Introduction to Homotopy Type Theory

Lecture 1: Type theory from a homotopy theory perspective

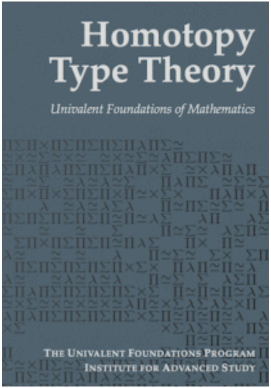
Fredrik Nordvall Forsberg
University of Strathclyde, Glasgow

EUTypes Summer school, Ohrid, 8 August 2018

Course plan

- Today: Type theory from a homotopy theory perspective
- Tomorrow: Equivalences, the Univalence Axiom
- Saturday: Propositional truncation, Univalent logic
- Sunday: Higher inductive types, synthethic homotopy theory

Main source material

- 

- Homotopy Type Theory blog
- Homotopy Type Theory Google group
- **Slides and exercises:** `https://tinyurl.com/hott-ohrid`

Homotopy Type Theory

Homotopy (Type Theory)

(Homotopy Type) Theory

Univalent Foundations and Homotopy Type Theory

Two separate origins:

- UF: Voevodsky [2010–].
- HoTT: Hofmann-Streicher [1995], Awodey-Warren [2009], Garner-van den Berg [2011], Lumsdaine [2010].

Vladimir Voevodsky (1966–2017)

Martin Hofmann (1965–2018)

Thomas Streicher

Steve Awodey

Michael Warren

Richard Garner

Benno van den Berg

Peter Lumsdaine

Univalent Foundations

A higher-dimensional foundation of mathematics: basic objects are not discrete sets, but ∞-groupoids.

Univalent Foundations

A higher-dimensional foundation of mathematics: basic objects are not discrete sets, but ∞-groupoids.

Main motivation: everything we write down should be invariant under equivalence.

Univalent Foundations

A higher-dimensional foundation of mathematics: basic objects are not discrete sets, but ∞-groupoids.

Main motivation: everything we write down should be invariant under equivalence.

Miracle: Martin-Löf Type Theory essentially already is such a foundation.

Univalent Foundations

A higher-dimensional foundation of mathematics: basic objects are not discrete sets, but ∞-groupoids.

Main motivation: everything we write down should be invariant under equivalence.

Miracle: Martin-Löf Type Theory essentially already is such a foundation.

Side remark: “univalent” derives from Russian word for “faithful” [Voevodsky IHP talk 2014].

Homotopy Type Theory

Started as investigations into models of Martin-Löf Type Theory into abstract homotopy theory.

Homotopy Type Theory

Started as investigations into models of Martin-Löf Type Theory into abstract homotopy theory.

Consequence: can prove results in homotopy theory synthetically using Type Theory, extended by axioms suggested by the models.

Homotopy Type Theory

Started as investigations into models of Martin-Löf Type Theory into abstract homotopy theory.

Consequence: can prove results in homotopy theory synthetically using Type Theory, extended by axioms suggested by the models.

Miracle: these axioms imply most of the “missing features” of plain Type Theory, such as function extensionality, and quotient types.

Homotopy Type Theory

Started as investigations into models of Martin-Löf Type Theory into abstract homotopy theory.

Consequence: can prove results in homotopy theory synthetically using Type Theory, extended by axioms suggested by the models.

Miracle: these axioms imply most of the "missing features" of plain Type Theory, such as function extensionality, and quotient types.

We now also know that these axioms are computationally well-behaved thanks to Cubical Type Theory [Cohen, Coquand, Huber, Mörtberg 2017].

Intuition

Type Theory	Interpretation
A type	space A
$a : A$	point a in space A
$A \equiv B$	spaces A and B are equal (on the nose)
$a \equiv a' : A$	points a and a' in space A are equal (on the nose)
$x : A \vdash B(x)$ type	fibration $B \to A$

Intuition

Type Theory	Interpretation
A type	space A
$a : A$	point a in space A
$A \equiv B$	spaces A and B are equal (on the nose)
$a \equiv a' : A$	points a and a' in space A are equal (on the nose)
$x : A \vdash B(x)$ type	fibration $B \to A$
$(\Sigma a : A)B(x)$	total space of the fibration $B \to A$
$(\Pi a : A)B(x)$	space of sections of fibration $B \to A$

Intuition

Type Theory	Interpretation
A type	space A
$a : A$	point a in space A
$A \equiv B$	spaces A and B are equal (on the nose)
$a \equiv a' : A$	points a and a' in space A are equal (on the nose)
$x : A \vdash B(x)$ type	fibration $B \to A$
$(\Sigma a : A)B(x)$	total space of the fibration $B \to A$
$(\Pi a : A)B(x)$	space of sections of fibration $B \to A$
$\mathbf{0}$	empty space
$\mathbf{1}$	trivial space
$\mathbf{2}$	discrete two-point space

Intuition

Type Theory	Interpretation
A type	space A
$a : A$	point a in space A
$A \equiv B$	spaces A and B are equal (on the nose)
$a \equiv a' : A$	points a and a' in space A are equal (on the nose)
$x : A \vdash B(x)$ type	fibration $B \to A$
$(\Sigma a : A)B(x)$	total space of the fibration $B \to A$
$(\Pi a : A)B(x)$	space of sections of fibration $B \to A$
$\mathbf{0}$	empty space
$\mathbf{1}$	trivial space
$\mathbf{2}$	discrete two-point space
universe $\mathcal{U}$	space of small spaces
$a =_A a'$	space of paths connecting a and a' in A

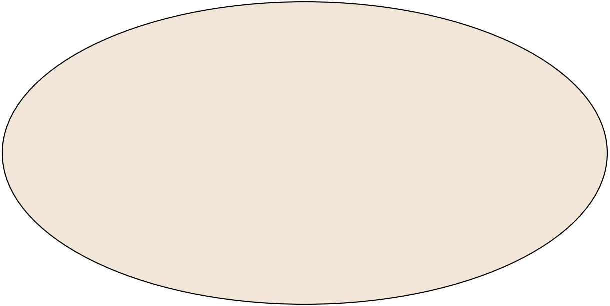
Remarks

A fibration is a "parameterised space with a homotopy lifting property" — the notion needed if identity is weakened to paths.

