Cálculo de Sistemas de Informação
Perfil: MÉTODOS FORMAIS DE PROGRAMAÇÃO

1.º Ano de MEI e MMC, Universidade do Minho
Ano Lectivo de 2022/23

Exame de recurso — 22 de Junho 2023
09h00–11h00, Sala E1-2.24

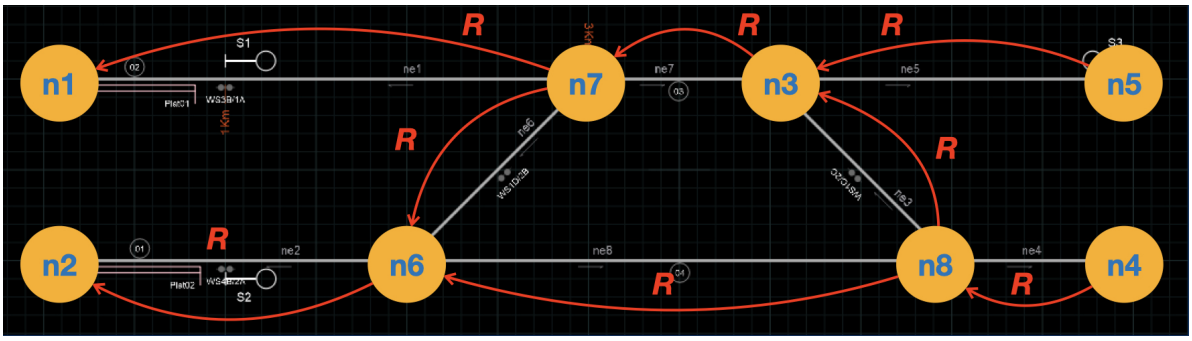
---

*Esta prova consta de **8** questões que valem, cada uma, 2.5 valores. O tempo **médio** estimado para resolução de cada questão é de 15 min.*

Recomenda-se que os alunos leiam a prova antes de decidirem por que ordem querem responder às questões que são colocadas.

PROVA COM CONSULTA (2 horas)

Questão 1 Recorde a modelação relacional de uma pequena rede ferroviária que foi abordada nas aulas:



Caracterize R quanto a ser:

- inteira
- irreflexiva
- injectiva
- antissimétrica

Justifique convenientemente a sua resposta.

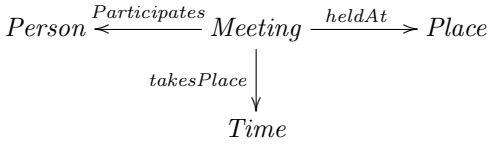
Questão 2 Considere os requisitos:

> *About a group meeting we know (a) the people* who *participate in the meeting; (b)* when *the meeting takes place; (c)* where *the meeting takes place. Please give me an Alloy model capable of recording the following property: nobody can be at the same time in two meetings held in different places.*

Desafiado com esses requisitos, o ChatGPT fez a seguinte proposta:

```
sig Person {
  participates: set Meeting
}

sig Meeting {
  heldAt: Place,
  takesPlace: Time
}

sig Place {}

sig Time {}

fact NoOverlappingMeetings {
  all p: Person, m1, m2: Meeting | m1 != m2 =>
    (m1.takesPlace = m2.takesPlace && p.participates.m1 && p.participates.m2)
        implies m1.heldAt = m2.heldAt
}

run {} for 4
```

O Alloy proposto lida com as três relações do seguinte diagrama:



Comente a qualidade do modelo proposto e, independentemente da sua validade, mostre que o mesmo problema pode ser expresso relacionalmente de forma bastante mais succinta usando conceitos de álgebra relacional estudados nesta disciplina. (Justifique a sua resposta.)

Sugestão: use a ordem que compara relações quanto à sua injectividade.

Questão 3 Considere o seguinte operador relacional

$$R \ominus S = R \cap \bot \,/\, S^{\circ} \tag{F1}$$

que restringe a relação R às suas entradas onde S não está definida:

$$b\ (R \ominus S)\ a \ \Leftrightarrow\ \left\{ \begin{array}{l} b\ R\ a \\ \langle \forall\ x\ ::\ \neg\ (x\ S\ a) \rangle \end{array} \right.$$

Mostre que esta operação é permutativa, isto é:

$$(R \ominus S) \ominus Q = (R \ominus Q) \ominus S \tag{F2}$$

Sugestão: use igualdade indirecta.

Questão 4 Um modelo habitual para guardar informação que foi assunto de um teste anterior desta disciplina baseia-se em pares chave (Key) / valor ($Data$) — isto é, relações *simples* $S : Key \rightarrow Data$. Em memórias de estado sólido, vulgarmente designadas por memórias *flash*, esse tipo de informação toma a forma

$$Addr \times Key \rightarrow Data + 1 \tag{F3}$$

onde $Addr$ é o espaço de endereçamento disponível e $Data + 1$ regista informação guardada *ou* apagada.

