Métodos Formais de Programação
$4^{\underline{o}}$ Ano da LMCC (701067) + LESI (531360)
Ano Lectivo de 1995/96

Exame (época normal) — 29 de Janeiro 1996
9h30
Anfiteatro A4

NB: *sempre que os seus cálculos utilizarem resultados, leis, teoremas* etc. *disponíveis no material pedagógico desta disciplina não os copie para a sua folha de exame — limite-se a citar a sua referência (*e.g. *lei (2.102), exercício 2.15,* etc.*).*

PROVA COM CONSULTA (3 $horas$)

Questão 1 Considere as produções seguintes de uma gramática livre de contexto G.

$$\begin{array}{lcl} \langle A \rangle & ::= & \mathtt{a}\,\langle B \rangle \mid \mathtt{b}\,\langle A \rangle\,\mathtt{a}\,\langle B \rangle^* \\ \langle B \rangle & ::= & \mathtt{a}\ \mathtt{b} \mid \mathtt{a}\,\langle A \rangle^+\,\mathtt{c} \end{array}$$

Construa a assinatura Σ_G correspondente.

RESOLUÇÃO: A gramática proposta é equivalente a:

$$\begin{array}{lcl} \langle A \rangle & ::= & \mathtt{a}\,\langle B \rangle \mid \mathtt{b}\,\langle A \rangle\,\mathtt{a}\,\langle C \rangle \\ \langle C \rangle & ::= & \epsilon \mid \langle B \rangle \langle C \rangle \\ \langle B \rangle & ::= & \mathtt{a}\ \mathtt{b} \mid \mathtt{a}\,\langle D \rangle\,\mathtt{c} \\ \langle D \rangle & ::= & \langle A \rangle \mid \langle A \rangle \langle D \rangle \end{array}$$

(relembre-se (1.4) e (1.5) e o exercício 1.5 da pág. 13 da sebenta) o que conduz à assinatura:

$$\Sigma_G = \left\{ \begin{array}{lcl} \sigma_1 & : & \langle B \rangle \to \langle A \rangle \\ \sigma_2 & : & \langle A \rangle \times \langle C \rangle \to \langle A \rangle \\ \sigma_3 & : & \to \langle C \rangle \\ \sigma_4 & : & \langle B \rangle \times \langle C \rangle \to \langle C \rangle \\ \sigma_5 & : & \to \langle B \rangle \\ \sigma_6 & : & \langle D \rangle \to \langle B \rangle \\ \sigma_7 & : & \langle A \rangle \to \langle D \rangle \\ \sigma_8 & : & \langle A \rangle \times \langle D \rangle \to \langle D \rangle \end{array} \right.$$

□

Questão 2 Dada uma aplicação $f : A \longrightarrow B$, $S \subseteq A$ e $s \in A^*$, relembre os "filtros" $f[S]$ — (1.22) da pág. 23 da sebenta — e $<\!f(a) \mid a \leftarrow s\!>$ — (1.27) da pág. 25 da sebenta. Vamos defini-las pela notação alternativa que se segue:

$$\begin{array}{lcl} 2^f & : & 2^A \longrightarrow 2^B \\ 2^f(S) & \stackrel{\text{def}}{=} & f[S] \end{array} \qquad e \qquad \begin{array}{lcl} f^* & : & A^* \longrightarrow B^* \\ f^*(s) & \stackrel{\text{def}}{=} & <\!f(a) \mid a \leftarrow s\!> \end{array}$$

Identifique, justificando, quais as leis que estudou (ou consultou) correspondem, uma vez escritas na notação alternativa que se acaba de propôr, às igualdades seguintes:

$$elems \circ f^* = 2^f \circ elems \tag{1}$$
$$g^* \circ f^* = (g \circ f)^* \tag{2}$$
$$length \circ f^* = length \tag{3}$$
$$tail \circ f^* = f^* \circ tail \tag{4}$$

RESOLUÇÃO:

(1) — esta igualdade converte-se em

$$\begin{array}{rcl} (elems \circ f^*)(s) & = & (2^f \circ elems)(s) \\ & \updownarrow & \\ elems(f^*(s)) & = & 2^f(elems(s)) \\ & \updownarrow & \\ elems(<f(a) \mid a \leftarrow s>) & = & f[elems(s)] \end{array}$$

isto é, na lei (A.70).

