

Computer-Checked Proofs in the Logic of Homotopy Theory

Dan Licata
Institute for Advanced Study

Homotopy theory

A branch of topology,
the study of spaces and continuous deformations



[image from wikipedia]

Homotopy

Deformation of one path into another

a

p

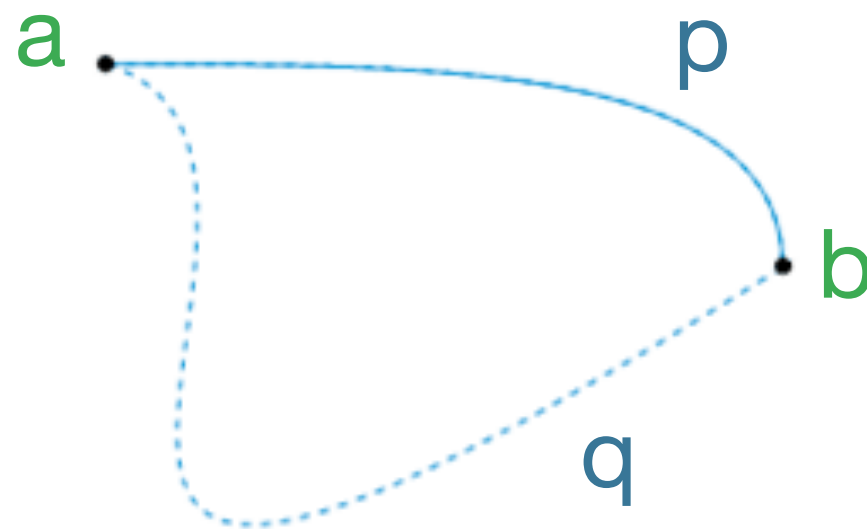
b

q

[image from wikipedia]

Homotopy

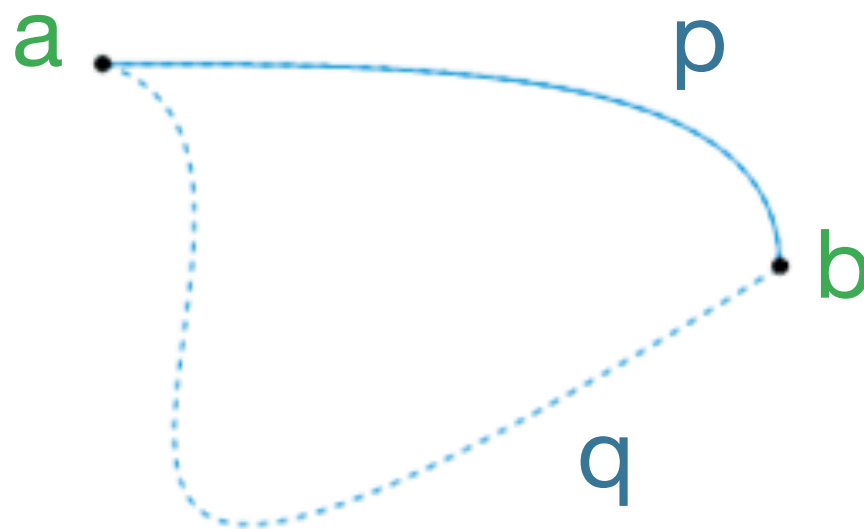
Deformation of one path into another



[image from wikipedia]

Homotopy

Deformation of one path into another

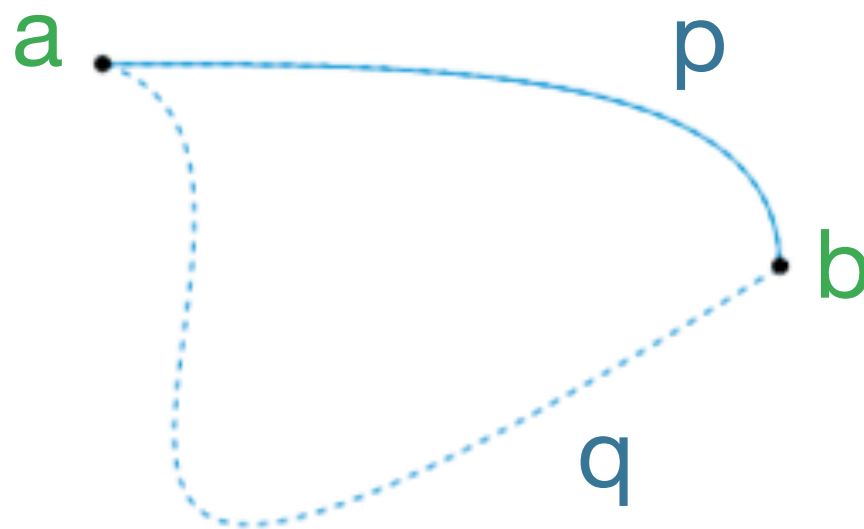


= 2-dimensional *path between paths*

[image from wikipedia]

