Métodos Formais de Programação II +
Opção - Métodos Formais de Programação II

4.º Ano da LMCC (7008N2) + LESI (5308P3)
Ano Lectivo de 2006/07

Exame (1.ª chamada da época normal) — 25 de Junho 2007
14h00
Salas 1216, 1220, 1308

NB*: Esta prova consta de **8** alíneas todas com a mesma cotação.*

PROVA SEM CONSULTA (2 horas)

Questão 1 As rotinas mais críticas de uma biblioteca de cálculo científico foram, há vários anos, desenvolvidas a partir de especificações formais. Uma delas — $sqrt$ — implementa o cálculo de raizes quadradas e foi derivada (diz a documentação) como refinamento de

```
Sqrt(i: real) r: real
pre i < 10*9
post abs(i*i-r) < 0.001;
```

Estando a decorrer o processo de re-estruturaçao de toda a biblioteca verifica-se que há, afinal, outra função raiz quadrada, $sqrt1$, desenvolvida por um grupo diferente como refinemento de

```
Sqrt1(x: real) s: real
pre x < 10*8
post abs(s-x*x) < 0.0001;
```

O que a empresa produtora do software pretende saber é se é seguro eliminar uma destas funções (qual?), substituindo-a simplesmente pela outra. Dê a sua opinião e justique-a formalmente com base na relação de refinamento (9) estudada nesta disciplina.

Questão 2 Nas aulas desta disciplina mostrou-se que a função $abs\, i \stackrel{\text{def}}{=}$ **if** $i < 0$ **then** $-i$ **else** i satisfaz a especificação

$$\begin{array}{l} Abs(i : \mathbb{Z})\ r : \mathbb{Z} \\ \textbf{post}\ \ 0 \leq r\ \wedge\ (r = i \vee r = -i) \end{array}$$

calculando as respectivas transformadas-PF,

$$abs = N \rightarrow sym\ ,\ id \quad (1)$$
$$Abs = P_0 \cdot (id \cup sym) \quad (2)$$

— onde $sym\ i \stackrel{\text{def}}{=} -i$ e N e P_0 são, respectivamente, $\lceil \lambda i.i < 0 \rceil$ e $\lceil \lambda i.i \geq 0 \rceil$ — e mostrando que $Abs \vdash abs$. Contudo, é possível ir mais longe e mostrar que Abs é, afinal, *igual* a abs. Mostre como, apresentando os seus cálculos. Anote os factos seguintes,

$$sym = sym^{\circ} \quad (3)$$
$$N \cup P_0 = id \quad (4)$$
$$N_0 \cdot sym = sym \cdot P_0 \quad (5)$$

que lhe podem ser úteis.

Questão 3 A uma empresa de desenvolvimento de *software* foi feita, há anos, a encomenda de uma base de dados para uma biblioteca. A base de dados foi implementada sem problemas e está operacional. Recentemente, a biblioteca voltou a contactar a empresa no sentido de serem melhoradas algumas funcionalidades. Da equipa de projectistas inicial já nenhum está na empresa, pelo que foi solicitado à universidade um estagiário que possa estudar a documentação, cujos requisitos foram modelados em VDM.

Suponha que *você* é esse estagiário e que, consultado o dossier do projecto, verifica que a especificação inicial do sistema foi a seguinte,

```
System   :: books: map ISBN to Book
            taxonomy: Tax;
Book     :: description: token
            keywords: set of Subject;
Tax       = map Subject to Tax;
ISBN      = token;
Subject   = token;
```

e que os casos de teste incluem a taxonomia exemplo

```
tax0 = { "Compilers" |->
           { "Lexical-analysis" |-> {|->},
             "Syntax-analysis"  |->
                   { "LL" |-> {|->},
                     "LR" |-> {|->} } } };
```

Mais à frente, verifica que o esquema da base de dados relacional foi derivado a partir de um modelo intermédio que, ainda em VDM, é o seguinte:

```
SystemDb :: books1: map ISBN to token
            books2: set of (ISBN * Subject)
            root:   nat
            tax1:   set of nat
            tax2:   map (nat * Subject) to nat
        inv db == concreteInv(db);
```

Contudo, perdeu-se o cálculo desse modelo e a especificação de $concreteInv$.

1. Reconstitua o processo de cálculo que se perdeu, anotando cada passo com as leis que foram utilizadas e/ou as respectivas relações de representação e de abstracção.
2. Apresente uma representação válida, ao nível $SystemDb$, para o valor de teste `s0 = mk_System({|->},tax0);`
3. Redefina o invariante $concreteInv$ cuja definição que se perdeu.

