Non-Recursive Truncations

Nicolai Kraus

University of Nottingham
nicolai.kraus@nottingham.ac.uk

Abstract

I discuss non-recursive higher inductive types in homotopy type theory, and universal properties of truncations with respect to arbitrary types.

Homotopy type theory, often abbreviated as *HoTT*, is a branch of intensional Martin-Löf type theory based on the observation that types can be interpreted as some form of spaces. Besides univalent universes, the so-called *higher inductive types* (HITs) are a major new concept which is considered in HoTT. These are a powerful generalisation of inductive types. A HIT has not only constructors which define elements (*point constructors*), it may also have constructors that define equalities (*higher* or *path constructors*). A popular example is the circle $\mathbb{S}^1$, which we can represent by a point constructor $\mathsf{base} : \mathbb{S}^1$ and a path constructor $\mathsf{loop} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base}$. Another seemingly innocent HIT is the one implementing the *propositional truncation*: For any type A, we have the HIT $\|A\|$ with one point constructor $|-| : A \to \|A\|$ and one path constructor $\mathsf{t} : \Pi_{x,y:\|A\|}\, x = y$. The propositional truncation is probably the most prominent concept that can (non-trivially) be implemented as a HIT (see e.g. Awodey's and Bauer's *bracket types* in extensional type theory [1]). The type $\|A\|$ represents the *proposition* (i.e. has at most one inhabitant) that A is inhabited without requiring a concrete element of A to be specified. Many more examples of HITs can be found in the standard reference [5].

The HIT $\mathbb{S}^1$ is somewhat easy to visualise geometrically: first, we have a point; and second, we have a path from this point to itself. This is a particularly easy example of a finitely represented CW-complex (which already form a very well-behaved class of spaces themselves). The universal property of $\mathbb{S}^1$ says that, for any type X, functions $\mathbb{S}^1 \to X$ correspond exactly to pairs $\Sigma\,(x : X)\,.\,x = x$.

General HITs are much harder to understand because of the higher *inductive* component. We say that $\mathbb{S}^1$ is a *non-recursive* HIT because no constructor quantifies over elements of $\mathbb{S}^1$ itself. In the presentation of $\|A\|$ that we have given, the constructor t does quantify over elements of $\|A\|$ itself. This has the consequence that the induction and recursion principle, and the universal property, are somewhat restricted: a priori, we know that functions $\|A\| \to X$ correspond to functions $A \to X$, but only if X is propositional itself (i.e. fulfils the condition posed by the constructor t).

The described situation gives rise to two related questions for a given ("recursive") HIT H. First, can we formulate a version of H's universal property (or elimination principle) which is applicable when eliminating into *any* type? And, second, can H be represented as a non-recursive HIT, and does this give rise to a *useful* elimination principle into general types?

These questions are unsolved for a general H, but some answers have been found for the canonical example – the propositional truncation. In this talk, I want to compare three different constructions and results on this topic. The first can be found in my own article [3] (TYPES'14 post-proceedings), where I have shown a correspondence between *coherently constant functions* $A \to B$ and functions $\|A\| \to B$. The second is van Doorn's construction of the propositional truncation as a non-recursive HIT [6] (CPP'16). The third is my construction of the propositional truncation as a non-recursive HIT using a sequence of actual *approximations* [4] (to appear at LICS'16). In some more detail, the three mentioned constructions can be described as follows:

1. In [3], I have constructed *coherently constant functions* as morphisms between type-valued presheaves. In type-theoretic notation, a coherently constant function is given by an infinite tower of components, starting with: on level $[0]$, a function $f : A \to B$; on level $[1]$, a proof c of "weak constancy", i.e. $c : \mathsf{wconst}_f :\equiv \Pi_{a_1,a_2:A} f(a_1) = f(a_2)$; on level $[2]$, a proof of coherence for c, in the sense of $\mathsf{coh}_{f,c} :\equiv \Pi_{a_1,a_2,a_3:A}(a_1, a_2) \cdot c(a_2, a_3) = c(a_1, a_3)$; and so on.
   In the special case where B is n-truncated, all but the first $(n+2)$ components become trivial and can be omitted. This gives a reasonably clean characterisation of maps $\|A\| \to B$. We can reformulate this finite case as follows: consider the HIT $H^n(A)$ with a constructor $f : A \to H^n(A)$, a constructor $c : \mathsf{wconst}_f$, and so on ($n+2$ constructors). Then, we have the equivalence $\|H^n(A)\|_n \simeq \|A\|$. Unfortunately, we cannot write down the "full" HIT with infinitely many constructors.
2. Van Doorn has constructed $\|A\|$ as a non-recursive HIT [6]. The idea is, from my point of view, that we can simply take $H^0(A)$, i.e. drop all but the first two constructors, and make up for this by iterating H^0 infinitely often. This can be expressed as a colimit over a graph and is thus internal. A shortcoming is that this construction is not well-behaved if the iteration is done only finitely many times; as an example, already $H^0(H^0(1))$ is hard to visualise. In particular, $\left\|H^0(\ldots(H^0(A))\ldots)\right\|_n$ is nearly never equivalent to $\|A\|$.
3. My construction in [4] is an attempt to internalise an idea of [3]. At the same time, it looks similar to van Doorn's construction in that it is the colimit of a sequence. The difference is that van Doorn's construction forces everything to become equal in each step (which is much more than necessary), while mine only forces everything in some higher path space to become equal, depending on which step we are at. This results in a sequence of actual approximations of $\|A\|$, in the sense that the "connectedness level" increases in every step. The condition of the derived elimination principle is easier to satisfy than van Doorn's, from which we get a concrete consequence for the finite cases in [6]. It allows the formulation of an elimination principle for k-truncations into n-types, generalising the main result of [2].

References

[1] Steve Awodey and Andrej Bauer. Propositions as [types]. *Journal of Logic and Computation*, 14(4):447–471, 2004.

[2] Paolo Capriotti, Nicolai Kraus, and Andrea Vezzosi. Functions out of higher truncations. In Stephan Kreutzer, editor, *24th EACSL Annual Conference on Computer Science Logic (CSL) 2015)*, volume 41 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 359–373, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[3] Nicolai Kraus. The general universal property of the propositional truncation. In Hugo Herbelin, Pierre Letouzey, and Matthieu Sozeau, editors, *20th International Conference on Types for Proofs and Programs (TYPES 2014)*, volume 39 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 111–145, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[4] Nicolai Kraus. Constructions with non-recursive higher inductive types. To appear in the proceedings of LICS'16, 2016.

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

[6] Floris van Doorn. Constructing the propositional truncation using non-recursive hits. In *Proceedings of the 5th ACM SIGPLAN Conference on Certified Programs and Proofs, Saint Petersburg, FL, USA, January 20-22, 2016*, pages 122–129, 2016.