Homotopy

Deformation of one path into another



= 2-dimensional *path between paths*

Homotopy theory is the study of spaces by way of their paths, homotopies, homotopies between homotopies,

[image from wikipedia]

Synthetic vs Analytic

Synthetic geometry (Euclid)

POSTULATES.

I.

LET it be granted that a straight line may be drawn from any one point to any other point.

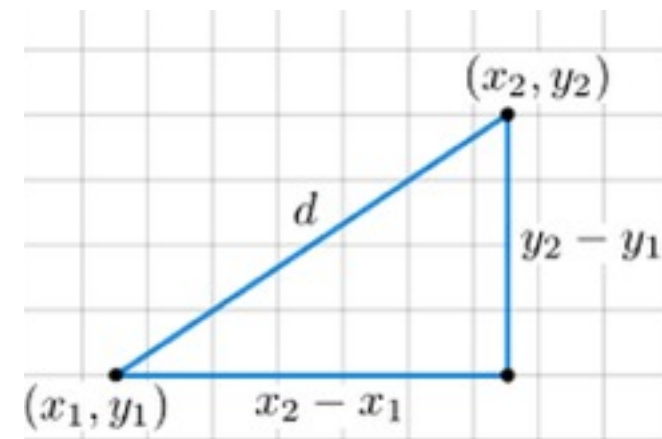
II.

That a terminated straight line may be produced to any length in a straight line.

III.

And that a circle may be described from any centre, at any distance from that centre.

Analytic geometry (Descartes)



[image from wikipedia]

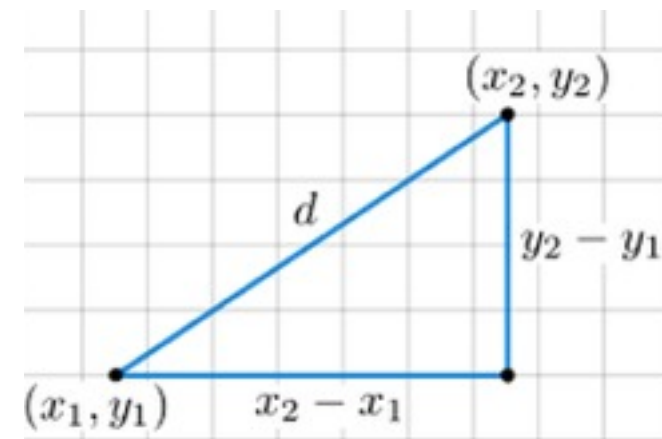
Synthetic vs Analytic

Synthetic geometry (Euclid)

POSTULATES.

- I.
LET it be granted that a straight line may be drawn from any one point to any other point.
- II.
That a terminated straight line may be produced to any length in a straight line.
- III.
And that a circle may be described from any centre, at any distance from that centre.

Analytic geometry (Descartes)



Classical homotopy theory is analytic:

- * a space is a set of points equipped with a topology
- * a path is a continuous map $[0,1] \rightarrow X$

[image from wikipedia]

Synthetic homotopy theory

homotopy theory

space

points

paths

homotopies

⋮

type theory

$\langle \text{type} \rangle$

$\langle \text{element} \rangle : \langle \text{type} \rangle$

$\langle \text{proof} \rangle : \langle \text{elem}_1 \rangle = \langle \text{elem}_2 \rangle$

$\langle 2\text{-proof} \rangle : \langle \text{proof}_1 \rangle = \langle \text{proof}_2 \rangle$

⋮

Synthetic homotopy theory

homotopy theory

space
points
paths
homotopies
⋮

type theory

$\langle \text{type} \rangle$
 $\langle \text{element} \rangle : \langle \text{type} \rangle$
 $\langle \text{proof} \rangle : \langle \text{elem}_1 \rangle = \langle \text{elem}_2 \rangle$
 $\langle 2\text{-proof} \rangle : \langle \text{proof}_1 \rangle = \langle \text{proof}_2 \rangle$
⋮

