Homotopy Type Theory

A Gentle Introduction

Dr Marco Benini

marco.benini@uninsubria.it

Dipartimento di Scienza e Alta Tecnologia
Università degli Studi dell'Insubria

25th September 2023

Introduction

Homotopy Type Theory, **HoTT**, is

- a type system
- a functional programming language
- a way to describe higher order logic
- a way to think ∞-groupoids
- a way to describe homotopical spaces
- it claims to be a foundational system
- a fashion in current Mathematics

Introduction

$$\textbf{HoTT} = \textbf{MLTT} + \textbf{univalence}$$

Syntactically, HoTT is a variant of Martin-Löf type theory enriched with a complex axiom, *univalence*.

The syntax allows an interpretation, the *Curry-Howard isomorphism*, in which types are read as propositions and terms as their derivations.

The syntax allows an interpretation, which is the focus of this talk, in which types are homotopy spaces.

Despite the intriguing topological interpretation, which justifies its study, there is still a lot of research work to pursue, and the foundations of HoTT are unexplored (and misunderstood) in many essential aspects.

Martin-Löf type theory

MLTT is based on the notion of *judgement*

$$\Gamma\ \mathsf{ctx} \qquad \Gamma \vdash a : T \qquad \Gamma \vdash a \equiv b : T$$

We define judgements by induction: judgements are generated by a (large) collection of *inference rules*.

It is important to remark that the set of inference rules is *open*: we may add new types as far as their associated rules have a quite rigid *inductive* structure.

Contexts

$$\frac{}{\bullet\ \mathsf{ctx}}\ \mathsf{ctx\text{-}EMP} \qquad \frac{\Gamma \vdash A : \mathscr{U}_i}{\Gamma, x : A\ \mathsf{ctx}}\ \mathsf{ctx\text{-}EXT}$$

$$\frac{x_1 : A_1, \ldots, x_n : A_n\ \mathsf{ctx}}{x_1 : A_1, \ldots, x_n : A_n \vdash x_i : A_i}\ \mathsf{Vble}$$

Variable declaration and use
Hypotheses

Judgemental equality

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash a \equiv a : A}\ \equiv\text{–refl} \qquad \frac{\Gamma \vdash a \equiv b : A}{\Gamma \vdash b \equiv a : A}\ \equiv\text{–sym} \qquad \frac{\Gamma \vdash a \equiv b : A \quad \Gamma \vdash b \equiv c : A}{\Gamma \vdash a \equiv c : A}\ \equiv\text{–trans}$$

$$\frac{\Gamma \vdash a : A \quad \Gamma \vdash A \equiv B : \mathcal{U}_i}{\Gamma \vdash a : B}\ \equiv\text{–subst} \qquad \frac{\Gamma \vdash a \equiv b : A \quad \Gamma \vdash A \equiv B : \mathcal{U}_i}{\Gamma \vdash a \equiv b : B}\ \equiv\text{–subst–eq}$$

Conversion, reduction
Definitional equivalence
Computation

Universes

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathscr{U}_i : \mathscr{U}_{i+1}}\ \mathscr{U}\text{–intro}$$

$$\frac{\Gamma \vdash A : \mathscr{U}_i}{\Gamma \vdash A : \mathscr{U}_{i+1}}\ \mathscr{U}\text{–cumul} \qquad \frac{\Gamma \vdash A \equiv B : \mathscr{U}_i}{\Gamma \vdash A \equiv B : \mathscr{U}_{i+1}}\ \mathscr{U}\text{–cumul–eq}$$

Type of (small) types
Huge source of problems
Unavoidable in HoTT

Function spaces

$$\frac{\Gamma \vdash A : \mathscr{U}_i \quad \Gamma, x : A \vdash B : \mathscr{U}_i}{\Gamma \vdash \Pi x : A.\, B : \mathscr{U}_i}\ \Pi\text{−form}$$

$$\frac{\Gamma \vdash A \equiv A' : \mathscr{U}_i \quad \Gamma, x : A \vdash B \equiv B' : \mathscr{U}_i}{\Gamma \vdash \Pi x : A.\, B \equiv \Pi x : A'.\, B' : \mathscr{U}_i}\ \Pi\text{−form−eq}$$

