1 Grammar

$$
\begin{array}{lll}
\rho ::= x & & \textit{primitives.} \\
\quad | \quad r & & \\
\\
d ::= \texttt{def}\ m(x:\tau):\tau & & \textit{unlabeled decls.} \\
\\
u ::= \rho & & \textit{unlabeled exprs.} \\
\quad | \quad \texttt{new}_d\ x \Rightarrow \overline{d = u} & & \\
\quad | \quad u.m(u) & & \\
\quad | \quad u.\pi & & \\
\\
l ::= \rho & & \textit{labeled exprs.} \\
\quad | \quad \texttt{new}_\sigma\ x \Rightarrow \overline{\sigma = l} & & \\
\quad | \quad l.m(l) & & \\
\quad | \quad l.\pi & & \\
\\
\sigma ::= d\ \texttt{with}\ \varepsilon & & \textit{labeled decls.} \\
\\
e ::= u \mid l & & \textit{exprs.} \\
\\
\tau ::= \{\bar{\sigma}\} & & \textit{types} \\
\quad | \quad \{r\} & & \\
\quad | \quad \{\bar{d}\} & & \\
\quad | \quad \{\bar{d}\ \texttt{captures}\ \varepsilon\} & &
\end{array}
$$

Notes:

– $\hat{e}$ is the form of expressions which are *deeply* labeled; e is the form of unlabeled expressions.

2 Static Semantics

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

$$\frac{}{\Gamma,\ x:\tau \vdash x:\tau}\ \text{(T-Var)} \qquad \frac{}{\Gamma,\ r:\{r\} \vdash r:\{r\}}\ \text{(T-Resource)}$$

$$\frac{\Gamma, x:\{\bar{d}\} \vdash \overline{d = u}\ \texttt{OK}}{\Gamma \vdash\ \texttt{new}_d\ x \Rightarrow \overline{d = u} : \{\bar{d}\}}\ \text{(T-New}_d\text{)} \qquad \frac{\Gamma \vdash u_1 : \{r\}}{\Gamma \vdash u_1.\pi : \texttt{Unit}}\ \text{(T-OperCall)}$$

$$\frac{\Gamma \vdash u_1 : \{\bar{d}\} \quad \texttt{def}\ m(y:\tau_2):\tau_3 \in \{\bar{d}\} \quad \Gamma \vdash u_2 : \tau_2}{\Gamma \vdash u_1.m(u_2) : \tau_3}\ \text{(T-MethCall)}$$

$\boxed{\Gamma \vdash u : \tau\ \texttt{with}\ \varepsilon}$

$$\frac{\varepsilon_c = \texttt{effects}(\Gamma') \quad \Gamma' \subseteq \Gamma \quad \Gamma', x:\{\bar{d}\ \texttt{captures}\ \varepsilon_c\} \vdash \overline{d = u}\ \texttt{OK}}{\Gamma \vdash\ \texttt{new}_d\ x \Rightarrow \overline{d = u} : \{\bar{d}\ \texttt{captures}\ \varepsilon_c\}\ \texttt{with}\ \varnothing}\ \text{(T-NewInf)}$$

$$\frac{\Gamma \vdash u_1 : \{\bar{d}\ \texttt{captures}\ \varepsilon_c\}\ \texttt{with}\ \varepsilon_1 \quad \Gamma \vdash u_2 : \tau_2\ \texttt{with}\ \varepsilon_2 \quad d_i = \ \texttt{def}\ m_i(y:\tau_2):\tau_3}{\Gamma \vdash u_1.m_i(u_2) : \tau_3\ \texttt{with}\ \varepsilon_1 \cup \varepsilon_2 \cup \texttt{effects}(\tau_2) \cup \varepsilon_c}\ \text{(T-MethCallInf)}$$

$\boxed{\Gamma \vdash d = u \texttt{ OK}}$

$$\frac{d = \texttt{def } m(y : \tau_2) : \tau_3 \quad \Gamma, y : \tau_2 \vdash u : \tau_3}{\Gamma \vdash d = u \texttt{ OK}} \text{ (T-ValidImpl)}$$

