Cálculo de Programas
Algebra of Programming

UNIVERSIDADE DO MINHO
Lic. em Engenharia Informática (3º ano)
Lic. Ciências da Computação (2º ano)

2023/24 - Ficha *(Exercise sheet)* nr. 7

O quadro abaixo representa a **propriedade universal** que define o combinador **catamorfismo**, com duas instâncias — números naturais $\mathbb{N}_0$ e listas finitas A^*, onde $\widehat{f}$ abrevia $\mathsf{uncurry}\, f$.

*The table below depicts the **universal property** that defines the **catamorphism** combinator, with two instances — natural numbers $\mathbb{N}_0$ and finite lists A^*, where $\widehat{f}$ abbreviates* $\mathsf{uncurry}\, f$*:*

Catamorfismo *(Catamorphism)*:

$$\begin{tikzcd}
\mathsf{T} \arrow[rr, bend left, "\mathsf{out}"] \arrow[rr, phantom, "\cong"] \arrow[d, "k"'] & & \mathsf{F}\ \mathsf{T} \arrow[ll, bend left, "\mathsf{in}"] \arrow[d, "\mathsf{F}\ k"] \\
B & & \mathsf{F}\ B \arrow[ll, bend left, "g"]
\end{tikzcd}$$

$$k = (\!|\, g \,|\!) \;\Leftrightarrow\; k \cdot \mathsf{in} = g \cdot \mathsf{F}\ k \qquad \text{(F1)}$$

Listas *(Lists)*:

$$\left\{\begin{array}{l}
\mathsf{T} = A^* \\
\left\{\begin{array}{l} \mathsf{in} = [\mathsf{nil}\ , \mathsf{cons}] \\ \mathsf{nil}\ _ = [\,] \\ \mathsf{cons}\ (h, t) = h : t \end{array}\right. \\
\left\{\begin{array}{l} \mathsf{F}\ X = 1 + A \times X \\ \mathsf{F}\ f = id + id \times f \end{array}\right.
\end{array}\right. \qquad \mathsf{foldr}\ f\ i = (\!|\, [\underline{i}\ , \widehat{f}] \,|\!)$$

Números naturais *(Natural numbers)*:

$$\left\{\begin{array}{l}
\mathsf{T} = \mathbb{N}_0 \\
\left\{\begin{array}{l} \mathsf{in}_{\mathbb{N}_0} = [\underline{0}\ , \mathsf{succ}] \\ \mathsf{succ}\ x\ n = n + 1 \end{array}\right. \\
\left\{\begin{array}{l} \mathsf{F}\ X = 1 + X \\ \mathsf{F}\ f = id + f \end{array}\right.
\end{array}\right. \qquad \mathsf{for}\ b\ i = (\!|\, [\underline{i}\ , b] \,|\!)$$

1. Fazendo $\mathsf{T} = \mathbb{N}_0$, codifique — recorrendo à biblioteca Cp.hs e à definição de out feita numa ficha anterior — o combinador:

   Taking $\mathsf{T} = \mathbb{N}_0$, encode — loading the Cp.hs library and using out *defined in a previous exercise sheet, the combinator:*

   $$(\!|\, g \,|\!) = g \cdot (id + (\!|\, g \,|\!)) \cdot \mathsf{out} \qquad \text{(F2)}$$

   De seguida implemente e teste a seguinte função:

   Then implement and test de following function:

   $$rep\ a = (\!|\, [\mathsf{nil}\ , (a:)] \,|\!) \qquad \text{(F3)}$$

O que faz ela? *What is its purpose?*

---

2. Na sequência da questão anterior, codifique *As follow up of the previous question, encode*

$$f = \pi_2 \cdot aux \ \mathbf{where} \ aux = \mathsf{for} \ \langle \mathsf{succ} \cdot \pi_1, \mathsf{mul} \rangle \ (1, 1) \qquad \text{(F4)}$$

e inspecione o seu comportamento. Que função f é essa? *and inspect its behavior. Which function is f?*

---

3. Identifique como catamorfismos de listas ($k = (\!| g |\!)$) as funções seguintes, indicando o gene g para cada caso (apoie a sua resolução com diagramas): *Identify as list catamorphisms ($k = (\!| g |\!)$) the following functions, indicating the corresponding 'gene' g for each case (support your answer with diagrams):*

   (a) k é a função que multiplica todos os elementos de uma lista. *(a) k is the function that multiplies all elements of a list.*

   (b) $k = \mathsf{reverse}$ *(b) $k = \mathsf{reverse}$*

   (c) $k = \mathsf{concat}$ *(c) $k = \mathsf{concat}$*

   (d) k é a função $\mathsf{map}\ f$, para um dado $f : A \to B$. *(d) k is the function $\mathsf{map}\ f$, for a given $f : A \to B$.*

