Modalities in HoTT

A Survey I

Cosimo Perini Brogi
DiMa - Università degli Studi di Genova

Logica Categoriale ed Elementi di Teoria dei Topos

20 Sept. 2019

Outline

Table of Contents

HoTT - Univalent Foundations

Foundations of mathematics (according to V. Voevodsky):

- Syntax for mathematical objects;
- Logic (i.e. notions of proposition and proof);
- Interpretation of the syntax in the context of mathematical objects.

HoTT - Univalent Foundations

Foundations of mathematics (according to V. Voevodsky):

- Syntax for mathematical objects $\rightsquigarrow$ HoTT;
- Logic (i.e. notions of proposition and proof) $\rightsquigarrow$ h-Props;
- Interpretation of the syntax in the context of mathematical objects $\rightsquigarrow$ Types as Kan complexes.

Dependent Type Theory

Functional programming language of types and terms, mainly used in mechanized mathematics (proof-assistants, proof theory,...).

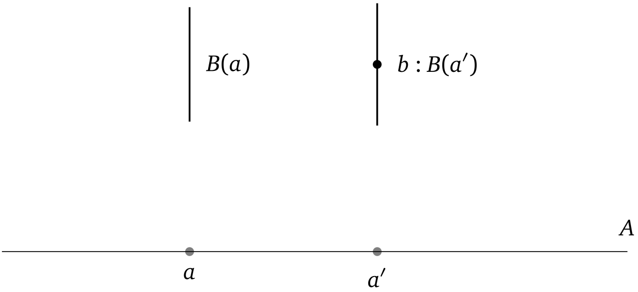
Syntax

Γ	list of variable declaration
$\Gamma \vdash A : \mathscr{U}$	A is a well-formed type in context Γ
$\Gamma \vdash a : A$	a is a well-formed term of type A
$\Gamma \vdash A \equiv B : \mathscr{U}$	A and B are judgementally equal (convertible)
$\Gamma \vdash a \equiv a' : A$	a is judgementally equal (convertible) to a' in A

Dependent Type Theory

Dependent Types

The judgement $x : A \vdash B(x) : \mathscr{U}$ defines a dependent type, thought of as family B of types indexed by A.



Dependent Type Theory

Inference Rules

An inference rule is an entailment of judgements

$$\frac{J_1 \qquad \cdots \qquad J_n}{J}$$

Well-typing

A term a is well-typed of type A in a context Γ, if there is a derivation of the judgement $\Gamma \vdash a : A$

The System

In dependent type theory there are

Structural rules: substitution, exchange, weakening;

"Logical" rules: to handle types and terms:

- Formation: how to build a type;
- Introduction: how to construct *canonical* terms of that type;
- Elimination: how to use a term of the introduced type to build other terms;
- Computation: how to "rewrite" an introduction followed by an elimination.

Universe Type(s)

Here we assume a type $\mathscr{U}$ is given. It corresponds to the type of all types.

In fact, in order to avoid paradoxes, one needs a hierarchy of universes which is cumulative and infinite.

$$\frac{}{\Gamma \vdash \mathscr{U}_i : \mathscr{U}_{i+1}} \qquad \frac{\Gamma \vdash A : \mathscr{U}_i}{\Gamma \vdash A : \mathscr{U}_{i+1}}$$

Unit Type

Formation: $1 : \mathscr{U}$;

Introduction: $\star : 1$;

Elimination: If $\Gamma, x : 1 \vdash C : \mathscr{U}$ and $c : C(\star)$, and $a : 1$, then $\mathsf{ind}_1(x.C, c, a) : C(a)$;

Computation: $\mathsf{ind}_1(x.C, c, \star) \equiv c : C(\star)$.

Empty Type

Formation: $0 : \mathscr{U}$;

Introduction:

Elimination: If $\Gamma, x : 0 \vdash C : \mathscr{U}$ and $a : 0$, then $\mathsf{ind}_0(x.C, a) : C(a)$;

Computation:

Dependent Pair Types

Formation: If $x : A \vdash B : \mathscr{U}$, then $\sum_{x:A} B(x) : \mathscr{U}$;

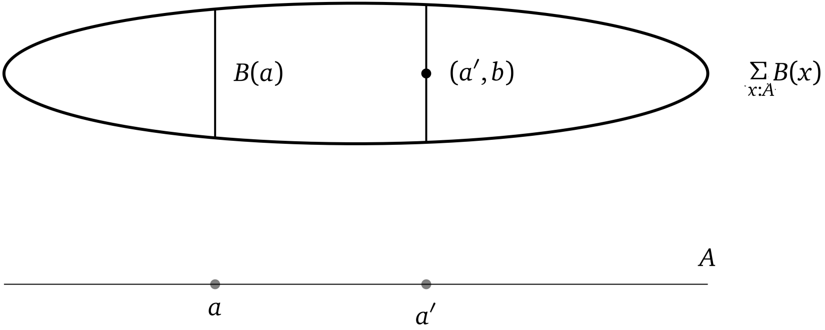
Introduction: If $a : A$ and $b : B(a)$, then $(a, b) : \sum_{x:A} B(x)$;

Elimination: If $t : \sum_{x:A} B$, then $\mathsf{fst}(t) : A$ and $\mathsf{snd}(t) : B(\mathsf{fst}(t))$;

Computation: $\mathsf{fst}(a, b) \equiv a$ and $\mathsf{snd}(a, b) \equiv b$.

