Outline

Lectures 1–3: Rules for type theory w/ $\Pi, \Sigma, \emptyset, \mathbb{1}, +, \mathbb{N}, =$

(§1–5 of Rijke)

Today:
- Universes $\mathcal{U}$ (§6)
- Propositions as types, Curry-Howard interpretation (§7)

Why do we need universes?

1) To prove $0 \neq 1$ in $\mathbb{N}$, i.e., $(0 =_{\mathbb{N}} 1) \to \emptyset$.

And, more generally, define type families by induction.

2) To write polymorphic terms, e.g., instead of defining

$\mathrm{id}_A : A \to A$ for each type, we'll be able to define

$\mathrm{id}^{\mathcal{U}} : \prod_{X:\mathcal{U}} \mathcal{T}(X) \to \mathcal{T}(X)$ for every universe $\mathcal{U}$

3) To do category theory in a stream-lined way,

compare Grothendieck universes.

Just as $=$ is an internalization of $\doteq$,

a universe $\mathcal{U}$ is an internalization of the judgment A type

We got universes by reflection on what we do with types.

What is a universe?

<u>Slogan:</u> Whatever we can do with types, we can do inside a universe.

So a universe better reflect what we've done so far! $(\Pi, \Sigma, \emptyset, \mathbb{1}, +, \mathbb{N}, =)$

<u>Def</u> (at meta level) A universe is a type family

$$\mathcal{U} \text{ type}, \quad X:\mathcal{U} \vdash \mathcal{T}(X) \text{ type}$$

together with:

- $\check{\Pi} : \Pi_{X:\mathcal{U}}((\mathcal{T}(X) \to \mathcal{U}) \to \mathcal{U})$ s.t. $\mathcal{T}(\check{\Pi}(X,Y)) \doteq \Pi_{x:\mathcal{T}(X)} \mathcal{T}(Yx)$ for any $X:\mathcal{U}$, $Y:\mathcal{T}(X) \to \mathcal{U}$
- $\check{\Sigma} : \Pi_{X:\mathcal{U}}((\mathcal{T}(X) \to \mathcal{U}) \to \mathcal{U})$ s.t. $\mathcal{T}(\check{\Sigma}(X,Y)) \doteq \Sigma_{x:\mathcal{T}(X)} \mathcal{T}(Yx)$ —‖—
- $\check{\emptyset}, \check{\mathbb{1}}, \check{\mathbb{N}} : \mathcal{U}$ s.t. $\mathcal{T}(\check{\emptyset}) \doteq \emptyset$, $\mathcal{T}(\check{\mathbb{1}}) \doteq \mathbb{1}$, $\mathcal{T}(\check{\mathbb{N}}) \doteq \mathbb{N}$
- $\_\check{+}\_ : \mathcal{U} \to \mathcal{U} \to \mathcal{U}$ s.t. $\mathcal{T}(X \check{+} Y) \doteq \mathcal{T}(X) + \mathcal{T}(Y)$ for $X, Y : \mathcal{U}$
- $\_\check{=}\_ : \Pi_{X:\mathcal{U}}(\mathcal{T}(X) \to \mathcal{T}(X) \to \mathcal{U})$ s.t. $\mathcal{T}(x \mathrel{\check{=}_X} y) \doteq (x =_{\mathcal{T}(X)} y)$ for $X:\mathcal{U}$, $x, y : \mathcal{T}(X)$

Assuming enough universes.

To repeat: whatever we can do with types, we should be able to do in a universe.

Postulate (at meta-level) Whenever we have type families

$$\Gamma_1 \vdash A_1 \text{ type} \quad \cdots \quad \Gamma_n \vdash A_n \text{ type}$$

there is a universe $(\mathcal{U}, \tau)$ (in the empty context) containing these, i.e., w/ terms $\Gamma_i \vdash \check{A}_i : \mathcal{U}$ s.t. $\Gamma_i \vdash \tau(\check{A}_i) \doteq A_i$ type for $i = 1, \ldots, n$

