1 Functions and Local Scopes in Stack Machine

To support functions and local scopes the stack machine has to be essentially re-designed.

First, we add a new notion — *location* ($\mathscr{Loc}$) — to the definition of stack machine. A location specifies where a non-stack operand of an instruction resides. For now the three kinds of locations are sufficient:

$$\begin{array}{rl}
\textbf{global}\ \mathscr{X} & \text{— global variable}\\
\textbf{local}\ \mathbb{N} & \text{— local variable}\\
\textbf{arg}\ \mathbb{N} & \text{— function argument}
\end{array}$$

Thus, now operands for instructions `ST`, `LD` and `LDA` are locations. Moreover, the set of *values* for stack machine now contains references to locations as well as plain integer numbers:

$$\mathscr{V} = \mathbb{Z} \mid \textbf{ref}\ \mathscr{Loc}$$

Next, we need a whole new bunch of instructions:

$$\begin{array}{rl}
\texttt{GLOBAL}\ \mathscr{X} & \text{— declaration of global variable}\\
\texttt{CALL}\ \mathscr{X}\ \mathbb{N} & \text{— function call}\\
\texttt{BEGIN}\ \mathscr{X}\ \mathbb{N}\ \mathbb{N} & \text{— begin of function}\\
\texttt{END} & \text{— end of function}
\end{array}$$

Next to last, in addition to a regular state we add the notion of *local state*:

$$\Sigma_{loc} = (\mathbb{N} \to \mathscr{V}) \times (\mathbb{N} \to \mathscr{V})$$

Local states keep values of arguments and local variables, indexed by their numbers, respectively.

Finally, we modify the configuration for stack machine:

$$\mathscr{C} = \mathscr{V}^* \times (\Sigma_{loc} \times \mathscr{P})^* \times (\Sigma_{loc} \times \Sigma) \times \mathscr{W}$$

In addition to a regular stack of values, global state and a world now the configurations contains two more items:

- a *control stack*, which is a stack of pairs of local state and programs, which keeps track of return points;
- a local state, which keeps a current local state.

For extended state we need to refedine the primitives for reading

$$\begin{array}{llcl}
\langle\langle a, l\rangle, g\rangle & [\textbf{local}\ n] & = & l\,(n)\\
\langle\langle a, l\rangle, g\rangle & [\textbf{arg}\ n] & = & a\,(n)\\
\langle\langle a, l\rangle, g\rangle & [\textbf{global}\ x] & = & g\,(x)
\end{array}$$

and the assignment

$$P \vdash c \xrightarrow{\varepsilon}_{\mathscr{SM}} c \qquad [\textsc{Stop}_{SM}]$$

$$\frac{P \vdash \langle (x \oplus y)s,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow{p}_{\mathscr{SM}} c'}{P \vdash \langle yxs,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow{[\texttt{BINOP}\,\otimes]p}_{\mathscr{SM}} c'} \qquad [\textsc{Binop}_{SM}]$$

$$\frac{P \vdash \langle zs,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow{p}_{\mathscr{SM}} c'}{P \vdash \langle s,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow{[\texttt{CONST}\,z]p}_{\mathscr{SM}} c'} \qquad [\textsc{Const}_{SM}]$$

$$\frac{P \vdash \langle z,\, \omega'\rangle = \textbf{read}\ \omega,\ \langle zs,\, s_c,\, \sigma,\, \omega'\rangle \xrightarrow{p}_{\mathscr{SM}} c'}{P \vdash \langle s,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow{\texttt{READ}\,p}_{\mathscr{SM}} c'} \qquad [\textsc{Read}_{SM}]$$

$$\frac{P \vdash \langle s,\, s_c,\, \sigma,\, \textbf{write}\ z\omega\rangle \xrightarrow{p}_{\mathscr{SM}} c'}{P \vdash \langle zs,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow{\texttt{WRITE}\,p}_{\mathscr{SM}} c'} \qquad [\textsc{Write}_{SM}]$$

$$\frac{P \vdash \langle s,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow{p}_{\mathscr{SM}} c'}{P \vdash \langle xs,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow{[\texttt{DROP}]p}_{\mathscr{SM}} c'} \qquad [\textsc{Drop}_{SM}]$$

Figure 1: Stack machine: basic rules

$$\begin{array}{lll}
\langle\langle a,\, l\rangle,\, g\rangle & [\textbf{local}\ n \leftarrow v] & = \langle\langle a,\, l[i \leftarrow v]\rangle,\, g\rangle \\
\langle\langle a,\, l\rangle,\, g\rangle & [\textbf{arg}\ n \leftarrow v] & = \langle\langle a[i \leftarrow v],\, l\rangle,\, g\rangle \\
\langle\langle a,\, l\rangle,\, g\rangle & [\textbf{global}\ x \leftarrow v] & = \langle\langle a,\, l\rangle,\, g[x \leftarrow v]\rangle
\end{array}$$