Note that ordinary product type $A \times B$ is just a special case where B does not depend on A.

Dependent Pair Types



Dependent Function Types

Formation: If $x : A \vdash B : \mathscr{U}$, then $\prod_{x:A} B(x) : \mathscr{U}$;

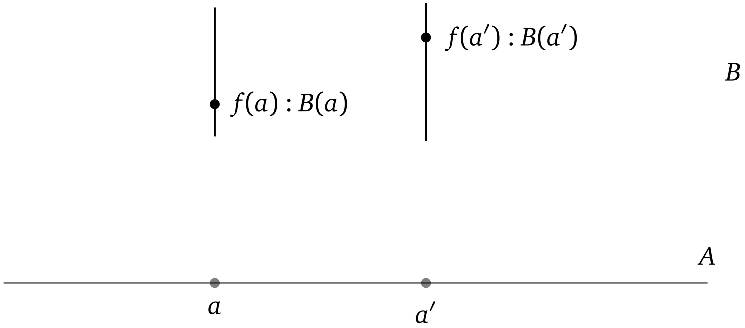
Introduction: If $a : A \vdash b : B$, then $\lambda x : A.b : \prod_{x:A} B(x)$;

Elimination: If $f : \prod_{x:A} B$ and $a : A$, then $f(a) : B(a)$;

Computation: $(\lambda x : A.b)a \equiv b[a/x]$.

Note that ordinary arrow type $A \rightarrow B$ is just a special case where B does not depend on A.

Dependent Function Types



Coproduct Types

Formation: If $A : \mathscr{U}$ and $B : \mathscr{U}$, then $A + B : \mathscr{U}$;

Introduction: If $a : A$, then $\mathsf{inl}(a) : A + B$, and if $b : B$, then $\mathsf{inr}(b) : A + B$;

Elimination: If $z : A + B \vdash C : \mathscr{U}$, $x : A \vdash c : C(\mathsf{inl}(x))$, $y : B \vdash c' : C(\mathsf{inr}(y))$, and $e : A + B$, then $\mathsf{ind}_{A+B}(z.C, x.c, y.c', e) : C(e)$;

Computation: $\mathsf{ind}_{A+B}(z.C, x.c, y.c', \mathsf{inl}(a)) \equiv c[a/x] : C(\mathsf{inl}(a))$, and $\mathsf{ind}_{A+B}(z.C, x.c, y.c', \mathsf{inr}(b)) \equiv c'[b/y] : C(\mathsf{inr}(b))$.

Natural Numbers Type

Formation: $\mathbb{N} : \mathscr{U}$;

Introduction: If $0 : \mathbb{N}$, and if $n : \mathbb{N}$ $\mathsf{succ}(n) : \mathbb{N}$;

Elimination: If $x : \mathbb{N} \vdash C : \mathscr{U}$, $c_0 : C(0)$, $x : \mathbb{N}, y : C \vdash c_s : C(\mathsf{succ}(x))$, and $n : \mathbb{N}$, then $\mathsf{ind}_{\mathbb{N}}(x.C, c_0, x.y.c_s, n) : C(n)$;

Computation: $\mathsf{ind}_{\mathbb{N}}(x.C, c_0, x.y.c_s, 0) \equiv c_0 : C(0)$, and
$\mathsf{ind}_{\mathbb{N}}(x.C, c_0, x.y.c_s, \mathsf{succ}(n)) \equiv$
$c_s[n, \mathsf{ind}_{\mathbb{N}}(x.C, c_0, x.y.c_s, n)/x, y] : C(\mathsf{succ}(n))$.

Note that elimination and computation rules allow to define functions by primitive recursion: we can define $f : \prod_{x:\mathbb{N}} C(x)$ whenever we have $c_0 : C(0)$ and $c_s : \prod_{x:\mathbb{N}} (C(x) \to C(\mathsf{succ}(x)))$ with the defining equations

$$f(0) :\equiv c_0$$

$$f(\mathsf{succ}(x)) :\equiv c_s(x, f(x))$$

Identity Types

Formation: If $a : A$ and $a' : A$, then $a =_A a' : \mathscr{U}$;

Introduction: If $a : A$, then $\mathsf{refl}(a) : a =_A a$;

Elimination: If $(x, y : A)(p : x =_A y) \vdash C(x, y, p)$ and $(x : A) \vdash t(x) : C(x, x, \mathsf{refl}(x))$, then $(x, y : A)(p : x =_A y) \vdash \mathsf{ind}_{=_A}(x.t(x), x, y, p) : C(x, y, p)$;

Computation: $\mathsf{ind}_{=_A}(x.t(x), x, x, \mathsf{refl}(x)) \equiv t(x)$.