When x is not free in B,

$$A \to B :\equiv \Pi x : A.\, B$$

Dependent functions
Fibrations
Universal quantifier, implication

Function spaces

$$\frac{\Gamma, x : A \vdash b : B}{\Gamma \vdash \lambda x : A.\, b : \Pi x : A.\, B}\ \Pi\text{−intro}$$

$$\frac{\Gamma, x : A \vdash b \equiv b' : B \quad \Gamma \vdash A \equiv A' : \mathscr{U}_i}{\Gamma \vdash \lambda x : A.\, b \equiv \lambda x : A'.\, b' : \Pi x : A.\, B}\ \Pi\text{−intro−eq}$$

λ-abstraction, intensional functions
Implication introduction, forall introduction

Function spaces

$$\frac{\Gamma \vdash f : \Pi x : A. B \quad \Gamma \vdash a : A}{\Gamma \vdash f\, a : B[a/x]}\ \Pi\text{−elim}$$

$$\frac{\Gamma \vdash f \equiv g : \Pi x : A. B \quad \Gamma \vdash a \equiv a' : A}{\Gamma \vdash f\, a \equiv g\, a' : B[a/x]}\ \Pi\text{−elim−eq}$$

Functional application
Implication elimination, specialisation

Function spaces

$$\frac{\Gamma, x : A \vdash b : B \quad \Gamma \vdash a : A}{\Gamma \vdash (\lambda x : A.\, b)\, a \equiv b[a/x] : B[a/x]}\ \Pi\text{−comp}$$

$$\frac{\Gamma \vdash f : \Pi x : A.\, B}{\Gamma \vdash \lambda x : A.\, f\, x \equiv f : \Pi x : A.\, B}\ \Pi\text{−uniq}$$

β-reduction
η-reduction
Substitution

Structural rules

The rules so far form the structural part of Martin-Löf type theory.

Open problems:

- (strong) normalisation
- structural proof theory
- computational properties
- ...

Everything is almost perfect **without** universes
Subject reduction fails
Lot of folklore, sometimes unjustified

Inductive types

- one *formation rule*
- one *introduction rule* for each *constructor*

$$+$$

- one *elimination rule*, coding induction (and recursion)

$$+$$

- one *computation rule* for each constructor

Formation, introduction, and elimination are *constant introductions*
Elimination and computation are automatically synthesised
Types and type families

Dependent pairs

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \Sigma : \Pi A : \mathscr{U}_i, B : A \to \mathscr{U}_i . \mathscr{U}_i}\ \Sigma\text{–form}$$

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathsf{pair} : \Pi A : \mathscr{U}_i, B : A \to \mathscr{U}_i, a : A, b : B\, a . \Sigma\, A\, B}\ \Sigma\text{–intro}$$

When x is not free in B,

$$A \times B :\equiv \Sigma x : A . B$$

Dependent Cartesian product
Existential quantification and conjunction

Abbreviated writing:

- $\Sigma x : A . B$ is $\Sigma A (\lambda x : A . B)$, or $\Sigma A B$ is $\Sigma x : A . B x$
- $\mathsf{pair}\, A\, B\, a\, b$ is abbreviated in (a, b) when A and B are known

Zero type

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathbf{0} : \mathscr{U}_i}\ \mathbf{0}\text{–form}$$

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathsf{ind}_{\mathbf{0}} : \Pi P : \mathbf{0} \to \mathscr{U}_i . \Pi e : \mathbf{0} . P\, e}\ \mathbf{0}\text{–elim}$$

Empty type
Falsity
Induction is $\bot$-elimination

Unit type

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathbf{1} : \mathscr{U}_i}\ \mathbf{1}\text{–form} \qquad \frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash * : \mathbf{1}}\ \mathbf{1}\text{–intro}$$

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathsf{ind}_{\mathbf{1}} : \Pi P : \mathbf{1} \to \mathscr{U}_i . \Pi c_1 : P * . \Pi e : \mathbf{1} . P\, e}\ \mathbf{1}\text{–elim}$$

$$\frac{\Gamma \vdash P : \mathbf{1} \to \mathscr{U}_i \quad \Gamma \vdash c_1 : P *}{\Gamma \vdash \mathsf{ind}_{\mathbf{1}}\, P\, c_1 * \equiv c_1 : P *}\ \mathbf{1}\text{–comp}$$

