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 10: Big Step Semantics

Assignment 1

We use the following abbreviations: l is the statement `a := a+n; b := b*n; n := n-1` and w the statement `while n#0 do` l `end`. Moreover, we introduce the following abbreviation:

$$[v_1, .., v_k \mapsto n_1, .., n_k]$$

stands for the state

$$\sigma[v_1 \mapsto n_1]..[v_k \mapsto n_k]$$

where σ is the initial state mentioned in the exercise.

We construct the derivation tree shown in the following page.

$$\dfrac{\dfrac{}{\langle \texttt{a := a+n}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 0,1,2]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,1,2]}\ \textsc{Ass}_{NS} \quad \dfrac{\dfrac{}{\langle \texttt{b := b*n}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,1,2]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,2,2]}\ \textsc{Ass}_{NS} \quad \dfrac{}{\langle \texttt{n := n-1}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,2,2]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,2,1]}\ \textsc{Ass}_{NS}}{\langle \texttt{b := b*n; n := n-1}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,1,2]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,2,1]}\ \textsc{Seq}_{NS}}{\langle \texttt{(a := a+n; b := b*n); n := n-1}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 0,1,2]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,2,1]}\ \textsc{Seq}_{NS} \quad \dfrac{t_1}{\langle \texttt{while n\#0 do l end}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 0,1,2]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,0]}\ \textsc{WhT}_{NS}$$

where t_1 is the following derivation tree:

$$\dfrac{\dfrac{}{\langle \texttt{a := a+n}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,2,1]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,1]}\ \textsc{Ass}_{NS} \quad \dfrac{\dfrac{}{\langle \texttt{b := b*n}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,1]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,1]}\ \textsc{Ass}_{NS} \quad \dfrac{}{\langle \texttt{n := n-1}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,1]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,0]}\ \textsc{Ass}_{NS}}{\langle \texttt{b := b*n; n := n-1}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,1]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,0]}\ \textsc{Seq}_{NS}}{\langle \texttt{(a := a+n; b := b*n); n := n-1}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,2,1]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,0]}\ \textsc{Seq}_{NS} \quad \dfrac{}{\langle w, [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,0]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,0]}\ \textsc{WhF}_{NS}}{\langle \texttt{while n\#0 do l end}, [\texttt{a},\texttt{b},\texttt{n} \mapsto 2,2,1]\rangle \to [\texttt{a},\texttt{b},\texttt{n} \mapsto 3,2,0]}\ \textsc{WhT}_{NS}$$

Assignment 2

For the direction from right to left, we consider the derivation tree for

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

The last applied rule in this derivation tree is a rule for the if-then-else statement. So, the derivation tree has either the form

$$\frac{\dfrac{\vdots}{\langle s; \texttt{while } b \texttt{ do } s \texttt{ end}, \sigma\rangle \to \sigma''}}{\langle \texttt{if } b \texttt{ then } s; \texttt{while } b \texttt{ do } s \texttt{ end else skip end}, \sigma\rangle \to \sigma''}\ \textsc{IfT}_{NS} \tag{1}$$

or

$$\frac{\langle \texttt{skip}, \sigma\rangle \to \sigma''}{\langle \texttt{if } b \texttt{ then } s; \texttt{while } b \texttt{ do } s \texttt{ end else skip end}, \sigma\rangle \to \sigma''}\ \textsc{IfF}_{NS} \tag{2}$$

