

Cálculo de Programas

2.º ano

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

2020/21 - Ficha nr.º 1

1. A composição de funções define-se, em Haskell, tal como na matemática:

$$(f \cdot g) x = f (g x)$$

(a) Calcule $(f \cdot g) x$ para os casos seguintes:

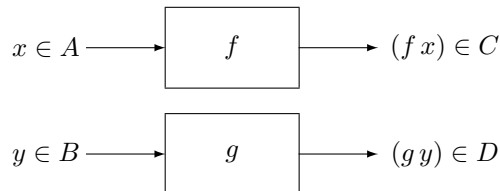
$$\begin{cases} f x = 2 * x \\ g x = x + 1 \end{cases} \quad \begin{cases} f = \text{succ} \\ g x = 2 * x \end{cases} \quad \begin{cases} f = \text{succ} \\ g = \text{length} \end{cases} \quad \begin{cases} g(x, y) = x + y \\ f = \text{succ} \cdot (2*) \end{cases}$$

Anime as composições funcionais acima num interpretador de Haskell.

(b) Mostre que $(f \cdot g) \cdot h = f \cdot (g \cdot h)$, quaisquer que sejam f, g e h .

(c) A função $id :: a \rightarrow a$ é tal que $id x = x$. Mostre que $f \cdot id = id \cdot f = f$ qualquer que seja f .

2. O diagrama de blocos



descreve o combinador funcional *produto*

$$f \times g = \langle f \cdot \pi_1, g \cdot \pi_2 \rangle \tag{F1}$$

que capta a aplicação *paralela e independente* de duas funções $A \xrightarrow{f} C$ e $B \xrightarrow{g} D$:

$$\begin{array}{ccc} A & B & A \times B \\ f \downarrow & g \downarrow & \downarrow f \times g \\ C & D & C \times D \end{array}$$

(a) Mostre que $(f \times g) (x, y) = (f x, g y)$.

(b) Mostre ainda que

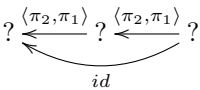
$$\pi_1 \cdot (f \times g) = f \cdot \pi_1 \tag{F2}$$

$$\pi_2 \cdot (f \times g) = g \cdot \pi_2 \tag{F3}$$

$$id \times id = id \tag{F4}$$

$$(f \times g) \cdot (h \times k) = f \cdot h \times g \cdot k \tag{F5}$$

Desenhe os diagramas destas igualdades e anime-as em Haskell, para f, g, h e k à sua escolha.

3. Preencha da forma mais genérica possível os “?” do diagrama 

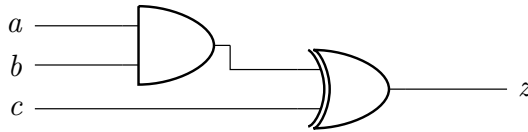
4. Considere as funções seguintes:

$$f = \langle \pi_1 \cdot \pi_1, \pi_2 \times id \rangle$$

$$g = \langle id \times \pi_1, \pi_2 \cdot \pi_2 \rangle$$

Identifique os tipos de f e g . Acompanhe a sua resolução com a construção dos respectivos diagramas.

5. Considere o circuito booleano



que calcula a função $f((a, b), c) = (a \wedge b) \oplus c$, onde \oplus é a operação “exclusive-or”.

- Escreva uma definição dessa função $(\mathbb{B} \times \mathbb{B}) \times \mathbb{B} \xrightarrow{f} \mathbb{B}$ que não recorra às variáveis a , b ou c ¹ e desenhe o respectivo diagrama.
 - Qual é o tipo da função $g = \langle \pi_1, f \rangle$?
6. Suponha que tem a relação $db_1 \in (Dat \times Jog^*)^*$ de todos os jogos que se efectuaram numa dada competição, organizados por data. Seja ainda $db_2 \in (Jog \times Atl^*)^*$ a indicação, para cada jogo, dos atletas que nele participaram.

Um comentador desportivo pede-lhe que derive de db_1 e de db_2 a relação, ordenada por nome, das datas em que cada atleta jogou, datas essas também ordenadas:

$$f : (Dat \times Jog^*)^* \rightarrow (Jog \times Atl^*)^* \rightarrow (Atl \times Dat^*)^*$$

$$f db_1 db_2 = \dots$$

Mostre que f pode ser escrita numa só linha usando os combinadores $f \cdot g$, $f \times g$, etc que até agora estudou, desde que tenha à sua disposição a seguinte biblioteca de funções **genéricas**:

- $sort : A^* \rightarrow A^*$
— ordena listas de A segundo uma ordem previamente assumida (sobre A)
- $collect : (A \times B)^* \rightarrow (A \times B^*)^*$
— agrupa uma sequência de pares segundo os respectivos primeiros elementos, e.g.
 $collect [(1, 2), (5, 6), (1, 3)] = [(1, [2, 3]), (5, [6])]$
- $discollect : (A \times B^*)^* \rightarrow (A \times B)^*$
— inversa da anterior
- $converse : (A \times B)^* \rightarrow (B \times A)^*$
— troca os elementos de cada par entre si
- $comp : (A \times B)^* \rightarrow (B \times C)^* \rightarrow (A \times C)^*$
— encadeia as sequências de entrada de acordo com os elementos em comum (de tipo B).

Use um interpretador de Haskell para verificar que o seu plano para f está bem tipificado, *sem* implementar as funções dadas.

¹Definições de funções que recorrem a variáveis dizem-se “pointwise”; as correspondentes versões sem variáveis dizem-se “point-free”.