Operational Semantics
Part I

Jim Royer

CIS 352

February 18, 2016

References

- Andrew Pitts' Lecture Notes on Semantics of Programming Languages `http://www.inf.ed.ac.uk/teaching/courses/lsi/sempl.pdf`. We'll be following the Pitts' notes for a while and mostly using his notation.
- Matthew Hennessy's Semantics of programming languages: `https://www.scss.tcd.ie/Matthew.Hennessy/splexternal2015/LectureNotes/Notes14%20copy.pdf` is very readable and very good.
- There are many of other good references in Hennessy's reading list: `https://www.scss.tcd.ie/Matthew.Hennessy/splexternal2015/reading.php`

Aexp, A little language for arithmetic expressions

Syntactic categories

$n \in \mathbf{Num}$ Numerals
$a \in \mathbf{Aexp}$ Arithmetic expressions

Grammar

$$
\begin{aligned}
a &::= n \\
&\mid (a_1 + a_2) \\
&\mid (a_1 - a_2) \\
&\mid (a_1 * a_2) \\
n &::= \dots
\end{aligned}
$$

Conventions

- Metavariables: n, a, b, w, x, etc.
- We write $\underline{\underline{35}}$ for the numeral 35.

Examples

- $\underline{\underline{2}}$
- $(\underline{\underline{2}} + \underline{\underline{5}})$
- $(((\underline{\underline{2}} + \underline{\underline{5}}) * \underline{\underline{13}}) - \underline{\underline{9}})$

Syntax

Concrete syntax

≈ phonemes, characters, words, tokens — the raw stuff of language

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, . . .

Grammar

≈ collection of formation rules to organize parts into a whole. E.g.,

- words into noun phrases, verb phrases, . . . , sentences
- key words, tokens, . . . into expressions, statements, . . . , programs

Abstract syntax

≈ a structure (e.g., labeled tree or data structure) showing how a "phrase" breaks down into pieces according to a specific rule.

Aexp's abstract syntax

Grammar

$$
\begin{aligned}
a \;&::=\; n \\
&\mid\; (a_1 + a_2) \\
&\mid\; (a_1 - a_2) \\
&\mid\; (a_1 * a_2) \\
n \;&::=\; \ldots
\end{aligned}
$$

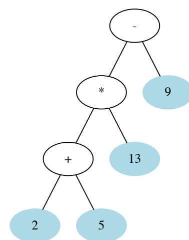
In Haskell

```
data AExp = Num Integer
          | Plus AExp AExp
          | Minus AExp AExp
          | Times AExp AExp
```

$(((\underline{\underline{2}} + \underline{\underline{5}}) * \underline{\underline{13}}) - \underline{\underline{9}})$

```
Minus (Times (Plus (Num 2) (Num 5))
             (Num 13))
      (Num 9)
```

As a Parse Tree



What do **Aexp** expression mean? Big-step rules

$a \;::=\; n \mid (a_1 + a_2) \mid (a_1 - a_2) \mid (a_1 * a_2)$

$$\textit{PLUS:}\ \frac{a_1 \Downarrow v_1 \qquad a_2 \Downarrow v_2}{(a_1 + a_2) \Downarrow v}\ (v = v_1 + v_2)$$

$$\textit{MINUS:}\ \frac{a_1 \Downarrow v_1 \qquad a_2 \Downarrow v_2}{(a_1 - a_2) \Downarrow v}\ (v = v_1 - v_2)$$

$$\textit{MULT:}\ \frac{a_1 \Downarrow v_1 \qquad a_2 \Downarrow v_2}{(a_1 * a_2) \Downarrow v}\ (v = v_1 * v_2)$$

$$\textit{NUM:}\ \frac{}{n \Downarrow v}\ (\mathcal{N}[\![n]\!] = v)$$

Notes

- $a \Downarrow v \equiv$ expression a evaluates to value v.
- $\Downarrow$ is an evaluation relation.
- Upstairs assertions are called premises.
- Downstairs assertions are called conclusions.
- Parenthetical equations on the side are called side conditions.
- $\mathcal{N}$: numerals $\to \mathbb{Z}$. I.e., $\mathcal{N}[\![\, \underline{\underline{-43}}\,]\!] = -43$.
- The NUM_{BSS} rule is an example of an axiom.

