Homotopy Type Theory in Lean

Floris van Doorn Jakob von Raumer Ulrik Buchholtz

github.com/leanprover/lean2

September 29, 2017

Homotopy Theory

In topology a topological property is invariant under homeomorphisms.

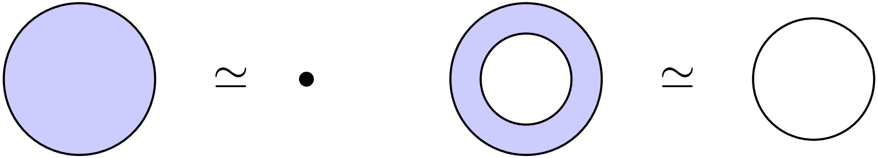
Examples. Separation axioms, first-countability, compactness, connectedness, . . .

Homotopy Theory

In topology a topological property is invariant under homeomorphisms.

Examples. Separation axioms, first-countability, compactness, connectedness, . . .

Some of these properties are invariant under homotopy equivalence.



Examples. Homotopy, homology and cohomology groups, path connectedness, . . .

Homotopy Equivalence

A homeomorphism $X \cong Y$ between two topological spaces consists of continuous functions $f : X \to Y$ and $g : Y \to X$ such that $g \circ f = \mathrm{id}_X$ and $f \circ g = \mathrm{id}_Y$.

Homotopy Equivalence

A homotopy equivalence $X \simeq Y$ between two topological spaces consists of continuous functions $f : X \to Y$ and $g : Y \to X$ such that $g \circ f \sim \mathrm{id}_X$ and $f \circ g \sim \mathrm{id}_Y$.

Homotopy Equivalence

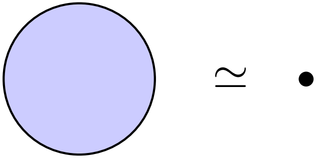
A homotopy equivalence $X \simeq Y$ between two topological spaces consists of continuous functions $f : X \to Y$ and $g : Y \to X$ such that $g \circ f \sim \mathrm{id}_X$ and $f \circ g \sim \mathrm{id}_Y$.

A homotopy $H : f \sim g$ is a continuous deformation of f into g, i.e. a map $H : [0,1] \times X \to Y$ such that $H(0, -) = f$ and $H(1, -) = g$.

Homotopy Equivalence

A homotopy equivalence $X \simeq Y$ between two topological spaces consists of continuous functions $f : X \to Y$ and $g : Y \to X$ such that $g \circ f \sim \mathrm{id}_X$ and $f \circ g \sim \mathrm{id}_Y$.

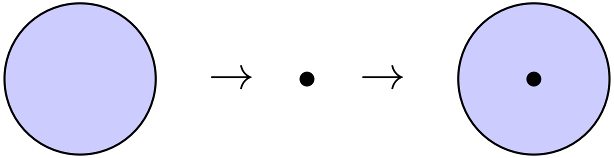
A homotopy $H : f \sim g$ is a continuous deformation of f into g, i.e. a map $H : [0,1] \times X \to Y$ such that $H(0, -) = f$ and $H(1, -) = g$.



Homotopy Equivalence

A homotopy equivalence $X \simeq Y$ between two topological spaces consists of continuous functions $f : X \to Y$ and $g : Y \to X$ such that $g \circ f \sim \mathrm{id}_X$ and $f \circ g \sim \mathrm{id}_Y$.

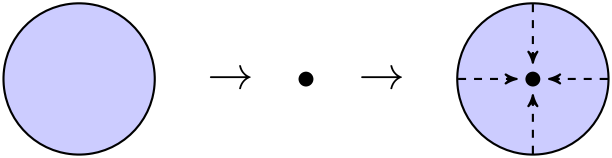
A homotopy $H : f \sim g$ is a continuous deformation of f into g, i.e. a map $H : [0,1] \times X \to Y$ such that $H(0,-) = f$ and $H(1,-) = g$.



Homotopy Equivalence

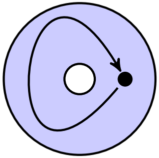
A homotopy equivalence $X \simeq Y$ between two topological spaces consists of continuous functions $f : X \to Y$ and $g : Y \to X$ such that $g \circ f \sim \mathrm{id}_X$ and $f \circ g \sim \mathrm{id}_Y$.

A homotopy $H : f \sim g$ is a continuous deformation of f into g, i.e. a map $H : [0, 1] \times X \to Y$ such that $H(0, -) = f$ and $H(1, -) = g$.