Unit type
Distinguished singleton (!)
Truth

Booleans, **2**, are defined similarly

Natural numbers

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathbb{N} : \mathscr{U}_i}\ \mathbb{N}\text{–form}$$

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash 0 : \mathbb{N}}\ \mathbb{N}\text{–intro}_1 \qquad \frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathsf{succ} : \mathbb{N} \to \mathbb{N}}\ \mathbb{N}\text{–intro}_2$$

$$\frac{\Gamma\ \mathsf{ctx}}{\begin{array}{l}\Gamma \vdash \mathsf{ind}_{\mathbb{N}} : \Pi P : \mathbb{N} \to \mathscr{U}_i. \\ \qquad\qquad \Pi c_1 : P\,0. \\ \qquad\qquad \Pi c_2 : (\Pi x : \mathbb{N}. \Pi r : P\,x. P\,(\mathsf{succ}\,x)). \\ \qquad\qquad \Pi e : \mathbb{N}. P\,e\end{array}}\ \mathbb{N}\text{–elim}$$

Peano's definition
Induction is "proof aware"

Path spaces

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash = : \Pi A : \mathcal{U}_i, a : A, b : A. \mathcal{U}_i}\ {=}\text{–form}$$

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathsf{refl} : \Pi A : \mathcal{U}_i, a : A. a =_A a}\ {=}\text{–intro}$$

Identity type family
Equality

$= A\,x\,y$ is written $x =_A y$
We write $x = y$ in place of $x =_A y$ when A is understood
refl_x abbreviates $\mathsf{refl}\,A\,x$ when it is known $x : A$

Path spaces

$$\dfrac{\Gamma\ \mathsf{ctx}}{\begin{aligned}\Gamma \vdash \mathsf{ind}_= : \Pi A : \mathscr{U}_i.&\\ &\Pi P : (\Pi x : A, y : A.\, x =_A y \to \mathscr{U}_i).\\ &\Pi c_1 : (\Pi x : A.\, P\, x\, x\, (\mathsf{refl}\, A\, x)).\\ &\Pi a : A, b : A.\, \Pi e : a =_A b.\, P\, a\, b\, e\end{aligned}}\ {=}{-}\mathsf{elim}$$

"strict", syntactical generation is coded by Streicher's **K**-axiom
we want "path-aware" generation

Homotopy interpretation

Synthetic approach, like Euclid's geometry vs Descartes'

A type is a space
$a : A$ is a point a in the space A
$a =_A b$ is the space of *paths* in A from a to b

Spaces are formed by points, paths, homotopies, k-paths in general

This interpretation works under **one additional principle**:

two homotopically equivalent spaces are equal

This is the essence of *univalence*

Homotopy interpretation

The homotopy interpretation is *synthetic* because

k-paths are primitive entities

A point $a : A$ is a 0-path in the space A
$p : a =_A b$ is a 1-path in the space A

Observe how $p : a =_A b$ is a 0-path in the space $a =_A b$

Homotopy interpretation

Taken seriously, the homotopy interpretation differs from the standard interpretation of MLTT.

For example, consider the unit type $\mathbf{1}$:

- in the standard interpretation, $* : \mathbf{1}$ is the **unique** element in $\mathbf{1}$.
- in the homotopy interpretation, $* : \mathbf{1}$ is an element of $\mathbf{1}$ and any other element $x : \mathbf{1}$ is equal to it, that is, there is a path from x to $*$.
  Hence, the ball $\{x : |x| < r\}$ of radius r in $\mathbb{R}^3$ **is** the unit type.
- however, the sphere $\{x : |x| = r\}$ of radius r in $\mathbb{R}^3$ **is not** $\mathbf{1}$ since there is a 2-path which is not refl.

Homotopy interpretation

The slogan of the standard interpretation is

equality is identity

while the slogan of the homotopy interpretation is

equality is homotopy equivalence

Paths

Lemma 5.1 (Path inversion)

For every type A and every $x, y : A$ *there is a function* $(\_)^{-1} : x =_A y \to y =_A x$ *such that* $\mathsf{refl}_x^{-1} \equiv \mathsf{refl}_x$.