Digression: Rules, 1

General Format for Rules

$$\textit{Name:}\ \frac{\text{premise}_1 \quad \cdots \quad \text{premise}_k}{\text{conclusion}}\ (\text{side condition})$$

Example

- *Modus Ponens:* $\dfrac{p \implies q \qquad p}{q}$
- *Transitivity:* $\dfrac{x \equiv y \qquad y \equiv z}{x \equiv z}$
- $\textit{PLUS:}\ \dfrac{a_1 \Downarrow v_1 \qquad a_2 \Downarrow v_2}{(a_1 + a_2) \Downarrow v}\ (v = v_1 + v_2)$

Digression: Rules, 2

General Format for Rules

$$\textit{Name:}\ \frac{\text{premise}_1 \quad \cdots \quad \text{premise}_k}{\text{conclusion}}\ (\text{side condition})$$

Definition

A rule with no premises is an *axiom*.

Definition

A rule is *sound* if and only if the conclusion is true whenever the premises (and side-condition—if any) are true.

Question

So an axiom is sound when ...?

Digression: Rules, 3

General Format for Rules

$$\textit{Name:}\ \frac{\text{premise}_1 \quad \cdots \quad \text{premise}_k}{\text{conclusion}} \quad (\text{side condition})$$

Proofs from gluing together rule applications

$$\textit{Times:}\ \dfrac{\textit{Plus:}\ \dfrac{\textit{Num:}\ \dfrac{}{\underline{\underline{2}} \Downarrow 2} \quad \textit{Num:}\ \dfrac{}{\underline{\underline{5}} \Downarrow 5}}{(\underline{\underline{2}}+\underline{\underline{5}}) \Downarrow 7}\ (2+5=7) \qquad \textit{Num:}\ \dfrac{}{\underline{\underline{13}} \Downarrow 13}}{((\underline{\underline{2}}+\underline{\underline{5}}) * \underline{\underline{13}}) \Downarrow 91}\ \left(\begin{matrix}7*13\\ =91\end{matrix}\right)$$

Rules can also be the basis of a computation

$$\dfrac{\vdots}{((\underline{\underline{2}}+\underline{\underline{5}}) * \underline{\underline{13}}) \Downarrow ??}$$

$$\Downarrow\Downarrow$$

$$\dfrac{\dfrac{\vdots}{(\underline{\underline{2}}+\underline{\underline{5}}) \Downarrow ??} \qquad \dfrac{}{\underline{\underline{13}} \Downarrow 13}}{((\underline{\underline{2}}+\underline{\underline{5}}) * \underline{\underline{13}}) \Downarrow ??}$$

$$\Downarrow\Downarrow$$

$$\dfrac{\dfrac{\dfrac{}{\underline{\underline{2}} \Downarrow 2} \quad \dfrac{}{\underline{\underline{5}} \Downarrow 5}}{(\underline{\underline{2}}+\underline{\underline{5}}) \Downarrow ??} \qquad \dfrac{}{\underline{\underline{13}} \Downarrow 13}}{((\underline{\underline{2}}+\underline{\underline{5}}) * \underline{\underline{13}}) \Downarrow ??}$$

$$\Downarrow\Downarrow$$

$$\dfrac{\dfrac{\dfrac{}{\underline{\underline{2}} \Downarrow 2} \quad \dfrac{}{\underline{\underline{5}} \Downarrow 5}}{(\underline{\underline{2}}+\underline{\underline{5}}) \Downarrow 7} \qquad \dfrac{}{\underline{\underline{13}} \Downarrow 13}}{((\underline{\underline{2}}+\underline{\underline{5}}) * \underline{\underline{13}}) \Downarrow ??}$$

$$\Downarrow\Downarrow$$

$$\dfrac{\dfrac{\dfrac{}{\underline{\underline{2}} \Downarrow 2} \quad \dfrac{}{\underline{\underline{5}} \Downarrow 5}}{(\underline{\underline{2}}+\underline{\underline{5}}) \Downarrow 7} \qquad \dfrac{}{\underline{\underline{13}} \Downarrow 13}}{((\underline{\underline{2}}+\underline{\underline{5}}) * \underline{\underline{13}}) \Downarrow 91}$$