(2) — esta igualdade converte-se em

$$\begin{array}{rcl} (g^* \circ f^*)(s) & = & ((g \circ f)^*)(s) \\ & \updownarrow & \\ g^*(f^*(s)) & = & <(g \circ f)(a) \mid a \leftarrow s> \\ & \updownarrow & \\ g^*(<f(a) \mid a \leftarrow s>) & = & <g(f(a)) \mid a \leftarrow s> \\ & \updownarrow & \\ <g(b) \mid b \leftarrow <f(a) \mid a \leftarrow s>> & = & <(g(f(a)) \mid a \leftarrow s> \end{array}$$

isto é, na lei (A.60).

(3) — esta igualdade converte-se em

$$\begin{array}{rcl} (length \circ f^*)(s) & = & length(s) \\ & \updownarrow & \\ length(f^*(s)) & = & length(s) \\ & \updownarrow & \\ length(<f(a) \mid a \leftarrow s>) & = & length(s) \end{array}$$

isto é, na lei (A.66).

(4) — esta igualdade converte-se em

$$\begin{array}{rcl} (tail \circ f^*)(s) & = & (f^* \circ tail)(s) \\ & \updownarrow & \\ tail(f^*(s)) & = & f^*(tail(s)) \\ & \updownarrow & \\ tail(<f(a) \mid a \leftarrow s>) & = & <f(a) \mid a \leftarrow tail(s)> \end{array}$$

isto é, na lei (A.74).

□

Questão 3 Considere o seguinte fragmento (aliás incompleto) de um programa em C para gestão de um determinado tipo de tabelas de 'hashing':

```
#define    TAM   100
.
.
.
.
struct inf
{
```

```
  med      largura;
  med      altura;
};

struct arvbin
{
  int codigo;
  struct inf    inform;
  struct arvbin *dir;
  struct arvbin *esq;
};

typedef  struct inf INF;
typedef  struct arvbin *AB;

AB hash[TAM];
```

1. Converta para modelo de dados em $Sets$ o tipo da variável `hash`, à luz da analogia que estudou entre as construções primitivas de $Sets$ e a notação para definição de estruturas de dados em linguagens de programação estruturadas.
2. Sabendo que a função de 'hashing' opera sobre o campo `codigo` de `arvbin`, especifique o seguinte invariante a afixar a esse modelo de dados: *para cada índice do 'array' de 'hashing', os códigos que ocorrem em nós da respectiva árvore de colisão têm todos o mesmo valor de 'hashing', que é exactamente o índice respectivo.*

RESOLUÇÃO:

1. Tem-se:

$$\begin{array}{rcl} HASH & \cong & AB^{TAM} \\ TAM & \cong & 100 \\ AB & \cong & 1 + arvbin \\ arvbin & \cong & \mathbb{Z} \times INF \times AB \times AB \\ INF & \cong & med \times med \end{array}$$

onde se pode reconhecer, como forma de armazenamento de *sinónimos*, a estrutura *árvore binária*:

$$AB \quad \cong \quad 1 + \mathbb{Z} \times INF \times AB \times AB$$

2. Seja $H : \mathbb{Z} \longrightarrow 100$ a função de 'hashing'. Ter-se-á:

$$\begin{array}{rcl} \phi & : & HASH \longrightarrow 2 \\ \phi(t) & = & \forall i \in TAM : \phi'(i, t(i)) \end{array}$$

onde

$$\begin{array}{rcl} \phi' & : & TAM \times AB \longrightarrow 2 \\ \phi'(i,x) & = & \left\{ \begin{array}{rcl} x = \langle 1, NIL \rangle & \Rightarrow & V \\ x = \langle 2, a \rangle & \Rightarrow & H(\pi_1(a)) = i \wedge \phi'(i, \pi_3(a)) \wedge \phi'(i, \pi_4(a)) \end{array} \right. \end{array}$$