$\text{Id}(\langle \text{elem}_1 \rangle, \langle \text{elem}_2 \rangle)$




Synthetic homotopy theory

homotopy theory

space
points
paths
homotopies
⋮

type theory

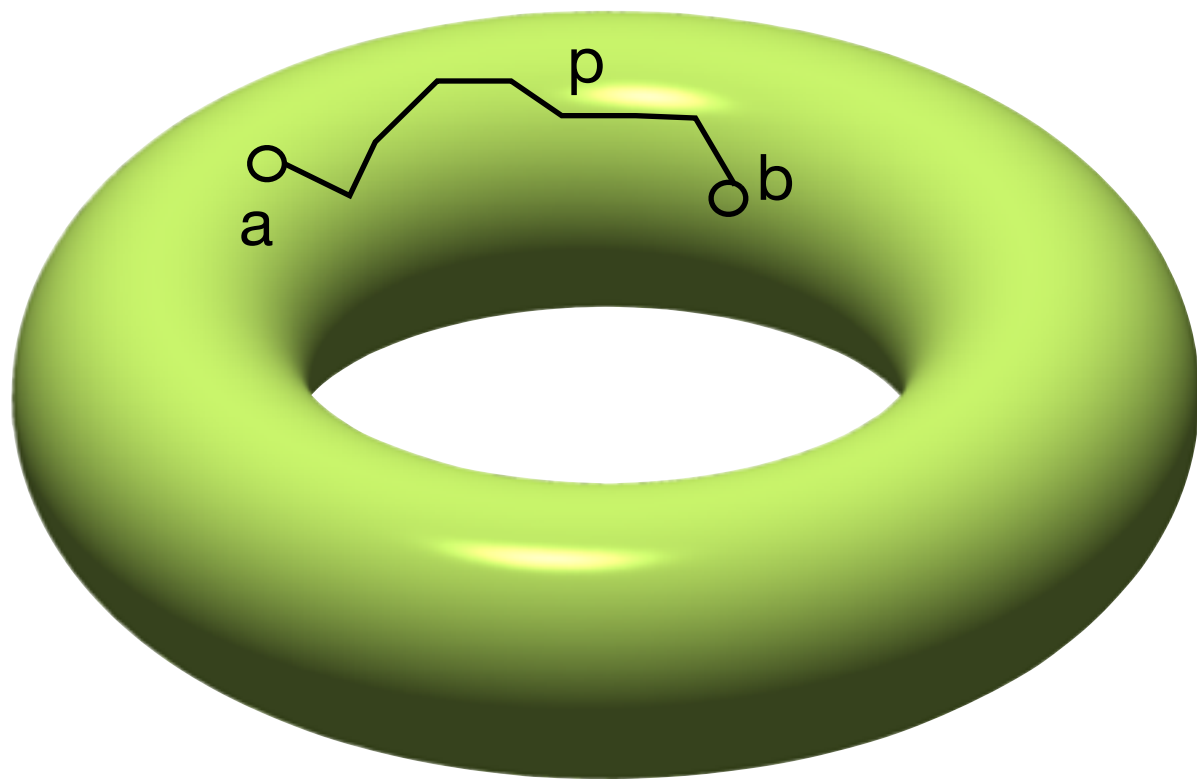
<type>
<element> : <type>
<proof> : <elem₁> = <elem₂>
<2-proof> : <proof₁> = <proof₂>
⋮



