ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

D. Basin and P. Müller

Formal Methods and Functional Programming

Solutions of Exercise Sheet 11: Small Step Semantics

Assignment 1: Implementing SOS

You find a solution of this assignment in the literate Haskell file `simpi.lhs`.

Assignment 2: SOS derivation sequence

Let `s'` be the body of the loop.

$$
\begin{aligned}
\langle \mathtt{s}, \sigma\rangle &\to_1 \langle \mathtt{if\ n\ \#\ 0\ then\ s';\ s\ else\ skip\ end}, \sigma\rangle\\
&\to_1 \langle \mathtt{((a\ :=\ a+n;\ b\ :=\ b*n);\ n\ :=\ n-1);\ s}, \sigma\rangle\\
&\to_1 \langle \mathtt{(b\ :=\ b*n;\ n\ :=\ n-1);\ s}, \sigma[\mathtt{a} \mapsto 2]\rangle\\
&\to_1 \langle \mathtt{n\ :=\ n-1;\ s}, \sigma[\mathtt{a}, \mathtt{b} \mapsto 2, 2]\rangle\\
&\to_1 \langle \mathtt{s}, \sigma[\mathtt{a}, \mathtt{b}, \mathtt{n} \mapsto 2, 2, 1]\rangle\\
&\to_1 \langle \mathtt{if\ n\ \#\ 0\ then\ s';\ s\ else\ skip\ end}, \sigma[\mathtt{a}, \mathtt{b}, \mathtt{n} \mapsto 2, 2, 1]\rangle\\
&\to_1 \langle \mathtt{((a\ :=\ a+n;\ b\ :=\ b*n);\ n\ :=\ n-1);\ s}, \sigma[\mathtt{a}, \mathtt{b}, \mathtt{n} \mapsto 2, 2, 1]\rangle\\
&\to_1 \langle \mathtt{(b\ :=\ b*n;\ n\ :=\ n-1);\ s}, \sigma[\mathtt{a}, \mathtt{b}, \mathtt{n} \mapsto 3, 2, 1]\rangle\\
&\to_1 \langle \mathtt{n\ :=\ n-1;\ s}, \sigma[\mathtt{a}, \mathtt{b}, \mathtt{n} \mapsto 3, 2, 1]\rangle\\
&\to_1 \langle \mathtt{s}, \sigma[\mathtt{a}, \mathtt{b}, \mathtt{n} \mapsto 3, 2, 0]\rangle\\
&\to_1 \langle \mathtt{if\ n\ \#\ 0\ then\ s';\ s\ else\ skip\ end}, \sigma[\mathtt{a}, \mathtt{b}, \mathtt{n} \mapsto 3, 2, 0]\rangle\\
&\to_1 \langle \mathtt{skip}, \sigma[\mathtt{a}, \mathtt{b}, \mathtt{n} \mapsto 3, 2, 0]\rangle\\
&\to_1 \sigma[\mathtt{a}, \mathtt{b}, \mathtt{n} \mapsto 3, 2, 0]
\end{aligned}
$$

Assignment 3: Composing executions

Proof: By strong induction on the length k of the derivation sequence.

We can assume that $k > 0$, since there doesn't exist a zero-length derivation sequence from a configuration into a final state.

The case ($k = 1$) is equivalent to the first rule of the structural semantics for sequential composition, which we can apply to conclude this case.

Let's consider the case where $k \geq 2$. Recall that our induction hypothesis lets us assume that the proposition holds for $n < k$, i.e. for all $n < k$ and for all statements p, q and states τ, τ':

$$\langle p, \tau \rangle \to_1^n \tau' \;\Rightarrow\; \langle p; q \;,\; \tau \rangle \to_1^n \langle q, \tau' \rangle \quad \text{(IH)}$$

Our assumption from the proposition tells us, that the derivation sequence

$$\langle s_1, \sigma \rangle \to_1^k \sigma' \quad \text{(A1)}$$

exists, and we want to prove that

$$\langle s_1; s_2 \;,\; \sigma \rangle \to_1^k \langle s_2, \sigma' \rangle$$

From (A1) and the fact that $k \geq 2$, we have that there is an intermediate configuration $\langle s_A, \sigma_A \rangle$, such that

$$\langle s_1, \sigma \rangle \to_1 \langle s_A, \sigma_A \rangle \to_1^{k-1} \sigma' \quad \text{(A2)}$$

We apply (IH) to the $k - 1$-long derivation sequence of (A2), instantiating: $\tau = \sigma_A$, $\tau' = \sigma'$, $p = s_A$ and $q = s_2$, and we get

$$\langle s_A; s_2 \;, \sigma_A \rangle \to_1^{k-1} \langle s_2, \sigma' \rangle$$

Thus, what remains to be proven is the first transition of our goal sequence:

$$\langle s_1; s_2, \sigma \rangle \to_1 \langle s_A; s_2 \;, \sigma_A \rangle$$

This is proven by the first transition of (A2) and the second rule of the structural semantics for sequential composition.

