Overview

Transition Semantics

- Configurations and the transition relation
- Executions and computation
- Inference rules for small-step structural operational semantics for the simple imperative language
- Transition semantics of failure
- Labeled transition semantics of input and output
- Relationship with (direct) denotational semantics

Operational (or Transition) Semantics

Idea: Define the execution of a program as a sequence $\gamma_0, \gamma_1, \ldots$ of configurations $\gamma_i \in \Gamma$.

Configurations are either terminal or nonterminal:

$$\Gamma = \Gamma_T \cup \Gamma_N \qquad \Gamma_T \cap \Gamma_N = \{\}$$

e.g. for the SIL

$$\Gamma_T = \Sigma \qquad \Gamma_N = comm \times \Sigma$$

$$[x : 42] \in \Gamma_T \qquad \langle \texttt{x:=x+1}, [x : 41] \rangle \in \Gamma_N$$

Define a transition relation $\mapsto$ from Γ_N to Γ:

informally, $\gamma \mapsto \gamma'$ if γ' is obtained "in one step" from γ, e.g.

$$\langle \texttt{x:=x+1}, [x : 41] \rangle \mapsto [x : 42]$$

Executions and Computation

An execution is a (finite or infinite) sequence of configurations $\gamma_0, \gamma_1, \ldots$ such that $\gamma_i \mapsto \gamma_{i+1}$ whenever γ_i and γ_{i+1} are in the sequence.

The relation of computation $\mapsto^*$

is the reflexive and transitive closure of $\mapsto$

$\gamma \mapsto^* \gamma'$ if there is a finite execution starting with γ and ending with γ'.

For the SIL we will define $\mapsto$ which is a total function from Γ_N to Γ,

$\Rightarrow$ for every $\gamma \in \Gamma$ there is a longest execution starting with γ;

if it is infinite, then γ diverges: $\gamma \uparrow$;

otherwise there is a unique $\gamma' \in \Gamma_T$ such that $\gamma \mapsto^* \gamma'$.

Plotkin Style Small-Step Structural Operational Semantics for the SIL

We define the relation $\mapsto$ in terms of inference rules.

$$\text{(skip)}\quad \frac{}{\langle \mathbf{skip},\ \sigma\rangle \mapsto \sigma}$$

$$\text{(assgn)}\quad \frac{}{\langle v := e,\ \sigma\rangle \mapsto [\sigma \mid v : [\![e]\!]_{intexp}\sigma]}$$

$$\text{(seq t)}\quad \frac{\langle c_0,\ \sigma\rangle \mapsto \sigma'}{\langle c_0\ ;\ c_1,\ \sigma\rangle \mapsto \langle c_1,\ \sigma'\rangle}$$

$$\text{(seq s)}\quad \frac{\langle c_0,\ \sigma\rangle \mapsto \langle c_0',\ \sigma'\rangle}{\langle c_0\ ;\ c_1,\ \sigma\rangle \mapsto \langle c_0'\ ;\ c_1,\ \sigma'\rangle}$$

Example:

$$\text{(by (seq s))}\quad \dfrac{\text{(by (seq t))}\quad \dfrac{\text{(by (assgn))}\quad \dfrac{}{\langle \mathsf{x := x+1},\ [\mathsf{x} : 4 \mid \mathsf{y} : 6]\rangle \mapsto [\mathsf{x} : 5 \mid \mathsf{y} : 6]}}{\langle \mathsf{x := x+1}\ ;\ \mathsf{y := y+x},\ [\mathsf{x} : 4 \mid \mathsf{y} : 6]\rangle \mapsto \langle \mathsf{y := y+x},\ [\mathsf{x} : 5 \mid \mathsf{y} : 6]\rangle}}{\langle \mathsf{x := x+1}\ ;\ \mathsf{y := y+x}\ ;\ \mathbf{skip},\ [\mathsf{x} : 4 \mid \mathsf{y} : 6]\rangle \mapsto \langle \mathsf{y := y+x}\ ;\ \mathbf{skip},\ [\mathsf{x} : 5 \mid \mathsf{y} : 6]\rangle}$$

