Big Step Semantics

Dr. Mattox Beckman

University of Illinois at Urbana-Champaign
Department of Computer Science

Objectives

- Describe the components of a big step semantic rule.
- Use semantic rules to document the meaning of simple programming language.
- Explain the correspondence between big step semantics and the `eval` function.

Grammar for Simple Imperative Programming Language

The Language

$$
\begin{array}{rcl}
S & ::= & \mathbf{skip} \\
& \mid & u := A \\
& \mid & S_1 ; S_2 \\
& \mid & \mathbf{if}\ B\ \mathbf{then}\ S_1\ \mathbf{else}\ S_2\ \mathbf{fi} \\
& \mid & \mathbf{while}\ B\ \mathbf{do}\ S_1\ \mathbf{od} \\
B & ::= & E \sim E \\
& \mid & \mathbf{true} \mid \mathbf{false} \\
E & ::= & u \\
& \mid & int \\
& \mid & E \oplus E
\end{array}
$$

- Let u be a possibly subscripted variable.
- E represents arithmetic expressions, $\oplus$ is an arithmetic operator.

The Downarrow Notation

- In small step semantics we use the $\rightarrow$ to represent one step of computations.
- In big step semantics we use $\Downarrow$ to represent an entire evaluation.

Statements

$$< S, \sigma > \Downarrow \sigma'$$

Expressions

$$< E, \sigma > \Downarrow_e v$$

Booleans

$$< B, \sigma > \Downarrow_b b$$

Expressions

Integers

$$\frac{}{< i, \sigma > \Downarrow_e i}\ \textsc{Const}$$

if i is an integer.

Variables

$$\frac{}{< u, \sigma > \Downarrow_e v}\ \textsc{Var}$$

if $u := v \in \sigma$.

Operations

$$\frac{< e_1, \sigma > \Downarrow_e v_1 \qquad < e_2, \sigma > \Downarrow_e v_2}{< e_1 \oplus e_2, \sigma > \Downarrow_e v_1 \oplus v_2}\ \textsc{Arith}$$

Here $\oplus$ represents typical binary operations like $+$, $-$, $\times$, etc.

Boolean Expressions

Booleans

$$\frac{}{< b, \sigma > \Downarrow_b b} \textsc{Const}$$

if b is a boolean.

Variables

$$\frac{}{< u, \sigma > \Downarrow_b v} \textsc{Var}$$

if $u := v \in \sigma$.

Relational Operators

$$\frac{< e_1, \sigma > \Downarrow_e v_1 \qquad < e_2, \sigma > \Downarrow_e v_2}{< e_1 \sim e_2, \sigma > \Downarrow_b v_1 \sim v_2} \textsc{Rel}$$

Here $\sim$ represents the binary relational operations $=, \leq, \geq, \neq, \geq, \leq$, etc.

Skip and Assignment

$$\frac{}{< \texttt{skip}\, , \sigma >\Downarrow \sigma}\ \textsc{Skip}$$

$$\frac{< e, \sigma >\Downarrow_e v}{< x := e, \sigma >\Downarrow \sigma[x := v]}\ \textsc{Assign}$$

Skip and Assignment

$$\frac{}{<\texttt{skip}, \sigma>\Downarrow \sigma}\ \textsc{Skip}$$

$$\frac{<e, \sigma>\Downarrow_e v}{<x := e, \sigma>\Downarrow \sigma[x := v]}\ \textsc{Assign}$$

Next is sequencing. See if you can guess what the rule looks like.

Sequencing

$$\frac{< S_1, \sigma > \Downarrow \sigma' \qquad < S_2, \sigma' > \Downarrow \sigma''}{< S_1; S_2, \sigma > \Downarrow \sigma''} \textsc{Seq}$$

Sequencing

$$\frac{< S_1, \sigma >\Downarrow \sigma' \qquad < S_2, \sigma' >\Downarrow \sigma''}{< S_1; S_2, \sigma >\Downarrow \sigma''} \textsc{Seq}$$

Next is `if` . There are two rules for this. See if you can guess what the rules looks like.

If Statements

$$\frac{< B, \sigma >\Downarrow_b \texttt{true} \qquad < S_1, \sigma >\Downarrow \sigma'}{< \texttt{if}\ B\ \texttt{then}\ S_1\ \texttt{else}\ S_2\ \texttt{fi}\ , \sigma >\Downarrow \sigma'}\ \mathsf{IF}_1$$

$$\frac{< B, \sigma >\Downarrow_b \texttt{false} \qquad < S_2, \sigma >\Downarrow \sigma'}{< \texttt{if}\ B\ \texttt{then}\ S_1\ \texttt{else}\ S_2\ \texttt{fi}\ , \sigma >\Downarrow \sigma'}\ \mathsf{IF}_2$$

If Statements

$$\frac{< B, \sigma >\Downarrow_b \texttt{true} \qquad < S_1, \sigma >\Downarrow \sigma'}{< \texttt{if } B \texttt{ then } S_1 \texttt{ else } S_2 \texttt{ fi }, \sigma >\Downarrow \sigma'}\ \mathsf{IF}_1$$

$$\frac{< B, \sigma >\Downarrow_b \texttt{false} \qquad < S_2, \sigma >\Downarrow \sigma'}{< \texttt{if } B \texttt{ then } S_1 \texttt{ else } S_2 \texttt{ fi }, \sigma >\Downarrow \sigma'}\ \mathsf{IF}_2$$

Next is `while` . There are two rules for this. See if you can guess what the rules looks like. The second one uses induction!

