Towards third generation HOTT

Part 1: Basic syntax

Michael Shulman

University of San Diego

joint work with Thorsten Altenkirch and Ambrus Kaposi

CMU HoTT Seminar
April 28, 2022

Outline

First generation: Book HoTT

Cf. Awodey–Warren, Voevodsky, The HoTT Book

1. Based on Intensional Martin-Löf Type Theory
2. Identity types characterized by path induction
3. Univalence is an axiom

Advantages:

- Easy to do in Coq/Agda: assume univalence and away you go.
- Has models in all higher toposes.

Disadvantages:

- Not computational (UA axiom is stuck)
- Many laws are not definitional:

$$\mathrm{Id}_{A\times B}((a,b),(a',b')) \not\equiv \mathrm{Id}_A(a,a') \times \mathrm{Id}_B(b,b')$$
$$\mathrm{ap}_{f\circ g}(p) \not\equiv \mathrm{ap}_f(\mathrm{ap}_g(p))$$
$$\overrightarrow{\mathrm{tr}}^{x\mapsto A(x)\times B(x)}(p,(a,b)) \not\equiv (\overrightarrow{\mathrm{tr}}^{A}(p,a), \overrightarrow{\mathrm{tr}}^{B}(p,b))$$

Second generation: Cubical type theories

Cf. Bezem-Coquand-Huber, Cohen-Coquand-Huber-Mörtberg, Angiuli-Brunerie-Coquand-Favonia-Harper-Licata

1. Paths defined as maps out of an interval "exo-type" $\mathbb{I}$
2. Cubical Kan operations asserted explicitly in syntax
3. Univalence proved from "glue types"

Advantages:

- Satisfies canonicity and normalization
- Many equalities become definitional
- Implemented in Cubical Agda, cooltt, ...

Disadvantages:

- Not yet known to have models in higher toposes...
  ...but it probably does (cf. ACCRS, cubical model for spaces).
- ...?

What's not to like about cubical type theories?

Martin-Löf J-elim is conceptually fundamental to "equality". In Book HoTT, this simple rule automatically yields higher structure.

Slogan for Book HoTT

Homotopy is implicitly present in the foundations of mathematics.

- A nice story to tell philosophers.
- Accessible to students.

In cubical type theory, identity is defined using a homotopy-theoretic idea (paths), and higher structure is "put in by hand" (Kan ops).

- Fine if you already know you want to do homotopy theory.
- Doesn't have the same philosophical import.
- Not as accessible to students.

That interval

The interval $\mathbb{I}$ is not an ordinary type, but appears in contexts.

- Complicates the meta-theory
- Harder to explain
- Harder to implement
  - Termination-checking of boundaries
  - Display of boundaries to the user
  - Higher-order unification

We're still learning how to implement cubical type theories.
But it's also worth exploring different approaches.

A problem shared by Book HoTT and CTT

Chapter 2 of the Book characterizes lots of identity types:

$$\mathrm{Id}_{A\times B}((a,b),(a',b')) \approx \mathrm{Id}_A(a,a') \times \mathrm{Id}_B(b,b')$$

$$\mathrm{Id}_{A\to B}(f,g) \approx \textstyle\prod_{(x:A)} \mathrm{Id}_B(f(x),g(x)) \qquad \mathrm{Id}_{\mathsf{U}}(A,B) \approx \mathrm{Equiv}(A,B)$$

- In Book HoTT, these are all only equivalences.
- In cubical type theory, most of them are isomorphisms. . .
  . . . except for Id_{U}, which is still only an equivalence!

This limits the everyday usability of univalence. Given an equivalence $f : A \rightleftarrows B : g$, if we pass it through univalence we can't recover f or g definitionally, only up to homotopy.

Towards H.O.T.T.

I will describe work in progress towards a theory called

Higher Observational Type Theory

with the following properties:

- It admits models in all higher toposes, including spaces.
- Univalence "by definition" (+ other Id characterizations).
- Homotopy theory is emergent rather than explicit;
  all rules have a convincing philosophical justification.
- Computation is a reasonable hope (no obvious stuck terms).