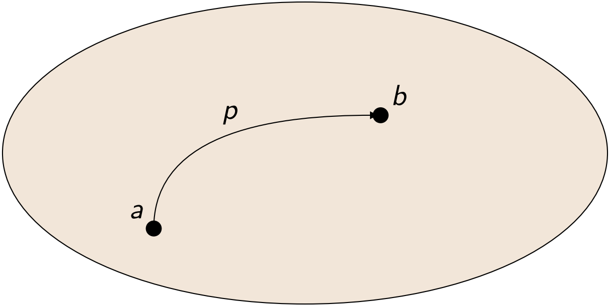
The total space of a fibration is the disjoint union of all the fibres.

A section is in particular a continuous function — worth keeping in mind when translating concepts.

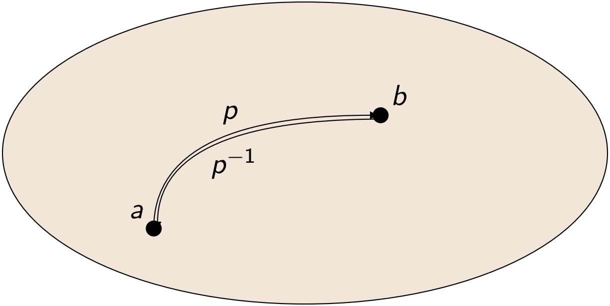
Thinking of identities as paths



Thinking of identities as paths

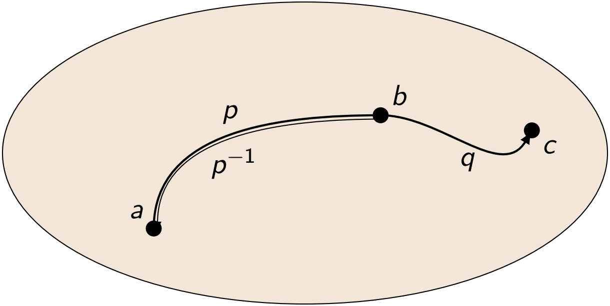


Thinking of identities as paths



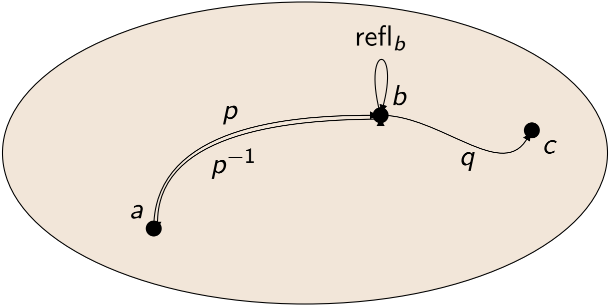
- Inverse paths p^{-1} (symmetry)

Thinking of identities as paths



- Inverse paths p^{-1} (symmetry)
- Path concatenation $p \cdot q$ (transitivity)

Thinking of identities as paths



- Inverse paths p^{-1} (symmetry)
- Path concatenation $p \cdot q$ (transitivity)
- Constant paths refl_b (reflexivity)

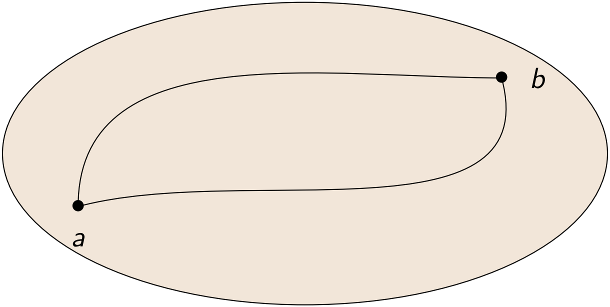
Identity Proofs are not Unique

There can be more than one path that connects two points, so we should not expect identity proofs to be unique.

Identity Proofs are not Unique

There can be more than one path that connects two points, so we should not expect identity proofs to be unique.

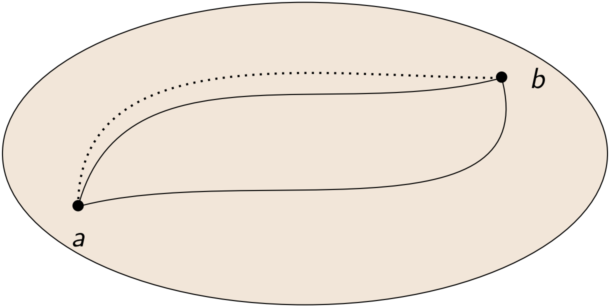
However, not because of this:



Identity Proofs are not Unique

There can be more than one path that connects two points, so we should not expect identity proofs to be unique.

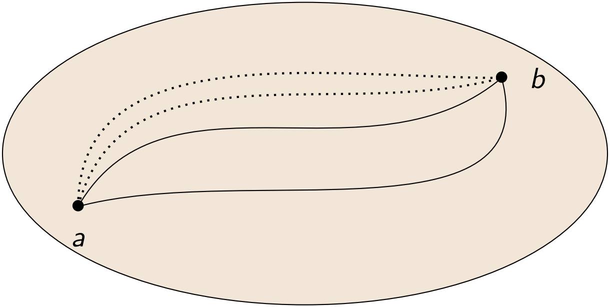
However, not because of this:



Identity Proofs are not Unique

There can be more than one path that connects two points, so we should not expect identity proofs to be unique.

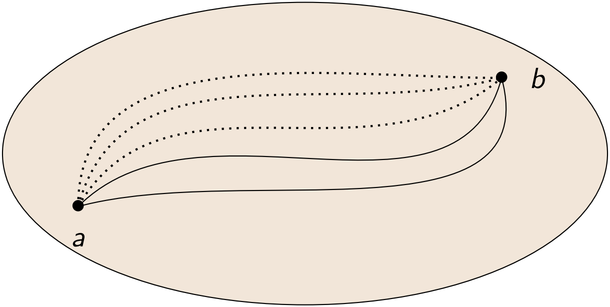
However, not because of this:



Identity Proofs are not Unique

There can be more than one path that connects two points, so we should not expect identity proofs to be unique.

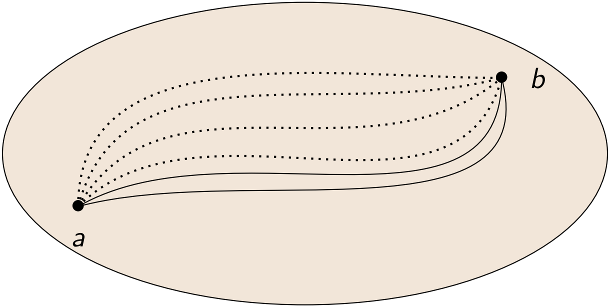
However, not because of this:



Identity Proofs are not Unique

There can be more than one path that connects two points, so we should not expect identity proofs to be unique.