- Let us first consider the case (2). The rule is only applicable when $\mathcal{B}[\![b]\!]\sigma = ff$. Furthermore, with the rule for `skip`, we conclude that $\sigma = \sigma''$. We construct the following derivation tree:
$$\frac{}{\langle \texttt{while } b \texttt{ do } s \texttt{ end}, \sigma\rangle \to \sigma}\ \textsc{WhF}_{NS}$$
- Let us now consider the case (1). The rule is only applicable when $\mathcal{B}[\![b]\!]\sigma = tt$. The next applied rule in the derivation tree must be for sequential composition. The last part of the derivation tree has the form
$$\frac{\dfrac{\dfrac{\vdots}{\langle s, \sigma\rangle \to \sigma'} \quad \dfrac{\vdots}{\langle \texttt{while } b \texttt{ do } s \texttt{ end}, \sigma'\rangle \to \sigma''}}{\langle s; \texttt{while } b \texttt{ do } s \texttt{ end}, \sigma\rangle \to \sigma''}\ \textsc{Seq}_{NS}}{\langle \texttt{if } b \texttt{ then } s; \texttt{while } b \texttt{ do } s \texttt{ end else skip end}, \sigma\rangle \to \sigma''}\ \textsc{IfT}_{NS}$$
Let T_1 be the derivation tree above $\langle s, \sigma\rangle \to \sigma'$ and let T_2 be the derivation tree above $\langle \texttt{while } b \texttt{ do } s \texttt{ end}, \sigma'\rangle \to \sigma''$. We construct the following derivation tree:
$$\frac{\dfrac{T_1}{\langle s, \sigma\rangle \to \sigma'} \quad \dfrac{T_2}{\langle \texttt{while } b \texttt{ do } s \texttt{ end}, \sigma'\rangle \to \sigma''}}{\langle \texttt{while } b \texttt{ do } s \texttt{ end}, \sigma\rangle \to \sigma''}\ \textsc{WhT}_{NS}$$

Assignment 3

Part (a)

The semantics (ii) can be defined by the following rules:

$$\frac{\langle x := e_1, \sigma\rangle \to \sigma'}{\langle \texttt{for } x := e_1 \texttt{ to } e_2 \texttt{ do } s \texttt{ end}, \sigma\rangle \to \sigma'}\ \mathcal{B}[\![x = e_2]\!]\sigma' = tt \quad (\textsc{ForT})$$

$$\frac{\langle x := e_1; \sigma\rangle \to \sigma'' \quad \langle s; \texttt{for } x := x+1 \texttt{ to } e_2 \texttt{ do } s \texttt{ end}, \sigma''\rangle \to \sigma'}{\langle \texttt{for } x := e_1 \texttt{ to } e_2 \texttt{ do } s \texttt{ end}, \sigma\rangle \to \sigma'} \; \mathcal{B}[\![x = e_2]\!]\sigma'' = \mathit{ff} \qquad (\textsc{ForF})$$

To get semantics (i), we can add an extra condition to the above rules that e_2 be a numeral, and an extra rule that performs the evaluation of e_2, if is it is not a numeral:

$$\frac{\langle \texttt{for } x := e_1 \texttt{ to } n \texttt{ do } s \texttt{ end}, \sigma\rangle \to \sigma'}{\langle \texttt{for } x := e_1 \texttt{ to } e_2 \texttt{ do } s \texttt{ end}, \sigma\rangle \to \sigma'} \; (*)$$

where the side condition (*) is:

$$e_2 \text{ is not a numeral and } \mathcal{A}[\![e_2]\!]\sigma = \mathcal{N}[\![n]\!]$$

Notice that both cases allow the body of a `for` to assign to the index variable x. If this is not desired, we can forbid it in various ways, for example we can have extra side conditions on the body s of the loop. Notice also that in our semantics (i), if x occurs in e_2 we will use the old value of x when evaluating e_2 to a numeral, while in our semantics (ii) we would first assign to x and then start evaluating the upper bound. It isn't totally clear which is preferable, in either case (although they do give different semantics to the construct).

Part (b)

Consider now semantics (ii). We will prove the proposition of the question, by induction on the derivation tree of

$$\langle \texttt{for } x := e_1 \texttt{ to } e_2 \texttt{ do } s \texttt{ end}, \sigma\rangle \to \sigma' \quad \text{(AS0)}$$

We want to prove that

$$\langle x := e_1;\ \texttt{while } x\#e_2 \texttt{ do } s; x := x+1 \texttt{ end }, \sigma\rangle \to \sigma' \quad \text{(PO)}$$

Let T be a derivation tree (AS0).