Examples
- $n = 0$: There is a base universe $(\mathcal{U}_0, \tau_0)$
- If $(\mathcal{U}, \tau)$ is a universe, there's a successor universe $(\mathcal{U}^+, \tau^+)$:

$$\mathcal{U} \text{ type} \quad X : \mathcal{U} \vdash \tau(X) \text{ type} \rightsquigarrow \check{\mathcal{U}} : \mathcal{U}^+, \; X : \mathcal{U} \vdash \check{\tau}(X) : \mathcal{U}^+$$

$$\tau^+(\check{\mathcal{U}}) \doteq \mathcal{U}, \quad X : \mathcal{U} \vdash \tau^+(\check{\tau}(X)) \doteq \tau(X)$$

We get $\mathrm{Lift} : \mathcal{U} \to \mathcal{U}^+$, $\mathrm{Lift}(X) :\doteq \check{\tau}(X)$

- If $(\mathcal{U}, \mathcal{T}_{\mathcal{U}})$, $(\mathcal{V}, \mathcal{T}_{\mathcal{V}})$ are universes, there is a <u>join universe</u>

$$\begin{array}{c} \mathcal{U} \sqcup \mathcal{V} \\ \mathcal{T} \end{array} : \quad \begin{array}{l} \mathcal{U} \text{ type} \\ X : \mathcal{U} \vdash \mathcal{T}_{\mathcal{U}}(X) \text{ type} \\ \mathcal{V} \text{ type} \\ X : \mathcal{V} \vdash \mathcal{T}_{\mathcal{V}}(X) \text{ type} \end{array} \qquad \begin{array}{l} \check{\mathcal{U}} : \mathcal{U} \sqcup \mathcal{V}, \quad \mathcal{T}(\check{\mathcal{U}}) \doteq \mathcal{U} \\ X : \mathcal{U} \vdash \check{\mathcal{T}}_{\mathcal{U}}(X) : \mathcal{U} \sqcup \mathcal{V}, \; X : \mathcal{U} \vdash \mathcal{T}(\check{\mathcal{T}}_{\mathcal{U}}(X)) \doteq \mathcal{T}_{\mathcal{U}}(X) \\ \text{———} \,\|\, \text{———} \end{array}$$

<u>NB</u> <u>no</u> requirement that $\mathcal{U}_0$, $\mathcal{U}^+$ or $\mathcal{U} \sqcup \mathcal{V}$ are minimal!

<u>Discussion</u>

- Universes are open-ended: If/when we add more type formers (W, HITs, ...) we'll want universes to be closed under these too.
- We get a hierarchy $\mathcal{U}_0$, $\mathcal{U}_0^+ \doteq: \mathcal{U}_1$, $\mathcal{U}_0^{++}$, ..., but a priori, there's no reason for these to exhaust all the types. OTOH, the reflection principle won't give us more than these.
- We might expect $\mathrm{Lift}(\check{\mathbb{N}}_0) \doteq \check{\mathbb{N}}_1$ for $\mathrm{Lift} : \mathcal{U}_0 \to \mathcal{U}_1$ by reflection, etc. this is called <u>cumulativity</u>. Not assumed here or in Agda, but useful!

Girard's, Hurkens' paradox:

In the 1971 version of his type theory, Martin-Löf had a universe $\mathcal{U}$ w/ $\check{\mathcal{U}} : \mathcal{U}$ and $T(\check{\mathcal{U}}) \doteq \mathcal{U}$. ("type in type")

Girard showed in his 1972 thesis that this is <u>inconsistent</u>, by adapting Burali-Forti's paradox: there can be no ordinal of all ordinals. This was simplified by Hurkens in 1995 (see Agda file).

With general inductive types, it's possible to give a very short proof of $\emptyset$ assuming such $\mathcal{U}$. (à la Russell's paradox).

Larger universes / further research

Reflecting on the reflection process, we can propose even larger universes, e.g., Palmgren's super-universe, closed under the universe successor operation.

Partially superceded by general induction-recursion

Universe polymorphism

Even w/ $\mathcal{U}$, we only have a polymorphic identity function ranging over $\mathcal{U}$, not all types, so proof assistants (Agda, Coq, Lean, etc.) have mechanisms for universe polymorphism. Newer research suggests having a universe level judgment, ℓ Level, which can then be internalized as a type.

<u>Convention</u>: we'll leave out $\mathcal{T}(-)$ and $\check{-}$'s for a universe $(\mathcal{U},\mathcal{T})$ as they can always be inferred from context (no pun intended).

