

Reconciling *Impredicative* Axiom and Universe

Stefan Monnier

`monnier@iro.umontreal.ca`

Université de Montréal

The skinny on impredicativity

According to the O.E.D:

im- + predicative, adj. & n.: With a sneaky form of circularity

~1389, Chaucer: *Can't trust this dude, he's too impredicative!*

The origin of the notion of *types*, from Bertrand Russell:

Let S be the set of all sets that do not contain themselves:

$$S \in S \Leftrightarrow S \notin S$$

Fix: introduce a stratification to prevent such self-applications

Anti-fix: some forms of impredicativity seem consistent and useful

Self-application

Can be harmless and useful:

$$\textit{identity} : \forall t. t \rightarrow t$$

$$\textit{identity} = \Lambda t. \lambda x : t. x$$

$$\textit{selfid} = \textit{identity}[\forall t. t \rightarrow t] \textit{identity}$$

Common source of circularity

Basis of many paradoxes, usually under the name *diagonalization*

$$Y = \lambda f. (\lambda x. f (x x)) (\lambda x. f (x x))$$

Note: Seen as a set, $\forall t. t \rightarrow t$ is self-nested!

Why do I care?

Working on Typer, an ML/Haskell with dependent types and macros

Typer: low-level λ -calculus intermediate language

Impredicativity used in:

- Encoding modules as tuples (with level-polymorphic definitions)
- The (mistaken?) desire to subsume System F
- Type-preserving closure conversion

Existing forms of impredicativity aren't sufficient

Prop is an irregularity in universe polymorphism

Forms of impredicativity

Impredicative universes, most common of them is Prop:

$$\tau_2 : \text{Prop} \implies (x : \tau_1) \rightarrow \tau_2 : \text{Prop}$$

As present in System F, Coq, Lean, and many others

Resizing axioms:

$$\tau : \text{Type}_\ell \wedge P(\tau) \implies \tau : \text{Type}_{\ell'}$$

Most famously, HoTT's propositional resizing

Unsound impredicativity:

$$\text{Type} : \text{Type} \quad \text{or} \quad \tau_2 : \text{Type}_\ell \implies (x : \tau_1) \rightarrow \tau_2 : \text{Type}_\ell$$

λ^* , System U, System U^- , original MLTT

The Question

How do forms of impredicativity relate?



How do impredicative universes compare to resizing axioms?



How does impredicative Prop compare to HoTT propositional resizing?



Show equivalence of close(?) relatives to those systems

Impredicative Prop universe

The “standard” form of impredicativity

$$\tau_2 : \text{Prop} \implies (x : \tau_1) \rightarrow \tau_2 : \text{Prop}$$

Needs to be the bottom universe: $\text{Prop} = \text{Type}_0$

In most cases, Prop is *erased*:

- Restriction on eliminations from Prop to Type_ℓ .
- Desirable to get *proof irrelevance*.
- Otherwise incompatible with several classical axioms.
- The restriction *very* rarely makes a difference.

HoTT's propositional resizing

$$\text{isProp } \tau = (x : \tau) \rightarrow (y : \tau) \rightarrow x = y$$

$$\text{resize} : (t : \text{Type}_{\ell+1}) \rightarrow \text{isProp } t \rightarrow \text{Type}_{\ell}$$

`isProp` is similar to proof irrelevance

Friends with *propositional truncation*:

$$|| \cdot || : \text{Type}_{\ell} \rightarrow \text{Type}_{\ell} \quad (\forall \ell)$$

$$| \cdot | : (t : \text{Type}_{\ell}) \rightarrow t \rightarrow ||t|| \quad (\forall \ell)$$

$$\begin{aligned} \text{elim}_{||} : (t_1 : \text{Type}_{\ell_1}) \rightarrow (t_2 : \text{Type}_{\ell_2}) \rightarrow \text{isProp } t_2 & \quad (\forall \ell_1, \ell_2) \\ \rightarrow ||t_1|| \rightarrow (t_1 \rightarrow t_2) \rightarrow t_2 \end{aligned}$$

Where `isProp(|| $\tau$ ||)`!