   (e) k é a função que calcula o máximo de uma lista de números naturais (${\mathbb{N}_0}^*$). *(e) k is the function that calculates the maximum of a list of natural numbers (${\mathbb{N}_0}^*$).*

   (f) $k = \mathsf{filter}\ p$ onde: *(f) $k = \mathsf{filter}\ p$ where:*

$$\begin{array}{l} \mathsf{filter}\ p\ [\,] = [\,] \\ \mathsf{filter}\ p\ (h : t) = x \mathbin{+\!\!+} \mathsf{filter}\ p\ t \ \mathbf{where}\ x = \mathbf{if}\ (p\ h)\ \mathbf{then}\ [h]\ \mathbf{else}\ [\,] \end{array}$$

---

4. Apresente justificações para a seguinte dedução da lei de fusão-cata a partir de (F1) para o caso de ciclos-for ($\mathsf{T} = \mathbb{N}_0$): *Justify the following calculation of the cata-fusion law from (F1) valid for for-loops ($\mathsf{T} = \mathbb{N}_0$):*

$$\begin{array}{ll}
 & f \cdot (\!| g |\!) = (\!| h |\!) \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & f \cdot (\!| g |\!) \cdot \mathsf{in} = h \cdot \mathsf{F}\ (f \cdot (\!| g |\!)) \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id + f \cdot (\!| g |\!)) \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id \cdot id + f \cdot (\!| g |\!)) \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & f \cdot g \cdot (id + (\!| g |\!)) = h \cdot (id + f) \cdot (id + (\!| g |\!)) \\
\Leftarrow & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & f \cdot g = h \cdot (id + f)
\end{array}$$

Em suma:

Summing up:

$$f \cdot (\!| g |\!) = (\!| h |\!) \quad \Leftarrow \quad f \cdot g = h \cdot (id + f) \tag{F5}$$

---

5. A função seguinte, em Haskell

The following function, in Haskell

$$\begin{aligned} &sumprod\ a\ [\,] = 0 \\ &sumprod\ a\ (h : t) = a * h + sumprod\ a\ t \end{aligned}$$

é o catamorfismo de listas

is the list-catamorphism

$$sumprod\ a \quad = \quad (\!| [\mathsf{zero}\ , \mathsf{add} \cdot ((a*) \times id)] |\!) \tag{F6}$$

onde $\mathsf{zero} = \underline{0}$ e $\mathsf{add}\ (x, y) = x + y$. Como exemplo de aplicação da propriedade de **fusão-cata** para listas, demonstre a igualdade

where $\mathsf{zero} = \underline{0}$ *and* $\mathsf{add}\ (x, y) = x + y$. *As an example of application of **cata-fusion**, prove the equality*

$$sumprod\ a \quad = \quad (a*) \cdot \mathsf{sum} \tag{F7}$$

onde $\mathsf{sum} = (\!| [\mathsf{zero}\ , \mathsf{add}] |\!)$. **NB:** não ignore propriedades elementares da aritmética que lhe possam ser úteis.

where $\mathsf{sum} = (\!| [\mathsf{zero}\ , \mathsf{add}] |\!)$. ***NB:** take into account elementary arithmetic properties that may be useful.*

---

6. Sabendo que $\mathsf{for}\ f\ i = (\!| [\underline{i}\ , f] |\!)$, recorra à lei de fusão-cata para demonstrar a propriedade:

Knowing that $\mathsf{for}\ f\ i = (\!| [\underline{i}\ , f] |\!)$, *use the law of cata-fusion to prove the property:*

$$f \cdot (\mathsf{for}\ f\ i) = \mathsf{for}\ f\ (f\ i) \tag{F8}$$

---

7. Mostre que as funções

Show that functions

$$\begin{aligned} f &= \mathsf{for}\ id\ i \\ g &= \mathsf{for}\ \underline{i}\ i \end{aligned}$$

são a mesma função. (Qual?)

are the same function. (Which one?)

---

8. A função $\mathsf{foldr}\ \overline{\pi_2}\ i$ é necessariamente uma função constante. Qual? Justifique com o respectivo cálculo.

Function $\mathsf{foldr}\ \overline{\pi_2}\ i$ *is a constant function, for any* i *– which constant function? Write down your calculations.*

---

9. Qualquer função $k = \mathsf{for}\ f\ i$ pode ser codificada em sintaxe C escrevendo

Any function $k = \mathsf{for}\ f\ i$ *can be encoded in the syntax of C by writing:*

```
int k(int n) {
  int r=i;
  int j;
  for (j=1;j<n+1;j++) {r=f(r);}
  return r;
};
```

Escreva em sintaxe C as funções $(a*) = \mathsf{for}\ (a+)\ 0$ e outros catamorfismos de naturais de que se tenha falado nas aulas da UC.

Encode function $(a*) = \mathsf{for}\ (a+)\ 0$ *in C and other catamorphisms that have been discussed in the previous classes.*