<u>Ex</u> $\mathrm{id} : \prod_{X:\mathcal{U}} X \to X$

$\mathrm{id} = \lambda X\ \lambda x.\ x$.

<u>Ex</u> $\text{is-true} : \mathrm{bool} \to \mathcal{U}$

$\text{is-true false} \doteq \emptyset$

$\text{is-true true} \doteq \mathbb{1}$

<u>Ex</u> true $\neq$ false in bool needs a universe

$\text{Eq-bool} : \mathrm{bool} \to \mathrm{bool} \to \mathcal{U}$

$\text{Eq-bool}\ x\ y = \text{ind-bool}^{\mathcal{U}}($

(x false) $\to$ $\text{ind-bool}^{\mathcal{U}}(\mathbb{1}, \emptyset, y)$,

(x true) $\to$ $\text{ind-bool}^{\mathcal{U}}(\emptyset, \mathbb{1}, y)$, x)

y false   y true

(With no universes, we have a "types as propositions" model, where all terms $x,y:X$ in a type are equal.)

Then $\text{Eq-bool false false} \doteq \mathbb{1}$

$\text{Eq-bool false true} \doteq \emptyset$

$\text{Eq-bool true false} \doteq \emptyset$

$\text{Eq-bool true true} \doteq \mathbb{1}$

true $\neq$ false cont'd

$\text{Eq-bool false false} \doteq \mathbb{1}$
$\text{Eq-bool false true} \doteq \emptyset$
$\text{Eq-bool true false} \doteq \emptyset$
$\text{Eq-bool true true} \doteq \mathbb{1}$

Eq-bool is reflexive: $\text{refl-Eq}_{\text{bool}} : \prod_{b:\text{bool}} \text{Eq-bool } b\ b$

by bool-induction.

<u>Thm</u> For all $b\ b' : \text{bool}$, $b = b' \leftrightarrow \text{Eq-bool } b\ b'$

Def $f : \prod_{b,b'} (b = b' \to \text{Eq-bool } b\ b')$ by path induction:

$$f(b, b', p) = \text{ind-eq}_b^{x\,r.\,\text{Eq-bool } b\ x}(\text{refl-Eq}_{\text{bool}}\ b,\ b',\ p)$$

and $g : \prod_{b,b'} (\text{Eq-bool } b\ b' \to b = b')$

$g \text{ false false } t = \text{refl}_{\text{false}}$

$g \text{ false true } u = \text{ind-}\emptyset(u)$

$g \text{ true false } u = \text{ind-}\emptyset(u)$

$g \text{ true true } t = \text{refl}_{\text{true}}$.

<u>Cor</u> $\text{true} = \text{false} \to \emptyset$

$f(\text{true}, \text{false})$ works.

Observational equality on $\mathbb{N}$

Same principle, but we need recursion. We want $\mathrm{Eq}\text{-}\mathbb{N} : \mathbb{N} \to \mathbb{N} \to \mathcal{U}$

s.t. for all $n, m : \mathbb{N}$:

$$\begin{aligned}
\mathrm{Eq}\text{-}\mathbb{N}\ 0\ 0 &\doteq \mathbb{1} \\
\mathrm{Eq}\text{-}\mathbb{N}\ 0\ (\mathrm{succ}\ m) &\doteq \emptyset \\
\mathrm{Eq}\text{-}\mathbb{N}\ (\mathrm{succ}\ n)\ 0 &\doteq \emptyset \\
\mathrm{Eq}\text{-}\mathbb{N}\ (\mathrm{succ}\ n)\ (\mathrm{succ}\ m) &\doteq \mathrm{Eq}\text{-}\mathbb{N}\ n\ m
\end{aligned}$$

Q: What if we put $n = m$ here instead?

We need double induction w/ strong IH for n:

$(X : \mathcal{U})$

$$\mathrm{Eq}\text{-}\mathbb{N}(n, m) := \mathrm{ind}\text{-}\mathbb{N}^{x.\,\mathbb{N}\to\mathcal{U}}\big(\lambda m.\ \mathrm{ind}\text{-}\mathbb{N}^{y.\,\mathcal{U}}(\mathbb{1}, yX.\ \emptyset, m),$$

$\leftarrow n \doteq 0$

$$xf.\ \lambda m.\ \mathrm{ind}\text{-}\mathbb{N}^{y.\,\mathcal{U}}(\emptyset, yX.\ f\ y, m),\ n\big)$$