*A path is **not** a map $[0, 1] \rightarrow X$; it is a basic notion*

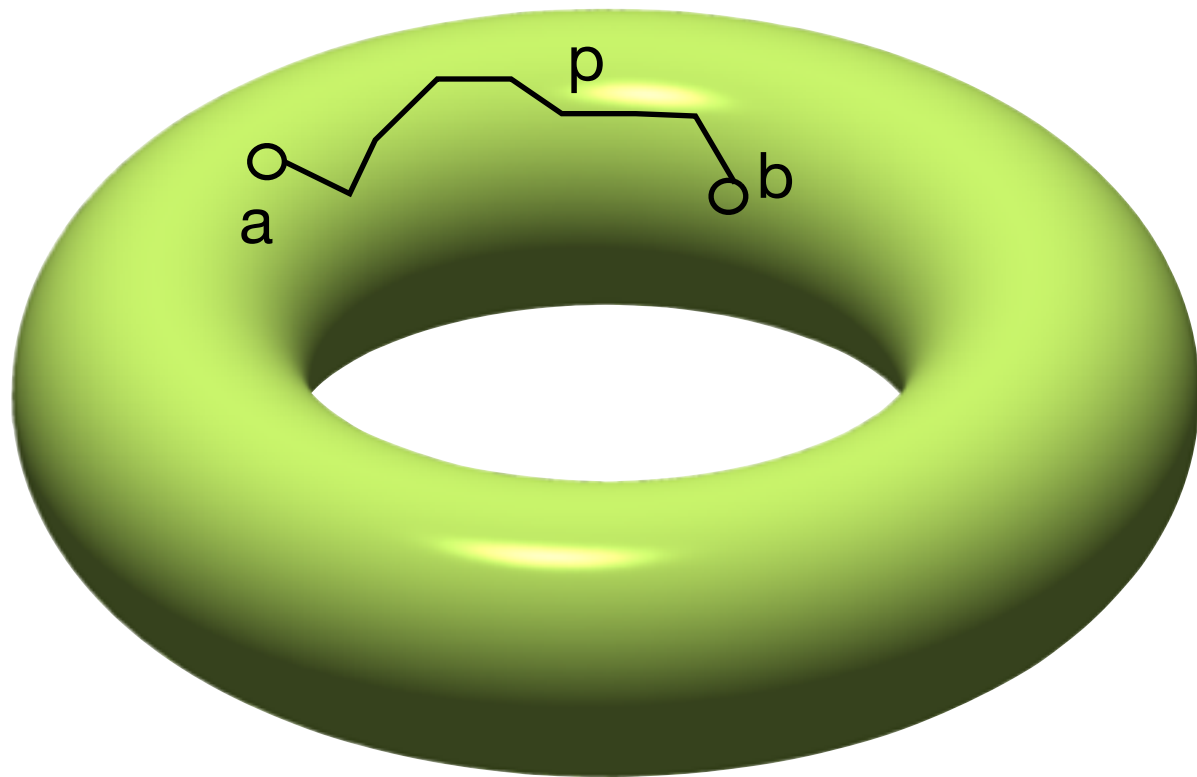
$\text{Id}(\text{<elem1>, <elem2>})$

Spaces as types



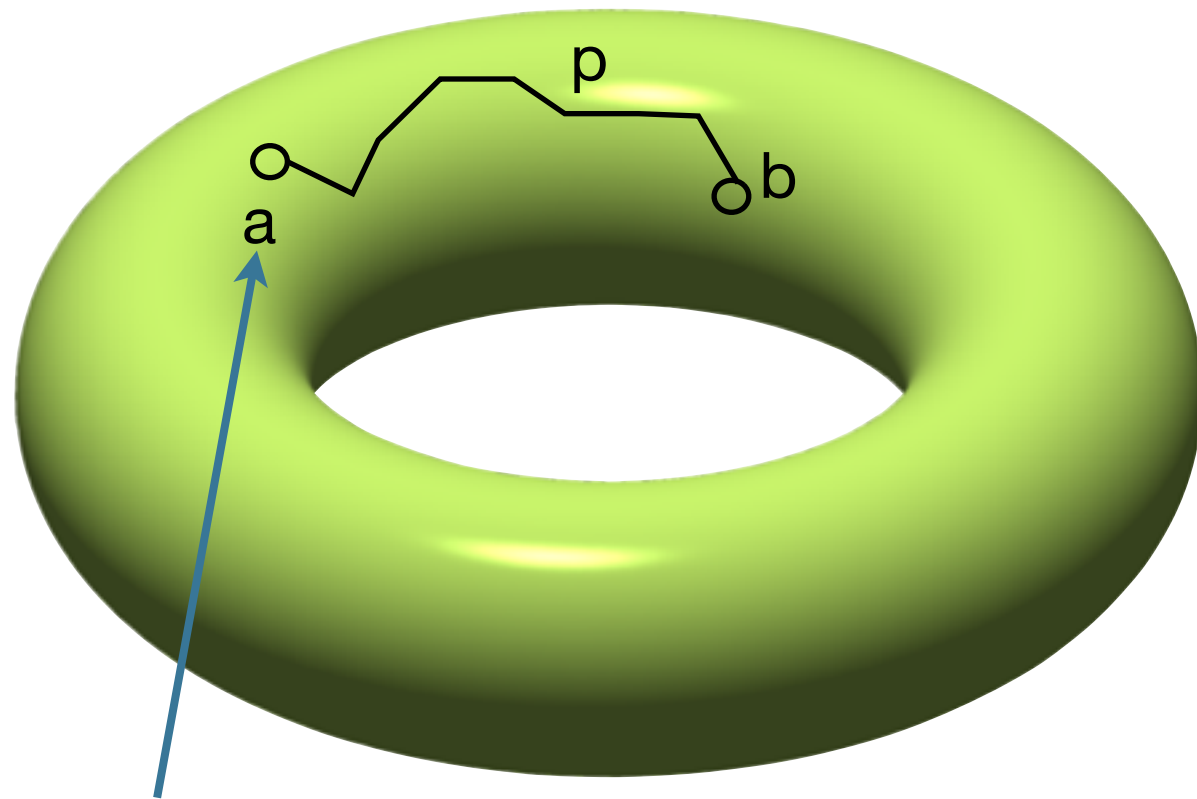
Spaces as types