$\boxed{\Gamma \vdash l : \tau \texttt{ with } \varepsilon}$

$$\frac{}{\Gamma,\ x : \tau \vdash x : \tau \texttt{ with } \varnothing} \text{ (ε-Var)} \qquad \frac{}{\Gamma,\ r : \{r\} \vdash r : \{r\} \texttt{ with } \varnothing} \text{ (ε-Resource)}$$

$$\frac{\Gamma,\ x : \{\bar{\sigma}\} \vdash \overline{\sigma = l} \texttt{ OK}}{\Gamma \vdash \texttt{new}_\sigma\ x \Rightarrow \overline{\sigma = l} : \{\bar{\sigma}\} \texttt{ with } \varnothing} \text{ (ε-NewObj)} \qquad \frac{\Gamma \vdash e_1 : \{r\} \texttt{ with } \varepsilon_1}{\Gamma \vdash l_1.\pi : \texttt{Unit with } \{r.\pi\} \cup \varepsilon_1} \text{ (ε-OperCall)}$$

$$\frac{\Gamma \vdash l_1 : \{\bar{\sigma}\} \texttt{ with } \varepsilon_1 \quad \Gamma \vdash l_2 : \tau_2 \texttt{ with } \varepsilon_2 \quad \sigma_i = \texttt{def } m_i(y : \tau_2) : \tau_3 \texttt{ with } \varepsilon_3}{\Gamma \vdash l_1.m_i(l_2) : \tau_3 \texttt{ with } \varepsilon_1 \cup \varepsilon_2 \cup \varepsilon_3} \text{ (ε-MethCall)}$$

$\boxed{\Gamma \vdash \sigma = e \texttt{ OK}}$

$$\frac{\Gamma,\ y : \tau_2 \vdash l : \tau_3 \texttt{ with } \varepsilon_3 \quad \sigma = \texttt{def } m(y : \tau_2) : \tau_3 \texttt{ with } \varepsilon_3}{\Gamma \vdash \sigma = l \texttt{ OK}} \text{ (ε-ValidImpl)}$$

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

$$\frac{}{\Gamma \vdash \tau <: \tau} \text{ (St-Reflexive)} \qquad \frac{\Gamma \vdash \tau_1 <: \tau_2 \quad \Gamma \vdash \tau_2 <: \tau_3}{\Gamma \vdash \tau_1 <: \tau_3} \text{ (St-Transitive)}$$

$$\frac{\Gamma \vdash e : \tau_1 \quad \Gamma \vdash \tau_1 <: \tau_2}{\Gamma \vdash e : \tau_2} \text{ (St-Subsumption)} \qquad \frac{\Gamma \vdash \tau_1 <: \tau_2 \qquad \varepsilon_1 \subseteq \varepsilon_2}{\Gamma \vdash \tau_1 \texttt{ with } \varepsilon_1 <: \tau_2 \texttt{ with } \varepsilon_2} \text{ (St-EffectTypes)}$$

$$\frac{\Gamma \vdash \{\bar{\sigma}\}_1 \text{ is a permutation of } \{\bar{\sigma}\}_2}{\Gamma \vdash \{\bar{\sigma}\}_1 <: \{\bar{\sigma}\}_2} \text{ (St-Permutation}_\sigma\text{)} \qquad \frac{\Gamma \vdash \{\bar{d}\}_1 \text{ is a permutation of } \{\bar{d}\}_2}{\Gamma \vdash \{\bar{d}\}_1 <: \{\bar{d}\}_2} \text{ (St-Permutation}_d\text{)}$$

$$\frac{\Gamma \vdash \sigma_i <:: \sigma_j}{\Gamma \vdash \{\sigma_i\ ^{i \in 1..n}\} <: \{\sigma_j\ ^{j \in 1..n}\}} \text{ (St-Depth}_\sigma\text{)} \qquad \frac{\Gamma \vdash d_i <:: d_j}{\Gamma \vdash \{d_i\ ^{i \in 1..n}\} <: \{d_j\ ^{j \in 1..n}\}} \text{ (St-Depth}_d\text{)}$$

$$\frac{n, k \geq 0}{\Gamma \vdash \{\sigma_i\ ^{i \in 1..n+k}\} <: \{\sigma_i\ ^{i \in 1..n}\}} \text{ (St-Width}_\sigma\text{)} \qquad \frac{n, k \geq 0}{\Gamma \vdash \{d_i\ ^{i \in 1..n+k}\} <: \{d_i\ ^{i \in 1..n}\}} \text{ (St-Width}_d\text{)}$$

