15-819 Homotopy Type Theory
Lecture Notes

Robert Lewis and Joseph Tassarotti

November 11 and 13, 2013

1 Contents

These notes cover Robert Harper's lectures on Homotopy Type Theory from November 11 and 13, 2013. Discussions include Hedberg's theorem, contractibility, propositional truncation, and the "axiom" of choice.

2 Refresher: Sets and Propositions

Recall from previous lectures the definitions of sets and propositions within HoTT. A type is called a set if there is only "one way" for any two of its elements to be equal:

$$\mathsf{isSet}(A) :\equiv \prod_{x,y:A} \prod_{p,q:x=_A y} (p =_{x=_A y} q)$$

Relatedly, a type is called a proposition if it is a "subsingleton": that is, it has at most one inhabitant.

$$\mathsf{isProp}(A) :\equiv \prod_{x,y:A} (x =_A y)$$

We can define isSet in terms of isProp: a type is a set if equality for that type is propositional.

$$\mathsf{isSet}(A) \equiv \prod_{x,y:A} \mathsf{isProp}(x =_A y)$$

What it means for a type to be a set is that there are no nontrivial relationships between elements of the type. The higher homotopy structure we

have seen in previous weeks does not exist in a set, as the only paths between elements are trivial, which trivializes the higher structure. In this sense, sets (and propositions) "reclaim" some properties of classical mathematics.

Proposition 1. *For any type A, we have* $\mathsf{isProp}(\neg A)$*, where* $\neg A :\equiv A \to 0$.

Proof. We want to find

$$\_ : \prod_{x,y:\neg A} (x =_{\neg A} y).$$

Since $\neg A$ is a function type, via funext it suffices to find

$$\_ : \prod_{u:A} (x(u) =_0 y(u)).$$

We have $\lambda u.\, \mathsf{abort}_{x(u)=_0 y(u)}(x(u))$ of this type. □

From this, we can derive the (perhaps surprising) result that $\neg\neg(\neg A) \to \neg A$, even though we do not necessarily have $\neg\neg A \to A$.

3 Hedberg's Theorem

These considerations lead us to the following important theorem.

Definition 1. *A type A has* decidable equality *if one can prove of any two inhabitants of A that they are either equal or unequal.*

$$\prod_{x,y:A} (\mathsf{Id}_A(x,y) \vee \neg\mathsf{Id}_A(x,y)).$$

A type A has stable equality *if double-negation elimination holds in its identity type:*

$$\prod_{x,y:A} (\neg\neg\mathsf{Id}_A(x,y) \to \mathsf{Id}_A(x,y))$$

Theorem 1. *A type with decidable equality is a set.*

Proof. The proof of Hedberg's theorem goes in two parts:

1. Decidable equality implies stable equality. In fact, we can prove in general that for any type A, $(A + \neg A) \to \neg\neg A \to A$. This part is simple and left as an exercise.
2. Stable equality implies sethood. This is the heart of Hedberg's theorem.

We prove 2. Suppose $h : \prod x, y : A.(\neg\neg x =_A y) \to (x =_A y)$ is evidence that equality in A is stable. To show $\mathsf{isSet}(A)$, is suffices to show that $x : A, p : x =_A x \vdash p =_{x=_A x} \mathsf{refl}_A(x)$, since we can reduce the identity of $p, q : x =_A y$ to showing that $p \cdot q^{-1} = \mathsf{refl}_A(x)$.

Fix $x : A$. We then have $h(x) : \prod y : A.(\neg\neg x =_A y) \to (x =_A y)$.

