プログラミング言語周りノート

-

2021 年 2 月 23 日

目次

第 1 章

Preliminaries

1.1　基本的な表記

量化子 (*quantifier*) の束縛をコンマ (,) で続けて書く．束縛の終わりをピリオド (.) で示す．例えば，

$$\forall x_1 \in X_1, x_2 \in X_2. \exists y_1 \in Y_1, y_2 \in Y_2. x_1 = y_1 \wedge x_2 = y_2$$

は，

$$\forall x_1 \in X_1. \forall x_2 \in X_2. \exists y_1 \in Y_1. \exists y_2 \in Y_2. x_1 = y_1 \wedge x_2 = y_2$$

と等しい．また，量化子の束縛において，*such that* を省略し，コンマ (,) で繋げて書く．例えば，

$$\forall x \in \{0,1\}, x \neq 0. x = 1$$

は，

$$\forall x \in \{0,1\}. x \neq 0 \implies x = 1$$

と等しい．また，$\Rightarrow$，$\iff$ が他の記号と混同する場合，それぞれ implies，iff を使用する．

集合 (*set*) について，以下の表記を用いる．

- 集合 A について，その**濃度** (*cardinality*) を $|A|$ と表記する．なお，A が**有限集合** (*finite set*) の時，濃度とは要素の個数のことである．
- 集合 A について，$a \in A$ を $a : A$ と表記する．
- **自然数** (*natural number*) の集合を $\mathbb{N} = \{0, 1, ...\}$ と表記する．
- 自然数 $n \in \mathbb{N}$ について，$\{1, \dots, n\}$ を $[n]$ と表記する．
- 集合 A の**冪集合**を $\mathcal{P}(A) = \{X \mid X \subseteq A\}$，有限冪集合を $\mathcal{P}_{fin}(A) = \{X \in \mathcal{P}(X) \mid X \text{ は有限集合 }\}$ と表記する．
- 集合 $A_1, \dots, A_n$ の**直積** (*cartesian product*) を $A_1 \times \cdots \times A_n = \{(a_1, \dots, a_n) \mid a_1 \in A_1, \dots, a_n \in A_n\}$ と表記する．集合 A の n 直積を $A^n = \underbrace{A \times \cdots \times A}_{n\text{ 項}}$ と表記する．特に，$A^0 = \{\epsilon\}$ である．
- 集合 $A_1, \dots, A_n$ の**直和** (*disjoin union*) を $A_1 \uplus \cdots \uplus A_n = (A_1 \times \{1\}) \cup \cdots (A_n \times \{n\})$ と表記する．なお，文脈から明らかな場合，直和の添字を省略し，$a \in A_i$ に対して，$a \in A_1 \uplus \cdots \uplus A_n$ と表記する．
- 集合 A の B との**差集合**を $A \setminus B = \{a \in A \mid a \notin B\}$ と表記する．

集合 Σ について，$\bigcup_{n \in \mathbb{N}} \Sigma^n$ を Σ^* と表記する．この時，$\alpha \in \Sigma^*$ を Σ による**列** (*sequence*) と呼ぶ．列について，以下の表記を用いる．

- $(\sigma_1, \dots, \sigma_n) \in \Sigma^n$ について，$(\sigma_1, \dots, \sigma_n)$ を $\sigma_1 \cdots \sigma_n$ と表記する．
- 列 $\alpha = \sigma_1 \cdots \sigma_n \in \Sigma^*$ について，その長さを $|\alpha| = n$ と表記する．

集合 A, B について，$R \subseteq A \times B$ を**関係** (*relation*) と呼ぶ．また，

$$A \rightharpoonup B \overset{\text{def}}{=} \{R \in \mathcal{P}(A \times B) \mid \forall x \in A, (x, y_1), (x, y_2) \in R. y_1 = y_2\}$$

という表記を導入し，関係 $f : A \rightharpoonup B$ を A から B への**部分関数** (*partial function*) と呼ぶ．さらに，

$$A \to B \overset{\text{def}}{=} \{f : A \rightharpoonup B \mid \forall x \in A. \exists y \in B. (x, y) \in f\}$$

という表記を導入し，部分関数 $f : A \to B$ を (**全**) **関数** (*function*) と呼ぶ．関係について，以下の表記を用いる．

- 関係 $R \subseteq A \times B$ について，$(a, b) \in R$ を $a \mathrel{R} b$ と表記する．
- 関係 $R \subseteq A \times B$ について，**定義域** (*domain*) を $\mathrm{dom}(R) = \{a \mid \exists b. (a, b) \in R\}$，**値域** (*range*) を $\mathrm{cod}(R) = \{b \mid \exists a. (a, b) \in R\}$ と表記する．
- 部分関数 $f : A \rightharpoonup B$ について，$(a, b) \in f$ を $f(a) = b$ と表記する．

- 関係 $R_1 \subseteq A \times B$，$R_2 \subseteq B \times C$ について，その**合成** (*composition*) を $R_1;R_2 = R_2 \circ R_1 = \{(x,z) \in A \times C \mid \exists y \in B. (x,y) \in R_1, (y,z) \in R_2\}$ と表記する．
- 関係 $R \subseteq A \times B$，集合 $X \subseteq A$ について，R の X による**制限** (*restriction*) を $R\restriction_X = \{(a,b) \in R \mid a \in X\}$ と表記する．特に関数 $f : A \to B$ の $X \subseteq A$ による制限は，関数 $f\restriction_X : X \to B$ になる．
- $a \in A$，$b \in B$ について，その組を $a \longmapsto b = (a,b)$，関数 $f : A \to B$ を $f = x \longmapsto f(x)$ と表記する．
- 2 項関係 $R \subseteq A^2$ について，その**推移閉包** (*transitive closure*)，つまり以下を満たす最小の 2 項関係を $R^+ \subseteq A^2$ と表記する．
  - 任意の $(a,b) \in R$ について，$(a,b) \in R^+$.
  - 任意の $(a,b) \in R^+$，$(b,c) \in R^+$ について，$(a,c) \in R^+$.
- 2 項関係 $R \subseteq A^2$ について，その**反射推移閉包** (*reflexive transitive closure*) を $R^* = R^+ \cup \{(a,a) \mid a \in A\}$ と表記する．

集合 I について，その要素で添字付けられた対象の列 $\{a_i\}_{i \in I}$ を I で添字づけられた**族** (*indexed family*) と呼ぶ．族について，以下の表記を用いる．

- 族の集合を $\prod_{i \in I} A_i = \{\{a_i\}_{i \in I} \mid \forall i \in I, a_i \in A_i\}$ と表記する．
- 集合の族 $A = \{A_i\}_{i \in I}$ について，次の条件を満たす時，A は**互いに素** (*pairwise disjoint*) であるという．

$$\forall i_1, i_2 \in I, i_1 \neq i_2. A_{i_1} \cap A_{i_2} = \emptyset$$

第 2 章

Basic Calculus

2.1 WIP: (Untyped) λ-Calculus

2.2 Simply Typed λ-Calculus

Alias: STLC, $\lambda^{\rightarrow}$ [GTL89]

2.2.1 Syntax

$$
\begin{array}{rll}
e & ::= x & \text{(variable)} \\
 & \mid e\ e & \text{(application)} \\
 & \mid \lambda x : \tau . e & \text{(abstraction)} \\
 & \mid c_A & \text{(constant)} \\
\tau & ::= A & \text{(atomic type)} \\
 & \mid \tau \rightarrow \tau & \text{(function type)} \\
\Gamma & ::= \cdot & \text{(empty)} \\
 & \mid \Gamma, x : \tau & \text{(cons)}
\end{array}
$$

Convention:

$$\tau_1 \rightarrow \tau_2 \rightarrow \cdots \rightarrow \tau_n \stackrel{\text{def}}{=} \tau_1 \rightarrow (\tau_2 \rightarrow (\cdots \rightarrow \tau_n) \cdots)$$
$$e_1\ e_2\ \cdots\ e_n \stackrel{\text{def}}{=} (\cdots (e_1\ e_2) \cdots)\ e_n)$$

Environment Reference:
$\boxed{\Gamma(x) = \tau}$

$$\frac{x = x'}{(\Gamma, x' : \tau)(x) = \tau} \qquad \frac{x \neq x' \quad \Gamma(x) = \tau}{(\Gamma, x' : \tau)(x) = \tau}$$

Free Variable:
$\boxed{fv(e) = \{\overline{x'}\}}$

$$\frac{}{fv(x) = \{x\}} \qquad \frac{fv(e_1) = X_1 \quad fv(e_2) = X_2}{fv(e_1\ e_2) = X_1 \cup X_2} \qquad \frac{fv(e) = X}{fv(\lambda x : \tau . e) = X \setminus \{x\}} \qquad \frac{}{fv(c_A) = \emptyset}$$

Substitution:
部分関数 $\{x_1 \longmapsto e_1, \ldots, x_n \longmapsto e_n\}$ を，$[x_1 \leftarrow e_1, \ldots, x_n \leftarrow e_n]$ または $[x_1, \ldots, x_n \leftarrow e_1, \ldots, e_n]$ と表記する．
$\boxed{e[\overline{x' \leftarrow e'}] = e''}$

$$\frac{[\overline{x'} \leftarrow \overline{e'}](x) = e}{x[\overline{x'} \leftarrow \overline{e'}] = e} \qquad \frac{x \notin \mathrm{dom}([\overline{x'} \leftarrow \overline{e'}])}{x[\overline{x'} \leftarrow \overline{e'}] = x}$$

$$\frac{e_1[\overline{x'} \leftarrow \overline{e'}] = e_1'' \quad e_2[\overline{x'} \leftarrow \overline{e'}] = e_2''}{(e_1\ e_2)[\overline{x'} \leftarrow \overline{e'}] = e_1''\ e_2''} \qquad \frac{e([\overline{x'} \leftarrow \overline{e'}] \upharpoonright_{\mathrm{dom}([\overline{x'} \leftarrow \overline{e'}]) \setminus \{x\}}) = e''}{(\lambda x : \tau . e)[\overline{x'} \leftarrow \overline{e'}] = \lambda x : \tau . e''} \qquad \frac{}{c_A[\overline{x'} \leftarrow \overline{e'}] = c_A}$$

α-Equality:
$\boxed{e_1 \equiv_\alpha e_2}$

$$\frac{x_1 = x_2}{x_1 \equiv_\alpha x_2} \qquad \frac{x' \notin fv(e_1) \cup fv(e_2) \quad e_1[x_1 \leftarrow x'] \equiv_\alpha e_2[x_2 \leftarrow x']}{\lambda x_1 : \tau . e_1 \equiv_\alpha \lambda x_2 : \tau . e_2} \qquad \frac{e_1 \equiv_\alpha e_2 \quad e_1' \equiv_\alpha e_2'}{e_1\ e_1' \equiv_\alpha e_2\ e_2'} \qquad \frac{}{c_A \equiv_\alpha c_A}$$

定理 1 (Correctness of Substitution). 式 e，置換 $[\overline{x'} \leftarrow \overline{e'}]$ について，$X = \mathrm{dom}([\overline{x'} \leftarrow \overline{e'}])$ とした時，

$$fv(e[\overline{x'} \leftarrow \overline{e'}]) = (fv(e) \setminus X) \cup \bigcup_{x \in fv(e) \cap X} fv([\overline{x'} \leftarrow \overline{e'}](x)).$$

□

定理 2 (α-Equality Does Not Touch Free Variables). $e_1 \equiv_\alpha e_2$ ならば，$fv(e_1) = fv(e_2)$. □

2.2.2 Typing Semantics

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

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau}\ \text{T-Var}$$

$$\frac{\Gamma, x : \tau_1 \vdash e : \tau_2}{\Gamma \vdash \lambda x : \tau_1 . e : \tau_1 \to \tau_2}\ \text{T-Abs}$$

$$\frac{\Gamma \vdash e_1 : \tau_2 \to \tau \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1\ e_2 : \tau}\ \text{T-App}$$

$$\frac{}{\Gamma \vdash c_A : A}\ \text{T-Const}$$

特に，$\cdot \vdash e : \tau$ の時，$e : \tau$ と表記.

2.2.3 Evaluation Semantics (Call-By-Value)

$$\begin{aligned} v &::= \lambda x : \tau . e \\ &\mid c_A \\ C &::= [] \\ &\mid C\ e \\ &\mid v\ C \end{aligned}$$

Small Step:

$\boxed{e \Rightarrow e'}$

$$\frac{}{(\lambda x : \tau . e)\ v \Rightarrow e[x \leftarrow v]}$$

$$\frac{e \Rightarrow e'}{C[e] \Rightarrow C[e']}$$

Big Step:

$\boxed{e \Downarrow v}$

$$\frac{e_1 \Downarrow \lambda x : \tau . e_1' \quad e_2 \Downarrow v_2 \quad e_1'[x \leftarrow v_2] \Downarrow v}{e_1\ e_2 \Downarrow v}$$

定理 3 (Adequacy of Small Step and Big Step). $e \Rightarrow^* v$ iff $e \Downarrow v$. □

定理 4 (Type Soundness). $e : \tau$ の時，$e \Rightarrow^* v$，$e \Downarrow v$ となる $v = \mathrm{nf}(\Rightarrow, e)$ が存在し，

- $\tau = \tau_1 \to \tau_2$ の時，$v \equiv_\alpha \lambda x' : \tau_1 . e'$ となる $\lambda x' : \tau' . e'$ が存在する.
- $\tau = A$ の時，$v \equiv_\alpha c_A$ となる c_A が存在する.

□

2.2.4 Equational Reasoning

$\boxed{\Gamma \vdash e_1 \equiv e_2 : \tau}$

$$\frac{\Gamma, x : \tau \vdash e_1 : \tau_2 \to \tau \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (\lambda x : \tau. e_1)\, e_2 \equiv e_1[x \leftarrow e_2] : \tau}\ \text{Eq-}\beta\text{-Lam} \qquad \frac{x \notin fv(e) \quad \Gamma \vdash e : \tau_1 \to \tau_2}{\Gamma \vdash (\lambda x : \tau_1. e\ x) \equiv e : \tau_1 \to \tau_2}\ \text{Eq-}\eta\text{-Lam}$$

$$\frac{e_1 \equiv_\alpha e_2 \quad \Gamma \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1 \equiv e_2 : \tau}\ \text{Eq-}\alpha\text{-Refl}$$

$$\frac{\Gamma \vdash e_2 \equiv e_1 : \tau}{\Gamma \vdash e_1 \equiv e_2 : \tau}\ \text{Eq-Sym} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau \quad \Gamma \vdash e_2 \equiv e_3 : \tau}{\Gamma \vdash e_1 \equiv e_3 : \tau}\ \text{Eq-Trans}$$

$$\frac{\Gamma, x : \tau \vdash e_1 \equiv e_2 : \tau'}{\Gamma \vdash \lambda x : \tau. e_1 \equiv \lambda x : \tau. e_2 : \tau \to \tau'}\ \text{Eq-Cong-Abs} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau' \to \tau \quad \Gamma \vdash e_1' \equiv e_2' : \tau'}{\Gamma \vdash e_1\, e_1' \equiv e_2\, e_2' : \tau}\ \text{Eq-Cong-App}$$

特に，$\cdot \vdash e_1 \equiv e_2 : \tau$ の時，$e_1 \equiv e_2 : \tau$ と表記.