Concatinating the last two sequences yields our goal sequence and concludes the proof.

Alternative: Since we can assume that $k > 0$, it is also possible to start the proof by unrolling our initially assumed derivation sequence (A1) once, and then perform a case analysis on the intermediate configuration. This results in the two cases, namely non-final configuration and final state, which then imply that $k = 1$ and $k \geq 2$, respectively. The proof would still proceed by strong induction on k.

Assignment 4: Semantic equivalence of NS and SOS

NS $\Rightarrow$ SOS: $\forall \sigma, \sigma' \in \texttt{State}, s \in \texttt{Stm} \cdot \langle s, \sigma\rangle \to \sigma' \Rightarrow \langle s, \sigma\rangle \to_1^* \sigma'$

Proof: By induction on the shape of the derivation tree for the NS transition.

Case $\textsc{Ass}_{\text{NS}}$:

From $\textsc{Ass}_{\text{NS}}$ we get that the transition is of the form $\langle x \texttt{ := } e, \sigma\rangle \to \sigma[x \mapsto \mathcal{A}[\![e]\!]\sigma]$ for some variable x and some arithmetic expression e, and thus, that $s = (x \texttt{ := } e)$ and $\sigma' = \sigma[x \mapsto \mathcal{A}[\![e]\!]\sigma]$.

We apply $\textsc{Ass}_{\text{SOS}}$ to get $\langle x \texttt{ := } e, \sigma\rangle \to_1 \sigma[x \mapsto \mathcal{A}[\![e]\!]\sigma]$ as required.

Case $\textsc{Skip}_{\text{NS}}$:

Analogous to $\textsc{Ass}_{\text{NS}}$.

Case $\textsc{WhF}_{\text{NS}}$:

Then $s = \texttt{while } b \texttt{ do } s' \texttt{ end}$ for some boolean expression b and some statement s', $\sigma' = \sigma$ and $\mathcal{B}[\![b]\!] = \mathit{ff}$.

We conclude with the following derivation sequence:

$$\begin{array}{ll} & \langle \texttt{while } b \texttt{ do } s' \texttt{ end}, \sigma\rangle \\ \to_1 & \langle \texttt{if } b \texttt{ then } s'\texttt{;while } b \texttt{ do } s' \texttt{ end else skip end}, \sigma\rangle \\ \to_1 & \langle \texttt{skip}, \sigma\rangle \\ \to_1 & \sigma \end{array}$$

The second transition is justified by $\textsc{IfF}_{\text{SOS}}$, since $\mathcal{B}[\![b]\!]\sigma = \mathit{ff}$.

Case $\textsc{Seq}_{\text{NS}}$:

Then $s = (s_1; s_2)$ for some statements s_1 and s_2, and the derivation tree is of the form:

$$\frac{\dfrac{T_1}{\langle s_1, \sigma\rangle \to \sigma''} \quad \dfrac{T_2}{\langle s_2, \sigma''\rangle \to \sigma'}}{\langle s_1; s_2, \sigma\rangle \to \sigma'}$$

for some σ'', where T_1 is some derivation tree deriving $\langle s_1, \sigma\rangle \to \sigma''$ and where T_2 is some derivation tree deriving $\langle s_2, \sigma''\rangle \to \sigma'$. Note, that the trees T_1 and T_2 are meant to include the transition they derive.

We apply the IH to T_1 and T_2, and get $\langle s_1, \sigma\rangle \to_1^* \sigma''$ and $\langle s_2, \sigma''\rangle \to_1^* \sigma'$.

We conclude this case with the following derivation sequence:

$$\langle s_1; s_2, \sigma\rangle \to_1^* \langle s_2, \sigma''\rangle \to_1^* \sigma'$$

where the first sub-derivation sequence is obtained by the result of assignment 3 of sheet 11.