a space is a type A



Spaces as types

a space is a type A

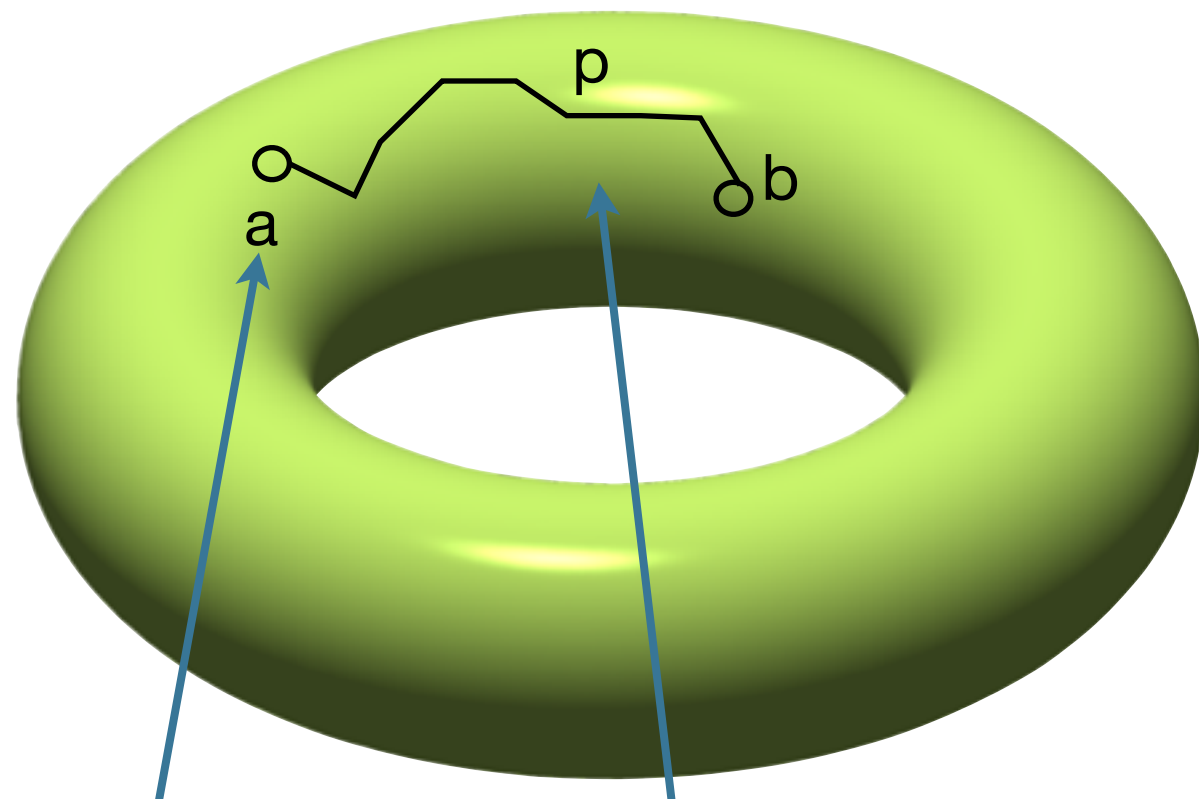


**points are
elements**

$a : A$

Spaces as types

a space is a type A



**points are
elements**

$a : A$