Identity Types?

Terms in $x \underset{A}{=} y$ behave like equality

- $\mathsf{refl} : \prod_{x:A} x \underset{A}{=} x$
- $(-)^{-1} : \prod_{x,y:A} x \underset{A}{=} y \to y \underset{A}{=} x$
- $(- \cdot -) : \prod_{x,y,z:A} x \underset{A}{=} y \times y \underset{A}{=} z \to x \underset{A}{=} z$
- $\mathsf{leib} : \prod_{x,y:A} x \underset{A}{=} y \to B(x) \to B(y)$

BUT

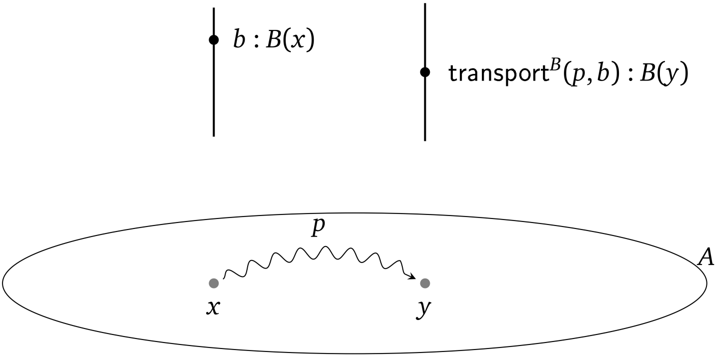
Terms in $x \underset{A}{=} y$ behave unlike equality

- $x \underset{A}{=} y,\ p \underset{x \underset{A}{=} y}{=} q,\ \ldots$
- Map $\cdot$ is associative at any level
- $\mathsf{UIP} : \prod_{x,y:A} \prod_{p,q:x \underset{A}{=} y} p \underset{x \underset{A}{=} y}{=} q$ is independent of type theory.

Terms in $x \underset{A}{=} y$ are then thought of as paths from x to y in A.

Transporting paths

$$\mathsf{transport} : \prod_{x,y:A} x \underset{A}{=} y \to \prod_{B:A\to\mathcal{U}} (B(x) \leftrightarrow B(y))$$



Mapping paths

$$\mathsf{ap} : \prod_{f:A\to B} \prod_{x,y:A} (x =_A y) \to (f(x) =_B f(y)) \text{ which is functorial.}$$

$$\mathsf{apd} : \prod_{f:\prod_{x:A} B} \prod_{x,y:A} \prod_{p:(x =_A y)} (\mathsf{transport}^B(p, f(x)) =_{B(y)} f(y))$$

$$\mathsf{transportconst} : \prod_{B:\mathcal{U}} \prod_{p:x =_A y} \prod_{b:B} \mathsf{transport}^B(p, b) = b$$

$$e : \prod_{f:A\to B} \prod_{p:x =_A y} \mathsf{apd}_f(p) = \mathsf{transportconst}^B_p(f(x)) \cdot \mathsf{ap}_f(p)$$

Voevodsky's Model

Syntax	Interpretation
$A, x \underset{A}{=} y, p \underset{x \underset{A}{=} y}{=} q, \ldots$	Kan complex A
$x : A$	$a \in A_0$
$A \to B$	Space of maps
$A \times B$	Product Space
$A + B$	Coproduct Space
$B : A \to \mathscr{U}$	Fibration $B \to A$ with fibers $B(x)$
$\sum_{x:A} B(x)$	Total space of fibration $B \to A$
$\prod_{x:A} B(x)$	Space of sections of fibration $B \to A$

Nota Bene

In set theory, a function $f : A \to B$ is a bijection iff

1. There exists $g : B \to A$ such that $g \circ f = id_A$ and $f \circ g = id_B$; iff
2. For any $b \in B$ there exists a unique $a \in A$ such that $f(a) = b$;iff
3. There exist $g : B \to A$ such that $g \circ f = id_A$, and $h : B \to A$ such that $f \circ h = id_B$.

Type Equivalences

Definition (Voevodsky)

A type A is contractible iff the following type is inhabited

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

Definition (Voevodsky)

An $f : A \to B$ is an equivalence iff the following type is inhabited

$$\mathsf{isequiv}(f) :\equiv \prod_{b:B} \mathsf{isContr}(\sum_{x:A} f(x) = b)\,.$$

Here the type of equivalences is

$$A \simeq B :\equiv \sum_{f:A \to B} \mathsf{isequiv}(f).$$

Type Equivalences

Note that the types corresponding to set-theoretic 1,2,3 are not equivalent.
However the following is equivalent to the type of equivalences:

Definition (Joyal)

$f : A \to B$ is an h-isomorphism iff the following type is inhabited

$$\mathsf{isHiso}(f) :\equiv \Big(\sum_{g:B\to A}\prod_{a:A} g(f(a)) = a\Big) \times \Big(\sum_{h:B\to A}\prod_{b:B} f(h(b)) = b\Big).$$

Paths in Types

It is now easy to prove that

$$(a,b) \underset{\sum_{x:A} B}{=} (a',b') \simeq \sum_{p:a=a'} \mathsf{transport}^B(p,b) = b'$$

and, as a special case,

$$(a,b) \underset{A\times B}{=} (a',b') \simeq (a=a') \times (b=b').$$

Remark

In dependent type theory one can also construct a map

$$\mathsf{happly} : f \underset{\prod_{x:A} B}{=} g \to f \sim g :\equiv \prod_{x:A} f(x) = g(x),$$

but the system does not suffice to build an equivalence.