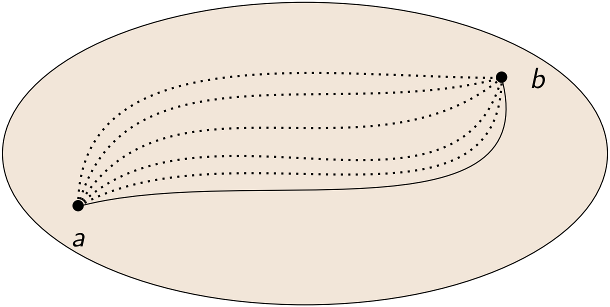
However, not because of this:



Identity Proofs are not Unique

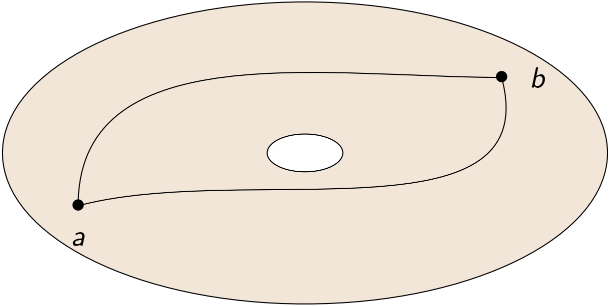
There can be more than one path that connects two points, so we should not expect identity proofs to be unique.

However, not because of this:

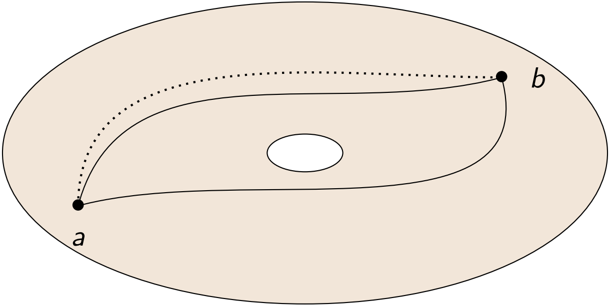


since there is a path (homotopy) between the paths.

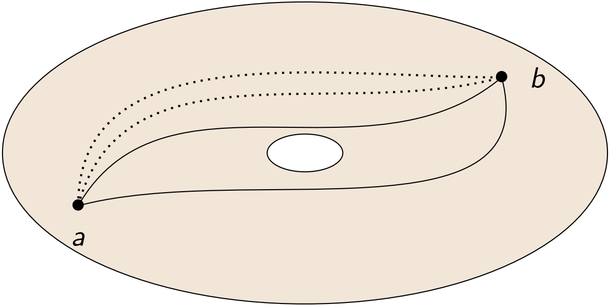
More problematic



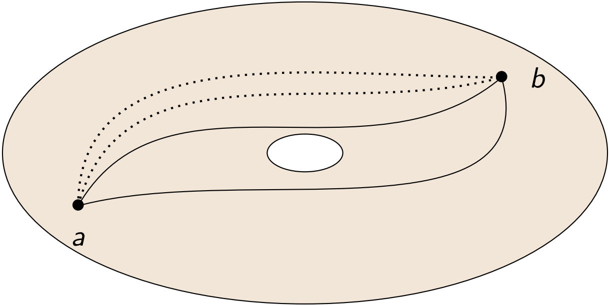
More problematic



More problematic

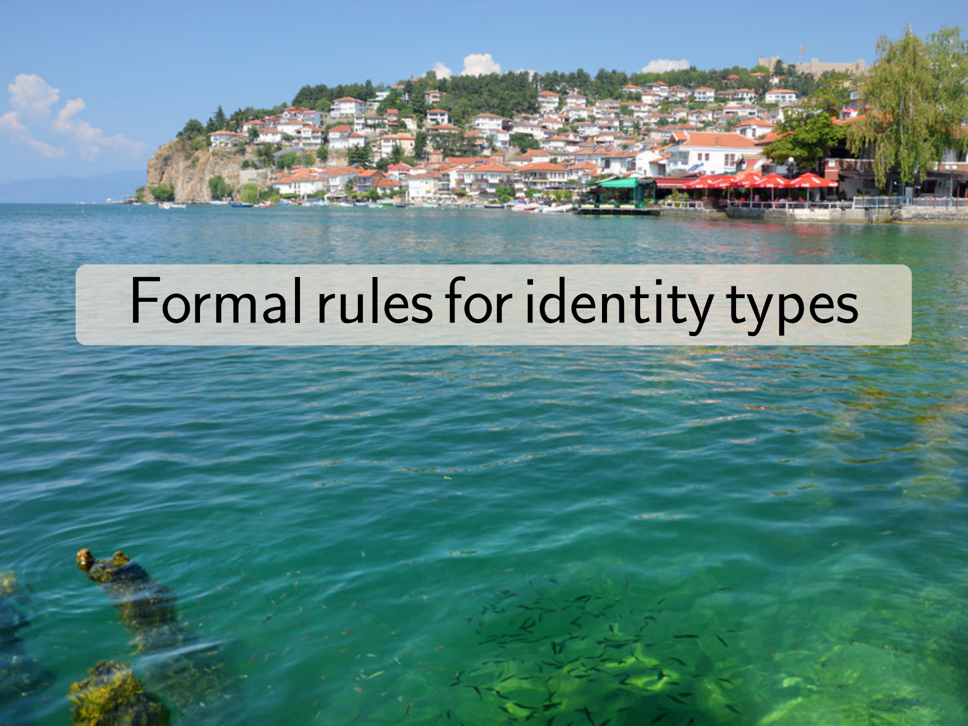


More problematic



Stuck!

Formal rules for identity types



Identity type rules

Formation If $a : A$ and $a' : A$ then $a =_A a'$ type.

Identity type rules

Formation If $a : A$ and $a' : A$ then $a =_A a'$ type.

Introduction If $a : A$ then $\mathsf{refl}_a : a =_A a$.

Identity type rules

Formation If $a : A$ and $a' : A$ then $a =_A a'$ type.

Introduction If $a : A$ then $\mathsf{refl}_a : a =_A a$.

Elimination If

- $x : A, y : A, p : x =_A y \vdash C(x, y, p)$ type,
- $x : A \vdash d(x) : C(x, x, \mathsf{refl}_x)$, and
- $a : A$, $a' : A$ and $p : a =_A a'$

then $\mathsf{ind}_{=_A}(C, d, a, a', p) : C(a, a', p)$.

