Schemas and Semantics for Higher Inductive Types

Peter LeFanu Lumsdaine

Stockholm University

Computer-aided Mathematical Proof
Isaac Newton Institute, Cambridge, July 2017

Higher Inductive Types

- Introduced 2011 (Bauer, Lumsdaine, Shulman, Warren); extensively developed and used 2012–13 IAS special year, HoTT book.
- Idea: type generated by elements and equalities/paths. (Cf. algebras presentated by generators and relations.)
- Crucial for “synthetic homotopy theory”
- Also some payoffs for set-level mathematics in type theory

Examples: pushout; torus; propositional truncation; set quotients; …Defined in HoTT book, implemented over existing proof assistants in various HoTT libraries.

Semantics seem reasonable: many important models of type theory have some HITs.

Formal general definition

Various HITs defined individually in HoTT book and elsewhere, implemented ad hoc in formalisations.

Problem

Still no general definition of HITs!

Ideal: like ordinary inductive types, should have schema of declarations for HIT's.

I.e.: rule schema paramatrised by specifications of the constructors; eliminator generated automatically.

Heuristically, works fine: can do this by hand. Surprisingly difficult to delineate formally…

Today: discuss proposed schemas, and their semantics.

Goals

Ideal general definition:

- Flexible, general scheme of declarations; cf. inductive types of CIC (Coq, Agda, Lean…)
- Practical for programming with
- Includes most/all established examples (HoTT book, etc)
- Reasonable semantic account (cf. "initial algebras for stricly positive endofunctors")
- Interpreted in important models: sets, groupoids, simplicial/cubical sets, …

Simpler alternative: "small basis".

- A few specific HITs; cf. Σ, $+$, **n**, **W** of Martin-Löf Type theory
- Sufficient to reconstruct most/all established examples
- Plausibly sufficient to reconstruct all from conjectured schema
- Inderpreted in important models

Indermediate approach: simple/restricted schema

Reducing ordinary inductive types to **W**-types

Heuristic (Folklore?)

Assume identity types, Π-types with function extensionality, Σ-types, +-types, 0 *and* 1*, and a universe closed under these. Then all the inductive types of CIC can be simulated*[1] *by* **W***-types.*

Recall: a **W**-type has a single constructor of the form $\mathbf{c}(a{:}A)(f{:}W_{A,B}^{B_a}) : \mathbf{W}_{A,B}$:

- type A parametrises the constructors (or: is non-recursive argument of constructor);
- type B_a gives recursive arguments of constructor a.

Idea: terms over a single-sorted signature with operation symbols $a{:}A$ of arities B_a.

[1] modulo "computation" rules only holding propositionally

Reducing ordinary inductive types to **W**-types

Sketch procedure.

- Mutual inductive types can be consolidated as an inductive family indexed over n.
- Inductive families indexed over I can be consolidated as (a subtype of) a single inductive type, and recovered as fibers of a function to I.
- A constructor with multiple arguments $\mathbf{c}\,(a_1{:}A_1)\,(a_2{:}A_2(a_1))\ldots$ can be replaced by a single-argument constructor $\mathbf{c}\,(a{:}\Sigma\,(x_1{:}A_1)\,A_2(x_1))\ldots$
- Multiple/individual recursive calls $(x_1, x_2{:}X)\,(f{:}X^B)$ in a constructor can be consolidated using sums as $f' : X^{1+1+B}$.
- Multiple constructors $\mathbf{c}_1\,(a_1{:}A_1)\ldots$, $\mathbf{c}_2\,(a_2{:}A_2)\ldots$ can be consolidated as $\mathbf{c}\,(a : A_1 + A_2)\ldots$

□

Formulating a schema for HIT's

How general are we aiming for?

Some features seem proper generalisations, e.g. Higher Inductive-Recursive Types. Not attempting to cover these, for now.

Other features can (we expect) be reduced to simpler forms, as with ordinary inductive types:

- all features of ordinary inductive declarations: indexing, mutuality, multiple constructors, …
- constructors for *higher* identity types can be reduced to point- and path-constructors, by hub-and-spoke
- recursive arguments in (higher) identity types $x : X, p : \mathrm{Id}_X(x, x)$ can be reduced to recursive arguments $f : X^{S^1}$ indexed by (previously-constructed) HITs

Want these, but can (hopefully) leave out of schema and recover afterwards.

