

Semânticas de Linguagens de Programação - Exercícios 1

Grafos de programas e semântica simples para a linguagem de comandos guardados GC

Nos exercícios seguintes deves aceder ao simulador do Formal Methods Learning Environment (Step-wise Execution):

<https://formalmethods.dk/fm4fun/>.

1. Considera o exemplo do *maximo*.
 - (a) Constrói grafo de programa usando a função $trans(q \rightsquigarrow q')\llbracket c \rrbracket$ e compara com o obtido no simulador.
 - (b) Considerando $s(x) = 3$ e $s(y) = 5$ calcula a sequência de configurações $\langle q_0, s \rangle \xRightarrow{\omega'}^* \langle q_f, s' \rangle$. Compara com a execução passo a passo no simulador.
 - (c) Testa no simulador o programa *factorial* e compara com o que foi dado da aula teórica. Executa passo a passo para os mesmos valores de x e y .
2. Para cada uma das funções abaixo:
 - constrói programas em GC
 - determina os grafos de programa
 - obtêm sequências de execução para a memória inicial s com $s(n) = 10$ e $s(m) = 6$.
 - escreve os programas em ficheiros que testa as alíneas anteriores no simulador.
 - (a) o modulo $\text{mod}(n, m)$ que retorna o resto da divisão inteira de n por m .
 - (b) o máximo divisor comum $\text{gdc}(n, m)$ de dois números n e m .
 - (c) o n -ésimo número de Fibonacci.
3. (a) Considerando que o comando condicional **if** b **then** c_1 **else** c_2 é equivalente ao comando guardado **if** $b \rightarrow c_1 \llbracket \neg b \rightarrow c_2 \text{ fi}$, estende a função $trans(q \rightsquigarrow q')\llbracket \cdot \rrbracket$ a esse comando.
- (b) Considera o seguinte comando **ord**

if $x > y$ **then** $z := x$ **else** $z := y$

e $s_0 : \mathbf{Var} \rightarrow \mathbb{Z}$ com $s_0(x) = 3$ e $s_0(y) = 5$.

- i. Desenha o grafo de programa $PG = (Q, q_0, q_f, Act, E)$ para **ord**. Nota: no simulador usa o comando guardado equivalente.
- ii. Usando o grafo de programa PG **ord**, calcula s_f tal que

$$\langle q_0, s_0 \rangle \xRightarrow{\omega'}^* \langle q_f, s_f \rangle.$$

4. (a) Considerando que o comando de ciclo **while** b **do** c é equivalente ao comando guardado **do** $b \rightarrow c$ **od**, determina o valor de $trans(q \rightsquigarrow q')\llbracket \text{while } b \text{ do } c \rrbracket$, para quaisquer $q, q' \in Q$.
- (b) Considera o seguinte comando **sum**

$w := 0; z := 1; \text{do } z \leq x \rightarrow (w := w + z; z := z + 1) \text{ od}$

e $s_0 : \mathbf{Var} \rightarrow \mathbb{Z}$ com $s_0(x) = 2$.

- i. Desenha o grafo de programa $PG = (Q, q_0, q_f, Act, E)$ para **sum**.
 - ii. Usando o grafo de programa PG **sum**, calcula s_f tal que $\langle q_0, s_0 \rangle \xRightarrow{\omega'}^* \langle q_f, s_f \rangle$, representando as memórias intermédias numa tabela.
5. Podemos adicionar arrays às expressões aritméticas. Seja **Arr** um conjunto de arrays e se $A \in \mathbf{Arr}$ então $length(A)$ é o seu tamanho. Acrescentamos às expressões a a expressão $A[a]$ e aos comandos c atribuições $A[a_1] := a_2$. Temos

$$\mathbf{Mem} = (\mathbf{Var} \cup \{A[i] \mid A \in \mathbf{Arr}, 0 \leq i \leq length(A)\}) \rightarrow \mathbb{Z}$$

e

$$\mathcal{S}[A[a_1] := a_2] s = \begin{cases} s[A[z_1] \mapsto z_2] & \text{se } z_1 = \mathcal{A}[a_1] s, z_2 = \mathcal{A}[a_2] s \text{ e } A[z_1] \in dom(s) \\ \textit{indefinido} & \text{caso contrário.} \end{cases}$$

- (a) Considera o exemplo *Insertion Sort* no simulador, analisa o grafo de programa e segue a execução para os seguintes valores $A[] = [3, 1, 5, 4]$.
- (b) Escreve um programa em GC para *pesquisa linear* de um valor x num array A . Testa no simulador, analisa o grafo de programa e segue a execução para os seguintes valores $A[] = [3, 1, 5, 4]$ e $x = 5$.
- (c) Compara o teu programa com o exemplo no simulador que usa construtores **try** e **catch**.
- (d) Escreve um programa em GC que dado um array de inteiros determine o maior e repete os passos o Exercício a.
- (e) Escreve um programa em GC para o *Bubble sort* de um array A e repete os passos o Exercício a.
- (f) Escreve um programa em GC para a pesquisa binária num A e repete os passos o Exercício b.
- (g) Escreve um programa em GC que dado um array de inteiros determine quantos são pares e quantos são ímpares. Repete os passos o Exercício a.