Eliminating Finitary Inductive-Inductive Types Without K

Szumi Xie*

Eötvös Loránd University (ELTE), Budapest, Hungary
szumi@inf.elte.hu

Introduction. Inductive-inductive types (IITs) [8] are a generalization of mutually inductive types and indexed inductive types. They allow types which are indexed over one another to be defined mutually. More precisely, the types and their constructors are allowed to be arbitrarily interleaved, where subsequent types and constructors may depend on any of the preceding ones.

IITs are the initial algebras of certain generalized algebraic theories [2] without equations. Chapman [3] used IITs to define an intrinsically typed syntax of dependent type theory.

We use the IIT shown below as the running example. It can be seen as a fragment of an intrinsically typed syntax of type theory. Note that Ty is indexed over Con, ext as a constructor of Con depends on Ty, and both el and pi depend on preceding constructors.

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

The current work follows several previous works constructing IITs in type theory. We summarize them in the following table.

	general elimination principle	without K	infinitary
Nordvall Forsberg [8]	no	no	no
Hugunin [5]	no	yes	yes
Kaposi, Kovács & Lafont [6]	yes	no	no
Sestini [9]	yes	no	yes
current work	yes	yes	no

Nordvall Forsberg and Hugunin derived only the simple elimination principle instead of the general elimination principle for IITs, which is not strong enough to prove that the IITs are the initial algebra of their corresponding generalized algebraic theories. Kaposi et al. and Sestini derived the general elimination principle, but relied on the uniqueness of identity proofs (UIP, equivalent to the axiom K), which is incompatible with homotopy type theory [10]. Sestini's reduction supports infinitary IITs, which allow recursive occurrences of types to appear in the codomain of arbitary functions, but we will only consider finitary IITs.

The goal of the current work is to construct finitary IITs and derive their general elimination principles without using UIP. We sketch the construction of the running example and its elimination principle below.

*The author was funded by the European Union (ERC, HOTT, 101170308). Views and opinions expressed are however those of the author only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

The construction. The main idea of the construction is to stratify the elements of the IIT based on a notion of height, then build up the IIT by course-of-values induction on this height. This construction is analogous to constructing simple inductive types from natural numbers using Adámek's fixed point theorem [1].

The advantage of this method is that the elements of the IIT are intrinsically typed from the beginning, unlike in the previous works, where the IITs were constructed using presyntax and typing relations. The necessity of UIP in the previous works arises when deriving the general elimination principle by induction on the presyntax, as they have to assume equations between types to ensure that the result is well typed, and these equations need to be equated in certain places.

To construct the running example, we first define a notion of height-bounded algebras for any height $n : \mathbb{N}$. The components of an algebra M are shown below. Some variables and constraints in braces are left implicit.

$$
\begin{aligned}
&\mathsf{Con}_M &&: (i : \mathbb{N}) \to \{1 \le i \le n\} \to \mathsf{Type}\\
&\mathsf{Ty}_M &&: (j : \mathbb{N}) \to \{i+1 \le j \le n\} \to \mathsf{Con}_{M,i} \to \mathsf{Type}\\
&\mathsf{nil}_M &&: \{1 \le n\} \to \mathsf{Con}_{M,1}\\
&\mathsf{ext}_M &&: \{j+1 \le n\} \to (\Gamma : \mathsf{Con}_{M,i}) \to \mathsf{Ty}_{M,j}\,\Gamma \to \mathsf{Con}_{M,j+1}\\
&\mathsf{univ}_M &&: \{i+1 \le n\} \to (\Gamma : \mathsf{Con}_{M,i}) \to \mathsf{Ty}_{M,i+1}\,\Gamma\\
&\mathsf{el}_M &&: \{i+3 \le n\} \to (\Gamma : \mathsf{Con}_{M,i}) \to \mathsf{Ty}_{M,i+3}\,(\mathsf{ext}_M\,\Gamma\,(\mathsf{univ}_M\,\Gamma))\\
&\mathsf{pi}_M &&: \{k+1 \le n\} \to (\Gamma : \mathsf{Con}_{M,i}) \to (A : \mathsf{Ty}_{M,j}\,\Gamma) \to \mathsf{Ty}_{M,k}\,(\mathsf{ext}_M\,\Gamma\,A) \to \mathsf{Ty}_{M,k+1}\,\Gamma
\end{aligned}
$$

Note that we index the sorts with their heights. With this notion of height, an element of Ty must have a greater height than its indexing Con, and constructors must have greater height than the maximum of the height of their arguments.

