## Cálculo de Programas Algebra of Programming

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

2022/23 - Ficha (Exercise sheet) nr. 12 – última (last)

1. Considere a função:

Let function

$$x \ominus y = \mathbf{if} \ x \leqslant y \ \mathbf{then} \ 0 \ \mathbf{else} \ 1 + x \ominus (y+1)$$

Quais os valores das expressões  $(3\ominus 2)\ominus 3$  e  $(3\ominus 4)+4$ ? Codifique  $\widehat{\ominus}:\mathbb{N}_0\times\mathbb{N}_0\to\mathbb{N}_0$  como um anamorfismo de naturais e faça o respectivo diagrama.

be given. Evaluate  $(3\ominus 2)\ominus 3$  and  $(3\ominus 4)+4$  and encode  $\widehat{\ominus}: \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$  as an anamorphism over  $\mathbb{N}_0$ . Draw the corresponding diagram.

2. (a) Mostre que a conhecida definição da função factorial segue o padrão recursivo

(a) Show that the well-known definition of the factorial function follows the recursive pattern

$$f \cdot \mathsf{in} = g \cdot \mathsf{G} \ \langle id, f \rangle \tag{F1}$$

identificando in, g e G. (b) A lei que se segue mostra que esse padrão é o hilomorfismo

by identifying in, g and G. (b) The following law shows that pattern (F1) is a hylomorphism

$$f \cdot \mathsf{in} = g \cdot \mathsf{G} \langle id, f \rangle \equiv f = (g) \cdot (\mathsf{G} \langle id, id \rangle \cdot \mathsf{out})$$
 (F2)

cuja base é o bifunctor:

whose basis is bifunctor:

$$\mathsf{B}\left(X,Y\right) = \mathsf{G}\left(X \times Y\right)$$

Apresente justificações para os passos do cálculo de (F2) que se segue:

*Provide justifications for each step of the calculation of (F2) that follows:* 

3. O algoritmo "bubble-sort" é o ciclo-for

The "bubble-sort" algorithm is the for-loop

```
\begin{array}{l} bSort \; xs = \text{for bubble} \; xs \; (\text{length} \; xs) \; \textbf{where} \\ \text{bubble} \; (x:y:xs) \\ \mid x>y=y: \text{bubble} \; (x:xs) \\ \mid \text{otherwise} = x: \text{bubble} \; (y:xs) \\ \text{bubble} \; x=x \end{array}
```

cujo corpo de ciclo (bubble) é um hilomorfismo bubble = [conquer, divide]. Identifique os genes divide e conquer desse hilomorfismo. Sugestão: siga a heurística que foi usada nas aulas teóricas para fazer o mesmo para a função de Fibonacci.

<sup>a</sup>Se não tiver vindo à aula teórica veja a mesma heurística a ser aplicada à função de Fibonacci no vídeo T9b, a começar em t=28:09.

whose loop-body (bubble) is a hylomorphism bubble = [conquer, divide]. Identify the genes divide and conquer of this hylomorphism. Suggestion: follow the heuristics that were used in the lectures to do the same for the Fibonacci function.<sup>a</sup>

4. Todo o ciclo-*while* que termina pode ser definido por

Every terminating while-loop can be defined by

while 
$$p f g = \mathbf{tailr} ((g+f) \cdot (\neg \cdot p)?)$$
 (F3)

recorrendo ao combinador de "tail recursion"

using the "tail recursion" combinator

$$\mathbf{tailr} f = [\![\mathsf{join}, f]\!] \tag{F4}$$

que é um hilomorfismo de base B (X, Y) = X + Y, para join = [id, id].

Derive a definição pointwise de while p f g, sabendo que qualquer  $h = [\![f,g]\!]$  que termine é tal que  $h = f \cdot F h \cdot g$ .

which is a hylomorphism of basis  $\mathsf{B}(X,Y) = X + Y$ , for  $\mathsf{join} = [id\ , id]$ .

Derive the pointwise definition of while p f g, knowing that any terminating  $h = [\![f,g]\!]$  is such that  $h = f \cdot F \cdot h \cdot g$ .