Lemma 5.2 (Path composition)

For every type A and every $x, y, z : A$ *there is a function*

$$\_ \cdot \_ : x = y \to y = z \to x = z$$

such that $\mathsf{refl}_x \cdot \mathsf{refl}_x \equiv \mathsf{refl}_x$ *for any* $x : A$.

Observe how Lemma 5.1 tells that equality is symmetric and Lemma 5.2 tells that equality is transitive, Equality is reflexive thanks to =−intro.

Paths

The interpretation of (propositional) equality $x =_A y$ is satisfactory because

- equality is an equivalence relation
- equality forms a groupoid w.r.t. path composition

However

- equality should be a congruence with respect to application, abstraction, and function spaces formation
- equality should be preserved by functions

The second fact is true, and it can be proved.

However, equality **is not** a congruence in the usual sense. It is almost a congruence, making everything much more complex.

Functions and functors

Lemma 5.3 (Application; Action on Paths)

Let $f : A \to B$. Then, for every $x, y : A$ there is

$$\mathrm{ap}_f : x =_A y \to f\,x =_B f\,y$$

such that $\mathrm{ap}_f(\mathrm{refl}_x) = \mathrm{refl}_{(f\,x)}$*. Usually* $\mathrm{ap}_f\,p$ *is written as* $f(p)$*.*

Lemma 5.4 (Functoriality of ap)

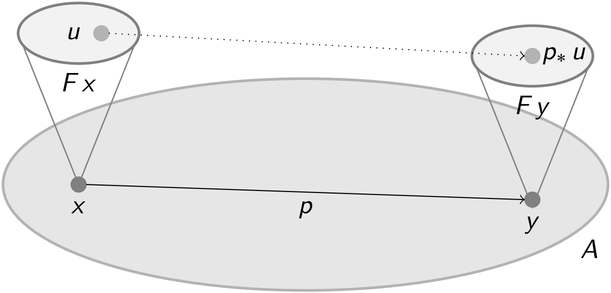
Let $f : A \to B$, $g : B \to C$, $p : x =_A y$, and $q : y =_A z$. Then

1. $f\,(p \cdot q) = (f\,p) \cdot (f\,q)$
2. $f\,(p^{-1}) = (f\,p)^{-1}$
3. $f\,(g\,p) = (g \circ f)\,p$
4. $\mathrm{id}_A\,p = p$

So functions are functors in the ∞-groupoid of paths.
Type-theoretically and logically, functions respect equality.
Homotopically, functions are *continuous*, i.e., they preserve paths.

Fibrations

Let $F : A \to \mathscr{U}_i$, $p : x =_A y$. Then we may think to F as a *fibration*:



Lemma 5.5 (Transport)

Let $P : A \to \mathscr{U}_i$ *and* $p : x =_A y$. *Then there is* $p_* : P\,x \to P\,y$.

We also write $\mathsf{transport}^P(p, \_) :\equiv p_* : P\,x \to P\,y$.

Homotopies

Definition 6.1 (Homotopy)

Let $f, g : \Pi x : A. B$. An *homotopy* from f to g is a point of

$$f \sim g :\equiv \Pi x : A. f\, x =_B g\, x$$

Lemma 6.2

Homotopy is an equivalence relation: the following types are inhabited

$$\Pi f : A \to B. f \sim f$$
$$\Pi f, g : A \to B. f \sim g \to g \sim f$$
$$\Pi f, g, h : A \to B. f \sim g \to (g \sim h \to f \sim h)$$

From now on, we say that $P : \mathscr{U}_i$ *holds* to mean that P is inhabited

Equivalence

Definition 6.3 (Equivalence)

Given $f : A \to B$, it is an *equivalence* if the following holds

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

Definition 6.4 (Equivalence type)

Given $A, B : \mathcal{U}_i$, $A \simeq B :\equiv \Sigma f : A \to B.\, \mathsf{isequiv}(f)$.
We say that A and B are *equivalent* when $A \simeq B$ holds

Lemma 6.5

Type equivalence is an equivalence relation.