Using dependent function application apd (defined previously), we see that

$$\mathsf{apd}_{h(x)}(p) : p_*(h(x)(x)) =_{(\neg\neg x =_A x) \to (x =_A x)} h(x)(x)$$

By lemma 2.9.6 of [1], it follows that for that for any $r : \neg\neg(x =_A x)$,

$$p_*(h(x)(x))(r) =_{x=_A x} h(x)(x)(p_* r).$$

Next, from a proven property of transport in identity types, we have that

$$p_*(h(x)(x))(r) =_{x=_A x} h(x)(x)(r) \cdot p$$

and because negated types are propositions (from above),

$$h(x)(x)(p_* r) =_{x=_A x} h(x)(x)(r)$$

so we have by transitivity

$$h(x)(x)(r) \cdot p =_{x=_A x} h(x)(x)(r)$$

and cancellation gives us $p =_{x=_A x} \mathsf{refl}_A(x)$ as desired. □

For an example of the power of Hedberg's theorem, note that it implies $\mathsf{isSet}(\mathsf{Nat})$. Using double induction, one can show

$$\prod_{x,y:\mathsf{Nat}} (x =_{\mathsf{Nat}} y) \vee \neg(x =_{\mathsf{Nat}} y).$$

This is left as an exercise.

4 More Results on Propositions and Sets

Theorem 2. *Every proposition is a set:*

$$\text{If } \prod_{x,y:A} x =_A y \text{ then } \prod_{x,y:A} \prod_{p,q:x=_A y} p =_{x=_A y} q$$

Proof. Suppose $f : \prod x, y : A.x =_A y$. Given two inhabitants of A, f returns a path between them.

Fix $x_0 : A$ and let $g :\equiv f(x_0) : \prod y : A.x_0 =_A y$. For $p : y =_A y'$, we have

$$\mathsf{apd}_g(p) : p_*(g(y)) =_{x_0 =_A y'} g(y')$$

and by property of transport within identity type,

$$p_*(g(y)) =_{x_0 =_A y'} g(y) \cdot p.$$

By transitivity, we thus have

$$g(y) \cdot p =_{x_0 =_A y'} g(y')$$

$$p =_{x_0 =_A y'} g(y)^{-1} \cdot g(y)$$

For $q : y =_A y'$, these same calculations give us

$$q =_{x_0 =_A y'} g(y)^{-1} \cdot g(y).$$

Thus, $p =_{x_0 =_A y'} q$ as desired. □

Theorem 3. $\mathsf{isProp}(\mathsf{isProp}(A))$ – *that is, there is only one proof that A has only one inhabitant.*

Proof. Let $f, g : \mathsf{isProp}(A)$ be given. To show $f =_{\mathsf{isProp}(A)} g$, it suffices to show (by funext) $x, y : A \vdash \_ : fxy =_{x =_A y} gxy$. This follows since $\mathsf{isProp}(A) \to \mathsf{isSet}(A)$. □

We can similarly prove $\mathsf{isProp}(\mathsf{isSet}(A))$. For $f : A \to B$, with the proper notion of $\mathsf{isEquiv}(f)$, we will soon be able to prove $\mathsf{isProp}(\mathsf{isEquiv}(f))$. If we take equivalence to mean having a quasi-inverse, though, this is not the case.

5 Contractibility

In preparation for what follows, we define the notion of contractibility:

$$\mathsf{isContr}(A) : \sum_{x:A} \prod_{y:A} x =_A y$$

That is, a type is contractible if there is some inhabitant which all other inhabitants are equal to. This is equivalent to saying that A is a prop, and it is inhabited. Alternatively, a A type is contractibile if it is equivalent to 1.

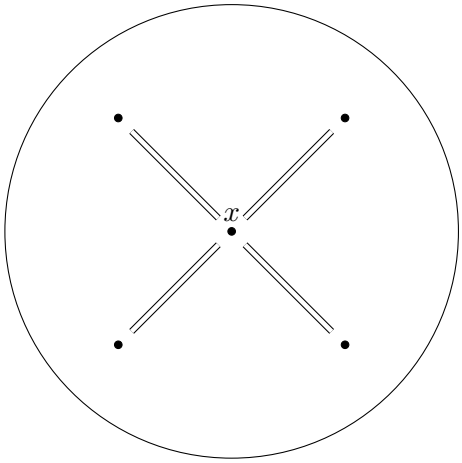

Figure 1: A contractible type. There is some element x, and paths between x and all other elements

Notice that for any type A, given $a{:}A$, we have that $\mathsf{isContr}(\Sigma x{:}A a = x)$. In particular, $(a, \mathsf{refl}_A(a)))$ is an inhabitant of this type which all other elements are equal to.