We know that for all subtrees $\hat{T}'$ of T, variables $\hat{x}$, expressions $\hat{e}_1, \hat{e}_2$, statements $\hat{s}$, states $\hat{\sigma}, \hat{\sigma}'$, such that $\hat{T}'$ proves

$$\langle \texttt{for } \hat{x} := \hat{e}_1 \texttt{ to } \hat{e}_2 \texttt{ do } \hat{s} \texttt{ end}, \hat{\sigma}\rangle \to \hat{\sigma}'$$

we have that

$$\langle \hat{x} := \hat{e}_1;\ \texttt{while } \hat{x}\#\hat{e}_2 \texttt{ do } \hat{s}; \hat{x} := \hat{x}+1 \texttt{ end }, \hat{\sigma}\rangle \to \hat{\sigma}'$$

We call this the induction hypothesis (IH).

The last rule applied in T is either $\textsc{ForT}$ or $\textsc{ForF}$. We split into two cases:

Case $\textsc{ForT}$: If the last rule applied was $\textsc{ForT}$, then we know that $\langle x := e_1, \sigma\rangle \to \sigma'$ and that $\mathcal{B}[\![x = e_2]\!]\sigma' = \mathit{tt}$, which implies that $\mathcal{B}[\![x\#e_2]\!]\sigma' = \mathit{ff}$. We can construct the derivation for (PO) as follows:

$$\frac{\langle x := e_1, \sigma\rangle \to \sigma' \quad \dfrac{}{\langle \texttt{while } x\#e_2 \texttt{ do } s; x := x+1 \texttt{ end }, \sigma'\rangle \to \sigma'}\ \textsc{WhF}_{NS}}{\langle x := e_1;\ \texttt{while } x\#e_2 \texttt{ do } s; x := x+1 \texttt{ end }, \sigma\rangle \to \sigma'}\ \textsc{Seq}_{NS}$$

Case FORF: If the last rule applied was FORF, then we know that there is a σ'' such that

$$\langle x := e_1, \sigma \rangle \to \sigma'' \quad \text{(AS1)}$$

and

$$\langle s; \texttt{for } x := x + 1 \texttt{ to } e_2 \texttt{ do } s \texttt{ end}, \sigma'' \rangle \to \sigma' \quad \text{(AS2)}$$

and that $\mathcal{B}[\![x = e_2]\!]\sigma'' = ff$, which implies that $\mathcal{B}[\![x \# e_2]\!]\sigma'' = tt$ (AS3).

There is a derivation tree for (AS2). The root of that tree must have been derived by the SEQ_{NS} rule, which means that there is a σ''' such that

$$\langle s, \sigma'' \rangle \to \sigma''' \quad \text{(AS4)}$$

and

$$\langle \texttt{for } x := x + 1 \texttt{ to } e_2 \texttt{ do } s \texttt{ end}, \sigma''' \rangle \to \sigma' \quad \text{(AS5)}$$

Let T' be the subtree of T which derives (AS5).

We apply (IH) for: $\hat{x} = x$, $\hat{e}_1 = (x + 1)$, $\hat{e}_2 = e_2$, $\hat{s} = s$, $\hat{\sigma} = \sigma'''$, $\hat{\sigma}' = \sigma'$, and $\hat{T}' = T'$. We get:

$$\langle x := x + 1;\ \texttt{while } x \# e_2 \texttt{ do } s; x := x + 1 \texttt{ end }, \sigma''' \rangle \to \sigma' \quad \text{(AS6)}$$

The last rule applied to the derivation tree for (AS6) must have been SEQ_{NS}. Therefore, there is a state σ_1 such that

$$\langle x := x + 1, \sigma''' \rangle \to \sigma_1 \quad \text{(AS7)}$$

and

$$\langle \texttt{while } x \# e_2 \texttt{ do } s; x := x + 1 \texttt{ end }, \sigma_1 \rangle \to \sigma' \quad \text{(AS8)}$$

We can get a derivation tree for (PO), as follows:

$$\dfrac{\dfrac{\vdots}{\text{(AS1)}} \qquad \dfrac{\dfrac{\dfrac{\vdots}{\text{(AS4)}} \quad \dfrac{\vdots}{\text{(AS7)}}}{\langle s; x := x + 1, \sigma'' \rangle \to \sigma_1}\ \text{SEQ}_{NS} \quad \dfrac{\vdots}{\text{(AS8)}}}{\langle \texttt{while } x \# e_2 \texttt{ do } s; x := x + 1 \texttt{ end }, \sigma'' \rangle \to \sigma'}\ \text{WHT}^*_{NS}}{\text{(PO)}}\ \text{SEQ}_{NS}$$

(*) this rule applies because of (AS3).

