Cálculo de Programas

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

2019/20 - Ficha nr.° 13 (última)

1. Repare que as projecções $A \xleftarrow{\pi_1} A \times B \xrightarrow{\pi_2} B$ são funções binárias e como tal podem ser "curried", $A^B \xleftarrow{\overline{\pi_1}} A \xrightarrow{\overline{\pi_2}} B^B$. Verifica-se que:

$$\overline{\pi_1} = \mathsf{const} \tag{F1}$$
$$\overline{\pi_2} = \underline{id} \tag{F2}$$

onde $\mathsf{const}\ a = \underline{a}$, a função constante que dá a como resultado. Apresente justificações para os passos das provas respectivas que se seguem:

$$\begin{array}{ll}
& \overline{\pi_1} = \mathsf{const} \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \mathsf{ap} \cdot (\mathsf{const} \times id) = \pi_1 \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \mathsf{ap} \cdot (\mathsf{const} \times id)\ (a, b) = \pi_1\ (a, b) \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \mathsf{ap}\ (\mathsf{const}\ a, b) = a \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \mathsf{const}\ a\ b = a \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \underline{a}\ b = a \\
\Box &
\end{array}$$

$$\begin{array}{ll}
& \overline{\pi_2} = \underline{id} \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \mathsf{ap} \cdot (\underline{id} \times id) = \pi_2 \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \mathsf{ap}\ ((\underline{id} \times id)\ (a, b)) = b \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \mathsf{ap}\ (id, b) = b \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& b = b \\
\Box &
\end{array}$$

2. Com base em (F2) demonstre

$$\underline{g} = \overline{g \cdot \pi_2} \tag{F3}$$

desenhando um diagrama explicativo dos tipos em jogo.

3. Baseie-se em (F3) para mostrar que, no mónade de acções sobre um estado, $\mathsf{St}\ S\ A = (A \times S)^S$, se tem

$$\mathbf{do}\ \{x; y\} = x \gg y = y \cdot \pi_2 \cdot x \tag{F4}$$

ou seja: o resultado da acção x é ignorado pela acção y. **Sugestão**: recorde das aulas teóricas que, neste mónade, $(\gg\!= f) = \widehat{f}^S$.

4. Considere as seguintes acções que habitam o mónade de estado $\mathsf{St}\ S\ A = (A \times S)^S$, para um dado espaço de estados S:

$$query\ f = \langle f, id \rangle \quad \text{(F5)}$$
$$modify\ g = \langle !, g \rangle \quad \text{(F6)}$$

A acção $query\ f$ interroga o estado aplicando-lhe a observação f e deixando-o inalterado; já a acção $modify\ g$ recorre a g para actualizar o valor corrente do estado, dando como resultado um mero "acknowledgement" da acção realizada.

Mostre que as igualdades

$$\mathbf{do}\ \{ modify\ id; query\ g \} = query\ g \quad \text{(F7)}$$
$$\mathbf{do}\ \{ modify\ f; modify\ g \} = modify\ (g \cdot f) \quad \text{(F8)}$$

se verificam.

5. Nos vídeos ds aulas teóricas aparece a função $mmap$ (localize-a), que resultou da *monadificação* da função map das listas, e a função sequence (localize-a também). Verifica-se que sequence $=$ $mmap\ f$ para um dado f. Qual? Responda com base na inspecção dos tipos das duas funções.

6. Localize a definição das acções get e put sobre o mónade de estado nos vídeos das aulas teóricas (ou nos apontamentos da cadeira) e demonstre a igualdade

$$\mathbf{do}\ \{ a \leftarrow get; put\ a \} = get \gg\!= put = modify\ id \quad \text{(F9)}$$

7. O seguinte esboço de código diz respeito a uma função

$$accum :: Num\ b \Rightarrow \mathsf{LTree}\ b \rightarrow \mathsf{LTree}\ b$$
$$accum\ t = \ldots\ldots\ldots\ldots\ B \ldots\ldots\ldots\ldots$$

que, a partir de uma árvore de folhas de inteiros, produz uma outra árvore (com a mesma forma) em que o conteúdo de cada folha é substituído pela sua soma com os conteúdos de todas as folhas à sua esquerda, recorrendo à função auxiliar:

$$aux :: Num\ a \Rightarrow \mathsf{LTree}\ a \rightarrow \mathsf{St}\ a\ (\mathsf{LTree}\ a)$$
$$aux\ (Leaf\ x) = \ldots\ldots\ldots\ldots\ A \ldots\ldots\ldots\ldots$$
$$aux\ (Fork\ (esq, dir)) = \mathbf{do}\ \{$$
$$\quad esq' \leftarrow aux\ esq;$$
$$\quad dir' \leftarrow aux\ dir;$$
$$\quad \mathsf{return}\ (Fork\ (esq', dir'))$$
$$\quad \}$$

Por exemplo, $accum\ (Fork\ (Leaf\ 1, Fork\ (Leaf\ 2, Leaf\ 3)))$ resulta em

$$Fork\ (Leaf\ 1, Fork\ (Leaf\ 3, Leaf\ 6)).$$

Com base em exemplos semelhantes das aulas teóricas, complete o código em A e B.