

Cálculo de Programas

Algebra of Programming

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

2024/25 - Ficha nr.º 5

1. No cálculo de programas, as definições condicionais do tipo *Conditional expressions of pattern*

$$h\ x = \text{if } p\ x \text{ then } f\ x \text{ else } g\ x \quad (\text{F1})$$

são escritas usando o combinador ternário *are expressed in the algebra of programming by the ternary combinator*

$$p \rightarrow f, g$$

conhecido pelo nome de *condicional de McCarthy*, cuja definição *known as the McCarthy conditional, whose definition*

$$p \rightarrow f, g = [f, g] \cdot p? \quad (\text{F2})$$

vem no formulário. Baseie-se em leis desse formulário para demonstrar a chamada 2ª-lei de fusão do condicional: *can be found in reference sheet. Use this reference sheet to prove the so-called 2nd fusion-law of conditionals:*

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

-
2. Numa máquina paralela pode fazer sentido, em (F1), não esperar por $p\ x$ para avaliar ou $f\ x$ ou $g\ x$, mas sim correr tudo em paralelo, *On a parallel machine it might make sense, concerning (F1), not to wait for $p\ x$ to evaluate either $f\ x$ or $g\ x$, but rather to run everything in parallel,*

$$\text{parallel } p\ f\ g = \langle \langle f, g \rangle, p \rangle$$

e depois fazer a escolha do resultado: *and then choose the outcome:*

$$\text{choose} = \pi_2 \rightarrow \pi_1 \cdot \pi_1, \pi_2 \cdot \pi_1$$

Mostre que, de facto: *Show that, indeed:*

$$\text{choose} \cdot \text{parallel } p\ f\ g = p \rightarrow f, g$$

3. Sabendo que as igualdades

Assuming

$$p \rightarrow k, k = k \quad (\text{F3})$$

$$(p? + p?) \cdot p? = (i_1 + i_2) \cdot p? \quad (\text{F4})$$

se verificam, demonstre as seguintes propriedades do mesmo combinador:

prove the following laws of the McCarthy conditional:

$$\langle (p \rightarrow f, h), (p \rightarrow g, i) \rangle = p \rightarrow \langle f, g \rangle, \langle h, i \rangle \quad (\text{F5})$$

$$\langle f, (p \rightarrow g, h) \rangle = p \rightarrow \langle f, g \rangle, \langle f, h \rangle \quad (\text{F6})$$

$$p \rightarrow (p \rightarrow a, b), (p \rightarrow c, d) = p \rightarrow a, d \quad (\text{F7})$$

4. Mostre que a propriedade de cancelamento da exponenciação

Show that the cancellation property

$$\text{ap} \cdot (\bar{f} \times \text{id}) = f \quad (\text{F8})$$

corresponde à definição

is nothing but the definition

$$\text{curry } f \ a \ b = f \ (a, b)$$

quando se escreve $\text{curry } f$ em lugar de $\bar{f}$.

once curry f is written instead of $\bar{f}$.

5. Mostre que a definição de uncurry se pode obter também de (F8) fazendo $f := \text{uncurry } g$, introduzindo variáveis e simplificando.

Show that the definition of uncurry can also be obtained from (F8) by instantiating $f := \text{uncurry } g$, introducing variables and simplifying.

6. Prove a igualdade

Prove the equality

$$\overline{f \cdot (g \times h)} = \overline{\text{ap} \cdot (\text{id} \times h) \cdot \bar{f} \cdot g} \quad (\text{F9})$$

usando as leis das exponenciais e dos produtos.

using the laws of products and exponentials.

7. É dada a definição

Let flip be defined by

$$\text{flip } f = \overline{\hat{f} \cdot \text{swap}} \quad (\text{F10})$$

de acordo com:

according to:

$$\begin{array}{ccccccc} (C^B)^A & \cong & C^{A \times B} & \cong & C^{B \times A} & \cong & (C^A)^B \\ f & \mapsto & \hat{f} & \mapsto & \hat{f} \cdot \text{swap} & \mapsto & \overline{\hat{f} \cdot \text{swap}} = \text{flip } f \end{array}$$

Mostre que flip é um isomorfismo por ser a sua própria inversa:

Show that it is an isomorphism because it is its own inverse:

$$\text{flip } (\text{flip } f) = f \quad (\text{F11})$$

Mostre ainda que:

Furthermore show:

$$\text{flip } f \ x \ y = f \ y \ x$$

8. Mostre que

Show that

$$junc \cdot unjunc = id \quad (F12)$$

$$unjunc \cdot junc = id \quad (F13)$$

se verificam, onde

hold for

$$A^{B+C} \begin{array}{c} \xrightarrow{unjunc} \\ \cong \\ \xleftarrow{junc} \end{array} A^B \times A^C \quad \left\{ \begin{array}{l} junc(f, g) = [f, g] \\ unjunc k = (k \cdot i_1, k \cdot i_2) \end{array} \right. \quad (F14)$$

9. Considere a seguinte sintaxe concreta em Haskell para um tipo que descreve pontos no espaço tridimensional:

Consider the following concrete syntax in Haskell for a type that describes 3D-points:

data Point a = Point { x :: a, y :: a, z :: a } deriving (Eq, Show)

Pelo GHCi apura-se:

GHCi tells:

Point :: a -> a -> a -> Point a

Raciocinando apenas em termos de tipos, conjecture a definição de in na seguinte conversão dessa sintaxe concreta para abstracta:

Reasoning only in terms of types, conjecture the definition of in in the following conversion from concrete to abstract syntax:

$$\text{Point } A \begin{array}{c} \xrightarrow{\text{out}=\langle x, y, z \rangle} \\ \cong \\ \xleftarrow{\text{in}=\dots} \end{array} (A \times A) \times A$$

10. **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. Dão-se a seguir os requisitos do problema.

Open assignment — This assignment will not be addressed in class. Students should try to solve it at home and, wishing so, publish their solutions in the **#geral** Slack channel, so as to trigger discussion among other colleagues. The requirements of the problem are given below.

Problem requirements: The solution given for a previous problem,

$$\text{store } c = \text{take } 10 \cdot \text{nub} \cdot (c:) \quad (F15)$$

calls the standard function

$$\text{nub} :: (Eq \ a) \Rightarrow [a] \rightarrow [a]$$

available from the Data.List library in Haskell.

After inspecting the standard implementation of this function, define f so that

$$\text{nub} = [\text{nil}, \text{cons}] \cdot f.$$

is an alternative to the standard definition, where $\text{nil} = []$ and $\text{cons } (h, t) = h : t$. Check that `store c (F15)` works properly once the standard `nub` is replaced by yours.

Important: Structure your solution across the $f \cdot g$, $\langle f, g \rangle$, $f \times g$, $[f, g]$ and $f + g$ combinators available from library `Cp.hs`. Use **diagrams** to plan your solution, in which you should avoid re-inventing functions over lists already available in the Haskell standard libraries.

□