Semantic functions

For the abstract machine:

$\mathcal{M}$: Code $\rightarrow$ (State $\hookrightarrow$ State)

$$\mathcal{M}[c]\ s = \begin{cases} s' & \text{if } (c,\ \varepsilon,\ s)\ \rhd^*\ (\varepsilon,\ e,\ s') \\ \text{undef} & \text{otherwise} \end{cases}$$

For the statements:

$\mathcal{S}_{am}$: Stm $\rightarrow$ (State $\hookrightarrow$ State)

$$\mathcal{S}_{am}[s] = (\mathcal{M} \circ \mathcal{CS})[s]$$

Theorem 3.20

For every statement S of While we have $\mathcal{S}_{ns}[S] = \mathcal{S}_{am}\,[S]$.

Natural Semantics

$$(x := a, s) \rightarrow s[x \mapsto \mathcal{A}[a]s]$$

$$(\texttt{skip}, s) \rightarrow s$$

$$\frac{(S_1, s) \rightarrow s', \quad (S_2, s') \rightarrow s''}{(S_1; S_2, s) \rightarrow s''}$$

$$\frac{(S_1, s) \rightarrow s'}{(\texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2, s) \rightarrow s'}$$

if $\mathcal{B}[b]s = \mathsf{tt}$

$$\frac{(S_2, s) \rightarrow s'}{(\texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2, s) \rightarrow s'}$$

if $\mathcal{B}[b]s = \mathsf{ff}$

$$\frac{(S, s) \rightarrow s', \quad (\texttt{while } b \texttt{ do } S, s') \rightarrow s''}{(\texttt{while } b \texttt{ do } S, s) \rightarrow s''}$$

if $\mathcal{B}[b]s = \mathsf{tt}$

$$(\texttt{while } b \texttt{ do } S, s) \rightarrow s$$

if $\mathcal{B}[b]s = \mathsf{ff}$

$$(c,\ e,\ s)\ \triangleright\ (c',\ e',\ s')$$

$$\big(\text{PUSH-}n{:}c,\ e,\ s\big)\ \triangleright\ (c,\ \mathcal{N}[n]{:}e,\ s)$$

$$\big(\text{ADD}{:}c,\ z_1{:}z_2{:}e,\ s\big)\ \triangleright\ (c,\ (z_1+z_2){:}e,\ s)$$
$$\text{if } z_1,\ z_2\ \in \mathsf{Z}$$

$$\big(\text{MULT}{:}c,\ z_1{:}z_2{:}e,\ s\big) \triangleright\ (c,\ (z_1 \star z_2){:}e,\ s)$$
$$\text{if } z_1,\ z_2\ \in \mathsf{Z}$$

$$\big(\text{SUB}{:}c,\ z_1{:}z_2{:}e,\ s\big)\ \triangleright\ (c,\ (z_1-z_2){:}e,\ s)$$
$$\text{if } z_1,\ z_2\ \in \mathsf{Z}$$

$$\big(\text{TRUE}{:}c,\ e,\ s\big)\ \triangleright\ (c,\ \mathsf{tt}{:}e,\ s)$$

$$\big(\text{FALSE}{:}c,\ e,\ s\big)\ \triangleright\ (c,\ \mathsf{ff}{:}e,\ s)$$

$$\big(\text{EQ}{:}c,\ z_1{:}z_2{:}e,\ s\big)\ \triangleright\ (c,\ (z_1{=}z_2){:}e,\ s)$$
$$\text{if } z_1,\ z_2\ \in \mathsf{Z}$$

$$\big(\text{LE}{:}c,\ z_1{:}z_2{:}e,\ s\big)\ \triangleright\ (c,\ (z_1 \leq z_2){:}e,\ s)$$
$$\text{if } z_1,\ z_2\ \in \mathsf{Z}$$

$$\big(\text{AND}{:}c,\ t_1{:}t_2{:}e,\ s\big)\ \triangleright\ (c,\ (t_1\ \wedge\ t_2){:}e,\ s)$$
$$\text{if } t_1,\ t_2\ \in \mathsf{T}$$

$$\big(\text{NEG}{:}c,\ t{:}e,\ s\big)\ \triangleright\ (c,\ (\neg t){:}e,\ s)$$
$$\text{if } t\ \in \mathsf{T}$$

$$\boxed{(c,\ e,\ s)\ \rhd\ (c',\ e',\ s')}$$

$$\big(\textsc{fetch}\text{-}x{:}c,\ e,\ s\big)\ \rhd\ \big(c,\ (s\ x){:}e,\ s\big)$$

$$\big(\textsc{store}\text{-}x{:}c,\ z{:}e,\ s\big)\ \rhd\ \big(c,\ e,\ s[x \mapsto z]\big)$$
$$\text{if } z\ \in \mathsf{Z}$$

$$\big(\textsc{noop}{:}c,\ e,\ s\big)\ \rhd\ \big(c,\ e,\ s\big)$$

$$\big(\textsc{branch}(c_1,\ c_2){:}c,\ t{:}e,\ s\big)$$
$$\rhd \begin{cases} (c_1 : c, e, s) & \text{if } t{=}\mathsf{tt} \\ (c_2 : c, e, s) & \text{if } t{=}\mathsf{ff} \end{cases}$$

$$\big(\textsc{loop}(c_1,\ c_2){:}c,\ e,\ s\big)$$
$$\rhd\ \big(c_1{:}\textsc{branch}(c_2{:}\textsc{loop}(c_1,\ c_2),$$
$$\textsc{noop}){:}c,\ e,\ s\big)$$

Lemma 3.18

For all arithmetic expressions a we have

$$(\mathcal{CA}[a],\, \varepsilon,\, s) \;\rhd^*\; (\varepsilon,\, \mathcal{A}[a]s,\, s)$$

Furthermore, all intermediate configurations of this computation sequence will have a non-empty evaluation stack.

