Operational Semantics, Part II

Jim Royer

CIS 352

February 12, 2019

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 use a lot of his notation.
- The reading list for Matthew Hennessey's Introduction to the Semantics of programming languages course:
  https://www.scss.tcd.ie/Matthew.Hennessy/splexternal2015/reading.php
  has lots of good references.
- Also, Hennessey's notes for the above course
  https://www.scss.tcd.ie/Matthew.Hennessy/splexternal2015/LectureNotes/Notes14%20copy.pdf
  are very good.

LC: A tiny programming language

$$
\begin{array}{rrcl}
\text{Phases} & P & ::= & A \mid B \mid C \\
\text{Arithmetic Expressons} & A & ::= & n \mid\, !\ell \mid A \circledast A \qquad (\circledast \in \{+,-,\times,\dots\}) \\
\text{Boolean Expressons} & B & ::= & b \mid A \circledast A \qquad (\circledast \in \{=,<,\geq,\dots\}) \\
\text{Commands} & C & ::= & \textbf{skip} \mid \ell := A \mid C; C \\
 & & \mid & \textbf{if } B \textbf{ then } C \textbf{ else } C \mid \textbf{while } B \textbf{ do } C
\end{array}
$$

$$
\begin{array}{rrl}
\text{Integers} & n \in \mathbb{Z} & = \{\dots,-3,-2,-1,0,1,2,3,\dots\} \\
\text{Booleans} & b \in \mathbb{B} & = \{\text{true, false}\} \\
\text{Locations} & \ell \in \mathbb{L} & = \{\ell_0, \ell_1, \ell_2, \dots\} \\
 & & !\ell \equiv \text{the int. currently stored in } \ell
\end{array}
$$

$$\textbf{skip} \approx \text{do nothing} \approx \text{an } \epsilon\text{-move}$$

An Example LC Program

Returns factorial($!\ell_0$): Pitts' version

$$
\begin{array}{l}
\ell_1 \ \texttt{:=}\ \texttt{1;} \\
\ell_2 \ \texttt{:=}\ \texttt{!}\ell_0\texttt{;} \\
\texttt{while (!}\ell_2\texttt{>0) do} \\
\quad \ell_1 \ \texttt{:=}\ \texttt{!}\ell_1\texttt{*!}\ell_2\texttt{;} \\
\quad \ell_2 \ \texttt{:=}\ \texttt{!}\ell_2\texttt{-1}
\end{array}
$$

Returns factorial(val(x0)): Our version

```
{ x1 := 1;
  x2 := val(x0);
  while (val(x2)>0) do {
    x1 := val(x1)*val(x2);
    x2 := val(x2)-1
  }
}
```

- Pitts' $\ell_i \equiv$ our xi
- Pitts' $!\ell_i \equiv$ our `val`(xi)
- Pitts uses indenting for command bracketing.

- We use $\{ \ldots \}$ for command bracketing.
- His version takes up less space. Our version is easier to parse.

Big-step (evaluation) semantics for LC

States

A *state* is a finite mapping of locations to values.

E.g.: $[\ell_0 \mapsto 11,\ \ell_1 \mapsto 29, \ell_{17} \mapsto 5]$

Configurations

A *configuration* is a pair $\langle\, P, s \,\rangle$ where P is a phrase *&* s is a state.

E.g.: $\langle\, !\ell_{17} * 9 + !\ell_1,\ [\ell_0 \mapsto 11,\ \ell_1 \mapsto 29, \ell_{17} \mapsto 5] \,\rangle$

E.g.: $\langle\, \ell_0 := 8,\qquad [\ell_0 \mapsto 11,\ \ell_1 \mapsto 29, \ell_{17} \mapsto 5] \,\rangle$

Terminal configurations

The *terminal* configurations are those of the form:

$\langle\, n, s \,\rangle$, $\langle\, \text{true}, s \,\rangle$, $\langle\, \text{false}, s \,\rangle$, and $\langle\, \mathbf{skip}, s \,\rangle$.

$\Downarrow$: The LC evaluation relation

The *LC evaluation relation*

$$\Downarrow \;\subseteq\; (\text{Phrases} \times \text{States}) \times (\text{Phrases} \times \text{States})$$

is defined inductively as follows . . .

Note:

$\langle P, s \rangle \Downarrow \langle P', s' \rangle \quad \approx \quad$ the ***final*** result of evaluating $\langle P, s \rangle$ is $\langle P', s' \rangle$.

Definition of $\Downarrow$, 1

$\Downarrow$-*Con*: $$\frac{}{\langle\, c, s \,\rangle \Downarrow \langle\, c, s \,\rangle} \qquad (c \in \mathbb{Z} \cup \mathbb{B})$$

$\Downarrow$-$\circledast$: $$\frac{\langle\, A_1, s \,\rangle \Downarrow \langle\, n_1, s' \,\rangle \quad \langle\, A_2, s' \,\rangle \Downarrow \langle\, n_2, s'' \,\rangle}{\langle\, A_1 \circledast A_2, s \,\rangle \Downarrow \langle\, c, s'' \,\rangle} \qquad (c = n_1 \circledast n_2)$$

Above $\circledast$ can be:

- $+$, $-$, or $*$ for the arithmetic case, *or*
- ==, /=, <, >, <=, or >= for the boolean (comparison case).

