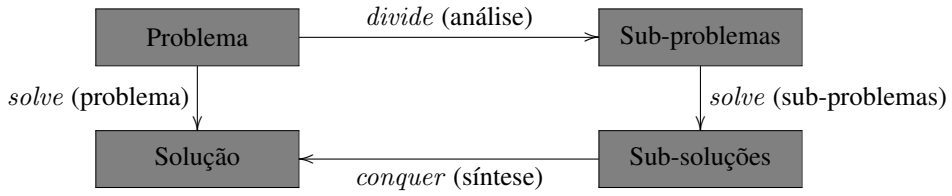
Cálculo de Programas

2.° ano
Lic. Ciências da Computação e Mestrado Integrado em Engenharia Informática
UNIVERSIDADE DO MINHO

2017/18 - Ficha nr.° 11

1. O desenho que se segue



descreve aquela que deve ser a principal competência de um programador: saber *dividir* um problema complexo em partes e juntar as respectivas sub-soluções para assim resolver o problema inicial.

No Cálculo de Programas, esse desenho é captado pelo conceito de *hilomorfismo*,

$$solve = conquer \cdot (\mathsf{F}\ solve) \cdot divide \qquad (F1)$$

$$\begin{tikzcd}
A \arrow[r, "divide"] \arrow[d, "solve"'] & \mathsf{F}\ A \arrow[d, "\mathsf{F}\ solve"] \\
B & \mathsf{F}\ B \arrow[l, "conquer"]
\end{tikzcd}$$

tendo-se[1]

$$solve = (\!|conquer|\!) \cdot [\!(divide)\!] \qquad (F2)$$

$$\begin{tikzcd}
A \arrow[rr, "divide"] \arrow[d, "{[\!(divide)\!]}"] \arrow[dd, bend right=60, "solve"'] & & \mathsf{F}\ A \arrow[d, "\mathsf{F}[\!(divide)\!]"'] \arrow[dd, bend left=60, "\mathsf{F}\ solve"] \\
\mathsf{T} \arrow[rr, bend left, "\mathsf{out}"] \arrow[rr, phantom, "\cong"] \arrow[d, "{(\!|conquer|\!)}"] & & \mathsf{F}\ \mathsf{T} \arrow[ll, bend left, "\mathsf{in}"] \arrow[d, "\mathsf{F}(\!|conquer|\!)"'] \\
B & & \mathsf{F}\ B \arrow[ll, "conquer"]
\end{tikzcd}$$

onde T é o tipo indutivo que tem F como base.[2] Apresente justificações para os passos seguintes da

[1] Normalmente escreve-se $solve = [\![conquer, divide]\!]$.

[2] 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.

demonstração do princípio da *hilo-factorização*, isto é, da equivalência entre (F1) e (F2):

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

2. Um ciclo-while é um hilomorfismo:

$$
\begin{array}{l}
\mathbf{while} :: (a \to \mathbb{B}) \to (a \to a) \to a \to a \\
\mathbf{while}\ p\ f = w\ \mathbf{where}\ w = [id\,, id] \cdot (w + id) \cdot (f + id) \cdot p?
\end{array}
$$

identificando o tipo intermédio T e as funções $divide$ e $conquer$. Mostre ainda que:

$$\mathbf{while}\ p\ f\ x = \mathbf{if}\ p\ x\ \mathbf{then}\ \mathbf{while}\ p\ f\ (f\ x)\ \mathbf{else}\ x$$

3. Considere o algoritmo de *insertion sort* tal como vem dado na biblioteca List.hs:

$$
\begin{array}{l}
iSort = (\!|[\mathsf{nil}\,, \mathsf{insert}]|\!)\ \mathbf{where} \\
\qquad\qquad \mathsf{insert}\ (x, [\,]) = [x] \\
\qquad\qquad \mathsf{insert}\ (x, a : l) \\
\qquad\qquad\quad \mid x < a = [x, a] +\!\!+ l \\
\qquad\qquad\quad \mid \mathsf{otherwise} = a : (\mathsf{insert}\ (x, l))
\end{array}
$$

A função auxiliar $\mathsf{insert} : A \times A^* \to A^*$ pode se construída como um hilomorfismo $\mathsf{insert} = [\![g, h]\!]$ onde

$$
\begin{array}{l}
h\ (x, [\,]) = i_1\ [x] \\
h\ (x, a : l) \\
\quad \mid x < a = i_1\ ([x, a] +\!\!+ l) \\
\quad \mid \mathsf{otherwise} = i_2\ (a, (x, l))
\end{array}
$$

$$
\begin{tikzcd}
A \times A^* \arrow[r, "h"] \arrow[d, "insert"'] & \ldots \arrow[d, "id+id\times\mathsf{insert}"] \\
A^* & \ldots \arrow[l, "g"]
\end{tikzcd}
$$

Identifique o gene g e complete as reticências do diagrama desse hilomorfisma acima esboçado, evidenciando o respectivo functor de base.

4. O algoritmo de "quick-sort" foi definido nas aulas teóricas como o hilomorfismo $qSort = (\!|inord|\!) \cdot [\!(qsep)\!]$ sobre árvores BTree, cujo catamorfismo recorre à função:

$$inord = [\mathsf{nil}\,, f]\ \mathbf{where}\ f\ (x, (l, r)) = l +\!\!+ [x] +\!\!+ r$$

Para um certo isomorfismo α, $h = (\!|inord \cdot \alpha|\!) \cdot [\!(qsep)\!]$ ordenará a lista de entrada por ordem inversa. (a) Identifique α, justificando informalmente. (b) Seja agora $h' = (\!|inord|\!) \cdot [\!(\alpha \cdot qsep)\!]$. Demonstre que $h' = h$.