

1 Functions and Local Scopes in Stack Machine

To support functions and local scopes the stack machine has to be essentially redesigned.

First, we add a new notion — location (*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:

global \mathcal{X} — global variable
local \mathbb{N} — local variable
arg \mathbb{N} — function argument

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:

$$\mathcal{V} = \mathbb{Z} \mid \mathbf{ref} \text{ } Loc$$

Next, we need a whole new bunch of instructions:

GLOBAL \mathcal{X} — declaration of global variable
 CALL $\mathcal{X} \mathbb{N}$ — function call
 BEGIN $\mathcal{X} \mathbb{N} \mathbb{N}$ — begin of function
 END — end of function

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

$$\Sigma_{loc} = (\mathbb{N} \rightarrow \mathcal{V}) \times (\mathbb{N} \rightarrow \mathcal{V})$$

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

Finally, we modify the configuration for stack machine:

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

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 redefine the primitives for reading

$$\begin{aligned} \langle \langle a, l \rangle, g \rangle \quad [\mathbf{local} \ n] &= l(n) \\ \langle \langle a, l \rangle, g \rangle \quad [\mathbf{arg} \ n] &= a(n) \\ \langle \langle a, l \rangle, g \rangle \quad [\mathbf{global} \ x] &= g(x) \end{aligned}$$

and the assignment

$$\begin{array}{c}
P \vdash c \xrightarrow{\varepsilon}_{\mathcal{S}_M} c \quad [\text{Stop}_{SM}] \\
\\
\frac{P \vdash \langle (x \oplus y)s, s_c, \sigma, \omega \rangle \xrightarrow{P}_{\mathcal{S}_M} c'}{P \vdash \langle yxs, s_c, \sigma, \omega \rangle \xrightarrow{[\text{BINOP } \otimes]p}_{\mathcal{S}_M} c'} \quad [\text{Binop}_{SM}] \\
\\
\frac{P \vdash \langle zs, s_c, \sigma, \omega \rangle \xrightarrow{P}_{\mathcal{S}_M} c'}{P \vdash \langle s, s_c, \sigma, \omega \rangle \xrightarrow{[\text{CONST } z]p}_{\mathcal{S}_M} c'} \quad [\text{Const}_{SM}] \\
\\
\frac{P \vdash \langle z, \omega' \rangle = \mathbf{read} \, \omega, \langle zs, s_c, \sigma, \omega' \rangle \xrightarrow{P}_{\mathcal{S}_M} c'}{P \vdash \langle s, s_c, \sigma, \omega \rangle \xrightarrow{\text{READ } p}_{\mathcal{S}_M} c'} \quad [\text{Read}_{SM}] \\
\\
\frac{P \vdash \langle s, s_c, \sigma, \mathbf{write} \, z \omega \rangle \xrightarrow{P}_{\mathcal{S}_M} c'}{P \vdash \langle zs, s_c, \sigma, \omega \rangle \xrightarrow{\text{WRITE } p}_{\mathcal{S}_M} c'} \quad [\text{Write}_{SM}] \\
\\
\frac{P \vdash \langle s, s_c, \sigma, \omega \rangle \xrightarrow{P}_{\mathcal{S}_M} c'}{P \vdash \langle xs, s_c, \sigma, \omega \rangle \xrightarrow{[\text{DROP}]p}_{\mathcal{S}_M} c'} \quad [\text{Drop}_{SM}]
\end{array}$$

Figure 1: Stack machine: basic rules

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

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

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

$$\begin{array}{c}
\frac{P \vdash \langle [\sigma(x)]s, s_c, \sigma, \omega \rangle \xRightarrow[\mathcal{SM}]{P} c'}{P \vdash \langle s, s_c, \sigma, \omega \rangle \xRightarrow[\mathcal{SM}]{[\text{LD } x]p} c'} \quad [\text{LD}_{SM}] \\
\\
\frac{P \vdash \langle [\mathbf{ref } x]s, s_c, \sigma, \omega \rangle \xRightarrow[\mathcal{SM}]{P} c'}{P \vdash \langle s, s_c, \sigma, \omega \rangle \xRightarrow[\mathcal{SM}]{[\text{LDA } x]p} c'} \quad [\text{LDA}_{SM}] \\
\\
\frac{P \vdash \langle vs, s_c, \sigma[x \leftarrow v], \omega \rangle \xRightarrow[\mathcal{SM}]{P} c'}{P \vdash \langle v[\mathbf{ref } x]s, s_c, \sigma, \omega \rangle \xRightarrow[\mathcal{SM}]{[\text{STI}]p} c'} \quad [\text{STI}_{SM}] \\
\\
\frac{\langle zs, s_c, \sigma[x \leftarrow z], \omega \rangle \xRightarrow[\mathcal{SM}]{P} c'}{\langle zs, s_c, \sigma, \omega \rangle \xRightarrow[\mathcal{SM}]{[\text{ST } x]p} c'} \quad [\text{ST}_{SM}]
\end{array}$$