The big-step semantics in Haskell

A Haskell version of the abstract syntax

```
data Aexp  = Num Integer
           | Add Aexp Aexp
           | Sub Aexp Aexp
           | Mult Aexp Aexp
```

The big-step semantics as an evaluator function

```
aBig (Add a1 a2)  = (aBig a1) + (aBig a2)
aBig (Sub a1 a2)  = (aBig a1) - (aBig a2)
aBig (Mult a1 a2) = (aBig a1) * (aBig a2)
aBig (Num n)      = n
```


Do these rules make sense?

Theorem

Suppose $e \in$ **Aexp**.
Then there is a unique integer v such that $e \Downarrow v$.

Proof (by rule induction).

Case: *NUM*. This is immediate.

Case: *PLUS*.
By IH, there are unique v_1 and v_2 such that $a_1 \Downarrow v_1$ and $a_2 \Downarrow v_2$.
By arithmetic, there is a unique v such that $v = v_1 + v_2$.
Hence, there is a unique v such that $a_1 + a_2 \Downarrow v$.

Cases: *MINUS* and *MULT*. These follow *mutatis mutandis*. □

$$\textit{PLUS}_{BSS}\!:\ \frac{a_1 \Downarrow v_1 \qquad a_2 \Downarrow v_2}{(a_1 + a_2) \Downarrow v}\ (v = v_1 + v_2) \quad \ldots \quad \textit{NUM}_{BSS}\!:\ \frac{}{n \Downarrow v}\ (\mathcal{N}[\![n]\!] = v)$$

What do **Aexp** expression mean? Small-step rules

$$a \;::=\; n \mid (a_1 + a_2) \mid (a_1 - a_2) \mid (a_1 * a_2) \mid v$$

$$\textit{PLUS-1}_{SSS}:\ \frac{a_1 \to a_1'}{(a_1 + a_2) \to (a_1' + a_2)}$$

$$\textit{PLUS-2}_{SSS}:\ \frac{a_2 \to a_2'}{(a_1 + a_2) \to (a_1 + a_2')}$$

$$\textit{PLUS-3}_{SSS}:\ \frac{}{(v_1 + v_2) \to v}\ (v = v_1 + v_2)$$

$$\vdots$$

$$\textit{NUM}_{SSS}:\ \frac{}{n \to v}\ (\mathcal{N}[\![n]\!] = v)$$

Notes

- These are rewrite rules.
- We now allow values in expressions.
- $a \to a'$ is a transition.
- $a \to a' \equiv$ expression a evaluates (or rewrites) to a' in one-step.
- v is a terminal expression.
- The rules for $-$ and $*$ follow the same pattern as the $+$-rules.

Class exercise

Show:

$$(((\underline{\underline{3}} * \underline{\underline{2}}) + (\underline{\underline{8}} - \underline{\underline{3}})) * (\underline{\underline{5}} - \underline{\underline{2}}))$$

$$\to \begin{cases} ((6 + (\underline{\underline{8}} - \underline{\underline{3}})) * (\underline{\underline{5}} - \underline{\underline{2}})) \\ (((\underline{\underline{3}} * \underline{\underline{2}}) + 5) * (\underline{\underline{5}} - \underline{\underline{2}})) \\ (((\underline{\underline{3}} * \underline{\underline{2}}) + (\underline{\underline{8}} - \underline{\underline{3}})) * 3) \end{cases}$$

