Métodos Formais em Engenharia de Software

1.º Ano de Mestrado de Informática da Universidade do Minho
Ano Lectivo de 2007/08

Exame de recurso — Módulos MF e CSI — 24 de Julho 2008
09h00 - 13h00
Sala 1.08

NB: *Esta prova consta de* ***6+6*** *alíneas todas com as mesma cotação.*

PROVA COM CONSULTA (4 horas no máximo)

GRUPO MF

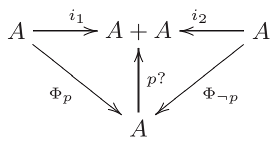
Questão 1 Há duas formas de definir o combinador condicional de McCarthy:

$$p \rightarrow R\,,S \quad = \quad R \cdot \Phi_p \cup S \cdot \Phi_{\neg p} \tag{1}$$

e

$$p \rightarrow R\,,S \quad = \quad [R\,,S] \cdot (p?) \tag{2}$$

onde $(p?)^\circ = [\Phi_p\;,\Phi_{\neg p}]$, *cf.* diagrama



Mostre, recorrendo à igualdade

$$[R\,,S] \cdot [T\,,U]^\circ \quad = \quad (R \cdot T^\circ) \cup (S \cdot U^\circ) \tag{3}$$

que essas duas definições são equivalentes.

Questão 2 Recorde a equivalência

$$R \times S \subseteq U \times V \quad \equiv \quad R \subseteq U \ \wedge\ S \subseteq V \tag{4}$$

e complete justificações e passos do seguinte cálculo de (4):

$$
\begin{array}{ll}
& R \times S \subseteq U \times V \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& R \times S \subseteq \langle U \cdot \pi_1, V \cdot \pi_2 \rangle \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \pi_1 \cdot (R \times S) \subseteq U \cdot \pi_1 \ \wedge\ \pi_2 \cdot (R \times S) \subseteq V \cdot \pi_2 \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \vdots \\
\equiv & \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \} \\
& \text{TRUE}
\end{array}
$$

Questão 3 Uma das estruturas mais úteis em computação para garantir comunicação entre processos é a de um *buffer*, tal como a seguir se modela em notação VDM:

```
Buffer :: buffer: seq of token
          maxsize : nat
          inv b == len b.buffer <= b.maxsize;

Output = Ok | <full> | <empty>;

Ok :: x:token;
```

Complete as pré e pós-condições que se seguem

```
BufferInit() r:Buffer
pre true
post r=mk_Buffer([],1000);

BufferIn(b:Buffer,x:token) r: Buffer*Output
pre ..........................................................................
post .........................................................................
     .........................................................................

BufferOut(b:Buffer) r: Buffer*Output
pre ..........................................................................
post .........................................................................
     .........................................................................
```

por forma a modelar as operações de chegada e saída de dados num *buffer*, respectivamente.

Questão 4 Recorde que uma dada especificação

$$Spec : (b : B) \leftarrow (a : A)$$
pre ...
post ...

se diz *satisfiável* se e só se, para toda a entrada prevista na sua pré-condição, existe (pelo menos uma) uma saída prevista pela sua pós-condição, isto é,

$$\langle \forall\ a\ :\ a \in A :\ \text{pre-}Spec\ a\ \Rightarrow\ \langle \exists\ b\ :\ b \in B :\ \text{post-}Spec(b, a)\rangle\rangle \tag{5}$$

ou, em notação-PF,

$$\Phi_{(\in A)} \cdot \Phi_{pre}\ \subseteq\ \top \cdot \Phi_{(\in B)} \cdot R_{post} \tag{6}$$

onde pre abrevia pre-$Spec$ e $b(R_{post})a \equiv$ post-$Spec(b, a)$.

Mostre que, no caso de não haver invariantes nos tipos A e B, as duas condições acima são equivalentes a

$$\Phi_{pre}\quad \preceq R_{post} \tag{7}$$

Daqui deduza que, nessas condições, se a pré-condição de uma especificação for TRUE, então a pós-condição é tal que R_{post} é inteira.

Questão 5 Neste módulo foi estudado um fragmento da especificação de um sistema de ficheiros de acordo com os requisitos que constam do *Intel ® Flash File System Core Reference Guide* (versão 1, Out. 2004).