Case $\textsc{IfT}_{\text{NS}}$:

Then $s = (\texttt{if}\ \ b\ \ \texttt{then}\ \ s_1\ \ \texttt{else}\ \ s_2\ \ \texttt{end})$ for some boolean expression b and some statements s_1 and s_2, and the derivation tree is of the form:

$$\frac{\dfrac{T}{\langle s_1, \sigma\rangle \to \sigma'}}{\langle \texttt{if}\ \ b\ \ \texttt{then}\ \ s_1\ \ \texttt{else}\ \ s_2\ \ \texttt{end}, \sigma\rangle \to \sigma'}\ \mathcal{B}[\![b]\!]\sigma = tt$$

We apply the IH to T, and get $\langle s_1, \sigma\rangle \to_1^* \sigma'$.

We conclude this case with the following derivation sequence:

$$\langle \texttt{if}\ \ b\ \ \texttt{then}\ \ s_1\ \ \texttt{else}\ \ s_2\ \ \texttt{end}, \sigma\rangle \to_1 \langle s_1, \sigma\rangle \to_1^* \sigma'$$

The first transition is justified by $\textsc{IfT}_{\text{SOS}}$, since $\mathcal{B}[\![b]\!]\sigma = tt$.

Case $\textsc{IfF}_{\text{NS}}$:

Analogous to $\textsc{IfT}_{\text{NS}}$.

Case $\textsc{WhT}_{\text{NS}}$:

Then $s = (\texttt{while}\ \ b\ \ \texttt{do}\ \ s'\ \ \texttt{end})$ for some boolean expression b and some statement s', and the derivation tree is of the following form:

$$\frac{\dfrac{T_1}{\langle s', \sigma\rangle \to \sigma''}\quad \dfrac{T_2}{\langle \texttt{while}\ \ b\ \ \texttt{do}\ \ s'\ \ \texttt{end}, \sigma''\rangle \to \sigma'}}{\langle \texttt{while}\ \ b\ \ \texttt{do}\ \ s'\ \ \texttt{end}, \sigma\rangle \to \sigma'}\ \mathcal{B}[\![b]\!]\sigma = tt$$

for some state σ'', where T_1 is some derivation tree deriving $\langle s', \sigma\rangle \to \sigma''$ and where T_2 is some derivation tree deriving $\langle \texttt{while}\ \ b\ \ \texttt{do}\ \ s'\ \ \texttt{end}, \sigma''\rangle \to \sigma'$.

We apply the IH to T_1 and T_2, and get $\langle s', \sigma\rangle \to_1^* \sigma''$ and $\langle \texttt{while}\ \ b\ \ \texttt{do}\ \ s'\ \ \texttt{end}, \sigma''\rangle \to_1^* \sigma'$.

We conclude this case with the following derivation sequence:

$$\begin{array}{ll} & \langle \texttt{while}\ \ b\ \ \texttt{do}\ \ s'\ \ \texttt{end}, \sigma\rangle \\ \to_1 & \langle \texttt{if}\ \ b\ \ \texttt{then}\ \ (s'; \texttt{while}\ \ b\ \ \texttt{do}\ \ s'\ \ \texttt{end})\ \ \texttt{else}\ \ \texttt{skip}\ \ \texttt{end}, \sigma\rangle \\ \to_1 & \langle (s'; \texttt{while}\ \ b\ \ \texttt{do}\ \ s'\ \ \texttt{end}), \sigma\rangle \\ \to_1^* & \langle \texttt{while}\ \ b\ \ \texttt{do}\ \ s'\ \ \texttt{end}, \sigma''\rangle \\ \to_1^* & \sigma' \end{array}$$

where the second transition is justified by $\textsc{IfT}_{\text{SOS}}$, since $\mathcal{B}[\![b]\!]\sigma = tt$, and where the second to last transition is obtained from the result of assignment 3 of sheet 11.

SOS $\Rightarrow$ NS: $\forall \sigma, \sigma' \in \texttt{State}, s \in \texttt{Stm}, k \in \mathbb{N} \cdot \langle s, \sigma \rangle \rightarrow_1^k \sigma' \Rightarrow \langle s, \sigma \rangle \rightarrow \sigma'$

Proof: By strong induction on the length k of the SOS derivation sequence.

Assume that the proposition holds $\forall m < k$, show it for k. We can assume $k > 0$, since there is no zero-length derivation sequence from a non-final configuration to a final state, i.e., $\langle s, \sigma \rangle \rightarrow_1^0 \sigma'$ doesn't exist.

We unroll the derivation sequence once to $\langle s, \sigma \rangle \rightarrow_1 \gamma \rightarrow_1^{k-1} \sigma'$ and inspect the derivation tree of the first transition, considering cases for the last rule applied:

Case $\textsc{Ass}_{\textsc{sos}}$:

Then $s = (x \texttt{:=} e)$ for some variable x and some arithmetic expression e,
and $\gamma' = \sigma[x \mapsto \mathcal{A}[\![e]\!]\sigma]$. Since γ is a final state there is no further derivation sequence ($k = 1$), and hence $\sigma' = \gamma = \sigma[x \mapsto \mathcal{A}[\![e]\!]\sigma]$.

