Introduction

Samuel Mimram

2025

École polytechnique

What is this course about?

In a nutshell,

$$\textbf{type} = \textbf{space}$$

and thus constructions on types correspond to geometric ones!

The dictionary extends quite far

$$\begin{aligned}
\text{term of type } A &\quad = \quad \text{point in } A \\
\text{proof of } x = y &\quad = \quad \text{path from } x \text{ to } y \\
\text{proof of } p = q \text{ with } p, q : x = y &\quad = \quad \text{homotopy between } p \text{ and } q \\
\text{type family } B : A \to \mathcal{U} &\quad = \quad \text{fibration with base } B
\end{aligned}$$

and so on...

The space semantics of logic

The **Curry-Howard correspondence** is the discovery that a proof of

$$A \Rightarrow B$$

is the same as a program of type

$$A \rightarrow B$$

We will see that it more generally makes sense to interpret a type as a **space**.

Moreover, $A \rightarrow B$ will denote the **continuous** functions from A to B.

Semantics of logic: new types

The boolean semantics of logic suggest introducing two basic types: 0 and 1.

The set semantics of logic suggest introducing many new basic types:

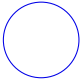
$$\mathbb{N} \qquad \mathbb{Z} \qquad \mathsf{Bool} \qquad \mathsf{Fin}_n \triangleq \{0, \ldots, n-1\} \qquad \ldots$$

The space semantics of logic: new types

In homotopy type theory, we still get types for usual sets:

$$\mathbb{N} \quad \hat{=} \quad \cdots \quad \overset{-2}{\bullet} \quad \overset{-1}{\bullet} \quad \overset{0}{\bullet} \quad \overset{1}{\bullet} \quad \overset{2}{\bullet} \quad \cdots$$

but we also have the n-spheres:

$\mathsf{S}^0 = \bullet \quad \bullet$ $\qquad$ $\mathsf{S}^1 =$  $\qquad$ $\mathsf{S}^2 =$  $\qquad \ldots$

as well as weird spaces

$$\mathbb{R}\mathrm{P}^n \qquad\qquad \mathrm{B}\,G \qquad\qquad \ldots$$

The space semantics of logic: new constructions

The set-theoretic interpretation suggests introducing new constructions such as coproducts:

$$\mathsf{Fin}\, n \quad = \quad 1 \sqcup 1 \sqcup \ldots \sqcup 1$$

Similarly, the geometric interpretation suggests new operations such as the join:

$$\mathsf{S}^n \quad = \quad \mathsf{S}^0 * \mathsf{S}^0 * \ldots * \mathsf{S}^0$$

In many languages, the useful types can be defined as inductive types, the new constructions can similarly be defined as *higher inductive types*:

$$\begin{array}{l} \mathsf{type}\ \mathbb{N}\ \mathsf{where} \\ \quad \mathsf{zero} : \mathbb{N} \\ \quad \mathsf{suc} \ \ : \mathbb{N} \to \mathbb{N} \end{array} \qquad \begin{array}{l} \mathsf{type}\ \mathsf{S}^1\ \mathsf{where} \\ \quad \mathsf{pt} \quad : \mathsf{S}^1 \\ \quad \mathsf{loop} : \mathsf{pt} = \mathsf{pt} \end{array} \qquad \begin{array}{l} \mathsf{type}\ A * B\ \mathsf{where} \\ \quad \mathsf{inl} \quad\ : A \to A * B \\ \quad \mathsf{inr} \quad\ : B \to A * B \\ \quad \mathsf{push} : (a : A)\,(b : B) \to \mathsf{inl}\, a = \mathsf{inr}\, b \end{array}$$

The space semantics of logic: synthetic geometry

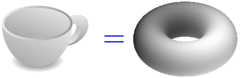
This framework allows for doing **synthetic** geometry:
all the types can be interpreted as spaces or constructions on spaces.

In particular, we never need to resort to "low-level stuff" such as topology.

Moreover, all constructions are homotopy invariant.

The space semantics of logic: homotopy invariance

By **space**, we really mean (nice) space up to **homotopy equivalence**:



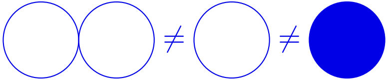
$$\mathbb{R} = 1$$

In topology, such an equivalence class was called a **homotopy type**, and thus:

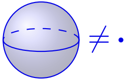
$$\textbf{homotopy (type theory)} = \textbf{(homotopy type) theory}$$

The space semantics of logic: homotopy invariance

A fundamental property of homotopy is that it preserves the number of holes (up to deformation) in every dimension. Thus,



or



This suggests many concepts:

- a space A is n-*truncated* when it has no holes in dimension $k > n$,
- a space A is an n-approximation of B when they have the same holes up to dim n
- etc.

Homotopy type theory for set theorists

It is not only for homotopy theorists!

We gain useful distinctions such as between **propositions** and **sets**.

All constructions are invariant under isomorphism for structures
(and equivalences for categories if we define them in the right way!).

Conversely, we can transfer properties if A and B are isomorphic and we know something on A then we can transfer it to B (more on this later).