Plan for the three talks:

1. Basic syntax of H.O.T.T.
2. Symmetries and semicartesian cubes
3. Semantics of univalent universes

Outline

Bishop's conception of set

Errett Bishop wrote that

> *A set is defined by describing exactly what must be done in order to construct an element of the set and what must be done in order to show that two elements are equal.*

MLTT follows this principle if "equal" refers to definitional equality, giving introduction and η rules:

$$\frac{a : A \qquad b : B}{(a, b) : A \times B} \qquad\qquad \frac{\pi_1 s \equiv \pi_1 t : A \qquad \pi_2 s \equiv \pi_2 t : B}{s \equiv t : A \times B}$$

The elimination and β rules are determined by "harmony" with the introduction rules.

Lower Observational Type Theory

(Lower) Observational Type Theory (Altenkirch-McBride) applies the same principle to propositional equality types Eq.

$$\frac{a : A \qquad b : B}{(a, b) : A \times B} \qquad \frac{p : \mathrm{Eq}_A(\pi_1 s, \pi_1 t) \qquad q : \mathrm{Eq}_B(\pi_2 s, \pi_2 t)}{(p \stackrel{=}{,} q) : \mathrm{Eq}_{A \times B}(s, t)}$$

But this theory is non-univalent by construction, with primitive UIP:

$$\frac{p : \mathrm{Eq}_A(x, y) \qquad q : \mathrm{Eq}_A(x, y)}{\mathrm{irr}(p, q) : \mathrm{Eq}_{\mathrm{Eq}_A(x,y)}(p, q)}.$$

Can we formulate a univalent version?

Notation and terminology

In a univalent context, we refer to identity types, with formation rule

$$\frac{A : \mathsf{U} \qquad a : A \qquad b : A}{\mathrm{Id}_A(a, b) : \mathsf{U}}$$

The elements of an identity type are identifications.

Green highlights indicate rules of H.O.T.T.

We omit unchanging ambient contexts "$\Gamma \vdash$".

Computing identity types

The OTT rule

$$\frac{p : \mathrm{Eq}_A(\pi_1 s, \pi_1 t) \qquad q : \mathrm{Eq}_B(\pi_2 s, \pi_2 t)}{(p \mathbin{;\!\!=} q) : \mathrm{Eq}_{A\times B}(s,t)}$$

says that $\mathrm{Eq}_{A\times B}$ behaves like $\mathrm{Eq}_A \times \mathrm{Eq}_B$. In a higher context, this would require infinitely many such rules for $\mathrm{Id}_{\mathrm{Id}_{A\times B}}$, etc. Instead, we make it a computation rule:

$$\mathrm{Id}_{A\times B}(s,t) \equiv \mathrm{Id}_A(\pi_1 s, \pi_1 t) \times \mathrm{Id}_B(\pi_2 s, \pi_2 t).$$

Then we can just apply the same rule multiple times:

$$\begin{aligned}\mathrm{Id}_{\mathrm{Id}_{A\times B}(s,t)}(p,q) &\equiv \mathrm{Id}_{\mathrm{Id}_A(\pi_1 s,\pi_1 t)\times \mathrm{Id}_B(\pi_2 s,\pi_2 t)}(p,q)\\ &\equiv \mathrm{Id}_{\mathrm{Id}_A(\pi_1 s,\pi_1 t)}(\pi_1 p, \pi_1 q) \times \mathrm{Id}_{\mathrm{Id}_B(\pi_2 s,\pi_2 t)}(\pi_2 p, \pi_2 q)\end{aligned}$$

Respect for equality

Here's Bishop again (paraphrased):

> *An operation f from A into B is called a* function *if we have $f(a) = f(a')$ whenever $a, a' \in A$ and $a = a'$.*

For definitional equality, MLTT has congruence rules for each primitive term former:

$$\frac{s \equiv t : A \times B}{\pi_1 s \equiv \pi_1 t : A \qquad \pi_2 s \equiv \pi_2 t : B}$$

Lower OTT similarly asserts congruence terms:

$$\frac{p : \mathrm{Eq}_{A \times B}(s, t)}{\pi_1^= p : \mathrm{Eq}_A(\pi_1 s, \pi_1 t) \qquad \pi_2^= p : \mathrm{Eq}_B(\pi_2 s, \pi_2 t)}$$

By structural induction, all terms respect both equalities.