Some full small-step derivations of transitions

$$\textit{MULT}_1\ \dfrac{\textit{PLUS}_2\ \dfrac{\textit{MINUS}_3\ \dfrac{}{(8-3) \to 5}}{(6+(8-3)) \to (6+5)}}{((6+(8-3))*(5-2)) \to ((6+5)*(5-2))}$$

$$\textit{MULT}_1\ \dfrac{\textit{PLUS}_3\ \dfrac{}{(6+5) \to 11}}{((6+5)*(5-2)) \to 11*(5-2)}$$

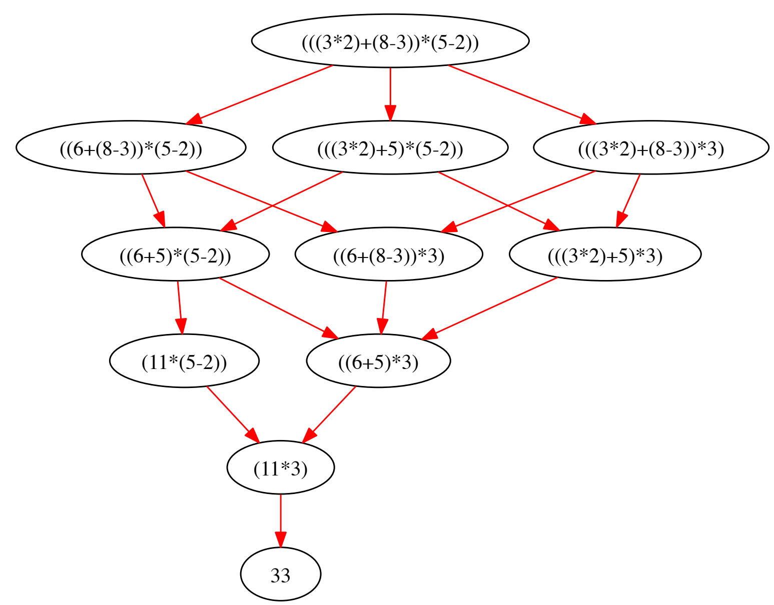
$$\textit{MULT}_2\ \dfrac{\textit{MINUS}_3\ \dfrac{}{(5-2) \to 3}}{(11*(5-2)) \to 11*3}$$

$$\textit{MULT}_3\ \dfrac{}{(11*3) \to 33}$$

The derivations show that the steps in the transition sequence below are legal (i.e., follow from the rules).

$$((6+(8-3))*(5-2)) \to ((6+5)*(5-2)) \to 11*(5-2) \to 11*3 \to 33$$

There is a lattice of transitions



Properties of operational semantics

Definition

A transition system $(\Gamma, \rightsquigarrow, T)$ is deterministic when for all a, a_1, and a_2:

$$\text{If } a \rightsquigarrow a_1 \text{ and } a \rightsquigarrow a_2, \text{ then } a_1 = a_2.$$

Theorem

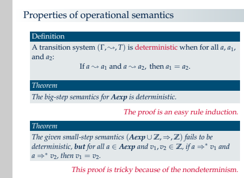
*The big-step semantics for **Aexp** is deterministic.*

The proof is an easy rule induction.

Theorem

*The given small-step semantics $(\mathbf{Aexp} \cup \mathbb{Z}, \Rightarrow, \mathbb{Z})$ fails to be deterministic, **but** for all $a \in \mathbf{Aexp}$ and $v_1, v_2 \in \mathbb{Z}$, if $a \Rightarrow^* v_1$ and $a \Rightarrow^* v_2$, then $v_1 = v_2$.*

This proof is tricky because of the nondeterminism.



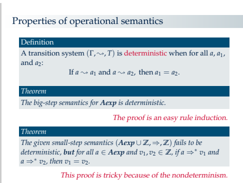
Theorem

*The given small-step semantics $(\mathbf{Aexp} \cup \mathbb{Z}, \Rightarrow, \mathbb{Z})$ fails to be deterministic, **but** for all $a \in \mathbf{Aexp}$ and $v_1, v_2 \in \mathbb{Z}$, if $a \Rightarrow^* v_1$ and $a \Rightarrow^* v_2$, then $v_1 = v_2$.*

Very sketchy proof-sketch.