While Statements

$$\frac{< B, \sigma >\Downarrow_b \texttt{false}}{< \texttt{while}\ B\ \texttt{do}\ S\ \texttt{od}\ , \sigma >\Downarrow \sigma}\ \textsc{While}_1$$

$$\frac{< B, \sigma >\Downarrow_b \texttt{true} \qquad < S, \sigma >\Downarrow \sigma' \qquad < \texttt{while}\ B\ \texttt{do}\ S\ \texttt{od}\ , \sigma' >\Downarrow \sigma''}{< \texttt{while}\ B\ \texttt{do}\ S\ \texttt{od}\ , \sigma >\Downarrow \sigma''}\ \textsc{While}_2$$

Proof Trees

- To show the effect of a program, we need to build proof trees.
- Let $\sigma = \{x := 3, y := 4\}$.
- We want to prove that $2 \times y + 9 \times x = 35$.

Here is what we want to evaluate. What kind of expression is this?

$$\frac{}{< 2 \times y + 9 \times x, \sigma > \Downarrow_e 35}$$

Proof Trees

- To show the effect of a program, we need to build proof trees.
- Let $\sigma = \{x := 3, y := 4\}$.
- We want to prove that $2 \times y + 9 \times x = 35$.

Because of precedence rules, we evaluate the + last.

$$\frac{\dfrac{}{< 2 \times y, \sigma >\Downarrow_e 8} \qquad \dfrac{}{< 9 \times x, \sigma >\Downarrow_e 27}}{< 2 \times y + 9 \times x, \sigma >\Downarrow_e 35}\ \textsc{Arith}$$

Proof Trees

- To show the effect of a program, we need to build proof trees.
- Let $\sigma = \{x := 3, y := 4\}$.
- We want to prove that $2 \times y + 9 \times x = 35$.

We go up one more level, and then we are done.

$$\dfrac{\dfrac{\dfrac{}{< 2, \sigma > \Downarrow_e 2}\ \textsc{Const} \qquad \dfrac{}{< y, \sigma > \Downarrow_e 4}\ \textsc{Var}}{< 2 \times y, \sigma > \Downarrow_e 8}\ \textsc{Arith} \qquad \dfrac{\dfrac{}{< 9, \sigma > \Downarrow_e 9}\ \textsc{Const} \qquad \dfrac{}{< x, \sigma > \Downarrow_e 3}\ \textsc{Var}}{< 9 \times x, \sigma > \Downarrow_e 27}\ \textsc{Arith}}{< 2 \times y + 9 \times x, \sigma > \Downarrow_e 35}\ \textsc{Arith}$$

Statement Proof Tree

- Let $\sigma = \{x := 10, y := 20\}$.
- Let $\sigma' = \{x := 10, y := 20, m := 20\}$.

Here is an example that will use all three versions of $\Downarrow$.

$$\frac{}{< \texttt{if } x > y \texttt{ then } m := x \texttt{ else } m := 2 \times x \texttt{ fi}, \sigma > \Downarrow \sigma'}$$

Can you figure out what this tree should look like?

Statement Proof Tree

- Let $\sigma = \{x := 10, y := 20\}$.
- Let $\sigma' = \{x := 10, y := 20, m := 20\}$.

$$\dfrac{\dfrac{}{< x > y, \sigma >\Downarrow_b \texttt{false}}\textsc{Rel} \qquad \dfrac{}{< m := 2 \times x, \sigma >\Downarrow \sigma'}\textsc{Assign}}{< \texttt{if}\ x > y\ \texttt{then}\ m := x\ \texttt{else}\ m := 2 \times x\ \texttt{fi}\ , \sigma >\Downarrow \sigma'}\textsc{If}_2$$

Statement Proof Tree

- Let $\sigma = \{x := 10, y := 20\}$.
- Let $\sigma' = \{x := 10, y := 20, m := 20\}$.

$$
\dfrac{
  \dfrac{
    \dfrac{}{< x, \sigma > \Downarrow_e 10}\ \textsc{Var}
    \quad
    \dfrac{}{< y, \sigma > \Downarrow_e 20}\ \textsc{Var}
  }{< x > y, \sigma > \Downarrow_b \mathtt{false}}\ \textsc{Rel}
  \quad
  \dfrac{
    \dfrac{}{< 2, \sigma > \Downarrow_e 2}\ \textsc{Const}
    \quad
    \dfrac{}{< x, \sigma > \Downarrow_e 10}\ \textsc{Var}
  }{< m := 2 \times x, \sigma > \Downarrow \sigma'}\ \textsc{Assign}
}{< \mathtt{if}\ x > y\ \mathtt{then}\ m := x\ \mathtt{else}\ m := 2 \times x\ \mathtt{fi}\ , \sigma > \Downarrow \sigma'}\ \textsc{If}_2
$$

Connecting to Interpreters

- The $\Downarrow$ is really just `eval` that you already know and love.
- The σ is just the `env` parameter.