0.1 Terms

Making the language no-longer differentiate between values and computations.

0.1.1 Value Terms

$$
\begin{aligned}
v ::= & x \\
& \mid \lambda x : A.v \\
& \mid \mathtt{C}^A \\
& \mid \mathtt{()} \\
& \mid \mathtt{true} \mid \mathtt{false} \\
& \mid \Lambda\alpha.v \\
& \mid v\ \epsilon \\
& \mid \mathtt{if}_A\ v\ \mathtt{then}\ v_1\ \mathtt{else}\ v_2 \\
& \mid v_1\ v_2 \\
& \mid \mathtt{do}\ x \leftarrow v_1\ \mathtt{in}\ v_2 \\
& \mid \mathtt{return} v
\end{aligned} \tag{1}
$$

0.2 Type System

0.2.1 Ground Effects

The effects should form a monotonous, pre-ordered monoid $(E, \cdot, \mathtt{1}, \leq)$ with ground elements e.

0.2.2 Effect Po-Monoid Under a Effect Environment

Derive a new Po-Monoid for each Φ:

$$(E_\Phi, \cdot_\Phi, \mathtt{1}, \leq_\Phi) \tag{2}$$

Where meta-variables, ϵ, range over E_Φ Where

$$E_\Phi = E \cup \{\alpha \mid \alpha \in \Phi\} \tag{3}$$

And

$$() \frac{\epsilon_3 = \epsilon_1 \cdot \epsilon_2}{\epsilon_3 = \epsilon_1 \cdot_\Phi \epsilon_2} \tag{4}$$

Otherwise, $\cdot_\Phi$ is symbolic in nature.

$$\epsilon_1 \leq_\Phi \epsilon_2 \Leftrightarrow \forall \sigma \downarrow .\epsilon_1 [\sigma \downarrow] \leq \epsilon_2 [\sigma \downarrow] \tag{5}$$

Where $\sigma \downarrow$ denotes any ground-substitution of Φ. That is any substitution of all effect-variables in Φ to ground effects. Where it is obvious from the context, I shall use $\leq$ instead of $\leq_\Phi$.

0.2.3 Types

Ground Types There exists a set γ of ground types, including `Unit`, `Bool`

Term Types

$$A, B, C ::= \gamma \mid A \to B \mid \mathtt{M}_\epsilon A \mid \forall \alpha.A$$

0.2.4 Type and Effect Environments

A type environment is a snoc-list of tern-variable, type pairs, $G ::= \diamond \mid \Gamma, x : A$. An effect environment is a snoc-list of effect-variables.

$\Phi ::= \diamond \mid \Phi, \alpha$

Domain Function on Type Environments

- $\texttt{dom}(\diamond) = \emptyset$
- $\texttt{dom}(\Gamma, x : A) = \texttt{dom}(\Gamma) \cup \{x\}$

Membership of Effect Environments Informally, $\alpha \in \Phi$ if α appears in the list represented by Φ.

Ok Predicate On Effect Environments

- (Atom) $\frac{}{\diamond \texttt{Ok}}$
- (A) $\frac{\Phi \texttt{Ok} \quad \alpha \notin \Phi}{\Phi, \alpha \texttt{Ok}}$

Well-Formed-ness of effects We define a relation $\Phi \vdash \epsilon$.

- (Ground) $\frac{\Phi \texttt{Ok}}{\Phi \vdash e}$
- (Var) $\frac{\Phi, \alpha \texttt{Ok}}{\Phi, \alpha \vdash \alpha}$
- (Weaken) $\frac{\Phi \vdash \alpha}{\Phi, \beta \vdash \alpha}$ (if $\alpha \neq \beta$)
- (Monoid Op) $\frac{\Phi \vdash \epsilon_1 \quad \Phi \vdash \epsilon_2}{\Phi \vdash \epsilon_1 \cdot \epsilon_2}$

Well-Formed-ness of Types We define a relation $\Phi \vdash \tau$ on types.

- (Ground) $\frac{}{\Phi \vdash \gamma}$
- (Lambda) $\frac{\Phi \vdash A \quad \Phi \vdash B}{\Phi \vdash A \to B}$
- (Computation) $\frac{\Phi \vdash A \quad \Phi \vdash \epsilon}{\Phi \vdash \texttt{M}_\epsilon A}$
- (For-All) $\frac{\Phi, \alpha \vdash A}{\Phi \vdash \forall \alpha . A}$

Ok Predicate on Type Environments We now define a predicate on type environments and effect environments: $\Phi \vdash \Gamma \texttt{Ok}$

- (Nil) $\frac{}{\Phi \vdash \diamond \texttt{Ok}}$
- (Var) $\frac{\Phi \vdash \Gamma \texttt{Ok} \quad x \notin \texttt{dom}(\Gamma) \quad \Phi \vdash A}{\Phi \vdash \Gamma, x : A \texttt{Ok}}$

