

# Canonicity for Indexed Inductive-Recursive Types

**András Kovács**

University of Gothenburg & Chalmers University of Technology

1st Oct 2025, Workshop in Honour of Peter Dybjer

# Inductive-Recursive Types

Mutual definition of an inductive type and a function acting on it.

**mutual**

**data** Code : Set<sub>0</sub> **where**

Nat' : Code

Π' : (A : Code) → (El A → Code) → Code

El : Code → Set<sub>0</sub>

El Nat' = Nat

El (Π' A B) = (a : El A) → El (B a)

Early and informal use by Per Martin-Löf [ML75, ML84].

Formal syntax & semantics developed by Peter and Anton [Dyb00, DS99, DS03, DS06].

Use-cases:

- Metatheory for TTs with various universe hierarchies:
  - Normalization for TTs with countable universes [ML75, AÖV18, PT23, ADE23].
  - Consistency [Kov22] and canonicity [CW25] for first-class universe levels.
- Others: partial functions [BC01], generic programming [BDJ03, Die17], large countable ordinals [Kov23, Kyu25].

# Inductive-Recursive Types

Use-cases:

- Metatheory for TTs with various universe hierarchies:
  - Normalization for TTs with countable universes [ML75, AÖV18, PT23, ADE23].
  - Consistency [Kov22] and canonicity [CW25] for first-class universe levels.
- Others: partial functions [BC01], generic programming [BDJ03, Die17], large countable ordinals [Kov23, Kyu25].

**Canonicity:** can we evaluate every closed term in MLTT+IR to a value?

IR has been supported for a long time in Agda with no canonicity issues, but without formal proof.

# Inductive-Recursive Types

Use-cases:

- Metatheory for TTs with various universe hierarchies:
  - Normalization for TTs with countable universes [ML75, AÖV18, PT23, ADE23].
  - Consistency [Kov22] and canonicity [CW25] for first-class universe levels.
- Others: partial functions [BC01], generic programming [BDJ03, Die17], large countable ordinals [Kov23, Kyu25].

**Canonicity:** can we evaluate every closed term in  $\text{MLTT} + \text{IR}$  to a value?

IR has been supported for a long time in Agda with no canonicity issues, but without formal proof.

In this talk & upcoming paper: formal proof of canonicity.

# How to specify an IR type?

The type of **IR signatures** is an inductive type.

```
data Sig  $i \{j\}$  ( $O : \text{Set}_j$ ) :  $\text{Set}_{(i+1 \sqcup j)}$  where  
   $\iota : O \rightarrow \text{Sig } i \ O$   
   $\sigma : (A : \text{Set } i) \rightarrow (A \rightarrow \text{Sig } i \ O) \rightarrow \text{Sig } i \ O$   
   $\delta : (A : \text{Set } i) \rightarrow ((A \rightarrow O) \rightarrow \text{Sig } i \ O) \rightarrow \text{Sig } i \ O$ 
```

The signature of the previous Code example:

```
SigCode : Sig 0 Set0  
SigCode  $\equiv \sigma \text{ Bool } \lambda t. \text{case } t \text{ of}$   
  true  $\rightarrow \iota \text{ Nat}$   
  false  $\rightarrow \delta \top \lambda EIA. \delta (EIA \text{ tt}) \lambda EIB. \iota ((x : EIA \text{ tt}) \rightarrow EIB \ x)$ 
```

For each signature, we postulate type formation, term formation, elimination and computation rules, for the described IR type.

# How to prove canonicity?

Every closed term with IR type should be convertible to an IR constructor. E.g. a closed term  $t : \text{Code}$  should be either  $\text{Nat}'$  or  $\Pi'$ .

Type-theoretic Artin gluing [Coq19, KHS19]:

- A closed type is interpreted as a proof-relevant predicate over its closed terms.
- A closed term is interpreted as a predicate witness.
- Open types & terms are generalized over closed substitutions.

For each type former, we have to choose a predicate that implies canonicity.

# How to prove canonicity?

**Notation:** we write  $\mathbf{Ty}$  for the set of closed types,  $\mathbf{Tm}\ A$  for sets of closed terms, and  $\mathbf{U} : \mathbf{Ty}$  for object-theoretic universes (omitting levels).