Function Extensionality

Axiom

The map

$$\mathsf{happly} : f \underset{\prod_{x:A} B}{=} g \to f \sim g$$

is an equivalence for any $f, g : \prod_{x:A} B$.

Recovering Logic

Curry-Howard Correspondence

Every type is a proposition; every (well-)typed term is a proof of the corresponding proposition.

Univalent Reasoning

Some types are propositions; terms of those types are proofs.

Definition

A type $A : \mathscr{U}$ is a (h-)proposition if the following type is inhabited

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

Recovering Logic

- 0 and 1 are propositions $\rightsquigarrow \top, \bot$
- if A and B are propositions, so is $A \times B$ $\rightsquigarrow \wedge$
- if B is a proposition, so is $A \to B$ $\rightsquigarrow \supset, \neg$
- if $B(a)$ is a proposition for any $a : A$, so is $\prod_{x:A} B(x)$ $\rightsquigarrow \forall x : A$

$$\vee\,,\ \exists\ ??$$

Propositional Truncation

Formation: If $A : \mathcal{U}$, then $\|A\| : \mathcal{U}$;

Introduction: If $a : A$, then $|a| : \|A\|$ and $p(A) : \prod_{x,y:\|A\|} x = y$;

Elimination: If $f : A \to B$, and $B :$ Prop, then $|f| : \|A\| \to B$;

Computation: $|f|(|a|) \equiv f(a)$.

$$A \vee B :\equiv \|A + B\|$$

$$\exists x : A, B(x) :\equiv \|\sum_{x:A} B(x)\|$$

Homotopy Levels

0. A is contractible iff the following type is inhabited

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

1. A is a proposition iff the following type is inhabited

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

2. A is a set iff the following type is inhabited

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

Homotopy Levels

0. A is contractible iff the following type is inhabited

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

1. A is a proposition iff the following type is inhabited

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

2. A is a set iff the following type is inhabited

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

Homotopy n-Types

Definition

Define a map $\lambda n.\text{is-}n\text{-Type} : \mathbb{N} \to \mathscr{U} \to \text{Prop}$ by natural induction:

$$\text{is-}0\text{-Type}(X) :\equiv \text{isContr}(X)$$

$$\text{is-succ}(n)\text{-Type} :\equiv \prod_{x,x':X} \text{is-}n\text{-Type}(x = x')$$

Homotopy n-Types

Note that:

- If A and B are n-types, so is $A \times B$
- If B is an n-type, so is $A \to B$
- If $B(a)$ is an n-type for any $a : A$, so is $\prod_{x:A} B(x)$
- If A is an n-type and $B(a)$ is also an n-type for any $a : A$, so is $\sum_{x:A} B(x)$
- If A is an n-type, and $A \simeq B$, then B is an n-type.
- If A is an n-type, it is also an $n+1$-type.

Paths between Types

The type theory developed so far allows to build a map

$$\mathsf{idtoequiv} : (A =_{\mathcal{U}} B) \to (A \simeq B) .$$

Univalence Axiom

$$\mathsf{univalence} : \prod_{A,B:\mathcal{U}} \mathsf{isequiv}(\mathsf{idtoequiv}(A, B)) .$$

Univalent Type Theory

- If $\mathscr{U}$ is univalent, it is not a set.
- Function extensionality holds in any $\mathscr{U}$ which is univalent.
- $\mathscr{U}$ is univalent iff the type $\sum_{B:\mathscr{U}} A \simeq B$ is contractible for any $A : \mathscr{U}$.
- $\mathscr{U}$ is univalent iff for any $A : \mathscr{U}$ and every type family $P : \prod_{B:\mathscr{U}} (A \simeq B) \to \mathscr{U}$ the map:

$$\lambda f. f(A, id_A) : \left(\prod_{B:\mathscr{U}} \prod_{e:A\simeq B} P(B,e) \right) \to P(A, id_A)$$

  has a section.
- For any $P, Q :$ Prop in $\mathscr{U}$ univalent, $(P \simeq Q) \simeq (P \leftrightarrow Q)$

Table of Contents

Monads

A monad over a category $\mathscr{C}$ is given by an endofunctor $T : \mathscr{C} \to \mathscr{C}$ along with:

- A natural transformation $\eta_A : A \to T(A)$, called the unit;
- A natural transformation $\mu_A : T^2(A) \to T(A)$, called the join;

such that the following commute:

$$
\begin{tikzcd}
T^3 \arrow[r, "T\mu"] \arrow[d, "\mu_T"'] & T^2 \arrow[d, "\mu"] \\
T^2 \arrow[r, "\mu"'] & T
\end{tikzcd}
\qquad
\begin{tikzcd}
T \arrow[r, "\eta_T"] \arrow[dr, "id_T"'] & T^2 \arrow[d, "\mu"] & T \arrow[l, "T\eta"'] \arrow[dl, "id_T"] \\
& T &
\end{tikzcd}
$$

When μ is a natural isomorphism, T is called idempotent.

