Large-step semantics

Lecture 4 Thursday, February 1, 2024

1 Large-step semantics

So far we have defined the small step evaluation relation $\longrightarrow \subseteq$ **Config** $\times$ **Config** for our simple language of arithmetic expressions, and used its transitive and reflexive closure $\longrightarrow^*$ to describe the execution of multiple steps of evaluation. In particular, if $\langle e, \sigma \rangle$ is some start configuration, and $\langle n, \sigma' \rangle$ is a final configuration, the evaluation $\langle e, \sigma \rangle \longrightarrow^* \langle n, \sigma' \rangle$ shows that by executing expression e starting with the store σ, we get the result n, and the final store σ'.

Large-step semantics is an alternative way to specify the operational semantics of a language. Large-step semantics directly give the final result.

We'll use the same configurations as before, but define a large step evaluation relation:

$$\Downarrow \subseteq \textbf{Config} \times \textbf{FinalConfig}$$

where

$$\textbf{Config} = \textbf{Exp} \times \textbf{Store}$$
$$\text{and } \textbf{FinalConfig} = \textbf{Int} \times \textbf{Store} \subseteq \textbf{Config}.$$

We write $\langle e, \sigma \rangle \Downarrow \langle n, \sigma' \rangle$ to mean that $(\langle e, \sigma \rangle, \langle n, \sigma' \rangle) \in \Downarrow$. In other words, configuration $\langle e, \sigma \rangle$ evaluates in one big step directly to final configuration $\langle n, \sigma' \rangle$. In general, the big step semantics takes a configuration to an "answer". For our language of arithmetic expressions, "answers" are a subset of configurations, but in general the "answer" to a computation might be something other than a configuration.

The large step semantics boils down to defining the relation $\Downarrow$. We use inference rules to inductively define the relation $\Downarrow$, similar to how we specified the small-step operational semantics $\longrightarrow$.

$$\textsc{Int}_{\textsc{Lrg}}\ \frac{}{\langle n, \sigma \rangle \Downarrow \langle n, \sigma \rangle} \qquad\qquad \textsc{Var}_{\textsc{Lrg}}\ \frac{}{\langle x, \sigma \rangle \Downarrow \langle n, \sigma \rangle}\ \text{where } \sigma(x) = n$$

$$\textsc{Add}_{\textsc{Lrg}}\ \frac{\langle e_1, \sigma \rangle \Downarrow \langle n_1, \sigma'' \rangle \qquad \langle e_2, \sigma'' \rangle \Downarrow \langle n_2, \sigma' \rangle}{\langle e_1 + e_2, \sigma \rangle \Downarrow \langle n, \sigma' \rangle}\ \text{where } n \text{ is the sum of } n_1 \text{ and } n_2$$

$$\textsc{Mul}_{\textsc{Lrg}}\ \frac{\langle e_1, \sigma \rangle \Downarrow \langle n_1, \sigma'' \rangle \qquad \langle e_2, \sigma'' \rangle \Downarrow \langle n_2, \sigma' \rangle}{\langle e_1 \times e_2, \sigma \rangle \Downarrow \langle n, \sigma' \rangle}\ \text{where } n \text{ is the product of } n_1 \text{ and } n_2$$

$$\textsc{Asg}_{\textsc{Lrg}}\ \frac{\langle e_1, \sigma \rangle \Downarrow \langle n_1, \sigma'' \rangle \qquad \langle e_2, \sigma''[x \mapsto n_1] \rangle \Downarrow \langle n_2, \sigma' \rangle}{\langle x := e_1; e_2, \sigma \rangle \Downarrow \langle n_2, \sigma' \rangle}$$

To see how we use these rules, here is a proof tree that shows that $\langle \mathsf{foo} := 3; \mathsf{foo} \times \mathsf{bar}, \sigma \rangle \Downarrow \langle 21, \sigma' \rangle$ for a store σ such that $\sigma(\mathsf{bar}) = 7$, and $\sigma' = \sigma[\mathsf{foo} \mapsto 3]$.

