

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.^o 3

1. Considere o diagrama

Consider the following diagram

$$(A \times B) \times C \xrightleftharpoons[\text{assoc}]{\cong} A \times (B \times C)$$

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

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

2. (a) Codifique (F1) directamente em Haskell e verifique o comportamento 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,

$$k = [f, g] \equiv \begin{cases} k \cdot i_1 = f \\ k \cdot i_2 = g \end{cases}$$

Demonstre a igualdade

Prove the equality

$$[k, k] = k \quad (\text{F2})$$

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

$$\begin{array}{ccc} A \times (B + C) & \xrightarrow{\cong} & A \times B + A \times C \\ \text{distr} \curvearrowleft & & \curvearrowright \text{undistr} \end{array}$$

estudados na aula teórica estão codificados na biblioteca *Cp.hs*. Supondo $A = \text{String}$, $B = \mathbb{B}$ e $C = \mathbb{Z}$, (a) aplique no GHCi `undistr`, alternativamente, aos pares ("CP", TRUE) ou ("LEI", 1); (b) verifique que (`distr · 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 $A = \text{String}$, $B = \mathbb{B}$ and $C = \mathbb{Z}$, (a) apply in GHCi `undistr`, alternatively, to the pairs ("CP", TRUE) or ("LAW", 1); (b) check that (`distr · undistr`) $x = x$ for these (and any other) situations you can test.

5. Recorde a função

Recall

$$\alpha = [\langle \text{FALSE}, \text{id} \rangle, \langle \text{TRUE}, \text{id} \rangle]$$

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:

$$\begin{aligned} \alpha(i_1 a) &= (\text{FALSE}, a) \\ \alpha(i_2 a) &= (\text{TRUE}, a) \end{aligned}$$

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,

$$\begin{aligned} \text{fac } 0 &= 1 \\ \text{fac } (n+1) &= (n+1) * \text{fac } n \end{aligned}$$

é equivalente à equação seguinte

is equivalent the following equation,

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

onde

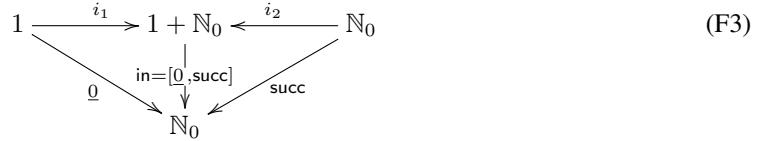
where

$$\text{succ } n = n + 1$$

$$\text{mul } (a, b) = a * b$$

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

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



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 ,

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

resolvendo em ordem a out a equação

by solving the equation

$$\text{out} \cdot \text{in} = id \quad (\text{F5})$$

e introduzindo variáveis.

for out and adding variables.

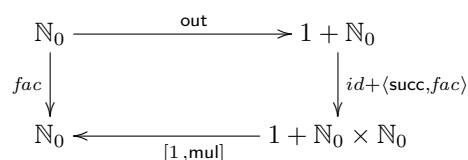
8. Verifique no GHCI que a seguinte função

On GHCI check that function

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

a que corresponde o diagrama

captured by diagram



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 on bibliographic databases from which the following information can be extracted:

$$\text{Bib} = (\text{Key} \times \text{Aut}^*)^*$$

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

$$\text{Aux} = (\text{Pag} \times \text{Key}^*)^*$$

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

$$\text{Ind} = (\text{Aut} \times \text{Pag}^*)^*$$

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

Write a Haskell function `mkInd : Bib × Aux → 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.

□