Judgments

- **Well-formed Type (Type Formation)** $\Gamma \vdash A$ type
- **Judgmentally Equal Types** $\Gamma \vdash A \doteq B$ type
- **Well-formed Term** $\Gamma \vdash a : A$
- **Judgmentally Equal Terms** $\Gamma \vdash a \doteq b : A$

Context (Telescope)

$x_1 : A_1, \ldots, x_{k-1} : A_{k-1}(x_1, \ldots, x_{k-2}) \vdash A_{k(x_1, \ldots, x_{k-1})}$ type

Dependent Stuff

- **type family (dependent types)** $\Gamma, x : A \vdash B(x)$ type
- **section (dependent terms)** $\Gamma, x : A \vdash b(x) : B(x)$

NOTE Why is it called "section"? Category warning!

TODO

Rules

- **Structural Rules**
  1. Equivalence, as you expect
  2. Variable Conversion

$$\frac{\Gamma \vdash A \doteq A' \text{ type} \qquad \Gamma, x : A, \Delta \vdash \mathcal{J}\ [B(x)]}{\Gamma, x : A', \Delta \vdash \mathcal{J}[B(x)]}$$

NOTE the following rule (*element conversion*) is derivable, as we'll see later

$$\frac{\Gamma \vdash A \doteq A' \text{ type} \qquad \Gamma \vdash a : A}{\Gamma \vdash a : A'}$$

  3. Substitution

$$\frac{\Gamma \vdash a : A \qquad \Gamma, x : A, \Delta \vdash \mathcal{J}}{\Gamma, \Delta[a/x] \vdash \mathcal{J}[a/x]}\,\text{Subst}$$

This $\mathcal{J}$ here stands for whatever judgment we are talking about. It could be *type formation, judgmental equal terms/types*, or *typing judgments*. In real life there will be 4 rules here.

Also note how Δ is getting substituted because it can depends on x.

NOTE Usually we want Subst to be an admissible rule. For today, we see it as an axiom.

  4. Weakening

$$\frac{\Gamma \vdash A \text{ type} \qquad \Gamma, \Delta \vdash \mathcal{J}}{\Gamma, x : A, \Delta \vdash \mathcal{J}}\,W$$

Note how we are adding x to Γ.

  5. Contraction, which is derivable from how we defined substitution.

TODO I think defining substitution as an axiom is not a good idea, especially when it mixes substitution and structural properties up. My intuition is that structural properties are a direct consequence of how you define substitution, which in turn is the very core definition of the semantics of your type theory.

  6. Generic element (variable rule)

$$\frac{\Gamma \vdash A \text{ type}}{\Gamma, x : A \vdash x : A}\,\delta$$

Derivation

A finite tree where

- **nodes**: inferences
- **root**: conclusion
- **leaves**: hypotheses

E.g.

Thm (*element conversion*):

$$\frac{\Gamma \vdash A \doteq A' \text{ type} \qquad \Gamma \vdash a : A}{\Gamma \vdash a : A'}$$

Proof.

$$\dfrac{\Gamma \vdash a : A \qquad \dfrac{\dfrac{\Gamma \vdash A \doteq A' \text{ type}}{\Gamma \vdash A' \doteq A \text{ type}} \doteq \text{Sym} \qquad \dfrac{\Gamma \vdash A' \text{ type}}{\Gamma, x : A' \vdash x : A'} x}{\Gamma, x : A \vdash x : A'} \text{VarConv}}{\Gamma \vdash a : A'} \text{Subst}$$

□

Types

We define types by giving its formation (F), congruence ($=$), introduction (I), elimination (E), and computation (β/η) rules.

Pi (Dependent function)

