

Semânticas de Linguagens de Programação - Exercícios 3
Semântica operacional *small-step* para a linguagem **While**

Semântica operacional estrutural (*small-step*) da linguagem **While**

1. Considera a semântica operacional estrutural dada no curso.

(a) Sendo $s_0 = [x = 3, y = 5]$, determina o estado após a execução do comando

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

(b) Sendo $s_0 = [x = 3]$, determinar o estado após a execução de:

$y := 1$; **while** $\neg(x = 1)$ **do** ($y := y \times x$; $x := x - 1$)

(c) Sendo $s_0 = [x = 10, y = 5]$, determinar o estado após a execução de:

$z := 0$; **while** $y \leq x$ **do** ($z := z + 1$; $x := x - y$)

2. Define uma semântica operacional estrutural para o comando **repeat** c **until** b (sem usar a **do while**).
3. Define uma semântica operacional estrutural para o comando **for** $x := a_1$ **to** a_2 **do** c (sem usar a **do while**).
4. Mostra que para todo o $k \geq 1$, se $\langle c_1, s \rangle \Rightarrow^k s'$ então $\langle c_1; c_2, s \rangle \Rightarrow^k \langle c_2, s' \rangle$.
5. Mostra que a semântica operacional estrutural dada para a linguagem **While** é determinística.
6. Mostra a equivalência semântica dos seguintes comandos:
- (a) **skip**; c e c
- (b) **while** b **do** c e **if** b **then** (c ; **while** b **do** c) **else** **skip**
7. Implementa em Haskell a semântica operacional estrutural para a linguagem **While**, usando as estruturas de dados definidas no exercício dado na prática 2.

Semântica operacionais para While

Semântica operacional natural (*big-step*)

$$\begin{array}{ll}
\text{att}_{sn} & \langle x := a, s \rangle \longrightarrow s[x \mapsto \mathcal{A}[[a]] s] \\
\text{skip}_{sn} & \langle \text{skip}, s \rangle \longrightarrow s \\
\text{comp}_{sn} & \frac{\langle c_1, s \rangle \longrightarrow s', \langle c_2, s' \rangle \longrightarrow s''}{\langle c_1; c_2, s \rangle \longrightarrow s''} \\
\text{if}^v_{sn} & \frac{\langle c_1, s \rangle \longrightarrow s'}{\langle \text{if } b \text{ then } c_1 \text{ else } c_2, s \rangle \longrightarrow s'} \text{ se } \mathcal{B}[[b]] s = \text{true} \\
\text{if}^f_{sn} & \frac{\langle c_2, s \rangle \longrightarrow s'}{\langle \text{if } b \text{ then } c_1 \text{ else } c_2, s \rangle \longrightarrow s'} \text{ se } \mathcal{B}[[b]] s = \text{false} \\
\text{while}^v_{sn} & \frac{\langle c, s \rangle \longrightarrow s', \langle \text{while } b \text{ do } c, s' \rangle \longrightarrow s''}{\langle \text{while } b \text{ do } c, s \rangle \longrightarrow s''} \\
& \text{se } \mathcal{B}[[b]] s = \text{true} \\
\text{while}^f_{sn} & \langle \text{while } b \text{ do } c, s \rangle \longrightarrow s \text{ se } \mathcal{B}[[b]] s = \text{false}
\end{array}$$

Semântica operacional estrutural (*small-step*)

$$\begin{array}{ll}
\text{att}_{sos} & \langle x := a, s \rangle \Rightarrow s[x \mapsto \mathcal{A}[[a]] s] \\
\text{skip}_{sos} & \langle \text{skip}, s \rangle \Rightarrow s \\
\text{comp}^1_{sos} & \frac{\langle c_1, s \rangle \Rightarrow \langle c'_1, s' \rangle}{\langle c_1; c_2, s \rangle \Rightarrow \langle c'_1; c_2, s' \rangle} \\
\text{comp}^2_{sos} & \frac{\langle c_1, s \rangle \Rightarrow s'}{\langle c_1; c_2, s \rangle \Rightarrow \langle c_2, s' \rangle} \\
\text{if}^v_{sos} & \langle \text{if } b \text{ then } c_1 \text{ else } c_2, s \rangle \Rightarrow \langle c_1, s \rangle \text{ se } \mathcal{B}[[b]] s = \text{true} \\
\text{if}^f_{sos} & \langle \text{if } b \text{ then } c_1 \text{ else } c_2, s \rangle \Rightarrow \langle c_2, s \rangle \text{ se } \mathcal{B}[[b]] s = \text{false} \\
\text{while}_{sos} & \langle \text{while } b \text{ do } c, s \rangle \Rightarrow \langle \text{if } b \text{ then } (c; \text{while } b \text{ do } c) \\
& \text{else skip}, s \rangle
\end{array}$$