More SOS Rules

(cond t) $$\frac{}{\langle \textbf{if } b \textbf{ then } c \textbf{ else } c',\ \sigma\rangle \mapsto \langle c,\ \sigma\rangle}$$ when $[\![b]\!]_{boolexp}\sigma = \mathit{true}$

(cond f) $$\frac{}{\langle \textbf{if } b \textbf{ then } c \textbf{ else } c',\ \sigma\rangle \mapsto \langle c',\ \sigma\rangle}$$ when $[\![b]\!]_{boolexp}\sigma = \mathit{false}$

(while t) $$\frac{}{\langle \textbf{while } b \textbf{ do } c,\ \sigma\rangle \mapsto \langle c \,;\, \textbf{while } b \textbf{ do } c,\ \sigma\rangle}$$ when $[\![b]\!]_{boolexp}\sigma = \mathit{true}$

(while f) $$\frac{}{\langle \textbf{while } b \textbf{ do } c,\ \sigma\rangle \mapsto \sigma}$$ when $[\![b]\!]_{boolexp}\sigma = \mathit{false}$

However the naïve rule for variable declaration

$$\frac{}{\langle \textbf{newvar } v := e \textbf{ in } c,\ \sigma\rangle \mapsto \langle c \,;\, v := n,\ [\sigma \mid v : [\![e]\!]_{intexp}\sigma]\rangle}$$ where $n = \sigma\, v$

exposes the local variable name in the result,

which becomes a problem when we extend the language.

SOS Rule for Local Variable Declaration

Idea: Use the declaration to reflect changes in the value of the variable.

$$(\text{decl t})\ \frac{\langle c,\ [\sigma \mid v : [\![e]\!]_{intexp}\sigma]\rangle \mapsto \sigma'}{\langle \textbf{newvar}\ v := e\ \textbf{in}\ c,\ \sigma\rangle \mapsto [\sigma' \mid v : \sigma\, v]}$$

$$(\text{decl s})\ \frac{\langle c,\ [\sigma \mid v : [\![e]\!]_{intexp}\sigma]\rangle \mapsto \langle c',\ \sigma'\rangle}{\langle \textbf{newvar}\ v := e\ \textbf{in}\ c,\ \sigma\rangle \mapsto \langle \textbf{newvar}\ v := \sigma'\, v\ \textbf{in}\ c',\ [\sigma' \mid v : \sigma\, v]\rangle}$$

$$(\text{decl s})\ \frac{(\text{seq t})\ \dfrac{(\text{assgn})\ \dfrac{}{\langle \mathsf{x := x+1},\ [\mathsf{x} : 24 \mid \mathsf{y} : 10]\rangle \mapsto [\mathsf{x} : 25 \mid \mathsf{y} : 10]}}{\langle \mathsf{x := x+1\ ;\ y := x+2},\ [\mathsf{x} : 24 \mid \mathsf{y} : 10]\rangle \mapsto \langle \mathsf{y := x+2},\ [\mathsf{x} : 25 \mid \mathsf{y} : 10]\rangle}}{\begin{array}{l}\langle \textbf{newvar}\ \mathsf{x := x+3}\ \textbf{in}\ \mathsf{x := x+1\ ;\ y := x+2},\ [\mathsf{x} : 21 \mid \mathsf{y} : 10]\rangle \\ \mapsto \langle \textbf{newvar}\ \mathsf{x := 25}\ \textbf{in}\ \mathsf{y := x+2},\ [\mathsf{x} : 21 \mid \mathsf{y} : 10]\rangle\end{array}}$$