□

Questão 4 Uma noção "intermédia" entre conjuntos (2^A) e sequências (A^*) é a de *multi-conjunto*, *i.é* um conjunto em que qualquer elemento $a \in A$ pode ocorrer mais do que uma vez:

$$MSet(A) \quad \cong \quad A \hookrightarrow \mathbb{N} \tag{5}$$

Assim, o multiconjunto vazio é representado pela função finita vazia $(\quad)$, e a união de multiconjuntos corresponde à adição de multiciplidades. Especifique a operação de *intersecção* de multi-conjuntos.

RESOLUÇÃO:

$$\Cap \; : \; MSet(A) \times MSet(A) \longrightarrow MSet(A)$$

$$x \Cap y \stackrel{\text{def}}{=} \begin{pmatrix} a \\ inf(x(a), y(a)) \end{pmatrix}_{a \in dom(x) \cap dom(y)}$$

onde

$$inf \; : \; \mathbb{N} \times \mathbb{N} \longrightarrow \mathbb{N}$$

$$inf(x,y) \stackrel{\text{def}}{=} \begin{cases} x \leq y & \Rightarrow \quad x \\ x > y & \Rightarrow \quad y \end{cases}$$

□

Questão 5 Nas aulas foi abordado o seguinte modelo matemático para sistemas hierárquicos de ficheiros (tipo UNIX ou MS/DOS):

FS	$\cong$	$Id \hookrightarrow (File + Dir)$	*/*FS = 'file system' */* */*Id = identificador */* */*File = ficheiro */*
Dir	$\cong$	FS	*/* Dir = directoria,* i.é *sub-file system */*
$Path$	$\cong$	Id^*	*/* Path = trajectória que selecciona uma directoria ou ficheiro */*

Complete o seguinte esboço de uma função sobre FS que especifica a semântica do comando `rmdir p`, onde `p` é uma 'path':

$$rmdir \; : \; FS \times Path \longrightarrow FS$$

$$rmdir(fs,p) \stackrel{\text{def}}{=} \begin{cases} length(p) = 0 \;\Rightarrow\; fs \\ length(p) = 1 \;\Rightarrow\; \begin{array}{ll} let & i = head(p) \\ in & \begin{cases} i \notin dom(fs) \;\Rightarrow\; fs \\ i \in dom(fs) \;\Rightarrow\; \begin{cases} \text{is-}File(fs(i)) \;\Rightarrow\; fs \\ \text{is-}Dir(fs(i)) \;\Rightarrow\; \ldots \end{cases} \end{cases} \end{array} \\ length(p) > 1 \;\Rightarrow\; \ldots \end{cases}$$

RESOLUÇÃO: As primeiras reticências completam-se com:

$$\ldots \;=\; \begin{cases} fs(i) = (\;\;) \;\Rightarrow\; fs \setminus \{i\} \\ \neg(fs(i) = (\;\;)) \;\Rightarrow\; fs \end{cases}$$

As segundas reticências completam-se com:

$$\ldots \;=\; \begin{array}{ll} let & i = head(p) \\ & t = tail(p) \\ in & \begin{cases} i \notin dom(fs) \;\Rightarrow\; fs \\ i \in dom(fs) \;\Rightarrow\; \begin{cases} \text{is-}File(fs(i)) \;\Rightarrow\; fs \\ \text{is-}Dir(fs(i)) \;\Rightarrow\; fs \dagger \begin{pmatrix} i \\ rmdir(fs(i), t) \end{pmatrix} \end{cases} \end{cases} \end{array}$$

□

Questão 6 Uma tabela relacional é um conjunto de tuplos, sendo um tuplo uma atribuição de valores a atributos. Podem existir atributos omissos, por exemplo o atributo C no segundo tuplo desta tabela,

A	B	C
$a1$	$b1$	$c1$
$a1$	$b2$	
$a2$	$b1$	$c3$
$a2$	$b1$	$c1$
$a1$		$c1$
$a3$		$c1$
	$b1$	$c2$

(6)

e outros.

