Investigating the Tarpeian rock : Paradoxes in Type Theory

Thanks to Thierry for explanation and inspiration!

Thorsten Altenkirch

Functional Programming Laboratory
School of Computer Science
University of Nottingham

August 25, 2022

All attemps to strengthen this system the system of constructions in particular to temp er with the fourth level should be very cautious: the Tarpeian Rock is close to the Capitol

J.Y. Girard

The paradox of trees

Coquand 92

The paradox of trees in type theory.
BIT Numerical Mathematics. 1992 Mar;32(1):10-4.

- Russell paradox in Type Theory with Type : Type
- Sets as trees

Using Type:Type

```
data Tree : Set where
  node : (I : Set) → (I → Tree) → Tree
```

```
[] : Tree
[] = node ⊥ λ ()
[[]] : Tree
[[]] = node ⊤ (λ _ → [])
[[],[[]]] : Tree
[[],[[]]] = node Bool (λ x → if x then [] else [[]])
```

Derving the paradox

```
data _∈_ : Tree → Tree → Set where
  ∈-in : {I : Set} (f : I → Tree) (i : I) → f i ∈ node I f

Good : Tree → Set
Good t = ¬ (t ∈ t)

Russell : Tree
Russell = node (Σ[ t ∈ Tree ] Good t) proj₀

strange : Good Russell ⇔ ¬ (Good Russell)
proj₀ strange h = ∈-in proj₀ (Russell , h)
proj₁ strange (∈-in ∘ (proj₀) (.Russell , good)) = good

ex : ¬ (¬ P ⇔ P)
ex = ?
```

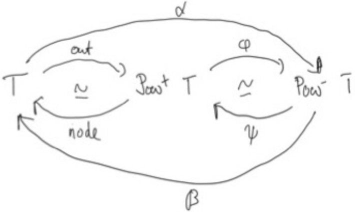
Analyzing the tree paradox

Pow- , Pow+ : Set $\rightarrow$ Set

Pow- A = A $\rightarrow$ Set
Pow+ A = Σ[I $\in$ Set] (I $\rightarrow$ A)
(α , β) : Pow+ A $\simeq$ Pow- A
(node, out) : Pow+ Tree $\simeq$ Tree

Cantor: The following is inconsistent:

α : X $\rightarrow$ Pow- X β : Pow- X $\rightarrow$ X
ret : (P : Pow- X) (x : X) $\rightarrow$ α (β P) X $\Leftrightarrow$ P X



Are we done?

- The paradox of trees is intuitively clear and has a short formalisation.
- But is it as strong as Girard's paradox?
- Girard's paradox doesn't require Type:Type and only need Π-types.

Higher order logic (HOL)

- In Higher Order Logic we can quantify over propositions.
- As an example we can encode inductive definitions, that is given $A : \mathsf{Set}$ and $\_<\_ : A \to A \to \mathsf{Prop}$ we can replace the inductive definition

$$
\begin{array}{l}
\textbf{data}\ \mathsf{Acc} : A \to \mathsf{Prop}\ \textbf{where} \\
\quad \mathsf{acc} : \{a : A\} \to (\{x : A\} \to x < a \to \mathsf{Acc}\ x) \to \mathsf{Acc}\ a
\end{array}
$$

by

$$
\begin{array}{l}
\mathsf{Acc} : A \to \mathsf{Prop} \\
\mathsf{Acc}\ x = (P : A \to \mathsf{Prop}) \\
\quad \to ((a : A) \to (\{b : A\} \to b < a \to P\ b) \to P\ a) \\
\quad \to P\ x
\end{array}
$$

- Gödel introduced System T as a functional calculus corresponding to Heyting Arithmetic (HA)
- Similarly Girard introduced System F as a functional calculus corresponding to Higher Order Logic.

$$\frac{\text{System T}}{\text{HA}} = \frac{\text{System F}}{\text{HOL}}$$

- $\text{HA}^\omega = \text{HA} + \text{System T}$
- $\text{SystemU} = \text{HOL} + \text{System F}$
- Alas, System U is inconsistent.

System U in Agda

```
Type = Set₁
Prp = Set₀
record Pi {ℓ} (A : Set ℓ) (B : A → Type) : Type where
  constructor lam
  field
    _$_ : (a : A) → B a
record All {ℓ} (A : Set ℓ) (B : A → Prp)  : Prp where
  constructor lam-p
  field
    _$p_ : (a : A) → B a
```

Using `NO_UNIVERSE_CHECK`.

Girard's paradox

Coquand 86

An analysis of Girard's paradox LICS 86

- Girard uses the Burali-Forti paradox about the well-order of all well orderings.
- In System U we can define a universal System to represent relations:

$$
\begin{aligned}
&\mathrm{U} \;:\; \mathrm{Type} \\
&\mathrm{U} \;=\; (\Pi[\, \mathrm{B} \in \mathrm{Type} \,]\, ((\mathrm{B} \;\rightarrow\; \mathrm{B} \;\rightarrow\; \mathrm{Prp}) \;\rightarrow\; \mathrm{Prp})) \;\rightarrow\; \mathrm{Prp} \\
&\text{in-U} \;:\; (\mathrm{A} \;:\; \mathrm{Type})\, (\mathrm{R} \;:\; \mathrm{A} \;\rightarrow\; \mathrm{A} \;\rightarrow\; \mathrm{Prp}) \;\rightarrow\; \mathrm{U} \\
&\text{in-U A R f} \;=\; (\mathrm{f} \;\$\; \mathrm{A})\, \mathrm{R}
\end{aligned}
$$

- We can define the order on well-orders on U.
- It is important that in-U A R $\equiv$ in-U B R implies that there is a monotone function from A , R to B , R

- We can also construct the Burali-Forti paradox on T defining the tree of all well-founded trees.
- Can we just use irreflexive relations instead of well-founded ones for Girard's paradox?