Eilenberg-Moore Categories

Let T be a monad over $\mathscr{C}$. A T-algebra in $\mathscr{C}$ is given by a $\mathscr{C}$-object A along with an arrow $\nu : TA \to A$ such that the following commute:

$$\begin{tikzcd} T^2(A) \arrow[r, "T\nu"] \arrow[d, "\mu_A"'] & T(A) \arrow[d, "\nu"] \\ T(A) \arrow[r, "\nu"'] & A \end{tikzcd} \qquad \begin{tikzcd} A \arrow[r, "\eta_A"] \arrow[dr, "id_A"'] & T(A) \arrow[d, "\nu"] \\ & A \end{tikzcd}$$

Then $\mathscr{C}^T$ is the category having T-algebras as objects, and a morphism between T-algebras $(A, \nu_A), (B, \nu_B)$ is given by a $\mathscr{C}$-arrow $f : A \to B$ such that

$$\nu_b \circ T(f) = f \circ \nu_A.$$

Reflective Subcategories

A reflective subcategory of $\mathscr{B}$ is a full subcategory $\mathscr{A}$ of $\mathscr{B}$ closed under object-isomorphism with $\mathscr{B}$-object and such that the inclusion functor $I : \mathscr{A} \hookrightarrow \mathscr{B}$ has a left adjoint $R : \mathscr{B} \to \mathscr{A}$ called the reflector.

Note that $\mathscr{A}$ is a reflective subcategory of $\mathscr{B}$ iff

- there exists a functor $R : \mathscr{B} \to \mathscr{B}$ with values in the full subcategory $\mathscr{A}$ such that

  $$\mathscr{A}(R(B), A) \cong \mathscr{B}(B, A)$$

  natural in B and A; iff
- for any $\mathscr{B}$-object B there is an $\mathscr{A}$-object RB and an arrow $\eta_B : B \to RB$ such that for every $\mathscr{A}$-arrow $g : B \to A$, there exists a unique $\mathscr{A}$-arrow $f : RB \to A$ such that $g = f \circ \eta_B$.

Key Equivalence

Theorem

Let $\mathscr{C}$ be a category. There is a coincidence, modulo equivalences of categories, between

1. the reflective subcategories of $\mathscr{C}$;
2. the Eilenberg-Moore categories $\mathscr{C}^T$ for the idempotent monads T over $\mathscr{C}$.

Proof.

Let $\mathscr{D}$ be a reflective subcategory of $\mathscr{C}$ with reflector R and canonical natural transformations $\alpha : Id_{\mathscr{C}} \Rightarrow I \circ R$ and $\beta : R \circ I \Rightarrow Id_{\mathscr{D}}$. Since I is fully faithful, β is a natural isomorphism. Then the monad $(I \circ R, \alpha, 1_I \beta 1_R)$ has an isomorphism as join, and it is thus idempotent. But note that giving a $\mathscr{D}$-arrow $f : X \to Y$ is equivalent to giving a $\mathscr{C}^{IR}$-arrow $f : (X, \beta_X) \to (Y, \beta_Y)$ just by naturality of β. Moreover, if a monad is idempotent, then the action v of any of its algebras is an isomorphism, thus for every $\mathscr{C}^{IR}$-object (C, v), C is indeed in $\mathscr{D}$ (by the closure condition on reflective subcategories). This means that the functor $\mathscr{D} \to \mathscr{C}^{IR}$ defined by $X \mapsto (X, \beta_X)$ and $(f : X \to Y) \mapsto f$ is indeed an equivalence of categories.
The converse is immediate, by noting that if T is an idempotent monad, then the forgetful functor $\mathscr{C}^T \to \mathscr{C}$ is fully faithful and has $F : \mathscr{C} \to \mathscr{C}^T$ as left adjoint. □

Table of Contents

Introducing Modalities

We consider types as objects of $\mathscr{U}$, and typed terms as arrows between them.
In this perspective, a modal operator $\bigcirc$ is just an endofunctor on $\mathscr{U}$ satisfying specific conditions which characterize its values as a reflective subcategory.

Reflective Subuniverses

A reflective subuniverse is given by a predicate $\mathrm{isModal} : \mathscr{U} \to \mathrm{Prop}$ such that for any $A : \mathscr{U}$ there exist a type $\bigcirc A$ and a map $\eta_A : A \to \bigcirc A$ such that $\mathrm{isModal}(\bigcirc A)$, and for any $B : \mathscr{U}$ such that $\mathrm{isModal}(B)$ the map

$$\lambda f . f \circ \eta_A : (\bigcirc A \to B) \to (A \to B)$$

is an equivalence.