- The argument is by induction on the number of operators (i.e., $+$, $-$, and $*$) occurring in a.
- **Base case:** a is a numeral, so it hasn't any operators and is a terminal expression. Hence if $a \Rightarrow^* v$, then $v = a$ is our only choice.
- Suppose by induction the theorem is true for all expressions of n or fewer operators and suppose $a = a_1 + a_2$ has $n + 1$ many operators.

  (The arguments for $a = a_1 - a_2$ and $a = a_1 * a_2$ will be similar.)



Very sketchy proof-sketch, continued.

- The a_1 and a_2 are expressions with n or fewer operators.
- The last step in any transition sequence $a \Rightarrow^* v$ is of the form $v_1 + v_2 \Rightarrow v$ and justified by $PLUS_3$.
- In each step before the last, the final rule in the justification of the step was either a $PLUS_1$ or a $PLUS_2$.
- If we look at the premises of the $PLUS_1$'s, they give a small-step derivation $a_1 \Rightarrow^* v_1$. By the IH, we know that any $\Rightarrow$-reduction sequence for a_1 that ends with a value *must* produce v_1.
- Similarly, $a_2 \Rightarrow^* v_2$ is also determined.
- So, it follows that if $a \Rightarrow^* v$, we must have $v = v_1 + v_2$.

A deterministic small-step semantics for **Aexp**

$$a \ ::= \ n \mid (a_1 + a_2) \mid (a_1 - a_2) \mid (a_1 * a_2) \mid v$$

$$\mathit{PLUS\text{-}1'_{SSS}}: \ \frac{a_1 \rightarrow a_1'}{a_1 + a_2 \rightarrow a_1' + a_2}$$

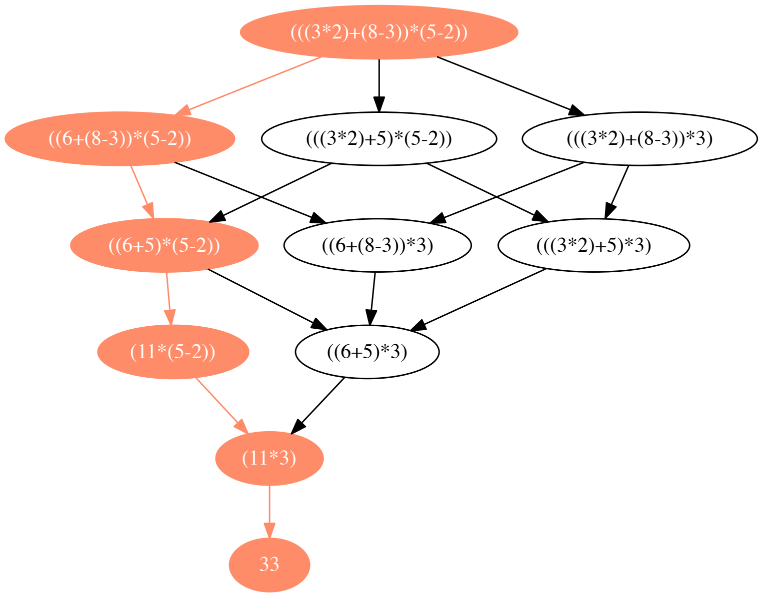
$$\mathit{PLUS\text{-}2'_{SSS}}: \ \frac{a_2 \rightarrow a_2'}{v_1 + a_2 \rightarrow v_1 + a_2'}$$

$$\mathit{PLUS\text{-}3'_{SSS}}: \ \frac{}{v_1 + v_2 \rightarrow v} \ (v = v_1 + v_2)$$

$$\vdots$$

$$\mathit{NUM_{SSS}}: \ \frac{}{n \rightarrow v} \ (\mathcal{N}[\![n]\!] = v)$$

The leftmost path through the lattice of transitions



Why multiple flavors of semantics?

They provide different views of computations.

- Big-step is good for reasoning about how the (big) pieces of things fit together.
- Small step is good at reasoning about the (small) steps of a computation fit together.
- Small step semantics is much better at modeling inherent nondeterminism (e.g., in concurrent programs).