1 Arrays

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

$$\begin{array}{lll} \mathscr{E} & += & [\mathscr{E}^*] \\ & & \mathscr{E}\,[\mathscr{E}] \\ & & \mathscr{E}\ .\ \text{length} \\ & & \text{elemRef}\,\mathscr{E}\,[\mathscr{E}] \end{array}$$

Additional well-formedness rules for the new constructs:

$$\frac{\mathbf{Val} \vdash e_1 \ldots \mathbf{Val} \vdash e_k}{\mathbf{Val} \vdash [e_1, \ldots, e_k]} \qquad \frac{\mathbf{Val} \vdash e_1 \ldots \mathbf{Val} \vdash e_k}{\mathbf{Weak} \vdash [e_1, \ldots, e_k]} \qquad \frac{\mathbf{Val} \vdash e_1 \ldots \mathbf{Val} \vdash e_k}{\mathbf{Void} \vdash \text{ignore}\ \ [e_1, \ldots, e_k]}$$

$$\frac{\mathbf{Val} \vdash e \quad \mathbf{Val} \vdash i}{\mathbf{Val} \vdash e\,[\,i\,]} \qquad \frac{\mathbf{Val} \vdash e \quad \mathbf{Val} \vdash i}{\mathbf{Weak} \vdash e\,[\,i\,]} \qquad \frac{\mathbf{Val} \vdash e \quad \mathbf{Val} \vdash i}{\mathbf{Void} \vdash \text{ignore}\ e\,[\,i\,]}$$

$$\frac{\mathbf{Val} \vdash e \quad \mathbf{Val} \vdash i}{\mathbf{Ref} \vdash \text{elemRef}\,e\,[\,i\,]}$$

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 $\mathscr{A}(\mathscr{X})$ can be defined as follows:

$$\mathscr{A}(X) = \mathbb{N} \times (\mathbb{N} \to X)$$

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

$$\bullet[\bullet] : \mathscr{A}(X) \to \mathbb{N} \to X$$

$$\langle n, f \rangle\,[i] = \begin{cases} f(i) & , \quad i < n \\ \bot & , \quad \text{otherwise} \end{cases}$$

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

$$\mathscr{L} = \{l_0, l_1, \ldots\}$$

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

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

To access arrays, we introduce an abstraction of memory:

$$\mathscr{M} = \mathscr{L} \to \mathscr{A}\,(\mathscr{V})$$

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 additinal components to configurations) The rules for the new kinds of expressions are as follows:

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

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

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

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

We also need one additional rule for assignment:

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

3 Stack Machine

In stack machine we add the following new instructions:

$$\begin{array}{lll} \mathscr{I} & += & \text{ARRAY}\ \mathbb{N} \\ & & \text{ELEM} \\ & & \text{STA} \end{array}$$

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:

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

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

$$\frac{P \vdash \langle vs, s_c, \sigma, \omega, \mu[a \leftarrow \langle \textbf{fst } (\mu(a)), (\textbf{snd } (\mu(a)))[i \leftarrow v]\rangle], l\rangle \xRightarrow[\mathscr{S}\mathscr{M}]{p} c}{P \vdash \langle vias, s_c, \sigma, \omega, \mu, l\rangle \xRightarrow[\mathscr{S}\mathscr{M}]{[\text{STA}]p} c} \quad [\text{STA}_{\mathscr{S}\mathscr{M}}]$$