Basic Properties

Lemma (1)

A type $X : \mathscr{U}$ is modal iff η_X is an equivalence.

Proof.

If $\eta_X : X \simeq \bigcirc X$, then isModal$(X) \simeq$ isModal$(\bigcirc X)$, hence X is modal, since $\bigcirc X$ is so by definition.
Conversely, if X is modal, then $id_X : X \to X$ is such that $\lambda f . f \circ id_X : (X \to Z) \to (X \to Z)$ is an equivalence for *any* $Z : \mathscr{U}$.
Now, since it is easy to see that the type of triples (Y, f, I) satisfying the defining conditions of reflective subuniverses is an h-prop, η_X is indeed an equivalence. □

Basic Properties

Lemma (2)

Every reflective subuniverse is closed under retracts. In particular, $X : \mathscr{U}$ is modal if η_X has a retraction.

Proof.

Let $f \circ \eta_X = id_X$. Then $\eta_X \circ f \circ \eta_X = \eta_X$, and by the defining condition on η_X we have $\eta_X \circ f = id_{\bigcirc X}$. Therefore η_X is an equivalence, and so X is modal. □

Lemma (3)

Every reflective subuniverse is given by a functor $\bigcirc$ up to homotopy.

Proof.

Define, for $f : A \to B$, $\bigcirc f$ as the unique map such that $\bigcirc f \circ \eta_A = \eta_B \circ f$. □

Basic Properties

Lemma (4)

For any reflective subuniverse, if $P(x)$ is modal for any $x : X$, then so is $\prod_{x:X} P(x)$.

Proof.

Let $P(x)$ be modal for any $x : X$. For $x : X$, define $\mathsf{ev}_x : \prod_{x:X} P(x) \to P(x)$ by $\mathsf{ev}_x(g) :\equiv gx$. Then we have $\psi :\equiv \lambda f.\lambda x.\eta_{P(x)} f x : \prod_{x:X} P(x) \to \prod_{x:X} \bigcirc P(x)$. In order to find a retraction f of $\eta_{\prod_{x:X} P(x)}$, note that following extensions are equivalent

$$
\begin{tikzcd}
\prod_{x:X} P(x) \arrow[r, equal] \arrow[d, "\eta"'] & \prod_{x:X} P(x) \arrow[d, "\psi"] \\
\bigcirc \prod_{x:X} P(x) \arrow[r, dotted] & \prod_{x:X} \bigcirc P(x)
\end{tikzcd}
\qquad
\begin{tikzcd}
\prod_{x:X} P(x) \arrow[r, "\mathsf{ev}_x"] \arrow[d, "\eta"'] & P(x) \arrow[d, "\eta_{P(x)}"] \\
\bigcirc \prod_{x:X} P(x) \arrow[r, dotted, "\bigcirc \mathsf{ev}"'] & \bigcirc P(x)
\end{tikzcd}
$$

Define then $f :\equiv \lambda m.\lambda x.\bigcirc(\mathsf{ev}_x) m$ as solution of the first diagram. Since $P(x)$ is modal by assumption, ψ is an equivalence, and we are done.

Basic Properties

Corollary (4.1)

The reflector $\bigcirc$ preserves finite products.

Proof.

It suffices to show that $\bigcirc X \times \bigcirc Y$ behaves just like $\bigcirc(X \times Y)$:

$$\begin{aligned} (X \times Y \to Z) &\simeq X \to (Y \to Z) \\ &\simeq X \to (\bigcirc Y \to Z) \\ &\simeq \bigcirc X \to (\bigcirc Y \to Z) \\ &\simeq \bigcirc X \times \bigcirc Y \to Z \end{aligned}$$

for any Z modal. □

Corollary (4.2)

The reflector $\bigcirc$ preserves propositions.

Proof.

P is a proposition iff $\delta : P \to P \times P$ is an equivalence. □

Uniquely Eliminating Operators

The extension condition given on η for any reflective subuniverse is like a recursion principle. If we want to "eliminate" to dependent types, we need to relax that condition.

Definition

Given $\bigcirc : \mathcal{U} \to \mathcal{U}$ and $\eta : \prod_{A:\mathcal{U}} A \to \bigcirc A$, we have a uniquely eliminating operator when

$$\lambda f. f \circ \eta_A : (\prod_{x:\bigcirc A} \bigcirc (P(x))) \to (\prod_{a:A} \bigcirc (P(\eta_A(a))))$$

is an equivalence for any $A : \mathcal{U}$ and any $P : \bigcirc A \to \mathcal{U}$. $X : \mathcal{U}$ is said to be $\bigcirc$-modal when $X \simeq \bigcirc X$.

Reflective Subuniverse from U.E.Operator

Lemma (5)

Any uniquely eliminating operator defines a reflective subuniverse.

Proof.