$\boxed{\Gamma \vdash \sigma <:: \sigma}$

$$\frac{\sigma_i = \texttt{def}\ m_A(y : \tau_1) : \tau_2\ \texttt{with}\ \varepsilon_A \qquad \sigma_j = \texttt{def}\ m_B(y : \tau_1') : \tau_2'\ \texttt{with}\ \varepsilon_B \qquad \Gamma \vdash \tau_1' <: \tau_1 \qquad \Gamma \vdash \tau_2 <: \tau_2' \qquad \varepsilon_A \subseteq \varepsilon_B}{\Gamma \vdash \sigma_i <:: \sigma_j}\ (\textsc{St-Method}_\sigma)$$

$\boxed{\Gamma \vdash d <:: d}$

$$\frac{d_i = \texttt{def}\ m_A(y : \tau_1) : \tau_2 \qquad d_j = \texttt{def}\ m_B(y : \tau_1') : \tau_2' \qquad \Gamma \vdash \tau_1' <: \tau_1 \qquad \Gamma \vdash \tau_2 <: \tau_2'}{\Gamma \vdash d_i <:: d_j}\ (\textsc{St-Method}_d)$$

Notes:

- The interesting typing rules are T-NewInf and T-MethCallInf, which perform effect inference on unlabeled programs.
- In T-NewObjInf the object is labeled as capturing the effects in some $\Gamma' \subseteq \Gamma$ (exact definition in next section). We must add the $\texttt{effects}(\tau_2)$ to the static effects of the object, because the method body will have authority over the resources captured by τ_2 (the type of the argument passed into the method).
- A good choice of Γ' for T-NewObjInf is the intersection of Γ with the free variables in the object.
- By convention we use ε_c to denote the output of the `effects` function.

3 Definition: `effects` Function

The `effects` function returns the set of effects captured in a particular context.

- $\texttt{effects}(\varnothing) = \varnothing$
- $\texttt{effects}(\Gamma, x : \tau) = \texttt{effects}(\Gamma) \cup \texttt{effects}(\tau)$
- $\texttt{effects}(\{\bar{r}\}) = \{(r, \pi) \mid r \in \bar{r}, \pi \in \Pi\}$
- $\texttt{effects}(\{\bar{\sigma}\}) = \bigcup_{\sigma \in \bar{\sigma}} \texttt{effects}(\sigma)$
- $\texttt{effects}(\{\bar{d}\}) = \bigcup_{d \in \bar{d}} \texttt{effects}(d)$
- $\texttt{effects}(d\ \texttt{with}\ \varepsilon) = \varepsilon \cup \texttt{effects}(d)$
- $\texttt{effects}(\texttt{def}\ \texttt{m}(x : \tau_1) : \tau_2) = \texttt{effects}(\tau_2)$
- $\texttt{effects}(\{\bar{d}\ \texttt{captures}\ \varepsilon_c\}) = \varepsilon_c$

Notes:

1. The function is monotonic; if $\Gamma_1 \subseteq \Gamma_2$, then $\texttt{effects}(\Gamma_1) \subseteq \texttt{effects}(\Gamma_2)$.

4 Dynamic Semantics

$\boxed{e \longrightarrow e \mid \varepsilon}$

$$\frac{e_1 \longrightarrow e_1' \mid \varepsilon}{e_1.m(e_2) \longrightarrow e_1'.m(e_2) \mid \varepsilon}\ \text{(E-MethCall1)}$$

$$\frac{v_1 = \texttt{new}_\sigma\ x \Rightarrow \overline{\sigma = l} \quad e_2 \longrightarrow e_2' \mid \varepsilon}{v_1.m(e_2) \longrightarrow v_1.m(e_2') \mid \varepsilon}\ \text{(E-MethCall2}_\sigma\text{)} \qquad \frac{v_1 = \texttt{new}_d\ x \Rightarrow \overline{d = u} \quad e_2 \longrightarrow e_2' \mid \varepsilon}{v_1.m(e_2) \longrightarrow v_1.m(e_2') \mid \varepsilon}\ \text{(E-MethCall2}_d\text{)}$$

$$\frac{v_1 = \texttt{new}_\sigma\ x \Rightarrow \overline{\sigma = l} \quad \texttt{def m}(y : \tau_1) : \tau_2\ \texttt{with}\ \varepsilon = l \in \overline{\sigma = l}}{v_1.m(v_2) \longrightarrow [v_1/x, v_2/y]l \mid \varnothing}\ \text{(E-MethCall3}_\sigma\text{)}$$

