Semântica das Linguagens de Programação

Exame (15 de Julho de 2020, 2:30) / Duração: 3:00

Responda às questões do exame em folhas manuscritas. Coloque o seu nome e número no topo de cada folha. No final digitalize ou fotografe as folhas e faça o upload das suas respostas no Blackboard, dentro do prazo de duração do exame.
Cotação Questão 1: 1.5, 1, 1.5, 1.5, 1.5. Questão 2: 1, 2. Questão 3: 2, 1. Questão 4: 2, 2. Questão 5: 2, 1.

Questão 1 Sejam $C, P \in \mathbf{Stm}$ os seguintes programas:

$$C \equiv \texttt{if } y + x \leq 7 \texttt{ then } \{\, x := x + y \,;\, y := 2 * x \,\} \texttt{ else skip}$$

$$P \equiv \{x := 3 \,;\, y := z + y \,\} \,;\, C$$

1. Recorrendo à semântica de transições (*small-step*) simule a execução do programa P a partir do estado inicial s, em que $s\, z = 5$ e $s\, y = 10$. Apresente as árvores de prova para as 2 primeiras transições apenas.

2. Indique o resultado da tradução do programa C código da máquina abstracta AM.

3. Sendo s' o estado em que $s'\, z = -5$ e $s'\, y = 4$, apresente a árvore de derivação do juizo de avaliação *big-step*: $\langle P, s' \rangle \to s'[x \mapsto 2, y \mapsto 4]$.

4. Apresente a interpretação denotacional do programa P.

5. Apresente uma árvore de prova para o seguinte triplo de Hoare: $\{z + y > 4\}\, P\, \{y > 4\}$

Questão 2 Pretende-se estender a linguagem **While**, acrescentando-lhe uma nova forma de ciclo de acordo com a seguinte sintaxe abstracta:

$$\mathbf{Stm} \ni C ::= \ldots \mid \texttt{do } C_1 \texttt{ while } b$$

A descrição informal da semântica deste comando é a seguinte:

> *O comando C_1 é executado repetidamente enquanto o valor da expressão b for verdadeiro, sendo o teste feito depois da execução do comando.*

1. Especifique formalmente o comportamento deste novo ciclo, escrevendo regras apropriadas para a *semântica natural* da linguagem. As regras não devem fazer referência a outros ciclos.

2. Uma propriedade expectável á a equivalência entre os dois comandos seguintes:

$$\texttt{do } C \texttt{ while } b \qquad \text{e} \qquad C \,;\, \texttt{while } b \texttt{ do } C$$

Escreva formalmente essa propriedade e, tendo em conta as regras que propôs, demonstre uma das implicacõess (à sua escolha) envolvidas na prova dessa equivalência.

Questão 3 Considere os seguintes termos do lambda calculus puro:

$$\begin{array}{lcllcl} I & \equiv & (\lambda x.\, x) & K & \equiv & (\lambda a.\lambda b.\, a) \\ A & \equiv & (\lambda a.\lambda b.\, b\, a) & S & \equiv & (\lambda x.\lambda y.\lambda z.\, x\, z\, (y\, z)) \end{array}$$

1. Apresente a sequência correspondente à *ordem normal* de redução da espressão

$$K\, S\, (A\, K\, I)\, I\, K$$

até à sua *forma normal*. Sublinhe o β-redex que é seleccionado em cada passo de redução.

2. Considere a expressão $K\, S$. Coloque anotações de tipo nas variáveis que estão a ser abstraídas de forma a que esta expressão seja tipificável, e indique qual o tipo da expressão. Não precisa de apresentar a prova formal do juizo de tipificação.

Questão 4 Considere a seguinte expressão E da linguagem de programação funcional estudada:

$$\begin{array}{l} \mathsf{let}\ \ \mathsf{fun} \equiv \lambda l1.\lambda l2.\, \mathsf{listcase}\ l1\ \mathsf{of}\ (l2, \lambda h.\lambda t.\, h :: l2) \\ \mathsf{in}\ \ \ \ \mathsf{fun}\, (5 :: (9+3) :: \mathsf{nil})\, ((\lambda x.\, x)\, \mathsf{nil}) \end{array}$$

1. Utilizando a semântica de avaliação *call-by-value*, calcule, passo a passo, o valor da espressão E.

2. Construa uma árvore de prova do juízo

$$a : \mathsf{List\,Int} \ \vdash\ (\mathsf{let\ fun} \equiv \lambda l1.\lambda l2.\, \mathsf{listcase}\ l1\ \mathsf{of}\ (l2, \lambda h.\lambda t.\, h :: l2)\ \mathsf{in\ fun}\, a\, \mathsf{nil}) : \mathsf{List\,Int}$$

Questão 5 Pretende-se estender a linguagem de programação funcional, com um novo tipo de dados para representar árvores binárias polimórficas, ou seja, o equivalente ao seguinte tipo de dados do Haskell:

```
data BTree a = Empty | Node a (BTree a) (BTree a)
```

1. Defina a sintaxe abstracta das novas expressões e do novo tipo, e as regras de inferência de tipo para as novas expressões. Defina também as novas formas canónicas da linguagem e as novas regras de avaliação *call-by-name*.

2. Apresente uma expressão que codifique uma árvore binária infinita, que tem em cada nodo o número inteiro correspomdente à profundidade a que o nodo se encontra (isto é, a sua distância à raiz). Ou seja, na raiz tem 0, no nível seguinte tem 1, etc. Deve definir também todas as funções auxiliares que utilizar.