Identity type rules

Formation If $a : A$ and $a' : A$ then $a =_A a'$ type.

Introduction If $a : A$ then $\mathsf{refl}_a : a =_A a$.

Elimination If

- $x : A, y : A, p : x =_A y \vdash C(x, y, p)$ type,
- $x : A \vdash d(x) : C(x, x, \mathsf{refl}_x)$, and
- $a : A$, $a' : A$ and $p : a =_A a'$

then $\mathsf{ind}_{=_A}(C, d, a, a', p) : C(a, a', p)$.

Computation $\mathsf{ind}_{=_A}(C, d, a, a, \mathsf{refl}_a) = d(a)$.

Identity type rules

Formation If $a : A$ and $a' : A$ then $a =_A a'$ type.

Introduction If $a : A$ then $\mathsf{refl}_a : a =_A a$.

Elimination If

- $x : A, y : A, p : x =_A y \vdash C(x, y, p)$ type,
- $x : A \vdash d(x) : C(x, x, \mathsf{refl}_x)$, and
- $a : A$, $a' : A$ and $p : a =_A a'$

then $\mathsf{ind}_{=_A}(C, d, a, a', p) : C(a, a', p)$.

Computation $\mathsf{ind}_{=_A}(C, d, a, a, \mathsf{refl}_a) = d(a)$.

Elimination, informally

In order to do something with an arbitrary $p : a =_A a'$, it suffices to consider the case $\mathsf{refl}_a : a =_A a$.

Equality is symmetric

In practice: if you can write it down, it is trivial to prove it.

Equality is symmetric

In practice: if you can write it down, it is trivial to prove it.

Theorem

If $p : a =_A b$ *then there is* $p^{-1} : b =_A a$*.*

Equality is symmetric

In practice: if you can write it down, it is trivial to prove it.

Theorem

If $p : a =_A b$ then there is $p^{-1} : b =_A a$.

Proof.

Consider elimination motive $C(x, y, q) \equiv y =_A x$. We can give $d(x) :\equiv \mathsf{refl}_x : C(x, x, \mathsf{refl})$, hence by the elimination principle we can take $p^{-1} :\equiv \mathsf{ind}_{=_A}(C, d, a, b, p) : b =_A a$. □

Equality is symmetric

In practice: if you can write it down, it is trivial to prove it.

Theorem

If $p : a =_A b$ then there is $p^{-1} : b =_A a$.

Proof.

Consider elimination motive $C(x, y, q) \equiv y =_A x$. We can give $d(x) :\equiv \mathsf{refl}_x : C(x, x, \mathsf{refl})$, hence by the elimination principle we can take $p^{-1} :\equiv \mathsf{ind}_{=_A}(C, d, a, b, p) : b =_A a$. □

Second proof.

By the elimination principle, we can assume p is refl, in which case we need to give $\mathsf{refl}_a^{-1} : a = a$. Obviously $\mathsf{refl}_a^{-1} :\equiv \mathsf{refl}_a$ works. □

Equality is transitive

Theorem

If $p : a = b$ *and* $q : b = c$ *then there is* $p \cdot q : a = c$.

Equality is transitive

Theorem

If $p : a = b$ *and* $q : b = c$ *then there is* $p \cdot q : a = c$.

Proof.

We may assume b is a and p is refl_a, in which case $q : a = c$ has the right type, so $\mathrm{refl} \cdot q :\equiv q$ works. □

Equality is transitive

Theorem

If $p : a = b$ and $q : b = c$ then there is $p \cdot q : a = c$.

Proof.

We may assume b is a and p is refl_a, in which case $q : a = c$ has the right type, so $\mathsf{refl} \cdot q :\equiv q$ works. □

Second proof.

Elimination with motive $C(x, y, r) \equiv (\Pi s : y =_A c)(x =_A c)$ applied to p (for r) and q (for s). □

Equality is unique?

Claim?

If $p, q : a =_A b$, then $p =_{a =_A b} q$?

Equality is unique?

Claim?

If $p, q : a =_A b$, then $p =_{a =_A b} q$?

Proof?

We may assume p and q are refl; if so, $\mathsf{refl}_{\mathsf{refl}_a}$ obviously works. □

Equality is unique?

Claim?

If $p, q : a =_A b$, then $p =_{a =_A b} q$?

Proof?

We may assume p and q are refl; if so, $\mathsf{refl}_{\mathsf{refl}_a}$ obviously works. □

Second proof?

Let's do this formally: we want to prove

$$(\Pi x, y : A)(\Pi r : x = y)(\Pi s : x = y)(r = s)$$

□

Equality is unique?

Claim?

If $p, q : a =_A b$, then $p =_{a =_A b} q$?

Proof?

We may assume p and q are refl; if so, $\mathrm{refl}_{\mathrm{refl}_a}$ obviously works. □

Second proof?

Let's do this formally: we want to prove

$$(\Pi x, y : A)(\Pi r : x = y)(\Pi s : x = y)(r = s)$$

so our motive should be $C(x, y, r) \equiv (\Pi s : x = y)(r = s)$.

□

Equality is unique?

Claim?

If $p, q : a =_A b$, then $p =_{a =_A b} q$?

Proof?

We may assume p and q are refl; if so, $\mathsf{refl}_{\mathsf{refl}_a}$ obviously works. □

Second proof?

Let's do this formally: we want to prove

$$(\Pi x, y : A)(\Pi r : x = y)(\Pi s : x = y)(r = s)$$

so our motive should be $C(x, y, r) \equiv (\Pi s : x = y)(r = s)$. We need to give $d(x) : C(x, x, \mathsf{refl})$

□

Equality is unique?

Claim?

If $p, q : a =_A b$, then $p =_{a =_A b} q$?

Proof?

We may assume p and q are refl; if so, $\mathrm{refl}_{\mathrm{refl}_a}$ obviously works. □

Second proof?

Let's do this formally: we want to prove

$$(\Pi x, y : A)(\Pi r : x = y)(\Pi s : x = y)(r = s)$$

so our motive should be $C(x, y, r) \equiv (\Pi s : x = y)(r = s)$. We need to give $d(x) : C(x, x, \mathrm{refl})$, i.e.

$$d(x) : (\Pi s : x = x)(\mathrm{refl} =_{x=x} s)$$

but we are stuck: the elimination rule does not apply! □

Equality is unique?

Claim?

If $p, q : a =_A b$, then $p =_{a =_A b} q$?

Proof?