Homotopy type theory for type theorists

The rules for equality in type theory have never been entirely clear:

- what should the uniqueness rule for identity types be?
  (the naive answer makes equality undecidable)
- in practice we need quotient types: how can this be achieved in a decent way?
  (strict quotient make equality undecidable, setoids are a hell)
- should we accept principles such as function extensionality?
  (more on this later)

Homotopy type theory offers a satisfactory answer to this with a single axiom:

univalence

Homotopy type theory for constructivists

Some people like to be **constructive**:

- we focus on proofs rather than provability
- we want to be able to actually compute things:
  when proving $\exists(n : \mathbb{N}).P(n)$ we should be able to exhibit an actual number

In homotopy type theory,

- we have witnesses for proofs of equality $p : x = y$,
  but also equalities between equalities $\alpha : p = q$, etc.
- we can show useful equalities such as $\mathbb{N}_{\text{binary}} = \mathbb{N}_{\text{unary}}$
  (note: we expect that there should be multiple such equalities!)
- we can transfer constructions and this computes!
  (e.g. operations on binary numbers can be transported to unary numbers)

Homotopy type theory for constructivists

In homotopy type theory, we can deduce **function extensionality**:

$$\forall x. f(x) = g(x) \quad \Rightarrow \quad f = g$$

In particular, this means that all sorting algorithms $f : \mathsf{List} \to \mathsf{List}$ are equal.

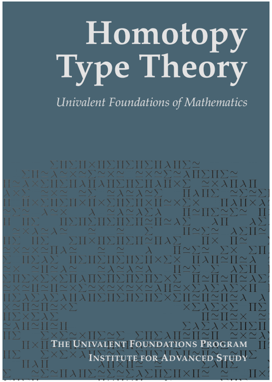
There is no contradiction with the fact that such a function corresponds to a program: equality has a computational content and means more than just plain identification.

Homotopy type theory for historians

- 1994: the groupoid model of identity types (Hoffman, Streicher) [HS98]
- 2006: models of MLTT+Id in model categories (Awodey, Warren) [AW09]
- 2006: conjectural model of MLTT+Id in Kan complexes (Voevodsky) [Voe06]
- 2008: types are weak ω-groupoids (vdBerg, Garner, Lumsdaine) [VDBG11, Lum10]
- 2009: the univalence axiom (Voevodsky) [Voe10]
- 2012-13: special year at IAS, the HoTT book

Course notes

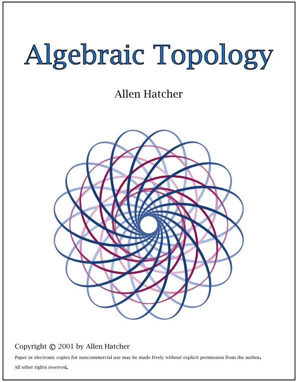
We will be mostly following the **HoTT book**:



which can be obtained for free at `https://homotopytypetheory.org/book/`

Advertisement

This is not at all required, but if you want to know more about algebraic topology, I would suggest:



which can be obtained for free at

`https://pi.math.cornell.edu/~hatcher/AT/ATpage.html`

Labs

All the labs will consist in formalizing stuff in the Agda proof assistant

We chose it because

- it is "pure" (no tactics): we control what we do, faster learning curve
- it is the only proof assistant with support for higher inductive types

Most of the labs depend on each other, work regularly!

Grading

The grading will consist in

- 50%: labs
- 50%: final exam on paper

For the labs:

- create a private github repository, add me (`smimram`), and send me a mail (you can also send me your files by mail if you are reluctant to technology)
- again, work regularly

Feedback

This is the first year I am giving this course:

- I might have to adapt the timing, grading, etc.
- any feedback is welcome at any time

[AW09] Steve Awodey and Michael A Warren.
Homotopy theoretic models of identity types.
In *Mathematical proceedings of the cambridge philosophical society*, volume 146, pages 45–55. Cambridge University Press, 2009.
`arXiv:0709.0248`, `doi:10.1017/S0305004108001783`.

[HS98] Martin Hofmann and Thomas Streicher.
The groupoid interpretation of type theory.
Twenty-five years of constructive type theory (Venice, 1995), 36:83–111, 1998.
`https://ncatlab.org/nlab/files/HofmannStreicherGroupoidInterpretation.pdf`.

[Lum10] Peter LeFanu Lumsdaine.
Weak omega-categories from intensional type theory.
Logical Methods in Computer Science, 6, 2010.
`arXiv:0812.0409`, `doi:10.2168/LMCS-6(3:24)2010`.

[VDBG11] Benno Van Den Berg and Richard Garner.
Types are weak ω-groupoids.
Proceedings of the london mathematical society, 102(2):370–394, 2011.
`arXiv:0812.0298`, `doi:10.1112/plms/pdq026`.

[Voe06] Vladimir Voevodsky.
A very short note on the homotopy λ-calculus.
Mail on the TYPES mailing list, 2006.
https://www.math.ias.edu/vladimir/sites/math.ias.edu.vladimir/files/Hlambda_short_current.pdf.

[Voe10] Vladimir Voevodsky.
The equivalence axiom and univalent models of type theory.
Talk at CMU, February 2010.
arXiv:1402.5556.