Definition of $\Downarrow$, 2

$\Downarrow$*-Skip:*
$$\frac{}{\langle\, \mathbf{skip}, s \,\rangle \Downarrow \langle\, \mathbf{skip}, s \,\rangle}$$

$\Downarrow$*-Loc:*
$$\frac{}{\langle\, !\ell, s \,\rangle \Downarrow \langle\, s(\ell), s \,\rangle} \qquad (\ell \in \mathit{dom}(s))$$

$\Downarrow$*-Set:*
$$\frac{\langle\, A, s \,\rangle \Downarrow \langle\, n, s' \,\rangle}{\langle\, \ell := A, s \,\rangle \Downarrow \langle\, \mathbf{skip}, s'[\ell \mapsto n] \,\rangle}$$

Notation: $s[\ell \mapsto k]$ is a modification of state s such that:

- $(s[\ell \mapsto k])(\ell) = k.$
- $(s[\ell \rightarrow k])(\ell') = s(\ell')$, for $\ell' \neq \ell$.

E.g.: For $s = [\ell_0 \mapsto 12, \ell_1 \mapsto 3,\ \ell_2 \mapsto 9]$,
$s[\ell_1 \mapsto 20] = [\ell_0 \mapsto 12, \ell_1 \mapsto 20, \ell_2 \mapsto 9]$.

Digression: Sorts of Configurations

Suppose $\langle P, s \rangle$ is a configuration.

Stuck

$\langle P, s \rangle$ is *stuck* when there is no rule that applies to it.

E.g.: $\langle\, !\ell_1, \{\, \ell_0 \to 11 \,\}\, \rangle$.

Divergent

$\langle P, s \rangle$ is *divergent* when it is not stuck, but there is no finite derivation of $\langle P, s \rangle \Downarrow$ *something*.

E.g.: $\langle\, \textbf{while}\ \text{true}\ \textbf{do}\ \textbf{skip}, s \,\rangle$.

Terminating

$\langle P, s \rangle$ is *terminating* when it is neither stuck nor divergent.

Definition of $\Downarrow$, 3

$\Downarrow$-*Seq*:
$$\frac{\langle C_1, s \rangle \Downarrow \langle \mathbf{skip}, s' \rangle \quad \langle C_2, s' \rangle \Downarrow \langle \mathbf{skip}, s'' \rangle}{\langle C_1; C_2, s \rangle \Downarrow \langle \mathbf{skip}, s'' \rangle}$$

$\Downarrow$-*If*$_1$:
$$\frac{\langle B, s \rangle \Downarrow \langle \text{true}, s' \rangle \quad \langle C_1, s' \rangle \Downarrow \langle \mathbf{skip}, s'' \rangle}{\langle \mathbf{if}\ B\ \mathbf{then}\ C_1\ \mathbf{else}\ C_2, s \rangle \Downarrow \langle \mathbf{skip}, s'' \rangle}$$

$\Downarrow$-*If*$_2$:
$$\frac{\langle B, s \rangle \Downarrow \langle \text{false}, s' \rangle \quad \langle C_2, s' \rangle \Downarrow \langle \mathbf{skip}, s'' \rangle}{\langle \mathbf{if}\ B\ \mathbf{then}\ C_1\ \mathbf{else}\ C_2, s \rangle \Downarrow \langle \mathbf{skip}, s'' \rangle}$$

Definition of $\Downarrow$, 4

$\Downarrow$-*While*$_1$:

$$\frac{\langle B, s \rangle \Downarrow \langle \text{true}, s' \rangle \quad \langle C, s' \rangle \Downarrow \langle \textbf{skip}, s'' \rangle \quad \langle \textbf{while } B \textbf{ do } C, s'' \rangle \Downarrow \langle \textbf{skip}, s''' \rangle}{\langle \textbf{while } B \textbf{ do } C, s \rangle \Downarrow \langle \textbf{skip}, s''' \rangle}$$

$\Downarrow$-*While*$_2$:

$$\frac{\langle B, s \rangle \Downarrow \langle \text{false}, s' \rangle}{\langle \textbf{while } B \textbf{ do } C, s \rangle \Downarrow \langle \textbf{skip}, s' \rangle}$$

An Example from Pitts (page 30)

Let:

$$C \;\; =_{def} \;\; \mathbf{while}\ !\ell > 0\ \mathbf{do}\ \ell := 0 \qquad\qquad s \;\; =_{def} \;\; \{\, \ell \mapsto 1 \,\}$$