$$\frac{v_1 = \texttt{new}_d\ x \Rightarrow \overline{d = u} \quad \texttt{def m}(y : \tau_1) : \tau_2 = e \in \overline{d = u}}{v_1.m(v_2) \longrightarrow [v_1/x, v_2/y]u \mid \varnothing}\ \text{(E-MethCall3}_d\text{)}$$

$$\frac{e_1 \longrightarrow e_1' \mid \varepsilon}{e_1.\pi \longrightarrow e_1'.\pi \mid \varepsilon}\ \text{(E-OperCall1)} \qquad \frac{}{r.\pi \longrightarrow \texttt{unit} \mid \{r.\pi\}}\ \text{(E-OperCall2)}$$

$\boxed{e \longrightarrow_* e \mid \varepsilon}$

$$\frac{}{e \longrightarrow_* e \mid \varnothing}\ \text{(E-MultiStep1)} \qquad \frac{e \longrightarrow e' \mid \varepsilon}{e \longrightarrow_* e' \mid \varepsilon}\ \text{(E-MultiStep2)}$$

$$\frac{e \longrightarrow_* e' \mid \varepsilon_1 \quad e' \longrightarrow_* e'' \mid \varepsilon_2}{e \longrightarrow_* e'' \mid \varepsilon_1 \cup \varepsilon_2}\ \text{(E-MultiStep3)}$$

Notes:

- E-MethCall2$_d$ and E-MethCall2$_\sigma$ are really doing the same thing, but one applies to labeled objects (the σ version) and the other to unlabeled objects. Same goes for E-MethCall3$_\sigma$ and E-MethCall3$_d$.
- E-MethCall1 can be used for both labeled and unlabeled objects.

5 Definition (`substitution`)

TODO

6 Lemma (Canonical Forms)

Lemma. If u is a value then $\Gamma \vdash u : \tau\ \texttt{with}\ \varnothing$, and one of the following is true:

1. $u = r$ and $\tau = \texttt{Unit}$.
2. $u = x$ and $x : \tau \in \Gamma$.
3. $u = \texttt{new}_d\ x \Rightarrow \overline{d = u}$ and $\tau = \{\bar{d}\}$

Proof. By inspection.

7 Lemma (Substitution)

Lemma. Suppose the following is true:

1. $\Gamma, z : \tau' \vdash e : \tau$ `with` ε
2. $\Gamma \vdash e' : \tau'$ `with` ε'

Then $\Gamma \vdash [e'/z]e : \tau$ `with` ε.

Proof. TODO (Should be same as the proof in previous grammar, just need to convert everything to new grammar)

8 Definition (`label`)

An unlabeled program may be converted into a labeled program. This is a function from u-terms to l-terms. It is always defined relative to some Γ (the appropriate Γ is usually clear from context). The process is well-defined on u if $\Gamma \vdash u : \tau$. Then `label` is defined below.

1. $\texttt{label}(\rho) = \rho$
2. $\texttt{label}(u_1.\pi) = \texttt{label}(u_1).\pi$
3. $\texttt{label}(u_1.m(u_2)) = \texttt{label}(u_1).m(\texttt{label}(u_2))$
4. $\texttt{label}(\texttt{new}_d\ x \Rightarrow \overline{d = u}) = \texttt{new}_\sigma\ x \Rightarrow \texttt{label-decl}(\overline{d = u})$

The helper function `label-decl` works by labeling each declaration with what it captures in the context Γ. We abbreviate this as $\texttt{effects}(\Gamma \cap \texttt{freevars}(e))$. The helper is defined below.

5. $\texttt{label-decl}(d = u) = d$ `with` $\texttt{effects}(\Gamma \cap \texttt{freevars}(e)) = \texttt{label}(u)$

Notes:

- The image of $\texttt{label}(u)$ is an l-term (proof by induction on definition).
- u is a value $\iff$ $\texttt{label}(u)$ is a value.

9 Theorem (Runtime Invariant Under Labeling)

If the following are true:

1. $\Gamma \vdash e_A : \tau_A$ `with` ε_A
2. $e_A \longrightarrow e_B \mid \varepsilon$
3. $\hat{e}_A = \texttt{label}(e_A, \Gamma)$

Then $\hat{e}_A \longrightarrow \hat{e}_B \mid \varepsilon$ and $\hat{e}_B = \texttt{label}(e_B, \Gamma)$.