Recorde que o sistema de ficheiros é modelado por um par de 'mappings', um dos quais (de tipo $FStore$) regista o conteúdo e informação associada a cada ficheiro/directoria, dado o seu *'path'*:

$$FStore = Path \rightharpoonup File$$
$$\textbf{inv}\ N \triangleq \langle \forall\, p\ :\ p \in dom\ N :\ dirName(p) \in dom\ N\ \wedge$$
$$fileType(attributes(N(dirName\, p))) = Directory \rangle$$

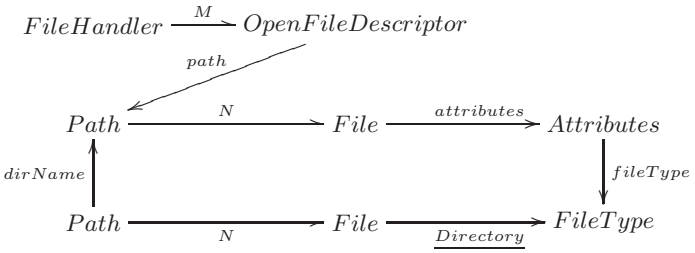
A função $dirName$ (omitida no fragmento apresentado) indica, de acordo com os requisitos, a directoria a que pertence um dado ficheiro ou directoria. Quanto aos outros tipos de dados, sabe-se o seguinte:

$$File = \{attributes : Attributes, contents : Data\}$$
$$Attributes = \{fileType : Type, ...\}$$

1. Mostre que a transformada-PF do invariante de $FStore$ é

$$\text{inv-}FStore\ N \quad \triangleq \quad \underline{Directory} \cdot N \subseteq fileType \cdot attributes \cdot N \cdot dirName \tag{8}$$

   cf. diagrama:



   Sugestão: recorde que, para N simples, $b\ N\ a$ tem o mesmo valor lógico que

$$a \in dom\ N\ \wedge\ b = N\ a \tag{9}$$

2. Especifique a operação que adiciona um ficheiro $x \in File$ a a um $FStore\ N$, dado o seu $Path\ p$, tendo o cuidado de incluir uma pré-condição que garanta que o invariante (8) é preservado. **Sugestão:** empregue uma notação que tenha estudado na UCE, à sua escolha.

GRUPO CSI

***NB:** Neste grupo, resolva 6 das 8 alíneas que se seguem, à sua escolha.*

Questão 6 Derive o teorema grátis da função polimórfica seguinte, escrita em notação VDM

```
concatMap[@A,@B] : (@A -> seq of @B) -> seq of @A -> seq of @B
concatMap(f)(l) == conc [f(l(i)) | i in set inds l];
```

e instancie-o para o caso em que todas as relações em jogo são funções.

Questão 7 Recorde a relação de satisfação entre especificações e funções:

$$S \vdash f \quad \equiv \quad f \cdot \delta\, S \subseteq S \tag{10}$$

1. Mostre que $S \vdash max$ se verifica, onde max é a função

$$max : \mathbb{Z} \leftarrow (\mathbb{Z} \times \mathbb{Z})$$
$$max(i,j) \triangleq \textbf{ if } i \leq j \textbf{ then } j \textbf{ else } i$$

   e S a especificação

$$S : (r : \mathbb{Z}) \leftarrow (i : \mathbb{Z}, j : \mathbb{Z})$$
$$\textbf{post } (r = i \vee r = j) \wedge i \leq r \wedge j \leq r$$

   Sugestão: instancie (10), introduza variáveis e simplifique.

2. Mostre que a transformada-PF de $\mathbb{Z} \xleftarrow{S} \mathbb{Z} \times \mathbb{Z}$ é

$$S \quad = \quad \in \cap \langle \leq, \leq \rangle^{\circ} \tag{11}$$

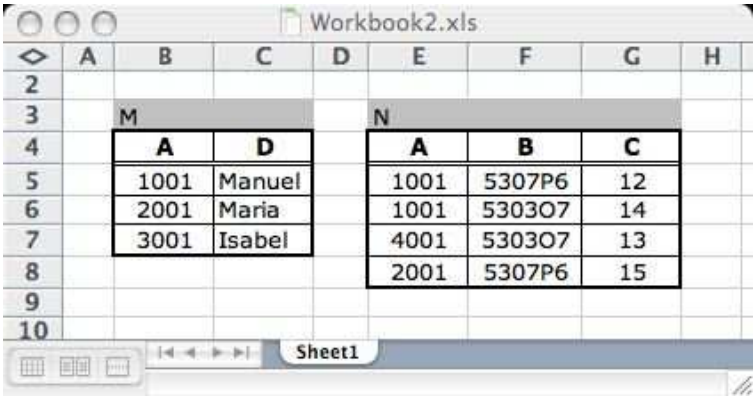
Questão 8 É conhecido o papel da lei

$$A \rightharpoonup (D \times (B \rightharpoonup C)) \quad \underset{\bowtie_n}{\overset{\triangle_n}{\leq}} \quad (A \rightharpoonup D) \times (A \times B \rightharpoonup C) \tag{12}$$

na decomposição de modelos de dados com vista à sua implementação em RDBMS. Escrita em notação VDM, (12) fica

$$\texttt{map}\ A\ \texttt{to}\ (D * (\texttt{map}\ B\ \texttt{to}\ C)) \quad \underset{njoin}{\overset{unnjoin}{\leq}} \quad (\texttt{map}\ A\ \texttt{to}\ D) * (\texttt{map}\ A * B\ \texttt{to}\ C) \tag{13}$$

É fácil ver que o par (M, N) que abaixo se ilustra,



A	D
1001	Manuel
2001	Maria
3001	Isabel

A	B	C
1001	5307P6	12
1001	5303O7	14
4001	5303O7	13
2001	5307P6	15

nunca poderia ter sido gerado por $unnjoin$. Diga porquê, identificando a propriedade de $unnjoin$ que este contra-exemplo evidencia. Escreva ainda, em notação *pointfree* ou *pointwise*, a propriedade a que o par (M, N) deveria obedecer para poder ter sido gerado por $unnjoin$.