Then:

$$\dfrac{\dfrac{\dfrac{\quad}{\langle !\ell, s\rangle \Downarrow \langle 1, s\rangle}\,(\Downarrow_{\mathrm{loc}}) \quad \dfrac{\quad}{\langle 0, s\rangle \Downarrow \langle 0, s\rangle}\,(\Downarrow_{\mathrm{con}})}{\langle !\ell > 0, s\rangle \Downarrow \langle \mathbf{true}, s\rangle}\,(\Downarrow_{\mathrm{op}}) \quad \dfrac{\dfrac{\quad}{\langle 0, s\rangle \Downarrow \langle 0, s\rangle}\,(\Downarrow_{\mathrm{con}})}{\langle \ell := 0, s\rangle \Downarrow \langle \mathbf{skip}, s'\rangle}\,(\Downarrow_{\mathrm{set}}) \quad \dfrac{\dfrac{\dfrac{\quad}{\langle !\ell, s'\rangle \Downarrow \langle 0, s'\rangle}\,(\Downarrow_{\mathrm{loc}}) \quad \dfrac{\quad}{\langle 0, s'\rangle \Downarrow \langle 0, s'\rangle}\,(\Downarrow_{\mathrm{con}})}{\langle !\ell > 0, s'\rangle \Downarrow \langle \mathbf{false}, s'\rangle}\,(\Downarrow_{\mathrm{op}})}{\langle C, s'\rangle \Downarrow \langle \mathbf{skip}, s'\rangle}\,(\Downarrow_{\mathrm{wh2}})}{\langle C, s\rangle \Downarrow \langle \mathbf{skip}, s'\rangle}\,(\Downarrow_{\mathrm{wh1}}).$$

Big-step semantics as an implementation guide

See:

- `LC.hs`
- `LCbs.hs`

Exercise

Let:

$$\begin{aligned} C &=_{def} \textbf{while } B \textbf{ do } C' \\ B &=_{def} !\ell > 0 \\ C' &=_{def} \ell' := !\ell * !\ell';\ \ \ell := !\ell - 1 \\ s &=_{def} \{\, \ell \mapsto 3,\ \ell' \mapsto 1 \,\} \end{aligned}$$

Show (as much as you can stand of):

$$\langle\, C, s \,\rangle \Downarrow \langle\, \textbf{skip}, s[\ell \mapsto 0, \ell' \mapsto 6] \,\rangle.$$

Do these rules make sense?, 1

¿Theorem?

$(\forall\langle A, s\rangle)(\exists! c)[\langle A, s\rangle \Downarrow \langle c, s\rangle]$. ($\exists! \equiv$ *there exists a unique*)

Counterexample: $\langle\, !\ell_1,\ \{\, \ell_0 \mapsto 11 \,\}\,\rangle$ (since $\ell_1 \notin dom(s)$).

Definition

$\langle P, s\rangle$ is *sensible* when every location that occurs in P is in $dom(s)$.

¡Theorem!

a. *Suppose* $\langle A, s\rangle$ *is sensible. Then* $(\exists! c)[\langle A, s\rangle \Downarrow \langle c, s\rangle]$.
b. *Suppose* $\langle B, s\rangle$ *is sensible. Then* $(\exists! b)[\langle B, s\rangle \Downarrow \langle b, s\rangle]$.

$\exists! x \equiv$ there exists a unique x … [How to prove?]

Do these rules make sense?, 2

¿Theorem?

Suppose $\langle\, C, s \,\rangle$ *is sensible.* *Then* $(\exists! s')[\langle\, C, s \,\rangle \Downarrow \langle\, \mathbf{skip}, s' \,\rangle]$.

Counterexample: $C = \mathbf{while}$ true $\mathbf{do}$ $\mathbf{skip}$.

¡Theorem!

Suppose $\langle\, C, s \,\rangle$ *is sensible. Then:*

- a $\langle\, C, s \,\rangle$ *is not stuck.*
- b *There is* ***at most one*** s' *such that* $\langle\, C, s \,\rangle \Downarrow \langle\, \mathbf{skip}, s' \,\rangle$.

[How to prove?]

A CEK machine for LC

Abstract machines for interpreting LC: (Note: Abstract machine $\neq$ VM.)

- In §1.2 Pitts details an SMC (= Stack, Memory, Control) abstract machine for interpreting LC. *(Plotkin)*
- Here we sketch a CEK (= Context, Environment, Kontinuation) for interpreting LC. *(Felleisen and Friedman)*

CEK configurations: (c, s, ks)

c = the current phrase being evaluated

s = the state

ks = a "to-do" stack of things needed to complete pending evaluations. *(Examples forthcoming)*

See `LCCEK.hs`.

Digression: Transition systems

Definition

A transition system consists of

- a set (of states) S and
- a (transition) relation $\rightarrow \subseteq S \times S$.

The "states" can be configurations, game-board positions, etc.

Example

- Machines/computations
- Games/plays
- Protocols/runs
- ...