0.2.5 Sub-typing

There exists a sub-typing pre-order relation $\leq:_\gamma$ over ground types that is:

- (Reflexive) $\frac{}{A \leq:_\gamma A}$
- (Transitive) $\frac{A \leq:_\gamma B \quad B \leq:_\gamma C}{A \leq:_\gamma C}$

We extend this relation with the function and effect-lambda sub-typing rules to yield the full sub-typing relation under an effect environment, Φ, $\leq:_\Phi$

- (ground) $\frac{A \leq:_\gamma B}{A \leq:_\Phi B}$
- (Fn) $\frac{A \leq:_\Phi A' \quad B' \leq:_\Phi B}{A' \to B' \leq:_\Phi A \to B}$
- (All) $\frac{A \leq:_\Phi A'}{\forall \alpha . A \leq:_\Phi \forall a . A'}$
- (Effect) $\frac{A \leq:_\Phi B \quad \epsilon_1 \leq_\Phi \epsilon_2}{\mathtt{M}_{\epsilon_1} A \leq:_\Phi \mathtt{M}_{\epsilon_2} B}$

0.2.6 Type Rules

- (Const) $\frac{\Phi \vdash \Gamma \mathtt{Ok} \quad \Phi \vdash A}{\Phi | \Gamma \vdash \mathtt{C}^A : A}$
- (Unit) $\frac{\Phi \vdash \Gamma \mathtt{Ok}}{\Phi | \Gamma \vdash \mathtt{()} : \mathtt{Unit}}$
- (True) $\frac{\Phi \vdash \Gamma \mathtt{Ok}}{\Phi | \Gamma \vdash \mathtt{true} : \mathtt{Bool}}$
- (False) $\frac{\Phi \vdash \Gamma \mathtt{Ok}}{\Phi | \Gamma \vdash \mathtt{false} : \mathtt{Bool}}$
- (Var) $\frac{\Phi \vdash \Gamma , x : A \mathtt{Ok}}{\Phi | \Gamma , x : A \vdash x : A}$
- (Weaken) $\frac{\Phi | \Gamma \vdash x : A \quad \Phi \vdash B}{\Phi | \Gamma , y : B \vdash x : A}$ (if $x \neq y$)
- (Fn) $\frac{\Phi | \Gamma , x : A \vdash v : \beta}{\Phi | \Gamma \vdash \lambda x : A . v : A \to B}$
- (Sub) $\frac{\Phi | \Gamma \vdash v : A \quad A \leq:_\Phi B}{\Phi | \Gamma \vdash v : B}$
- (Effect-Abs) $\frac{\Phi , \alpha | \Gamma \vdash v : A}{\Phi | \Gamma \vdash \Lambda \alpha . v : \forall \alpha . A}$
- (Effect-apply) $\frac{\Phi | \Gamma \vdash v : \forall \alpha . A \quad \Phi \vdash \epsilon}{\Phi | \Gamma \vdash v \ \epsilon : A[\epsilon / \alpha]}$
- (Return) $\frac{\Phi | \Gamma \vdash v : A}{\Phi | \Gamma \vdash \mathtt{return} v : \mathtt{M}_{\mathtt{1}} A}$
- (Apply) $\frac{\Phi | \Gamma \vdash v_1 : A \to \mathtt{M}_\epsilon B \quad \Phi | \Gamma \vdash v_2 : A}{\Phi | \Gamma \vdash v_1 \ v_2 : \mathtt{M}_\epsilon B}$
- (If) $\frac{\Phi | \Gamma \vdash v : \mathtt{Bool} \quad \Phi | \Gamma \vdash v_1 : A \quad \Phi | \Gamma \vdash v_2 : A}{\Phi | \Gamma \vdash \mathtt{if}_A \ V \ \mathtt{then} \ v_1 \ \mathtt{else} \ v_2 : A}$
- (Do) $\frac{\Phi | \Gamma \vdash v_1 : \mathtt{M}_{\epsilon_1} A \quad \Phi | \Gamma , x : A \vdash v_2 : \mathtt{M}_{\epsilon_2} B}{\Phi | \Gamma \vdash \mathtt{do} \ x \leftarrow v_1 \ \mathtt{in} \ v_2 : \mathtt{M}_{\epsilon_1 \cdot \epsilon_2} B}$

0.2.7 Ok Lemma

If $\Phi \mid \Gamma \vdash t : \tau$ then $\Phi \vdash \Gamma \mathtt{Ok}$.

Proof If $\Gamma, x : A$`Ok` then by inversion Γ`Ok` Only the type rule `Weaken` adds terms to the environment from its preconditions to its post-condition and it does so in an `Ok` preserving way. Any type derivation tree has at least one leaf. All leaves are axioms which require $\Phi \vdash \Gamma$`Ok`. And all non-axiom derivations preserve the `Ok` property.