**Example:** canonicity predicate for natural numbers.

```
data  $\text{Nat}^\circ : \mathbf{Tm}\ \text{Nat} \rightarrow \mathbf{Set}$  where  
   $\text{zero}^\circ : \text{Nat}^\circ\ \text{zero}$   
   $\text{suc}^\circ : (n : \mathbf{Tm}\ \text{Nat}) \rightarrow \text{Nat}^\circ\ n \rightarrow \text{Nat}^\circ\ (\text{suc}\ n)$ 
```

Generally: the canonicity predicate for an inductive type is a **metatheoretic indexed inductive type**.



# How to prove canonicity?

**Example:** the canonicity predicate for Code is the following **metatheoretic indexed IR type**:

**data**  $\text{Code}^\circ : \text{Tm Code} \rightarrow \text{Set}$

$\text{Nat}'^\circ : \text{Code}^\circ \text{Nat}'$

$\Pi'^\circ : \{A : \text{Tm Code}\} (A^\circ : \text{Code}^\circ A)$

$\{B : \text{Tm} (\text{El } A \rightarrow \text{Code})\} (B^\circ : \{a : \text{Tm} (\text{El } a)\} \rightarrow \text{El}^\circ A^\circ a \rightarrow \text{Code}^\circ (B a))$   
 $\rightarrow \text{Code}^\circ (\Pi' A B)$

$\text{El}^\circ : \{t : \text{Tm Code}\} \rightarrow \text{Code}^\circ t \rightarrow (\text{Tm} (\text{El } t) \rightarrow \text{Set})$

$\text{El}^\circ \text{Nat}'^\circ t = \text{Nat}^\circ t$

$\text{El}^\circ (\Pi'^\circ A^\circ B^\circ) f = \{a : \text{Tm } A\} \rightarrow \text{El}^\circ A^\circ a \rightarrow \text{El}^\circ B^\circ (f a)$

If I have  $t : \text{Tm Code}$ , the gluing interpretation hands me an element of  $\text{Code}^\circ t$ .

# How to prove canonicity?

In the object theory, let's assume general IR types specified by an internal Sig type.

- Previously: the object theory supports Code, the canonicity proof involves  $\text{Code}^\circ$ .
- Now: the object theory has all IR types, the canonicity proof involves all canonicity predicates.

# How to prove canonicity?

In the object theory, let's assume general IR types specified by an internal Sig type.

- Previously: the object theory supports Code, the canonicity proof involves  $\text{Code}^\circ$ .
- Now: the object theory has all IR types, the canonicity proof involves all canonicity predicates.

Interpreting IR types in the glued model:

- We write  $\text{IR } S : \text{Tm } U$  for the IR *type formation* rule, for  $S : \text{Tm } (\text{Sig } O)$ .

# How to prove canonicity?

In the object theory, let's assume general IR types specified by an internal Sig type.

- Previously: the object theory supports Code, the canonicity proof involves  $\text{Code}^\circ$ .
- Now: the object theory has all IR types, the canonicity proof involves all canonicity predicates.

Interpreting IR types in the glued model:

- We write  $\text{IR } S : \text{Tm } U$  for the IR *type formation* rule, for  $S : \text{Tm } (\text{Sig } O)$ .
- To interpret  $\text{IR } S$ :
  - By induction hypothesis, we get  $S^\circ : \text{Sig}^\circ S$  witnessing the canonicity of the signature  $S$  itself.
  - We compute a metatheoretic indexed IR signature by induction on  $S^\circ$ . This yields the canonicity predicate for  $\text{IR } S$ .

# How to prove canonicity?

In the object theory, let's assume general IR types specified by an internal Sig type.

- Previously: the object theory supports Code, the canonicity proof involves  $\text{Code}^\circ$ .
- Now: the object theory has all IR types, the canonicity proof involves all canonicity predicates.

Interpreting IR types in the glued model:

- We write  $\text{IR } S : \text{Tm } U$  for the IR *type formation* rule, for  $S : \text{Tm } (\text{Sig } O)$ .
- To interpret  $\text{IR } S$ :
  - By induction hypothesis, we get  $S^\circ : \text{Sig}^\circ S$  witnessing the canonicity of the signature  $S$  itself.
  - We compute a metatheoretic indexed IR signature by induction on  $S^\circ$ . This yields the canonicity predicate for  $\text{IR } S$ .
- We also need to interpret *term formation*, *elimination* and *computation* rules.
- For these, we have to show that universal properties of IR types are preserved through the indexed IR encoding.

# How to prove canonicity?

The construction is a moderately technical “generic programming” exercise, where we have to do some tricky induction over signatures.

We use metatheoretic IR to show canonicity of object-theoretic IR. The metatheory has to have more universes; in our case the metatheory has extra levels  $\omega$  and  $\omega + 1$ <sup>1</sup>.