CEK Transitions

CEK configurations: (c, s, ks)

c = the current phrase being evaluated

s = the state

ks = a "to-do" stack of things needed to complete pending evaluations. *(Examples forthcoming)*

CEK transitions

$(c, \boxed{s}, ks) \rightsquigarrow (c', \boxed{s'}, ks')$ means:

according to the rules (forthcoming) configuration $(c, \boxed{s}, ks)$ can move to configuration $(c', \boxed{s'}, ks')$ in one step.

Note: The highlighted $\boxed{s}$'s are to make configurations easier to visually parse.

Integer expressions

$$(!\ell, \boxed{s}, ks) \leadsto (s(\ell), \boxed{s}, ks) \qquad (\ell \in dom(s))$$

$$\begin{aligned}
(e_1 \circledast e_2, \boxed{s}, ks) &\leadsto (e_1, \boxed{s}, (DoIOp1\ e_2\ \circledast) : ks) \\
(n_1, \boxed{s}, (DoIOp1\ e_2\ \circledast) : ks) &\leadsto (e_2, \boxed{s}, (DoIOp2\ \circledast\ n_1) : ks) \\
(n_2, \boxed{s}, (DoIOp2\ \circledast\ n_1) : ks) &\leadsto (n, \boxed{s}, ks) \qquad (n = n_1 \circledast n_2)
\end{aligned}$$

The big-step rules for integer expressions

$$\Downarrow\text{-}Loc: \quad \frac{}{\langle\, !\ell, s \,\rangle \Downarrow \langle\, s(\ell), s \,\rangle} \quad (\ell \in dom(s))$$

$$\Downarrow\text{-}\circledast: \quad \frac{\langle\, A_1, s \,\rangle \Downarrow \langle\, n_1, s' \,\rangle \quad \langle\, A_2, s' \,\rangle \Downarrow \langle\, n_2, s'' \,\rangle}{\langle\, A_1 \circledast A_2, s \,\rangle \Downarrow \langle\, c, s'' \,\rangle} \quad (c = n_1 \circledast n_2)$$

Exercise

Evaluate

$$\langle\, ((!\ell_1 + 2) * !\ell_2, [\ell_1 \mapsto 1, \ell_2 \mapsto 5]\,\rangle$$

by both big-step rule and the CEK.

☞ Notice how the CEK computation amounts to a stack-based traversal of the big-step derivation.

The set command

$$(\ell := a, \boxed{s}, ks) \rightsquigarrow (a, \boxed{s}, (DoSet\ \ell) : ks)$$

$$(n, \boxed{s}, (DoSet\ \ell) : ks) \rightsquigarrow (\mathbf{skip}, \boxed{s[\ell \mapsto n]}, ks)$$

The big-step rules for the set command

$$\Downarrow\text{-}Set: \quad \frac{\langle A, s \rangle \Downarrow \langle n, s' \rangle}{\langle \ell := A, s \rangle \Downarrow \langle \mathbf{skip}, s'[\ell \mapsto n] \rangle}$$

Sequencing

$$(C_1;\ C_2, \boxed{s}, ks) \rightsquigarrow (C_1, \boxed{s}, (\mathit{DoSeq}\ C_2) : ks)$$

$$(\mathbf{skip}, \boxed{s}, (\mathit{DoSeq}\ C_2) : ks) \rightsquigarrow (C_2, \boxed{s}, ks)$$

The big-step rules for sequencing

$$\Downarrow\text{-}\mathit{Seq}: \quad \frac{\langle\, C_1, s \,\rangle \Downarrow \langle\, \mathbf{skip}, s' \,\rangle \qquad \langle\, C_2, s' \,\rangle \Downarrow \langle\, \mathbf{skip}, s'' \,\rangle}{\langle\, C_1;\ C_2, s \,\rangle \Downarrow \langle\, \mathbf{skip}, s'' \,\rangle}$$

If-then-else

$$(\textbf{if } \mathit{be} \textbf{ then } C_1 \textbf{ else } C_2, \boxed{s}, ks) \rightsquigarrow (\mathit{be}, \boxed{s}, (\mathit{DoIf}\ C_1\ C_2) : ks)$$

$$(\text{true}, \boxed{s}, (\mathit{DoIf}\ C_1\ C_2) : ks) \rightsquigarrow (C_1, \boxed{s}, ks)$$

$$(\text{false}, \boxed{s}, (\mathit{DoIf}\ C_1\ C_2) : ks) \rightsquigarrow (C_2, \boxed{s}, ks)$$