Respect for higher equality

We instead assert a general congruence term:

$$\frac{x : A \vdash f : B \qquad p : \mathrm{Id}_A(a, a')}{\mathrm{ap}_{x.f}(p) : \mathrm{Id}_B(f[a/x], f[a'/x]).}$$

In $\mathrm{ap}_{x.f}(p)$, the variable x is bound in the term f.
(For now, B doesn't depend on A; we'll come back to that later.)

We give it computation rules on standard term-formers:

$$\mathrm{ap}_{x.(a,b)}(p) \equiv (\mathrm{ap}_{x.a}(p), \mathrm{ap}_{x.b}(p))$$

$$\mathrm{ap}_{x.\pi_1 s}(p) \equiv \pi_1 \,\mathrm{ap}_{x.s}(p) \qquad \mathrm{ap}_{x.\pi_2 s}(p) \equiv \pi_2 \,\mathrm{ap}_{x.s}(p)$$

These are well-typed by the previous rule $\mathrm{Id}_{A\times B} \equiv \mathrm{Id}_A \times \mathrm{Id}_B$.
And for higher equalities, we can apply them multiple times.

Reflexivity

Everything is the same as itself, definitionally and observationally:

$$\frac{a : A}{a \equiv a : A} \qquad \frac{a : A}{\overline{a} : \mathrm{Eq}_A(a, a)}$$

Similarly, we assert reflexivity terms:

$$\frac{a : A}{\mathrm{refl}_a : \mathrm{Id}_A(a, a)}$$

with computation rules on standard term-formers:

$$\mathrm{refl}_{(a,b)} \equiv (\mathrm{refl}_a, \mathrm{refl}_b) \qquad \mathrm{refl}_{\pi_1 s} \equiv \pi_1 \mathrm{refl}_s \qquad \mathrm{refl}_{\pi_2 s} \equiv \pi_2 \mathrm{refl}_s$$

Again, these rules are well-typed because $\mathrm{Id}_{A \times B} \equiv \mathrm{Id}_A \times \mathrm{Id}_B$.

ap on neutrals and redexes

Note that $\mathrm{ap}_{x.f}(p)$ computes on all terms that f could be, even those not headed by a constructor. The rules are consistent with computations occurring inside f; for instance,

$$\mathrm{ap}_{x.\pi_1(a,b)}(p) \equiv \pi_1\, \mathrm{ap}_{x.(a,b)}(p) \equiv \pi_1\, (\mathrm{ap}_{x.a}(p), \mathrm{ap}_{x.b}(p)) \equiv \mathrm{ap}_{x.a}(p)$$

can also be obtained by reducing $\pi_1(a, b) \equiv a$ in the bound term.

We complete the picture with rules for variables:

$$\mathrm{ap}_{x.x}(p) \equiv p \qquad \mathrm{ap}_{x.y}(p) \equiv \mathrm{refl}_y \quad (\text{if } y \text{ is a variable} \neq x)$$

Then an ap term is never* a normal form: it can always reduce. Think of it as a higher-dimensional explicit substitution "$f[\![p /\!/ x]\!]$".

* modulo some detail we'll come back to later...

Functorial laws for ap

Since ap always reduces, we can deduce by induction on terms the following "admissible" equalities:

$$\mathsf{ap}_{x.f}(\mathsf{refl}_a) \equiv \mathsf{refl}_{f[a/x]} \qquad \mathsf{ap}_{y.g}(\mathsf{ap}_{x.f}(p)) \equiv \mathsf{ap}_{x.g[f/y]}(p)$$

$$\mathsf{ap}_{x.t}(p) \equiv \mathsf{refl}_t \quad (\text{if } x \text{ does not appear in } t)$$

Outline

Towards identity of functions

The obvious rule for equality of functions is function extensionality:

$$¿ \quad \mathrm{Id}_{B\to C}(f,g) \equiv \textstyle\prod_{(y:B)} \mathrm{Id}_C(f(y),g(y)) \quad ?$$