We may assume p and q are refl; if so, $\mathsf{refl}_{\mathsf{refl}_a}$ obviously works. $\square$

Second proof?

Let's do this formally: we want to prove

$$(\Pi x, y : A)(\Pi r : x = y)(\Pi s : x = y)(r = s)$$

so our motive should be $C(x, y, r) \equiv (\Pi s : x = y)(r = s)$. We need to give $d(x) : C(x, x, \mathsf{refl})$, i.e.

$$d(x) : (\Pi s : x = x)(\mathsf{refl} =_{x=x} s)$$

but we are stuck: the elimination rule does not apply! $\square$

We could try to generalise the inner motive, but then refl does not type check anymore. We cannot write it down.

~~Equality is unique?~~

~~Claim?~~
If $p, q : a =_A b$, then $p =_{a=_A b} q$?

~~Proof?~~.
~~We may assume p and q are refl; if so, $\mathsf{refl}_{\mathsf{refl}_a}$ obviously works.~~ □

Second proof?

Let's do this formally: we want to prove

$$(\Pi x, y : A)(\Pi r : x = y)(\Pi s : x = y)(r = s)$$

so our motive should be $C(x, y, r) \equiv (\Pi s : x = y)(r = s)$. We need to give $d(x) : C(x, x, \mathsf{refl})$, i.e.

$$d(x) : (\Pi s : x = x)(\mathsf{refl} =_{x=x} s)$$

but we are stuck: the elimination rule does not apply! □

We could try to generalise the inner motive, but then refl does not type check anymore. We cannot write it down.

Groupoid structure of paths

Theorem

- $p \cdot \mathsf{refl}_b = p$
- $\mathsf{refl}_a \cdot p = p$
- $p \cdot p^{-1} = \mathsf{refl}_a$
- $p^{-1} \cdot p = \mathsf{refl}_b$
- $(p^{-1})^{-1} = p$
- $p \cdot (q \cdot r) = (p \cdot q) \cdot r$

Groupoid structure of paths

Theorem

- $p \cdot \mathsf{refl}_b = p$
- $\mathsf{refl}_a \cdot p = p$
- $p \cdot p^{-1} = \mathsf{refl}_a$
- $p^{-1} \cdot p = \mathsf{refl}_b$
- $(p^{-1})^{-1} = p$
- $p \cdot (q \cdot r) = (p \cdot q) \cdot r$

These equalities are not strict; they only hold up to paths, which in turn are coherent, but only up to higher paths, ...

Groupoid structure of paths

Theorem

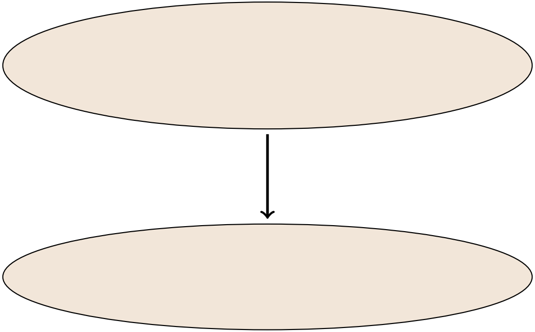
- $p \cdot \mathsf{refl}_b = p$
- $\mathsf{refl}_a \cdot p = p$
- $p \cdot p^{-1} = \mathsf{refl}_a$
- $p^{-1} \cdot p = \mathsf{refl}_b$
- $(p^{-1})^{-1} = p$
- $p \cdot (q \cdot r) = (p \cdot q) \cdot r$

These equalities are not strict; they only hold up to paths, which in turn are coherent, but only up to higher paths, . . .

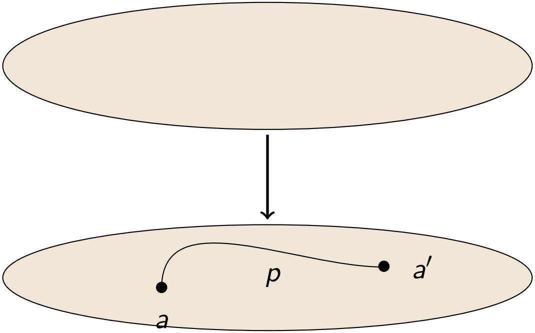
Theorem (Lumsdaine [2010], van den Berg-Garner [2011])

For every type A, $(A, =_A, =_{=_A}, \ldots)$ *form an* ∞*-groupoid.*

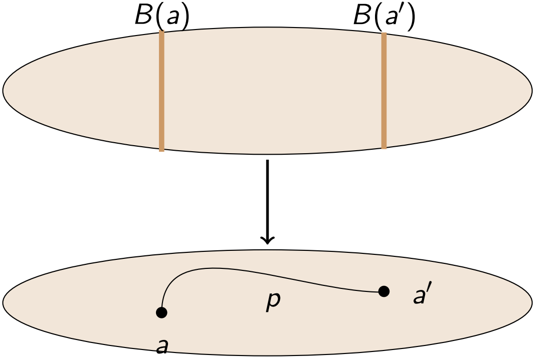
Transporting along paths



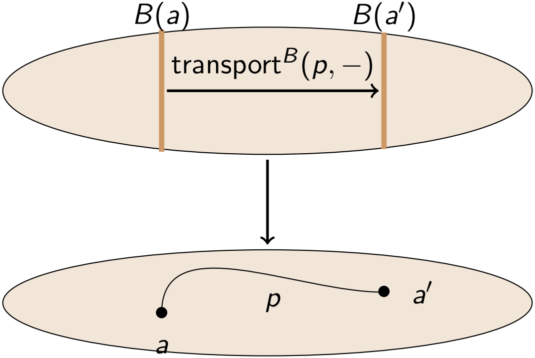
Transporting along paths



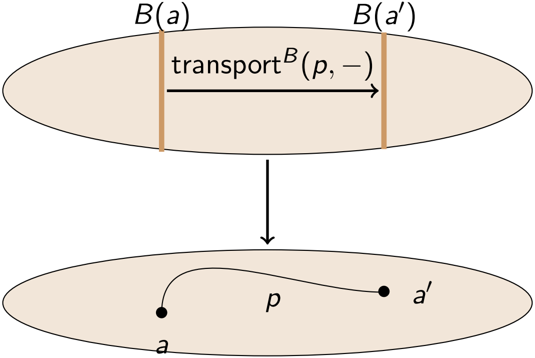
Transporting along paths



Transporting along paths



Transporting along paths



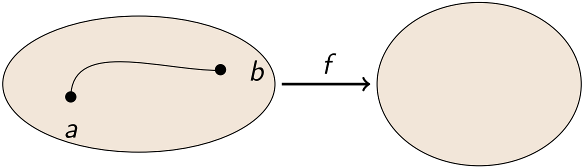
Theorem

Let $x : A \vdash B(x)$ *type. If* $p : a =_A a'$ *then there is*

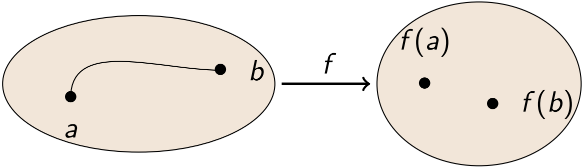
$$\mathsf{transport}^B(p, -) : B(a) \to B(a')$$

with $\mathsf{transport}^B(\mathsf{refl}_a, -) = \mathsf{id}_{B(a)}$.

