

1 Virtual Machine Abstract Syntax

S	$::=$	$d; S$ $ $ e	<i>statements</i>		
d	$::=$	$\text{val } f = e$ $ $ $\text{def } m(\overline{x : \tau}) : \tau = S$ $ $ $\text{type } L = \tau$	<i>declarations</i>	β	$::=$ Unit $ $ L <i>base type</i>
e	$::=$	x $ $ $\text{new } x : \tau \{ \overline{d} \}$ $ $ $e.m(\overline{e})$ $ $ $e.f$ $ $ \mathcal{L}	<i>expressions</i>	τ	$::=$ $\beta\{\overline{\sigma}\}$ <i>type</i>
σ	$::=$	$\text{val } f : \tau$ $ $ $\text{def } m(\overline{x : \tau}) : \tau$ $ $ $\text{type } L = \tau$	<i>decl type</i>		
\mathcal{L}	$::=$	n	<i>literals</i>		

Notation: overbar means a list of elements, as in Java

2 Standard prelude

```

type Int
def +(i : Int) : Int
def -(i : Int) : Int
def *(i : Int) : Int
def /(i : Int) : Int

```

3 Virtual Machine Typing Rules

$\boxed{\Gamma \vdash e : \tau}$

$$\frac{\Gamma \vdash d : \sigma \quad \Gamma, \sigma \vdash S : \tau}{\Gamma \vdash d; S : \tau} \text{ (T-STMT)}$$

$$\frac{x : \tau \in \Gamma}{\Gamma \vdash x : \tau} \text{ (T-VAR)}$$

$$\frac{\Gamma \mid x : \tau \vdash \overline{d} : \text{unfold}_{\Gamma}(\tau)}{\Gamma \vdash \text{new } x : \tau \{ \overline{d} \} : \tau} \text{ (T-NEW)}$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \text{def } m(x : \tau_2) : \tau \in \text{unfold}_{\Gamma}(\tau_1) \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1.m(e_2) : \tau} \text{ (T-INVK)}$$

$$\frac{\Gamma \vdash e : \tau' \quad \text{val } f : \tau \in \text{unfold}_{\Gamma}(\tau')}{\Gamma \vdash e.f : \tau} \text{ (T-FIELD)}$$

$$\overline{\Gamma \vdash n : \text{Int}} \text{ (T-INT)}$$

Technically Γ is a list of σ , but we often write $x : \tau$ for $\text{val } x : \tau$.

$$\boxed{\Gamma \mid x : \tau \vdash d : \sigma}$$

$$\frac{\Gamma, x : \tau \vdash \bar{\tau} \text{ wf} \quad \Gamma, \bar{y} : \bar{\tau} \vdash e : \tau_2}{\Gamma \mid x : \tau \vdash \text{def } m(\bar{y} : \bar{\tau}) : \tau_2 = e : \text{def } m(\bar{y} : \bar{\tau}) : \tau_2} \text{ (T-DEF)}$$

$$\frac{\Gamma \vdash e : \tau}{\Gamma \mid x : \tau \vdash \text{val } f = e : \text{val } f : \tau} \text{ (T-VAL)}$$

$$\frac{\Gamma, x : \tau \vdash \text{type } L = \tau \text{ wf}}{\Gamma \mid x : \tau \vdash \text{type } L = \tau : \text{type } L = \tau} \text{ (T-TYPE)}$$

$$\boxed{\text{unfold}_\Gamma(\tau) = \bar{\sigma}}$$

$$\overline{\text{unfold}_\Gamma(\text{Unit})} = \bullet$$

$$\frac{\text{type } L = \tau \in \Gamma \quad \text{unfold}_\Gamma(\tau) = \bar{\sigma}}{\text{unfold}_\Gamma(L) = \bar{\sigma}}$$

$$\frac{\text{unfold}_\Gamma(\beta) = \bar{\sigma}}{\text{unfold}_\Gamma(\beta\{\bar{\sigma}'\}) = \bar{\sigma} \leftarrow \bar{\sigma}'}$$

Note: $\bar{\sigma} \leftarrow \bar{\sigma}'$ means that we append the two lists, except that when the same symbol is defined in both $\bar{\sigma}$ and $\bar{\sigma}'$, we include only the (overriding) definition in $\bar{\sigma}'$.

Now, finally, type and declaration type well-formedness rules: $\boxed{\Gamma \vdash \tau \text{ wf}}$

$$\overline{\Gamma \vdash \text{Unit} \text{ wf}}$$

$$\frac{\text{unfold}_\Gamma(L) = \bar{\sigma}}{\Gamma \vdash L \text{ wf}}$$

$$\frac{\Gamma \vdash \beta \text{ wf} \quad \Gamma \vdash \bar{\sigma}' \text{ wf}}{\Gamma \vdash \beta\{\bar{\sigma}'\} \text{ wf}}$$

$$\boxed{\Gamma \vdash \sigma \text{ wf}}$$

$$\frac{\Gamma \vdash \bar{y} : \bar{\tau} \text{ wf} \quad \Gamma \vdash \tau_2 \text{ wf}}{\Gamma \vdash \text{def } m(\bar{y} : \bar{\tau}) : \tau_2 \text{ wf}}$$

$$\frac{\Gamma \vdash \tau \text{ wf}}{\Gamma \vdash \text{val } f : \tau \text{ wf}}$$

$$\frac{\Gamma, \text{type } L = \tau \vdash \tau \text{ wf}}{\Gamma \vdash \text{type } L = \tau \text{ wf}}$$