But this is trouble for ap on application. Given $x : A \vdash f : B \to C$ and $x : A \vdash b : B$ while $p : \mathrm{Id}_A(a, a')$, we want to compute

$$\begin{aligned} \mathrm{ap}_{x.fb}(p) &: \mathrm{Id}_C((f\,b)[a/x], (f\,b)[a'/x]) \\ &\equiv \mathrm{Id}_C(f[a/x](b[a/x]), f[a'/x](b[a'/x])) \end{aligned}$$

to something involving

$$\begin{aligned} & \mathrm{ap}_{x.f}(p) : \mathrm{Id}_{B\to C}(f[a/x], g[a'/x]) \\ ¿ \quad & \equiv \textstyle\prod_{(y:B)} \mathrm{Id}_C(f[a/x](y), f[a'/x](y)) \quad ? \\ \text{and} \quad & \mathrm{ap}_{x.b}(p) : \mathrm{Id}_B(b[a/x], b[a'/x]). \end{aligned}$$

We need an equality in C between $f[a/x]$ and $f[a'/x]$ applied to different inputs $b[a/x]$ and $b[a'/x]$, but this $\mathrm{ap}_{x.f}(p)$ can't give that.

Identity of functions

A better rule is (still ignoring dependence of B on A)

$$\mathsf{Id}_{B\to C}(f,g) \equiv \textstyle\prod_{(u:B)}\prod_{(v:B)}\prod_{(q:\mathsf{Id}_B(u,v))}\mathsf{Id}_C(f(u),g(v)).$$

Once we have singleton contractibility, this will be equivalent to the naïve version. But it also gives us

$$\mathsf{ap}_{x.f}(p) : \textstyle\prod_{(u:B)}\prod_{(v:B)}\prod_{(q:\mathsf{Id}_B(u,v))}\mathsf{Id}_C(f[a/x](u), f[a'/x](v))$$
$$\mathsf{ap}_{x.b}(p) : \mathsf{Id}_B(b[a/x], b[a'/x]).$$

so we can compute

$$\mathsf{ap}_{x.fb}(p) : \mathsf{Id}_C(f[a/x](b[a/x]), f[a'/x](b[a'/x]))$$

$$\mathsf{ap}_{x.fb}(p) \equiv \mathsf{ap}_{x.f}(p)\big(b[a/x],\, b[a'/x],\, \mathsf{ap}_{x.b}(p)\big).$$

Identity of abstractions

Let $x : A, y : B \vdash t : C$, hence $x : A \vdash \lambda y.t : B \to C$. Given $p : \mathsf{Id}_A(a_0, a_1)$, we can form $\mathsf{ap}_{x.(\lambda y.t)}(p)$, which has type

$$\mathsf{Id}_{B\to C}((\lambda y.t)[a_0/x], (\lambda y.t)[a_1/x])$$
$$\equiv \prod_{(u:B)} \prod_{(v:B)} \prod_{(q:\mathsf{Id}_B(u,v))} \mathsf{Id}_C(t[a_0/x, u/y], t[a_1/x, v/y])$$

How do we compute this?

Identity of abstractions

Let $x : A, y : B \vdash t : C$, hence $x : A \vdash \lambda y.t : B \to C$. Given $p : \mathrm{Id}_A(a_0, a_1)$, we can form $\mathrm{ap}_{x.(\lambda y.t)}(p)$, which has type

$$\mathrm{Id}_{B\to C}((\lambda y.t)[a_0/x], (\lambda y.t)[a_1/x])$$
$$\equiv \prod_{(u:B)} \prod_{(v:B)} \prod_{(q:\mathrm{Id}_B(u,v))} \mathrm{Id}_C(t[a_0/x, u/y], t[a_1/x, v/y])$$

How do we compute this? We want to "ap" the term t on both p and q simultaneously. So we introduce a multi-variable ap:

$$\frac{x_1 : A_1, \ldots, x_n : A_n \vdash t : C \qquad p_1 : \mathrm{Id}_{A_1}(a_1, b_1) \quad \cdots \quad p_n : \mathrm{Id}_{A_n}(a_n, b_n)}{\mathrm{ap}_{x_1.\cdots x_n.t}(p_1, \ldots, p_n) : \mathrm{Id}_C(t[\vec{a}], t[\vec{b}])}$$

$$\mathrm{ap}_{x.(\lambda y.t)}(p) \equiv \lambda u.\lambda v.\lambda q.\mathrm{ap}_{x.y.t}(p, q).$$

(Still ignoring dependence in $A_1, \ldots, A_n, C$.)