定理 5 (Respect Typing). $\Gamma \vdash e_1 \equiv e_2 : \tau$ ならば，$\Gamma \vdash e_1 : \tau$ かつ $\Gamma \vdash e_2 : \tau$. □

定理 6 (Respect Evaluation). $e_1 \equiv e_2 : \tau$ の時，$e_1' \Rightarrow^* e_1$，$e_2 \Rightarrow^* e_2'$ ならば $e_1' \equiv e_2' : \tau$. □

系 7. $e_1 \equiv e_2 : \tau$ の時，$e_1 \Rightarrow^* e_1'$，$e_2 \Rightarrow^* e_2'$ ならば $e_1' \equiv e_2' : \tau$. □

証明. $e_1 \Rightarrow^* e_1$ より，定理 6 から $e_1 \equiv e_2' : \tau$. よって，T-Sym から $e_2' \equiv e_1 : \tau$ であり，$e_2' \Rightarrow^* e_2'$ より定理 6 から $e_2' \equiv e_1' : \tau$. 故に，T-Sym から $e_1' \equiv e_2' : \tau$. ■

2.3 WIP: System-T

2.4 WIP: PCF

2.5 System-F

Alias: F，Second Order Typed Lambda Calculus，$\lambda 2$ [GTL89]

2.5.1 Syntax

$$
\begin{aligned}
e &::= x && \text{(variable)}\\
&\mid \lambda x : \tau . e && \text{(abstraction)}\\
&\mid e\ e && \text{(application)}\\
&\mid \Lambda t . e && \text{(universal abstraction)}\\
&\mid e\ \tau && \text{(universal application)}\\
\tau &::= t && \text{(type variable)}\\
&\mid \tau \to \tau && \text{(function type)}\\
&\mid \forall t . \tau && \text{(polymorphic type)}\\
\Gamma &::= \cdot && \text{(empty)}\\
&\mid \Gamma, x : \tau && \text{(variable cons)}\\
&\mid \Gamma, t : \Omega && \text{(type variable cons)}
\end{aligned}
$$

Convention:

$$
\tau_1 \to \tau_2 \to \cdots \to \tau_n \overset{\text{def}}{=} \tau_1 \to (\tau_2 \to (\cdots \to \tau_n)\cdots)
$$
$$
e_1\ e_2\ \cdots\ e_n \overset{\text{def}}{=} (\cdots(e_1\ e_2)\cdots)\ e_n)
$$

Environment Reference:
$\boxed{\Gamma(x) = \tau}$

$$
\frac{x = x'}{(\Gamma, x' : \tau)(x) = \tau} \qquad \frac{x \neq x' \quad \Gamma(x) = \tau}{(\Gamma, x' : \tau)(x) = \tau} \qquad \frac{\Gamma(x) = \tau}{(\Gamma, t : \Omega)(x) = \tau}
$$
$$
\frac{t = t'}{(\Gamma, t' : \Omega)(t) = \Omega} \qquad \frac{t \neq t' \quad \Gamma(t) = \Omega}{(\Gamma, t' : \Omega)(t) = \Omega} \qquad \frac{\Gamma(t) = \Omega}{(\Gamma, x : \tau)(t) = \Omega}
$$

Free Variable:
$\boxed{fv(e) = \{\overline{x}\}}$

$$
\frac{}{fv(x) = \{x\}} \qquad \frac{fv(e_1) = X_1 \quad fv(e_2) = X_2}{fv(e_1\ e_2) = X_1 \cup X_2} \qquad \frac{fv(e) = X}{fv(\lambda x : \tau . e) = X \setminus \{x\}} \qquad \frac{fv(e) = X}{fv(e\ \tau) = X} \qquad \frac{fv(e) = X}{fv(\Lambda t . e) = X}
$$

Substitution:
部分関数 $\{x_1 \longmapsto e_1, \dots, x_n \longmapsto e_n\}$ を，$[x_1 \leftarrow e_1, \dots, x_n \leftarrow e_n]$ または $[x_1, \dots, x_n \leftarrow e_1, \dots, e_n]$ と表記する．
$\boxed{e[\overline{x'} \leftarrow \overline{e'}] = e''}$

$$
\frac{[\overline{x'} \leftarrow \overline{e'}](x) = e}{x[\overline{x'} \leftarrow \overline{e'}] = e} \qquad \frac{x \notin \mathrm{dom}([\overline{x'} \leftarrow \overline{e'}])}{x[\overline{x'} \leftarrow \overline{e'}] = x}
$$
$$
\frac{e_1[\overline{x'} \leftarrow \overline{e'}] = e_1'' \quad e_2[\overline{x'} \leftarrow \overline{e'}] = e_2''}{(e_1\ e_2)[\overline{x'} \leftarrow \overline{e'}] = e_1''\ e_2''} \qquad \frac{e([\overline{x'} \leftarrow \overline{e'}] \restriction_{\mathrm{dom}([\overline{x'} \leftarrow \overline{e'}]) \setminus \{x\}}) = e''}{(\lambda x : \tau . e)[\overline{x'} \leftarrow \overline{e'}] = \lambda x : \tau . e''}
$$

$$\frac{e[\overline{x'} \leftarrow \overline{e'}] = e''}{(e\ \tau)[\overline{x'} \leftarrow \overline{e'}] = e''\ \tau} \qquad \frac{e[\overline{x'} \leftarrow \overline{e'}] = e''}{(\Lambda t.\, e)[\overline{x'} \leftarrow \overline{e'}] = \Lambda t.\, e''}$$

Type Free Variable:

$\boxed{tyfv(e) = \{\overline{x}\}}$

$$\frac{}{tyfv(x) = \emptyset} \qquad \frac{tyfv(e_1) = T_1 \quad tyfv(e_2) = T_2}{tyfv(e_1\ e_2) = T_1 \cup T_2} \qquad \frac{tyfv(\tau) = T_1 \quad tyfv(e) = T_2}{tyfv(\lambda x : \tau.\, e) = T_1 \cup T_2}$$

$$\frac{tyfv(e) = T_1 \quad tyfv(\tau) = T_2}{tyfv(e\ \tau) = T_1 \cup T_2} \qquad \frac{tyfv(e) = T}{tyfv(\Lambda t.\, e) = T \setminus \{t\}}$$

$$\frac{}{tyfv(t) = \{t\}} \qquad \frac{tyfv(\tau_1) = T_1 \quad tyfv(\tau_2) = T_2}{tyfv(\tau_1 \to \tau_2) = T_1 \cup T_2} \qquad \frac{tyfv(\tau) = T}{tyfv(\forall t.\, \tau) = T \setminus \{t\}}$$

Type Substitution:

部分関数 $\{t_1 \longmapsto \tau_1, \dots, t_n \longmapsto \tau_n\}$ を，$[t_1 \leftarrow \tau_1, \dots, t_n \leftarrow \tau_n]$ または $[t_1, \dots, t_n \leftarrow t_1, \dots, t_n]$ と表記する．

$\boxed{e[\overline{t \leftarrow \tau}] = e'}$

$$\frac{}{x[\overline{t'} \leftarrow \overline{\tau'}] = x} \qquad \frac{e_1[\overline{t'} \leftarrow \overline{\tau'}] = e_1'' \quad e_2[\overline{t'} \leftarrow \overline{\tau'}] = e_2''}{(e_1\ e_2)[\overline{t'} \leftarrow \overline{\tau'}] = e_1''\ e_2''} \qquad \frac{\tau[\overline{t'} \leftarrow \overline{\tau'}] = \tau'' \quad e[\overline{t'} \leftarrow \overline{\tau'}] = e''}{(\lambda x : \tau.\, e)[\overline{t'} \leftarrow \overline{\tau'}] = \lambda x : \tau''.\, e''}$$

$$\frac{e[\overline{t'} \leftarrow \overline{\tau'}] = e'' \quad \tau[\overline{t'} \leftarrow \overline{\tau'}] = \tau''}{(e\ \tau)[\overline{t'} \leftarrow \overline{\tau'}] = e''\ \tau''} \qquad \frac{e([\overline{t'} \leftarrow \overline{\tau'}] \restriction_{\mathrm{dom}([\overline{t'} \leftarrow \overline{\tau'}]) \setminus \{t\}}) = e''}{(\Lambda t.\, e)[\overline{t'} \leftarrow \overline{\tau'}] = \Lambda t.\, e''}$$

$\boxed{\tau[\overline{t' \leftarrow \tau'}] = \tau''}$

$$\frac{[\overline{t'} \leftarrow \overline{\tau'}](t) = \tau}{t[\overline{t'} \leftarrow \overline{\tau'}] = \tau} \qquad \frac{t \notin \mathrm{dom}([\overline{t'} \leftarrow \overline{\tau'}])}{t[\overline{t'} \leftarrow \overline{\tau'}] = t} \qquad \frac{\tau_1[\overline{t'} \leftarrow \overline{\tau'}] = \tau_1'' \quad \tau_2[\overline{t'} \leftarrow \overline{\tau'}] = \tau_2''}{(\tau_1 \to \tau_2)[\overline{t'} \leftarrow \overline{\tau'}] = \tau_1'' \to \tau_2''} \qquad \frac{\tau([\overline{t'} \leftarrow \overline{\tau'}] \restriction_{\mathrm{dom}([\overline{t'} \leftarrow \overline{\tau'}]) \setminus \{t\}}) = \tau''}{(\forall t.\, \tau)[\overline{t'} \leftarrow \overline{\tau'}] = \forall t.\, \tau''}$$

α-Equality:

$\boxed{e_1 \equiv_\alpha e_2}$

$$\frac{x_1 = x_2}{x_1 \equiv_\alpha x_2} \qquad \frac{e_1 \equiv_\alpha e_2 \quad e_1' \equiv_\alpha e_2'}{e_1\ e_1' \equiv_\alpha e_2\ e_2'} \qquad \frac{\tau_1 \equiv_\alpha \tau_2 \quad x' \notin fv(e_1) \cup fv(e_2) \quad e_1[x_1 \leftarrow x'] \equiv_\alpha e_2[x_2 \leftarrow x']}{\lambda x_1 : \tau_1.\, e_1 \equiv_\alpha \lambda x_2 : \tau_2.\, e_2}$$

$$\frac{e_1 \equiv_\alpha e_2 \quad \tau_1 \equiv_\alpha \tau_2}{e_1\ \tau_1 \equiv_\alpha e_2\ \tau_2} \qquad \frac{t' \notin tyfv(e_1) \cup tyfv(e_2) \quad e_1[t_1 \leftarrow t'] \equiv_\alpha e_2[t_2 \leftarrow t']}{\Lambda t_1.\, e_1 \equiv_\alpha \Lambda t_2.\, e_2}$$

$\boxed{\tau_1 \equiv_\alpha \tau_2}$

$$\frac{t_1 = t_2}{t_1 \equiv_\alpha t_2} \qquad \frac{\tau_1 \equiv_\alpha \tau_2 \quad \tau_1' \equiv_\alpha \tau_2'}{\tau_1 \to \tau_1' \equiv_\alpha \tau_2 \to \tau_2'} \qquad \frac{t' \notin tyfv(\tau_1) \cup tyfv(\tau_2) \quad \tau_1[t_1 \leftarrow t'] \equiv_\alpha \tau_2[t_2 \leftarrow t']}{\forall t_1.\, \tau_1 \equiv_\alpha \forall t_2.\, \tau_2}$$

定理 8 (Correctness of Substitution). 置換 $[\overline{x'} \leftarrow \overline{e'}]$ について，$X = \mathrm{dom}([\overline{x'} \leftarrow \overline{e'}])$ とした時,

$$fv(e[\overline{x'} \leftarrow \overline{e'}]) = (fv(e) \setminus X) \cup \bigcup_{x \in fv(e) \cap X} fv([\overline{x'} \leftarrow \overline{e'}](x)).$$

□

定理 9 (Correctness of Type Substitution). 式 e，型 τ，型置換 $[\overline{t'} \leftarrow \overline{\tau'}]$ について，$T = \mathrm{dom}([\overline{t'} \leftarrow \overline{\tau'}])$ とした時,

$$tyfv(e[\overline{t'} \leftarrow \overline{\tau'}]) = (tyfv(e) \setminus T) \cup \bigcup_{t \in tyfv(e) \cap T} tyfv([\overline{t'} \leftarrow \overline{\tau'}](t))$$

$$tyfv(\tau[\overline{t'} \leftarrow \overline{\tau'}]) = (tyfv(\tau) \setminus T) \cup \bigcup_{t \in tyfv(\tau) \cap T} tyfv([\overline{t'} \leftarrow \overline{\tau'}](t)).$$

□

定理 10 (α-Equality Does Not Touch Free Variables).

- $\tau_1 \equiv_\alpha \tau_2$ ならば $tyfv(\tau_1) = tyfv(\tau_2)$.
- $e_1 \equiv_\alpha e_2$ ならば，$fv(e_1) = fv(e_2)$，$tyfv(e_1) = tyfv(e_2)$.

□

2.5.2　Typing Semantics

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

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau}\ \text{T-Var}$$

$$\frac{\Gamma, x : \tau_1 \vdash e : \tau_2}{\Gamma \vdash \lambda x : \tau_1 . e : \tau_1 \to \tau_2}\ \text{T-Abs}$$

$$\frac{\Gamma \vdash e_1 : \tau_2 \to \tau \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1\ e_2 : \tau}\ \text{T-App}$$

$$\frac{\Gamma, t : \Omega \vdash e : \tau}{\Gamma \vdash \Lambda t . e : \forall t . \tau}\ \text{T-UnivAbs}$$

$$\frac{\Gamma \vdash e : \forall t . \tau_1}{\Gamma \vdash e\ \tau_2 : \tau_1[t \leftarrow \tau_2]}\ \text{T-UnivApp}$$

$$\frac{\Gamma \vdash \tau \equiv_\alpha \tau' : \Omega \quad \Gamma \vdash e : \tau'}{\Gamma \vdash e : \tau}\ \text{T-}\alpha\text{-Equiv}$$

特に，$\cdot \vdash e : \tau$ の時，$e : \tau$ と表記.

2.5.3　Evaluation Semantics (Call-By-Value)

$$\begin{aligned} v &::= \lambda x : \tau . e \\ &\mid \Lambda t . e \\ C &::= [] \\ &\mid C\ e \\ &\mid v\ C \\ &\mid C\ \tau \end{aligned}$$

Small Step:
$\boxed{e \Rightarrow e'}$

$$\frac{}{(\lambda x : \tau . e)\ v \Rightarrow e[x \leftarrow v]}$$

$$\frac{}{(\Lambda t . e)\ \tau \Rightarrow e[t \leftarrow \tau]}$$

$$\frac{e \Rightarrow e'}{C[e] \Rightarrow C[e']}$$

Big Step:
$\boxed{e \Downarrow v}$

$$\frac{e_1 \Downarrow \lambda x : \tau. e_1' \quad e_2 \Downarrow v_2 \quad e_1'[x \leftarrow v_2] \Downarrow v}{e_1\ e_2 \Downarrow v}$$

$$\frac{e \Downarrow \Lambda t. e_1' \quad e_1'[t \leftarrow \tau] \Downarrow v}{e\ \tau \Downarrow v}$$

定理 11 (Adequacy of Small Step and Big Step). $e \Rightarrow^* v$ iff $e \Downarrow v$. □

定理 12 (Type Soundness). $e : \tau$ の時，$e \Rightarrow^* v$，$e \Downarrow v$ となる $v = \mathrm{nf}(\Rightarrow, e)$ が存在し，

- $\tau = \tau_1 \to \tau_2$ の時，$v \equiv_\alpha \lambda x' : \tau_1. e'$ となる $\lambda x' : \tau_1. e'$ が存在する．
- $\tau = \forall t. \tau_1$ の時，$v \equiv_\alpha \Lambda t. e'$ となる $\Lambda t. e'$ が存在する．

□

2.5.4 Equational Reasoning

$\boxed{\Gamma \vdash e_1 \equiv e_2 : \tau}$

$$\frac{\Gamma, x : \tau_2 \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (\lambda x : \tau_2. e_1)\ e_2 \equiv e_1[x \leftarrow e_2] : \tau}\ \text{Eq-}\beta\text{-Lam} \qquad \frac{x \notin fv(e) \quad \Gamma \vdash e : \tau_1 \to \tau_2}{\Gamma \vdash (\lambda x : \tau_1. e\ x) \equiv e : \tau_1 \to \tau_2}\ \text{Eq-}\eta\text{-Lam}$$

$$\frac{\Gamma, t : \Omega \vdash e : \tau}{\Gamma \vdash (\Lambda t. e)\ \tau_2 \equiv e[t \leftarrow \tau_2] : \tau[t \leftarrow \tau_2]}\ \text{Eq-}\beta\text{-UnivLam} \qquad \frac{t \notin tyfv(e) \quad \Gamma \vdash e : \forall t'. \tau}{\Gamma \vdash (\Lambda t. e\ t) \equiv e : \forall t'. \tau}\ \text{Eq-}\eta\text{-UnivLam}$$

$$\frac{e_1 \equiv_\alpha e_2 \quad \Gamma \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1 \equiv e_2 : \tau}\ \text{Eq-}\alpha\text{-Refl} \qquad \frac{\tau \equiv_\alpha \tau' \quad \Gamma \vdash e_1 \equiv e_2 : \tau'}{\Gamma \vdash e_1 \equiv e_2 : \tau}\ \text{Eq-}\alpha\text{-Type}$$

$$\frac{\Gamma \vdash e_2 \equiv e_1 : \tau}{\Gamma \vdash e_1 \equiv e_2 : \tau}\ \text{Eq-Sym} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau \quad \Gamma \vdash e_2 \equiv e_3 : \tau}{\Gamma \vdash e_1 \equiv e_3 : \tau}\ \text{Eq-Trans}$$

$$\frac{\Gamma, x : \tau \vdash e_1 \equiv e_2 : \tau'}{\Gamma \vdash \lambda x : \tau. e_1 \equiv \lambda x : \tau. e_2 : \tau \to \tau'}\ \text{Eq-Cong-Abs} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau' \to \tau \quad \Gamma \vdash e_1' \equiv e_2' : \tau'}{\Gamma \vdash e_1\ e_1' \equiv e_2\ e_2' : \tau}\ \text{Eq-Cong-App}$$

$$\frac{\Gamma, t : \Omega \vdash e_1 \equiv e_2 : \tau}{\Gamma \vdash \Lambda t. e_1 \equiv \Lambda t. e_2 : \forall (t. \tau)}\ \text{Eq-Cong-UnivAbs} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \forall t. \tau}{\Gamma \vdash e_1\ \tau' \equiv e_2\ \tau' : \tau[t \leftarrow \tau']}\ \text{Eq-Cong-UnivApp}$$