Let $f : A \to B$ for B modal. Then, by uniquely elimination, there exists only one $\tilde{f} : \bigcirc A \to B$ such that

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "\eta"'] & B \\ \bigcirc A \arrow[ur, dashed, "\tilde{f}"'] & \end{tikzcd}$$

It remains to show that $\bigcirc A$ is modal: By the universal condition on η we have $f : \bigcirc\bigcirc A \to \bigcirc A$ such that $f \circ \eta_{\bigcirc A} = id_{\bigcirc A}$. But the same condition assures that the type

$$\sum_{g : \bigcirc\bigcirc A \to \bigcirc\bigcirc A} g \circ \eta_{\bigcirc A} = \eta_{\bigcirc A}$$

is contractible. Then $id_{\bigcirc\bigcirc A} = \eta_{\bigcirc A} \circ f$, so $\eta_{\bigcirc A} : \bigcirc A \simeq \bigcirc\bigcirc A$ as desired.

$\square$

$\sum$-closure

Remark

Every reflective subuniverse derived from a uniquely eliminating operator is $\sum$-closed.

Proof.

Let A be modal and $P : A \to \mathscr{U}$ be such that each $P(x)$ is modal. Define h by composing

$$\bigcirc\Big(\sum_{x:A} P(x)\Big) \xrightarrow[\bigcirc \mathsf{pr1}]{} \bigcirc A \xrightarrow[(\eta_A)^{-1}]{} A$$

Then, for $z : \sum_{x:A} P(x)$ we have

$$\begin{aligned} h(\eta(z)) &= \eta^{-1}(\bigcirc \mathsf{pr1}(\eta z)) && \text{def. of } h \\ &= \eta^{-1}(\eta(\mathsf{pr1} z)) && \text{nat. of } \eta \\ &= \mathsf{pr1} z \end{aligned}$$

Let p_z be this path. Let $C : \bigcirc \sum_{x:A} P(x) \to \mathscr{U}$ such that $C(w) :\equiv P(h(w))$.

Then we have

$$g :\equiv \lambda z.\mathsf{transport}(p_z, \mathsf{pr2}(z)) : \prod_{z:\sum_{x:A} P(x)} C(\eta(z)).$$

Σ-closure

Proof (continued).

By unique elimination this yields

$$f : \prod_{w : \bigcirc \sum_{x:A} P(x)} C(w) \quad \text{such that} \quad f(\eta z) = g(z).$$

Define then $k : \bigcirc \sum_{x:A} P(x) \to \sum_{x:A} P(x)$ by

$$k(w) :\equiv (h(w), f(w)).$$

Finally, the paths $p_z : h(\eta(z)) = \mathsf{pr1}\, z$ and $f(\eta z) = g(z) = \mathsf{transport}(p_z, \mathsf{pr2}(z))$ show that $k \circ \eta = id_{\sum_{x:A} P(x)}$. Lemma 2 gives the result.

$\square$

U.E.Operator from $\sum$-closed reflective subuniverse

Lemma (6)

Every $\sum$-closed reflective subuniverse defines a uniquely eliminating operator.

Proof.

Let $\bigcirc$ be the reflector of a $\sum$-closed reflective subuniverse, A be a type and $P : \bigcirc A \to \mathcal{U}$ be such that each $P(a)$ is modal. I need to show that for any $g : \prod_{x:A} P(\eta x)$ there exists an $f : \prod_{z:\bigcirc A} P(z)$ such that $f(\eta a) = ga$ for any $a : A$.

Note that $\sum_{z:\bigcirc A} P(z)$ is modal, therefore, by defining $g' : A \to \sum_{z:\bigcirc A} P(z)$ such that $g'(a) :\equiv (\eta a, ga)$ we have $\bigcirc g' : \bigcirc A \to \sum_{z:\bigcirc A} P(z)$ such that $\bigcirc g'(\eta a) = (\eta a, ga)$.

Therefore $\mathrm{pr1} \circ \bigcirc g' \circ \eta_A = id_{\bigcirc A} \circ \eta_A = \eta_A$ which yields $\mathrm{pr1} \circ \bigcirc g' = id_{\bigcirc A}$.

Then we have $p_z : \mathrm{pr1}(\bigcirc g'(z)) = z$ for any $z : \bigcirc A$, so we can define $f(z) :\equiv \mathrm{transport}(p_z, \mathrm{pr2}(\bigcirc g'(z)))$. Since $p_{\eta a} : \mathrm{pr1}(\bigcirc g'(\eta a)) = \eta a$, we have $f(\eta a) = ga$, as desired. $\square$

Identity Types

Lemma (7)

If A is modal, so is $x \underset{A}{=} y$ for any $x, y : A$.

Proof.