Girard's paradox ?

- A complete formalisation of Girard's paradox is quite complex.
- Can we avoid using Burali-Forti?
- Can we use an impredicative inductive definition to simplify the paradox?

A new paradox?

Coquand 95

A new paradox in type theory Studies in Logic and the Foundations of Mathematics. Vol. 13 86

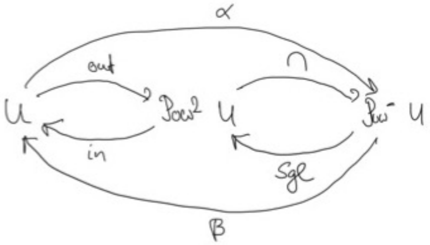
Reynolds 84

Polymorphism is not set-theoretic International Symposium on Semantics of Data Type

- This paradox is based on Reynold's observation that there is no set theoretic semantics of System F.
- We can encode a fixpoint of $\mathsf{Pow}^2\ \mathsf{A}\ =\ \mathsf{Pow\text{-}}\ (\mathsf{Pow\text{-}}\ \mathsf{A})$

$$
\begin{aligned}
&\mathsf{U}\ :\ \mathsf{Set}\\
&\mathsf{U}\ =\ (\mathsf{X}\ :\ \mathsf{Set})\ \to\ (\mathsf{Pow}^2\ \mathsf{X}\ \to\ \mathsf{X})\ \to\ \mathsf{X}\\
&\mathsf{inn}\ :\ \mathsf{Pow}^2\ \mathsf{U}\ \to\ \mathsf{U}\\
&\mathsf{out}\ :\ \mathsf{U}\ \to\ \mathsf{Pow}^2\ \mathsf{U}
\end{aligned}
$$

$\bigcap \ : \ \text{Pow}^2 \ U \ \rightarrow \ \text{Pow-} \ U$
$\bigcap Q\,x \ = \ (P \ : \ \text{Pow-} \ U) \ \rightarrow \ Q\,P \ \rightarrow \ P\,x$
$\text{Sgl} \ : \ \text{Pow-} \ U \ \rightarrow \ \text{Pow}^2 \ U$
$\text{Sgl}\,P \ = \ \lambda\,Q \ \rightarrow \ (x \ : \ U) \ \rightarrow \ P\,x \Leftrightarrow Q\,x$
$\cap\,(\text{Sgl}\,P)\,X \Leftrightarrow P\,X$



Partial Equivalence Relations (PERs)

- We need that out (inn x) $\equiv$ x !
- But the impredicative encoding only gives us
  Pow^2 (inn $\circ$ out) $\equiv$ out $\circ$ inn
- This can be addressed by interpreting the construction in the PER model.
- We define (U,Eq) using an impredicative definition.

$$\begin{aligned}\mathsf{Eq}\ x\ y\ =\ (R\ :\ U\ \to\ U\ \to\ \mathsf{Set})\ \to\ \mathsf{per}\ R \\ \to\ \mathsf{mor}\ (\mathsf{pow}^2 R\ R)\ R\ \mathsf{inn}\ \to\ R\ x\ y\end{aligned}$$

Hurken's paradox

Hurken 95

A Simplification of Girard's paradox TLCA 95

- Hurken uses the same U as Coquand.
- He avoids the use of PERs!
- He encodes a variant of the Burali-Forti paradox.

```
Ind : Pow² U
Ind X = ∀[ x ∈ U ] (out x X → X x)

Acc : Pow U
Acc x = ∀[ X ∈ Pow U ] (Ind X → X x)

Ω : U
Ω = inn Ind

accΩ : Acc Ω
accΩ = λp[ X ] λ iX → (iX $p Ω) (λp[ y ] (iX $p (inn (out y))))

_<_ : U → Pow U
x < y = ∀[ X ∈ Pow U ] (out y X → X x)

¬Δ : Pow U
¬Δ y = ¬ (inn (out y) < y)

lem : Ind ¬Δ
lem = λp[ x ] λ h p → (p $p ¬Δ) h (λp[ X ] λ q → (p $p λ y → X (inn (ou

¬accΩ : ¬ (Acc Ω)
¬accΩ h = (h $p ¬Δ) lem (λp[ X ] (λ p → (h $p λ y → X (inn (out y))) p))

paradox : ⊥
paradox = ¬accΩ accΩ
```

Comparing paradoxes

The paradox of trees Short and easy to understand, but needs Type:Type and W.

Girard's paradox We need to formalize Burali-Forti!

Coquand's paradox We need to develop the PER machinery.

Hurken's paradox Nice and short but hard to explain.