Base calculus: pCC_{ω}

We start from a *predicative* PTS with a tower of universes:

$$e, \tau ::= s \mid x \mid (x : \tau_1) \rightarrow \tau_2 \mid \lambda x : \tau. e \mid e_1 e_2$$

$$\mathcal{S} = \{ \text{Type}_{\ell} \mid \ell \in \mathbb{N} \}$$

$$\mathcal{A} = \{ (\text{Type}_{\ell} : \text{Type}_{\ell+1}) \mid \ell \in \mathbb{N} \}$$

$$\mathcal{R} = \{ (\text{Type}_{\ell_1}, \text{Type}_{\ell_2}, \text{Type}_{\ell_1 \sqcup \ell_2}) \mid \ell_1, \ell_2 \in \mathbb{N} \}$$

Stratification of λ^* to avoid the circularity!

Not a superset of CoC or even System F:

$$\forall t : \text{Type}_0. t \rightarrow t \text{ is in } \text{Type}_1 \text{ rather than } \text{Type}_0$$

Impredicative universe: $uCC\omega$

We add impredicative quantification to $pCC\omega$'s bottom universe:

$$e, \tau ::= s \mid x \mid (x : \tau_1) \rightarrow \tau_2 \mid \lambda x : \tau. e \mid e_1 e_2$$

$$\mathcal{S} = \{ \text{Type}_\ell \mid \ell \in \mathbb{N} \}$$

$$\mathcal{A} = \{ (\text{Type}_\ell : \text{Type}_{\ell+1}) \mid \ell \in \mathbb{N} \}$$

$$\begin{aligned} \mathcal{R} = & \{ (\text{Type}_{\ell_1}, \text{Type}_{\ell_2}, \text{Type}_{\ell_1 \sqcup \ell_2}) \mid \ell_1 \in \mathbb{N}, \ell_2 \in \mathbb{N}^* \} \\ & \cup \{ (\text{Type}_\ell, \text{Type}_0, \text{Type}_0) \mid \ell \in \mathbb{N} \} \end{aligned}$$

Now $\forall t : \text{Type}_0. t \rightarrow t$ is in Type_0 rather than Type_1

Resizing axiom: rCC_{ω}

We start from an erasure monad

A combination of HoTT's propositional truncation with its resizing axiom:

$$|| \cdot || : \text{Type}_{\ell} \rightarrow \text{Type}_0 \quad (\forall \ell)$$

$$| \cdot | : (t : \text{Type}_{\ell}) \rightarrow t \rightarrow ||t|| \quad (\forall \ell)$$

$$\begin{aligned} \text{bind} : (t_1 : \text{Type}_{\ell_1}) \rightarrow (t_2 : \text{Type}_{\ell_2}) & \quad (\forall \ell_1, \ell_2) \\ \rightarrow ||t_1|| \rightarrow (t_1 \rightarrow ||t_2||) \rightarrow ||t_2|| \end{aligned}$$

$$\Gamma \vdash \text{bind}_{\tau_1, \tau_2} |e_1| \lambda x : \tau_1. e_2 \simeq e_2[e_1/x] : \tau_2$$

Similar monad used by Arnaud Spiwack to axiomatize Hurkens's paradox

Warm up: translating rCC_ω to uCC_ω

Encoding rCC_ω into uCC_ω is easy. We can just provide definitions:

$$||\tau|| = (t : \text{Type}_0) \rightarrow (\tau \rightarrow t) \rightarrow t$$

$$|e|_\tau = \lambda t : \text{Type}_0. \lambda k : (\tau \rightarrow t). k \ e$$

$$\text{bind}_{\tau_1, \tau_2} \ e_1 \ e_2 = e_1 \ ||\tau_2|| \ e_2$$

And indeed it satisfies the desired reduction:

$$\text{bind} \ |e_1| \ \lambda x : \tau_1. e_2$$

$$\simeq (\lambda \alpha : \text{Type}_0. \lambda f : (\tau_1 \rightarrow \alpha). f \ e_1) \ \tau_2 \ (\lambda x : \tau_1. e_2)$$

$$\simeq e_2[e_1/x]$$

Encoding Prop using resizing

We want to encode (written $[\cdot]$) $\text{uCC}\omega$ terms into $\text{rCC}\omega$

