

1 Arrays

Arrays are added at the expression level by means of the following extension:

$$\begin{aligned} \mathcal{E} \quad + = \quad & [\mathcal{E}^*] \\ & \mathcal{E}[\mathcal{E}] \\ & \mathcal{E} . \mathbf{length} \\ & \mathbf{elemRef} \ \mathcal{E}[\mathcal{E}] \end{aligned}$$

Additional well-formedness rules for the new constructs:

$$\begin{array}{ccc} \frac{\mathbf{Val} \vdash e_1 \dots \mathbf{Val} \vdash e_k}{\mathbf{Val} \vdash [e_1, \dots, e_k]} & \frac{\mathbf{Val} \vdash e_1 \dots \mathbf{Val} \vdash e_k}{\mathbf{Weak} \vdash [e_1, \dots, e_k]} & \frac{\mathbf{Val} \vdash e_1 \dots \mathbf{Val} \vdash e_k}{\mathbf{Void} \vdash \mathbf{ignore} [e_1, \dots, e_k]} \\ \frac{\mathbf{Val} \vdash e \quad \mathbf{Val} \vdash i}{\mathbf{Val} \vdash e[i]} & \frac{\mathbf{Val} \vdash e \quad \mathbf{Val} \vdash i}{\mathbf{Weak} \vdash e[i]} & \frac{\mathbf{Val} \vdash e \quad \mathbf{Val} \vdash i}{\mathbf{Void} \vdash \mathbf{ignore} e[i]} \\ & \frac{\mathbf{Val} \vdash e \quad \mathbf{Val} \vdash i}{\mathbf{Ref} \vdash \mathbf{elemRef} e[i]} \end{array}$$

2 Operational Semantics

An array can be represented as a pair: the length of the array and a mapping from indices to elements. If we denote X the set of elements then the set of all arrays $\mathcal{A}(X)$ can be defined as follows:

$$\mathcal{A}(X) = \mathbb{N} \times (\mathbb{N} \rightarrow X)$$

For an array $\langle n, f \rangle$ we assume $\text{dom } f = [0..n-1]$. An element selection function:

$$\begin{aligned} \bullet[\bullet] : \mathcal{A}(X) &\rightarrow \mathbb{N} \rightarrow X \\ \langle n, f \rangle [i] &= \begin{cases} f(i) & , \quad i < n \\ \perp & , \quad \text{otherwise} \end{cases} \end{aligned}$$

We represent arrays by *references*. Thus, we introduce a (linearly) ordered set of locations

$$\mathcal{L} = \{l_0, l_1, \dots\}$$

Now, the set of all values the programs operate on can be described as follows:

$$\mathcal{V} = \mathbb{Z} \mid \mathcal{L}$$

To access arrays, we introduce an abstraction of memory:

$$\mathcal{M} = \mathcal{L} \rightarrow \mathcal{A}(\mathcal{V})$$

We now add two more components to the configurations: a memory function μ and the first free memory location l_m , and define the following primitive

$$\mathbf{mem} \langle \sigma, \omega, \mu, l_m \rangle = \mu$$

which gives a memory function from a configuration.

The definition of state does not change, hence all existing rules are preserved (modulo adding additional components to configurations) The rules for the new kinds of expressions are as follows:

$$\frac{c \xrightarrow{e_0, \dots, e_k}_{\mathcal{E}^*} \langle \langle \sigma, \omega, \mu, l \rangle, v_1, \dots, v_k \rangle}{c \xrightarrow{[e_0, \dots, e_k]}_{\mathcal{E}} \langle \langle \sigma, \omega, \mu[l \leftarrow \langle k+1, i \mapsto v_i \rangle], l+1 \rangle, l \rangle} \quad [\text{ARRAY}]$$

$$\frac{c \xrightarrow{ei}_{\mathcal{E}^*} \langle c', lv \rangle \quad l \in \mathcal{L} \quad v \in \mathbb{Z}}{c \xrightarrow{e[i]}_{\mathcal{E}} \langle c', ((\mathbf{mem} \ c')(l))[i] \rangle} \quad [\text{ARRAYELEM}]$$

$$\frac{c \xrightarrow{ei}_{\mathcal{E}^*} \langle c', lv \rangle \quad l \in \mathcal{L} \quad v \in \mathbb{Z}}{c \xrightarrow{\mathbf{elemRef} \ e \ [i]}_{\mathcal{E}} \langle c', \mathbf{elemRef} \ l \ v \rangle} \quad [\text{ARRAYELEMREF}]$$

$$\frac{c \xrightarrow{e}_{\mathcal{E}} \langle c', l \rangle \quad l \in \mathcal{L}}{c \xrightarrow{e. \ \mathbf{length}}_{\mathcal{E}} \langle c', \mathbf{fst} \ (\mathbf{mem} \ c')(l) \rangle} \quad [\text{ARRAYLENGTH}]$$

We also need one additional rule for assignment:

$$\frac{c \xrightarrow{lr}_{\mathcal{E}^*} \langle \langle \sigma, \omega, \mu, l \rangle, [\mathbf{elemRef} \ a \ i]v \rangle \quad a \in \mathcal{L}}{c \xrightarrow{l := r}_{\mathcal{E}} \langle \langle \sigma, \omega, \mu[a \leftarrow \langle \mathbf{fst} \ \mu(a), (\mathbf{snd} \ \mu(a))[i \leftarrow v] \rangle], l \rangle, v \rangle} \quad [\text{ASSIGNARRAY}]$$

3 Stack Machine

In stack machine we add the following new instructions:

$$\mathcal{I} \quad + = \quad \text{ARRAY } \mathbb{N} \\ \text{ELEM} \\ \text{STA}$$

We also add memory function and current location components to the configuration; as state components are preserved, all rules are preserved as well. The new rules are:

$$\frac{P \vdash \langle s[n, \dots], s_c, \sigma, \omega, \mu[l \leftarrow \langle n, i \mapsto s[n-i-1] \rangle], l+1 \rangle \xrightarrow{P}_{\mathcal{S.M.}} c}{P \vdash \langle s, s_c, \sigma, \omega, \mu, l \rangle \xrightarrow{[\text{ARRAY } n]P}_{\mathcal{S.M.}} c} \quad [\text{ARRAY}_{\mathcal{S.M.}}]$$

$$\frac{P \vdash \langle [(\mu(a))[i]]s, s_c, \sigma, \omega, \mu, l \rangle \xrightarrow{P}_{\mathcal{S.M.}} c}{P \vdash \langle ias, s_c, \sigma, \omega, \mu, l \rangle \xrightarrow{[\text{ELEM}]P}_{\mathcal{S.M.}} c} \quad [\text{ELEM}_{\mathcal{S.M.}}]$$

$$\frac{P \vdash \langle vs, s_c, \sigma, \omega, \mu[a \leftarrow \langle \mathbf{fst}(\mu(a)), (\mathbf{snd}(\mu(a)))[i \leftarrow v] \rangle], l \rangle \xrightarrow{p} \mathcal{S.M} c}{P \vdash \langle \mathit{vias}, s_c, \sigma, \omega, \mu, l \rangle \xrightarrow{[\mathbf{STA}]p} \mathcal{S.M} c} \quad [\mathbf{STA}_{\mathcal{S.M}}]$$