

Cálculo de Programas

Algebra of Programming

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

2026/27 — Ficha (Worksheet) nr.º 2

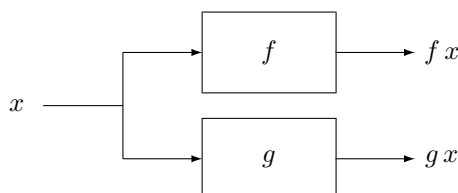
1. Na ficha anterior explorou-se o conceito de composição **sequencial** de computações. Um outro tipo de composição, dita **paralela**,

*The previous class explored the concept of **sequential** composition of two computations. Another type of composition, called **parallel** composition,*

$$\langle f, g \rangle x = (f \ x, g \ x) \quad (\text{F1})$$

é muito importante pois permite-nos fazer correr as duas funções ao mesmo tempo, como mostra o desenho que se segue:

is very important because it allows us to run both functions at the same time, as the drawing shows:



Recordando da aula teórica as funções de projecção

Recalling from the theory class the following projection functions

$$\begin{cases} \pi_1(x, y) = x \\ \pi_2(x, y) = y \end{cases} \quad (\text{F2})$$

demonstre as seguintes propriedades da computação paralela:

prove the following properties of parallel composition:

$$\pi_1 \cdot \langle f, g \rangle = f \quad (\text{F3})$$

$$\pi_2 \cdot \langle f, g \rangle = g \quad (\text{F4})$$

2. O combinador $\langle f, g \rangle$ — isto é, “*f em paralelo com g*” — satisfaz a seguinte propriedade, dita **universal**:

*The $\langle f, g \rangle$ combinator — that is, “f in parallel with g” — satisfies the following property, called **universal**:*

$$k = \langle f, g \rangle \equiv \begin{cases} \pi_1 \cdot k = f \\ \pi_2 \cdot k = g \end{cases} \quad (\text{F5})$$

Identifique-a no formulário. Que outra propriedade desse formulário obtém fazendo $k = id$ e simplificando?

Find it in the reference sheet. Which other property of this form do you get by making $k = id$ and simplifying?

3. O formulário define também esta variante da composição paralela:

In the same reference sheet the following variant of parallel composition is defined:

$$f \times g = \langle f \cdot \pi_1, g \cdot \pi_2 \rangle \quad (F6)$$

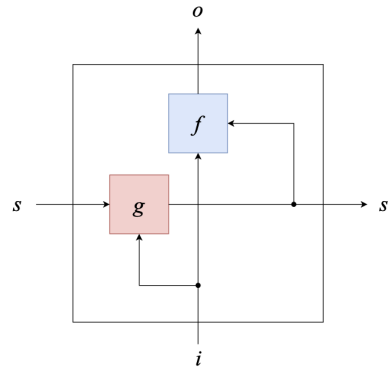
Mostre que $f \times g$ é a função:

Show that $f \times g$ is the function:

$$(f \times g)(x, y) = (f x, g y)$$

4. Determinados tipos de *redes neuronais*, muito usadas em **IA**, baseiam-se em unidades funcionais como a que o desenho mostra:

Certain types of neural networks, widely used in AI, are based on functional units like the one shown in the diagram:



Podemos ver o desenho como uma representação da função,

We can see this diagram as a representation of the function,

$$u : (S \times I) \rightarrow (S \times O) \\ u(s, i) = \text{let } s' = g(s, i) \text{ in } (s', f(s', i)) \quad (F7)$$

assumindo f e g pré-definidas.

assuming f and g pre-defined.

- (a) Mostre que a função u pode ser definida, sem variáveis, de forma bem mais simples:

(a) Show that function u can be defined, without variables, in a much simpler way:

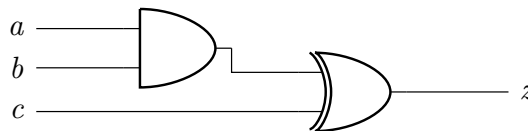
$$u = \langle \pi_1, f \rangle \cdot \langle g, \pi_2 \rangle \quad (F8)$$

- (b) Explique por suas palavras a diferença entre u e a função $v = \langle g, f \rangle$, para as mesmas funções f e g de (F8).