Exercise 3.19

For all boolean expressions b we have

$$(\mathcal{CB}[b],\, \varepsilon,\, s) \;\rhd^*\; (\varepsilon,\, \mathcal{B}[b]s,\, s)$$

Furthermore, all intermediate configurations of this computation sequence will have a non-empty evaluation stack.

Exercise 3.4

If $(c_1, e_1, s) \rhd^k (c', e', s')$

then $(c_1 : c_2, e_1 : e_2, s) \rhd^k (c' : c_2, e' : e_2, s')$

Exercise 3.5

If $(c_1 : c_2, e, s) \rhd^k (\varepsilon, e'', s'')$

then there exists e', s', k_1 and k_2 such that

$(c_1, e, s) \rhd^{k_1} (\varepsilon, e', s')$ and

$(c_2, e', s') \rhd^{k_2} (\varepsilon, e'', s'')$ and

$k = k_1 + k_2$

Exercise 3.6

The abstract machine is deterministic: if

$(c, e, s) \rhd^* (\varepsilon, e', s')$ and

$(c, e, s) \rhd^* (\varepsilon, e'', s'')$

then $e' = e''$ and $s' = s''$.

$\mathcal{A}$: Aexp $\rightarrow$ State $\rightarrow$ N

$$
\begin{aligned}
\mathcal{A}[n]s &= \mathcal{N}[n] \\
\mathcal{A}[x]s &= s\ x \\
\mathcal{A}[a_1 + a_2]s &= \mathcal{A}[a_1]s + \mathcal{A}[a_2]s \\
\mathcal{A}[a_1\ *\ a_2]s &= \mathcal{A}[a_1]s\ *\ \mathcal{A}[a_2]s \\
\mathcal{A}[a_1 - a_2]s &= \mathcal{A}[a_1]s - \mathcal{A}[a_2]s
\end{aligned}
$$

Code generation for expressions

$$
\begin{aligned}
\mathcal{CA}[n] &= \text{PUSH-}n \\
\mathcal{CA}[x] &= \text{FETCH-}x \\
\mathcal{CA}[a_1{+}a_2] &= \mathcal{CA}[a_2]{:}\mathcal{CA}[a_1]{:}\text{ADD} \\
\mathcal{CA}[a_1 \star a_2] &= \mathcal{CA}[a_2]{:}\mathcal{CA}[a_1]{:}\text{MULT} \\
\mathcal{CA}[a_1{-}a_2] &= \mathcal{CA}[a_2]{:}\mathcal{CA}[a_1]{:}\text{SUB} \\
\mathcal{CB}[\texttt{true}] &= \text{TRUE} \\
\mathcal{CB}[\texttt{false}] &= \text{FALSE} \\
\mathcal{CB}[a_1 = a_2] &= \mathcal{CA}[a_2]{:}\mathcal{CA}[a_1]{:}\text{EQ} \\
\mathcal{CB}[a_1 \leq a_2] &= \mathcal{CA}[a_2]{:}\mathcal{CA}[a_1]{:}\text{LE} \\
\mathcal{CB}[\neg\, b] &= \mathcal{CB}[b]{:}\text{NEG} \\
\mathcal{CB}[b_1 \wedge b_2] &= \mathcal{CB}[b_2]{:}\mathcal{CB}[b_1]{:}\text{AND}
\end{aligned}
$$

Lemma 3.21

For every statement S of While and states s and s', we have that

if $(S,\ s) \to s'$

then $(\mathcal{CS}[S],\ \varepsilon,\ s)\ \rhd^*\ (\varepsilon,\ \varepsilon,\ s')$

Lemma 3.22

For every statement S of While and states s and s', we have that

if $(\mathcal{CS}[S],\ \varepsilon,\ s)\ \rhd^k\ (\varepsilon,\ e,\ s')$

then $(S,\ s) \to s'$ and $e = \varepsilon$

Code generation for statements

$\mathcal{CS}[x := a]$
$\quad = \mathcal{CA}[a]$:STORE-x

$\mathcal{CS}[\texttt{skip}]$
$\quad =$ NOOP

$\mathcal{CS}[S_1;S_2]$
$\quad = \mathcal{CS}[S_1]:\mathcal{CS}[S_2]$

$\mathcal{CS}[\texttt{if } b \texttt{ then } S_1 \texttt{ else } S_2]$
$\quad = \mathcal{CB}[b]$:BRANCH$(\mathcal{CS}[S_1],\mathcal{CS}[S_2])$

$\mathcal{CS}[\texttt{while } b \texttt{ do } S]$
$\quad =$ LOOP$(\mathcal{CB}[b],\mathcal{CS}[S])$