Inference Rules for the Computation

The reflexive and transitive closure of $\mapsto$ can also be defined using inference rules:

$$\text{(incl)}\quad \frac{\gamma \mapsto \gamma'}{\gamma \mapsto^* \gamma'}$$

$$\text{(refl)}\quad \frac{}{\gamma \mapsto^* \gamma}$$

$$\text{(trans)}\quad \frac{\gamma \mapsto^* \gamma' \quad \gamma' \mapsto^* \gamma''}{\gamma \mapsto^* \gamma''}$$

Meaning of Commands

$\mapsto \in \Gamma_N \to \Gamma$ (total function)

$\Rightarrow$ $\forall \gamma \in \Gamma$ there is a longest execution starting from γ,

either infinite or ending with a $\gamma' \in \Gamma_T = \Sigma$.

$$[\![c]\!]_{comm}\sigma = \begin{cases} \bot, & \text{if } \langle c, \sigma \rangle \uparrow \\ \sigma', & \text{if } \langle c, \sigma \rangle \mapsto^* \sigma' \end{cases}$$

Transition Semantics of Failure

Define $\Gamma_T = \Sigma \cup (\{\mathbf{abort}\} \times \Sigma)$. Then

$$\text{(fail)} \quad \frac{}{\langle \mathbf{fail},\ \sigma\rangle \mapsto \langle \mathbf{abort},\ \sigma\rangle}$$

Propagation of failure:

$$\text{(seq x)} \quad \frac{\langle c_0,\ \sigma\rangle \mapsto \langle \mathbf{abort},\ \sigma\rangle}{\langle c_0\ ;\ c_1,\ \sigma\rangle \mapsto \langle \mathbf{abort},\ \sigma'\rangle}$$

$$\text{(decl x)} \quad \frac{\langle c,\ [\sigma \mid v : [\![e]\!]_{intexp}\sigma]\rangle \mapsto \langle \mathbf{abort},\ \sigma'\rangle}{\langle \mathbf{newvar}\ v\,\mathbf{:=}\,e\ \mathbf{in}\ c,\ \sigma\rangle \mapsto \langle \mathbf{abort},\ [\sigma' \mid v : \sigma\, v]\rangle}$$

The semantics of commands becomes

$$[\![c]\!]_{comm}\sigma = \begin{cases} \bot, & \text{if } \langle c,\ \sigma\rangle \uparrow \\ \sigma', & \text{if } \langle c,\ \sigma\rangle \mapsto^* \sigma' \\ \langle \mathbf{abort}\ \sigma'\rangle, & \text{if } \langle c,\ \sigma\rangle \mapsto^* \langle \mathbf{abort}\ \sigma'\rangle \end{cases}$$

Labeled Transition Semantics of Input and Output

Informally: Write labels on transitions to show input or output.

Rules:

$$\text{(output)} \quad \frac{}{\langle \mathbf{!}e,\ \sigma\rangle \overset{!n}{\mapsto} \sigma} \qquad \text{when } n = [\![e]\!]_{intexp}\sigma$$

$$\text{(input)} \quad \frac{}{\langle \mathbf{?}v,\ \sigma\rangle \overset{?n}{\mapsto} [\sigma \mid v : n]}$$

Formally, the transition "relation" becomes ternary:

$\mapsto \subseteq \Gamma_N \times \Lambda \times \Gamma$, where

$$\Lambda = \{\epsilon\} \cup \{?n \mid n \in \mathbf{Z}\} \cup \{!n \mid n \in \mathbf{Z}\} \qquad (\epsilon \text{ is silent})$$

and $\langle c,\ \sigma\rangle \overset{\lambda}{\mapsto} \gamma$ stands for $\langle\langle c,\ \sigma\rangle,\ \lambda,\ \gamma\rangle \in \mapsto$.