The notions of contraction and truncation are related to the (n)-type hierarchy: specifically, contractions sit at the bottom of the hierarchy. For historical reasons, we begin with (-2) types, and define inductively:

$$\begin{aligned} \mathsf{is\text{-}}(-2)\mathsf{\text{-}type}(A) &:\equiv \mathsf{isContr}(A) \\ \mathsf{is\text{-}}(n+1)\mathsf{\text{-}type}(A) &:\equiv \prod_{x,y:A} \mathsf{is\text{-}}n\mathsf{\text{-}type}(x =_A y) \end{aligned}$$

Now, we have that:

$$\mathsf{isProp}(A) \leftrightarrow \prod_{x,y:A} \mathsf{isContr}(x =_A y).$$

which implies that

$$\mathsf{isProp}(A) \leftrightarrow \mathsf{is\text{-}}(-1)\mathsf{\text{-}type}(A).$$

We can further prove the following:

$$\begin{aligned} \mathsf{isSet}(A) &\leftrightarrow \mathsf{is\text{-}0\text{-}type}(A) \\ \mathsf{isGpd}(A) &\leftrightarrow \mathsf{is\text{-}1\text{-}type}(A) \\ \mathsf{is\text{-}2\text{-}Gpd}(A) &\leftrightarrow \mathsf{is\text{-}2\text{-}type}(A) \\ &\vdots \end{aligned}$$

As well as

$$\prod_{A:\mathcal{U}} (\mathsf{is\text{-}}n\mathsf{\text{-}type}(A) \to \mathsf{is\text{-}}(n+1)\mathsf{\text{-}type}(A)) .$$

However, it is *not* the case that every type is an n-type for some n: consider $\mathcal{U}$.

6 Propositional Truncation (Squashing)

The notion of "squashing" introduced last week was perhaps too heavy-handed: it was used toward a number of goals, among them to recover classical logic within constructive logic. We now introduce the more general idea of *abstract truncation*, which for now will be taken as a primitive idea of HoTT. Truncation serves to reduce types to sets, without all the byproducts of the double negation translation.

Let $\|A\|_{-1}$ be read as the *(-1)-truncation* of A. When the context is clear, we omit the subscript. When the type $\|A\|$ is inhabited, we say A is "merely inhabited", to emphasize that this is a proof-irrelevant setting. We have the following $\|\cdot\|$-introduction rules:

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash |M| : \|A\|} \qquad \frac{}{\Gamma, x : \|A\|, y : \|A\| \vdash \mathsf{squash}(x, y) : \mathsf{Id}_{\|A\|}(x, y)}$$

As we would expect, $\mathsf{isProp}(\|A\|)$ because of the rule for $\mathsf{squash}(\cdot,\cdot)$. The corresponding elimination rule s:

$$\frac{\Gamma \vdash M : \|A\| \quad \Gamma, x : A \vdash N : B \quad \Gamma \vdash P : \mathsf{isProp}(B)}{\mathsf{elim}[B](M; x.N; P) : B}\ \|\cdot\|E$$

We require a proof of $\mathsf{isProp}(B)$ to ensure that the behavior of N does not depend on the representative $x : A$. There are other ways to ensure

this property: for instance, we could instead require $u, v : A \vdash [u/x]N =_B [v/x]N$. Note that $\mathsf{isProp}(B)$ implies that this is the case.