Univalence

Lemma 7.1

Given $A, B : \mathscr{U}_i$*, it holds*

$$\mathsf{idtoeqv} : A =_{\mathscr{U}_i} B \to A \simeq B$$

Axiom (Univalence)

For any $A, B : \mathscr{U}_i$*,*

$$(A =_{\mathscr{U}_i} B) \simeq (A \simeq B)$$

In particular, the axiom states that there is a distinct element

$$\mathsf{ua} : (A \simeq B) \to (A =_{\mathscr{U}_i} B)$$

which inverts idtoeqv*.*

Function extensionality

Lemma 7.2
If $f, g : \Pi x : A.B$ *then*

$$(f = g) \to (f \sim g)$$

However, the converse requires univalence (and it is a complex proof)

Theorem 7.3
If $f, g : \Pi x : A.B$ *then*

$$(f \sim g) \to (f = g)$$

Together, $(f = g) = (f \sim g)$, which is called *function extensionality*

Homotopical interpretation

In the overall, the homotopical interpretation

- provides HoTT with a strong and powerful guideline
- allows to derive "natural" results in topological terms
- is solid and well coordinated with the Curry-Howard isomorphism

However

- it is "complex" to deal with
- results are hard to check by hand
- in the current stage, the link between the logical and homotopy interpretations has not yet been fully exploited

As a foundational theory, HoTT lacks a systematic development

Higher Inductive Types

An inductive type generates its elements through induction

Since we are in a world in which paths are the basic elements, why to limit ourselves to define the generation of **points** and not consider to control the generation of **paths**, too?

This is the idea behind *Higher Inductive Types*

Higher Inductive Types

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathbb{S}^1 : \mathscr{U}_i}\ \mathbb{S}^1\text{–form} \qquad \frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathsf{base} : \mathbb{S}^1}\ \mathbb{S}^1\text{–intro}$$

$$\frac{\Gamma\ \mathsf{ctx}}{\Gamma \vdash \mathsf{loop} : \mathsf{base} =_{\mathbb{S}^1} \mathsf{base}}\ \mathbb{S}^1\text{–intro}$$

The space $\mathbb{S}^1$ contains a distinguished point, base, and a distinguished path, loop, from base to itself. Induction on $\mathbb{S}^1$ says that, given

- a property $P : \mathbb{S}^1 \to \mathscr{U}_i$
- a point $b : P\,\mathsf{base}$
- a path $\ell : \mathsf{base} =^{P}_{\mathsf{loop}} \mathsf{base}$, from the transport of base along loop in the fibration P to base

the type $P\,b$ is inhabited by a canonical term $\mathsf{ind}_{\mathbb{S}^1}\, b\, \ell$.

Hence $\mathbb{S}^1$ **is** the 1-sphere, or the perimeter of a circle, if you prefer

Higher Inductive Types

Similarly, one can define k-spheres, for any $k > 0$, the torus, suspensions, cell complexes, and other topological objects

Also non-topological higher inductive types can be constructed.
Truncations have a special place
Given $A : \mathscr{U}_i$, the *truncation* $||A||$ of A is the type inductively generated by

- a function $|\_| : A \to ||A||$
- for each $x, y : ||A||$, a path $x = y$

One may think to $||A||$ as A deprived from its homotopy structure

But, logically, $||A||$ is A in classical logic...

...and this is just the beginning of another story...

References

The main reference is *The Univalent Foundation Program*, Homotopy Type Theory: Univalent Foundations of Mathematics, Institute for Advanced Study (2013), `https://homotopytypetheory.org/book`.

Martin-Löf type theory is described in *Per Martin-Löf*, An intuitionistic theory of types: Predicative part, *H.E. Rose* and *J.C. Shepherdson* eds., Logic Colloquium '73, Studies in Logic and the Foundations of Mathematics 80, Elsevier (1975), pp. 73—118.
We suggest also *Per Martin-Löf*, Intuitionistic Type Theory: Notes by Giovanni Sambin of a series of lectures given in Padua, June 1980, Studies in Proof Theory 1, Bibliopolis, Naples, Italy (1984).

The end

©Marco Benini, Patio in the forest, Seoul