Core problem: $(x : \tau_1) \rightarrow \tau_2$ where $\tau_1 : \text{Type}_{>0}$ and $\tau_2 : \text{Type}_0$

- $\text{uCC}\omega$ puts them in Type_0
- $\text{rCC}\omega$ would put them in Type_ℓ

Basic approach: use $|| \cdot ||$ to bring them down to Type_0 :

$$[(x : \tau_1) \rightarrow \tau_2] = ||(x : [\tau_1]) \rightarrow [\tau_2]||$$

Inspired by Coq, we erase all Type_0 terms

Translating uCC_ω to rCC_ω : first try

$$[x] = x$$

$$[\text{Type}_\ell] = \text{Type}_\ell$$

$$[(x:\tau_1) \rightarrow \tau_2] = \begin{cases} ||(x:[\tau_1]) \rightarrow [\tau_2]|| & \text{if in the Type}_0 \text{ universe} \\ (x:[\tau_1]) \rightarrow [\tau_2] & \text{otherwise} \end{cases}$$

$$[\lambda x:\tau. e] = \begin{cases} |\lambda x:[\tau]. [e]| & \text{if in the Type}_0 \text{ universe} \\ \lambda x:[\tau]. [e] & \text{otherwise} \end{cases}$$

$$[e_1 \ e_2] = \begin{cases} \text{bind } [e_1] \ \lambda f. f \ [e_2] & \text{if } e_1 \text{ in the Type}_0 \text{ universe} \\ [e_1] \ [e_2] & \text{otherwise} \end{cases}$$

¿Type non-preservation?

When we try to show

$$\Gamma \vdash e : \tau \implies [\Gamma] \vdash [e] : [\tau]$$

! We fail on $[e_1 \ e_2] \implies \text{bind } [e_1] \ \lambda f.f \ [e_2]!$

- $f \ [e_2]$ has type $[\tau_2]$
- We know that $\tau_2 : \text{Type}_0$ so we know $[\tau_2]$ *will be* erased.
- But the type does not always reflect that: $[t \rightarrow t] = ||t \rightarrow t||$

¿Type non-preservation?

When we try to show

$$\Gamma \vdash e : \tau \implies [\Gamma] \vdash [e] : [\tau]$$

! We fail on $[e_1 \ e_2] \implies \text{bind } [e_1] \ \lambda f.f \ [e_2]!$

- $f \ [e_2]$ has type $[\tau_2]$
- We know that $\tau_2 : \text{Type}_0$ so we know $[\tau_2]$ *will be* erased.
- But the type does not always reflect that: $[t \rightarrow t] = ||t \rightarrow t||$

$\implies$ Weaken `bind` so it does not require literally $||\tau_2||$

Instead, require a proof of erasure

$\implies$ Less of a monad, a bit more like propositional truncation

$\implies$ Also, we need pairs

New rCC_{ω} axioms

bind : $(t_1 : \text{Type}_{\ell}) \rightarrow (t_2 : \text{Type}_0) \rightarrow (\forall \ell)$

$||t_1|| \rightarrow (t_1 \rightarrow (t_2 \times \text{IsErased } t_2)) \rightarrow$
 $(t_2 \times \text{IsErased } t_2)$

IsErased : $\text{Type}_0 \rightarrow \text{Type}_0$

iserased : $(t : \text{Type}_{\ell}) \rightarrow \text{IsErased } ||t|| (\forall \ell)$

$\cdot \times \cdot$: $\text{Type}_0 \rightarrow \text{Type}_0 \rightarrow \text{Type}_0$

$\langle \cdot, \cdot \rangle$: $(t_1 : \text{Type}_0) \rightarrow (t_2 : \text{Type}_0) \rightarrow t_1 \rightarrow t_2 \rightarrow (t_1 \times t_2)$

$\cdot.0$: $(t_1 : \text{Type}_0) \rightarrow (t_2 : \text{Type}_0) \rightarrow (t_1 \times t_2) \rightarrow t_1$