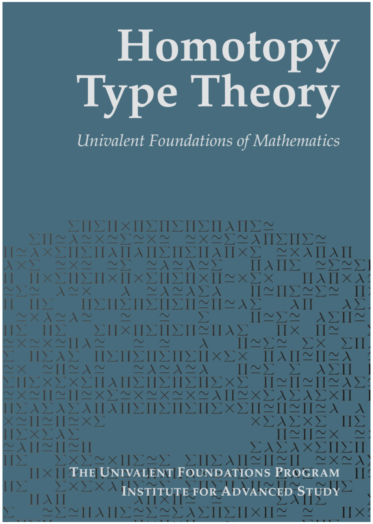


Fundamental Group

Given a space X with basepoint $x_0 \in X$ the fundamental group of $\pi_1(X, x_0)$ is the set of equivalence classes of continuous functions $f : [0, 1] \to X$ such that $f(0) = f(1) = x_0$ where f is related to g if there is a homotopy $h : f \sim g$ such that $h(x, 0) = h(x, 1) = a_0$ for all $x \in [0, 1]$. This forms a group.



Homotopy Type Theory


Homotopy Type Theory
Univalent Foundations of Mathematics
The Univalent Foundations Program
Institute for Advanced Study

homotopytypetheory.org/book/

Homotopy Type Theory

A discovery by Voevodsky [2006] and Awodey-Warren [2008] is that type theory has a natural interpretation in homotopy theory.

More specifically, in Martin-Löf dependent type theory there is an identity type $a =_A b$ for $a, b : A$.

We can form the type $p =_{a =_A b} q$ for $p, q : a =_A b$. One natural question is: what is this type?

Homotopy Type Theory

A discovery by Voevodsky [2006] and Awodey-Warren [2008] is that type theory has a natural interpretation in homotopy theory.

More specifically, in Martin-Löf dependent type theory there is an identity type $a =_A b$ for $a, b : A$.

We can form the type $p =_{a =_A b} q$ for $p, q : a =_A b$. One natural question is: what is this type?

Traditional answer: these types are trivial.
Axiom K: for all $p, q : a = b$ we have $p = q$.

Homotopy Type Theory

A discovery by Voevodsky [2006] and Awodey-Warren [2008] is that type theory has a natural interpretation in homotopy theory.

More specifically, in Martin-Löf dependent type theory there is an identity type $a =_A b$ for $a, b : A$.

We can form the type $p =_{a =_A b} q$ for $p, q : a =_A b$. One natural question is: what is this type?

The answer of homotopy type theory: these types have a meaning.

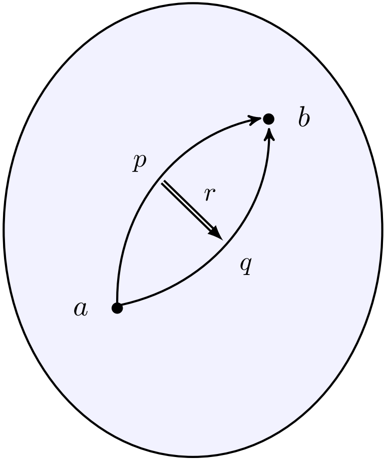
- A is a space;
- elements of $a = b$ are paths in A;
- elements of $p = q$ are homotopies.

Types as Spaces

A type A can have

- points $a, b : A$
- paths $p, q : a = b$
- paths between paths $r : p = q$

$$\vdots$$



Synthetic Homotopy Theory

In homotopy type theory, types are interpreted as spaces.

This leads to a new program, synthetic homotopy theory:

To study types in type theory as spaces in homotopy theory.

This method for homotopy theory is

- more general than ordinary homotopy theory;
- constructive;
- easier to formally verify in a proof assistant

We use the Univalence axiom [Voevodsky] and higher inductive types [Shulman, Lumsdaine].

Types

Homotopy type theory combines type theory with homotopy theory.

	type theory	**logic**	**homotopy theory**
A	type	formula	space*
$a : A$	term/element	proof	point
$A \times B$	product type	conjunction	binary product sp.
$A \to B$	function type	implication	mapping space
$P : A \to \text{Type}$	dependent type	predicate	fibration
$\Sigma(x : A),\ P(x)$	sigma type	ex. quantifier	total space
$\Pi(x : A),\ P(x)$	dep. fn. type	un. quantifier	space of sections
$a =_A b$	identity type	equality	path space

I will use these notions interchangably.

