Cálculo de Programas
Algebra of Programming

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

2024/25 - Ficha nr.° 3

1. Considere o diagrama

Consider the following diagram

$$
\begin{tikzcd}
(A \times B) \times C \arrow[rr, bend left, "\mathsf{assocr}"] & \cong & A \times (B \times C) \arrow[ll, bend left, "\mathsf{assocl}"]
\end{tikzcd}
$$

onde $\mathsf{assocl} = \langle id \times \pi_1, \pi_2 \cdot \pi_2 \rangle$. Apresente justificações para o cálculo que se segue em que se resolve em ordem a assocr a equação $\mathsf{assocl} \cdot \mathsf{assocr} = id$:

where $\mathsf{assocl} = \langle id \times \pi_1, \pi_2 \cdot \pi_2 \rangle$. *The reasoning below solves the equation* $\mathsf{assocl} \cdot \mathsf{assocr} = id$ *for variable* assocr. *Fill in justifications for each step in the reasoning:*

$$
\begin{array}{ll}
& \mathsf{assocl} \cdot \mathsf{assocr} = id \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \left\{ \begin{array}{l} (id \times \pi_1) \cdot \mathsf{assocr} = \pi_1 \\ \pi_2 \cdot \pi_2 \cdot \mathsf{assocr} = \pi_2 \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \left\{ \begin{array}{l} \left\{ \begin{array}{l} \pi_1 \cdot \mathsf{assocr} = \pi_1 \cdot \pi_1 \\ \pi_1 \cdot \pi_2 \cdot \mathsf{assocr} = \pi_2 \cdot \pi_1 \end{array} \right. \\ \pi_2 \cdot \pi_2 \cdot \mathsf{assocr} = \pi_2 \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \left\{ \begin{array}{l} \pi_1 \cdot \mathsf{assocr} = \pi_1 \cdot \pi_1 \\ \left\{ \begin{array}{l} \pi_1 \cdot \pi_2 \cdot \mathsf{assocr} = \pi_2 \cdot \pi_1 \\ \pi_2 \cdot \pi_2 \cdot \mathsf{assocr} = \pi_2 \end{array} \right. \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \left\{ \begin{array}{l} \pi_1 \cdot \mathsf{assocr} = \pi_1 \cdot \pi_1 \\ \pi_2 \cdot \mathsf{assocr} = \langle \pi_2 \cdot \pi_1, \pi_2 \rangle \end{array} \right. \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \mathsf{assocr} = \langle \pi_1 \cdot \pi_1, \pi_2 \times id \rangle
\end{array}
\tag{F1}
$$

2. (a) Codifique (F1) directamente em Haskell e verifique o comportamemto dessa função no GHCi; (b) De seguida, converta — por igualdade extensional — (F1) para notação Haskell *pointwise* que não recorra a nenhum combinador nem projecção e verifique no GHCi que as duas versões dão os mesmos resultados.

   (a) Encode (F1) directly in Haskell and check its behavior in GHCi; (b) Then convert — by extensional equality — (F1) to pointwise *Haskell code dispensing with combinators or projections, and let GHCi check that both versions give the same results.*

---

3. Recorde a propriedade universal do combinador $[f\,,g]$,

   Recall the universal property of the $[f\,,g]$ combinator,

$$k = [f\,,g] \quad\equiv\quad \left\{ \begin{array}{l} k \cdot i_1 = f \\ k \cdot i_2 = g \end{array} \right.$$

   Demonstre a igualdade

   Prove the equality

$$[\underline{k}\,,\underline{k}] = \underline{k} \tag{F2}$$

   recorrendo à propriedade universal acima e a uma lei que qualquer função constante $\underline{k}$ satisfaz. (Ver no formulário.)

   using the universal property given above and a law that any constant function $\underline{k}$ satisfies. (Check the reference sheet.)

---

4. Os isomorfismos

   Both isomorphisms of

$$\begin{tikzcd} A \times (B + C) \arrow[rr, bend left, "\mathsf{distr}"] \arrow[rr, phantom, "\cong"] & & A \times B + A \times C \arrow[ll, bend left, "\mathsf{undistr}"] \end{tikzcd}$$

   estudados na aula teórica estão codificados na biblioteca $Cp.hs$. Supondo $A = String$, $B = \mathbb{B}$ e $C = \mathbb{Z}$, (a) aplique no GHCi $\mathsf{undistr}$, alternativamente, aos pares $(\texttt{"CP"}, \textsc{True})$ ou $(\texttt{"LEI"}, 1)$; (b) verifique que $(\mathsf{distr} \cdot \mathsf{undistr})\ x = x$ para essas (e quaisquer outras) situações que possa testar.

   studied in the theory class are encoded in the library $Cp.hs$. Assuming that $A = String$, $B = \mathbb{B}$ and $C = \mathbb{Z}$, (a) apply in GHCi $\mathsf{undistr}$*, alternatively, to the pairs $(\texttt{"CP"}, \textsc{True})$ or $(\texttt{"LAW"}, 1)$; (b) check that $(\mathsf{distr} \cdot \mathsf{undistr})\ x = x$ for these (and any other) situations you can test.*

---

5. Recorde a função

   Recall

$$\alpha = [\langle \underline{\textsc{False}}, id\rangle\,, \langle \underline{\textsc{True}}, id\rangle]$$

   da ficha anterior. Mostre, usando a propriedade *universal*-$+$, que α se pode escrever em Haskell da forma seguinte:

   from the previous exercise sheet. Show, using the $+$-universal law*, that α can be written in pointwise Haskell as follows:*

$$\begin{array}{l} \alpha\ (i_1\ a) = (\textsc{False}, a) \\ \alpha\ (i_2\ a) = (\textsc{True}, a) \end{array}$$

   Codifique α e teste-a no GHCi, onde i_1 (resp. i_2) se escreve `Left` (resp. `Right`).