特に，$\cdot \vdash e_1 \equiv e_2 : \tau$ の時，$e_1 \equiv e_2 : \tau$ と表記．

定理 13 (Respect Typing). $\Gamma \vdash e_1 \equiv e_2 : \tau$ ならば，$\Gamma \vdash e_1 : \tau$ かつ $\Gamma \vdash e_2 : \tau$. □

定理 14 (Respect Evaluation). $e_1 \equiv e_2 : \tau$ の時，$e_1' \Rightarrow^* e_1$，$e_2 \Rightarrow^* e_2'$ ならば $e_1' \equiv e_2' : \tau$. □

系 15. $e_1 \equiv e_2 : \tau$ の時，$e_1 \Rightarrow^* e_1'$，$e_2 \Rightarrow^* e_2'$ ならば $e_1' \equiv e_2' : \tau$. □

証明. $e_1 \Rightarrow^* e_1$ より，定理 14 から $e_1 \equiv e_2' : \tau$. よって，T-Sym から $e_2' \equiv e_1 : \tau$ であり，$e_2' \Rightarrow^* e_2'$ より定理 14 から $e_2' \equiv e_1' : \tau$. 故に，T-Sym から $e_1' \equiv e_2' : \tau$. ■

2.5.5 Definability

Product

Product of τ_1 and τ_2:

$$\tau_1 \times \tau_2 \overset{\text{def}}{=} \forall t. (\tau_1 \to \tau_2 \to t) \to t$$

$$\langle e_1, e_2 \rangle \overset{\text{def}}{=} \Lambda t. \lambda x : \tau_1 \to \tau_2 \to t. x\ e_1\ e_2$$

$$\pi_1 e \overset{\text{def}}{=} e\ \tau_1\ \lambda x_1.\lambda x_2.\,x_1$$
$$\pi_2 e \overset{\text{def}}{=} e\ \tau_2\ \lambda x_1.\lambda x_2.\,x_2$$

Admissible typing rule:

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

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \langle e_1, e_2 \rangle : \tau_1 \times \tau_2}\ \text{T-Product} \qquad \frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \pi_1 e : \tau_1}\ \text{T-Proj-1} \qquad \frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \pi_2 e : \tau_2}\ \text{T-Proj-2}$$

Admissible equality:

$\boxed{\Gamma \vdash e_1 \equiv e_2 : \tau}$

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \pi_1\langle e_1, e_2 \rangle \equiv e_1 : \tau_1}\ \text{Eq-}\beta\text{-Product-1} \qquad \frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \pi_2\langle e_1, e_2 \rangle \equiv e_2 : \tau_2}\ \text{Eq-}\beta\text{-Product-2}$$
$$\frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \langle \pi_1 e, \pi_2 e \rangle \equiv e : \tau_1 \times \tau_2}\ \text{Eq-}\eta\text{-Product}$$

Existential Type

Existence of $\exists t.\,\tau$:

$$\exists t.\,\tau \overset{\text{def}}{=} \forall t'.\,(\forall t.\,\tau \to t') \to t'$$
$$\mathsf{pack}\langle \tau_t, e \rangle \overset{\text{def}}{=} \Lambda t'.\,\lambda x : (\forall t.\,\tau \to t').\,x\ \tau_t\ e$$
$$\mathsf{unpack}\langle t, x \rangle = e_1.\,\tau_2.\,e_2 \overset{\text{def}}{=} e_1\ \tau_2\ (\Lambda t.\,\lambda x : \tau.\,e_2)$$

Admissible typing rule:

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

$$\frac{\Gamma \vdash e : \tau[t \leftarrow \tau_t]}{\Gamma \vdash \mathsf{pack}\langle \tau_t, e \rangle : \exists t.\,\tau}\ \text{T-Pack} \qquad \frac{\Gamma \vdash e_1 : \exists t.\,\tau \quad \Gamma, t : \Omega, x : \tau \vdash e_2 : \tau_2 \quad t \notin tyfv(\tau_2)}{\Gamma \vdash \mathsf{unpack}\langle t, x \rangle = e_1.\,\tau_2.\,e_2 : \tau_2}\ \text{T-Unpack}$$

Admissible equality:

$\boxed{\Gamma \vdash e_1 \equiv e_2 : \tau}$

$$\frac{\Gamma \vdash e_1 : \tau_1[t \leftarrow \tau_t] \quad \Gamma, t : \Omega, x : \tau_1 \vdash e_2 : \tau_2 \quad t \notin tyfv(\tau_2)}{\Gamma \vdash \mathsf{unpack}\langle t, x \rangle = \mathsf{pack}\langle \tau_t, e_1 \rangle.\,\tau_2.\,e_2 \equiv e_2[t \leftarrow \tau_t][x \leftarrow e_1] : \tau_2}\ \text{Eq-}\beta\text{-Exist}$$
$$\frac{\Gamma \vdash e : \exists t'.\,\tau \quad \tau' \equiv_\alpha \exists t'.\,\tau}{\Gamma \vdash \mathsf{unpack}\langle t, x \rangle = e.\,\tau'.\,\mathsf{pack}\langle t, x \rangle \equiv e : \exists t'.\,\tau}\ \text{Eq-}\eta\text{-Exist}$$

2.6 System-F **ω**

Alias: F ω，$\lambda\omega$ [RRD14]

2.6.1 Syntax

$$
\begin{array}{llr}
e & ::= x & \text{(variable)} \\
 & \mid \lambda x : \tau . e & \text{(abstraction)} \\
 & \mid e\ e & \text{(application)} \\
 & \mid \Lambda t : \kappa . e & \text{(universal abstraction)} \\
 & \mid e\ \tau & \text{(universal application)} \\
\tau & ::= t & \text{(type variable)} \\
 & \mid \tau \rightarrow \tau & \text{(function type)} \\
 & \mid \forall t : \kappa . \tau & \text{(polymorphic type)} \\
 & \mid \lambda t : \kappa . \tau & \text{(type abstraction)} \\
 & \mid \tau\ \tau & \text{(type application)} \\
\kappa & ::= \Omega & \text{(type kind)} \\
 & \mid \kappa \rightarrow \kappa & \text{(arrow kind)} \\
\Gamma & ::= \cdot & \text{(empty)} \\
 & \mid \Gamma, x : \tau & \text{(variable cons)} \\
 & \mid \Gamma, t : \kappa & \text{(type variable cons)}
\end{array}
$$

Convention:

$$\tau_1 \rightarrow \tau_2 \rightarrow \cdots \rightarrow \tau_n \stackrel{\text{def}}{=} \tau_1 \rightarrow (\tau_2 \rightarrow (\cdots \rightarrow \tau_n) \cdots)$$
$$e_1\ e_2\ \cdots\ e_n \stackrel{\text{def}}{=} (\cdots (e_1\ e_2) \cdots)\ e_n)$$

Environment Reference:
$\boxed{\Gamma(x) = \tau}$

$$\frac{x = x'}{(\Gamma, x' : \tau)(x) = \tau} \qquad \frac{x \neq x' \quad \Gamma(x) = \tau}{(\Gamma, x' : \tau)(x) = \tau} \qquad \frac{\Gamma(x) = \tau}{(\Gamma, t : \kappa)(x) = \tau}$$
$$\frac{t = t'}{(\Gamma, t' : \kappa)(t) = \kappa} \qquad \frac{t \neq t' \quad \Gamma(t) = \kappa}{(\Gamma, t' : \kappa')(t) = \kappa} \qquad \frac{\Gamma(t) = \kappa}{(\Gamma, x : \tau)(t) = \kappa}$$

Free Variable:
$\boxed{fv(e) = \{\overline{x}\}}$

$$\frac{}{fv(x) = \{x\}} \qquad \frac{fv(e) = X}{fv(\lambda x : \tau . e) = X \setminus \{x\}} \qquad \frac{fv(e_1) = X_1 \quad fv(e_2) = X_2}{fv(e_1\ e_2) = X_1 \cup X_2} \qquad \frac{fv(e) = X}{fv(\Lambda t : \kappa . e) = X} \qquad \frac{fv(e) = X}{fv(e\ \tau) = X}$$

Substitution:
部分関数 $\{x_1 \longmapsto e_1, \ldots, x_n \longmapsto e_n\}$ を，$[x_1 \leftarrow e_1, \ldots, x_n \leftarrow e_n]$ または $[x_1, \ldots, x_n \leftarrow e_1, \ldots, e_n]$ と表記する．
$\boxed{e[\overline{x' \leftarrow e'}] = e''}$

$$\frac{[\overline{x'} \leftarrow \overline{e'}](x) = e}{x[\overline{x'} \leftarrow \overline{e'}] = e} \qquad \frac{x \notin \mathrm{dom}([\overline{x'} \leftarrow \overline{e'}])}{x[\overline{x'} \leftarrow \overline{e'}] = x}$$

$$\frac{e([\overline{x'} \leftarrow \overline{e'}] \restriction_{\mathrm{dom}([\overline{x'}\leftarrow\overline{e'}])\setminus\{x\}}) = e''}{(\lambda x : \tau.\, e)[\overline{x'} \leftarrow \overline{e'}] = \lambda x : \tau.\, e''} \quad \frac{e_1[\overline{x'} \leftarrow \overline{e'}] = e_1'' \quad e_2[\overline{x'} \leftarrow \overline{e'}] = e_2''}{(e_1\ e_2)[\overline{x'} \leftarrow \overline{e'}] = e_1''\ e_2''}$$

$$\frac{e[\overline{x'} \leftarrow \overline{e'}] = e''}{(\Lambda t : \kappa.\, e)[\overline{x'} \leftarrow \overline{e'}] = \Lambda t : \kappa.\, e''} \quad \frac{e[\overline{x'} \leftarrow \overline{e'}] = e''}{(e\ \tau)[\overline{x'} \leftarrow \overline{e'}] = e''\ \tau}$$

Type Free Variable:

$\boxed{tyfv(e) = \{\overline{t}\}}$

$$\frac{}{tyfv(x) = \emptyset} \quad \frac{tyfv(\tau) = T_1 \quad tyfv(e) = T_2}{tyfv(\lambda x : \tau.\, e) = T_1 \cup T_2} \quad \frac{tyfv(e_1) = T_1 \quad tyfv(e_2) = T_2}{tyfv(e_1\ e_2) = T_1 \cup T_2}$$

$$\frac{tyfv(e) = T}{tyfv(\Lambda t : \kappa.\, e) = T \setminus \{t\}} \quad \frac{tyfv(e) = T_1 \quad tyfv(\tau) = T_2}{tyfv(e\ \tau) = T_1 \cup T_2}$$

$\boxed{tyfv(\tau) = \{\overline{t}\}}$

$$\frac{}{tyfv(t) = \{t\}} \quad \frac{tyfv(\tau_1) = T_1 \quad tyfv(\tau_2) = T_2}{tyfv(\tau_1 \to \tau_2) = T_1 \cup T_2} \quad \frac{tyfv(\tau) = T}{tyfv(\forall t : \kappa.\, \tau) = T \setminus \{t\}}$$

$$\frac{tyfv(\tau) = T}{tyfv(\lambda t : \kappa.\, \tau) = T \setminus \{t\}} \quad \frac{tyfv(\tau_1) = T_1 \quad tyfv(\tau_2) = T_2}{tyfv(\tau_1\ \tau_2) = T_1 \cup T_2}$$

Type Substitution:

部分関数 $\{t_1 \longmapsto \tau_1, \dots, t_n \longmapsto \tau_n\}$ を，$[t_1 \leftarrow \tau_1, \dots, t_n \leftarrow \tau_n]$ または $[t_1, \dots, t_n \leftarrow t_1, \dots, t_n]$ と表記する．

$\boxed{e[\overline{t' \leftarrow \tau'}] = e'}$

$$\frac{}{x[\overline{t'} \leftarrow \overline{\tau'}] = x} \quad \frac{e_1[\overline{t'} \leftarrow \overline{\tau'}] = e_1'' \quad e_2[\overline{t'} \leftarrow \overline{\tau'}] = e_2''}{(e_1\ e_2)[\overline{t'} \leftarrow \overline{\tau'}] = e_1''\ e_2''} \quad \frac{\tau[\overline{t'} \leftarrow \overline{\tau'}] = \tau'' \quad e[\overline{t'} \leftarrow \overline{\tau'}] = e''}{(\lambda x : \tau.\, e)[\overline{t'} \leftarrow \overline{\tau'}] = \lambda x : \tau''.\, e''}$$

$$\frac{e[\overline{t'} \leftarrow \overline{\tau'}] = e'' \quad \tau[\overline{t'} \leftarrow \overline{\tau'}] = \tau''}{(e\ \tau)[\overline{t'} \leftarrow \overline{\tau'}] = e''\ \tau''} \quad \frac{e([\overline{t'} \leftarrow \overline{\tau'}] \restriction_{\mathrm{dom}([\overline{t'}\leftarrow\overline{\tau'}])\setminus\{t\}}) = e''}{(\Lambda t : \kappa.\, e)[\overline{t'} \leftarrow \overline{\tau'}] = \Lambda t : \kappa.\, e''}$$

$\boxed{\tau[\overline{t' \leftarrow \tau'}] = \tau''}$

$$\frac{[\overline{t'} \leftarrow \overline{\tau'}](t) = \tau}{t[\overline{t'} \leftarrow \overline{\tau'}] = \tau} \quad \frac{t \notin \mathrm{dom}([\overline{t'} \leftarrow \overline{\tau'}])}{t[\overline{t'} \leftarrow \overline{\tau'}] = t}$$

$$\frac{\tau_1[\overline{t'} \leftarrow \overline{\tau'}] = \tau_1'' \quad \tau_2[\overline{t'} \leftarrow \overline{\tau'}] = \tau_2''}{(\tau_1 \to \tau_2)[\overline{t'} \leftarrow \overline{\tau'}] = \tau_1'' \to \tau_2''} \quad \frac{\tau([\overline{t'} \leftarrow \overline{\tau'}] \restriction_{\mathrm{dom}([\overline{t'}\leftarrow\overline{\tau'}])\setminus\{t\}}) = \tau''}{(\forall t : \kappa.\, \tau)[\overline{t'} \leftarrow \overline{\tau'}] = \forall t : \kappa.\, \tau''}$$

$$\frac{\tau([\overline{t'} \leftarrow \overline{\tau'}] \restriction_{\mathrm{dom}([\overline{t'}\leftarrow\overline{\tau'}])\setminus\{t\}}) = \tau''}{(\lambda t : \kappa.\, \tau)[\overline{t'} \leftarrow \overline{\tau'}] = \lambda t : \kappa.\, \tau''} \quad \frac{\tau_1[\overline{t'} \leftarrow \overline{\tau'}] = \tau_1'' \quad \tau_2[\overline{t'} \leftarrow \overline{\tau'}] = \tau_2''}{(\tau_1\ \tau_2)[\overline{t'} \leftarrow \overline{\tau'}] = \tau_1''\ \tau_2''}$$

α-Equality:

$\boxed{e_1 \equiv_\alpha e_2}$

$$\frac{x_1 = x_2}{x_1 \equiv_\alpha x_2} \quad \frac{e_1 \equiv_\alpha e_2 \quad e_1' \equiv_\alpha e_2'}{e_1\ e_1' \equiv_\alpha e_2\ e_2'} \quad \frac{\tau_1 \equiv_\alpha \tau_2 \quad x' \notin fv(e_1) \cup fv(e_2) \quad e_1[x_1 \leftarrow x'] \equiv_\alpha e_2[x_2 \leftarrow x']}{\lambda x_1 : \tau_1.\, e_1 \equiv_\alpha \lambda x_2 : \tau_2.\, e_2}$$

$$\frac{e_1 \equiv_\alpha e_2 \quad \tau_1 \equiv_\alpha \tau_2}{e_1\ \tau_1 \equiv_\alpha e_2\ \tau_2} \quad \frac{t' \notin tyfv(e_1) \cup tyfv(e_2) \quad e_1[t_1 \leftarrow t'] \equiv_\alpha e_2[t_2 \leftarrow t']}{\Lambda t_1 : \kappa.\, e_1 \equiv_\alpha \Lambda t_2 : \kappa.\, e_2}$$

$\boxed{\tau_1 \equiv_\alpha \tau_2}$

$$\frac{t_1 = t_2}{t_1 \equiv_\alpha t_2} \quad \frac{\tau_1 \equiv_\alpha \tau_2 \quad \tau_1' \equiv_\alpha \tau_2'}{\tau_1 \to \tau_1' \equiv_\alpha \tau_2 \to \tau_2'} \quad \frac{t' \notin tyfv(\tau_1) \cup tyfv(\tau_2) \quad \tau_1[t_1 \leftarrow t'] \equiv_\alpha \tau_2[t_2 \leftarrow t']}{\forall t_1 : \kappa.\tau_1 \equiv_\alpha \forall t_2 : \kappa.\tau_2}$$

$$\frac{t' \notin tyfv(\tau_1) \cup tyfv(\tau_2) \quad \tau_1[t_1 \leftarrow t'] \equiv_\alpha \tau_2[t_2 \leftarrow t']}{\lambda t_1 : \kappa.\tau_1 \equiv_\alpha \lambda t_2 : \kappa.\tau_2} \quad \frac{\tau_1 \equiv_\alpha \tau_2 \quad \tau_1' \equiv_\alpha \tau_2'}{\tau_1\ \tau_1' \equiv_\alpha \tau_2\ \tau_2'}$$

定理 16 (Correctness of Substitution). 置換 $[\overline{x'} \leftarrow \overline{e'}]$ について，$X = \mathrm{dom}([\overline{x'} \leftarrow \overline{e'}])$ とした時,

$$fv(e[\overline{x'} \leftarrow \overline{e'}]) = (fv(e) \setminus X) \cup \bigcup_{x \in fv(e) \cap X} fv([\overline{x'} \leftarrow \overline{e'}](x)).$$

□

定理 17 (Correctness of Type Substitution). 式 e，型 τ，型置換 $[\overline{t'} \leftarrow \overline{\tau'}]$ について，$T = \mathrm{dom}([\overline{t'} \leftarrow \overline{\tau'}])$ とした時,

$$tyfv(e[\overline{t'} \leftarrow \overline{\tau'}]) = (tyfv(e) \setminus T) \cup \bigcup_{t \in tyfv(e) \cap T} tyfv([\overline{t'} \leftarrow \overline{\tau'}](t))$$
$$tyfv(\tau[\overline{t'} \leftarrow \overline{\tau'}]) = (tyfv(\tau) \setminus T) \cup \bigcup_{t \in tyfv(\tau) \cap T} tyfv([\overline{t'} \leftarrow \overline{\tau'}](t)).$$

□

定理 18 (α-Equality Does Not Touch Free Variables).

- $\tau_1 \equiv_\alpha \tau_2$ ならば $tyfv(\tau_1) = tyfv(\tau_2)$.
- $e_1 \equiv_\alpha e_2$ ならば，$fv(e_1) = fv(e_2)$，$tyfv(e_1) = tyfv(e_2)$.

□

2.6.2 Typing Semantics

Kinding:

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

$$\frac{\Gamma(t) = \kappa}{\Gamma \vdash t : \kappa}\ \text{K-Var}$$

$$\frac{\Gamma \vdash \tau_1 : \Omega \quad \Gamma \vdash \tau_2 : \Omega}{\Gamma \vdash \tau_1 \to \tau_2 : \Omega}\ \text{K-Arrow}$$

$$\frac{\Gamma, t : \kappa \vdash \tau : \Omega}{\Gamma \vdash \forall t : \kappa.\tau : \Omega}\ \text{K-Forall}$$

$$\frac{\Gamma, t : \kappa_1 \vdash \tau : \kappa_2}{\Gamma \vdash \lambda t : \kappa_1.\tau : \kappa_1 \to \kappa_2}\ \text{K-Abs}$$

$$\frac{\Gamma \vdash \tau_1 : \kappa_2 \to \kappa \quad \Gamma \vdash \tau_2 : \kappa_2}{\Gamma \vdash \tau_1\ \tau_2 : \kappa}\ \text{K-App}$$

Type equivalence:

$\boxed{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}$

$$\frac{\Gamma, t : \kappa_2 \vdash \tau_1 : \kappa \quad \Gamma \vdash \tau_2 : \kappa_2}{\Gamma \vdash (\lambda t : \kappa_2.\tau_1)\ \tau_2 \equiv \tau_1[t \leftarrow \tau_2] : \kappa}\ \text{T-Eq-}\beta\text{-Lam} \quad \frac{t \notin tyfv(\tau) \quad \Gamma \vdash \tau : \kappa_1 \to \kappa_2}{\Gamma \vdash (\lambda t : \kappa_1.\tau\ t) \equiv \tau : \kappa_1 \to \kappa_2}\ \text{T-Eq-}\eta\text{-Lam}$$

$$\frac{\tau_1 \equiv_\alpha \tau_2 \quad \Gamma \vdash \tau_1 : \kappa \quad \Gamma \vdash \tau_2 : \kappa}{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}\ \text{T-Eq-}\alpha\text{-Refl}$$

$$\frac{\Gamma \vdash \tau_2 \equiv \tau_1 : \kappa}{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}\ \text{T-Eq-Sym} \qquad \frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa \quad \Gamma \vdash \tau_2 \equiv \tau_3 : \kappa}{\Gamma \vdash \tau_1 \equiv \tau_3 : \kappa}\ \text{T-Eq-Trans}$$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \Omega \quad \Gamma \vdash \tau_1' \equiv \tau_2' : \Omega}{\Gamma \vdash \tau_1 \to \tau_1' \equiv \tau_2 \to \tau_2' : \Omega}\ \text{T-Eq-Cong-Arrow} \qquad \frac{\Gamma, t : \kappa \vdash \tau_1 \equiv \tau_2 : \Omega}{\Gamma \vdash \forall t : \kappa . \tau_1 \equiv \forall t : \kappa . \tau_2 : \Omega}\ \text{Eq-Cong-Forall}$$

$$\frac{\Gamma, t : \kappa \vdash \tau_1 \equiv \tau_2 : \kappa'}{\Gamma \vdash \lambda t : \kappa . \tau_1 \equiv \lambda t : \kappa . \tau_2 : \kappa \to \kappa'}\ \text{T-Eq-Cong-Abs} \qquad \frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa' \to \kappa \quad \Gamma \vdash \tau_1' \equiv \tau_2' : \kappa'}{\Gamma \vdash \tau_1\, \tau_1' \equiv \tau_2\, \tau_2' : \kappa}\ \text{Eq-Cong-App}$$

定理 19 (Respect Kinding). $\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa$ ならば，$\Gamma \vdash \tau_1 : \kappa$ かつ $\Gamma \vdash \tau_2 : \kappa$. □

Typing:

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

$$\frac{\Gamma \vdash \tau : \Omega \quad \Gamma(x) = \tau}{\Gamma \vdash x : \tau}\ \text{T-Var}$$

$$\frac{\Gamma \vdash \tau_1 : \Omega \quad \Gamma, x : \tau_1 \vdash e : \tau_2}{\Gamma \vdash \lambda x : \tau_1 . e : \tau_1 \to \tau_2}\ \text{T-Abs}$$

$$\frac{\Gamma \vdash e_1 : \tau_2 \to \tau \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1\, e_2 : \tau}\ \text{T-App}$$

$$\frac{\Gamma, t : \kappa \vdash e : \tau}{\Gamma \vdash \Lambda t : \kappa . e : \forall t : \kappa . \tau}\ \text{T-UnivAbs}$$

$$\frac{\Gamma \vdash e : \forall t : \kappa . \tau_1 \quad \Gamma \vdash \tau_2 : \kappa}{\Gamma \vdash e\, \tau_2 : \tau_1[t \leftarrow \tau_2]}\ \text{T-UnivApp}$$

$$\frac{\Gamma \vdash \tau \equiv \tau' : \Omega \quad \Gamma \vdash e : \tau'}{\Gamma \vdash e : \tau}\ \text{T-Equiv}$$

特に，$\cdot \vdash e : \tau$ の時，$e : \tau$ と表記.

定理 20 (Respect Type Kind). $\Gamma \vdash e : \tau$ ならば，$\Gamma \vdash \tau : \Omega$. □

2.6.3 Evaluation Semantics (Call-By-Value)

$$\begin{aligned} v &::= \lambda x : \tau . e \\ &\mid \Lambda t : \kappa . e \\ C &::= [] \\ &\mid C\, e \\ &\mid v\, C \\ &\mid C\, \tau \end{aligned}$$

Small Step:

$\boxed{e \Rightarrow e'}$

$$\frac{}{(\lambda x : \tau . e)\, v \Rightarrow e[x \leftarrow v]}$$

$$\frac{}{(\Lambda t : \kappa . e)\, \tau \Rightarrow e[t \leftarrow \tau]}$$

$$\frac{e \Rightarrow e'}{C[e] \Rightarrow C[e']}$$

Big Step:

$\boxed{e \Downarrow v}$

$$\frac{e_1 \Downarrow \lambda x : \tau . e_1' \quad e_2 \Downarrow v_2 \quad e_1'[x \leftarrow v_2] \Downarrow v}{e_1\, e_2 \Downarrow v}$$

$$\frac{e \Downarrow \Lambda t : \kappa . e_1' \quad e_1'[t \leftarrow \tau] \Downarrow v}{e\,\tau \Downarrow v}$$

定理 21 (Adequacy of Small Step and Big Step). $e \Rightarrow^* v$ iff $e \Downarrow v$. □

定理 22 (Type Soundness). $e : \tau$ の時，$e \Rightarrow^* v$，$e \Downarrow v$ となる $v = \mathrm{nf}(\Rightarrow, e)$ が存在し，

- $\tau = \tau_1 \to \tau_2$ の時，$v \equiv_\alpha \lambda x' : \tau_1 . e'$ となる $\lambda x' : \tau_1 . e'$ が存在する．
- $\tau = \forall t : \kappa . \tau_1$ の時，$v \equiv_\alpha \Lambda t : \kappa . e'$ となる $\Lambda t : \kappa . e'$ が存在する．

□

2.6.4 Equational Reasoning

$\boxed{\Gamma \vdash e_1 \equiv e_2 : \tau}$

$$\frac{\Gamma, x : \tau_2 \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (\lambda x : \tau_2 . e_1)\, e_2 \equiv e_1[x \leftarrow e_2] : \tau}\ \text{Eq-}\beta\text{-Lam} \qquad \frac{x \notin fv(e) \quad \Gamma \vdash e : \tau_1 \to \tau_2}{\Gamma \vdash (\lambda x : \tau_1 . e\ x) \equiv e : \tau_1 \to \tau_2}\ \text{Eq-}\eta\text{-Lam}$$

$$\frac{\Gamma, t : \kappa \vdash e : \tau}{\Gamma \vdash (\Lambda t : \kappa . e)\, \tau_2 \equiv e[t \leftarrow \tau_2] : \tau[t \leftarrow \tau_2]}\ \text{Eq-}\beta\text{-UnivLam} \qquad \frac{t \notin tyfv(e) \quad \Gamma \vdash e : \forall t : \kappa . \tau}{\Gamma \vdash (\Lambda t : \kappa . e\ t) \equiv e : \forall t : \kappa . \tau}\ \text{Eq-}\eta\text{-UnivLam}$$

$$\frac{e_1 \equiv_\alpha e_2 \quad \Gamma \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1 \equiv e_2 : \tau}\ \text{Eq-}\alpha\text{-Refl} \qquad \frac{\tau \equiv_\alpha \tau' \quad \Gamma \vdash e_1 \equiv e_2 : \tau'}{\Gamma \vdash e_1 \equiv e_2 : \tau}\ \text{Eq-}\alpha\text{-Type}$$

$$\frac{\Gamma \vdash e_2 \equiv e_1 : \tau}{\Gamma \vdash e_1 \equiv e_2 : \tau}\ \text{Eq-Sym} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau \quad \Gamma \vdash e_2 \equiv e_3 : \tau}{\Gamma \vdash e_1 \equiv e_3 : \tau}\ \text{Eq-Trans}$$

$$\frac{\Gamma, x : \tau \vdash e_1 \equiv e_2 : \tau'}{\Gamma \vdash \lambda x : \tau . e_1 \equiv \lambda x : \tau . e_2 : \tau \to \tau'}\ \text{Eq-Cong-Abs} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau' \to \tau \quad \Gamma \vdash e_1' \equiv e_2' : \tau'}{\Gamma \vdash e_1\, e_1' \equiv e_2\, e_2' : \tau}\ \text{Eq-Cong-App}$$

$$\frac{\Gamma, t : \kappa \vdash e_1 \equiv e_2 : \tau}{\Gamma \vdash \Lambda t : \kappa . e_1 \equiv \Lambda t : \kappa . e_2 : (\forall t : \kappa . \tau)}\ \text{Eq-Cong-UnivAbs}$$

$$\frac{\Gamma \vdash e_1 \equiv e_2 : \forall t : \kappa . \tau \quad \Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}{\Gamma \vdash e_1\, \tau_1 \equiv e_2\, \tau_2 : \tau[t \leftarrow \tau_1]}\ \text{Eq-Cong-UnivApp}$$

特に，$\cdot \vdash e_1 \equiv e_2 : \tau$ の時，$e_1 \equiv e_2 : \tau$ と表記．

定理 23 (Respect Typing). $\Gamma \vdash e_1 \equiv e_2 : \tau$ ならば，$\Gamma \vdash e_1 : \tau$ かつ $\Gamma \vdash e_2 : \tau$. □

定理 24 (Respect Evaluation). $e_1 \equiv e_2 : \tau$ の時，$e_1' \Rightarrow^* e_1$，$e_2 \Rightarrow^* e_2'$ ならば $e_1' \equiv e_2' : \tau$. □

系 25. $e_1 \equiv e_2 : \tau$ の時，$e_1 \Rightarrow^* e_1'$，$e_2 \Rightarrow^* e_2'$ ならば $e_1' \equiv e_2' : \tau$. □

証明. $e_1 \Rightarrow^* e_1$ より，定理 14 から $e_1 \equiv e_2' : \tau$．よって，T-Sym から $e_2' \equiv e_1 : \tau$ であり，$e_2' \Rightarrow^* e_2'$ より定理 14 から $e_2' \equiv e_1' : \tau$．故に，T-Sym から $e_1' \equiv e_2' : \tau$. ■

2.6.5 Definability

Product

Product of τ_1 and τ_2:

$$\begin{aligned}
\tau_1 \times \tau_2 &\stackrel{\text{def}}{=} \forall t : \Omega . (\tau_1 \to \tau_2 \to t) \to t \\
\langle e_1, e_2 \rangle &\stackrel{\text{def}}{=} \Lambda t : \Omega . \lambda x : \tau_1 \to \tau_2 \to t . x\ e_1\ e_2 \\
\pi_1 e &\stackrel{\text{def}}{=} e\ \tau_1\ \lambda x_1 . \lambda x_2 . x_1 \\
\pi_2 e &\stackrel{\text{def}}{=} e\ \tau_2\ \lambda x_1 . \lambda x_2 . x_2
\end{aligned}$$

Admissible kinding:

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

$$\frac{\Gamma \vdash \tau_1 : \Omega \quad \Gamma \vdash \tau_2 : \Omega}{\Gamma \vdash \tau_1 \times \tau_2 : \Omega}\ \text{T-Product}$$

Admissible type equality:

$\boxed{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \Omega \quad \Gamma \vdash \tau_1' \equiv \tau_2' : \Omega}{\Gamma \vdash \tau_1 \times \tau_1' \equiv \tau_2 \times \tau_2' : \Omega}\ \text{T-Eq-Product}$$

Admissible typing:

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

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \langle e_1, e_2 \rangle : \tau_1 \times \tau_2}\ \text{T-Product} \qquad \frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \pi_1 e : \tau_1}\ \text{T-Proj-1} \qquad \frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \pi_2 e : \tau_2}\ \text{T-Proj-2}$$