The big-step rules for if-then-else

$$\Downarrow\text{-}\mathit{If}_1\!: \quad \frac{\langle B, s \rangle \Downarrow \langle \text{true}, s' \rangle \qquad \langle C_1, s' \rangle \Downarrow \langle \textbf{skip}, s'' \rangle}{\langle \textbf{if } B \textbf{ then } C_1 \textbf{ else } C_2, s \rangle \Downarrow \langle \textbf{skip}, s'' \rangle}$$

$$\Downarrow\text{-}\mathit{If}_1\!: \quad \frac{\langle B, s \rangle \Downarrow \langle \text{false}, s' \rangle \qquad \langle C_2, s' \rangle \Downarrow \langle \textbf{skip}, s'' \rangle}{\langle \textbf{if } B \textbf{ then } C_1 \textbf{ else } C_2, s \rangle \Downarrow \langle \textbf{skip}, s'' \rangle}$$

While

$$(\textbf{while } \mathit{be} \textbf{ do } C), \boxed{s}, ks)$$

$$\rightsquigarrow$$

$$(\textbf{if } \mathit{be} \textbf{ then } \{\, C;\ \textbf{while } \mathit{be} \textbf{ do } C \,\} \textbf{ else skip}, \boxed{s}, ks)$$

The big-step rules for if-then-else

$\Downarrow$-*While*$_1$:

$$\frac{\langle B, s \rangle \Downarrow \langle \text{true}, s' \rangle \quad \langle C, s' \rangle \Downarrow \langle \textbf{skip}, s'' \rangle \quad \langle \textbf{while } B \textbf{ do } C, s'' \rangle \Downarrow \langle \textbf{skip}, s''' \rangle}{\langle \textbf{while } B \textbf{ do } C, s \rangle \Downarrow \langle \textbf{skip}, s''' \rangle}$$

$\Downarrow$-*While*$_2$:

$$\frac{\langle B, s \rangle \Downarrow \langle \text{false}, s' \rangle}{\langle \textbf{while } B \textbf{ do } C, s \rangle \Downarrow \langle \textbf{skip}, s' \rangle}$$

Exercise

Let:

$$
\begin{aligned}
C &\;{\color{red}=_{def}}\; \textbf{while } !\ell > 0 \textbf{ do } \ell := 0 \\
s &\;{\color{red}=_{def}}\; \{\, \ell \mapsto 1 \,\}
\end{aligned}
$$

Trace the CEK evaluation of $\langle\, C, s \,\rangle$ and compare to:

$$
\dfrac{
\dfrac{
\dfrac{}{\langle !\ell, s\rangle \Downarrow \langle 1, s\rangle}\,(\Downarrow_{\text{loc}})
\quad
\dfrac{}{\langle 0, s\rangle \Downarrow \langle 0, s\rangle}\,(\Downarrow_{\text{con}})
}{\langle !\ell > 0, s\rangle \Downarrow \langle \textbf{true}, s\rangle}\,(\Downarrow_{\text{op}})
\quad
\dfrac{
\dfrac{}{\langle 0, s\rangle \Downarrow \langle 0, s\rangle}\,(\Downarrow_{\text{con}})
}{\langle \ell := 0, s\rangle \Downarrow \langle \textbf{skip}, s'\rangle}\,(\Downarrow_{\text{set}})
\quad
\dfrac{
\dfrac{
\dfrac{}{\langle !\ell, s'\rangle \Downarrow \langle 0, s'\rangle}\,(\Downarrow_{\text{loc}})
\quad
\dfrac{}{\langle 0, s'\rangle \Downarrow \langle 0, s'\rangle}\,(\Downarrow_{\text{con}})
}{\langle !\ell > 0, s'\rangle \Downarrow \langle \textbf{false}, s'\rangle}\,(\Downarrow_{\text{op}})
}{\langle C, s'\rangle \Downarrow \langle \textbf{skip}, s'\rangle}\,(\Downarrow_{\text{wh2}})
}{\langle C, s\rangle \Downarrow \langle \textbf{skip}, s'\rangle}\,(\Downarrow_{\text{wh1}}).
$$

Proof of equivalence with the big-step semantics

Theorem

For all $\langle P, s \rangle$ *and all terminal* $\langle V, s' \rangle$*:*

$$\langle P, s \rangle \Downarrow \langle V, s' \rangle \quad \Longleftrightarrow \quad \langle P, s, [Halt] \rangle \leadsto^* \langle V, s', [Halt] \rangle$$

Proof of $\Longrightarrow$.

Roughly, the CEK rules run a left-to-right traversal of the evaluation tree. $\square$

Proof of $\Longleftarrow$.

Key idea: Show that if $\langle P, s, ks \rangle \leadsto^* \langle V, s', ks \rangle$,
then you can reconstruct the evaluation tree for $\langle P, s \rangle \Downarrow \langle V, s \rangle$. $\square$

Small-step (transition) semantics of LC

The *LC transition relation*

$$\rightarrow \quad\subseteq\quad (\text{Phrases} \times \text{States}) \times (\text{Phrases} \times \text{States})$$

is defined inductively as follows . . .

Note:

$\langle\, P, s \,\rangle \rightarrow \langle\, P', s' \,\rangle \quad\approx\quad \langle\, P, s \,\rangle$ "rewrites" to $\langle\, P', s' \,\rangle$ in one step.