Computing with multi-variable ap

Multi-variable ap computes with all the same rules we had before. The variable rules are

$$\mathsf{ap}_{x_1.\cdots x_n.x_i}(p_1, \ldots, p_n) \equiv p_i$$

$$\mathsf{ap}_{x_1.\cdots x_n.y}(p_1, \ldots, p_n) \equiv \mathsf{refl}_y \quad (\text{if } y \text{ is a variable} \notin \{x_1, \ldots, x_n\})$$

In addition, we can identify reflexivity with the 0-ary ap (no bound variables in the subscript):

$$\mathsf{refl}_a \equiv \mathsf{ap}_{().a}().$$

Then all the computation rules for refl become special cases of those for n-ary ap.

Outline

Towards definitional univalence

We want univalence to hold "by definition", meaning $\mathsf{Id}_{\mathsf{U}}(A, B)$ consists of equivalences. But what is an equivalence?

Chapter 4 of the Book discusses several possibilites:

1. Maps with contractible fibers (Voevodsky equivalences)
2. Half-adjoint equivalences
3. Bi-invertible maps

But philosophically, these all have problems:

- None feels "canonical": why choose one over another?
- None is (definitionally) symmetrical in A and B.
- Some are hard to motivate without homotopy theory *a priori*.

What is definitional univalence?

The HoTT Book gave three properties of a type $\mathrm{Equiv}(A, B)$ to be a "good notion of equivalence":

1. There is an embedding $\mathrm{Equiv}(A, B) \hookrightarrow (A \to B)$.
2. $\mathrm{QInv}(A, B) \to \mathrm{Equiv}(A, B)$ over $A \to B$.
3. $\mathrm{Equiv}(A, B) \to \mathrm{QInv}(A, B)$ over $A \to B$.

Here $\mathrm{QInv}(A, B)$ is the naïve type of "quasi-invertible maps":

$$\mathrm{QInv}(A, B) \equiv \sum_{(f:A\to B)} \sum_{(g:B\to A)} \mathrm{Id}(g \circ f, 1_A) \times \mathrm{Id}(f \circ g, 1_B).$$

Univalence ("$\mathrm{idtoeqv} : \mathrm{Id}_{\mathsf{U}}(A, B) \to \mathrm{Equiv}(A, B)$ is an equivalence") can be stated equivalently using any such Equiv (but not QInv).

What is definitional univalence?

The HoTT Book gave three properties of a type $\mathrm{Equiv}(A, B)$ to be a "good notion of equivalence":

1. There is an embedding $\mathrm{Equiv}(A, B) \hookrightarrow (A \to B)$.
2. $\mathrm{QInv}(A, B) \to \mathrm{Equiv}(A, B)$ over $A \to B$.
3. $\mathrm{Equiv}(A, B) \to \mathrm{QInv}(A, B)$ over $A \to B$.

Here $\mathrm{QInv}(A, B)$ is the naïve type of "quasi-invertible maps":

$$\mathrm{QInv}(A, B) \equiv \sum_{(f:A\to B)} \sum_{(g:B\to A)} \mathrm{Id}(g \circ f, 1_A) \times \mathrm{Id}(f \circ g, 1_B).$$

Univalence ("idtoeqv : $\mathrm{Id}_{\mathsf{U}}(A, B) \to \mathrm{Equiv}(A, B)$ is an equivalence") can be stated equivalently using any such Equiv (but not QInv).

But as soon as univalence holds, Id_U also satisfies these properties!

Can univalence ever hold non-definitionally?

What is definitional univalence, really?

Concrete definitions of $\mathrm{Equiv}(A, B)$ include maps $f : A \to B$ and $g : B \to A$ as data. It's useful to remember exactly what these are, definitionally, to compute with them.

Definition

Univalence holds definitionally (at level 1) if, for some definition

$$\mathrm{Equiv}(A, B) :\equiv \sum_{(f:A\to B)} \sum_{(g:B\to A)} \bigcirc$$ we have

$$\mathrm{Equiv}(A, B) \overset{\uparrow}{\to} \mathrm{Id}_{\mathsf{U}}(A, B) \overset{\downarrow}{\to} \mathrm{Equiv}(A, B)$$

such that $(f, g, \bigcirc_1)\uparrow\downarrow \equiv (f, g, \bigcirc_2)$. (Perhaps $\bigcirc_1 \not\equiv \bigcirc_2$.)