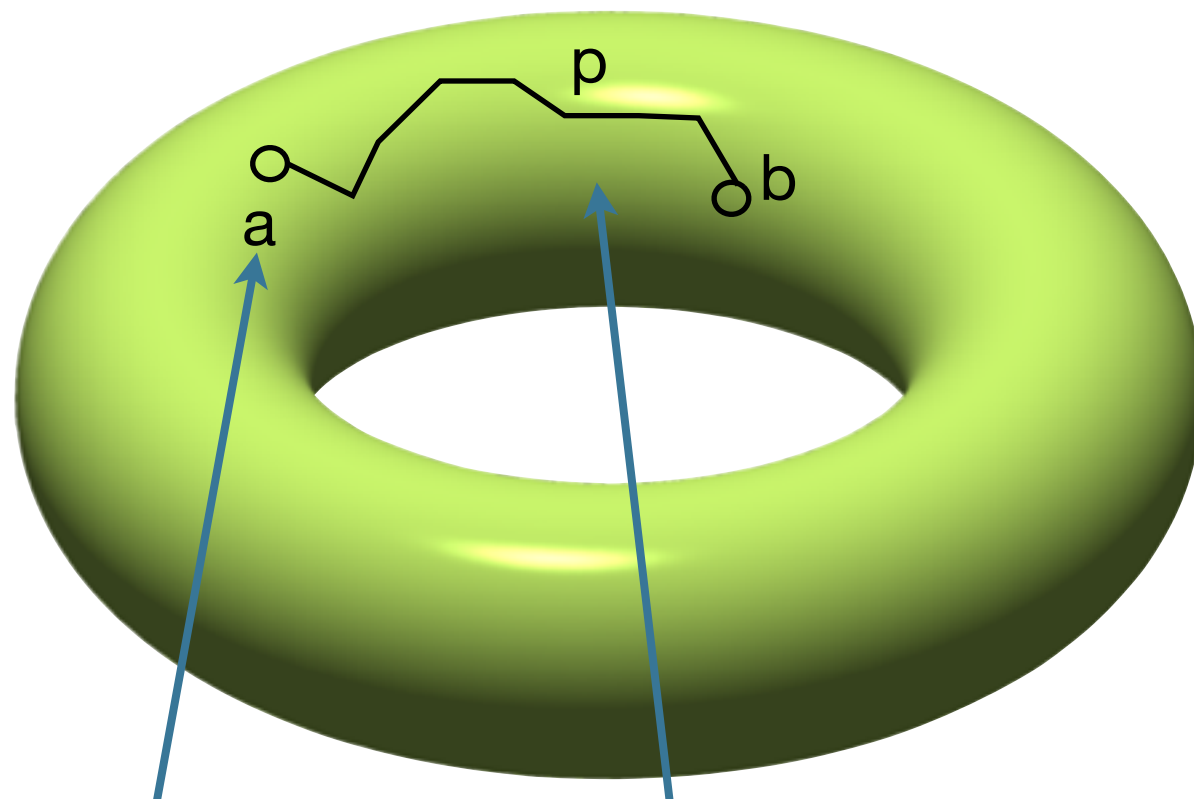
**paths are
*proofs of equality***

$p : a =_A b$

Spaces as types

a space is a type A

path operations



points are
elements

$a : A$

paths are
proofs of equality