Pretende-se especificar o problema da geração de histogramas relativos aos atributos de uma dada tabela. Por exemplo, dada a tabela 6, têm-se os seguinte histogramas para os atributos A, B e C, respectivamente:

Especifique um modelo $\mathcal{A} : \Sigma \to Sets$ para a seguinte assinatura,

$$\Sigma = \begin{cases} valores : AtrId \times Tabela \to Valores \\ atributos : Tabela \to AtrIds \\ contagem : AtrId \times Valor \times Tabela \to Nat \\ histograma : AtrId \times Tabela \to Histograma \end{cases}$$

que pretende exibir a funcionalidade que se pretende e de acordo com a semântica informal seguinte:

$valores(a, t)$ — deverá retornar o conjunto de todos os valores que ocorrem na coluna a da tabela t, onde a designa o identificador do respectivo atributo;

$atributos(t)$ — deverá retornar o conjunto de todos os atributos que ocorrem em pelo menos um tuplo da tabela;

$contagem(a, v, t)$ — calcula o número de ocorrências de um dado valor v na coluna a de uma dada tabela t;

$histograma(a, t)$ — fornece o histograma do atributo a na tabela t.

RESOLUÇÃO:

$$\begin{aligned}
\mathcal{A} &: \Sigma \longrightarrow Sets \\
\mathcal{A}(Tabela) &\stackrel{\text{def}}{=} 2^{\mathcal{A}(Tuplo)} \\
\mathcal{A}(Tuplo) &\stackrel{\text{def}}{=} \mathcal{A}(AtrId) \hookrightarrow \mathcal{A}(Valor) \\
\mathcal{A}(AtrIds) &\stackrel{\text{def}}{=} 2^{\mathcal{A}(AtrId)} \\
\mathcal{A}(Histogama) &\stackrel{\text{def}}{=} \mathcal{A}(Valor) \hookrightarrow \mathbb{N} \\
\mathcal{A}(Valores) &\stackrel{\text{def}}{=} 2^{\mathcal{A}(Valor)} \\
\mathcal{A}(Nat) &\stackrel{\text{def}}{=} \mathbb{N}_0 \\
\mathcal{A}(valores)(a, r) &\stackrel{\text{def}}{=} \{t(a) \mid \ t \in r \wedge a \in dom(t)\} \\
\mathcal{A}(atributos)(r) &\stackrel{\text{def}}{=} \bigcup_{t \in r} dom(t) \\
\mathcal{A}(contagem)(a, v, r) &\stackrel{\text{def}}{=} card(\{t \mid \ t \in r \wedge a \in dom(t) \wedge t(a) = v\}) \\
\mathcal{A}(histograma)(a, r) &\stackrel{\text{def}}{=} \begin{pmatrix} v \\ \mathcal{A}(contagem)(a, v, r) \end{pmatrix}_{v \in \mathcal{A}(valores)(a, r)}
\end{aligned}$$

NB: Não é necessário especificar modelos para as espécies $Valor$ e $AtrId$, pois o modelo nada presume sobre elas (*cf.* parametrização). $\square$

Questão 7 Considere a seguinte função que especifica um tipo de inserção em lista de naturais: estratégia de

$$
\begin{array}{lcl}
ins & : & \mathbb{N}^* \times \mathbb{N} \longrightarrow \mathbb{N}^* \\
ins(l,a) & \stackrel{\text{def}}{=} & \left\{ \begin{array}{lcll}
l = <> & \Rightarrow & <a> & \\
l \neq <> & \Rightarrow & let & h = head(l) \\
 & & & t = tail(l) \\
 & & in & \left\{ \begin{array}{lcl} a = h & \Rightarrow & l \\ a < h & \Rightarrow & cons(a,l) \\ a > h & \Rightarrow & cons(h, ins(t,a)) \end{array} \right.
\end{array} \right.
\end{array}
$$

1. Demonstre ou refute se ins preserva o invariante sobre listas que se segue:

$$\phi(l) \stackrel{\text{def}}{=} length(l) = card(elems(l))$$

2. Qual o significado informal deste invariante? É capaz de especificar um outro invariante que ins (também) satisfaça? Justifique informalmente.