Proposal: just pushouts!

First proposed small basis: pushout types only (over usual MLTT/CIC).

- Easily suffice to construct "all" non-recursive HITs
- Non-obviously suffice for many recursive HITs (van Doorn, Kraus, Rijke, …)
- Implemented in HoTT/UniMath libraries

Modelled in:

- sets and groupoids, fairly straightforwardly;
- simplicial sets and similar homotopy-theoretic models, but with some size issues (Lumsdaine–Shulman)
- cubical sets, without size issues (Coquand–Huber–??)

Proposal: just pushouts!

However, pushouts do not seem to suffice to construct “all HITs”:

Proposition (Blass)

Suppose “ZFC + a proper class of strongly compact cardinals” is consistent. Then ZF does not prove “every infinitary equational theory has an initial algebra”.

Such initial algebras can be presented as HITs, so (under the given consistency assumption) ZF cannot prove the set model supports these HITs. But ZF certainly proves the set model supports pushouts.

Corollary

Under the above consistency assumption, pushouts do not imply existence of free algebras for infinitary equational theories, over MLTT.

(Specifically, Blass’s counterexample, modified in Lumsdaine–Shulman, is a certain single-sorted algebraic theory, with finitely many finitary/$\mathbf{N}$-ary operations, and $\mathbf{N}^{\mathbf{N}}$-many $\mathbf{N}$-ary equations.)

Anatomy of a HIT declaration

Relevant examples: pushout; torus.

Most issues involved, e.g. allowable constructor arguments: same as for ordinary inductive types.

Two essential new twists:

- path-constructor specification involves terms **s**, **t** giving source, target of path
- via these, later constructors can refer to earlier ones (also echoed in eliminator premises and computation rules)

Key choice in schemas: how general to allow **s**, **t** to be?

- not general enough: will fail to cover important example HITs
- too general: will be unable to write eliminator, or result will be inconsistent

Meanwhile, dependence of later constructors on earlier means constructors cannot be consolidated to a single one; also causes technical headaches due to non-canonicality of ap, etc.

Requirement: naturality

Obvious choice: let **s**, **t** be anything derivable in appropriate context.

Try: in any context, allow HIT X generated by

$$\mathbf{c}_0\,(a{:}A_0)\,(f{:}X^{B_0(a)}) : X$$
$$\mathbf{c}_1\,(a{:}A_1)\,(f{:}X^{B_1(a)}) : \mathbf{s}(a,f) = \mathbf{t}(a,f)$$

where $\mathbf{s}(a,f)$, $\mathbf{t}(a,f)$ arbitrary terms in context of:

- the ambient context;
- a type X;
- map c, typed like $\mathbf{c}_0$;
- $a{:}A_1$, and $f : X^{B_1(a)}$.

Requirement: naturality

Obvious choice: let $\mathbf{s}$, $\mathbf{t}$ be anything derivable in appropriate context.

Try: in any context, allow HIT X generated by

$$\mathbf{c}_0\,(a{:}A_0)\,(f{:}X^{B_0(a)}) : X$$
$$\mathbf{c}_1\,(a{:}A_1)\,(f{:}X^{B_1(a)}) : \mathbf{s}(a,f) = \mathbf{t}(a,f)$$

where $\mathbf{s}(a,f)$, $\mathbf{t}(a,f)$ arbitrary terms in context of:

- the ambient context;
- a type X;
- map c, typed like $\mathbf{c}_0$;
- $a{:}A_1$, and $f : X^{B_1(a)}$.

Problem: can't define the eliminator!

Concretely: assume some $p : \prod_{X:\mathsf{U}} X \longrightarrow X$, not assumed natural. Now consider X generated by $\mathbf{c}_0 : X$, and $\mathbf{c}_1 : p_X(\mathbf{c}_0) = \mathbf{c}_0$. What should elimination premises be?? Good exercise: try this!

Requirement: naturality

Issue: cannot lift **s**, **t** to dependent analogues in dependent families over the inductive type, as needed in premises of eliminator.