Labeled Transition Semantics cont'd

The other rules are generalized to propagate the labels, e.g.

$$\text{(seq t)}\quad \frac{\langle c_0,\ \sigma\rangle \overset{\lambda}{\mapsto} \sigma'}{\langle c_0\ ;\ c_1,\ \sigma\rangle \overset{\lambda}{\mapsto} \langle c_1,\ \sigma'\rangle}$$

$$\text{(seq s)}\quad \frac{\langle c_0,\ \sigma\rangle \overset{\lambda}{\mapsto} \langle c_0',\ \sigma'\rangle}{\langle c_0\ ;\ c_1,\ \sigma\rangle \overset{\lambda}{\mapsto} \langle c_0'\ ;\ c_1,\ \sigma'\rangle}$$

$$\text{(seq x)}\quad \frac{\langle c_0,\ \sigma\rangle \overset{\lambda}{\mapsto} \langle \mathbf{abort},\ \sigma\rangle}{\langle c_0\ ;\ c_1,\ \sigma\rangle \overset{\lambda}{\mapsto} \langle \mathbf{abort},\ \sigma'\rangle}$$

Properties of the Labeled Transition Semantics

If $\gamma = \langle c, \sigma \rangle \in \Gamma_N$, then exactly one of these holds:

- $\exists ! \gamma' \in \Gamma$ such that $\gamma \mapsto \gamma'$ (silent transition)
- $\exists ! \gamma' \in \Gamma, n \in \mathbf{Z}$ such that $\gamma \overset{!n}{\mapsto} \gamma'$
- $\{\lambda \in \Lambda \,|\, \gamma' \in \Gamma \text{ and } \gamma \overset{\lambda}{\mapsto} \gamma'\} = \{?n \,|\, n \in \mathbf{Z}\}$

Hence for every $\gamma \in \Gamma$ there is a longest sequence of silent transitions which is either

- infinite
- ends with a $\gamma' \in \Gamma_T$
- ends with a $\gamma' \in \Gamma_N$ such that $\exists ! \gamma'' \in \Gamma, n \in \mathbf{Z}$ such that $\gamma' \overset{!n}{\mapsto} \gamma''$
- ends with a $\gamma' \in \Gamma_N$ such that $\forall n \in \mathbf{Z}.\ \exists \gamma'' \in \Gamma.\ \gamma' \overset{?n}{\mapsto} \gamma''$.

Relationship with the Denotational Semantics

So, for every $\gamma \in \Gamma$ there is a longest sequence of silent transitions which is either

- infinite
- ends with a $\gamma' \in \Gamma_T$
- ends with a $\gamma' \in \Gamma_N$ such that $\exists! \gamma'' \in \Gamma, n \in \mathbf{Z}$ such that $\gamma' \stackrel{!n}{\mapsto} \gamma''$
- ends with a $\gamma' \in \Gamma_N$ such that $\forall n \in \mathbf{Z}.\, \exists \gamma'' \in \Gamma.\, \gamma' \stackrel{?n}{\mapsto} \gamma''$.

If $\Omega \cong (\hat{\Sigma} + (\mathbf{Z} \times \Omega) + [\mathbf{Z} \to \Omega])_\perp$ and $F \in [\Gamma \to \Omega]$ is the least solution of

$$F\,\gamma = \begin{cases} \perp, & \text{if } \gamma \uparrow \\ \iota_{\mathsf{term}}\,\sigma', & \text{if } \gamma \mapsto^* \sigma' \\ \iota_{\mathsf{abort}}\,\sigma', & \text{if } \gamma \mapsto^* \langle \mathbf{abort},\ \sigma' \rangle \\ \iota_{\mathsf{out}}\,\langle n,\ F\,\gamma'' \rangle, & \text{if } \exists \gamma'.\, \gamma \mapsto^* \gamma' \text{ and } \gamma' \stackrel{!n}{\mapsto} \gamma'' \\ \iota_{\mathsf{in}}\,(\lambda n \in \mathbf{Z}.\, F\,\gamma_n), & \text{if } \exists \gamma' \in \Gamma.\, \forall n \in \mathbf{Z}.\, \gamma \mapsto^* \gamma' \text{ and } \gamma' \stackrel{?n}{\mapsto} \gamma_n \end{cases}$$

then $[\![c]\!]_{comm}\sigma = F\,\langle c,\ \sigma \rangle$.