Can also consider higher levels, extracting homotopies as well.

Even current cubical type theories (CCHM, ABCFHL, Cubical Agda) do not satisfy this! Can't even extract f definitionally.

One-to-one correspondences

The "best" Equiv is the type of one-to-one correspondences:

$$\text{1-1-Corr}(A,B) :\equiv \sum_{(R:A\to B\to \mathsf{U})} \left(\prod_{(a:A)} \text{isContr}\left(\sum_{(b:B)} R(a,b)\right)\right) \times \left(\prod_{(b:B)} \text{isContr}\left(\sum_{(a:A)} R(a,b)\right)\right).$$

Remark

An $R : A \to B \to \mathsf{U}$ is a correspondence. It is one-to-one if each element of A or B has exactly one correspondent in the other.
(Prefer not to call it a "relation" unless it's proposition-valued.)

- Definitionally symmetric in A and B.
- A direct propositions-as-types version of classical "bijective relation" (and reduces to it when A, B are sets), so it's easy to motivate without homotopy theory.
- In a larger universe than $A, B \ldots$ but so is $\mathrm{Id}_{\mathsf{U}}(A,B)$.
- Also works really well...

1-1 correspondences vs equivalences

If $R : A \to B \to \mathsf{U}$ is 1-1, the centers of contraction yield $f : A \to B$ and $g : B \to A$, which form an equivalence.

Conversely, if $f : A \to B$ is an equivalence with inverse $g : B \to A$, we make a 1-1 correspondence by $R_f(a, b) :\equiv \mathsf{Id}_B(b, fa)$.

- $\sum_{(b:B)} \mathsf{Id}_B(b, fa)$ is contractible with center $(f\,a, \mathsf{refl}_{fa})$.
- $\sum_{(a:A)} \mathsf{Id}_B(b, fa)$ is contractible with center $(g\,b, \varepsilon_b)$.

If we re-extract an equivalence, we get f and g definitionally.

With a fancier definition of R_f, we can even remember the homotopies $\varepsilon_b : \mathsf{Id}_B(b, fgb)$ and $\eta_a : \mathsf{Id}_A(a, gfa)$.

Computing Id_{U}

For (philosophical, syntactic, and semantic) reasons (later), instead of $\mathsf{Id}_{\mathsf{U}}(A, B) \equiv \text{1-1-Corr}(A, B)$, we make Id_{U} primitive, with intro, elim, and β but no η. (Like a coinductive type with one destructor.)

$$\frac{R : \text{1-1-Corr}(A, B)}{R{\uparrow} : \mathsf{Id}_{\mathsf{U}}(A, B)} \qquad \frac{A_2 : \mathsf{Id}_{\mathsf{U}}(A_0, A_1)}{A_2{\downarrow} : \text{1-1-Corr}(A_0, A_1)}$$

$$\frac{p : \text{1-1-Corr}(A, B)}{p{\uparrow}{\downarrow} \equiv p}$$

This β rule is sufficient for definitional univalence.

$$\text{Equiv}(A, B) \to \text{1-1-Corr}(A, B) \xrightarrow{\uparrow} \mathsf{Id}_{\mathsf{U}}(A, B) \xrightarrow{\downarrow} \text{1-1-Corr}(A, B) \to \text{Equiv}(A, B)$$

Outline

Towards ∞-groupoid structure

Now every $A : \mathsf{U}$ needs some $\mathsf{refl}_A\!\downarrow : \text{1-1-Corr}(A, A)$. The obvious choice for its underlying correspondence is $\mathsf{Id}_A : A \to A \to \mathsf{U}$:

$$\pi_1(\mathsf{refl}_A\!\downarrow) \equiv \mathsf{Id}_A$$

The other parts of refl_A then give us singleton contractibility!

