Free Algebraic Theories as Higher Inductive Types

Henning Basold[1]*, Niels van der Weide[2], and Niccolò Veltri[3]

[1] CNRS, ENS Lyon
henning.basold@ens-lyon.fr
[2] Radboud University Nijmegen
nweide@cs.ru.nl
[3] IT University of Copenhagen
nive@itu.dk

In recent years, there has been an increasing interest in higher inductive types. There are several reasons behind this, like synthetic homotopy theory [5], implementation of rewrite rules [1], quotients [2], and other colimits. Here we are interested in applications of higher inductive types in universal algebra and category theory, and the possibility of extending inductive and coinductive types beyond strictly positive types. One of the most basic constructions in universal algebra is that of a free algebra. This construction can be carried out by using higher inductive types, as we will now briefly show.

We present some basic notions as Agda code. In what follows, we will work in Martin-Löf type theory with two basic universes $\mathcal{U}_0 : \mathcal{U}_1$. We will also require function extensionality, which can, however, be dropped if we restrict ourselves to finitary signature. The following record type defines a *signature* (also polynomial or container) in Agda.

```
record Signature : 𝒰₁ where
  sym : 𝒰₀
  ar  : sym → 𝒰₀
```

From a signature Σ : Signature, we can construct the type Term Σ X of terms (W-type) over Σ with leaves labelled in X. This type comes with the obvious iteration principle that we denote by Term-iter. An *algebraic theory* is given by a signature and a set of equations between terms over that signature. These equations may have free variables but may not use other properties than equality on the variables. Thus, we represent equations as parametric binary relations [4], as in the following record type.

```
record AlgTheory : 𝒰₁ where
  sig : Signature
  eqs : ∀ {X : 𝒰₀} → Rel (Term sig X)
```

Note that the requirement that the variable type X is in a fixed universe $\mathcal{U}_0$ will also fix the universe in which the algebras for a theory live. This is, however, not a severe constraint, as the induction principle still allows for large elimination.

Given an algebraic theory T : AlgTheory over a signature Σ, we can define what an algebra and an induction scheme for an algebra of T is. Algebras are given in two steps: First, we define pre-algebras that do not have to fulfil the equations of the theory, but only are an algebra for the functor induced by the signature Σ. The fact that the carrier of a pre-algebra is a set (in the sense of homotopy type theory) is technical necessity, which we would like to lift in the future. Note that this pre-algebra immediately extends to all terms by freeness. An actual algebra for T is then a pre-algebra that fulfils also the equations required by the theory. This idea is formalised in the following two record types.

*This work is supported by the European Research Council (ERC) under the EU's Horizon 2020 programme (CoVeCe, grant agreement No. 678157)

```
record PreAlgebra : 𝒰₁ where
    carrier     : 𝒰₀
    carrier-set : is-set carrier
    algebra : (s : sym Σ) (α : ar Σ s → carrier) → carrier

  algebra* : Term Σ carrier → carrier
  algebra* = Term-iter (λ x → x) algebra

record Algebra : 𝒰₁ where
    pre-algebra : PreAlgebra
    resp-eq     : ∀ {t u : Term Σ carrier} →
                  eqs t u → algebra* t == algebra* u
```

One can then also define an induction scheme for algebras of an algebraic theory, cf. [3]. With the basic definitions in place, we construct the initial algebra of an algebraic theory as higher inductive type with the following (type) constructors, where the function node* of type Term Σ (FreeAlgebra T) → FreeAlgebra T is the extension of node to terms, cf. algebra* above.

```
FreeAlgebra : (T : AlgTheory) → 𝒰₀
node        : (s : sym Σ) (α : ar Σ s → FreeAlgebra T) → FreeAlgebra T
eq          : ∀ {t u} → eqs t u → node* t == node* u
quot        : is-set (FreeAlgebra T)
```

This HIT comes with an iteration principle for T-algebras and an induction principle. These can be used to show that FreeAlgebra T is indeed the initial T-algebra. The principles are strong enough to define the free algebra functor and show it is the left adjoint of the forgetful functor. Examples of this construction include the free monoid, which can be shown to be equivalent to lists. `https://perso.ens-lyon.fr/henning.basold/code/AlgTheoryHIT`. An alternative development in UniMath can be found in [6].

Why would we be interested in such a construction of free algebras? First of all, we wish to formalise parts of universal algebras, like quotient algebras and the isomorphism theorems. Higher inductive types seem to provide the appropriate mechanism for such an endeavour. Moreover, once we can construct free algebras, we could also consider inductive and coinductive types over algebraic theories as an extension of strictly positive types. In particular, semantics of finitely branching transition systems could be obtained in the final coalgebra for the finite powerset functor, which is the free join-semilattice and therefore representable in our framework. Finally, we aim to extend the construction to HITs that are not just sets, but groupoids. This would enable us, for example, to construct free symmetric monoidal categories.

In the talk, we will present the algebras and the induction scheme in more detail, show applications of the above construction, and discuss future directions.

References

[1] T. Altenkirch and A. Kaposi. Type theory in type theory using quotient inductive types. In R. Bodík and R. Majumdar, editors, *Proc. of POPL 2016*, pages 18–29. ACM, 2016.

[2] H. Basold, H. Geuvers, and N. van der Weide. Higher Inductive Types in Programming. *J.UCS*, David Turner's Festschrift – Functional Programming: Past, Present, and Future, 2017.

[3] C. Hermida and B. Jacobs. Structural Induction and Coinduction in a Fibrational Setting. *Information and Computation*, 145:107–152, 1997.

[4] C. Hermida, U. S. Reddy, and E. P. Robinson. Logical Relations and Parametricity - A Reynolds Programme for Category Theory and Programming Languages. *ENTCS*, 303:149–180, 2014.

[5] T. Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study, 2013.

[6] N. van der Weide and H. Geuvers. The Construction of Set-Truncated Higher Inductive Types. Submitted.