10 Theorem (Refinement)

Theorem. Suppose $\Gamma \vdash u : \tau_A$ `with` ε_A. Then $\Gamma \vdash \texttt{label}(u) : \tau_B$ `with` ε_B and:

1. $\tau_B <: \tau_A$
2. $\varepsilon_B \subseteq \varepsilon_A$

Proof. By induction on $\Gamma \vdash u : \tau_A$ `with` ε_A.

Case. T-NewInf.
Then the following are known.

5. $u = \texttt{new}_d\ x \Rightarrow \overline{d = u}$
6. $\tau_A = \{\bar{d}$ `captures` $\varepsilon_C\}$
7. $\varepsilon_A = \varnothing$
8. $\Gamma' \subseteq \Gamma$
9. $\Gamma', x : \{\bar{d}$ `captures` $\varepsilon_C\} \vdash \overline{d = u}$ `OK`
10. $\varepsilon_C = \texttt{effects}(\Gamma')$

Let $l = \texttt{label}(u)$. Applying the definition of label to (5) we get:

$l = \texttt{label}(u)$
$= \texttt{label}(\texttt{new}_d\ x \Rightarrow \overline{d = u})$
$= \texttt{new}_\sigma\ x \Rightarrow \overline{\texttt{label-decl}(d = u)}$
$= \texttt{new}_\sigma\ x \Rightarrow \overline{d\ \texttt{with effects}(\Gamma \cap \texttt{freevars}(u)) = \texttt{label}(u)}$

Fix some method declaration $d_i = u_i$ in the original unlabeled object u. From (9) we know $\Gamma', x : \{\bar{d}\ \texttt{captures}\ \varepsilon_c\} \vdash d_i = u_i\ \texttt{OK}$. The only rule with this judgement is T-ValidImpl. By inversion we obtain:

11. $d_i = \texttt{def}\ m(y : \tau_2) : \tau_3$
12. $\Gamma', x : \{\bar{d}\ \texttt{captures}\ \varepsilon_c\}, y : \tau_2 \vdash e : \tau_3$

Case. T-MethCallInf.
hi

11 Theorem (Soundness)

Theorem. Suppose $\Gamma \vdash u_A : \tau_A\ \texttt{with}\ \varepsilon_A$ and $u_A \longrightarrow u_B \mid \varepsilon$. The following are true:

1. $\Gamma \vdash u_B : \tau_B\ \texttt{with}\ \varepsilon_B$
2. $\tau_B <: \tau_A$
3. $\varepsilon_B \cup \varepsilon = \varepsilon_A$

Proof. Without loss of generality assume u_A is not a value. Let $l_A = \texttt{label}(u_A)$. By Theorem 10 (Refinement) we learn:

4. $\Gamma \vdash l_A : \hat{\tau}_A\ \texttt{with}\ \hat{\varepsilon}_A$
5. $\hat{\tau}_A <: \tau_A$
6. $\hat{\varepsilon}_A \subseteq \varepsilon_A$

Because u_A is not a value, neither is l_A. By Theorem 9 the runtime is invariant under labeling, so $l_A \longrightarrow l_B \mid \varepsilon$, where $l_B = \texttt{label}(u_B)$. Because the typing rules for l-terms are sound, $\Gamma \vdash l_B : \hat{\tau}_B\ \texttt{with}\ \hat{\varepsilon}_B$ where:

7. $\hat{\tau}_B <: \hat{\tau}_A$
8. $\hat{\varepsilon}_B \subseteq \hat{\varepsilon}_A$

It's at this point we invoked Refinement again to relate l_B and u_B. However, the judgement for l_B comes from the soundness theorem. We don't know if this is the same judgement guaranteed by the refinement theorem, so it's not right to equate them.

What we need to know: if $\Gamma \vdash \texttt{label}(u_B) : \hat{\tau}_B\ \texttt{with}\ \hat{\varepsilon}_B$, then for *any* judgement $\Gamma \vdash u_B : \tau_B\ \texttt{with}\ \varepsilon_B$, that $\hat{\tau}_B <: \tau_B$ and $\hat{\varepsilon}_B \subseteq \varepsilon_B$ (in general that's not true though, since you can just type $\hat{\tau}_B$ as $\top$ or some other type which isn't useful).