$\Pi_{x:A} B(x)$

$$\frac{\Gamma, x : A \vdash B(x) \text{ type}}{\Gamma \vdash \Pi_{x:A} B(x) \text{ type}} \Pi_F \qquad \frac{\Gamma \vdash A \doteq A' \qquad \Gamma, x : A \vdash B(x) \doteq B'(x)}{\Gamma \vdash \Pi_{x:A} B(x) \doteq \Pi_{x:A'} B'(x) \text{ type}} \Pi_{\text{eq}}$$

$$\frac{\Gamma, x : A \vdash b(x) : B(x)}{\Gamma \vdash \lambda x.b(x) : \Pi_{x:A} B(x)} \Pi_I \qquad \frac{\Gamma \vdash f : \Pi_{x:A} B(x)}{\Gamma, x : A \vdash f\ x : B(x)} \Pi_E$$

$$\frac{\Gamma, x : A \vdash b(x) : B(x)}{\Gamma, x : A \vdash (\lambda y.b(y))\ x \doteq b(x) : B(x)} \Pi_\beta \qquad \frac{\Gamma \vdash f : \Pi_{x:A} B(x)}{\Gamma \vdash \lambda x.f\ x \doteq f : \Pi_{x:A} B(x)} \Pi_\eta$$

NOTE $x.b(x)$ (binding) is a general structural construct, while λ applies to a binding and it is MLTT-specific. It allows a form of "local" substitution.

NOTE Π_η is normally derivable from Extensionality.

$$\frac{\Gamma \vdash f : \Pi_{x:A} B(x) \qquad \Gamma \vdash g : \Pi_{x:A} B(x) \qquad \Gamma, x : A \vdash f(x) \doteq g(x) : B(x)}{\Gamma \vdash f \doteq g : \Pi_{x:a} B(x)} \text{Extensionality}$$

It's convenient to have Π_η rule in terms of computation but normally when we study metatheory we want extensionality.

Non-dependent function

A special const case of Π where codomain does not have dependency on the domain.

$$\dfrac{\dfrac{\dfrac{\Gamma \vdash A \text{ type} \qquad \Gamma \vdash B \text{ type}}{\Gamma, x : A \vdash B \text{ type}}\,\text{Weaken}}{\Gamma \vdash \Pi_{x:A} B \text{ type}}\,\Pi_F}{\Gamma \vdash A \to B \coloneqq \Pi_{x:A} B \text{ type}}\,{\to_F}$$

Notice how B does not depend on x.

E.g.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, f : A \to B \vdash f : A \to B}\,x}{\Gamma, f : A \to B, x : A \vdash f\ x : B}\,\Pi_E}{\Gamma, g : B \to C, f : A \to B, x : A \vdash f\ x : B}\,\text{Weaken} \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, g : B \to C \vdash g : B \to C}\,x}{\Gamma, g : B \to C, y : B \vdash g\ y : C}\,\Pi_E}{\Gamma, g : B \to C, f : A \to B, x : A, y : B \vdash g\ y : C}\,\text{Weaken}}{\Gamma, g : B \to C, f : A \to B, x : A \vdash g\ (f\ x) : C}\,\text{Subst}}{\Gamma, g : B \to C, f : A \to B \vdash \lambda x.g\ (f\ x) : A \to C}\,\Pi_I}{\Gamma, g : B \to C \vdash \lambda f.\lambda x.g\ (f\ x) : (A \to B) \to (A \to C)}\,\Pi_I}{\Gamma \vdash \lambda g.\lambda f.\lambda x.g\ (f\ x) : (B \to C) \to (A \to B) \to (A \to C)}\,\Pi_I$$

$\mathbb{N}$ Natural Numbers

$$\dfrac{}{\vdash \mathbb{N} \text{ type}}\,\mathbb{N}_F \qquad \dfrac{}{\vdash 0_{\mathbb{N}} : \mathbb{N}}\,\mathbb{N}_{I_0} \qquad \dfrac{}{\vdash S_{\mathbb{N}} : \mathbb{N} \to \mathbb{N}}\,\mathbb{N}_{I_S}$$

$$\dfrac{\Gamma, n : \mathbb{N} \vdash P(n) \text{ type} \qquad \Gamma \vdash p_0 : P(0_{\mathbb{N}}) \qquad \Gamma \vdash p_S : \Pi_{n:\mathbb{N}} P(n) \to P(S_{\mathbb{N}}\ n)}{\Gamma \vdash \text{ind}_{\mathbb{N}}(p_0, p_S) : \Pi_{n:\mathbb{N}} P(n)}\,\mathbb{N}_E$$

$$\dfrac{\dots}{\Gamma \vdash \text{ind}_{\mathbb{N}}(p_0, p_S, 0_{\mathbb{N}}) \doteq p_0 : P(0_{\mathbb{N}})}\,\mathbb{N}_{\beta_0}$$

$$\dfrac{\dots}{\Gamma, n : \mathbb{N} \vdash \text{ind}_{\mathbb{N}}(p_0, p_S, S_{\mathbb{N}}\ n) \doteq \text{ind}_{\mathbb{N}}(p_0, p_S, n) : P(S_{\mathbb{N}}\ n)}\,\mathbb{N}_{\beta_S}$$

