Eliminating finitary inductive-inductive types without K

Szumi Xie

Eötvös Loránd University (ELTE)

TYPES 2026, Göteborg

Example of an inductive-inductive type (IIT)

$$
\begin{aligned}
\mathsf{Con} &: \mathsf{Type} \\
\mathsf{Ty} &: \mathsf{Con} \to \mathsf{Type} \\
\\
\mathsf{nil} &: \mathsf{Con} \\
\mathsf{ext} &: (\Gamma : \mathsf{Con}) \to \mathsf{Ty}\ \Gamma \to \mathsf{Con} \\
\\
\mathsf{univ} &: (\Gamma : \mathsf{Con}) \to \mathsf{Ty}\ \Gamma \\
\mathsf{el} &: (\Gamma : \mathsf{Con}) \to \mathsf{Ty}\ (\mathsf{ext}\ \Gamma\ (\mathsf{univ}\ \Gamma)) \\
\mathsf{pi} &: (\Gamma : \mathsf{Con}) \to (A : \mathsf{Ty}\ \Gamma) \to \mathsf{Ty}\ (\mathsf{ext}\ \Gamma\ A) \to \mathsf{Ty}\ \Gamma
\end{aligned}
$$

The elimination principle of an IIT

Given

$$
\begin{aligned}
&P &&: \mathsf{Con} \to \mathsf{Type}\\
&Q &&: P\,\Gamma \to \mathsf{Ty}\,\Gamma \to \mathsf{Type}\\
&z &&: P\,\mathsf{nil}\\
&f &&: (x : P\,\Gamma) \to Q\,x\,A \to P\,(\mathsf{ext}\,\Gamma\,A)\\
&\vdots
\end{aligned}
$$

The eliminator consists of

$$
\begin{aligned}
&\mathsf{elimCon} &&: (\Gamma : \mathsf{Con}) \to P\,\Gamma\\
&\mathsf{elimTy} &&: (A : \mathsf{Ty}\,\Gamma) \to Q\,(\mathsf{elimCon}\,\Gamma)\,A\\
&\mathsf{nil}\beta &&: \mathsf{elimCon}\,\mathsf{nil} = z\\
&\mathsf{ext}\beta &&: \mathsf{elimCon}\,(\mathsf{ext}\,\Gamma\,A) = f\,(\mathsf{elimCon}\,\Gamma)\,(\mathsf{elimTy}\,A)\\
&\vdots
\end{aligned}
$$

Related work

	general elimination?	without K?
Nordvall Forsberg (2013)	no	no
Hugunin (2019)	no	yes
Kaposi, Kovács & Lafont (2020)	yes	no
Sestini (2023)	yes	no
current work	yes	yes

K (or UIP) means all types are h-sets, incompatible with HoTT

The main idea

Con–Ty (day o)

Con–Ty (day 1)

$$\begin{aligned} \mathrm{Con}_1 &: \mathrm{Type} \\ \mathrm{nil}_1 &: \mathrm{Con}_1 \end{aligned}$$

Con–Ty (day 2)

$$
\begin{aligned}
\mathsf{Con}_2 &: \mathsf{Type} \\
\mathsf{nil}_2 &: \mathsf{Con}_2 \\
\mathsf{Ty}_2 &: \mathsf{Con}_1 \to \mathsf{Type} \\
\mathsf{univ}_2 &: (\Gamma : \mathsf{Con}_1) \to \mathsf{Ty}_2\ \Gamma
\end{aligned}
$$

Con–Ty (day 3)

$$
\begin{aligned}
\mathsf{Con}_3 &: \mathsf{Type} \\
\mathsf{nil}_3 &: \mathsf{Con}_3 \\
\mathsf{Ty}_3 &: \mathsf{Con}_2 \to \mathsf{Type} \\
\mathsf{univ}_3 &: (\Gamma : \mathsf{Con}_2) \to \mathsf{Ty}_3\ \Gamma \\
\mathsf{ext}_3 &: (\Gamma : \mathsf{Con}_1) \to \mathsf{Ty}_2\ \Gamma \to \mathsf{Con}_3
\end{aligned}
$$