RESOLUÇÃO:

1. O facto a demonstrar ou refutar é

$$\forall l \in \mathbb{N}^*, a \in \mathbb{N} : \phi(l) \Rightarrow \phi(ins(l,a))$$

O seu significado informal é o de garantir que uma lista l não tem elementos repetidos. Vamos tentar provar que ins preserva ϕ. Prova por indução sobre $A^\star$:

(a) *Caso de Base*: ($l = <>$). Tem-se, neste caso

$$
\begin{array}{rcl}
\phi(<>) & \Rightarrow & \phi(ins(<>,a)) \\
 & \Updownarrow & \\
length(<>) = card(elems(<>)) & \Rightarrow & length(ins(<>,a)) = card(elems(ins(<>,a))) \\
 & \Updownarrow & \\
0 = card(\emptyset) & \Rightarrow & length(<a>) = card(elems(<a>)) \\
 & \Updownarrow & \\
0 = 0 & \Rightarrow & 1 = card(\{a\}) \\
 & \Updownarrow & \\
V & \Rightarrow & 1 = 1 \\
 & \Updownarrow & \\
V & \Rightarrow & V \\
 & \Updownarrow & \\
 & V &
\end{array}
$$

(b) *Salto indutivo*: ($l \neq <>$)

i. *Hipótese de indução*:

$$\phi(tail(l)) \Rightarrow \phi(ins(tail(l),a))$$

ii. *Passo*: A demonstração a fazer é uma implicação de implicações,

$$(\phi(tail(l)) \Rightarrow \phi(ins(tail(l),a))) \quad \Rightarrow \quad (\phi(l) \Rightarrow \phi(ins(l,a)))$$

que se reduz a

$$\phi(ins(tail(l),a)) \wedge \phi(l) \quad \Rightarrow \quad \phi(ins(l,a)) \tag{7}$$

uma vez que $\phi(l)$ é mais forte que $\phi(tail(l))$ [1]. Mas repare-se que (7) é falso, isto é, que existem l e a tais que $\phi(ins(tail(l), a)) \wedge \phi(l) \wedge \neg\phi(ins(l, a))$ — faça-se, por exemplo, $l = <2, 1>$ e $a = 1$. Logo a prova fracassa e assim se mostra que ins não preserva o invariante ϕ.

2. A prova anterior só fracassou porque l pode não estar estritamente ordendada. Por outro lado, ins parece inserir ordenadamente um elemento numa lista. Isto sugere que uma propriedade que ins preserva como invariante é

$$\varphi(l) \stackrel{\text{def}}{=} \forall i, j \leq length(l) : i < j \Rightarrow l(i) < l(j)$$

Repare-se que φ é, aliás, logicamente mais forte que ϕ.

□

---

[1] Veja-se o porquê desta simplificação, que é normalmente útil em demonstração de invariantes:

$$\begin{gathered}(a \Rightarrow b) \Rightarrow (c \Rightarrow d)\\ \Updownarrow \\ ((a \Rightarrow b) \wedge c) \Rightarrow d\\ \Updownarrow \\ ((\neg a \vee b) \wedge c) \Rightarrow d\\ \Updownarrow \\ (\neg a \wedge c \vee b \wedge c) \Rightarrow d\\ \Updownarrow \\ (b \wedge c) \Rightarrow d\end{gathered}$$

O último passo só é válido quando a é logicamente mais fraco que c.