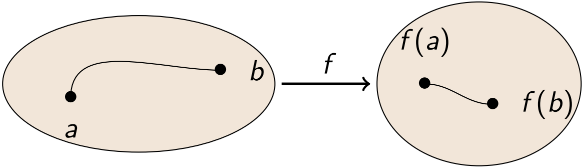
Functions act on paths



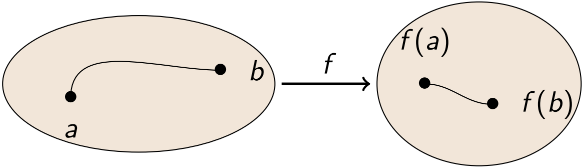
Functions act on paths



Functions act on paths



Functions act on paths



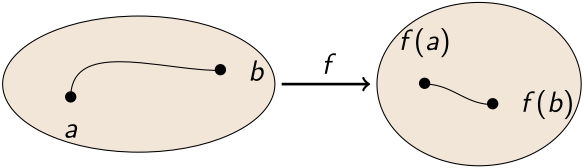
Theorem

Let $f : A \to B$*. There is*

$$\mathsf{ap}_f : (\Pi x, y : A)\big(x =_A y \to f(x) =_B f(y)\big)$$

with $\mathsf{ap}_f(x, x, \mathsf{refl}_x) :\equiv \mathsf{refl}_{f(x)}$.

Functions act on paths



Theorem

Let $f : A \to B$. *There is*

$$\mathsf{ap}_f : (\Pi x, y : A)\big(x =_A y \to f(x) =_B f(y)\big)$$

with $\mathsf{ap}_f(x, x, \mathsf{refl}_x) :\equiv \mathsf{refl}_{f(x)}$.

Theorem

Let $f : (\Pi x : A)B(x)$. *There is*

$$\mathsf{apd}_f : (\Pi x, y : A)(\Pi p : x =_A y)\big(f(x) \underset{p}{=} f(y)\big)$$

with $\mathsf{apd}_f(x, x, \mathsf{refl}_x) :\equiv \mathsf{refl}_{f(x)}$.

Paths over paths (dependent paths)

Formation If $a, a' : A$ and $p : a =_A a'$, and $b : B(a)$, $b' : B(a')$ then $b \underset{p}{=} b'$ type.

Paths over paths (dependent paths)

Formation If $a, a' : A$ and $p : a =_A a'$, and $b : B(a)$, $b' : B(a')$ then $b \underset{p}{=} b'$ type.

Introduction If $b : B(a)$ then $\mathsf{refl}_b : b \underset{\mathsf{refl}_a}{=} b$.

Paths over paths (dependent paths)

Formation If $a, a' : A$ and $p : a =_A a'$, and $b : B(a)$, $b' : B(a')$ then $b \underset{p}{=} b'$ type.

Introduction If $b : B(a)$ then $\mathsf{refl}_b : b \underset{\mathsf{refl}_a}{=} b$.

Elimination ...

Computation ...

Paths over paths (dependent paths)

Formation If $a, a' : A$ and $p : a =_A a'$, and $b : B(a)$, $b' : B(a')$ then $b \underset{p}{=} b'$ type.

Introduction If $b : B(a)$ then $\mathsf{refl}_b : b \underset{\mathsf{refl}_a}{=} b$.

Elimination ...

Computation ...

Can by implemented by e.g.

$$(b \underset{p}{=} b') :\equiv (\mathsf{transport}^B(p, b) =_{B(a')} b')$$

or

$$(b \underset{\mathsf{refl}_a}{=} b') :\equiv (b =_{B(a)} b') \qquad \text{(using path induction)}$$

Characterising path spaces

Transporting in Cartesian products

Theorem

$$\mathsf{transport}^{z \mapsto A(z) \times B(z)}(p, x) = (\mathsf{transport}^{A}(p, \mathsf{fst}(x)), \mathsf{transport}^{B}(p, \mathsf{snd}(x)))$$

Transporting in Cartesian products

Theorem

$$\mathsf{transport}^{z \mapsto A(z) \times B(z)}(p, x) = (\mathsf{transport}^{A}(p, \mathsf{fst}(x)), \mathsf{transport}^{B}(p, \mathsf{snd}(x)))$$

Proof.

It is enough to consider $p \equiv \mathsf{refl}_x$, in which case the problem reduces to $x = (\mathsf{fst}(x), \mathsf{snd}(x))$.

□

Transporting in Cartesian products

Theorem

$$\mathsf{transport}^{z \mapsto A(z) \times B(z)}(p, x) = (\mathsf{transport}^{A}(p, \mathsf{fst}(x)), \mathsf{transport}^{B}(p, \mathsf{snd}(x)))$$

Proof.

It is enough to consider $p \equiv \mathsf{refl}_x$, in which case the problem reduces to $x = (\mathsf{fst}(x), \mathsf{snd}(x))$. True by the η-rule (or an induction on x). □

Paths in Cartesian products

Given $p : (a, b) =_{A \times B} (a', b')$, we have

$$(\mathsf{ap}_{\mathsf{fst}}(p), \mathsf{ap}_{\mathsf{snd}}(p)) : (a =_A a') \times (b =_B b')$$

Conversely:

Theorem

There is a function

$$\mathsf{pair}^= : (a =_A a') \times (b =_B b') \to (a, b) =_{A \times B} (a', b')$$

Paths in Cartesian products

Given $p : (a, b) =_{A\times B} (a', b')$, we have

$$(\mathsf{ap}_{\mathsf{fst}}(p), \mathsf{ap}_{\mathsf{snd}}(p)) : (a =_A a') \times (b =_B b')$$

Conversely:

Theorem

There is a function

$$\mathsf{pair}^{=} : (a =_A a') \times (b =_B b') \to (a, b) =_{A\times B} (a', b')$$

Paths in Cartesian products

Given $p : (a, b) =_{A\times B} (a', b')$, we have

$$(\mathsf{ap}_{\mathsf{fst}}(p), \mathsf{ap}_{\mathsf{snd}}(p)) : (a =_A a') \times (b =_B b')$$

Conversely:

Theorem

There is a function

$$\mathsf{pair}^{=} : (a =_A a') \times (b =_B b') \to (a, b) =_{A\times B} (a', b')$$

These two maps are inverse to each other in a precise sense; more tomorrow, but for now, this can be summarised by:

Theorem

$$\big((a, b) =_{A\times B} (a', b')\big) \simeq \big((a =_A a') \times (b =_B b')\big)$$

In particular, we have $\mathsf{isEquiv}((\mathsf{ap}_{\mathsf{fst}}(-), \mathsf{ap}_{\mathsf{snd}}(-)))$.

Paths in sigma types

Suppose $a, a' : A$ and $b : B(a)$ and $b' : B(a')$. A path

$$(a, b) =_{(\Sigma x:A)B(x)} (a', b')$$

should consist of:

Paths in sigma types

Suppose $a, a' : A$ and $b : B(a)$ and $b' : B(a')$. A path

$$(a, b) =_{(\Sigma x:A)B(x)} (a', b')$$

should consist of:

- a path $p : a =_A a'$

Paths in sigma types

Suppose $a, a' : A$ and $b : B(a)$ and $b' : B(a')$. A path

$$(a, b) =_{(\Sigma x:A)B(x)} (a', b')$$

should consist of:

- a path $p : a =_A a'$
- a path $q : b =_{B(a)} b'$

Paths in sigma types

Suppose $a, a' : A$ and $b : B(a)$ and $b' : B(a')$. A path

$$(a, b) =_{(\Sigma x:A)B(x)} (a', b')$$

should consist of:

- a path $p : a =_A a'$
- ~~a path $q : b =_{B(a)} b'$~~ not well-typed!

Paths in sigma types

Suppose $a, a' : A$ and $b : B(a)$ and $b' : B(a')$. A path

$$(a, b) =_{(\Sigma x:A)B(x)} (a', b')$$

should consist of:

- a path $p : a =_A a'$
- ~~a path $q : b =_{B(a)} b'$~~ not well-typed!
- a path $q : b \underset{p}{=} b'$

Paths in sigma types

Suppose $a, a' : A$ and $b : B(a)$ and $b' : B(a')$. A path

$$(a, b) =_{(\Sigma x:A)B(x)} (a', b')$$

should consist of:

- a path $p : a =_A a'$
- ~~a path $q : b =_{B(a)} b'$~~ not well-typed!
- a path $q : b \underset{p}{=} b'$

Theorem

$$\big((a, b) =_{(\Sigma x:A)B(x)} (a', b')\big) \simeq (\Sigma p : a =_A a')(b \underset{p}{=} b')$$

In particular, we have $\mathrm{isEquiv}((\mathrm{ap}_{\mathsf{fst}}(-), \mathrm{apd}_{\mathsf{snd}}(-)))$.

Transporting in path types

A prime example of "if you can write it down, it will be trivial to prove it".

Transporting in path types

A prime example of "if you can write it down, it will be trivial to prove it".

Lemma

Let $a : A$ *and* $p : x =_A x'$.

- $\mathsf{transport}^{z \mapsto a = z}(p, q) = q \cdot p$
- $\mathsf{transport}^{z \mapsto z = a}(p, q) = p^{-1} \cdot q$

Transporting in path types

A prime example of “if you can write it down, it will be trivial to prove it”.

Lemma

Let $a : A$ *and* $p : x =_A x'$*.*

- $\mathsf{transport}^{z \mapsto a = z}(p, q) = q \cdot p$
- $\mathsf{transport}^{z \mapsto z = a}(p, q) = p^{-1} \cdot q$
- $\mathsf{transport}^{z \mapsto z = z}(p, q) =$

Transporting in path types

A prime example of "if you can write it down, it will be trivial to prove it".

Lemma

Let $a : A$ *and* $p : x =_A x'$*.*

- $\mathsf{transport}^{z \mapsto a = z}(p, q) = q \cdot p$
- $\mathsf{transport}^{z \mapsto z = a}(p, q) = p^{-1} \cdot q$
- $\mathsf{transport}^{z \mapsto z = z}(p, q) = p^{-1} \cdot q \cdot p$

Transporting in path types

A prime example of "if you can write it down, it will be trivial to prove it".

Lemma

Let $a : A$ *and* $p : x =_A x'$. *Let* $f, g : A \to B$.

- $\text{transport}^{z \mapsto a=z}(p, q) = q \cdot p$
- $\text{transport}^{z \mapsto z=a}(p, q) = p^{-1} \cdot q$
- $\text{transport}^{z \mapsto z=z}(p, q) = p^{-1} \cdot q \cdot p$
- $\text{transport}^{z \mapsto f(z)=g(z)}(p, q) =$

Transporting in path types

A prime example of "if you can write it down, it will be trivial to prove it".

Lemma

Let $a : A$ *and* $p : x =_A x'$. *Let* $f, g : A \to B$.

- $\mathsf{transport}^{z \mapsto a = z}(p, q) = q \cdot p$
- $\mathsf{transport}^{z \mapsto z = a}(p, q) = p^{-1} \cdot q$
- $\mathsf{transport}^{z \mapsto z = z}(p, q) = p^{-1} \cdot q \cdot p$
- $\mathsf{transport}^{z \mapsto f(z) = g(z)}(p, q) = \mathsf{ap}_f(p^{-1}) \cdot q \cdot ap_g(p)$

Transporting in path types

A prime example of "if you can write it down, it will be trivial to prove it".

Lemma

Let $a : A$ *and* $p : x =_A x'$. *Let* $f, g : A \to B$.

- $\mathrm{transport}^{z \mapsto a = z}(p, q) = q \cdot p$
- $\mathrm{transport}^{z \mapsto z = a}(p, q) = p^{-1} \cdot q$
- $\mathrm{transport}^{z \mapsto z = z}(p, q) = p^{-1} \cdot q \cdot p$
- $\mathrm{transport}^{z \mapsto f(z) = g(z)}(p, q) = \mathrm{ap}_f(p^{-1}) \cdot q \cdot ap_g(p)$

We don't expect a general characterisation of paths in $=_A$ — this will depend on A.