Con–Ty (day 4)

$$
\begin{aligned}
\mathsf{Con}_4 &: \mathsf{Type} \\
\mathsf{nil}_4 &: \mathsf{Con}_4 \\
\mathsf{Ty}_4 &: \mathsf{Con}_3 \to \mathsf{Type} \\
\mathsf{univ}_4 &: (\Gamma : \mathsf{Con}_3) \to \mathsf{Ty}_4\ \Gamma \\
\mathsf{ext}_4 &: (\Gamma : \mathsf{Con}_2) \to \mathsf{Ty}_3\ \Gamma \to \mathsf{Con}_4 \\
\mathsf{el}_4 &: (\Gamma : \mathsf{Con}_1) \to \mathsf{Ty}_4\ (\mathsf{ext}_3\ \Gamma\ (\mathsf{univ}_2\ \Gamma))
\end{aligned}
$$

Con–Ty (day 5)

$$
\begin{aligned}
\mathsf{Con}_5 &: \mathsf{Type} \\
\mathsf{nil}_5 &: \mathsf{Con}_5 \\
\mathsf{Ty}_5 &: \mathsf{Con}_4 \to \mathsf{Type} \\
\mathsf{univ}_5 &: (\Gamma : \mathsf{Con}_4) \to \mathsf{Ty}_5\ \Gamma \\
\mathsf{ext}_5 &: (\Gamma : \mathsf{Con}_3) \to \mathsf{Ty}_4\ \Gamma \to \mathsf{Con}_5 \\
\mathsf{el}_5 &: (\Gamma : \mathsf{Con}_2) \to \mathsf{Ty}_5\ (\mathsf{ext}_4\ \Gamma\ (\mathsf{univ}_3\ \Gamma)) \\
\mathsf{pi}_5 &: (\Gamma : \mathsf{Con}_1) \to (A : \mathsf{Ty}_2\ \Gamma) \to \mathsf{Ty}_4\ (\mathsf{ext}_3\ \Gamma\ A) \to \mathsf{Ty}_5\ \Gamma
\end{aligned}
$$

Con–Ty (day ∞)

$$
\begin{aligned}
\mathsf{Con}_\infty &: \mathsf{Type} \\
\mathsf{nil}_\infty &: \mathsf{Con}_\infty \\
\mathsf{Ty}_\infty &: \mathsf{Con}_\infty \to \mathsf{Type} \\
\mathsf{univ}_\infty &: (\Gamma : \mathsf{Con}_\infty) \to \mathsf{Ty}_\infty\ \Gamma \\
\mathsf{ext}_\infty &: (\Gamma : \mathsf{Con}_\infty) \to \mathsf{Ty}_\infty\ \Gamma \to \mathsf{Con}_\infty \\
\mathsf{el}_\infty &: (\Gamma : \mathsf{Con}_\infty) \to \mathsf{Ty}_\infty\ (\mathsf{ext}_\infty\ \Gamma\ (\mathsf{univ}_\infty\ \Gamma)) \\
\mathsf{pi}_\infty &: (\Gamma : \mathsf{Con}_\infty) \to (A : \mathsf{Ty}_\infty\ \Gamma) \to \mathsf{Ty}_\infty\ (\mathsf{ext}_\infty\ \Gamma\ A) \to \mathsf{Ty}_\infty\ \Gamma
\end{aligned}
$$

Details

A shortcut to construct closed IITs

1. Construct the IIT using presyntax and well-formedness predicates
2. Define height functions by recursion on the presyntax
3. Construct the elimination principle bottom-up by induction on the heights