Definition of $\rightarrow$, 1

$\rightarrow$*-op1:* $$\frac{\langle A_1, s \rangle \rightarrow \langle A_1', s' \rangle}{\langle A_1 \circledast A_2, s \rangle \rightarrow \langle A_1' \circledast A_2, s' \rangle}$$

$\rightarrow$*-op2:* $$\frac{\langle A_2, s \rangle \rightarrow \langle A_2', s' \rangle}{\langle n_1 \circledast A_2, s \rangle \rightarrow \langle n_1 \circledast A_2', s' \rangle}$$

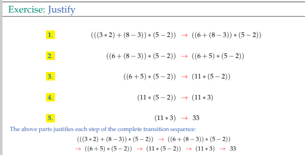
$\rightarrow$*-op3:* $$\frac{}{\langle n_1 \circledast n_2, s \rangle \rightarrow \langle c, s \rangle} \qquad (c = n_1 \circledast n_2)$$

Exercise: Justify

1. $(((3*2)+(8-3))*(5-2)) \rightarrow ((6+(8-3))*(5-2))$
2. $((6+(8-3))*(5-2)) \rightarrow ((6+5)*(5-2))$
3. $((6+5)*(5-2)) \rightarrow (11*(5-2))$
4. $(11*(5-2)) \rightarrow (11*3)$
5. $(11*3) \rightarrow 33$

The above parts justifies each step of the complete transition sequence:

$$\begin{aligned}&(((3*2)+(8-3))*(5-2)) \rightarrow ((6+(8-3))*(5-2))\\ &\rightarrow ((6+5)*(5-2)) \rightarrow (11*(5-2)) \rightarrow (11*3) \rightarrow 33\end{aligned}$$



Exercise: Justify

1. $(((3*2)+(8-3))*(5-2)) \rightarrow ((6+(8-3))*(5-2))$
2. $((6+(8-3))*(5-2)) \rightarrow ((6+5)*(5-2))$
3. $((6+5)*(5-2)) \rightarrow (11*(5-2))$
4. $(11*(5-2)) \rightarrow (11*3)$
5. $(11*3) \rightarrow 33$

The above parts justifies each step of the complete transition sequence:

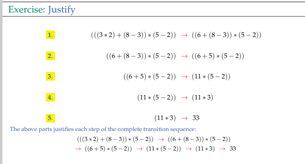
$$(((3*2)+(8-3))*(5-2)) \rightarrow ((6+(8-3))*(5-2)) \rightarrow ((6+5)*(5-2)) \rightarrow (11*(5-2)) \rightarrow (11*3) \rightarrow 33$$

Answer to 1.

$$\rightarrow\textit{-op1: }\frac{\rightarrow\textit{-op1: }\dfrac{\rightarrow\textit{-op3: }\dfrac{}{(3*2) \rightarrow 6}}{((3*2)+(8-3)) \rightarrow (6+(8-3))}}{(((3*2)+(8-3))*(5-2)) \rightarrow ((6+(8-3))*(5-2))}$$

Answer to 2.

$$\rightarrow\textit{-op1: }\frac{\rightarrow\textit{-op2: }\dfrac{\rightarrow\textit{-op3: }\dfrac{}{(8-3) \rightarrow 5}}{(6+(8-3)) \rightarrow (6+5)}}{((6+(8-3))*(5-2)) \rightarrow ((6+5)*(5-2))}$$



Answer to 3.

$$\rightarrow\textit{-op1:}\ \dfrac{\rightarrow\textit{-op3:}\ \dfrac{}{(6+5) \rightarrow 11}}{((6+5)*(5-2)) \rightarrow (11*(5-2))}$$

Answer to 4.

$$\rightarrow\textit{-op2:}\ \dfrac{\rightarrow\textit{-op3:}\ \dfrac{}{(5-2) \rightarrow 3}}{(11*(5-2)) \rightarrow (11*3)}$$

Answer to 5.

$$\rightarrow\textit{-op3:}\ \dfrac{}{(11*3) \rightarrow 33}$$

Definition of $\rightarrow$, 2

$\rightarrow$*-loc:*
$$\frac{}{\langle\, !\ell, s \,\rangle \rightarrow \langle\, s(\ell), s \,\rangle} \qquad (\ell \in dom(s))$$

$\rightarrow$*-set1:*
$$\frac{\langle\, A, s \,\rangle \rightarrow \langle\, A', s' \,\rangle}{\langle\, \ell := A, s \,\rangle \rightarrow \langle\, \ell := A', s' \,\rangle}$$

$\rightarrow$*-set2:*
$$\frac{}{\langle\, \ell := n, s \,\rangle \rightarrow \langle\, \mathbf{skip}, s[\ell \mapsto n] \,\rangle}$$