Figure 2: Stack machine: state operations

$$\begin{array}{c}
\frac{P \vdash c \xRightarrow{P} c'}{\mathcal{S}M} \\
\hline
P \vdash c \xRightarrow{[\text{LABEL } l]p} c' \quad \mathcal{S}M \quad [\text{Label}_{SM}]
\end{array}$$

$$\begin{array}{c}
\frac{P \vdash c \xRightarrow{P[l]} c'}{\mathcal{S}M} \\
\hline
P \vdash c \xRightarrow{[\text{JMP } l]p} c' \quad \mathcal{S}M \quad [\text{JMP}_{SM}]
\end{array}$$

$$\begin{array}{c}
\frac{z \neq 0, \quad P \vdash \langle s, s_c, \sigma, \omega \rangle \xRightarrow{P[l]} c'}{\mathcal{S}M} \\
\hline
P \vdash \langle zs, s_c, \sigma, \omega \rangle \xRightarrow{[\text{CJMP}_{nz} l]p} c' \quad \mathcal{S}M \quad [\text{CJMP}_{nzSM}^+]
\end{array}$$

$$\begin{array}{c}
\frac{z = 0, \quad P \vdash \langle s, s_c, \sigma, \omega \rangle \xRightarrow{P} c'}{\mathcal{S}M} \\
\hline
P \vdash \langle zs, s_c, \sigma, \omega \rangle \xRightarrow{[\text{CJMP}_{nz} l]p} c' \quad \mathcal{S}M \quad [\text{CJMP}_{nzSM}^-]
\end{array}$$

$$\begin{array}{c}
\frac{z = 0, \quad P \vdash \langle s, s_c, \sigma, \omega \rangle \xRightarrow{P[l]} c'}{\mathcal{S}M} \\
\hline
P \vdash \langle zs, s_c, \sigma, \omega \rangle \xRightarrow{[\text{CJMP}_z l]p} c' \quad \mathcal{S}M \quad [\text{CJMP}_zSM^+]
\end{array}$$

$$\begin{array}{c}
\frac{z \neq 0, \quad P \vdash \langle s, s_c, \sigma, \omega \rangle \xRightarrow{P} c'}{\mathcal{S}M} \\
\hline
P \vdash \langle zs, s_c, \sigma, \omega \rangle \xRightarrow{[\text{CJMP}_z l]p} c' \quad \mathcal{S}M \quad [\text{CJMP}_zSM^-]
\end{array}$$

Figure 3: Stack machine: control flow instructions

$$\begin{array}{c}
P \vdash \langle s, \varepsilon, \sigma, \omega \rangle \xRightarrow{[\text{END}]p}_{\mathcal{SM}} \langle s, \varepsilon, \sigma, \omega \rangle \quad [\text{EndStop}_{SM}] \\
\\
\frac{P \vdash \langle s, s_c, \langle \sigma_l, \sigma \rangle, \omega \rangle \xRightarrow{q}_{\mathcal{SM}} c'}{P \vdash \langle s, \langle \sigma_l, q \rangle s_c, \langle _, \sigma \rangle, \omega \rangle \xRightarrow{[\text{END}]p}_{\mathcal{SM}} c'} \quad [\text{End}_{SM}] \\
\\
\frac{\langle s', \sigma_l \rangle = \mathbf{createLocal} \ s \ n_a \ n_l \quad P \vdash \langle s', s_c, \langle \sigma_l, \sigma \rangle, \omega \rangle \xRightarrow{p}_{\mathcal{SM}} c'}{P \vdash \langle s, s_c, \langle _, \sigma \rangle, \omega \rangle \xRightarrow{[\text{BEGIN} \ _ \ n_a \ n_l]p}_{\mathcal{SM}} c'} \quad [\text{Begin}_{SM}] \\
\\
\frac{P \vdash \langle s, \langle \sigma_l, p \rangle s_c, \langle \sigma_l, \sigma \rangle, \omega \rangle \xRightarrow{P[f]}_{\mathcal{SM}} c'}{P \vdash \langle s, s_c, \langle \sigma_l, \sigma \rangle, \omega \rangle \xRightarrow{[\text{CALL} \ f \ _]p}_{\mathcal{SM}} c'} \quad [\text{Call}_{SM}]
\end{array}$$

Figure 4: Stack machine: functions, call, return