Universes In Type Theory

András Kovács[a]

April 8, 2021, EFOP Workshop

[a]The author was supported by the European Union, co-financed by the European Social Fund (EFOP-3.6.3-VEKOP-16-2017-00002).

Russell's paradox, or size issues

There is no set S in a consistent set theory such that $x \in S$ iff $x \notin x$.

Alternate formulations:

- Barber's paradox
- No set of all sets

Related:

- Real numbers are uncountable
- Gödel's first incompleteness theorem
- Undecidability of halting

(All instances of Lawvere's fixed point theorem)

Solution with universes

Solving Russell's paradox: set comprehension can only define a **subset** of a set.

But sometimes we still want to quantify over **all sets** of Zermelo-Fraenkel set theory.

Universe: a set which includes every ZF set.

Gödel: there is no universe inside ZF.

We can assume more universes if we want to quantify over more sets.

Universes in type theory

Type theories:

- Alternative to set theory as mathematical foundation.
- Used in most general-purpose theorem provers.
- Can be used for proving and programming at the same time.

Universes come up in type theories as well.

Every theorem prover has different universe-related features.

We want to give a general semantics covering many different features.

Example

Agda:

```
ℕ              : Set₀
Set₀           : Set₁
Set₁           : Set₂
(ℕ → Set₀)     : Set₁
(Set₀ → Bool) : Set₁
```

Identity function at all small sets:

```
id : (A : Set₀) → A → A
id A x = x
```

We have $\mathrm{Set}_i : \mathrm{Set}_{i+1}$ for all i.

Assuming $\mathrm{Set}_i : \mathrm{Set}_{i+1}$ implies contradiction (by Russell-like argument).

Extensions

The basic system can be tedious:

```
id₀ : (A : Set₀) → A → A
id₁ : (A : Set₁) → A → A
...
```

In Coq/Agda/Lean, various extra features are used. Example: universe polymorphism in Agda:

```
id : (l : Level)(A : Set l) → A → A
id l A x = x
```

Design choices and variations

How many universes? Agda/Coq: countably many.

Are universes totally ordered? Agda/Coq: yes.

What kind of level polymorphism?. Coq: bounded polymorphism. Agda: no bounds allowed. Bounded example:

```
myId : (i : ℕ) → i < 3 → (A : Set i) → A → A
myId i p A x = x
```

What kind of operations are available on levels? Agda is more liberal than Coq. Example:

```
ℕtoLevel : ℕ → Level
```

Are universes cumulative Agda: no. Coq: yes. Cumulativity: whenever `A : Set i`, we also have `A : Set (i + 1)`.

Research goals

We want to know that each point in the design space makes sense.

Making sense:

- Logical consistency.
- Is the type theory a proper programming language? Programs should compute to values and not get randomly stuck.
- Is proof checking decidable? (Not covered in current work).

Approach: use generic framework to cover as many features/variations as possible. Prove that everything in the framework is sensible.

Features covered by the framework

Universe levels may come from **any well-ordered set**, even transfinite:

```
Set i : Set (i + 1) : ... : Set ω : Set (ω + 1) : ...
```

Quantification over levels, arbitrary computation on levels.

```
myId : (i : ℕ) → (j : ℕ) → (A : Set (i + j)) → A → A
myId i j A x = x
```

Universes are cumulative

```
ℕ : Set₀
ℕ : Set₁
...
```

Implementation

We define a family of type theories with general universe features.

We prove all of them consistent by reducing all fancy features to a single previously known feature, called **induction-recursion**.

Induction-recursion is a type-theoretic analogue of assuming **Mahlo cardinals** in set theory.

```
Univ   : Set
Nat    : Univ
Π      : (A : Univ) → (Interp Univ → Univ) → Univ

Interp : Univ → Set
Interp Nat     = ℕ
Interp (Π A B) = (x : Interp A) → Interp (B x)
```

Publication

“Generalized Universe Hierarchies and First-Class Universe Levels”

Submission under review for FSCD 2021.

Thank you!