The β rule works as one might expect:

$$\mathsf{elim}(|M|; x.N; P) \equiv [M/x]N$$

One would similarly like some β like rule to hold for the 1-cell, squash. For example, something of the form

$$\mathsf{ap}(\lambda z.\mathsf{elim}(z; x.N; P))(\mathsf{squash}(|M|, |M'|)) \equiv P([M/x]N)([M'/x]N)$$

which corresponds to the idea that P is a proof that N is equal under substitutions of different terms of type A, because B is a proposition. However, just as in the case of seg, we do not have this.

7 Revisiting the Axiom of Choice

Previously, we explored how the axiom of choice is provable in ITT. That is, there is a term AC_∞ such that

$$AC_\infty : \prod_{A:U} \prod_{B:A\to U} \prod_{C:\Pi x:A.B\to U} \left(\left(\prod_{x:A} \sum_{y:B(x)} C(x,y) \right) \to \left(\sum_{f:\Pi x:A.B} \prod_{x:A} C(x, f(x)) \right) \right)$$

In fact, we can strengthen this and say that the two types are equivalent. Recall that this type is inhabited because the witness that the relation C is total provides precisely the choice we should make, because the specification is proof relevant. The situation is similar to the example that motivated our introduction of propositional truncation, where we wanted to write a total function that returned the index of the first occurrence of 0 in an infinite sequence. The proof that the infinite sequence actually contained a 0 also immediately told us where the 0 was.

Now that we have developed the notion of propositional truncation, we can state a version of the axiom of choice that is closer in meaning to its typical statement:

$$\prod_{A:U} \prod_{B:A\to U} \prod_{C:\Pi x:A.B\to U} \left(\mathsf{isSet}(A) \to \left(\prod_{x:A} \mathsf{isSet}(B(x)) \right) \to \left(\prod_{x:A} \prod_{y:B(x)} \mathsf{isProp}(C(x,y)) \right) \right.$$
$$\left. \to \left(\left(\prod_{x:A} \left\| \sum_{y:B_x} C(x,y) \right\| \right) \to \left\| \sum_{f:\Pi x:A.B} \prod_{x:A} C(x, f(x)) \right\| \right) \right)$$

This restated form is not provable. What has happened is now that we say that there merely exists some y for each x such that $C(x, y)$. The axiom says that given such weaker evidence, there merely exists such a function f where for each x, $C(x, f(x))$. We might call an axiom of such a type AC_{-1}, to emphasize that it involves the -1-truncation.

Now, the truncations of equivalent types are equivalent. Since the non-truncated form of the axiom of choice, AC_∞, gives us the equivalence

$$\left(\left(\prod_{x:A}\sum_{y:B(x)} C(x,y)\right) \simeq \left(\sum_{f:\Pi x:A.B}\prod_{x:A} C(x,f(x))\right)\right)$$

so that the type of AC_{-1} is equivalent to

$$\prod_{A:U}\prod_{B:A\to U}\prod_{C:\Pi x:A.B\to U}\left(\mathsf{isSet}(A) \to \left(\prod_{x:A}\mathsf{isSet}(B(x))\right) \to \left(\prod_{x:A}\prod_{y:B(x)}\mathsf{isProp}(C(x,y))\right)\right.$$
$$\left.\to \left(\left(\prod_{x:A}\left\|\sum_{y:B_x} C(x,y)\right\|\right) \to \left\|\prod_{x:A}\sum_{Y:B(x)} C(x,y)\right\|\right)\right)$$

Now, for all $Y : A \to U$, we have that $\Pi_{x:A}Y(x) \simeq \Pi_{x:A}(\Sigma_{a:Y(x)}1)$, so we can simplify the above type to:

$$\prod_{A:U}\prod_{Y:A\to U}\left(\mathsf{isSet}(A) \to \left(\prod_{x:A}\mathsf{isSet}(Y(x))\right) \to \left(\left(\prod_{x:A}||Y(x)||\right) \to \left\|\prod_{x:A}Y(x)\right\|\right)\right)$$