Example,

```haskell
addN: N -> N -> N
addN m 0 = m
addN m (S n) = S (addN m n)
```

$$\dfrac{m : \mathbb{N} \vdash \text{add}_{\mathbb{N}}0 \coloneqq m : \mathbb{N} \qquad m : \mathbb{N}, \text{add}_{\mathbb{N}}S : \mathbb{N} \to (\mathbb{N} \to \mathbb{N}) \vdash \text{add}_{\mathbb{N}}S \coloneqq \text{TODO} : \mathbb{N} \to (\mathbb{N} \to \mathbb{N})}{m : \mathbb{N} \vdash \text{add}_{\mathbb{N}}\ m \coloneqq \text{ind}_{\mathbb{N}}(\text{add}_{\mathbb{N}}0\ m, \text{add}_{\mathbb{N}}S\ m) : \mathbb{N} \to \mathbb{N}}$$

TODO

1

$$\dfrac{}{\Gamma \vdash 1 \text{ type}}\,1_F \qquad \dfrac{}{\Gamma \vdash \star : 1}\,1_I$$

NOTE No elimination rule

0

$$\dfrac{}{\Gamma \vdash 0 \text{ type}}\,0_F \qquad \dfrac{}{\Gamma \vdash \text{ind}_0 : \Pi_{x:0} P(x)}\,0_E$$

or, the non-dependent version, $\text{ind}'_0 : 0 \to P$

NOTE $\neg P := P \to 0$

Sigma (dependent sum/pair)

$$\frac{\Gamma, x : A \vdash B(x) \text{ type}}{\Gamma \vdash \Sigma_{x:A} B(x) \text{ type}} \Sigma_F \qquad \frac{}{\Gamma \vdash \text{pair}}$$

TODO

Identity

$$\frac{\Gamma \vdash a : A \quad \Gamma \vdash b : A}{\Gamma \vdash a =_A b \text{ type}} =_F \qquad \frac{\Gamma \vdash a : A}{\Gamma \vdash \text{refl}_a : a =_A a} =_I$$

$$\frac{\Gamma, x : A, y : A, p : x = y \vdash P(x, y, p) \text{ type}}{\Gamma \vdash J : \Pi_{x:A} P(x, x, \text{refl}_x) \to \Pi_{x:A} \Pi_{y:A} \Pi_{g:x =_A y} P(x, y, g)} =_E$$

$$\frac{\Gamma, x : A, y : A, p : x =_A y \vdash P(x, y, p) \text{ type} \quad \Gamma \vdash d : \Pi_{x:A} P(x, x, \text{refl}_x)}{\Gamma \vdash J(d, a, a, \text{refl}_a) \doteq d(a) : P(a, a, \text{refl}_a)} =_{\text{compute}}$$

So morally, J rule is converting a proof on two *judgmentally equal* things to two *definitionally equal* things, by eliminating the `refl` constructor that smuggles a *judgmental equality* inside a *definitional equality*.

NOTE Notice that by its computation rule, J is not computable if the proof is not `refl`.

Example.

Thm (Transitivity). TODO

TODO Talk about how PAs hide J rule away from users, and show two different representations (J rule vs pm)

Thm (Symmetry).

Thm (*Ap*). If $a =_A b$ and $f : A \to B$, then $f(a) =_B f(b)$.

Thm (*Transport*). If $a =_A b$ and $B(a)$, then $B(b)$.

C.H.

TODO

- Prop - Type
- Proof - Element
- $\top$ - 1
- $\bot$ - 0
- $P \vee Q$ - $A + B$
- $P \wedge Q$ - $A * B$
- $P \Rightarrow Q$ - $A \to B$
- $\neg P$ - $A \to 0$
- $\exists_x P(x)$ - $\Sigma_{x:A} P(x)$
- $\forall_x P(x)$ - $\Pi_{x:A} P(x)$
- $P = Q$ - $P = Q$