From $\textsc{Ass}_{\textsc{ns}}$ we get $\langle x \ \texttt{:=} \ e, \sigma \rangle \rightarrow \sigma'$, as required.

Case $\textsc{Skip}_{\textsc{sos}}$:

Similar to $\textsc{Ass}_{\textsc{sos}}$, we apply the corresponding NS rule and are done.

Case $\textsc{Seq1}_{\textsc{sos}}$, $\textsc{Seq2}_{\textsc{sos}}$:

Then $s = (s_1; s_2)$ for some statements s_1 and s_2, and the entire derivation sequence is of the form $\langle s_1; s_2, \sigma \rangle \rightarrow_1^k \sigma'$.

From the lemma on p. 132 of the lecture slides, we get $\langle s_1, \sigma \rangle \rightarrow_1^{k_1} \sigma''$ and $\langle s_2, \sigma'' \rangle \rightarrow_1^{k_2} \sigma'$, for some state σ'' and some natural numbers k_1 and k_2, such that $k_1 + k_2 = k$. Since both sub-derivations sequences are from non-final configurations to final states, we have $k_1, k_2 \geq 1$, and therefore $k_1, k_2 < k$.

Applying the IH to both sub-derivations yields $\langle s_1, \sigma \rangle \rightarrow \sigma''$ and $\langle s_2, \sigma'' \rangle \rightarrow \sigma'$, which we combine by applying $\textsc{Seq}_{\textsc{ns}}$ to get $\langle s_1; s_2, \sigma \rangle \rightarrow \sigma'$, as required.

Case $\textsc{IfT}_{\textsc{sos}}$:

Then $s = (\texttt{if} \ b \ \texttt{then} \ s_1 \ \texttt{else} \ s_2 \ \texttt{end})$, for some boolean expression b and some statements s_1 and s_2, $\mathcal{B}[\![b]\!]\sigma = tt$, and the unrolled derivation sequence is of the form:

$$\langle \texttt{if} \ b \ \texttt{then} \ s_1 \ \texttt{else} \ s_2 \ \texttt{end}, \sigma \rangle \rightarrow_1 \langle s_1, \sigma \rangle \rightarrow_1^{k-1} \sigma'$$

We apply the IH to the tail sequence, and get $\langle s_1, \sigma \rangle \rightarrow \sigma'$, which enables us to conclude this case by applying $\textsc{IfT}_{\textsc{ns}}$:

$$\frac{\langle s_1, \sigma \rangle \rightarrow \sigma'}{\langle \texttt{if} \ b \ \texttt{then} \ s_1 \ \texttt{else} \ s_2 \ \texttt{end}, \sigma \rangle \rightarrow \sigma'} \ \mathcal{B}[\![b]\!]\sigma = tt$$

Case $\textsc{If}\text{F}_{\text{SOS}}$:

Analogous to $\textsc{If}\text{T}_{\text{SOS}}$.

Case $\textsc{Wh}_{\text{SOS}}$:

Then $s = (\texttt{while}\ b\ \texttt{do}\ s'\ \texttt{end})$ for some boolean expression b and some statement s', $\gamma = \langle \texttt{if}\ b\ \texttt{then}\ s'; \texttt{while}\ b\ \texttt{do}\ s'\ \texttt{end}\ \texttt{else}\ \texttt{skip}\ \texttt{end}, \sigma\rangle$, and the unrolled derivation sequence is of the form:

$$\begin{array}{ll} & \langle \texttt{while}\ b\ \texttt{do}\ s'\ \texttt{end}, \sigma\rangle \\ \rightarrow_1 & \langle \texttt{if}\ b\ \texttt{then}\ s'; \texttt{while}\ b\ \texttt{do}\ s'\ \texttt{end}\ \texttt{else}\ \texttt{skip}\ \texttt{end}, \sigma\rangle \\ \rightarrow_1^{k-1} & \sigma' \end{array}$$

We apply the IH to the tail sequence, and get

$$\langle \texttt{if}\ b\ \texttt{then}\ s'; \texttt{while}\ b\ \texttt{do}\ s'\ \texttt{end}\ \texttt{else}\ \texttt{skip}\ \texttt{end}, \sigma\rangle \rightarrow \sigma'.$$

From the semantic equivalence proved on p. 87 we get $\langle \texttt{while}\ b\ \texttt{do}\ s'\ \texttt{end}, \sigma\rangle \rightarrow \sigma'$, which concludes this case.