In this form, we see that the axiom is saying that a product of a family of merely inhabited sets is merely inhabited, which is well-known to be equivalent to the axiom of choice in classical mathematics. It is crucial here that $\mathsf{isSet}(A)$, since if A is not a set, there is a counter example (see lemma 3.8.5 in [1])

8 Equivalence and Propositions

A few weeks ago, we defined what it meant for a map $f : A \to B$ to be an equivalence between A and B, written $\mathsf{isequiv}(f)$. The definition we gave was:

$$\mathsf{isequiv}(f) :\equiv (\Sigma g : B \to A.f \circ g \sim \mathsf{id}_B) \times (\Sigma h : B \to A.h \circ f \sim \mathsf{id}_A).$$

We also gave the related notion of a quasi-inverse, written $\mathsf{qinv}(f)$, which was defined as:

$$\mathsf{qinv}(f) :\equiv \Sigma g : B \to A.(f \circ g \sim \mathsf{id}_B \times g \circ f \sim \mathsf{id}_A).$$

In some ways, the definition of qinv may at first appear to be a more natural definition than isequiv, since it states that there is some function g which is a left and right inverse of f. This is the definition of an isomorphism in category theory, for example. Of course, we explained that $\mathsf{isequiv}(f) \to \mathsf{qinv}(f)$ and $\mathsf{qinv}(f) \to \mathsf{isequiv}(f)$, and this enabled us to use whatever definition was more convenient over the past few weeks.

Why did we choose the above definition for isequiv instead of using the definition we gave for qinv? The issue is that we would like there to be only one proof that a given function f is an equivalence. That is, we want $\mathsf{isProp}(\mathsf{isequiv}(f))$ to hold for all f. This is the case for the definition we gave, but it is not true that $\mathsf{isProp}(\mathsf{qinv}(f))$ for every f. We can show this by establishing two lemmas:

Proposition 2. *If* $f : A \to B$ *and* $e : \mathsf{qinv}(f)$ *then* $\mathsf{qinv}(f) \simeq \Pi x{:}A.(x = x)$

Proposition 3. *There exists some type* X *such that* $\Pi x{:}X(x = x)$ *is not a proposition.*

See the discussion in section 4.1 in [1] for proofs of these lemmas. This makes qinv unsuitable as a definition for isequiv. Nevertheless, we would still like a definition of isequiv to be interprovable with qinv, while also being a proposition. There are three candidates which satisfy these properties, all of which are equivalent:

1. $\mathsf{biequiv}(f) :\equiv (\Sigma g : B \to A.f \circ g \sim \mathsf{id}_B) \times (\Sigma h : B \to A.h \circ f \sim \mathsf{id}_A)$.
   We say that f is *bi-invertible*, which means that f has a left inverse and a right inverse. This is the definition we have been using.
2. $\mathsf{isContr}(f) :\equiv \Pi y{:}B.\, \mathsf{isContr}(\mathsf{fib}_f(y))$ where $\mathsf{fib}_f(y) = \Sigma x{:}A.f(x) = y$.
   This says that f is contractible if given any y in the codomain, the set of all things that f maps to y (the fiber), is contractible. That is, for every point in the codomain, there is an element x in the domain such that $f(x) = y$, and if $f(x') = y$ then $x = x'$. But that precisely means that f is a bijection up to homotopy.
3. $\mathsf{ishae}(f) :\equiv \Sigma g{:}B \to A.\Sigma\alpha{:}(f \circ g \sim id).\Sigma\beta{:}(g \circ f \sim id).$
   $\Pi x{:}\ A.f(\beta x) = \alpha(fx)$:
   We read this as saying that f is a *half-adjoint equivalence*. We did not talk about this definition in class, but a discussion can be found in section

4.2 of [1]. Roughly, we can motivate this by noticing that it is similar to the definition of qinv (where the homotopies α and β were unnamed) with an additional coherence condition relating how these homotopies interact with f.

References

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