---

<sup>1</sup>But  $\omega + 1$  is only for convenience and could be omitted.

# Indexed Induction-Recursion

What about canonicity for **indexed** IR types?

We only show canonicity for plain IR types, but also show that indexed IR types are *constructible* in  $\text{MLTT} + \text{IR}$ , with *definitional computation rules*.

We use the well-known construction that converts indices to parameters and propositional identities.<sup>2</sup>

Here, since we work in plain MLTT without function extensionality and UIP, we have to do a modest amount of HoTT reasoning.

---

<sup>2</sup>Popularized as “fording” by Conor McBride.

## Closing notes

- Future work: normalization for IR types. For this, we first have to show that presheaf models have IR types (which is already quite technical!).
- Paper: under review, I'll publish a preprint in a few weeks.



- Future work: normalization for IR types. For this, we first have to show that presheaf models have IR types (which is already quite technical!).
- Paper: under review, I'll publish a preprint in a few weeks.

**Thank you!**

Andreas Abel, Nils Anders Danielsson, and Oskar Eriksson.

A graded modal dependent type theory with a universe and erasure, formalized.

*Proc. ACM Program. Lang.*, 7(ICFP):920–954, 2023.

Andreas Abel, Joakim Öhman, and Andrea Vezzosi.

Decidability of conversion for type theory in type theory.

*Proc. ACM Program. Lang.*, 2(POPL):23:1–23:29, 2018.

Ana Bove and Venanzio Capretta.

Nested general recursion and partiality in type theory.

In Richard J. Boulton and Paul B. Jackson, editors, *Theorem Proving in Higher Order Logics, 14th International Conference, TPHOLs 2001, Edinburgh, Scotland, UK, September 3-6, 2001, Proceedings*, volume 2152 of *Lecture Notes in Computer Science*, pages 121–135. Springer, 2001.

Marcin Benke, Peter Dybjer, and Patrik Jansson.

Universes for generic programs and proofs in dependent type theory.

*Nord. J. Comput.*, 10(4):265–289, 2003.

Thierry Coquand.

Canonicity and normalization for dependent type theory.

*Theor. Comput. Sci.*, 777:184–191, 2019.

Jonathan Chan and Stephanie Weirich.

Bounded first-class universe levels in dependent type theory.

CoRR, abs/2502.20485, 2025.

Larry Diehl.

*Fully Generic Programming over Closed Universes of Inductive-Recursive Types.*

PhD thesis, Portland State University, 2017.

Peter Dybjer and Anton Setzer.

A finite axiomatization of inductive-recursive definitions.

In Jean-Yves Girard, editor, *Typed Lambda Calculi and Applications, 4th International Conference, TLCA'99, L'Aquila, Italy, April 7-9, 1999, Proceedings*, volume 1581 of *Lecture Notes in Computer Science*, pages 129–146. Springer, 1999.

Peter Dybjer and Anton Setzer.

Induction-recursion and initial algebras.

*Ann. Pure Appl. Log.*, 124(1-3):1–47, 2003.

Peter Dybjer and Anton Setzer.

Indexed induction-recursion.

*J. Log. Algebraic Methods Program.*, 66(1):1–49, 2006.

Peter Dybjer.

A general formulation of simultaneous inductive-recursive definitions in type theory.

*Journal of Symbolic Logic*, 65:525–549, 2000.

Ambrus Kaposi, Simon Huber, and Christian Sattler.

Gluings for type theory.

In Herman Geuvers, editor, *4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany*, volume 131 of *LIPICs*, pages 25:1–25:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

András Kovács.

Generalized universe hierarchies and first-class universe levels.

In Florin Manea and Alex Simpson, editors, *30th EACSL Annual Conference on Computer Science Logic, CSL 2022, February 14-19, 2022, Göttingen, Germany (Virtual Conference)*, volume 216 of *LIPICs*, pages 28:1–28:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

András Kovács.

Large Countable Ordinals in Agda, 2023.

Chou Kyuhei.

Brouwer tree barrier ordinal, 2025.

Per Martin-Löf.

An intuitionistic theory of types: Predicative part.

In *Studies in Logic and the Foundations of Mathematics*, volume 80, pages 73–118. Elsevier, 1975.

Per Martin-Löf.

*Intuitionistic type theory*, volume 1 of *Studies in Proof Theory*.

Bibliopolis, 1984.

Loïc Pujet and Nicolas Tabareau.

Impredicative observational equality.

*Proc. ACM Program. Lang.*, 7(POPL):2171–2196, 2023.