

Cálculo de Programas

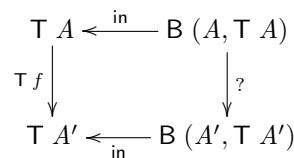
Algebra of Programming

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

2022/23 - Ficha (*Exercise sheet*) nr. 10

1. O isomorfismo $\text{in} : B(A, T A) \rightarrow T A$ construtor dos habitantes de um tipo recursivo (paramétrico) de base B é ele próprio paramétrico em A . Complete o seguinte diagrama que capta a propriedade natural (grátis) de in :

Isomorphism $\text{in} : B(A, T A) \rightarrow T A$ constructing inhabitants of a recursive (parametric) base type B is itself parametric on A . Complete the following diagram that captures the natural (free) property of in :



Instancie essa propriedade para listas, em que

Instantiate this property for lists, where

$$\begin{cases}
 T A = A^* \\
 B(X, Y) = 1 + X \times Y \\
 T f = \text{map } f
 \end{cases}$$

Desenvolva essa igualdade até chegar à sua formulação sem qualquer dos construtores *pointfree* estudados nesta disciplina. O que é que obteve, afinal?

Unfold the equality until a formulation is reached involving none of the pointfree constructors studied in this course. What did you get, after all?

2. O formulário desta disciplina apresenta duas definições alternativas para o functor $T f$, uma como *catamorfismo* e outra como *anamorfismo*. Identifique-as e acrescente justificações à seguinte prova de que essas definições são equivalentes:

The reference sheet of this course presents two alternative definitions for functor $T f$, one as a catamorphism and another as an anamorphism. Identify them and fill in justifications in the following proof that such definitions are equivalent:

$$\begin{aligned}
 & T f = (\text{in} \cdot B(f, id)) \\
 \equiv & \{ \dots \} \\
 & T f \cdot \text{in} = \text{in} \cdot B(f, id) \cdot F(T f) \\
 \equiv & \{ \dots \}
 \end{aligned}$$

$$\begin{aligned}
& T f \cdot \text{in} = \text{in} \cdot B (id, T f) \cdot B (f, id) \\
\equiv & \{ \dots \} \\
& \text{out} \cdot T f = F (T f) \cdot B (f, id) \cdot \text{out} \\
\equiv & \{ \dots \} \\
& T f = \llbracket B (f, id) \cdot \text{out} \rrbracket \\
& \square
\end{aligned}$$

3. Mostre que o catamorfismo de listas $\text{length} = \llbracket [\text{zero}, \text{succ} \cdot \pi_2] \rrbracket$ é a mesma função que o anamorfismo de naturais $\llbracket (id + \pi_2) \cdot \text{out}_{\text{List}} \rrbracket$. *Show that the list catamorphism $\text{length} = \llbracket [\text{zero}, \text{succ} \cdot \pi_2] \rrbracket$ is the same function as the \mathbb{N}_0 -anamorphism $\llbracket (id + \pi_2) \cdot \text{out}_{\text{List}} \rrbracket$.*

4. Recorra às leis dos catamorfismos para verificar a propriedade natural *Use the laws of catamorphisms to check the natural property*

$$(LTree f) \cdot \text{mirror} = \text{mirror} \cdot (LTree f) \tag{F1}$$

onde mirror é o catamorfismo *where mirror is the catamorphism*

$$\begin{cases} \text{mirror} :: LTree a \rightarrow LTree a \\ \text{mirror} = \llbracket \text{in} \cdot (id + \text{swap}) \rrbracket \end{cases} \tag{F2}$$

que “espelha” uma árvore e *that “mirrors” a tree, and*

$$LTree f = \llbracket \text{in} \cdot (f + id) \rrbracket$$

é o correspondente functor de tipo. *is the corresponding type functor.*

5. Mostre que o anamorfismo que calcula os sufixos de uma lista *Show that the anamorphism that computes the suffixes of a list*

$$\text{suffixes} = \llbracket g \rrbracket \textbf{ where } g = (id + \langle \text{cons}, \pi_2 \rangle) \cdot \text{out}$$

é a função: *is the function:*

$$\begin{aligned}
\text{suffixes } [] &= [] \\
\text{suffixes } (h : t) &= (h : t) : \text{suffixes } t
\end{aligned}$$

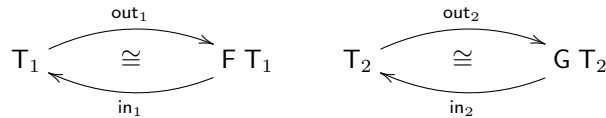
6. Mostre que a função mirror (F2) se pode definir como o anamorfismo *Show that the function mirror (F2) can be defined as the anamorphism*

$$\text{mirror} = \llbracket (id + \text{swap}) \cdot \text{out} \rrbracket \tag{F3}$$

onde out é a conversa de in . Volte a demonstrar a propriedade $\text{mirror} \cdot \text{mirror} = id$, desta vez recorrendo à lei de fusão dos anamorfismos. *where out is the converse of in . Prove $\text{mirror} \cdot \text{mirror} = id$ again, this time resorting to the fusion-law of anamorphisms.*

7. O facto de $\text{length} : A^* \rightarrow \mathbb{N}_0$ poder ser definida tanto como *catamorfismo* de listas como *anamorfismo* de naturais (questão 3) pode generalizar-se da forma seguinte: sejam dados dois tipos indutivos

The fact that $\text{length} : A^ \rightarrow \mathbb{N}_0$ is both a list-catamorphism and a \mathbb{N}_0 -anamorphism (question 3) generalizes as follows: let two inductive types be given*



e $\alpha : F X \rightarrow G X$, isto é, α satisfaz a propriedade *grátis*:

and $\alpha : F X \rightarrow G X$, that is, α satisfying the free property:

$$G f \cdot \alpha = \alpha \cdot F f \tag{F4}$$

Então $\langle \text{in}_2 \cdot \alpha \rangle = \llbracket \alpha \cdot \text{out}_1 \rrbracket$, como se mostra a seguir (complete as justificações):

Then $\langle \text{in}_2 \cdot \alpha \rangle = \llbracket \alpha \cdot \text{out}_1 \rrbracket$, as shown below (complete the justifications):

$$\begin{aligned} & k = \langle \text{in}_2 \cdot \alpha \rangle \\ \equiv & \{ \dots\dots\dots \} \\ & k \cdot \text{in}_1 = \text{in}_2 \cdot \alpha \cdot F k \\ \equiv & \{ \dots\dots\dots \} \\ & \text{out}_2 \cdot k = G k \cdot \alpha \cdot \text{out}_1 \\ \equiv & \{ \dots\dots\dots \} \\ & k = \llbracket \alpha \cdot \text{out}_1 \rrbracket \\ & \square \end{aligned}$$

Finalmente, identifique T_1 , T_2 e α para o caso de $k = \text{length}$.

Finally, identify T_1 , T_2 and α in the case $k = \text{length}$.