For $f : A \to C$ and $g : B \to C$ let $A \underset{C}{\times} B :\equiv \sum_{a:A} \sum_{b:B} (f(a) = g(b))$ be the type-theoretic pullback of f and g. It is easy to see that $x \underset{A}{=} y \simeq 1 \underset{A}{\times} 1$. Then we have

$$
\begin{tikzcd}
\bigcirc(x=y) \arrow[dr, dashed, "f"] \arrow[drr, bend left] \arrow[ddr, bend right] & & \\
& x=y \arrow[ul, bend left, "\eta"] \arrow[r, "!"'] \arrow[d, "!"'] & 1 \arrow[d, "x"] \\
& 1 \arrow[r, "y"'] & A
\end{tikzcd}
$$

Since A is modal by assumption, the outer rectangle commutes, for the extensions of $x = y \to A$ are unique. Then we have $x \circ ! \circ f = y \circ ! \circ f$ by which $x \circ ! \circ f \circ \eta = y \circ ! \circ f \circ \eta$, so that $f \circ \eta = id_{x=y}$, and we are done. □

Language for modal types

Remark

The basic properties of types and terms in HoTT are built from identity types, $\sum$*- and* $\prod$*-types. This means that a* $\sum$*-closed reflective subuniverse is closed under them as well.*

Higher Modalities

Lemma (8)

Every $\sum$-closed reflective subuniverse defines a higher modality given by

(i) $\eta : \prod_{A:\mathcal{U}} A \to \bigcirc A$;

(ii) *for any $A : \mathcal{U}$ and any $P : \bigcirc A \to \mathcal{U}$ a map*

$$\mathrm{ind}_A : \prod_{a:A} \bigcirc P(\eta a) \to \prod_{z:\bigcirc A} \bigcirc P(z)$$

such that $\mathrm{ind}_A(f)(\eta(a)) = f\,a$ *for any* $f : \prod_{a:A} \bigcirc P(\eta a)$;

(iii) *for any $x, y : \bigcirc A$, $\eta_{x=y}$ is an equivalence.*

Higher Inductive Types

Any HIT has:

- term formation and path formation rules
- elimination rules which consider path formation rules
- computation rules given by a path (not $\equiv$)

HITs extends the system considered so far, but here higher modalities are defined within UniTT.

Lemma (8)

Every $\sum$-closed reflective subuniverse defines a higher modality given by

(i) $\eta : \prod_{A:\mathcal{U}} A \to \bigcirc A$;

(ii) for any $A : \mathcal{U}$ and any $P : \bigcirc A \to \mathcal{U}$ a map

$$\mathrm{ind}_A : \prod_{a:A} \bigcirc P(\eta a) \to \prod_{z:\bigcirc A} \bigcirc P(z)$$

such that $\mathrm{ind}_A(f)(\eta(a)) = f\,a$ for any $f : \prod_{a:A} \bigcirc P(\eta a)$;

(iii) for any $x, y : \bigcirc A$, $\eta_{x=y}$ is an equivalence.

Proof.

(iii) is given by Lemma 7.

(ii) follows by Lemma 6 and remark 2.

(i) follows by definition of subuniverse.

$\square$

$\sum$-closed reflective subuniverse from H.Modality

Lemma (9)

Every higher modality defines a $\sum$-closed reflective subuniverse.

Proof.

Assume a higher modality is given. It suffices to construct a uniquely eliminating operator, i.e. to prove that

$$\lambda g.g \circ \eta_A : \prod_{x:\bigcirc A} \bigcirc P(x) \to \prod_{a:A} \bigcirc P(\eta a)$$

is an equivalence. Lemma 8 gives ind_A as right inverse of $\lambda g.g \circ \eta_A$.
Given $s : \prod_{x:\bigcirc A} \bigcirc P(x)$, we need a homotopy

$$\prod_{x:\bigcirc A} (sx = \mathrm{ind}_A(s \circ \eta_A)x).$$

Since $\bigcirc P(x)$ is modal for any $x : \bigcirc A$, so is $sx = \mathrm{ind}_A(s \circ \eta_A)x$, therefore it suffices to find, by Lemma 8(ii), a homotopy $\prod_{a:A} \bigcirc (s(\eta a) = \mathrm{ind}_A(s \circ \eta_A)(\eta a))$ i.e. a homotopy $\prod_{a:A} (s(\eta a) = \mathrm{ind}_A(s \circ \eta_A)(\eta a))$, which is given by Lemma 8(ii). □

A unique family of types

Lemma (10)

The types of $\sum$-closed reflective subuniverses, uniquely eliminating operators, and higher modalities are all equivalent.

Proof.

Note first that:

- The defining conditions for a reflective subuniverse define a type which is a proposition;
- The defining conditions for a uniquely eliminating operator define a type which is a proposition;
- The defining conditions for a higher modality define a type which is a proposition.

Therefore, Lemmas 6, 8, 9, and remark 2 give the result. $\square$

Thank you for listening!

References I

U. Buchholz.
Proof Theory of Homotopy Type Theory. FOMUS 2016.
https://youtu.be/ZQyBLXJQs2w.

P. Dybjer, E. Palmgren.
Intuitionistic Type Theory. SEP (Winter 2016 Edition)
https://plato.stanford.edu/archives/win2016/entries/type-theory-intuitionistic/.

E. Rijke.
Introduction to Homotopy Type Theory.
https://hott.github.io/HoTT-2019/images/hott-intro-rijke.pdf.

E. Rijke, M. Shulman, B. Spitters.
Modalities in Homotopy Type Theory.
https://arxiv.org/pdf/1706.07526v4.pdf.

The HoTT Library

https://github.com/HoTT/HoTT.