(b) Explain in your own words the difference between u and the function $v = \langle g, f \rangle$, for the same functions f and g from (F8).

5. O circuito booleano

The following Boolean circuit



pode descrever-se pela função f que se segue, *can be described by the following function f*

$$\begin{cases} f : (\mathbb{B} \times \mathbb{B}) \times \mathbb{B} \rightarrow \mathbb{B} \\ f = \text{xor} \cdot (\text{and} \times \text{id}) \end{cases} \quad (\text{F9})$$

onde $\text{and } (a, b) = a \wedge b$ e $\text{xor } (x, y) = x \oplus y$. *where $\text{and } (a, b) = a \wedge b$ and $\text{xor } (x, y) = x \oplus y$.*

Mostre que f se pode também definir como se segue: *Show that f can also be defined as follows:*

$$f((a, b), c) = (a \wedge b) \oplus c$$

6. Implemente e teste f (F9) no GHCi, após carregar a biblioteca *Cp.hs* disponível no material pedagógico.

NB: recorde que $x \oplus y = x \neq y$.

*Encode and test f (F9) on GHCi, after loading the library *Cp.hs* available from the course website.*

NB: recall $x \oplus y = x \neq y$.

7. Derive a partir de (F5) a lei

From (F5) derive the law

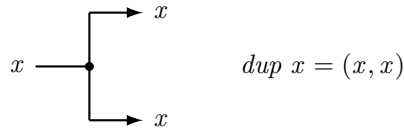
$$\langle h, k \rangle \cdot f = \langle h \cdot f, k \cdot f \rangle$$

que também consta desse formulário sob a designação *fusão- $\times$* .

that also appears in the reference sheet under the name $\times$ -fusion.

8. Uma das operações essenciais em processamento da informação é a sua *duplicação*:

A core operation in information processing is data duplication:



Recorra à lei de fusão- $\times$ para demonstrar a seguinte propriedade da duplicação de informação:

Derive from the $\times$ -fusion law the following property of data duplication:

$$\text{dup} \cdot f = \langle f, f \rangle$$

9. O combinador

The following combinator

$$\text{const} :: a \rightarrow b \rightarrow a$$

$$\text{const } a \ b = a$$

está disponível em Haskell para construir funções constantes, sendo habitual designarmos $\text{const } k$ por $\underline{k}$. Demonstre a igualdade

is available in Haskell to build constant functions. (One usually abbreviates $\text{const } k$ to $\underline{k}$.) Prove the equality

$$\underline{(b, a)} = \langle \underline{b}, \underline{a} \rangle \quad (\text{F10})$$

a partir da propriedade universal do produto e das propriedades das funções constantes que constam do formulário.

based on the $\times$ -universal property and the properties of constant functions that appear in the reference sheet.

10. Definindo $\text{swap} = \langle \pi_2, \pi_1 \rangle$, demonstre

Having defined $\text{swap} = \langle \pi_2, \pi_1 \rangle$, prove

$$(g \times f) \cdot \text{swap} = \text{swap} \cdot (f \times g) \quad (\text{F11})$$

usando as leis que conhece do Cálculo de Programas

using the laws of the Algebra of Programming studied thus far.

11. **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 **#all-cp2526** 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 #all-cp2526 Slack channel, so as to trigger discussion among other colleagues.*
The requirements of the problem are given below.

Problem requirements: *Given a name, for instance "Jose Nuno Oliveira" we wish to obtain its acronym and its short version, as suggested below:*

```
*Cp> acronym "Jose Nuno Oliveira"
"JNO"
*Cp> short "Jose Nuno Oliveira"
"Jose Oliveira"
*Cp>
```

Define

```
acronym = ...
short = ...
```

subject to the following restrictions:

- *you cannot use argument variables ($x, y, \dots$)*
- *you can use function composition $f \cdot g$ and the parallel combinator $\langle f, g \rangle$ as well as any function available from module `Cp.hs`*
- *you can resort to Haskell standard functions such as e.g. `map`, `filter` and so on.*