$$\pi_1\pi_2(\mathsf{refl}_A\!\downarrow) : \textstyle\prod_{(a:A)} \mathsf{isContr}(\sum_{(b:A)} \mathsf{Id}_A(a, b))$$
$$\pi_2\pi_2(\mathsf{refl}_A\!\downarrow) : \textstyle\prod_{(b:A)} \mathsf{isContr}(\sum_{(a:A)} \mathsf{Id}_A(a, b)).$$

In particular, this yields composition operations: given $p : \mathsf{Id}_A(a, x)$ and $q : \mathsf{Id}_A(a, y)$, we have $(x, p), (y, q) : \sum_{(b:A)} \mathsf{Id}_A(a, b)$. But $\sum_{(b:A)} \mathsf{Id}_A(a, b)$ is contractible, so we get (in particular)

$$p^{-1} \cdot q : \mathsf{Id}_A(x, y).$$

Computing ∞-groupoid structure

Now refl is supposed to compute on all terms. And for U, the "constructors" are type formers. So we must specify how to compute, e.g., $\mathsf{refl}_{A\times B}$ using refl_A and refl_B.

In the first component, this is just the computation of identity types:

$$\begin{aligned}\mathsf{refl}_{A\times B}\downarrow &\equiv (\pi_1\mathsf{refl}_{A\times B}\downarrow, \pi_2\mathsf{refl}_{A\times B}\downarrow)\\ &\equiv (\mathsf{Id}_{A\times B}, \dots)\\ &\equiv (\mathsf{Id}_A \times \mathsf{Id}_B, \dots)\\ &\equiv (\pi_1\mathsf{refl}_A\downarrow \times \pi_1\mathsf{refl}_B\downarrow, \dots).\end{aligned}$$

We give rules for the other components that compute the ∞-groupoid structure of each type former, e.g. in $A \times B$

$$(p, q)^{-1} \cdot (r, s) \equiv (p^{-1} \cdot r\,,\; q^{-1} \cdot s)$$

Dependent identity types

We can also apply non-nullary ap to terms in U.

If $z : A \vdash B : \mathsf{U}$ and $p : \mathsf{Id}_A(x, y)$, we have

$$\mathsf{ap}_{z.B}(p) : \mathsf{Id}_{\mathsf{U}}(B(x), B(y))$$
$$\pi_1(\mathsf{ap}_{z.B}(p)\downarrow) : B(x) \to B(y) \to \mathsf{U}$$

This is how we define the dependent/heterogeneous identity type:

$$\mathsf{Id}^{p}_{z.B}(u, v) :\equiv \pi_1(\mathsf{ap}_{z.B}(p)\downarrow)(u, v)$$

Rules for dependent identity types

Since ap also computes on all terms, we give rules like

$$\mathsf{Id}^{p}_{z.B\times C}(u,v) \equiv \mathsf{Id}^{p}_{z.B}(\pi_1 u, \pi_1 v) \times \mathsf{Id}^{p}_{z.C}(\pi_2 u, \pi_2 v).$$

The rule $\mathsf{ap}(\mathsf{refl}) \equiv \mathsf{refl}$ implies that heterogeneous identity types over refl reduce to homogeneous ones:

$$\mathsf{Id}^{\mathsf{refl}_a}_{z.B}(u,v) \equiv \mathsf{Id}_{B[a/z]}(u,v).$$

Similarly, the rule $\mathsf{ap}_{\mathsf{constant}}(p) \equiv \mathsf{refl}$ implies that dependent identity types in constant families also reduce to homogeneous ones:

$$\mathsf{Id}^{p}_{z.B}(u,v) \equiv \mathsf{Id}_{B}(u,v) \qquad (\text{if } z \text{ doesn't appear in } B)$$

Finally, functoriality of ap gives

$$\mathsf{Id}^{\mathsf{ap}_{x.f}(p)}_{z.C}(u,v) \equiv \mathsf{Id}^{p}_{x.C[f/z]}(u,v)$$

Transport

Still with $z : A \vdash B : \mathrm{U}$ and $p : \mathrm{Id}_A(x, y)$, we also have $\pi_2(\mathrm{ap}_{z.B}(p)\downarrow)$ proving $\mathrm{Id}^p_B : B(x) \to B(y) \to \mathrm{U}$ is one-to-one.