Assignment 4

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

Assignment 5 - Headache of the week

To support `break`, an idea is to add a flag to the state that makes it explicit if a `break` statement has been executed. We define a new set of states $State'$ that contains this information:

$$State' = \{tt, ff\} \times State$$

Let $\tau, \tau', \tau_i..$ range over elements of set $State'$ and $\sigma, \sigma', \sigma_i..$ over elements of set $State$.

The behavior of the `break` statement is to set this flag to true:

$$\frac{}{\langle \texttt{break}, (v, \sigma)\rangle \to (tt, \sigma)}$$

When the flag is true, this means that a break statement has been activated. No change to the state must happen, until the flag is reset to false. This changes the rules for the assignment as follows:

$$\frac{}{\langle x := e, (\mathit{ff}, \sigma)\rangle \to (\mathit{ff}, \sigma[x \mapsto \mathcal{A}[\![e]\!]\sigma])}$$

$$\frac{}{\langle x := e, (tt, \sigma)\rangle \to (tt, \sigma)}$$

Skipping should not change the state, regardless of the flag:

$$\frac{}{\langle \texttt{skip}, \tau\rangle \to \tau}$$

The sequential composition should behave as before, regardless of the flag. In case the flag is true, this propagates to the end of the sequential composition:

$$\frac{\langle s_1, \tau\rangle \to \tau'' \quad \langle s_2, \tau''\rangle \to \tau'}{\langle s_1; s_2, \tau\rangle \to \tau'}$$

A conditional can be ignored if the flag is initially true:

$$\frac{}{\langle \texttt{if } b \texttt{ then } s_1 \texttt{ else } s_2, (tt, \sigma)\rangle \to (tt, \sigma)}$$

If the flag is initially false, it behaves as before:

$$\frac{\langle s_1, (\mathit{ff}, \sigma)\rangle \to \tau}{\langle \texttt{if } b \texttt{ then } s_1 \texttt{ else } s_2, (\mathit{ff}, \sigma)\rangle \to \tau}\ \mathcal{B}[\![b]\!]\sigma = tt$$

$$\frac{\langle s_2, (\mathit{ff}, \sigma)\rangle \to \tau}{\langle \texttt{if } b \texttt{ then } s_1 \texttt{ else } s_2, (\mathit{ff}, \sigma)\rangle \to \tau}\ \mathcal{B}[\![b]\!]\sigma = \mathit{ff}$$

A loop is also ignored if the flag is initially true:

$$\frac{}{\langle \texttt{while } b \texttt{ do } s \texttt{ end}, (v, \sigma)\rangle \to (v, \sigma)}\ v = tt \text{ or } \mathcal{B}[\![b]\!]\sigma = \mathit{ff}$$

If the flag is initially false and the loop condition is initially true, then the loop should execute. However, its behavior depends on whether the loop body raises the flag or not. If the body does not raise the flag, then the loop execution continues after the first iteration

$$\frac{\langle s, (\mathit{ff}, \sigma)\rangle \to (\mathit{ff}, \sigma'') \quad \langle \texttt{while } b \texttt{ do } s \texttt{ end}, (\mathit{ff}, \sigma'')\rangle \to (\mathit{ff}, \sigma')}{\langle \texttt{while } b \texttt{ do } s \texttt{ end}, (\mathit{ff}, \sigma)\rangle \to (\mathit{ff}, \sigma')}\ \mathcal{B}[\![b]\!]\sigma = tt$$

If the body raises the flag, then there are no more iterations: the loop execution is broken. Notice that the flag is reset to false:

$$\frac{\langle s, (\mathit{ff}, \sigma)\rangle \to (tt, \sigma')}{\langle \texttt{while } b \texttt{ do } s \texttt{ end}, (\mathit{ff}, \sigma)\rangle \to (\mathit{ff}, \sigma')}\ \mathcal{B}[\![b]\!]\sigma = tt$$