Note: We can also "manually" conclude this case, i.e. not use the semantic equivalence. This requires a case split on which branch of the if-statement is taken, and some decomposing and recomposing of the resulting derivation tree.

Assignment 5 - Headache of the Week: Transactions

In order to maintain a backup of the state that existed before statement s in $\texttt{revert}\ s\ \texttt{if}\ b\ \texttt{end}$ had been executed, we extend our state to be a list of regular states:

$$State' = [State]$$

The extended state is used as a stack of backups. The top-most state is the current state, i.e. the one that is used to evaluate expressions and that is changed by assigments, and the other states are used to rollback actions, if necessary. We decompose a state by Haskell-like pattern matching, $\sigma' = \sigma : \sigma_s$ means that the extended state σ' consists of a head state σ and tail states σ_s.

The first new rule pushes the current state as a backup onto the state stack, and inserts a marker statement that is used to conditionally trigger a rollback.

$$\textsc{Rev}_{SOS}\ \frac{}{\langle \texttt{revert}\ s\ \texttt{if}\ b, \sigma : \sigma_s\rangle \rightarrow_1 \langle s; \texttt{rollback-if}\ b, \sigma : \sigma : \sigma_s\rangle}$$

The semantics of the marker statement are captured by the next two rules. The first one performs a rollback by discarding the head state, whereas the second one keeps the head state and instead

removes the backup state. Note that `rollback-if` is assumed to not occur in the source programs, i.e., it is only inserted into the program by $\textsc{Rev}_{SOS}$.

$$\textsc{RbiT}_{SOS}\ \frac{}{\langle \texttt{rollback-if}\ \ b, \sigma : \sigma_s\rangle \to_1 \sigma_s}\ \text{if } \mathcal{B}[\![b]\!]\sigma = tt$$

$$\textsc{RbiF}_{SOS}\ \frac{}{\langle \texttt{rollback-if}\ \ b, \sigma : \sigma' : \sigma_s\rangle \to_1 \sigma : \sigma_s}\ \text{if } \mathcal{B}[\![b]\!]\sigma = ff$$

All other rules have to be adapted to the extended state. For some rules it is sufficient to simply propagate the extended state, e.g. for $\textsc{Skip}_{SOS}$ or $\textsc{Seq}_{SOS}$. For others, such as $\textsc{Ass}_{SOS}$, we have to decompose the extended state in order to preserve the original semantics.

$$\textsc{Skip'}_{SOS}\ \frac{}{\langle \texttt{skip}, \sigma_s\rangle \to_1 \sigma_s}$$

$$\textsc{Ass'}_{SOS}\ \frac{}{\langle x\texttt{:=}e, \sigma : \sigma_s\rangle \to_1 \sigma[x \mapsto \mathcal{A}[\![e]\!]\sigma] : \sigma_s}$$

$$\textsc{Seq1'}_{SOS}\ \frac{\langle s_1, \sigma_s\rangle \to_1 \sigma_s'}{\langle s_1\,\texttt{;}\,s_2, \sigma_s\rangle \to_1 \langle s_2, \sigma_s'\rangle}$$

$$\textsc{Seq2'}_{SOS}\ \frac{\langle s_1, \sigma_s\rangle \to_1 \langle s_1', \sigma_s'\rangle}{\langle s_1\,\texttt{;}\,s_2, \sigma_s\rangle \to_1 \langle s_1'\,\texttt{;}\,s_2, \sigma_s'\rangle}$$

$$\textsc{IfT'}_{SOS}\ \frac{}{\langle \texttt{if}\ b\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{end}, \sigma : \sigma_s\rangle \to_1 \langle s_1, \sigma : \sigma_s\rangle}\ \text{if } \mathcal{B}[\![b]\!]\sigma = tt$$

$$\textsc{IfF'}_{SOS}\ \frac{}{\langle \texttt{if}\ b\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{end}, \sigma : \sigma_s\rangle \to_1 \langle s_2, \sigma : \sigma_s\rangle}\ \text{if } \mathcal{B}[\![b]\!]\sigma = ff$$

$$\textsc{While'}_{SOS}\ \frac{}{\langle \texttt{while}\ b\ \texttt{do}\ s\ \texttt{end}, \sigma_s\rangle \to_1 \langle \texttt{if}\ b\ \texttt{then}\ s\texttt{;while}\ b\ \texttt{do}\ s\ \texttt{end else skip end}, \sigma_s\rangle}$$