Now we need to specify the operational semantics for the stack machine (see Fig. 1 – Fig. 4). The primitive **createLocal** is defined as follows:

$$\textbf{createLocal}\ s\ n_a\ n_l = \langle s[n_a...],\, \langle [i \in [0..n_a-1] \mapsto s[n_a-i-1]],\, [i \in [0..n_l-1] \mapsto 0]\rangle\rangle\}$$

$$\frac{P \vdash \langle [\sigma(x)]s,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow[\mathscr{S}M]{p} c'}{P \vdash \langle s,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow[\mathscr{S}M]{[\texttt{LD}\, x]p} c'} \quad [\mathrm{LD}_{SM}]$$

$$\frac{P \vdash \langle [\mathbf{ref}\, x]s,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow[\mathscr{S}M]{p} c'}{P \vdash \langle s,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow[\mathscr{S}M]{[\texttt{LDA}\, x]p} c'} \quad [\mathrm{LDA}_{SM}]$$

$$\frac{P \vdash \langle vs,\, s_c,\, \sigma[x \leftarrow v],\, \omega\rangle \xrightarrow[\mathscr{S}M]{p} c'}{P \vdash \langle v[\mathbf{ref}\, x]s,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow[\mathscr{S}M]{[\texttt{STI}]p} c'} \quad [\mathrm{STI}_{SM}]$$

$$\frac{\langle zs,\, s_c,\, \sigma[x \leftarrow z],\, \omega\rangle \xrightarrow[\mathscr{S}M]{p} c'}{\langle zs,\, s_c,\, \sigma,\, \omega\rangle \xrightarrow[\mathscr{S}M]{[\texttt{ST}\, x]p} c'} \quad [\mathrm{ST}_{SM}]$$

Figure 2: Stack machine: state operations

$$\frac{P \vdash c \xRightarrow{p}_{\mathscr{S}M} c'}{P \vdash c \xRightarrow{[\texttt{LABEL}\ l]p}_{\mathscr{S}M} c'} \qquad [\textsc{Label}_{SM}]$$

$$\frac{P \vdash c \xRightarrow{P[l]}_{\mathscr{S}M} c'}{P \vdash c \xRightarrow{[\texttt{JMP}\ l]p}_{\mathscr{S}M} c'} \qquad [\text{JMP}_{SM}]$$

$$\frac{z \neq 0, \quad P \vdash \langle s, s_c, \sigma, \omega\rangle \xRightarrow{P[l]}_{\mathscr{S}M} c'}{P \vdash \langle zs, s_c, \sigma, \omega\rangle \xRightarrow{[\texttt{CJMP}_{nz}\ l]p}_{\mathscr{S}M} c'} \qquad [\text{CJMP}^{+}_{nz\,SM}]$$

$$\frac{z = 0, \quad P \vdash \langle s, s_c, \sigma, \omega\rangle \xRightarrow{p}_{\mathscr{S}M} c'}{P \vdash \langle zs, s_c, \sigma, \omega\rangle \xRightarrow{[\texttt{CJMP}_{nz}\ l]p}_{\mathscr{S}M} c'} \qquad [\text{CJMP}^{-}_{nz\,SM}]$$

$$\frac{z = 0, \quad P \vdash \langle s, s_c, \sigma, \omega\rangle \xRightarrow{P[l]}_{\mathscr{S}M} c'}{P \vdash \langle zs, s_c, \sigma, \omega\rangle \xRightarrow{[\texttt{CJMP}_{z}\ l]p}_{\mathscr{S}M} c'} \qquad [\text{CJMP}^{+}_{z\,SM}]$$

$$\frac{z \neq 0, \quad P \vdash \langle s, s_c, \sigma, \omega\rangle \xRightarrow{p}_{\mathscr{S}M} c'}{P \vdash \langle zs, s_c, \sigma, \omega\rangle \xRightarrow{[\texttt{CJMP}_{z}\ l]p}_{\mathscr{S}M} c'} \qquad [\text{CJMP}^{-}_{z\,SM}]$$

Figure 3: Stack machine: control flow instructions

$$P \vdash \langle s, \varepsilon, \sigma, \omega\rangle \xrightarrow[\mathscr{S}M]{[\texttt{END}]p} \langle s, \varepsilon, \sigma, \omega\rangle \qquad [\textsc{EndStop}_{SM}]$$

$$\frac{P \vdash \langle s, s_c, \langle \sigma_l, \sigma\rangle, \omega\rangle \xrightarrow[\mathscr{S}M]{q} c'}{P \vdash \langle s, \langle \sigma_l, q\rangle\, s_c, \langle \_, \sigma\rangle, \omega\rangle \xrightarrow[\mathscr{S}M]{[\texttt{END}]p} c'} \qquad [\textsc{End}_{SM}]$$

$$\frac{\langle s', \sigma_l\rangle = \textbf{createLocal}\; s\; n_a\; n_l \quad P \vdash \langle s', s_c, \langle \sigma_l, \sigma\rangle, \omega\rangle \xrightarrow[\mathscr{S}M]{p} c'}{P \vdash \langle s, s_c, \langle \_, \sigma\rangle, \omega\rangle \xrightarrow[\mathscr{S}M]{[\texttt{BEGIN}\ \_\ n_a\ n_l]p} c'} \qquad [\textsc{Begin}_{SM}]$$

$$\frac{P \vdash \langle s, \langle \sigma_l, p\rangle\, s_c, \langle \sigma_l, \sigma\rangle, \omega\rangle \xrightarrow[\mathscr{S}M]{P[f]} c'}{P \vdash \langle s, s_c, \langle \sigma_l, \sigma\rangle, \omega\rangle \xrightarrow[\mathscr{S}M]{[\texttt{CALL}\ f\ \_]p} c'} \qquad [\textsc{Call}_{SM}]$$

Figure 4: Stack machine: functions, call, return