Admissible equality:

$\boxed{\Gamma \vdash e_1 \equiv e_2 : \tau}$

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \pi_1 \langle e_1, e_2 \rangle \equiv e_1 : \tau_1}\ \text{Eq-}\beta\text{-Product-1} \qquad \frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \pi_2 \langle e_1, e_2 \rangle \equiv e_2 : \tau_2}\ \text{Eq-}\beta\text{-Product-2}$$

$$\frac{\Gamma \vdash e : \tau_1 \times \tau_2}{\Gamma \vdash \langle \pi_1 e, \pi_2 e \rangle \equiv e : \tau_1 \times \tau_2}\ \text{Eq-}\eta\text{-Product}$$

Existential Type

Existence of $\exists t : \kappa. \tau$:

$$\begin{aligned}
\exists t : \kappa. \tau &\stackrel{\text{def}}{=} \forall t' : \Omega. (\forall t : \kappa. \tau \to t') \to t' \\
\text{pack}\langle \tau_t, e \rangle_{\exists t : \kappa. \tau} &\stackrel{\text{def}}{=} \Lambda t' : \Omega. \lambda x : (\forall t : \kappa. \tau \to t'). x\ \tau_t\ e \\
\text{unpack}\langle t : \kappa, x : \tau \rangle = e_1. \tau_2. e_2 &\stackrel{\text{def}}{=} e_1\ \tau_2\ (\Lambda t : \kappa. \lambda x : \tau. e_2)
\end{aligned}$$

Admissible kinding:

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

$$\frac{\Gamma, t : \kappa \vdash \tau : \Omega}{\Gamma \vdash \exists t : \kappa. \tau : \Omega}\ \text{T-Exist}$$

Admissible type equality:

$\boxed{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}$

$$\frac{\Gamma, t : \kappa \vdash \tau_1 \equiv \tau_2 : \Omega}{\Gamma \vdash \exists t : \kappa. \tau_1 \equiv \exists t : \kappa. \tau_2 : \Omega}\ \text{T-Eq-Cong-Exist}$$

Admissible typing rule:

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

$$\frac{\Gamma, t : \kappa \vdash \tau : \Omega \quad \Gamma \vdash \tau_t : \kappa \quad \Gamma \vdash e : \tau[t \leftarrow \tau_t]}{\Gamma \vdash \mathrm{pack}\langle \tau_t, e\rangle_{\exists t : \kappa.\tau} : \exists t : \kappa.\tau} \text{ T-Pack}$$

$$\frac{\Gamma \vdash e_1 : \exists t : \kappa.\tau \quad \Gamma, t : \kappa, x : \tau \vdash e_2 : \tau_2 \quad t \notin tyfv(\tau_2)}{\Gamma \vdash \mathrm{unpack}\langle t : \kappa, x : \tau\rangle = e_1.\, \tau_2.\, e_2 : \tau_2} \text{ T-Unpack}$$

Admissible equality:

$\boxed{\Gamma \vdash e_1 \equiv e_2 : \tau}$

$$\frac{\Gamma \vdash \tau_t : \kappa \quad \Gamma \vdash e_1 : \tau_1[t \leftarrow \tau_t] \quad \Gamma, t : \kappa, x : \tau_1 \vdash e_2 : \tau_2 \quad t \notin tyfv(\tau_2)}{\Gamma \vdash \mathrm{unpack}\langle t : \kappa, x : \tau_1\rangle = \mathrm{pack}\langle \tau_t, e_1\rangle_{\exists t : \kappa.\tau_1}.\, \tau_2.\, e_2 \equiv e_2[t \leftarrow \tau_t][x \leftarrow e_1] : \tau_2} \text{ Eq-}\beta\text{-Exist}$$

$$\frac{\Gamma \vdash e : (\exists t : \kappa.\tau) \quad \tau' \equiv \exists t : \kappa.\tau}{\Gamma \vdash \mathrm{unpack}\langle t : \kappa, x : \tau\rangle = e.\, \tau'.\, \mathrm{pack}\langle t, x\rangle_{\exists t : \kappa.\tau} \equiv e : (\exists t : \kappa.\tau)} \text{ Eq-}\eta\text{-Exist}$$

2.7 WIP: λ μ-Calculus

2.8 WIP: Lambda Bar Mu Mu Tilde Calculus

$\bar{\lambda}\,\mu\,\tilde{\mu}$-Calculus

2.9 WIP: π-Calculus

第 3 章

Modules and Phase Distinction

3.1 F-ing modules

[RRD14]

3.1.1 Syntax

$$
\begin{array}{rll}
X & ::= \cdots & \text{(identifier)} \\
K & ::= \cdots & \text{(kind)} \\
T & ::= \cdots \mid P & \text{(type)} \\
E & ::= \cdots \mid P & \text{(expression)} \\
P & ::= M & \text{(path)} \\
M & ::= X & \text{(identifier)} \\
 & \mid \{B\} & \text{(bindings)} \\
 & \mid M.X & \text{(projection)} \\
 & \mid \mathsf{fun}\, X : S \Rightarrow M & \text{(functor)} \\
 & \mid X\ X & \text{(functor application)} \\
 & \mid X :> S & \text{(sealing)} \\
B & ::= \mathsf{val}\, X = E & \text{(value binding)} \\
 & \mid \mathsf{type}\, X = T & \text{(type binding)} \\
 & \mid \mathsf{module}\, X = M & \text{(module binding)} \\
 & \mid \mathsf{signature}\, X = S & \text{(signature binding)} \\
 & \mid \mathsf{include}\, M & \text{(module including)} \\
 & \mid \epsilon & \text{(empty binding)} \\
 & \mid B; B & \text{(binding concatenation)} \\
S & ::= P & \text{(signature path)} \\
 & \mid \{D\} & \text{(declarations)} \\
 & \mid (X : S) \to S & \text{((generative) functor signature)} \\
 & \mid S\ \mathsf{where\ type}\, \overline{X} = T & \text{(bounded signature)} \\
D & ::= \mathsf{val}\, X : T & \text{(value declaration)} \\
 & \mid \mathsf{type}\, X = T & \text{(type binding)} \\
 & \mid \mathsf{type}\, X : K & \text{(type declaration)} \\
 & \mid \mathsf{module}\, X : S & \text{(module declaration)} \\
 & \mid \mathsf{signature}\, X = S & \text{(signature binding)} \\
 & \mid \mathsf{include}\, S & \text{(signature including)} \\
 & \mid \epsilon & \text{(empty declaration)} \\
 & \mid D; D & \text{(declaration concatenation)}
\end{array}
$$

3.1.2 Internal Language

Having same power as System F ω

Syntax:

$$
\begin{aligned}
\kappa &::= \Omega \mid \kappa \to \kappa \\
\tau &::= t \mid \tau \to \tau \mid \{\overline{l : \tau}\} \mid \forall t : \kappa.\tau \mid \exists t : \kappa.\tau \mid \lambda t : \kappa.\tau \mid \tau\ \tau \\
e &::= x \mid \lambda x : \tau.e \mid e\ e \mid \{\overline{l = e}\} \mid e.l \mid \Lambda t : \kappa.e \mid e\ \tau \mid \mathsf{pack}\langle \tau, e\rangle_\tau \mid \mathsf{unpack}\langle t : \kappa, x : \tau\rangle = e\ \mathsf{in}\ e
\end{aligned}
$$

$$\Gamma ::= \cdot \mid \Gamma, t : \kappa \mid \Gamma, x : \tau$$

Abbreviation:

$$\Sigma.\bar{l} \stackrel{\text{def}}{=} \begin{cases} (\Sigma.l).\overline{l'} & (\bar{l} = l\ \overline{l'}) \\ \Sigma & (\bar{l} = \epsilon) \end{cases}$$

$$\overline{\tau_1} \to \tau_2 \stackrel{\text{def}}{=} \begin{cases} \tau_1 \to (\overline{\tau'_1} \to \tau_2) & (\overline{\tau_1} = \tau_1\ \overline{\tau'_1}) \\ \tau_2 & (\overline{\tau_1} = \epsilon) \end{cases}$$

$$\lambda\overline{x : \tau}.e \stackrel{\text{def}}{=} \begin{cases} \lambda x : \tau.\lambda\overline{x' : \tau'}.e & (\overline{x : \tau} = x : \tau\ \overline{x' : \tau'}) \\ e & (\overline{x : \tau} = \epsilon) \end{cases}$$

$$e_0\ \overline{e_1} \stackrel{\text{def}}{=} \begin{cases} e_0\ e_1\ \overline{e'_1} & (\overline{e_1} = e_1\ \overline{e'_1}) \\ e_0 & (\overline{e_1} = \epsilon) \end{cases}$$

$$\forall\overline{t : \kappa}.\tau \stackrel{\text{def}}{=} \begin{cases} \forall t : \kappa.\forall\overline{t' : \kappa'}.\tau & (\overline{t : \kappa} = t : \kappa\ \overline{t' : \kappa'}) \\ \tau & (\overline{t : \kappa} = \epsilon) \end{cases}$$

$$\Lambda\overline{t : \kappa}.e \stackrel{\text{def}}{=} \begin{cases} \Lambda t : \kappa.\Lambda\overline{t' : \kappa'}.e & (\overline{t : \kappa} = t : \kappa\ \overline{t' : \kappa'}) \\ e & (\overline{t : \kappa} = \epsilon) \end{cases}$$

$$e\ \overline{\tau} \stackrel{\text{def}}{=} \begin{cases} e\ \tau\ \overline{\tau'} & (\overline{\tau} = \tau\ \overline{\tau'}) \\ e & (\overline{\tau} = \epsilon) \end{cases}$$

$$\text{let}\,\overline{x : \tau = e_1}\ \overline{t : \kappa = \tau}\ \text{in}\ e_2 \stackrel{\text{def}}{=} (\lambda\overline{x : \tau}.\Lambda\overline{t : \kappa}.e_2)\ \overline{e_1}\ \overline{\tau}$$

$$\exists\overline{t : \kappa}.\tau \stackrel{\text{def}}{=} \begin{cases} \exists t : \kappa.\exists\overline{t' : \kappa'}.\tau & (\overline{t : \kappa} = t : \kappa\ \overline{t' : \kappa'}) \\ \tau & (\overline{t : \kappa} = \epsilon) \end{cases}$$

$$\text{pack}\langle\overline{\tau}, e\rangle_{\exists\overline{t:\kappa}.\tau_0} \stackrel{\text{def}}{=} \begin{cases} \text{pack}\langle\tau, \text{pack}\langle\overline{\tau'}, e\rangle_{\exists\overline{t':\kappa'}.\tau_0}\rangle_{\exists\overline{t:\kappa}.\tau_0} & (\overline{\tau} = \tau\ \overline{\tau'}, \overline{t : \kappa} = t : \kappa\ \overline{t' : \kappa'}) \\ e & (\overline{\tau} = \epsilon, \overline{t : \kappa} = \epsilon) \end{cases}$$

$$(\text{unpack}\langle\overline{t : \kappa}, x : \tau\rangle = e_1\ \text{in}\ e_2) \stackrel{\text{def}}{=} \begin{cases} \begin{array}{c}\text{unpack}\langle t : \kappa, x_1 : \exists\overline{t' : \kappa'}.\tau\rangle = e_1\ \text{in} \\ \text{unpack}\langle\overline{t' : \kappa'}, x_2 : \tau\rangle = x_1\ \text{in}\ e_2\end{array} & (\overline{t : \kappa} = t : \kappa\ \overline{t' : \kappa'}) \\ \text{let}\,x : \tau = e_1\ \text{in}\ e_2 & (\overline{t : \kappa} = \epsilon) \end{cases}$$

Kinding:

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

$$\frac{\Gamma(t) = \kappa}{\Gamma \vdash t : \kappa} \qquad \frac{\Gamma \vdash \tau_1 : \Omega \quad \Gamma \vdash \tau_2 : \Omega}{\Gamma \vdash \tau_1 \to \tau_2 : \Omega} \qquad \frac{\bigwedge_l \Gamma \vdash \tau_l : \Omega}{\Gamma \vdash \{\overline{l : \tau_l}\} : \Omega}$$

$$\frac{\Gamma, t : \kappa \vdash \tau : \Omega}{\Gamma \vdash \forall t : \kappa.\tau : \Omega} \qquad \frac{\Gamma, t : \kappa \vdash \tau : \Omega}{\Gamma \vdash \exists t : \kappa.\tau : \Omega} \qquad \frac{\Gamma, t : \kappa_1 \vdash \tau : \kappa_2}{\Gamma \vdash \lambda t : \kappa_1.\tau : \kappa_1 \to \kappa_2} \qquad \frac{\Gamma \vdash \tau_1 : \kappa_2 \to \kappa \quad \Gamma \vdash \tau_2 : \kappa_2}{\Gamma \vdash \tau_1\ \tau_2 : \kappa}$$

Type equivalence:

$\boxed{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}$

$$\frac{\Gamma, t : \kappa_2 \vdash \tau_1 : \kappa \quad \Gamma \vdash \tau_2 : \kappa_2}{\Gamma \vdash (\lambda t : \kappa_2.\tau_1)\ \tau_2 \equiv \tau_1[t \leftarrow \tau_2] : \kappa} \qquad \frac{t \notin tyfv(\tau) \quad \Gamma \vdash \tau : \kappa_1 \to \kappa_2}{\Gamma \vdash (\lambda t : \kappa_1.\tau\ t) \equiv \tau : \kappa_1 \to \kappa_2}$$

$$\frac{\tau_1 \equiv_\alpha \tau_2 \quad \Gamma \vdash \tau_1 : \kappa \quad \Gamma \vdash \tau_2 : \kappa}{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa} \qquad \frac{\Gamma \vdash \tau_2 \equiv \tau_1 : \kappa}{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa} \qquad \frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa \quad \Gamma \vdash \tau_2 \equiv \tau_3 : \kappa}{\Gamma \vdash \tau_1 \equiv \tau_3 : \kappa}$$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \Omega \quad \Gamma \vdash \tau'_1 \equiv \tau'_2 : \Omega}{\Gamma \vdash \tau_1 \to \tau'_1 \equiv \tau_2 \to \tau'_2 : \Omega} \qquad \frac{\Gamma, t : \kappa \vdash \tau_1 \equiv \tau_2 : \Omega}{\Gamma \vdash \forall t : \kappa.\tau_1 \equiv \forall t : \kappa.\tau_2 : \Omega}$$

$$\frac{\Gamma, t : \kappa \vdash \tau_1 \equiv \tau_2 : \kappa'}{\Gamma \vdash \lambda t : \kappa.\tau_1 \equiv \lambda t : \kappa.\tau_2 : \kappa \to \kappa'} \qquad \frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa' \to \kappa \quad \Gamma \vdash \tau'_1 \equiv \tau'_2 : \kappa'}{\Gamma \vdash \tau_1\ \tau'_1 \equiv \tau_2\ \tau'_2 : \kappa}$$

Typing:

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

$$\frac{\Gamma \vdash \tau : \Omega \quad \Gamma(x) = \tau}{\Gamma \vdash x : \tau} \qquad \frac{\Gamma \vdash \tau \equiv \tau' : \Omega \quad \Gamma \vdash e : \tau'}{\Gamma \vdash e : \tau}$$

$$\frac{\Gamma \vdash \tau_1 : \Omega \quad \Gamma, x : \tau_1 \vdash e : \tau_2}{\Gamma \vdash \lambda x : \tau_1 . e : \tau_1 \to \tau_2} \qquad \frac{\Gamma \vdash e_1 : \tau_2 \to \tau \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1\ e_2 : \tau}$$

$$\frac{\bigwedge_l \Gamma \vdash e_l : \tau_l}{\Gamma \vdash \{\overline{l = e_l}\} : \{\overline{l = \tau_l}\}} \qquad \frac{\Gamma \vdash e : \{\overline{l' = \tau_{l'}}\}}{\Gamma \vdash e.l : \tau_l}$$

$$\frac{\Gamma, t : \kappa \vdash e : \tau}{\Gamma \vdash \Lambda t : \kappa . e : (\forall t : \kappa . \tau)} \qquad \frac{\Gamma \vdash e : (\forall t : \kappa . \tau_1) \quad \Gamma \vdash \tau_2 : \kappa}{\Gamma \vdash e\ \tau_2 : \tau_1[t \leftarrow \tau_2]}$$

$$\frac{\Gamma, t : \kappa \vdash \tau : \Omega \quad \Gamma \vdash \tau_t : \kappa \quad \Gamma \vdash e : \tau[t \leftarrow \tau_t]}{\Gamma \vdash \mathrm{pack}\langle \tau_t, e\rangle_{\exists t : \kappa . \tau} : (\exists t : \kappa . \tau)} \qquad \frac{\Gamma \vdash e_1 : (\exists t : \kappa . \tau_1) \quad \Gamma, t : \kappa, x : \tau_1 \vdash e_2 : \tau}{\Gamma \vdash \mathrm{unpack}\langle t : \kappa, x : \tau_1\rangle = e_1 \ \mathrm{in}\ e_2 : \tau}$$