Such lifting is approximately equivalent to **s**, **t** being natural (up to propositional equality) with respect to maps preserving the earlier constructors.

When more constructors are added, one finds: the naturality paths need to be coherent. Usual ∞-categorical headaches: no finite amount of data seems to be enough!

Heuristic

Given declaration as above (possibly with more constructors), can define eliminators and computation rules just if the source/target terms of each constructor are ∞-natural.

Proposal: “finitary inductive types”, Dybjer–Moenaclaey

Source/target terms built from restricted grammar: may use only

- variables from recursive calls;
- earlier lower-dimensional-constructors;
- (in source/target of a 2-path) composition, identities, inverses

Achievements:

- Schema defined for path- and 2-path constructors
- Interpretation given in the groupoid model

Limitations:

- grammar of **s**, **t** seems too restricted to cover various examples, e.g. (again) free algebras for infinitary equational theories

Similar proposal also given by Basold–Geuvers–Van der Weide.

Proposal: “Higher W-types”

Rephrasing of naturality:

“Natural transformations $X^2 \longrightarrow X$, for algebras X”
$\approx$ “Natural elements of X, for algebras equipped with 2 distinguished elements”
$\approx$ “Elements of the free algebra on 2 elements”.

Analogy extends to ∞-naturality, in settings where we can state that.

In type theory: can’t state that, but we do have “elements of the free algebra”.

Proposed small basis: specify **s**, **t** using inductive type generated by all earlier constructors, plus recursive calls.

Achievements:

- plausibly “fully general” for a fixed number of constructors

Limitations:

- not defined for arbitrarily many constructors (gets nasty after about 3 or 4)
- inconvenient to recover other HITs from

HITs in Cubical Type Theory

(Coquand–Huber–Mortberg–??, work in progress)

Achievements:

- Implementation of CTT allows wide, flexible schema of HIT declarations
- Compute well; convenient to program with

Limitations:

- Only defined in implementation, not written up (?yet)
- Not clear if implemented schema is consistent: may be too wide?

Other approaches err on side of caution; this schema errs on side of experimentation.

Cell monad semantics

"A semantics for a schema not yet defined"

Theorem (Lumsdaine–Shulman)

Let $\mathcal{M}$ be an excellent[2] *Quillen model category, and $\mathbb{T}$ a cell monad with parameters on $\mathcal{M}$.*
Then the local universe splitting of the comprehension category $\mathcal{M}_f$ admits strictly stable typal initial $\mathbb{T}\text{-Alg}_f$-algebras.

General semantic construction, like "intitial algebras for polynomal endofunctors". "Cell monad": algebras described in stages, corresponding to adding constructors. "Typal initial algebras": provide models of HITs.

Corollary

($\mathcal{M}$ as above.) Then $\mathcal{M}$ models type theory with most established HITs.

---

[2] simplicial, combinatorial, right proper, LCC, all monos cofibs, cofibs stable under limits

Cell monad semantics

Limitations of cell monad semantics:

- Semantic construction used to model individual HITs in ad hoc way
- Still does not seem to suffice for "all HITs"
- Size blowup when parametrising over large contexts: small-indexed families of HITs remain small (e.g. spheres), but cannot model e.g. "universe is closed under suspension"

Cell monad semantics

Limitations of cell monad semantics:

- Semantic construction used to model individual HITs in ad hoc way
- Still does not seem to suffice for "all HITs"
- Size blowup when parametrising over large contexts: small-indexed families of HITs remain small (e.g. spheres), but cannot model e.g. "universe is closed under suspension"

Another semantics: HITs in cubical model (Coquand–Huber–??):

- Various individual HITs modelled in cubical sets: at least truncations, pushouts
- …without size blowup!
- Construction ad hoc for each HIT

Summary

Established

- Definitions of many individual HITs
- Good small basis (pushouts) giving most studied HITs
- Completely satisfactory models of these in sets, groupoids, cubical sets
- Models in wide range of homotopical settings (incl. simplicial sets), with some size blowup issues

Open

- Good general schema for HITs
- Plausible small basis for “all HITs”
- General models of established HITs without size blowup
- Models of general schema