5. Considere a seguinte lei de fusão de tailr, válida sempre que  $(\mathbf{tailr}\ g) \cdot f$  termina:

Consider the following fusion-law of tailr, valid whenever (tailr g)  $\cdot$  f terminates:

$$(\mathbf{tailr}\ g) \cdot f = \mathbf{tailr}\ h \iff (id + f) \cdot h = g \cdot f \tag{F5}$$

Complete a seguinte demonstração dessa lei.

Complete de following proof of (F5).

$$\begin{array}{lll} & (\mathbf{tailr}\;g)\cdot f = \mathbf{tailr}\;h \\ & & & & & & \\ & (\!( \nabla )\!)\cdot [\!(g)\!]\cdot f = (\!( \nabla )\!)\cdot [\!(h)\!] \\ & \leftarrow & & & & & \\ & (\!(g)\!]\cdot f = [\!(h)\!] \\ & \leftarrow & & & & & \\ & g\cdot f = (id+f)\cdot h & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & \\ & & & \\$$

<sup>&</sup>lt;sup>a</sup>If you missed the lectures, see the same heuristics to be applied to the Fibonacci function in video T9b, starting at t=28:09.

6. Demonstre a seguinte propriedade da composição monádica:

Prove the following property of monadic composition:

$$f \bullet [g, h] = [f \bullet g, f \bullet h] \tag{F6}$$

7. Seja M um monad e T um functor. Em Haskell, a instância para listas (T  $X=X^*$ ) da função monádica

Let M be a monad and T a functor. In Haskell, the instance for lists (T  $X = X^*$ ) of the monadic function

$$\mathsf{sequence} : \mathsf{T} \; (\mathsf{M} \; X) \to \mathsf{M} \; (\mathsf{T} \; X)$$

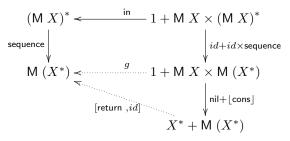
é o catamorfismo

is the catamorphism

sequence = 
$$(g)$$
 where  $g = [\text{return }, id] \cdot (\text{nil} + \lfloor \text{cons} \rfloor)$   
 $|f|(x, y) = \mathbf{do} \{a \leftarrow x; b \leftarrow y; \text{return } (f(a, b))\}$ 

tal como se mostra neste diagrama:

as in the following diagram:

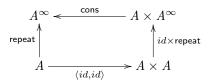


Partindo da propriedade universal-cata, derive uma versão de sequence em Haskell com variáveis que não recorra à composição de funções.

Starting from the universal-cata property, derive a version of sequence in Haskell with variables that doesn't resort to function composition.

8. Mostre que o anamorfismo repeat  $= [\![\langle id, id \rangle]\!]$  definido pelo diagrama

Show that the anamorphism repeat = ((id, id)) defined by the diagram



é a função: repeat a=a: repeat a. De seguida, recorrendo às leis dos anamorfismos mostre que, apesar de não terminar $^a$ , repeat satisfaz a propriedade map  $f \cdot \text{repeat} = \text{repeat} \cdot f$ .

is the function: repeat a=a: repeat a. Then, resorting to the laws of anamorphisms, show that, despite not ending  $^a$ , repeat satisfies the map f property  $\cdot$  repeat = repeat  $\cdot$  f.

 $<sup>^</sup>a\mathrm{Por}$  isso usamos, no diagrama,  $A^\infty$  em vez de  $A^*,$  para incluir também as listas infinitas.

b "Verifique" este facto comparando, por exemplo, (take  $10 \cdot$  map succ · repeat)  $1 \text{ com (take } 10 \cdot$  repeat · succ)  $1 \cdot$ 

 $<sup>^</sup>a$ That's why we use, in the diagram,  $A^{\infty}$  instead of  $A^*$ , to also include infinite lists.

 $<sup>^</sup>b$  "Verify" this fact by comparing, for example, (take  $10\cdot \text{map succ}\cdot \text{repeat})$  1 with (take  $10\cdot \text{repeat}\cdot \text{succ})$  1.