Flash Store

1	KEY4	DATA
2	KEY2	DATA
3	KEY1	DATA
4	KEY1a	DATA
5	KEY3	DATA
6	KEY3	DATA
7	KEY1	DATA
8	KEY1b	DATA
9	KEY4	DEL

Evita-se apagar ou (re)escrever nas mesmas células pois isso acaba por danificá-las. Em vez disso, vão-se usando endereços seguintes livres para registar a informação actualizada.

Para se extrair de uma memória flash $M : Addr \times Key \rightarrow Data + 1$ a relação $S : Key \rightarrow Data$ contendo a informação mais recente, alguém propôs

$$
\begin{array}{l}
S = R \upharpoonright Q \ \mathbf{where} \\
\quad R = \langle \pi_1, i_1^\circ \cdot M \rangle \cdot \pi_2^\circ \\
\quad Q = ...
\end{array}
$$

construída sobre o diagrama:

$$
\begin{array}{ccccc}
Addr & \xleftarrow{\pi_1} & Addr \times Key & \xrightarrow{\pi_2} & Key \\
 & & \downarrow{\scriptstyle M} & & \\
Data & \xrightarrow[i_1]{} & Data + 1 & \xleftarrow[i_2]{} & 1
\end{array}
$$

Identifique o tipo de R e faça a sua proposta para Q, justificando a sua resposta.

Questão 5 O domínio $\delta\ R$ de uma relação R, que é a maior coreflexiva incluída no seu núcleo ($\ker\ R$), satisfaz a propriedade

$$\delta\ \langle f \cdot R, \underline{k} \rangle = \delta\ R \qquad \text{(F4)}$$

Apresente as justificações que faltam na seguinte prova de (F4) por igualdade indirecta::

$$
\begin{array}{ll}
 & \delta\ \langle f \cdot R, \underline{k} \rangle = \delta\ R \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & \delta\ (f \cdot R) = \delta\ R \\
:: & \quad \{ \text{ igualdade indirecta } \} \\
 & \delta\ (f \cdot R) \ \subseteq\ \Phi \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & f \cdot R \ \subseteq\ \top \cdot \Phi \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & R \ \subseteq\ f^\circ \cdot (!)^\circ \cdot ! \cdot \Phi \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & R \ \subseteq\ \top \cdot \Phi \\
\equiv & \quad \{ \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \ \} \\
 & \delta\ R \ \subseteq\ \Phi \\
\Box &
\end{array}
$$

Questão 6 Os teoremas grátis de funções paramétricas sobre listas acabam sempre por precisar do *relator* R^*. Pode mostrar-se que este *relator* é caracterizado por duas propriedades, a saber

$$R^* \cdot \mathsf{nil} = \mathsf{nil} \qquad \text{(F5)}$$

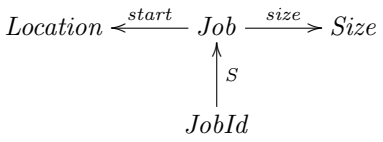
$$R^* \cdot \mathsf{cons} = \mathsf{cons} \cdot (R \times R^*) \qquad \text{(F6)}$$

onde $\mathsf{nil}\ \_ = [\,]$ e $\mathsf{cons}\ (h, t) = h : t$ são funções sobre listas bem conhecidas. Verifique quais das seguintes afirmações são verdadeiras, justificando formalmente:

$$\bot^* = \bot \tag{F7}$$

$$y\ (\bot^*)\ [\,] \ \Leftrightarrow\ y = [\,] \tag{F8}$$

Questão 7 Recorde das aulas o seguinte modelo simplificado do módulo de gestão de memória do "kernel" de um sistema operativo,



onde

- $JobId$ identifica cada processo (Job) em execução, de forma única;
- $start$ dá o endereço onde começa o bloco de memória reservado a cada processo em execução;
- $size$ dá o tamanho desse bloco de memória (contígua);
- S(cheduled) é a relação simples que associa $JobId$s a Jobs;
- $Location$ e $Size$ são números naturais.

A relação

$$k\ (Owns\_by\ S)\ a\ \Leftrightarrow\ \langle \exists\ x\ :\ x\ S\ k :\ start\ x \leqslant a \leqslant end\ x \rangle\ \mathbf{where}\ end\ x = start\ x + size\ x$$

sinaliza a "posse" (ou não) de um endereço a por parte de um processo k, de acordo com o *scheduler* S . Mostre que $Owns\_by$ se pode definir como se segue:

$$Owns\_by\ S = (\langle start, end \rangle \cdot S)^\circ \cdot \langle (\leqslant), (\geqslant) \rangle \tag{F9}$$

Questão 8 Na sequência da questão anterior, considere o invariante

$$inv\ S\ \Leftrightarrow\ \ker\ (Owns\_by\ S)\ \subseteq\ id \tag{F10}$$

que garante que, no *scheduler* S, nenhum processo executa em células de memória de outros processos.

Mostre que a operação

$$free\ i\ S = S - \top \cdot \underline{i}^\circ \tag{F11}$$

que retira do *scheduler* S o *job id* i de um processo que já terminou não precisa de qualquer pé-condição para garantir o invariante (F10).