Questão 9 Uma fábrica de cabos condutores eléctricos pretende uma aplicação para controlo dos seus 'stocks'. Há vários tipos de cabos, cada um identificado por um *código de artigo*. Cada segmento de cabo constitui uma *ponta*, é de um determinado tipo, tem um dado comprimento (múltiplo de $1m$) e está armazenado algures numa *bobine*. Cada *bobine* tem uma *matrícula* que a identifica e está num determinado *estado*, a saber: armazenada num dado *armazém*, em manutenção, em produção, em controlo de qualidade, no cliente, abatida ou morta. Na mesma bobine só podem estar armazenadas pontas do mesmo tipo. Todas as bobines do mesmo tipo estão armazenadas no mesmo armazém.

Partindo do seguinte esboço de modelo em VDM para este problema,

```
Bobines = map Matr to (seq of Ponta * Estado);
Ponta = CArt * Compr ;
Estado = CArm | <manutencao> | <producao> | <controlo> | <cliente> | <abatida> |
         <morta> ;
Matr = token ;
CArt = token ;
CArm = token ;
Compr = nat ;
```

use as leis de refinamento de dados estudadas nesta UCE para derivar uma sua versão relacional indicando, para cada passo, a lei utilizada.

Desenhe ainda um diagrama relacional que represente a implementação a que chegou.

Questão 10 Dada uma relação binária $B \xleftarrow{R} A$ e duas funções f, g tal como no diagrama que se segue,

$$\begin{array}{ccc} B & \xleftarrow{R} & A \\ {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle f} \\ C & \xleftarrow[\pi_{g,f}\,R]{} & D \end{array}$$

diz-se que R satisfaz a *dependência funcional* $f \rightharpoonup g$ se e só se

$$\mathit{ker}\,(f \cdot R^{\circ}) \subseteq \mathit{ker}\,g \tag{14}$$

se verifica. (Este conceito estende aquele que concerteza conhece das bases de dados.)

1. Mostre que (14) é equivalente a dizer-se que a projecção $\pi_{g,f} R$ do diagrama é simples.
2. É vulgar representarem-se *arrays* associativos (isto é, estruturas de tipo `map A to B`) sob a forma de listas de pares de tipo `seq of (A*B)`, tendo em conta o refinamento

$$A \rightharpoonup B \underset{f \cdot \Phi_i}{\overset{R}{\leq}} (B \times A)^\star \tag{15}$$

onde

$$f\ L \stackrel{\mathrm{def}}{=} \{\pi_2(L\ i) \mapsto \pi_1(L\ i) \mid i \in\ inds\ L\} \tag{16}$$

$$i\ L \stackrel{\mathrm{def}}{=} \langle \forall\ i, j\ :\ i, j \in\ inds\ L :\ \pi_2(L\ i) = \pi_2(L\ j)\ \Rightarrow\ \pi_1(L\ i) = \pi_1(L\ j) \rangle \tag{17}$$

Vendo L como uma relação simples de tipo $B \times A \xleftarrow{L} I\!N$ que indica que elementos na lista ocupam que posições, e recorrendo a (14), mostre que $i\ L\ \equiv\ \pi_2 \overset{\rho\,L}{\rightharpoonup} \pi_1$.
3. Complete a seguinte dedução da pre-condição $i\ L$ obtida calculando a WP que garante que f dá apenas resultados simples:

$i\ L \quad \equiv \quad f\ L$ is simple

$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$

$\mathit{img}\,(\pi_1 \cdot \rho\,L \cdot \pi_2^\circ) \subseteq id$

$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$

$\pi_1 \cdot \rho\,L \cdot \pi_2^\circ \cdot \pi_2 \cdot \rho\,L \cdot \pi_1^\circ \subseteq id$

$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$

$\rho\,L \cdot \pi_2^\circ \cdot \pi_2 \cdot \rho\,L \subseteq \mathit{ker}\,\pi_1$

$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$

$\mathit{ker}\,(\pi_2 \cdot \rho\,L) \subseteq \mathit{ker}\,\pi_1$

$\equiv \quad \{ \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad \}$

$\pi_2 \overset{\rho\,L}{\rightharpoonup} \pi_1$