In particular, we have transport: each $u : B(x)$ corresponds to some $\overrightarrow{\mathrm{tr}}^p_{z.B}(u) : B(y)$, with $\overrightarrow{\mathrm{lift}}^p_{z.B}(u) : \mathrm{Id}^p_{z.B}(u, \overrightarrow{\mathrm{tr}}^p_{z.B}(u))$.

We must give computation rules for π_2 ap, specifying how transport computes on type families:

$$\overrightarrow{\mathrm{tr}}^p_{z.B\times C}(u) \equiv (\overrightarrow{\mathrm{tr}}^p_{z.B}(\pi_1 u), \overrightarrow{\mathrm{tr}}^p_{z.C}(\pi_2 u))$$

$$\overrightarrow{\mathrm{lift}}^p_{z.B\times C}(u) \equiv (\overrightarrow{\mathrm{lift}}^p_{z.B}(\pi_1 u), \overrightarrow{\mathrm{lift}}^p_{z.C}(\pi_2 u))$$

Deriving path induction

As in cubical type theory, singleton contractibility and transport together imply Martin-Löf identity elimination J.

1. Given $C : \prod_{(x,y:A)} \mathrm{Id}_A(x,y) \to \mathrm{U}$ with $u : C(a,a,\mathrm{refl}_a)$ and $e : \mathrm{Id}_A(a,b)$, singleton contractibility yields:

$$q : \mathrm{Id}_{\sum_{(y:A)} \mathrm{Id}_A(a,y)}((a,\mathrm{refl}_a),(b,e))$$

2. Currying C to $\widetilde{C}_a : \left(\sum_{(y:A)} \mathrm{Id}_A(a,y)\right) \to \mathrm{U}$, we can transport:

$$\overrightarrow{\mathrm{tr}}^{q}_{\widetilde{C}_a}(u) : \widetilde{C}_a((b,e)) \equiv C(a,b,e).$$

Again as in cubical type theory, the β rule for J holds only typally.

Σ and Π

With dependent Id, we can define Id_Σ and Id_Π:

$$\mathsf{Id}_{\sum_{(x:A)} B(x)}(s,t) \equiv \sum_{(p:\mathsf{Id}_A(\pi_1 s, \pi_1 t))} \mathsf{Id}_B^p(\pi_2 s, \pi_2 t)$$

$$\mathsf{Id}_{\prod_{(x:A)} B(x)}(f,g) \equiv \prod_{(x:A)} \prod_{(y:A)} \prod_{(p:\mathsf{Id}_A(x,y))} \mathsf{Id}_B^p(f(x), g(y))$$

with corresponding rules for ap on their term formers,
and generalizations to dependent Id_Σ and Id_Π.

Dependent multi-variable ap and Id

Dependent Id also makes sense of dependencies in n-ary ap, using the evident derived notion of n-ary Id.

Each identification is dependent on the preceding ones.

$$\frac{\begin{array}{c} x_1{:}A_1, x_2{:}A_2(x_1), \ldots, x_n{:}A_n(x_1, \ldots, x_{n-1}) \vdash t : C(x_1, \ldots, x_n) \\ p_1 : \mathrm{Id}_{A_1}(a_1, b_1) \qquad p_2 : \mathrm{Id}^{p_1}_{x_1.A_2}(a_2, b_2) \cdots \qquad p_n : \mathrm{Id}^{p_1,\ldots,p_{n-1}}_{x_1.\ldots.x_{n-1}.A_n}(a_n, b_n) \end{array}}{\mathrm{ap}_{x_1.\ldots.x_n.t}(p_1, \ldots, p_n) : \mathrm{Id}^{p_1,\ldots,p_n}_{x_1.\ldots.x_n.C}(t[\vec{a}], t[\vec{b}])}$$

$$\mathrm{Id}^{p_1,\ldots,p_n}_{x_1.\ldots.x_n.C}(u, v) :\equiv \pi_1(\mathrm{ap}_{x_1.\ldots.x_n.C}(p_1, \ldots, p_n)\downarrow)(u, v).$$

When formalizing this, we may use a primitive notion of telescope (a context dependent on the ambient context).

Up next

Plan for the three talks:

1. Basic syntax of H.O.T.T.
2. Symmetries and semicartesian cubes
3. Semantics of univalent universes