Questão 4 O isomorfismo que se segue

$$A \to (B + C) \underset{\overset{+}{\bowtie}}{\overset{\triangle_+}{\cong}} (A \to B) \times (A \to C) \tag{6}$$

generaliza, para quaisquer relações, uma lei de refinamento que estudou, onde

$$M \overset{+}{\bowtie} N \overset{\text{def}}{=} i_1 \cdot M \cup i_2 \cdot N \tag{7}$$

$$\triangle_+ M \overset{\text{def}}{=} (i_1^\circ \cdot M, i_2^\circ \cdot M) \tag{8}$$

1. Mostre que $(M \overset{+}{\bowtie} N)^\circ \ = \ [M^\circ \, , N^\circ]$
2. Complete a seguinte dedução de que a função $\overset{+}{\bowtie}$ é a conversa da função $\triangle_+$ (sendo portanto ambas isomorfismos):

$$
\begin{array}{cl}
& X \overset{+}{\bowtie} (M, N) \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& X^\circ = [M^\circ \, , N^\circ] \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& M^\circ = X^\circ \cdot i_1 \ \wedge \ N^\circ = X^\circ \cdot i_2 \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& M = i_1^\circ \cdot X \ \wedge \ N = i_2^\circ \cdot X \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& (M, N) = (i_1^\circ \cdot X, i_2^\circ \cdot X) \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& (M, N) = \triangle_+ X \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& X \triangle_+^\circ (M, N) \\
:: & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \overset{+}{\bowtie} = \triangle_+^\circ
\end{array}
$$

Questão 5 O artigo `http://en.wikipedia.org/wiki/Euclidean_algorithm` apresenta duas versões do famoso algoritmo de Euclides para cálculo do *máximo divisor comum* que, escritas em VDM, ficam como se segue,

```
gcd : nat * nat -> nat
gcd(a, b) ==
    if b = 0
       then a
       else gcd(b, a mod b);
```

```
gcd' : nat * nat ==> nat
gcd'(a, b) ==
    (dcl x : nat := a,
         y : nat := b;
         while (y <> 0) do
            let z = y
            in (y := x mod y ;
                x := z);
     return x;
    );
```

sendo gcd' apresentado como implementação (mais eficiente) de gcd.

O cálculo que se segue confirma essa relação entre os dois algoritmos. Complete-o:

$$
\begin{array}{cl}
& gcd \vdash gcd' \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& gcd' \ = \ \langle \mu \, gcd \ :: \ (=0) \cdot \pi_2 \rightarrow \pi_1 \ , \ gcd \cdot \langle \pi_2, mod \rangle \rangle \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \pi_1 \cdot (\underline{while} \ (\neg \cdot (=0) \cdot \pi_2) \ \underline{do} \ \langle \pi_2, mod \rangle) \ = \ \langle \mu \, gcd \ :: \ [\pi_1 \, , gcd \cdot \langle \pi_2, mod \rangle] \cdot ((=0) \cdot \pi_2)? \rangle
\end{array}
$$

$\equiv$ { ..... }

$\pi_1 \cdot [\![[id\ , id], (id + \langle \pi_2, mod \rangle) \cdot ((= 0) \cdot \pi_2)?]\!] = [\![[\pi_1\ , id], (id + \langle \pi_2, mod \rangle) \cdot ((= 0) \cdot \pi_2)?]\!]$

$\Leftarrow$ { ..... }

.....

$\equiv$ { ..... }

.....

$\equiv$ { ..... }

TRUE

Anexo–Algumas leis de cálculo que podem ser úteis

Refinamento algorítmico:

$$S \vdash R \quad \equiv \quad (\delta\, S \subseteq \delta\, R) \ \wedge \ (R \cdot \delta\, S \subseteq S) \tag{9}$$

Refinamento de dados:

$$\begin{array}{rcll} A \rightharpoonup 1 & \cong & \mathcal{P} A & (10) \\ A \to B \times C & \leq & (A \to B) \times (A \to C) & (11) \\ A \to B & \leq & A \rightharpoonup \mathcal{P} B & (12) \\ (B \times C) \rightharpoonup A & \leq & B \rightharpoonup (C \rightharpoonup A) & (13) \\ A \rightharpoonup (D \times (B \rightharpoonup C)) & \leq & (A \rightharpoonup D) \times ((A \times B) \rightharpoonup C) & (14) \\ \mu \mathsf{F} & \leq & (K \rightharpoonup \mathsf{F}\, K) \times K & (15) \end{array}$$

Hilomorfismos:

$$[\![R, S]\!]^\circ = [\![S^\circ, R^\circ]\!] \tag{16}$$

$$V \cdot [\![S, R]\!] = [\![T, R]\!] \quad \Leftarrow \quad V \cdot S = T \cdot (\mathsf{F}\, V) \tag{17}$$

$$[\![S, R]\!] \cdot V = [\![S, U]\!] \quad \Leftarrow \quad R \cdot V = \mathsf{F}\, V \cdot U \tag{18}$$

$$[\![R, S]\!] \subseteq T \quad \Leftarrow \quad R \cdot \mathsf{F}\, T \cdot S \subseteq T \tag{19}$$

Factorização iterativa: para θ associativa, tem-se

$$\begin{array}{rcl} \langle \mu\ f\ ::\ p \to b\ ,\ \theta \cdot \langle d, f \cdot e \rangle \rangle & = & p \to b\ ,\ \theta \cdot (id \times b) \cdot w \cdot \langle d, e \rangle \\ & & \textit{onde} \\ & & w = \underline{\textit{while}}\ (\neg \cdot p \cdot \pi_2)\ \underline{\textit{do}}\ \langle \theta \cdot (id \times d), e \cdot \pi_2 \rangle \end{array} \tag{20}$$

Sendo (θ, u) um monoide, tem-se

$$\langle \mu\ f\ ::\ p \to \underline{u}\ ,\ \theta \cdot \langle d, f \cdot e \rangle \rangle \quad = \quad \pi_1 \cdot w \cdot \langle \underline{u}, id \rangle \tag{21}$$

onde w é o mesmo que em (20).