Definition of $\rightarrow$, 3

$\rightarrow$*-seq1:*

$$\frac{\langle C_1, s \rangle \rightarrow \langle C_1', s' \rangle}{\langle C_1;\ C_2, s \rangle \rightarrow \langle C_1';\ C_2, s' \rangle}$$

$\rightarrow$*-seq2:*

$$\frac{}{\langle \mathbf{skip};\ C, s \rangle \rightarrow \langle C, s \rangle}$$

$\rightarrow$*-while:*

$$\frac{}{\langle \mathbf{while}\ B\ \mathbf{do}\ C, s \rangle \rightarrow \langle \mathbf{if}\ B\ \mathbf{then}\ \{C;\ \mathbf{while}\ B\ \mathbf{do}\ C\ \}\ \mathbf{else}\ \mathbf{skip}, s \rangle}$$

Definition of $\rightarrow$, 4

$\rightarrow$*-if1:*

$$\frac{\langle B, s\rangle \rightarrow \langle B', s'\rangle}{\langle \textbf{if } B \textbf{ then } C_1 \textbf{ else } C_2, s\rangle \rightarrow \langle \textbf{if } B' \textbf{ then } C_1 \textbf{ else } C_2, s'\rangle}$$

$\rightarrow$*-if2:*

$$\frac{}{\langle \textbf{if } \text{true} \textbf{ then } C_1 \textbf{ else } C_2, s\rangle \rightarrow \langle C_1, s\rangle}$$

$\rightarrow$*-if3:*

$$\frac{}{\langle \textbf{if } \text{false} \textbf{ then } C_1 \textbf{ else } C_2, s\rangle \rightarrow \langle C_2, s\rangle}$$

A sample transition, 1

Let:

$$C =_{def} \textbf{while } B \textbf{ do } C' \qquad B =_{def} \,!\ell > 0$$

$$C' =_{def} \ell' := !\ell * !\ell';\ \ \ell := !\ell - 1 \qquad s =_{def} \{\ell \mapsto 3, \ell' \mapsto 1\}$$

The start of the full transition

$$\begin{aligned}
\langle C, s\rangle &\to \langle \textbf{if } B \textbf{ then } \{C';\ C\}\textbf{else skip}, s\rangle \\
&\to \langle \textbf{if } 3 > 0 \textbf{ then } \{C';\ C\}\textbf{else skip}, s\rangle \\
&\to \langle \textbf{if true then } \{C';\ C\}\textbf{else skip}, s\rangle \\
&\to \langle C';\ C, s\rangle \\
&\qquad \textit{\ldots after 40 some steps \ldots} \\
&\to \langle \textbf{skip}, s[\ell \mapsto 0, \ell' \mapsto 6]\rangle.
\end{aligned}$$

Note: Each step of a transition must be justified by a derivation.

A sample transition, 2

Let:

$$C =_{def} \textbf{while } B \textbf{ do } C' \qquad B =_{def} !\ell > 0$$

$$C' =_{def} \ell' := !\ell * !\ell';\ \ell := !\ell - 1 \qquad s =_{def} \{\ell \mapsto 3,\ \ell' \mapsto 1\}$$

Note: Each step of a transition must be justified by a derivation.

Exercise: Justify

1. $\langle C, s \rangle \to \langle \textbf{if } B \textbf{ then } \{C';\ C\} \textbf{else skip}, s \rangle$

2. $\langle \textbf{if } B \textbf{ then } \{C';\ C\} \textbf{else skip}, s \rangle \to \langle \textbf{if } 3 > 0 \textbf{ then } \{C';\ C\} \textbf{else skip}, s \rangle$

3. $\langle \textbf{if } 3 > 0 \textbf{ then } \{C';\ C\} \textbf{else skip}, s \rangle$
   $\to \langle \textbf{if true then } \{C';\ C\} \textbf{else skip}, s \rangle$

4. $\langle \textbf{if true then } \{C';\ C\} \textbf{else skip}, s \rangle \to \langle C';\ C, s \rangle$

A sample transition, 2

Let:

$$\begin{array}{lll@{\qquad}lll} C & =_{def} & \textbf{while } B \textbf{ do } C' & B & =_{def} & !\ell > 0 \\ C' & =_{def} & \ell' := !\ell * !\ell';\ \ \ell := !\ell - 1 & s & =_{def} & \{\, \ell \mapsto 3, \ell' \mapsto 1 \,\} \end{array}$$

Note: Each step of a transition must be justified by a derivation.