Path Induction

If we have a path $p : a = x$ where the right endpoint is a variable, we may assume that p is reflexivity and that x is a.

Example If A is a type with points x, y and z. If $p : x = y$ and $q : y = z$, we have a concatenation $p \cdot q : x = z$.

Path Induction

If we have a path $p : a = x$ where the right endpoint is a variable, we may assume that p is reflexivity and that x is a.

Example If A is a type with points x, y and z. If $p : x = y$ and $q : y = z$, we have a concatenation $p \cdot q : x = z$.

Proof Since the right endpoint of q is a variable, we may assume q is reflexivity and that y is z. Then we need to construct $p \cdot \mathsf{refl}_y : x = y$, which we define as $p \cdot \mathsf{refl}_y :\equiv p$.

Path Induction

If we have a path $p : a = x$ where the right endpoint is a variable, we may assume that p is reflexivity and that x is a.

Example If A is a type with points x, y and z. If $p : x = y$ and $q : y = z$, we have a concatenation $p \cdot q : x = z$.

Proof Since the right endpoint of q is a variable, we may assume q is reflexivity and that y is z. Then we need to construct $p \cdot \mathsf{refl}_y : x = y$, which we define as $p \cdot \mathsf{refl}_y :\equiv p$.

```
variables {A : Type} {w x y z : A}
definition concat (p : x = y) (q : y = z) : x = z :=
by induction q; exact p
```

Path Induction

If we have a path $p : a = x$ where the right endpoint is a variable, we may assume that p is reflexivity and that x is a.

Example If A is a type with points x, y and z. If $p : x = y$ and $q : y = z$, we have a concatenation $p \cdot q : x = z$.

Proof Since the right endpoint of q is a variable, we may assume q is reflexivity and that y is z. Then we need to construct $p \cdot \mathsf{refl}_y : x = y$, which we define as $p \cdot \mathsf{refl}_y :\equiv p$.

```
variables {A : Type} {w x y z : A}
definition concat (p : x = y) (q : y = z) : x = z :=
by induction q; exact p

definition con.assoc (p : w = x) (q : x = y) (r : y = z) :
  (p · q) · r = p · (q · r) :=
by induction r; reflexivity
```

Identity Type of Type Constructors

Homotopy of $f, g : A \to B$ can be defined as

$$(f \sim g) :\equiv \Pi(a : A),\ f(a) = g(a).$$

Homotopy equivalence $A \simeq B$ is:

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

Identity Type of Type Constructors

Homotopy of $f, g : A \to B$ can be defined as

$$(f \sim g) := \Pi(a : A),\ f(a) = g(a).$$

Homotopy equivalence $A \simeq B$ is:

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

We can characterize the equality in various types.

$(a, b) =_{A \times B} (a', b')$	is	$(a =_A a') \times (b =_B b')$	
$f = g$	is	$f \sim g$	(function extensionality)
$A =_{\mathrm{Type}} B$	is	$A \simeq B$	(univalence, Voevodsky)

Fundamental Group

Given a pointed type (A, a_0). The type $\Omega(A, a_0) := (a_0 = a_0, \mathsf{refl}_{a_0})$ has

- multiplication given by concatenation
- inverses given by path inverses
- identity given by reflexivity

These operations satisfy the group laws.

It is not quite a group since it has higher structure. For example, there might be multiple proofs that $(p \cdot q) \cdot r = p \cdot (q \cdot r)$.

We define $\pi_1(A, a_0) := \|\Omega(A, a_0)\|_0$, where $\|X\|_0$ is the set of connected components of X.
This construction is much shorter than the traditional definition.

Higher Inductive Types

In Type Theory there are inductive types, in which you specify its points.

Examples. $\mathbb{N}$ is generated by 0 and succ
$A + B$ is generated by either $a : A$ or $b : B$
$a =_A (-)$ is generated by $\mathsf{refl}_a : a =_A a$

In homotopy theory we can build cell complexes inductively.

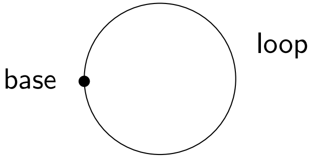
In HoTT we can combine these into higher inductive types [Shulman, Lumsdaine, 2012].

The Circle

Example. The circle $\mathbb{S}^1$

`HIT` $\mathbb{S}^1 :=$

- base : $\mathbb{S}^1$
- loop : base $=$ base



Recursion Principle. To define $f : \mathbb{S}^1 \to A$ we need to define $f(\text{base}) : A$ and a path $f(\text{base}) = f(\text{base})$ which is the path showing that f respects loop.