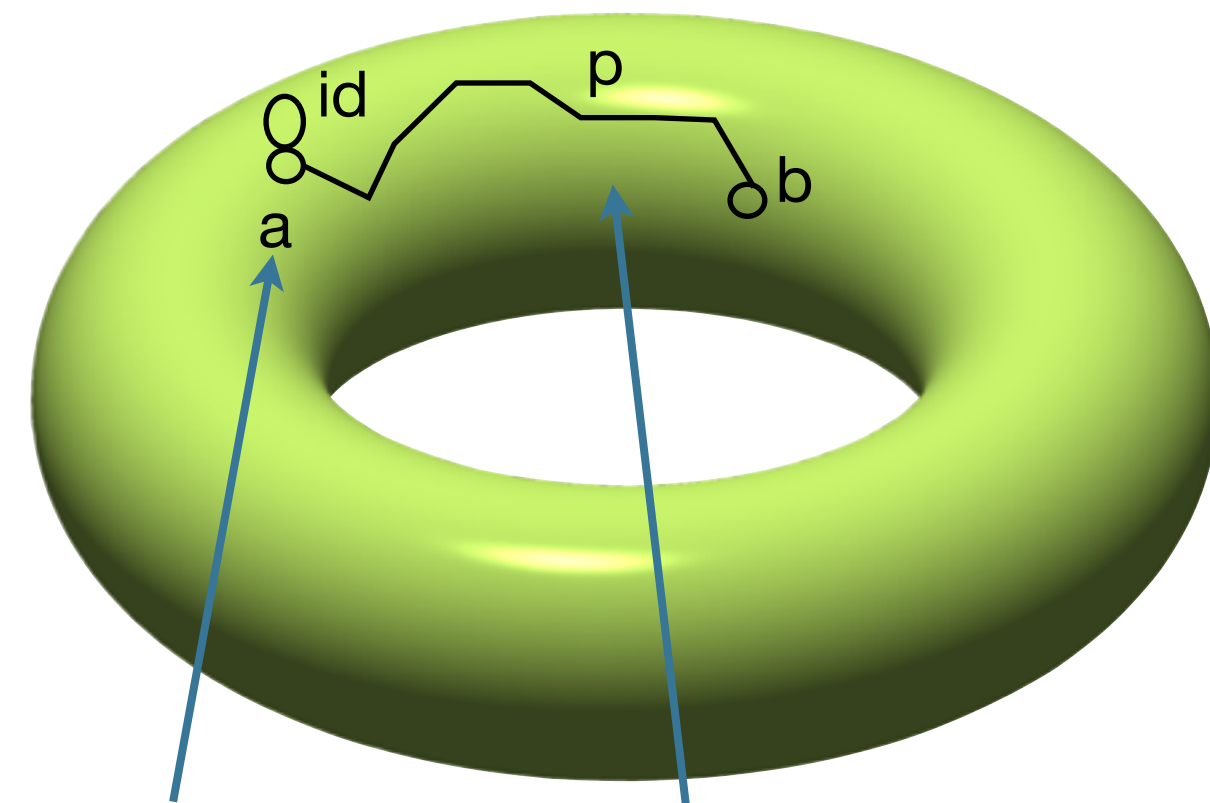
$p : a =_A b$

Spaces as types

a space is a type A

path operations

$\text{id} : a = a \text{ (refl)}$



**points are
elements**

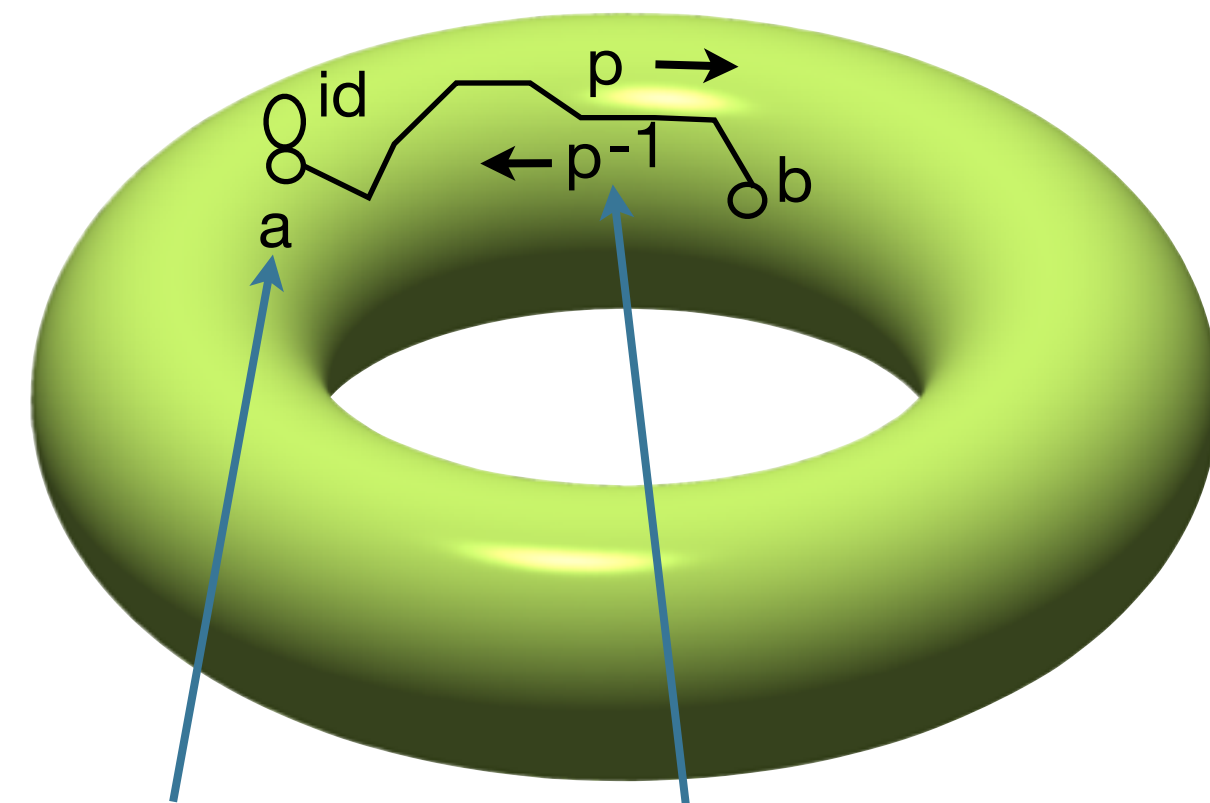
$a : A$