$\leftarrow n \doteq \mathrm{succ}\ x$

$(f : \mathbb{N} \to \mathcal{U})$

repr. $\mathrm{Eq}\text{-}\mathbb{N}(x, -)$

$m \doteq 0$

$m \doteq \mathrm{succ}\ y$

Again, we can prove Eq-$\mathbb{N}$ is refl. by $\mathbb{N}$-induction,

then show $(n = m) \leftrightarrow$ Eq-$\mathbb{N}$ n m

<u>Cor</u> $\prod_{n:\mathbb{N}} (0 = \mathrm{succ}\ n \leftrightarrow \emptyset)$

<u>Cor</u> $\prod_{n,m:\mathbb{N}} (\mathrm{succ}\ n = \mathrm{succ}\ m \leftrightarrow n = m)$

Curry–Howard interpretation, following BHK

We have already seen this in action many times:

propositions P	types A
proofs of P	terms/elements of A
red. of proofs	judgmental eq. of terms
$\top$	$\mathbb{1}$
$\bot$	$\emptyset$
$P \Rightarrow Q$	$A \to B$
$P \wedge Q$	$A \times B$
$\neg P$	$A \to \emptyset$
$P \vee Q$	$A + B$
$\forall_{x:A} P(x)$	$\Pi_{x:A} B(x)$
$\exists_{x:A} P(x)$	$\Sigma_{x:A} B(x)$
$x =_A y$	$x =_A y$

Brief history

1908 Brouwer: On the unreliability of logical laws

1920's, 30's: Heyting, Kolmogorov

1934: Curry

(1936: Turing machines)

1958: Curry–Feys's combinatory logic

1969: Howard (inspired by Curry, Kreisel and Tait)

$\swarrow$

Dana Scott, Per Martin-Löf

independently, de Bruijn.

In Howard's note, only for arithmetic, and aware that case of $\vee, \exists$ was delicate.

E.g., in logic we don't have formulas/propositions that depend on proof of $P \vee Q$ / $\exists_{x:A} P(x)$

In this case, we need to worry about overlap, i.e. both P and Q true or both $P(x_1)$ and $P(x_2)$ true.

Also, we need a refinement in order to ensure compatibility of classical logic with the types as spaces interpretation. (we'll return to this later)

Example Divisibility on $\mathbb{N}$

Def $k \mid n$ type if $k, n : \mathbb{N}$

$$(k \mid n) :\equiv \sum_{d : \mathbb{N}} k\, d = n$$

Prop For all n, $1 \mid n$ and $n \mid n$.

$$(n, p_n) \qquad\qquad (1, q_n)$$

$$p_n : 1 \cdot n = n \qquad\qquad q_n : n \cdot 1 = n$$

Prop For all n, $n \mid 0$. Use $(0, r_n)$ where $r_n : n \cdot 0 = 0$

NB For all n, $(n, s_n) : 0 \mid 0$, where $s_n : 0 \cdot n = 0$

So we have "$\mathbb{N}$ many proofs that $0 \mid 0$"

(make precise w/ equivalences next lecture)

Ex If $k \mid x$ and $k \mid y$, then $k \mid (x+y)$.

<u>Example</u> Type theoretic choice. Suppose A, B are types, $x:A, y:B \vdash R(x,y)$ type a correspondence/(typal) relation. Then $\left(\prod_{x:A} \sum_{y:B} R(x,y)\right) \leftrightarrow \sum_{f:A\to B} \prod_{x:A} R(x, f_x)$ is a C-H reading of the axiom of choice.

"$\rightarrow$" $H \mapsto (\lambda x.\, pr_1(Hx), \lambda x.\, pr_2(Hx))$

"$\leftarrow$" $(f,h) \mapsto \lambda x.(fx, hx)$

by Σ-ind

(We'll return to the "real AC" later.)

<u>Example</u>: Split surjections

Suppose $f: A \to B$. Apply C-H to the usual way of saying "f is surjective" gives the type

$$\left(\prod_{y:B} \sum_{x:A} fx = b\right) =: \text{is-split-surjective}(f)$$

By TTAC, this amounts to a map $g: B \to A$ w/ $h: \underbrace{\prod_{y:B} (f \circ g)(y) = y}_{\text{a homotopy!}}$