$\Gamma \vdash \langle e_1, e_2 \rangle.0 \simeq e_1 : \tau_1$

New translation of uCC_ω to rCC_ω

$$\begin{aligned}
 \llbracket \tau \rrbracket &= \begin{cases} \llbracket \tau \rrbracket \times \text{IsErased } \llbracket \tau \rrbracket & \text{if } \tau : \text{Type}_0 \\ \llbracket \tau \rrbracket & \text{otherwise} \end{cases} \\
 \llbracket (x : \tau_1) \rightarrow \tau_2 \rrbracket &= \begin{cases} ||(x : \llbracket \tau_1 \rrbracket) \rightarrow \llbracket \tau_2 \rrbracket|| & \text{if in the Type}_0 \text{ universe} \\ (x : \llbracket \tau_1 \rrbracket) \rightarrow \llbracket \tau_2 \rrbracket & \text{otherwise} \end{cases} \\
 \llbracket e_1 e_2 \rrbracket &= \begin{cases} \text{bind } \llbracket e_1 \rrbracket.0 \lambda f.f \llbracket e_2 \rrbracket & \text{if } e_1 \text{ in the Type}_0 \text{ universe} \\ \llbracket e_1 \rrbracket \llbracket e_2 \rrbracket & \text{otherwise} \end{cases} \\
 \llbracket \lambda x : \tau_1. e \rrbracket &= \begin{cases} \langle | \lambda x : \llbracket \tau_1 \rrbracket. \llbracket e \rrbracket |, \text{iserased } ((x : \llbracket \tau_1 \rrbracket) \rightarrow \llbracket \tau_2 \rrbracket) \rangle & \\ \lambda x : \llbracket \tau_1 \rrbracket. \llbracket e \rrbracket & \text{if not in the Type}_0 \text{ universe} \end{cases}
 \end{aligned}$$

Translating new rCC_ω to uCC_ω

$$||\tau|| \quad = \quad (t : \text{Type}_0) \rightarrow (\tau \rightarrow t) \rightarrow t$$

$$|e|_{\tau} \quad = \quad \lambda t : \text{Type}_0. \lambda k : (\tau \rightarrow t). k \ e$$

$$\text{bind}_{\tau_1, \tau_2} \ e_1 \ e_2 \quad = \quad e_1 \ (\tau_2 \times \text{IsErased } t_2) \ e_2$$

$$\text{IsErased } \tau \quad = \quad (t : \text{Type}_0) \rightarrow t \rightarrow t$$

$$\text{iserased } \tau \quad = \quad \lambda t : \text{Type}_0. \lambda x : t. x$$

$$\tau_1 \times \tau_2 \quad = \quad \tau_1$$

$$\langle e_1, e_2 \rangle \quad = \quad e_1$$

$$e.0 \quad = \quad e$$

Comparison to propositional resizing

The axioms of $\text{rCC}\omega$ are satisfied by HoTT's axioms:

$||\tau|| \leq$ *propositional truncation and resizing*

$\text{IsErased} \leq \text{isProp}$

$\text{bind} \leq$ the recursion principle of *propositional truncation*

They are less general than those of HoTT:

- $\text{rCC}\omega$'s $||\tau||$ does not imply proof irrelevance
- IsErased can be fulfilled only by erasure/truncation rather than by a proof of irrelevance

Differences to traditional Prop

The encoding from $\text{uCC}\omega$ to $\text{rCC}\omega$ does use a traditional Prop

The encoding from $\text{rCC}\omega$ targets a *different* $\text{uCC}\omega$!

- Predicative and impredicative functions from Type_ℓ to Type_0 :

Used to simplify the $\text{rCC}\omega$ to $\text{uCC}\omega$ encoding

Not supported in the $\text{uCC}\omega$ to $\text{rCC}\omega$ encoding

Could be replaced by cummulativity of universes or lift

Difference related to the use of Prop without need for impredicativity

Conclusion

- 😊 Show an almost-equivalence between basic $\text{uCC}\omega$ and $\text{rCC}\omega$
- 😊 Extends to UTT-style inductive types
- 😊 Suggests Prop and resizing might be closely related(?)
- 😊 Might hopefully help translate proofs between such systems
- 😞 Doesn't include η
- 😞 Not quite equivalence and non-identity round-trip
- 😞 Inductive types in Prop: WiP

Next up: resizing universe polymorphism?