$$\textsc{Asg}_{\textsc{Lrg}}\ \dfrac{\textsc{Int}_{\textsc{Lrg}}\ \dfrac{}{\langle 3,\sigma\rangle \Downarrow \langle 3,\sigma\rangle} \qquad \textsc{Mul}_{\textsc{Lrg}}\ \dfrac{\textsc{Var}_{\textsc{Lrg}}\ \dfrac{}{\langle \mathsf{foo},\sigma'\rangle \Downarrow \langle 3,\sigma'\rangle} \quad \textsc{Var}_{\textsc{Lrg}}\ \dfrac{}{\langle \mathsf{bar},\sigma'\rangle \Downarrow \langle 7,\sigma'\rangle}}{\langle \mathsf{foo}\times\mathsf{bar},\sigma'\rangle \Downarrow \langle 21,\sigma'\rangle}}{\langle \mathsf{foo} := 3; \mathsf{foo}\times\mathsf{bar},\sigma\rangle \Downarrow \langle 21,\sigma'\rangle}$$

A closer look to this structure reveals the relation between small step and large-step evaluation: a depth-first traversal of the large-step proof tree yields the sequence of one-step transitions in small-step evaluation.

2 Equivalence of semantics

So far, we have specified the semantics of our language of arithmetic expressions in two different ways: small-step operational semantics and large-step operational semantics. Are they expressing the same meaning of arithmetic expressions? Can we show that they express the same thing?

Theorem (Equivalence of semantics)**.** *For all expressions e, stores σ and σ', and integers n, we have:*

$$\langle e,\sigma\rangle \Downarrow \langle n,\sigma'\rangle \iff \langle e,\sigma\rangle \longrightarrow^* \langle n,\sigma'\rangle.$$

Proof sketch.

- $\Longrightarrow$. We proceed by structural induction on expressions e. The property we will prove by induction is:

$$P(e) = \forall\sigma,\sigma' \in \textbf{Store}.\ ,\forall n \in \textbf{Int}.\ \langle e,\sigma\rangle \Downarrow \langle n,\sigma'\rangle \Longrightarrow \langle e,\sigma\rangle \longrightarrow^* \langle n,\sigma'\rangle$$

  We have to consider each of the possible axioms and inference rules for constructing an expression.

  - **Case $e \equiv x$.**
    Here, we are considering the case where the expression e is equal to some variable x. Assume that for some σ, σ', and n we have $\langle x,\sigma\rangle \Downarrow \langle n,\sigma'\rangle$. That means that there is some derivation using the axioms and inference rules of the large-step operational semantics, whose conclusion is $\langle x,\sigma\rangle \Downarrow \langle n,\sigma'\rangle$. There is only one rule whose conclusion could look like this, the rule Var_{Lrg}. That rule requires that $n = \sigma(x)$, and that $\sigma' = \sigma$.

    (This reasoning is an example of *inversion*: using the inference rules in reverse. That is, we know that some conclusion holds—$\langle x,\sigma\rangle \Downarrow \langle n,\sigma'\rangle$—and we examine the inference rules to determine which rule must have been used in the derivation, and thus which premises must be true, and which side conditions satisfied.)

    Since $n = \sigma(x)$ we know that $\langle x,\sigma\rangle \longrightarrow \langle n,\sigma\rangle$ also holds, by using the small-step axiom $\textsc{Var}$. So we can conclude that $\langle x,\sigma\rangle \longrightarrow^* \langle n,\sigma\rangle$ holds, which is what we needed to show.

  - **Case $e \equiv n$.**
    Here, we consider the case where expression e is equal to some integer n. Assume that for some σ, σ', and n' we have $\langle n,\sigma\rangle \Downarrow \langle n',\sigma'\rangle$. Like the case above, by inversion, we know that the rule Int_{Lrg} was used to conclude that $\langle n,\sigma\rangle \Downarrow \langle n',\sigma'\rangle$, and so $n' = n$ and $\sigma' = \sigma$.

    So we need to show that $\langle n,\sigma\rangle \longrightarrow^* \langle n,\sigma\rangle$. But this holds trivially because of reflexivity of $\longrightarrow^*$.