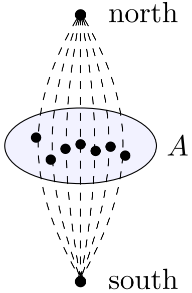
Using univalence, we can prove $\pi_1(\mathbb{S}^1) = \mathbb{Z}$.

The Suspension

Example. The suspension ΣA

`HIT` $\Sigma A :=$

- $\mathrm{north}, \mathrm{south} : \Sigma A$
- $\mathrm{merid} : A \to (\mathrm{north} = \mathrm{south})$



Remark. $\mathbb{S}^1 \simeq \Sigma\mathbf{2}$

Definition. The n-sphere is defined by $\mathbb{S}^{n+1} := \Sigma\mathbb{S}^n$ and $\mathbb{S}^0 := \mathbf{2}$

HoTT in Proof Assistants

There are various proof assistants supporting HoTT

- Coq (UniMath and Coq-HoTT)
- Agda
- Lean
- cubicaltt
- RedPRL

The Lean Theorem Prover

Lean is a new interactive theorem prover, developed principally by Leonardo de Moura at Microsoft Research, Redmond.

It was “announced” in the summer of 2015.

It is open source, released under a permissive license, Apache 2.0.

Lean's Kernel

Lean's kernel for HoTT implements dependent type theory with

- a hierarchy of universes:
  `Type.{0} : Type.{1} : Type.{2} : Type.{3} : ...`
- universe polymorphism:
  `definition id.{u} {A : Type.{u}} : A → A := λa, a`
- dependent products: `Πx : A, B`
- inductive types (à la Dybjer, constructors and recursors)

Lean's Kernel

The kernel is very small for a dependent type theory.

There are multiple reference checkers with about $1500 - 2000$ lines of code.

The kernel does not contain

- a termination checker
- fixpoint operators
- Pattern matching
- coinductive types
- inductive-inductive or inductive-recursive types
- universe cumulativity
- the eta rule for records

The HoTT Kernel

In the HoTT library, we add the following trusted components:

- the univalence axiom;
- two higher inductive types: quotients and truncation (a generalization of $\|-\|_0$).

```
HIT quotient (A : Type) (R : A → A → Type) : Type :=
| i : A → quotient A R
| e : Π{x y : A}, R x y → i x = i y
```

The HoTT Library

The HoTT library ($\sim$47k LOC) contains

- A good library with the basics of homotopy type theory: Path algebra, equivalences, truncated types, consequences of the univalence axiom, higher inductive types, pointed types;
- A category theory library;
- A large library of synthetic homotopy theory.

Contributors: vD, Jakob von Raumer, Ulrik Buchholtz, Jeremy Avigad, Egbert Rijke, Steve Awodey, Mike Shulman and others.

HITs in Lean

In Lean, the quotient and truncations are primitive HITs.

We define other HITs in terms of quotients and the truncation.

Not all HITs can be reduced to quotients. [Lumsdaine, Shulman, 2017]

It is not even clear what we mean by “all HITs.”

We have formalized many nontrivial reductions of commonly used HITs to quotients.

Synthetic Homotopy Theory

The library contains:

- Freudenthal suspension theorem
- Whitehead's Theorem
- Seifert-van Kampen theorem
- long exact sequence of homotopy groups
- complex and quaternionic Hopf fibration
- $\pi_k(\mathbb{S}^n)$ for $k \leq n$ and $\pi_3(\mathbb{S}^2)$.
- adjunction between the smash product and pointed maps.
- cohomology theory
- the Serre spectral sequence for cohomology
  - This is a very powerful tool to compute many homotopy and (co)homology groups of spaces.

Lean 3

The library was developed in an older version of Lean, Lean 2.

We are currently working on porting the library to the newest version, Lean 3.

This allows us to use the metaprogramming framework and write specialized automation.

Conclusions

- HoTT is a convenient language for homotopy theory.
  - It is more general than traditional homotopy theory;
  - The homotopy theoretic notions are primitives in type theory;
  - It gives novel ways of reasoning;
  - It is constructive (but not anti-classical);
  - It is possible to verify formally in practice.
- Lean is a good proof assistant for HoTT.
  - We have formalized significant results in homotopy theory, most notably a very general version of the Serre spectral sequence.