Reduction:

$$v ::= \lambda x : \tau . e \mid \{\overline{l = e}\} \mid \Lambda t : \kappa . e \mid \mathrm{pack}\langle \tau_t, e\rangle_{\exists t : \kappa . \tau}$$

$$C ::= [\,] \mid C\ e \mid v\ C \mid \{\overline{l = v}, l = C, \overline{l = e}\} \mid C.l \mid C\ \tau \mid \mathrm{pack}\langle \tau, C\rangle_\tau \mid \mathrm{unpack}\langle t : \kappa, x : \tau\rangle = C \ \mathrm{in}\ e$$

$\boxed{e \Rightarrow e'}$

$$\frac{}{(\lambda x : \tau . e) v \Rightarrow e[x \leftarrow v]} \qquad \frac{}{\{\overline{l' = v_{l'}}\}.l \Rightarrow v_l} \qquad \frac{}{(\Lambda t : \kappa . e)\tau \Rightarrow e[t \leftarrow \tau]}$$

$$\frac{}{\mathrm{unpack}\langle t : \kappa, x : \tau\rangle = \mathrm{pack}\langle \tau_t, v\rangle_{\tau_\exists} \ \mathrm{in}\ e \Rightarrow e[t \leftarrow \tau_t][x \leftarrow v]} \qquad \frac{e \Rightarrow e'}{C[e] \Rightarrow C[e']}$$

Equivalence:

$\boxed{\Gamma \vdash e_1 \equiv e_2 : \tau}$

$$\frac{\Gamma, x : \tau_2 \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (\lambda x : \tau_2 . e_1)\ e_2 \equiv e_1[x \leftarrow e_2] : \tau} \qquad \frac{x \notin fv(e) \quad \Gamma \vdash e : \tau_1 \to \tau_2}{\Gamma \vdash (\lambda x : \tau_1 . e\ x) \equiv e : \tau_1 \to \tau_2}$$

$$\frac{\bigwedge_{l'} \Gamma \vdash e_{l'} : \tau_{l'}}{\Gamma \vdash \{\overline{l' = e_{l'}}\}.l \equiv e_l : \tau_l} \qquad \frac{\Gamma \vdash e : \{\overline{l : \tau_l}\}}{\Gamma \vdash \{\overline{l = e.l}\} \equiv e : \{\overline{l : \tau_l}\}}$$

$$\frac{\Gamma, t : \kappa \vdash e : \tau}{\Gamma \vdash (\Lambda t : \kappa . e)\ \tau_2 \equiv e[t \leftarrow \tau_2] : \tau[t \leftarrow \tau_2]} \qquad \frac{t \notin tyfv(e) \quad \Gamma \vdash e : \forall t : \kappa . \tau}{\Gamma \vdash (\Lambda t : \kappa . e\ t) \equiv e : \forall t : \kappa . \tau}$$

$$\frac{\Gamma, t : \kappa \vdash \tau_1 \equiv \tau_1' : \Omega \quad \Gamma \vdash \tau_t : \kappa \quad \Gamma \vdash e_1 : \tau_1[t \leftarrow \tau_t] \quad \Gamma, t : \kappa, x : \tau_1 \vdash e_2 : \tau}{\Gamma \vdash \mathrm{unpack}\langle t : \kappa, x : \tau_1'\rangle = \mathrm{pack}\langle \tau_t, e_1\rangle_{\exists t : \kappa . \tau_1} \ \mathrm{in}\ e_2 \equiv e_2[t \leftarrow \tau_t][x \leftarrow e_1] : \tau}$$

$$\frac{\Gamma \vdash e : \exists t : \kappa . \tau \quad \Gamma, t : \kappa \vdash \tau \equiv \tau' : \Omega}{\Gamma \vdash \mathrm{unpack}\langle t : \kappa, x : \tau'\rangle = e \ \mathrm{in}\ \mathrm{pack}\langle t, x\rangle_{\exists t : \kappa . \tau} \equiv e : (\exists t : \kappa . \tau)}$$

$$\frac{e_1 \equiv_\alpha e_2 \quad \Gamma \vdash e_1 : \tau \quad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1 \equiv e_2 : \tau} \qquad \frac{\Gamma \vdash \tau \equiv \tau' : \Omega \quad \Gamma \vdash e_1 \equiv e_2 : \tau'}{\Gamma \vdash e_1 \equiv e_2 : \tau}$$

$$\frac{\Gamma \vdash e_2 \equiv e_1 : \tau}{\Gamma \vdash e_1 \equiv e_2 : \tau} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau \quad \Gamma \vdash e_2 \equiv e_3 : \tau}{\Gamma \vdash e_1 \equiv e_3 : \tau}$$

$$\frac{\Gamma, x : \tau \vdash e_1 \equiv e_2 : \tau'}{\Gamma \vdash \lambda x : \tau . e_1 \equiv \lambda x : \tau . e_2 : \tau \to \tau'} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau' \to \tau \quad \Gamma \vdash e_1' \equiv e_2' : \tau'}{\Gamma \vdash e_1\ e_1' \equiv e_2\ e_2' : \tau}$$

$$\frac{\bigwedge_l \Gamma \vdash e_{l,1} \equiv e_{l,2} : \tau_l}{\Gamma \vdash \{\overline{l = e_{l,1}}\} \equiv \{\overline{l = e_{l,2}}\} : \{\overline{l : \tau_l}\}} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \{l : \tau_l, \overline{l' : \tau'}\}}{\Gamma \vdash e_1.l \equiv e_2.l : \tau_l}$$

$$\frac{\Gamma, t : \kappa \vdash e_1 \equiv e_2 : \tau}{\Gamma \vdash \Lambda t : \kappa . e_1 \equiv \Lambda t : \kappa . e_2 : (\forall t : \kappa . \tau)} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \forall t : \kappa . \tau \quad \Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}{\Gamma \vdash e_1\ \tau_1 \equiv e_2\ \tau_2 : \tau[t \leftarrow \tau_1]}$$

$$\frac{\Gamma \vdash \tau_1' \equiv \tau_2' : \kappa \quad \Gamma \vdash e_1 \equiv e_2 : \tau_1[t \leftarrow \tau_1'] \quad \Gamma, t : \kappa. \tau_1 \equiv \tau_2 : \Omega}{\Gamma \vdash \mathrm{pack}\langle \tau_1', e_1 \rangle_{\exists t : \kappa. \tau_1} \equiv \mathrm{pack}\langle \tau_2', e_2 \rangle_{\exists t : \kappa. \tau_2} : (\exists t : \kappa. \tau_1)}$$

$$\frac{\Gamma, t : \kappa \vdash \tau_1' \equiv \tau_2' : \Omega \quad \Gamma \vdash e_1' \equiv e_2' : (\exists t : \kappa. \tau_1') \quad \Gamma, t : \kappa, x : \tau_1' \vdash e_1 \equiv e_2 : \tau}{\Gamma \vdash \mathrm{unpack}\langle t : \kappa, x : \tau_1' \rangle = e_1' \text{ in } e_1 \equiv \mathrm{unpack}\langle t : \kappa, x : \tau_2' \rangle = e_2' \text{ in } e_2 : \tau}$$

3.1.3 Signature

$$\begin{aligned}
\Xi &::= \exists \overline{t : \kappa}. \Sigma && \text{(abstract signature)} \\
\Sigma &::= [\tau] && \text{(atomic value declaration)} \\
&\mid [= \tau : \kappa] && \text{(atomic type declaration)} \\
&\mid [= \Xi] && \text{(atomic signature declaration)} \\
&\mid \{\overline{l_X : \Sigma}\} && \text{(structure signature)} \\
&\mid \forall \overline{t : \kappa}. \Sigma \to \Xi && \text{(functor signature)}
\end{aligned}$$

Atomic Signature:

$$\begin{aligned}
[\tau] &\stackrel{\text{def}}{=} \{\mathrm{val} : \tau\} \\
[e] &\stackrel{\text{def}}{=} \{\mathrm{val} = e\} \\
[= \tau : \kappa] &\stackrel{\text{def}}{=} \{\mathrm{type} : \forall t : (\kappa \to \Omega).\, t\ \tau \to t\ \tau\} \\
[\tau : \kappa] &\stackrel{\text{def}}{=} \{\mathrm{type} = \Lambda t : (\kappa \to \Omega).\, \lambda x : (t\ \tau).\, x\} \\
[= \Xi] &\stackrel{\text{def}}{=} \{\mathrm{sig} : \Xi \to \Xi\} \\
[\Xi] &\stackrel{\text{def}}{=} \{\mathrm{sig} = \lambda x : \Xi.\, x\}
\end{aligned}$$

$\boxed{\mathrm{NotAtomic}(\Sigma)}$

$$\frac{}{\mathrm{NotAtomic}(\{\overline{l_X : \Sigma}\})} \qquad \frac{}{\mathrm{NotAtomic}(\forall \overline{t : \kappa}. \Sigma \to \Xi)}$$

Admissible kinding:

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

$$\frac{\Gamma \vdash \tau : \Omega}{\Gamma \vdash [\tau] : \Omega}\ \text{K-A-Val}$$

$$\frac{\Gamma \vdash \tau : \kappa}{\Gamma \vdash [= \tau : \kappa] : \Omega}\ \text{K-A-Typ}$$

$$\frac{\Gamma \vdash \Xi : \Omega}{\Gamma \vdash [= \Xi] : \Omega}\ \text{K-A-Sig}$$

Admissible type equivalence:

$\boxed{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \Omega}{\Gamma \vdash [\tau_1] \equiv [\tau_2] : \Omega}\ \text{T-Eq-Cong-A-Val}$$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}{\Gamma \vdash [= \tau_1 : \kappa] \equiv [= \tau_2 : \kappa] : \Omega}\ \text{T-Eq-Cong-A-Typ}$$

$$\frac{\Gamma \vdash \Xi_1 \equiv \Xi_2 : \Omega}{\Gamma \vdash [= \Xi_1] \equiv [= \Xi_2] : \Omega}\ \text{T-Eq-Cong-A-Sig}$$

Admissible typing:

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

$$\frac{\Gamma \vdash e : \tau}{\Gamma \vdash [e] : [\tau]}\ \text{T-A-Val}$$

$$\frac{\Gamma \vdash \tau : \kappa}{\Gamma \vdash [\tau : \kappa] : [= \tau : \kappa]}\ \text{T-A-Typ}$$

$$\frac{\Gamma \vdash \Xi : \Omega}{\Gamma \vdash [\Xi] : [= \Xi]}\ \text{T-A-Sig}$$

Admissible equivalence:

$\boxed{\Gamma \vdash e_1 \equiv e_2 : \tau}$

$$\frac{\Gamma \vdash e : \tau}{\Gamma \vdash [e].\mathrm{val} \equiv e : \tau}\ \text{Eq-}\beta\text{-A-Val} \qquad \frac{\Gamma \vdash e : [\tau]}{\Gamma \vdash [e.\mathrm{val}] \equiv e : [\tau]}\ \text{Eq-}\eta\text{-A-Val} \qquad \frac{\Gamma \vdash e_1 \equiv e_2 : \tau}{\Gamma \vdash [e_1] \equiv [e_2] : [\tau]}\ \text{Eq-Cong-A-Val}$$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}{\Gamma \vdash [\tau_1 : \kappa] \equiv [\tau_2 : \kappa] : [= \tau_1 : \kappa]}\ \text{Eq-Cong-A-Typ}$$

$$\frac{\Gamma \vdash \Xi_1 \equiv \Xi_2 : \Omega}{\Gamma \vdash [\Xi_1] \equiv [\Xi_2] : [= \Xi_1]}\ \text{Eq-Cong-A-Sig}$$

3.1.4 (Generative) Elaboration

Signature:

$\boxed{\Gamma \vdash S \rightsquigarrow \Xi}$

$$\frac{\Gamma \vdash P : [= \Xi] \rightsquigarrow e}{\Gamma \vdash P \rightsquigarrow \Xi}\ \text{S-Path}$$

$$\frac{\Gamma \vdash D \rightsquigarrow \Xi}{\Gamma \vdash \{D\} \rightsquigarrow \Xi}\ \text{S-Struct}$$

$$\frac{\Gamma \vdash S_1 \rightsquigarrow \exists \overline{t : \kappa}.\, \Sigma \quad \Gamma, \overline{t : \kappa}, x_X : \Sigma \vdash S_2 \rightsquigarrow \Xi}{\Gamma \vdash (X : S_1) \to S_2 \rightsquigarrow \forall \overline{t : \kappa}.\, \Sigma \to \Xi}\ \text{S-Funct}$$

$$\frac{\Gamma \vdash S \rightsquigarrow \exists \overline{t_1 : \kappa_1}\ t : \kappa\ \overline{t_2 : \kappa_2}.\, \Sigma \quad \Sigma.\overline{l_X} = [= t : \kappa] \quad \Gamma \vdash T : \kappa \rightsquigarrow \tau}{\Gamma \vdash S \text{ where type } \overline{X} = T \rightsquigarrow \exists \overline{t_1 : \kappa_1}\ \overline{t_2 : \kappa_2}.\, \Sigma[t \leftarrow \tau]}\ \text{S-Where-Typ}$$

Declarations:

$\boxed{\Gamma \vdash D \rightsquigarrow \Xi}$

$$\frac{\Gamma \vdash T : \Omega \rightsquigarrow \tau}{\Gamma \vdash \mathrm{val}\, X : T \rightsquigarrow \{l_X : [\tau]\}}\ \text{D-Val}$$

$$\frac{\Gamma \vdash T : \kappa \rightsquigarrow \tau}{\Gamma \vdash \mathrm{type}\, X = T \rightsquigarrow \{l_X : [= \tau : \kappa]\}}\ \text{D-Typ-Eq}$$

$$\frac{\Gamma \vdash K \rightsquigarrow \kappa}{\Gamma \vdash \mathrm{type}\, X : K \rightsquigarrow \exists t : \kappa.\{l_X : [= t : \kappa]\}}\ \text{D-Typ}$$

$$\frac{\Gamma \vdash S \rightsquigarrow \exists \overline{t : \kappa}.\, \Sigma}{\Gamma \vdash \mathrm{module}\, X : S \rightsquigarrow \exists \overline{t : \kappa}.\{l_X : \Sigma\}}\ \text{D-Mod}$$

$$\frac{\Gamma \vdash S \rightsquigarrow \Xi}{\Gamma \vdash \mathrm{signature}\, X = S \rightsquigarrow \{l_X : [= \Xi]\}}\ \text{D-Sig-Eq}$$

$$\frac{\Gamma \vdash S \rightsquigarrow \exists \overline{t : \kappa}.\{\overline{l_X : \Sigma}\}}{\Gamma \vdash \mathrm{include}\, S \rightsquigarrow \exists \overline{t : \kappa}.\{\overline{l_X : \Sigma}\}}\ \text{D-Incl}$$

$$\frac{}{\Gamma \vdash \epsilon \rightsquigarrow \{\}}\ \text{D-Emt}$$

$$\frac{\{\overline{l_{X_1}}\} \cap \{\overline{l_{X_2}}\} = \emptyset \quad \Gamma \vdash D_1 \rightsquigarrow \exists \overline{t_1 : \kappa_1}.\{\overline{l_{X_1} : \Sigma_1}\} \quad \Gamma, \overline{t_1 : \kappa_1}, \overline{x_{X_1} : \Sigma_1} \vdash D_2 \rightsquigarrow \exists \overline{t_2 : \kappa_2}.\{\overline{l_{X_2} : \Sigma_2}\}}{\Gamma \vdash D_1; D_2 \rightsquigarrow \exists \overline{t_1 : \kappa_1}\ \overline{t_2 : \kappa_2}.\{\overline{l_{X_1} : \Sigma_1}\ \overline{l_{X_2} : \Sigma_2}\}}\ \text{D-Seq}$$

Matching:

$\boxed{\Gamma \vdash \Sigma_1 \leq \exists \overline{t : \kappa}.\Sigma_2 \uparrow \overline{\tau} \rightsquigarrow e}$

$$\frac{\Gamma \vdash \Sigma_1 \leq \Sigma_2[\overline{t \leftarrow \tau_t}] \rightsquigarrow e \quad \bigwedge_t \Gamma \vdash \tau_t : \kappa_t}{\Gamma \vdash \Sigma_1 \leq \exists \overline{t : \kappa_t}.\Sigma_2 \uparrow \overline{\tau_t} \rightsquigarrow e}\ \text{U-Match}$$

Subtyping:

$\boxed{\Gamma \vdash \Xi_1 \leq \Xi_2 \rightsquigarrow e}$

$$\frac{\Gamma \vdash \tau_1 \leq \tau_2 \rightsquigarrow e}{\Gamma \vdash [\tau_1] \leq [\tau_2] \rightsquigarrow \lambda x : [\tau_1].[e\ (x.\mathrm{val})]}\ \text{U-Val}$$

$$\frac{\Gamma \vdash \tau_1 \equiv \tau_2 : \kappa}{\Gamma \vdash [= \tau_1 : \kappa] \leq [= \tau_2 : \kappa] \rightsquigarrow \lambda x : [= \tau_1 : \kappa].x}\ \text{U-Typ}$$

$$\frac{\Gamma \vdash \Xi_1 \leq \Xi_2 \rightsquigarrow e_1 \quad \Gamma \vdash \Xi_2 \leq \Xi_1 \rightsquigarrow e_2}{\Gamma \vdash [= \Xi_1] \leq [= \Xi_2] \rightsquigarrow \lambda x : [= \Xi_1].[\Xi_2]}\ \text{U-Sig}$$

$$\frac{\bigwedge_l \Gamma \vdash \Sigma_{l1} \leq \Sigma_{l2} \rightsquigarrow e_l}{\Gamma \vdash \{\overline{l : \Sigma_{l1}}, \overline{l' : \Sigma'}\} \leq \{\overline{l : \Sigma_{l2}}\} \rightsquigarrow \lambda x : \{\overline{l : \Sigma_{l1}}, \overline{l' : \Sigma'}\}.\{\overline{l = e_l\ (x.l)}\}}\ \text{U-Struct}$$

$$\frac{\Gamma, \overline{t_2 : \kappa_2} \vdash \Sigma_2 \leq \exists \overline{t_1 : \kappa_1}.\Sigma_1 \uparrow \overline{\tau} \rightsquigarrow e_1 \quad \Gamma, \overline{t_2 : \kappa_2} \vdash \Xi_1[\overline{t_1 \leftarrow \tau}] \leq \Xi_2 \rightsquigarrow e_2}{\Gamma \vdash \forall \overline{t_1 : \kappa_1}.\Sigma_1 \to \Xi_1 \leq \forall \overline{t_2 : \kappa_2}.\Sigma_2 \to \Xi_2 \rightsquigarrow \begin{array}{l}\lambda x_1 : (\forall \overline{t_1 : \kappa_1}.\Sigma_1 \to \Xi_1).\\ \quad \lambda x_2 : \Sigma_2.e_2\ (x_1\ \overline{\tau}\ (e_1\ x_2))\end{array}}\ \text{U-Funct}$$

$$\frac{\Gamma, \overline{t_1 : \kappa_1} \vdash \Sigma_1 \leq \exists \overline{t_2 : \kappa_2}.\Sigma_2 \uparrow \overline{\tau} \rightsquigarrow e}{\Gamma \vdash \exists \overline{t_1 : \kappa_1}.\Sigma_1 \leq \exists \overline{t_2 : \kappa_2}.\Sigma_2 \rightsquigarrow \begin{array}{l}\lambda x_1 : (\exists \overline{t_1 : \kappa_1}.\Sigma_1).\\ \quad \mathrm{unpack}\langle \overline{t_1 : \kappa_1}, x_1' : \Sigma_1\rangle = x_1 \text{ in } \mathrm{pack}\langle \overline{\tau}, e\ x_1'\rangle_{\exists \overline{t_2 : \kappa_2}.\Sigma_2}\end{array}}\ \text{U-Abs}$$

Module:

$\boxed{\Gamma \vdash M : \Xi \rightsquigarrow e}$

$$\frac{\Gamma(x_X) = \Sigma}{\Gamma \vdash X : \Sigma \rightsquigarrow x_X}\ \text{M-Var}$$

$$\frac{\Gamma \vdash B : \Xi \rightsquigarrow e}{\Gamma \vdash \{B\} : \Xi \rightsquigarrow e}\ \text{M-Struct}$$

$$\frac{\Gamma \vdash M : \exists \overline{t : \kappa}.\{l_X : \Sigma, \overline{l_{X'} : \Sigma'}\} \rightsquigarrow e}{\Gamma \vdash M.X : \exists \overline{t : \kappa}.\Sigma \rightsquigarrow \mathrm{unpack}\langle \overline{t : \kappa}, x : \{l_X : \Sigma, \overline{l_{X'} : \Sigma'}\}\rangle = e \text{ in } \mathrm{pack}\langle \overline{t}, x.l_X\rangle_{\exists \overline{t : \kappa}.\Sigma}}\ \text{M-Dot}$$

$$\frac{\Sigma \vdash S \rightsquigarrow \exists \overline{t : \kappa}.\Sigma \quad \Gamma, \overline{t : \kappa}, x_X : \Sigma \vdash M : \Xi \rightsquigarrow e}{\Gamma \vdash \mathrm{fun}\, X : S \Rightarrow M : \forall \overline{t : \kappa}.\Sigma \to \Xi \rightsquigarrow \Lambda \overline{t : \kappa}.\lambda x_X : \Sigma.e}\ \text{M-Funct}$$

$$\frac{\Gamma(x_{X_1}) = \forall \overline{t : \kappa}.\Sigma' \to \Xi \quad \Gamma(x_{X_2}) = \Sigma \quad \Gamma \vdash \Sigma \leq \exists \overline{t : \kappa}.\Sigma' \uparrow \overline{\tau} \rightsquigarrow e}{\Gamma \vdash X_1\ X_2 : \Xi[\overline{t \leftarrow \tau}] \rightsquigarrow x_{X_1}\ \overline{\tau}\ (e\ x_{X_2})}\ \text{M-App}$$

$$\frac{\Gamma(x_X) = \Sigma \quad \Gamma \vdash S \rightsquigarrow \exists \overline{t : \kappa}.\Sigma' \quad \Gamma \vdash \Sigma \leq \exists \overline{t : \kappa}.\Sigma' \uparrow \overline{\tau} \rightsquigarrow e}{\Gamma \vdash X :> S : \exists \overline{t : \kappa}.\Sigma' \rightsquigarrow \mathrm{pack}\langle \overline{\tau}, e\ x_X\rangle_{\exists \overline{t : \kappa}.\Sigma'}}\ \text{M-Seal}$$

Bindings:

$\boxed{\Gamma \vdash B : \Xi \rightsquigarrow e}$

$$\frac{\Gamma \vdash E : \tau \rightsquigarrow e}{\Gamma \vdash \mathrm{val}\, X = E : \{l_X : [\tau]\} \rightsquigarrow \{l_X = e\}}\ \text{B-Val}$$

$$\frac{\Gamma \vdash T : \kappa \rightsquigarrow \tau}{\Gamma \vdash \mathsf{type}\, X = T : \{l_X : [= \tau : \kappa]\} \rightsquigarrow \{l_X = [\tau : \kappa]\}}\ \text{B-Typ}$$

$$\frac{\Gamma \vdash M : \exists \overline{t : \kappa}.\Sigma \rightsquigarrow e \quad \mathsf{NotAtomic}(\Sigma)}{\Gamma \vdash \mathsf{module}\, X = M : \exists \overline{t : \kappa}.\{l_X : \Sigma\} \rightsquigarrow \mathsf{unpack}\langle \overline{t : \kappa}, x : \Sigma\rangle = e \ \mathsf{in}\ \mathsf{pack}\langle \overline{t}, \{l_X = x\}\rangle_{\exists \overline{t : \kappa}.\{l_X : \Sigma\}}}\ \text{B-Mod}$$

$$\frac{\Gamma \vdash S \rightsquigarrow \Xi}{\Gamma \vdash \mathsf{signature}\, X = S : \{l_X : [= \Xi]\} \rightsquigarrow \{l_X = [\Xi]\}}\ \text{B-Sig}$$

$$\frac{\Gamma \vdash M : \exists \overline{t : \kappa}.\{\overline{l_X : \Sigma}\} \rightsquigarrow e}{\Gamma \vdash \mathsf{include}\, M : \exists \overline{t : \kappa}.\{\overline{l_X : \Sigma}\} \rightsquigarrow e}\ \text{B-Incl}$$

$$\frac{}{\Gamma \vdash \epsilon : \{\} \rightsquigarrow \{\}}\ \text{B-Emt}$$

$$\frac{\begin{array}{ll} \overline{l'_{X_1}} = \overline{l_{X_1}} \setminus \overline{l_{X_2}} \quad \overline{l'_{X_1} : \Sigma'_1} \subseteq \overline{l_{X_1} : \Sigma_1} & \Gamma \vdash B_1 : \exists \overline{t_1 : \kappa_1}.\{\overline{l_{X_1} : \Sigma_1}\} \rightsquigarrow e_1 \\ \Sigma = \{\overline{l'_{X_1} : \Sigma'_1}, \overline{l_{X_2} : \Sigma_2}\} & \Gamma, \overline{t_1 : \kappa_1}, \overline{x_{X_1} : \Sigma_1} \vdash B_2 : \exists \overline{t_2 : \kappa_2}.\{\overline{l_{X_2} : \Sigma_2}\} \rightsquigarrow e_2 \end{array}}{\Gamma \vdash B_1 ; B_2 : \exists \overline{t_1 : \kappa_1}\, \overline{t_2 : \kappa_2}.\Sigma \rightsquigarrow \begin{array}{l} \mathsf{unpack}\langle \overline{t_1 : \kappa_1}, x_1\rangle = e_1 \ \mathsf{in} \\ \quad \mathsf{unpack}\langle \overline{t_2 : \kappa_2}, x_2\rangle = (\mathsf{let}\ \overline{x_{X_1} : \Sigma_1 = x_1.l_{X_1}}\ \mathsf{in}\ e_2)\ \mathsf{in} \\ \quad\quad \mathsf{pack}\langle \overline{t_1}\, \overline{t_2}, \{\overline{l'_{X_1} = x_1.l'_{X_1}}, \overline{l_{X_2} = x_2.l_{X_2}}\}\rangle_{\exists \overline{t_1 : \kappa_1}\, \overline{t_2 : \kappa_2}.\Sigma} \end{array}}\ \text{B-Seq}$$

Path:

$\boxed{\Gamma \vdash P : \Sigma \rightsquigarrow e}$

$$\frac{\Gamma \vdash P : \exists \overline{t : \kappa}.\Sigma \quad \Gamma \vdash \Sigma : \Omega}{\Gamma \vdash P : \Sigma \rightsquigarrow \mathsf{unpack}\langle \overline{t : \kappa}, x\rangle = e \ \mathsf{in}\ x}\ \text{P-Mod}$$

$\boxed{\Gamma \vdash T : \kappa \rightsquigarrow \tau}$

$$\frac{\Gamma \vdash P : [= \tau : \kappa] \rightsquigarrow e}{\Gamma \vdash P : \kappa \rightsquigarrow \tau}\ \text{T-Elab-Path}$$

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

$$\frac{\Gamma \vdash P : [\tau] \rightsquigarrow e}{\Gamma \vdash P : \tau \rightsquigarrow e.\mathsf{val}}\ \text{E-Path}$$

3.1.5 Modules as First-Class Values

$$\begin{aligned} T &::= \cdots \mid \mathsf{pack}\, S \\ E &::= \cdots \mid \mathsf{pack}\, M : S \\ M &::= \cdots \mid \mathsf{unpack}\, E : S \end{aligned}$$

Rootedness:

$\boxed{t : \kappa \ \text{rooted in}\ \Sigma \ \text{at}\ \overline{l_X}}$

$$\frac{t = \tau'}{t : \kappa \ \text{rooted in}\ [= \tau : \kappa] \ \text{at}\ \epsilon} \qquad \frac{t : \kappa \ \text{rooted in}\ \{\overline{l_X : \Sigma}\}.l \ \text{at}\ \overline{l'}}{t : \kappa \ \text{rooted in}\ \{\overline{l_X : \Sigma}\} \ \text{at}\ l\, \overline{l'}}$$

Rooted ordering:

$$t_1 : \kappa_1 \leq_\Sigma t_2 : \kappa_2 \iff \min\{\overline{l} \mid t_1 : \kappa_1 \ \text{rooted in}\ \Sigma \ \text{at}\ \overline{l}\} \leq \min\{\overline{l} \mid t_2 : \kappa_2 \ \text{rooted in}\ \Sigma \ \text{at}\ \overline{l}\}$$

Signature normalization:

$$\frac{\mathrm{norm}_0(\tau) = \tau'}{\mathrm{norm}([\tau]) = [\tau']}$$

$$\frac{}{\mathrm{norm}([= \tau : \kappa]) = [= \tau : \kappa]}$$

$$\frac{\mathrm{norm}(\Xi) = \Xi'}{\mathrm{norm}([= \Xi]) = [= \Xi']}$$

$$\frac{\bigwedge_X \mathrm{norm}(\Sigma_X) = \Sigma'_X}{\mathrm{norm}(\{\overline{l_X : \Sigma_X}\}) = \{\overline{l_X : \Sigma'_X}\}}$$

$$\frac{\mathrm{sort}_{\leq_{\Sigma'}}(\overline{t : \kappa}) = \overline{t' : \kappa'} \quad \mathrm{norm}(\Sigma) = \Sigma' \quad \mathrm{norm}(\Xi) = \Xi'}{\mathrm{norm}(\forall \overline{t : \kappa}. \Sigma \to \Xi) = \forall \overline{t' : \kappa'}. \Sigma' \to \Xi'}$$

$$\frac{\mathrm{sort}_{\leq_{\Sigma'}}(\overline{t : \kappa}) = \overline{t' : \kappa'} \quad \mathrm{norm}(\Sigma) = \Sigma'}{\mathrm{norm}(\exists \overline{t : \kappa}. \Sigma) = \exists \overline{t' : \kappa'}. \Sigma'}$$

Type:

$\boxed{\Gamma \vdash T : \kappa \rightsquigarrow \tau}$

$$\frac{\Gamma \vdash S \rightsquigarrow \Xi}{\Gamma \vdash \mathrm{pack}\, S : \Omega \rightsquigarrow \mathrm{norm}(\Xi)} \text{ T-Pack}$$

Expression:

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

$$\frac{\Gamma \vdash S \rightsquigarrow \Xi \quad \Gamma \vdash \Xi' \leq \mathrm{norm}(\Xi) \rightsquigarrow e_1 \quad \Gamma \vdash M : \Xi' \rightsquigarrow e_2}{\Gamma \vdash (\mathrm{pack}\, M : S) : \mathrm{norm}(\Xi) \rightsquigarrow e_1\ e_2} \text{ E-Pack}$$

Module:

$\boxed{\Gamma \vdash M : \Xi \rightsquigarrow e}$

$$\frac{\Gamma \vdash S \rightsquigarrow \Xi \quad \Gamma \vdash E : \mathrm{norm}(\Xi) \rightsquigarrow e}{\Gamma \vdash (\mathrm{unpack}\, E : S) : \mathrm{norm}(\Xi) \rightsquigarrow e} \text{ M-Unpack}$$

3.1.6 Elaboration with Applicative Functor

$$\begin{aligned} S ::= {} & \cdots \\ \mid {} & (X : S) \Rightarrow S && \text{(applicative functor signature)} \end{aligned}$$

$$\begin{aligned} \varphi ::= {} & \mathrm{I} && \text{(impure effect)} \\ \mid {} & \mathrm{P} && \text{(pure effect)} \\ \Sigma ::= {} & \cdots \\ \mid {} & \{\overline{l_X : \Sigma}\} \\ \mid {} & \forall \overline{t : \kappa}. \Sigma \to_{\mathrm{I}} \Xi && \text{(generative functor signature)} \\ \mid {} & \forall \overline{t : \kappa}. \Sigma \to_{\mathrm{P}} \Sigma && \text{(applicative functor signature)} \end{aligned}$$