  - **Case $e \equiv e_1 + e_2$.**
    This is an inductive case. Expressions e_1 and e_2 are subexpressions of e, and so we can assume that $P(e_1)$ and $P(e_2)$ hold. We need to show that $P(e)$ holds. Let's write out $P(e_1)$, $P(e_2)$, and $P(e)$ explicitly.

$$\begin{aligned} P(e_1) &= \forall n,\sigma,\sigma' : \langle e_1,\sigma\rangle \Downarrow \langle n,\sigma'\rangle \Longrightarrow \langle e_1,\sigma\rangle \longrightarrow^* \langle n,\sigma'\rangle \\ P(e_2) &= \forall n,\sigma,\sigma' : \langle e_2,\sigma\rangle \Downarrow \langle n,\sigma'\rangle \Longrightarrow \langle e_2,\sigma\rangle \longrightarrow^* \langle n,\sigma'\rangle \\ P(e) &= \forall n,\sigma,\sigma' : \langle e_1 + e_2,\sigma\rangle \Downarrow \langle n,\sigma'\rangle \Longrightarrow \langle e_1 + e_2,\sigma\rangle \longrightarrow^* \langle n,\sigma'\rangle \end{aligned}$$

Assume that for some σ, σ' and n we have $\langle e_1 + e_2, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$. We now need to show that $\langle e_1 + e_2, \sigma\rangle \longrightarrow^* \langle n, \sigma'\rangle$.

We assumed that $\langle e_1 + e_2, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$. Let's use inversion again: there is some derivation whose conclusion is $\langle e_1 + e_2, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$. By looking at the large-step semantic rules, we see that only one rule could possible have a conclusion of this form: the rule ADD_{LRG}. So that means that the last rule use in the derivation was ADD_{LRG}. But in order to use the rule ADD_{LRG}, it must be the case that $\langle e_1, \sigma\rangle \Downarrow \langle n_1, \sigma''\rangle$ and $\langle e_2, \sigma''\rangle \Downarrow \langle n_2, \sigma'\rangle$ hold for some n_1 and n_2 such that $n = n_1 + n_2$ (i.e., there is a derivation whose conclusion is $\langle e_1, \sigma\rangle \Downarrow \langle n_1, \sigma''\rangle$ and a derivation whose conclusion is $\langle e_2, \sigma''\rangle \Downarrow \langle n_2, \sigma'\rangle$).

Using the inductive hypothesis $P(e_1)$, since $\langle e_1, \sigma\rangle \Downarrow \langle n_1, \sigma''\rangle$, we must have $\langle e_1, \sigma\rangle \longrightarrow^* \langle n_1, \sigma''\rangle$. Similarly, by $P(e_2)$, we have $\langle e_2, \sigma''\rangle \longrightarrow^* \langle n_2, \sigma\rangle$. By Lemma 1 below, we have

$$\langle e_1 + e_2, \sigma\rangle \longrightarrow^* \langle n_1 + e_2, \sigma''\rangle$$

and by another application of Lemma 1 we have

$$\langle n_1 + e_2, \sigma''\rangle \longrightarrow^* \langle n_1 + n_2, \sigma'\rangle$$

and by the rule ADD we have

$$\langle n_1 + n_2, \sigma'\rangle \longrightarrow \langle n, \sigma'\rangle.$$

Thus, we have $\langle e_1 + e_2, \sigma\rangle \longrightarrow^* \langle n, \sigma'\rangle$, which proves this case.

- **Case** $e \equiv e_1 \times e_2$**.** Similar to the case $e = e_1 + e_2$ above.
- **Case** $e \equiv x := e_1; e_2$**.** Omitted. Try it as an exercise.

- $\Longleftarrow$. We proceed by mathematical induction on the number of steps $\langle e, \sigma\rangle \longrightarrow^* \langle n, \sigma'\rangle$.
  - **Base case.** If $\langle e, \sigma\rangle \longrightarrow^* \langle n, \sigma'\rangle$ in zero steps, then we must have $e \equiv n$ and $\sigma' = \sigma$. Then, $\langle n, \sigma\rangle \Downarrow \langle n, \sigma\rangle$ by the large-step operational semantics rule INT_{LRG}.
  - **Inductive case.** Assume that $\langle e, \sigma\rangle \longrightarrow \langle e'', \sigma''\rangle \longrightarrow^* \langle n, \sigma'\rangle$, and that (the inductive hypothesis) $\langle e'', \sigma''\rangle \Downarrow \langle n, \sigma'\rangle$. That is, $\langle e'', \sigma''\rangle \longrightarrow^* \langle n, \sigma'\rangle$ takes m steps, and we assume that the property holds for it ($\langle e'', \sigma''\rangle \Downarrow \langle n, \sigma'\rangle$), and we are considering $\langle e, \sigma\rangle \longrightarrow^* \langle n, \sigma'\rangle$, which takes $m+1$ steps. We need to show that $\langle e, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$. This follows immediately from Lemma 2 below.