Exercise: Justify

1. $\langle\, C, s \,\rangle \rightarrow \langle\, \textbf{if } B \textbf{ then } \{C';\ C\} \textbf{else skip}, s \,\rangle$
2. $\langle\, \textbf{if } B \textbf{ then } \{C';\ C\} \textbf{else skip}, s \,\rangle \rightarrow \langle\, \textbf{if } 3 > 0 \textbf{ then } \{C';\ C\} \textbf{else skip}, s \,\rangle$
3. $\langle\, \textbf{if } 3 > 0 \textbf{ then } \{C';\ C\} \textbf{else skip}, s \,\rangle$ $\rightarrow \langle\, \textbf{if true then } \{C';\ C\} \textbf{else skip}, s \,\rangle$
4. $\langle\, \textbf{if true then } \{C';\ C\} \textbf{else skip}, s \,\rangle \rightarrow \langle\, C';\ C, s \,\rangle$

Answer to 1.

$$\rightarrow\textit{-while: } \frac{}{\langle\, C, s \,\rangle \rightarrow \langle\, \textbf{if } B \textbf{ then } \{C';\ C\} \textbf{else skip}, s \,\rangle}$$

Answer to 2. Recall $B =_{def} !\ell > 0$.

$$\rightarrow\textit{-if1: } \frac{\rightarrow\textit{-op1: } \dfrac{\rightarrow\textit{-loc: } \dfrac{}{\langle\, !\ell, s \,\rangle \rightarrow \langle\, 3, s \,\rangle}\ (\text{since } s(\ell) = 3)}{\langle\, B, s \,\rangle \rightarrow \langle\, 3 > 0, s \,\rangle}}{\langle\, \textbf{if } B \textbf{ then } \{C';\ C\} \textbf{else skip}, s \,\rangle \rightarrow \langle\, \textbf{if } 3 > 0 \textbf{ then } \{C';\ C\} \textbf{else skip}, s \,\rangle}$$

A sample transition, 2

Let:

$$C =_{def} \textbf{while } B \textbf{ do } C' \qquad B =_{def} \,!\ell > 0$$

$$C' =_{def} \ell' := !\ell * !\ell';\ \ell := !\ell - 1 \qquad s =_{def} \{\ell \mapsto 3, \ell' \mapsto 1\}$$

Note: Each step of a transition must be justified by a derivation.

Exercise: Justify

1. $\langle C, s \rangle \to \langle \textbf{if } B \textbf{ then } \{C';\ C\}\textbf{else skip},\ s \rangle$
2. $\langle \textbf{if } B \textbf{ then } \{C';\ C\}\textbf{else skip},\ s \rangle \to \langle \textbf{if } 3 > 0 \textbf{ then } \{C';\ C\}\textbf{else skip},\ s \rangle$
3. $\langle \textbf{if } 3 > 0 \textbf{ then } \{C';\ C\}\textbf{else skip},\ s \rangle$
   $\to \langle \textbf{if true then } \{C';\ C\}\textbf{else skip},\ s \rangle$
4. $\langle \textbf{if true then } \{C';\ C\}\textbf{else skip},\ s \rangle \to \langle C';\ C,\ s \rangle$

Answer to 3.

$$\to\textit{-if1: } \dfrac{\to\textit{-op3: } \dfrac{}{\langle 3 > 0, s \rangle \to \langle \textbf{true}, s \rangle}\ \text{(since } 3 > 0 \text{ is true)}}{\langle \textbf{if } 3 > 0 \textbf{ then } \{C';\ C\}\textbf{else skip},\ s \rangle \to \langle \textbf{if true then } \{C';\ C\}\textbf{else skip},\ s \rangle}$$

Answer to 4.

$$\to\textit{-if2: } \dfrac{}{\langle \textbf{if true then } \{C';\ C\}\textbf{else skip},\ s \rangle \to \langle \{C';\ C\}, s \rangle}$$

Some properties of $\rightarrow$

Theorem (Determinacy)

If $\langle P, s \rangle$ is neither stuck nor terminal, then $(\exists! \langle P', s' \rangle)[\langle P, s \rangle \rightarrow \langle P', s' \rangle]$.

Theorem (Subject reduction)

If $\langle P, s \rangle \rightarrow \langle P', s' \rangle$, then P and P' are the same type (i.e., command, integer-expression, boolean-expression).

Theorem (Expressions have no side-effects)

If P is an integer or boolean expression and $\langle P, s \rangle \rightarrow \langle P', s' \rangle$, then $s = s'$.

[How to prove?]

Equivalence of the big-step & small-step semantics

Theorem

For all $\langle P, s \rangle$ *and all terminal* $\langle V, s' \rangle$*:*

$$\langle P, s \rangle \Downarrow \langle V, s' \rangle \iff \langle P, s \rangle \rightarrow^{*} \langle V, s' \rangle$$

Proof.

One needs to show:

a $\langle P, s \rangle \Downarrow \langle V, s' \rangle \implies \langle P, s \rangle \rightarrow^{*} \langle V, s' \rangle$.

b $\langle P, s \rangle \rightarrow \langle P', s' \rangle \;\&\; \langle P', s' \rangle \Downarrow \langle V, s'' \rangle \implies \langle P, s \rangle \Downarrow \langle V, s'' \rangle$.

c $\langle P, s \rangle \rightarrow^{*} \langle V, s' \rangle \implies \langle P, s \rangle \Downarrow \langle V, s' \rangle$.

□