   Encode α and test it on $GHCi$.

---

6. Recorra às leis dos coprodutos para mostrar que a definição que conhece da função factorial,

Show by coproduct laws that the usual definition of the factorial function,

$$
\begin{array}{l}
fac\ 0 = 1 \\
fac\ (n+1) = (n+1) * fac\ n
\end{array}
$$

é equivalente à equação seguinte

is equivalent the following equation,

$$fac \cdot [\underline{0}, \mathsf{succ}] = [\underline{1}, \mathsf{mul} \cdot \langle \mathsf{succ}, fac \rangle]$$

onde

where

$$
\begin{array}{l}
\mathsf{succ}\ n = n+1 \\
\mathsf{mul}\ (a,b) = a * b
\end{array}
$$

---

7. A função $\mathsf{in} = [\underline{0}, \mathsf{succ}]$ da questão anterior exprime, para $\mathsf{succ}\ n = n+1$, a forma como os números naturais ($\mathbb{N}_0$) são gerados a partir do número 0, de acordo com o diagrama seguinte:

Function $\mathsf{in} = [\underline{0}, \mathsf{succ}]$ *in the previous question (where* $\mathsf{succ}\ n = n+1$*) expresses the way natural numbers (*$\mathbb{N}_0$*) are generated from the number* 0*, according to the diagram below:*

$$
\begin{tikzcd}
1 \arrow[r, "i_1"] \arrow[dr, "\underline{0}"'] & 1+\mathbb{N}_0 \arrow[d, "\mathsf{in}=[\underline{0},\mathsf{succ}]"] & \mathbb{N}_0 \arrow[l, "i_2"'] \arrow[dl, "\mathsf{succ}"] \\
& \mathbb{N}_0 &
\end{tikzcd}
\tag{F3}
$$

Sabendo que o tipo 1 coincide com o tipo $()$ em Haskell e é habitado por um único elemento, também designado por $()$, calcule a inversa de in,

Knowing that type 1 *matches type* $()$ *in Haskell and is inhabited by a single element, also denoted by* $()$*, find the inverse of* in*,*

$$
\left\{
\begin{array}{l}
\mathsf{out}\ 0 = i_1\ () \\
\mathsf{out}\ (n+1) = i_2\ n
\end{array}
\right.
\tag{F4}
$$

resolvendo em ordem a out a equação

by solving the equation

$$\mathsf{out} \cdot \mathsf{in} = id \tag{F5}$$

e introduzindo variáveis.

for out *and adding variables.*

---

8. Verifique no GHCi que a seguinte função

On GHCi check that function

$$fac = [\underline{1}, \mathsf{mul}] \cdot (id + \langle \mathsf{succ}, fac \rangle) \cdot \mathsf{out}$$

a que corresponde o diagrama

captured by diagram

$$
\begin{tikzcd}
\mathbb{N}_0 \arrow[r, "\mathsf{out}"] \arrow[d, "fac"'] & 1+\mathbb{N}_0 \arrow[d, "id+\langle \mathsf{succ}, fac \rangle"] \\
\mathbb{N}_0 & 1+\mathbb{N}_0 \times \mathbb{N}_0 \arrow[l, "{[\underline{1},\mathsf{mul}]}"]
\end{tikzcd}
$$

calcula o factorial da sua entrada, assumindo out (F4) e mul $(a, b) = a * b$ já definidas.

computes de factorial of its input, assuming out *(F4) and* mul $(a, b) = a * b$ *already defined.*

---

9. **Questão prática** — Este problema não irá ser abordado em sala de aula. Os alunos devem tentar resolvê-lo em casa e, querendo, publicarem a sua solução no canal **#geral** do Slack, com vista à sua discussão com colegas.
Os requisitos do problema são dados abaixo.
NB: usa-se a notação X^* para designar o tipo $[X]$ em Haskell.

***Open assignment** — This assignment will not be addressed in class. Students should try to solve it at home and, wishing so, publish their solution in the **#geral** Slack channel, so that it can be discussed among colleagues.*
The requirements of the problem are given below.
***NB**: notation X^* is used to denote the type $[X]$ in Haskell.*

***Problem requirements**:*

The automatic generation of bibliographies in the LaTeX text preparation system is based bibliographic databases from which the following information can be extracted:

$$Bib = (Key \times Aut^*)^*$$

It associates authors (Aut) to citation keys (Key).

Whenever LaTeX processes a text document, it compiles all occurrences of citation keys in an auxiliary file

$$Aux = (Pag \times Key^*)^*$$

associating pages (Pag) to the citation keys that occur in them.

*An **author index** is an appendix to a text (e.g. book) indicating, in alphabetical order, the names of authors mentioned and the ordered list of pages where their works are cited, for example:*

Arbib, M. A. – 10, 11
Bird, R. – 28
Horowitz, E. – 2, 3, 15, 16, 19
Hudak, P. – 11, 12, 29
Jones, C. B. – 3, 7, 28
Manes, E. G. – 10, 11
Sahni, S. – 2, 3, 15, 16, 19
Spivey, J.M. – 3, 7
Wadler, P. – 2, 3

The above structure can be represented by the type

$$Ind = (Aut \times Pag^*)^*$$

listing authors (Aut) and the respective pages where they are mentioned (Pag).

Write a Haskell function $mkInd : Bib \times Aux \to Ind$ that generates author indices (Ind) from Bib and Aux.

***Important**: Structure your solution across the $f \cdot g$, $\langle f, g \rangle$ and $f \times g$ combinators that can be found in library Cp.hs. Use **diagrams** to plan your proposed solution, which should avoid re-inventing functions over lists already available in the Haskell standard libraries.*
$\square$