Height functions

$$
\begin{aligned}
\mathsf{h}_{\mathsf{Con}}(\mathsf{nil}) &= 1 \\
\mathsf{h}_{\mathsf{Con}}(\mathsf{ext}\ \Gamma\ A) &= \mathsf{h}_{\mathsf{Ty}}(A) + 1 \\
\\
\mathsf{h}_{\mathsf{Ty}}(\mathsf{univ}\ \Gamma) &= \mathsf{h}_{\mathsf{Con}}(\Gamma) + 1 \\
\mathsf{h}_{\mathsf{Ty}}(\mathsf{el}\ \Gamma) &= \mathsf{h}_{\mathsf{Con}}(\Gamma) + 3 \\
\mathsf{h}_{\mathsf{Ty}}(\mathsf{pi}\ \Gamma\ A\ B) &= \mathsf{h}_{\mathsf{Ty}}(B) + 1
\end{aligned}
$$

The height-bounded eliminator

Given

$$
\begin{aligned}
&P &&: \mathsf{Con} \to \mathsf{Type}\\
&Q &&: P\ \Gamma \to \mathsf{Ty}\ \Gamma \to \mathsf{Type}\\
&z &&: P\ \mathsf{nil}\\
&f &&: (x : P\ \Gamma) \to Q\ x\ A \to P\ (\mathsf{ext}\ \Gamma\ A)\\
&\vdots
\end{aligned}
$$

An eliminator bounded by height $n : \mathbb{N}$ consists of

$$
\begin{aligned}
&\mathsf{elimCon}_n &&: (\Gamma : \mathsf{Con}) \to \{\mathsf{h}(\Gamma) \leq n\} \to P\ \Gamma\\
&\mathsf{elimTy}_n &&: (A : \mathsf{Ty}\ \Gamma) \to \{\mathsf{h}(A) \leq n\} \to Q\ (\mathsf{elimCon}_n\ \Gamma)\ A\\
&\mathsf{nil}\beta_n &&: \{1 \leq n\} \to \mathsf{elimCon}_n\ \mathsf{nil} = z\\
&\mathsf{ext}\beta_n &&: \{\mathsf{h}(A) + 1 \leq n\} \to \mathsf{elimCon}_n\ (\mathsf{ext}\ \Gamma\ A) = f\ (\mathsf{elimCon}_n\ \Gamma)\ (\mathsf{elimTy}_n\ A)\\
&\vdots
\end{aligned}
$$

Defining height-bounded eliminators

Trivially, we have an eliminator bounded by 0

Given an eliminator bounded by n, we can define an eliminator bounded by $n+1$

$$\begin{aligned}
&\text{elimCon}_{n+1}\ \text{nil} &&= x \\
&\text{elimCon}_{n+1}\ (\text{ext}\ \Gamma\ A) &&= f\ (\text{elimCon}_n\ \Gamma)\ (\text{elimTy}_n\ A) \\
&\vdots
\end{aligned}$$

Defining the eliminator

$$
\begin{aligned}
\mathsf{elimCon}\ \Gamma &= \mathsf{elimCon}_{\mathsf{h}(\Gamma)}\ \Gamma \\
\mathsf{elimTy}\ A &= \mathsf{elimTy}_{\mathsf{h}(A)}\ A \\
&\vdots
\end{aligned}
$$

Need to prove $\mathsf{elimCon}_{\mathsf{h}(A)}\ \Gamma = \mathsf{elimCon}_{\mathsf{h}(\Gamma)}\ \Gamma$, etc.

Conclusion

Current progress

We constructed the example $\mathbb{IIT}$

Beta rules for the eliminator only hold up to the identity type

Formalization in Cubical Agda with definitional irrelevance for $m \leq n$

Transport hell

Future work

Open IITs

More abstraction

IITs with recursively defined components:

- Syntax of the theory of IIT signatures
- Coherent syntax of axiomatic type theory
- Signatures for semisimplicial types

Coinductive-coinductive types