Paths in pi types

Suppose $f, g : (\Pi x : A)B(x)$. What should a path

$$f =_{(\Pi x:A)B(x)} g$$

consist of?

Paths in pi types

Suppose $f, g : (\Pi x : A)B(x)$. What should a path

$$f =_{(\Pi x:A)B(x)} g$$

consist of?

Theorem (using the Univalence Axiom)

$$\big(f =_{(\Pi x:A)B(x)} g\big) \simeq (\Pi x : A)(f(x) =_{B(x)} g(x))$$

In particular, we have isEquiv(happly)*, where*

$$\mathsf{happly} : \big(f = g\big) \to (\Pi x : A)(f(x) =_{B(x)} g(x))$$

is defined by $\mathsf{happly}(p, x) = \mathsf{ap}_{h \mapsto h(x)}(p)$.

Strong function extensionality from weak

Before HoTT, it was common to assume as an axiom a term

$$\mathsf{funext} : (\Pi x : A)(f(x) =_{B(x)} g(x)) \to \big(f = g\big)$$

(the non-trivial direction of $\big(f = g\big) \simeq (\Pi x : A)(f(x) =_{B(x)} g(x))$).

Strong function extensionality from weak

Before HoTT, it was common to assume as an axiom a term

$$\mathrm{funext} : (\Pi x : A)(f(x) =_{B(x)} g(x)) \to \big(f = g\big)$$

(the non-trivial direction of $\big(f = g\big) \simeq (\Pi x : A)(f(x) =_{B(x)} g(x))$).

Surprisingly, this weaker statement implies the stronger one:

Theorem (Voevodsky [Lumsdaine, HoTT blog])

If there is a term funext *as above, then* isEquiv(happly)*, i.e.*

$$\big(f =_{(\Pi x:A)B(x)} g\big) \simeq (\Pi x : A)(f(x) =_{B(x)} g(x))$$

Strong function extensionality from weak

Before HoTT, it was common to assume as an axiom a term

$$\mathsf{funext} : (\Pi x : A)(f(x) =_{B(x)} g(x)) \to \big(f = g\big)$$

(the non-trivial direction of $\big(f = g\big) \simeq (\Pi x : A)(f(x) =_{B(x)} g(x))$).

Surprisingly, this weaker statement implies the stronger one:

Theorem (Voevodsky [Lumsdaine, HoTT blog])
If there is a term funext *as above, then* isEquiv(happly)*, i.e.*

$$\big(f =_{(\Pi x:A)B(x)} g\big) \simeq (\Pi x : A)(f(x) =_{B(x)} g(x))$$

In cubical type theory, funext is trivial to define.

Paths in the universe

Suppose $A, B : \mathcal{U}$. What should a path

$$A =_{\mathcal{U}} B$$

consist of?

Paths in the universe

Suppose $A, B : \mathcal{U}$. What should a path

$$A =_{\mathcal{U}} B$$

consist of? Only sensible notion(?): $(A =_{\mathcal{U}} B) \simeq (A \simeq B)$.

Paths in the universe

Suppose $A, B : \mathcal{U}$. What should a path

$$A =_{\mathcal{U}} B$$

consist of? Only sensible notion(?): $(A =_{\mathcal{U}} B) \simeq (A \simeq B)$.

Again we can be more precise: we can define

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

by path induction: if $p : A = B$ is refl_A, we let

$$\mathsf{idtoeqv}(\mathsf{refl}_A) :\equiv \mathfrak{id}_A \equiv (\mathsf{id}_A, \mathsf{id}_A, \ldots)$$

Univalence Axiom

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

in particular, we have $\mathsf{isEquiv}(\mathsf{idtoeqv})$.

Paths in the universe

Suppose $A, B : \mathcal{U}$. What should a path

$$A =_{\mathcal{U}} B$$

consist of? Only sensible notion(?): $(A =_{\mathcal{U}} B) \simeq (A \simeq B)$.

Again we can be more precise: we can define

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

by path induction: if $p : A = B$ is refl_A, we let

$$\mathsf{idtoeqv}(\mathsf{refl}_A) :\equiv \mathfrak{id}_A \equiv (\mathsf{id}_A, \mathsf{id}_A, \ldots)$$

Univalence Axiom

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

in particular, we have isEquiv(idtoeqv).

In cubical type theory, Univalence is a theorem, not an axiom.

Consequences of Univalence

- "Isomorphic structures are equal"
- Propositional extensionality: $(P \leftrightarrow Q) \simeq (P = Q)$ for propositions P, Q.
- Function extensionality
- Large quotients exists
- Homotopy theory is non-trivial (there are two paths $\mathbf{2} =_{\mathcal{U}} \mathbf{2}$)
- Enough slack for large elimination of higher inductive types (Sunday)
- ...

Summary

New perspective on identity types based on intuitions from homotopical models.

Lack of uniqueness of identity proofs leads to path algebra: "if you can write it down, it is trivial to prove it".

Important characterisations/axioms: function extensionality and Univalence (more tomorrow).

Exercises

1. How does $\mathsf{transport}^B$ interact with the groupoid structure of paths? What about ap_f? Prove your claims.
2. State and prove lemmas for decomposing a transport in function types and sigma types (the latter is messier).
3. Use paths over paths to state and prove that the empty vector is a unit for vector concatenation, and that vector concatenation is associative. (Hint: you will need to generalise ap_f to paths over paths.)

References

S. Awodey and M. Warren
Homotopy theoretic models of identity types
Mathematical Proceedings of the Cambridge Philosophical Society 146, no. 1, 45–55, 2009.

M. Hofmann and T. Streicher
The groupoid interpretation of type theory
Twenty-five years of constructive type theory, pp. 83–111, 1998.

C. Cohen, T. Coquand, S. Huber and A. Mörtberg
Cubical Type Theory: a constructive interpretation of the univalence axiom
To appear in the post-proceedings of TYPES 2016

P. Lumsdaine
Weak ω-Categories from Intensional Type Theory
Logical Methods in Computer Science, Vol. 6, issue 23, paper 24, 2010

B. van den Berg and R. Garner
Types are weak ω-groupoids
Proceedings of the London Mathematical Society (3) 102, pp. 370–394, 2011

P. Lumsdaine
Strong functional extensionality from weak
Homotopy Type Theory blog,
https://homotopytypetheory.org/2011/12/19/strong-funext-from-weak

V. Voevodsky
The equivalence axiom and univalent models of type theory
Talk at CMU on February 4, 2010