□

Lemma 1. *If $\langle e, \sigma\rangle \longrightarrow^* \langle n, \sigma'\rangle$ then for all n_1, e_2 the following hold.*

- $\langle e + e_2, \sigma\rangle \longrightarrow^* \langle n + e_2, \sigma'\rangle$
- $\langle e \times e_2, \sigma\rangle \longrightarrow^* \langle n \times e_2, \sigma'\rangle$
- $\langle n_1 + e, \sigma\rangle \longrightarrow^* \langle n_1 + n, \sigma'\rangle$
- $\langle n_1 \times e, \sigma\rangle \longrightarrow^* \langle n_1 \times n, \sigma'\rangle$

Proof. By (mathematical) induction on the number of evaluation steps in $\longrightarrow^*$. □

Lemma 2. *For all e, e', σ, and n, if $\langle e, \sigma\rangle \longrightarrow \langle e', \sigma''\rangle$ and $\langle e', \sigma''\rangle \Downarrow \langle n, \sigma'\rangle$, then $\langle e, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$.*

2.1 Again: structural induction vs rule induction

Like in the previous lecture, here we can also prove that large-step evaluation implies small-step evaluation by induction on the large-step derivation rather than structural induction on the expression.

$\Longrightarrow$. We proceed by induction on the derivation of $\langle e, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$. The property we wish to show is

$$P(\langle e, \sigma\rangle \Downarrow \langle n, \sigma'\rangle) = \langle e, \sigma\rangle \longrightarrow^* \langle n, \sigma'\rangle$$

Suppose we have a derivation of $\langle e, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$ for some e, σ, n, and σ'. Assume that the inductive hypothesis holds for any subderivation $\langle e_0, \sigma_0\rangle \Downarrow \langle n_0, \sigma_0'\rangle$ used in the derivation of $\langle e, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$. Consider the last rule used in the derivation of $\langle e, \sigma\rangle \Downarrow \langle n, \sigma'\rangle$.

- INT_{LRG}. Here, $e \equiv n$ and $\sigma' = \sigma$. The result holds trivially, as $\langle n, \sigma\rangle \longrightarrow^* \langle n, \sigma\rangle$.
- VAR_{LRG}. Here $e \equiv x$ for some variable x, and $\sigma' = \sigma$ and $n = \sigma(x)$. Using the small step rule VAR, we can derive $\langle x, \sigma\rangle \longrightarrow \langle n, \sigma\rangle$, and thus $\langle x, \sigma\rangle \longrightarrow^* \langle n, \sigma\rangle$.
- ADD_{LRG}. Here $e \equiv e_1 + e_2$ for some expressions e_1 and e_2, and $n = n_1 + n_2$ for some n_1 and n_2, and we have derivations for $\langle e_1, \sigma\rangle \Downarrow \langle n_1, \sigma''\rangle$ and $\langle e_2, \sigma''\rangle \Downarrow \langle n_2, \sigma_1\rangle$.

  By Lemma 1, we have

  $$\langle e_1 + e_2, \sigma\rangle \longrightarrow^* \langle n_1 + e_2, \sigma''\rangle$$

  and by another application of Lemma 1 we have

  $$\langle n_1 + e_2, \sigma''\rangle \longrightarrow^* \langle n_1 + n_2, \sigma'\rangle$$

  and by the rule ADD we have

  $$\langle n_1 + n_2, \sigma'\rangle \longrightarrow \langle n, \sigma'\rangle.$$

  Thus, we have $\langle e_1 + e_2, \sigma\rangle \longrightarrow^* \langle n, \sigma'\rangle$, which proves this case.
- MUL_{LRG}. Similar to the case ADD_{LRG} above.
- ASG_{LRG}. Omitted. Try it as an exercise.