COSE312: Compilers

Lecture 13 — Semantic Analysis (3)

Hakjoo Oh
2017 Spring

Small-step Operational Semantics

The individual computation steps are described by the transition relation of the form:

$$\langle S, s\rangle \Rightarrow \gamma$$

where γ either is non-terminal state $\langle S', s'\rangle$ or terminal state s'. The transition expresses the first step of the execution of S from state s.

- If $\gamma = \langle S', s'\rangle$, then the execution of S from s is not completed and the remaining computation continues with $\langle S', s'\rangle$.
- If $\gamma = s'$, then the execution of S from s has terminated and the final state is s'.

Small-step Operational Semantics for **While**

$$\frac{}{\langle x := a, s\rangle \Rightarrow s[x \mapsto \mathcal{A}[\![\, a \,]\!](s)]}$$

$$\frac{}{\langle \texttt{skip}, s\rangle \Rightarrow s}$$

$$\frac{\langle S_1, s\rangle \Rightarrow \langle S_1', s'\rangle}{\langle S_1; S_2, s\rangle \Rightarrow \langle S_1'; S_2, s'\rangle}$$

$$\frac{\langle S_1, s\rangle \Rightarrow s'}{\langle S_1; S_2, s\rangle \Rightarrow \langle S_2, s'\rangle}$$

$$\frac{}{\langle \texttt{if}\ b\ S_1\ \ S_2, s\rangle \Rightarrow \langle S_1, s\rangle}\ \text{if } \mathcal{B}[\![\, b \,]\!](s) = \texttt{true}$$

$$\frac{}{\langle \texttt{if}\ b\ S_1\ \ S_2, s\rangle \Rightarrow \langle S_2, s\rangle}\ \text{if } \mathcal{B}[\![\, b \,]\!](s) = \texttt{false}$$

$$\frac{}{\langle \texttt{while}\ b\ S, s\rangle \Rightarrow \langle \texttt{if}\ \ b\ \ (S;\ \ \texttt{while}\ \ b\ \ S)\ \ \texttt{skip}, s\rangle}$$

Derivation Sequence

A *derivation sequence* of a statement S starting in state s is either

- A finite sequence

  $$\gamma_0, \gamma_1, \gamma_2, \cdots, \gamma_k$$

  which is sometimes written

  $$\gamma_0 \Rightarrow \gamma_1 \Rightarrow \gamma_2 \Rightarrow \cdots \Rightarrow \gamma_k$$

  such that

  $$\gamma_0 = \langle S, s\rangle, \quad \gamma_i \Rightarrow \gamma_{i+1} \text{ for } 0 \leq i \leq k$$

  and γ_k is either a terminal configuration or a stuck configuration.
- An infinite sequence

  $$\gamma_0, \gamma_1, \gamma_2, \cdots$$

  which is sometimes written

  $$\gamma_0 \Rightarrow \gamma_1 \Rightarrow \gamma_2 \Rightarrow \cdots$$

  consisting of configurations satisfying $\gamma_0 = \langle S, s\rangle$ and $\gamma_i \Rightarrow \gamma_{i+1}$ for $0 \leq i$.

Example

Let s be a state such that $s(x) = 5, s(y) = 7, s(z) = 0$. Consider the statement:

$$(\mathrm{z} := \mathrm{x}; \mathrm{x} := \mathrm{y}); \mathrm{y} := \mathrm{z}$$

Compute the derivation sequence starting in s.

Example: Factorial

Assume that $s(x) = 3$.

$\langle$`y:=1; while ¬(x=1) do (y:=y⋆x; x:=x-1)`$, s\rangle$
$\Rightarrow \langle$`while ¬(x=1) do (y:=y⋆x; x:=x-1)`$, s[y \mapsto 1]\rangle$
$\Rightarrow \langle$`if ¬(x=1) then ((y:=y⋆x; x:=x-1);while ¬(x=1) do (y:=y⋆x; x:=x-1)) else skip`$, s[y \mapsto 1]\rangle$
$\Rightarrow \langle$`(y:=y⋆x; x:=x-1);while ¬(x=1) do (y:=y⋆x; x:=x-1)`$, s[y \mapsto 1]\rangle$
$\Rightarrow \langle$`x:=x-1;while ¬(x=1) do (y:=y⋆x; x:=x-1)`$, s[y \mapsto 3]\rangle$
$\Rightarrow \langle$`while ¬(x=1) do (y:=y⋆x; x:=x-1)`$, s[y \mapsto 3][x \mapsto 2]\rangle$
$\Rightarrow \langle$`if ¬(x=1) then ((y:=y⋆x; x:=x-1);while ¬(x=1) do (y:=y⋆x; x:=x-1)) else skip`$, s[y \mapsto 3][x \mapsto 2]\rangle$
$\Rightarrow \langle$`(y:=y⋆x; x:=x-1);while ¬(x=1) do (y:=y⋆x; x:=x-1)`$, s[y \mapsto 3][x \mapsto 2]\rangle$
$\Rightarrow \langle$`x:=x-1;while ¬(x=1) do (y:=y⋆x; x:=x-1)`$, s[y \mapsto 6][x \mapsto 2]\rangle$
$\Rightarrow \langle$`while ¬(x=1) do (y:=y⋆x; x:=x-1)`$, s[y \mapsto 6][x \mapsto 1]\rangle$
$\Rightarrow s[y \mapsto 6][x \mapsto 1]$

Other Notations

- We write $\gamma_0 \Rightarrow^i \gamma_i$ to indicate that there are i steps in the execution from γ_0 to γ_i.
- We write $\gamma_0 \Rightarrow^* \gamma_i$ to indicate that there are a finite number of steps.
- We say that the execution of a statement S on a state s terminates if and only if there is a finite derivation sequence starting with $\langle S, s \rangle$.
- The execution loops if and only if there is an infinite derivation sequence starting with $\langle S, s \rangle$.

Semantic Equivalence

We say S_1 and S_2 are semantically equivalent if for all states s,

- $\langle S_1, s\rangle \Rightarrow^* \gamma$ if and only if $\langle S_2, s\rangle \Rightarrow^* \gamma$, whenever γ is a configuration that is either stuck or terminal, and
- there is an infinite derivation sequence starting in $\langle S_1, s\rangle$ if and only if there is one starting in $\langle S_2, s\rangle$.

Semantic Function

The semantic function $\mathcal{S}_s$ for small-step semantics:

$$\mathcal{S}_s : \mathbf{Stm} \to (\mathbf{State} \hookrightarrow \mathbf{State})$$

$$\mathcal{S}_s[\![\ S\ ]\!](s) = \begin{cases} s' & \text{if } \langle S, s\rangle \Rightarrow^* s' \\ \textbf{undef} & \end{cases}$$

Summary

We have defined the operational semantics of **While**.

- *Big-step operational semantics* describes how the overall results of executions are obtained.
- *Small-step operational semantics* describes how the individual steps of the computations take place.

cf) The big-step and small-step operational semantics are equivalent:

Theorem

For every statement S of **While**, *we have* $\mathcal{S}_b[\![S]\!] = \mathcal{S}_s[\![S]\!]$.