Abbreviation:

$$\tau_1 \to_\varphi \tau_2 \stackrel{\text{def}}{=} \tau_1 \to \{l_\varphi : \tau_2\}$$

$$\lambda_\varphi x : \tau.e \stackrel{\text{def}}{=} \lambda x : \tau.\{l_\varphi = e\}$$

$$(e_1\ e_2)_\varphi \stackrel{\text{def}}{=} (e_1\ e_2).l_\varphi$$

$$\Gamma^\varphi \stackrel{\text{def}}{=} \begin{cases} \cdot & (\varphi = \mathrm{I}) \\ \Gamma & (\varphi = \mathrm{P}) \end{cases}$$

$$tyenv(\Gamma) \stackrel{\text{def}}{=} \begin{cases} tyenv(\Gamma')\ t : \kappa & (\Gamma = \Gamma', t : \kappa) \\ tyenv(\Gamma') & (\Gamma = \Gamma', x : \tau) \\ \epsilon & (\Gamma = \cdot) \end{cases}$$

$$\forall_{\mathrm{P}}\Gamma.\tau_0 \stackrel{\text{def}}{=} \begin{cases} \forall_{\mathrm{P}}\Gamma'.\forall t : \kappa.\tau_0 & (\Gamma = \Gamma', t : \kappa) \\ \forall_{\mathrm{P}}\Gamma'.\tau \to_{\mathrm{P}} \tau_0 & (\Gamma = \Gamma', x : \tau) \\ \tau_0 & (\Gamma = \cdot) \end{cases}$$

$$\Lambda_{\mathrm{P}}\Gamma.e \stackrel{\text{def}}{=} \begin{cases} \Lambda_{\mathrm{P}}\Gamma'.\Lambda t : \kappa.e & (\Gamma = \Gamma', t : \kappa) \\ \Lambda_{\mathrm{P}}\Gamma'.\lambda_{\mathrm{P}} x : \tau.e & (\Gamma = \Gamma', x : \tau) \\ e & (\Gamma = \cdot) \end{cases}$$

$$(e\ \Gamma)_{\mathrm{P}} \stackrel{\text{def}}{=} \begin{cases} (e\ \Gamma')_{\mathrm{P}}\ t & (\Gamma = \Gamma', t : \kappa) \\ ((e\ \Gamma')_{\mathrm{P}}\ x)_{\mathrm{P}} & (\Gamma = \Gamma', x : \tau) \\ e & (\Gamma = \cdot) \end{cases}$$

Effect combining:

$\boxed{\varphi_1 \vee \varphi_2 = \varphi}$

$$\frac{}{\varphi \vee \varphi = \varphi} \qquad \frac{}{\mathrm{I} \vee \mathrm{P} = \mathrm{I}} \qquad \frac{}{\mathrm{P} \vee \mathrm{I} = \mathrm{I}}$$

Subeffects:

$\boxed{\varphi_1 \leq \varphi_2}$

$$\frac{}{\varphi \leq \varphi}\ \text{F-Refl} \qquad \frac{}{\mathrm{P} \leq \mathrm{I}}\ \text{F-Sub}$$

Signature:

$\boxed{\Gamma \vdash S \rightsquigarrow \Xi}$

$$\frac{\Gamma \vdash S_1 \rightsquigarrow \exists \overline{t_1 : \kappa_1}.\Sigma \quad \Gamma, \overline{t_1 : \kappa_1}, x_X : \Sigma \vdash S_2 \rightsquigarrow \Xi}{\Gamma \vdash (X : S_1) \to S_2 \rightsquigarrow \forall \overline{t_1 : \kappa_1}.\Sigma \to_{\mathrm{I}} \Xi}\ \text{S-Funct-I}$$

$$\frac{\Gamma \vdash S_1 \rightsquigarrow \exists \overline{t_1 : \kappa_1}.\Sigma_1 \quad \Gamma, \overline{t_1 : \kappa_1}, x_X : \Sigma_1 \vdash S_2 \rightsquigarrow \exists \overline{t_2 : \kappa_2}.\Sigma_2}{\Gamma \vdash (X : S_1) \Rightarrow S_2 \rightsquigarrow \exists \overline{t_2' : \overline{\kappa_1} \to \kappa_2}.\forall \overline{t_1 : \kappa_1}.\Sigma_1 \to_{\mathrm{P}} \Sigma_2[\overline{t_2 \leftarrow t_2'\ \overline{t_1}}]}\ \text{S-Funct-P}$$

Subtyping:

$\boxed{\Gamma \vdash \Xi_1 \leq \Xi_2 \rightsquigarrow e}$

$$\frac{\Gamma, \overline{t_2 : \kappa_2} \vdash \Sigma_2 \leq \exists \overline{t_1 : \kappa_1}.\Sigma_1 \uparrow \overline{\tau} \rightsquigarrow e_1 \quad \Gamma, \overline{t_2 : \kappa_2} \vdash \Xi_1[\overline{t_1 \leftarrow \tau}] \leq \Xi_2 \rightsquigarrow e_2 \quad \varphi_1 \leq \varphi_2}{\Gamma \vdash (\forall \overline{t_1 : \kappa_1}.\Sigma_1 \to_{\varphi_1} \Xi_1) \leq (\forall \overline{t_2 : \kappa_2}.\Sigma_2 \to_{\varphi_2} \Xi_2) \rightsquigarrow \begin{array}{l} \lambda x_1 : (\forall \overline{t_1 : \kappa_1}.\Sigma_1 \to_{\varphi_1} \Xi_1). \\ \quad \Lambda \overline{t_2 : \kappa_2}.\lambda_{\varphi_2} x_2 : \Sigma_2.e_2\ (x_1\ \overline{\tau}\ (e_1\ x_2))_{\varphi_1} \end{array}}\ \text{U-Funct}$$

Module:

$\boxed{\Gamma \vdash M :_\varphi \Xi \rightsquigarrow e}$

$$\frac{\Gamma(x_X) = \Sigma}{\Gamma \vdash X :_{\mathrm{P}} \Sigma \rightsquigarrow \Lambda_{\mathrm{P}}\Gamma.x_X}\ \text{M-Var}$$

$$\frac{\Gamma \vdash B :_{\varphi} \Xi \rightsquigarrow e}{\Gamma \vdash \{B\} :_{\varphi} \Xi \rightsquigarrow e}\ \text{M-Struct}$$

$$\frac{\Gamma \vdash M :_{\varphi} \exists\overline{t : \kappa}.\{l_X : \Sigma, \overline{l_{X'} : \Sigma'}\} \rightsquigarrow e}{\Gamma \vdash M.X :_{\varphi} \exists\overline{t : \kappa}.\Sigma \rightsquigarrow \mathrm{unpack}\langle\overline{t : \kappa}, x\rangle = e \text{ in } \mathrm{pack}\langle\bar{t}, \Lambda_{\mathrm{P}}\Gamma^{\varphi}.(x\ \Gamma^{\varphi})_{\mathrm{P}}.l_X\rangle}\ \text{M-Dot}$$

$$\frac{\Sigma \vdash S \rightsquigarrow \exists\overline{t : \kappa}.\Sigma \quad \Gamma, \overline{t : \kappa}, x_X : \Sigma \vdash M :_{\mathrm{I}} \Xi \rightsquigarrow e}{\Gamma \vdash \mathrm{fun}\, X : S \Rightarrow M :_{\mathrm{P}} \forall\overline{t : \kappa}.\Sigma \rightarrow_{\mathrm{I}} \Xi \rightsquigarrow \Lambda_{\mathrm{P}}\Gamma.\Lambda\overline{t : \kappa}.\lambda_{\mathrm{I}} x_X : \Sigma.\, e}\ \text{M-Funct-I}$$

$$\frac{\Sigma \vdash S \rightsquigarrow \exists\overline{t : \kappa}.\Sigma \quad \Gamma, \overline{t : \kappa}, x_X : \Sigma \vdash M :_{\mathrm{P}} \exists\overline{t_2 : \kappa_2}.\Sigma_2 \rightsquigarrow e}{\Gamma \vdash \mathrm{fun}\, X : S \Rightarrow M :_{\mathrm{P}} \exists\overline{t_2 : \kappa_2}.\forall\overline{t : \kappa}.\Sigma \rightarrow_{\mathrm{P}} \Sigma_2 \rightsquigarrow e}\ \text{M-Funct-P}$$

$$\frac{\Gamma(x_{X_1}) = \forall\overline{t : \kappa}.\Sigma' \rightarrow_{\varphi} \Xi \quad \Gamma(x_{X_2}) = \Sigma \quad \Gamma \vdash \Sigma \le \exists\overline{t : \kappa}.\Sigma' \uparrow \bar{\tau} \rightsquigarrow e}{\Gamma \vdash X_1\ X_2 :_{\varphi} \Xi[\overline{t \leftarrow \tau}] \rightsquigarrow \Lambda_{\mathrm{P}}\Gamma^{\varphi}.(x_{X_1}\ \bar{\tau}\ (e\ x_{X_2}))_{\varphi}}\ \text{M-App}$$

$$\frac{\overline{t_\Gamma : \kappa_\Gamma} = \mathit{tyenv}(\Gamma) \quad \Gamma(x_X) = \Sigma \quad \Gamma \vdash S \rightsquigarrow \exists\overline{t : \kappa}.\Sigma' \quad \Gamma \vdash \Sigma \le \exists\overline{t : \kappa}.\Sigma' \uparrow \bar{\tau} \rightsquigarrow e}{\Gamma \vdash X :> S :_{\mathrm{P}} \exists\overline{t' : \overline{t_\Gamma : \kappa_\Gamma} \rightarrow \kappa}.\Sigma'[\overline{t \leftarrow t'\ \overline{t_\Gamma}}] \rightsquigarrow \mathrm{pack}\langle\overline{\lambda\overline{t_\Gamma : \kappa_\Gamma}.\tau}, \Lambda_{\mathrm{P}}\Gamma.e\ x_X\rangle}\ \text{M-Seal}$$

定理 26 (Typing for module elaboration).

- $\Gamma \vdash M :_{\mathrm{I}} \Xi \rightsquigarrow e$ ならば, $\Gamma \vdash e : \Xi$.
- $\Gamma \vdash M :_{\mathrm{P}} \exists\overline{t : \kappa}.\Sigma \rightsquigarrow e$ ならば, $\cdot \vdash e : \exists\overline{t : \kappa}.\forall_{\mathrm{P}}\Gamma.\Sigma$.

□

Bindings:

$\boxed{\Gamma \vdash B :_{\varphi} \Xi \rightsquigarrow e}$

$$\frac{\Gamma \vdash E : \tau \rightsquigarrow e}{\Gamma \vdash \mathrm{val}\, X = E :_{\mathrm{P}} \{l_X : [\tau]\} \rightsquigarrow \Lambda_{\mathrm{P}}\Gamma.\{l_X = e\}}\ \text{B-Val}$$

$$\frac{\Gamma \vdash T : \kappa \rightsquigarrow \tau}{\Gamma \vdash \mathrm{type}\, X = T :_{\mathrm{P}} \{l_X : [= \tau : \kappa]\} \rightsquigarrow \Lambda_{\mathrm{P}}\Gamma.\{l_X = [\tau : \kappa]\}}\ \text{B-Typ}$$

$$\frac{\Gamma \vdash M :_{\varphi} \exists\overline{t : \kappa}.\Sigma \rightsquigarrow e \quad \mathrm{NotAtomic}(\Sigma)}{\Gamma \vdash \mathrm{module}\, X = M :_{\varphi} \exists\overline{t : \kappa}.\{l_X : \Sigma\} \rightsquigarrow \mathrm{unpack}\langle\overline{t : \kappa}, x\rangle = e \text{ in } \mathrm{pack}\langle\bar{t}, \Lambda_{\mathrm{P}}\Gamma^{\varphi}.\{l_X = x\ \Gamma^{\varphi}\}\rangle}\ \text{B-Mod}$$

$$\frac{\Gamma \vdash S \rightsquigarrow \Xi}{\Gamma \vdash \mathrm{signature}\, X = S :_{\mathrm{P}} \{l_X : [= \Xi]\} \rightsquigarrow \Lambda_{\mathrm{P}}\Gamma.\{l_X = [\Xi]\}}\ \text{B-Sig}$$

$$\frac{\Gamma \vdash M :_{\varphi} \exists\overline{t : \kappa}.\{\overline{l_X : \Sigma}\} \rightsquigarrow e}{\Gamma \vdash \mathrm{include}\, M :_{\varphi} \exists\overline{t : \kappa}.\{\overline{l_X : \Sigma}\} \rightsquigarrow e}\ \text{B-Incl}$$

$$\frac{}{\Gamma \vdash \epsilon :_{\mathrm{P}} \{\} \rightsquigarrow \Lambda_{\mathrm{P}}\Gamma.\{\}}\ \text{B-Emt}$$

$$\frac{\begin{array}{ll} \overline{l'_{X_1}} = \overline{l_{X_1}} \setminus \overline{l_{X_2}} \quad \overline{l'_{X_1} : \Sigma'_1} \subseteq \overline{l_{X_1} : \Sigma_1} & \Gamma \vdash B_1 :_{\varphi_1} \exists\overline{t_1 : \kappa_1}.\{\overline{l_{X_1} : \Sigma_1}\} \rightsquigarrow e_1 \\ \Sigma = \{\overline{l'_{X_1} : \Sigma'_1}, \overline{l_{X_2} : \Sigma_2}\} & \Gamma, \overline{t_1 : \kappa_1}, \overline{x_{X_1} : \Sigma_1} \vdash B_2 :_{\varphi_2} \exists\overline{t_2 : \kappa_2}.\{\overline{l_{X_2} : \Sigma_2}\} \rightsquigarrow e_2 \end{array}}{\begin{array}{l} \Gamma \vdash B_1; B_2 :_{\varphi_1 \vee \varphi_2} \exists\overline{t_1 : \kappa_1}\ \overline{t_2 : \kappa_2}.\Sigma \\ \quad \rightsquigarrow \mathrm{unpack}\langle\overline{t_1 : \kappa_1}, x_1\rangle = e_1 \text{ in} \\ \qquad \mathrm{unpack}\langle\overline{t_2 : \kappa_2}, x_2\rangle = (\mathrm{let}\, \overline{x_{X_1} = \Lambda_{\mathrm{P}}\Gamma^{\varphi_1 \vee \varphi_2}.(x_1\ \Gamma^{\varphi_1})_{\mathrm{P}}.l_{X_1}} \text{ in } e_2) \text{ in} \\ \qquad \mathrm{pack}\langle\overline{t_1}\ \overline{t_2}, \Lambda_{\mathrm{P}}\Gamma^{\varphi_1 \vee \varphi_2}.\, \mathrm{let}\, \overline{x_{X_1} = (x_1\ \Gamma^{\varphi_1})_{\mathrm{P}}.l_{X_1}} \text{ in} \\ \qquad\quad \{\overline{l'_{X_1} = (x_1\ \Gamma^{\varphi_1})_{\mathrm{P}}.l'_{X_1}}, \overline{l_{X_2} = (x_2\ (\Gamma, \overline{t_1 : \kappa_1}, \overline{x_{X_1} : \Sigma_1})^{\varphi_2})_{\mathrm{P}}.l_{X_2}}\}\rangle \end{array}}\ \text{B-Seq}$$

Path:

$\boxed{\Gamma \vdash P : \Sigma \rightsquigarrow e}$

$$\frac{\Gamma \vdash P :_{\varphi} \exists \overline{t : \kappa}. \Sigma \quad \Gamma \vdash \Sigma : \Omega}{\Gamma \vdash P : \Sigma \rightsquigarrow \mathrm{unpack}\langle \overline{t : \kappa}, x \rangle = e \ \mathrm{in} \ (x \ \Gamma^{\varphi})_{\mathrm{P}}} \ \text{P-Mod}$$

第 4 章

Control Operators

第 5 章

Type Checking and Inference

第 6 章

Static Memory Management and Regions

第 7 章

Dynamic Memory Management and Gabage Collection

第 8 章

I/O Management and Concurrency

第 9 章

Code Generation and Virtual Machines

第 10 章

Program Stability and Compatibility

第 11 章

Program Separation and Linking

第 12 章

Syntax and Parsing

第 13 章

Analysis and Optimizations

第 14 章

Meta-Programming and Multi-Stage Programming

第 15 章

Generic Programming

第 16 章

Advanced Calculus

参考文献

[GTL89] Jean-Yves Girard, Paul Taylor, and Yves Lafont, *Proofs and Types*, Cambridge University Press, apr 1989.

[RRD14] Andreas Rossberg, Claudio Russo, and Derek Dreyer, *F-ing modules*, Journal of Functional Programming **24** (2014), no. 5, 529–607.