Trivially, there is an algebra bounded by height 0. If we have an algebra bounded by n, then we can construct an algebra bounded by $n+1$ by defining non-recursive indexed inductive types parametrized by the algebra bounded by n. By induction on n, we construct an algebra bounded by height n for any n. We call this algebra $\mathsf{HBSyn}(n)$.

We can lift elements of $\mathsf{Con}_{\mathsf{HBSyn}(n),i}$ and $\mathsf{Ty}_{\mathsf{HBSyn}(n),j}$ to $\mathsf{Con}_{\mathsf{HBSyn}(m),i}$ and $\mathsf{Ty}_{\mathsf{HBSyn}(m),j}$ for any $m \ge n$. From this point, instead of following Adámek's fixed point theorem [1] by taking the colimit or quotienting, we define Con to be $(i : \mathbb{N}) \times \mathsf{Con}_{\mathsf{HBSyn}(i),i}$, which can be seen as a normal form of the quotient. We can similarly define Ty and the constructors, utilizing the lifting functions.

Normalization and its properties can be defined by induction on the height bound, which is then needed for defining the elimination principle, which itself can be defined by induction on the height bound.

Since we do not have UIP, this construction also requires proving higher-dimensional coherence equations involving lifting and normalization.

Conclusion. This is work in progress. We are formalizing the construction of the running example, but formalization is difficult due to "transport hell".

In theory, the construction should be possible in Martin-Löf type theory with the identity type, natural numbers, universes and the other basic type formers. However, to make formalization more feasible, we use Cubical Agda [11] and definitional irrelevance [4] for the $\le$ relation on natural numbers.

A different approach is to use presyntax and typing relations as in the previous works, then we can define a height function on the presyntax akin to the height indices in the intrinsic approach. The recursive-recursive elimination principle can then be derived by course-of-values induction on the height. This bypasses some of the difficulty with the intrinsic approach, however, it can only work for closed IITs or IITs with only h-set parameters.

The elimination principle derived using this construction only has computation rules up to the identity type. Deriving the general elimination principle with definitional computation rules remains out of reach without UIP.

The step after constructing the example would be to construct the theory of signatures [7] as done by Kaposi, Kovács and Lafont [6], which would show that all IITs can be constructed. It should also be possible to further extend the construction of the theory of signatures into a coherent syntax of weak type theory [12].

References

[1] Jiří Adámek. Free algebras and automata realizations in the language of categories. *Commentationes Mathematicae Universitatis Carolinae*, 015(4):589–602, 1974.

[2] John Cartmell. Generalised algebraic theories and contextual categories. *Annals of Pure and Applied Logic*, 32:209–243, 1986.

[3] James Chapman. Type theory should eat itself. In *International Workshop on Logical Frameworks and Metalanguages: Theory and Practice (LFMTP 2008)*, volume 32 of *ENTCS*, pages 21–36. Elsevier, 2009.

[4] Gaëtan Gilbert, Jesper Cockx, Matthieu Sozeau, and Nicolas Tabareau. Definitional proof-irrelevance without K. In *Symposium on Principles of Programming Languages (POPL)*, volume 3 of *PACMPL*. Association for Computing Machinery, 2019.

[5] Jasper Hugunin. Constructing inductive-inductive types in cubical type theory. In *Foundations of Software Science and Computation Structures (FoSSaCS)*, volume 11425 of *LNCS*, pages 295–312. Springer, 2019.

[6] Ambrus Kaposi, András Kovács, and Ambroise Lafont. For finitary induction-induction, induction is enough. In *International Conference on Types for Proofs and Programs (TYPES 2019)*, volume 175 of *LIPIcs*, pages 6:1–6:30. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

[7] András Kovács. *Type-Theoretic Signatures for Algebraic Theories and Inductive Types*. PhD thesis, Eötvös Loránd University, 2022.

[8] Fredrik Nordvall Forsberg. *Inductive-Inductive Definitions*. PhD thesis, Swansea University, 2013.

[9] Filippo Sestini. *Bootstrapping Extensionality*. PhD thesis, University of Nottingham, 2023.

[10] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. https://homotopytypetheory.org/book/, Institute for Advanced Study, 2013.

[11] Andrea Vezzosi, Anders Mörtberg, and Andreas Abel. Cubical Agda: A dependently typed programming language with univalence and higher inductive types. *Journal of Functional Programming*, 31:e8, 2021.

[12] Théo Winterhalter. *Formalisation and Meta-Theory of Type Theory*. PhD thesis, Université de Nantes, 2020.