**paths are
*proofs of equality***

$p : a =_A b$

Spaces as types

a space is a type A



points are
elements

$a : A$

paths are
proofs of equality

$p : a =_A b$

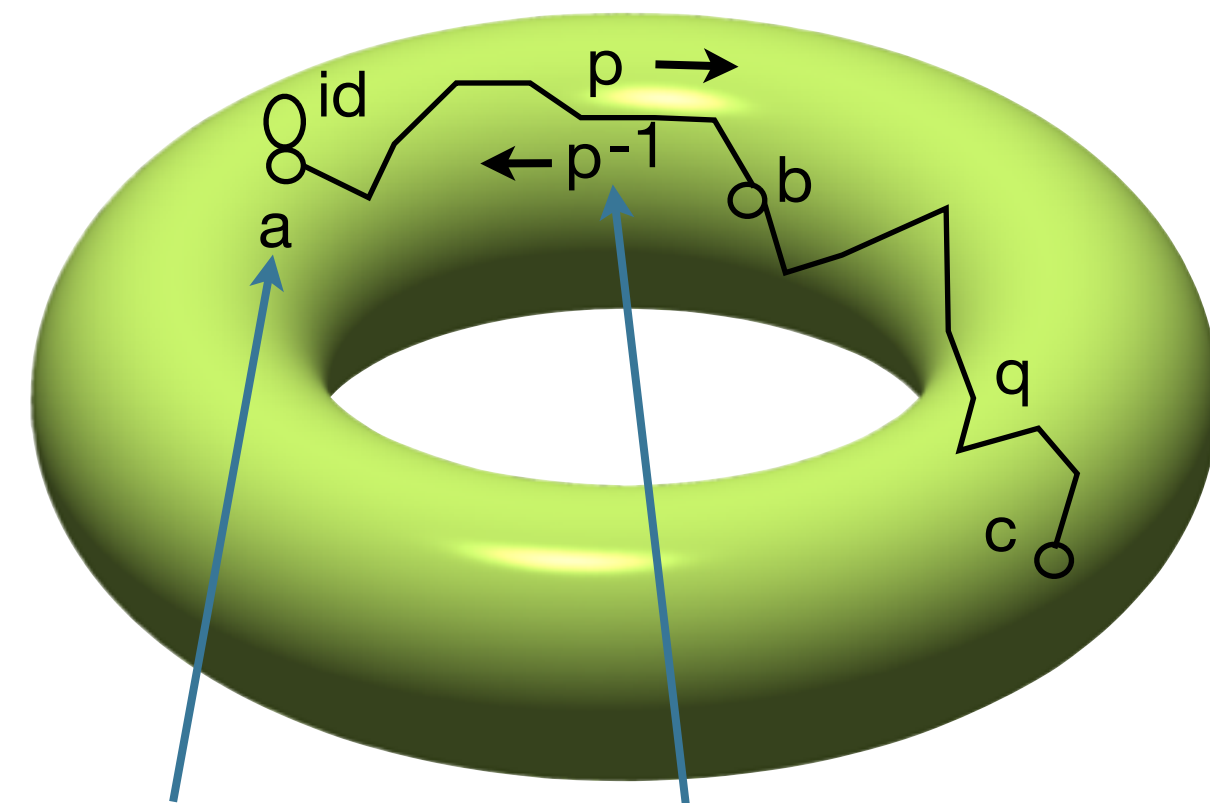
path operations

$\text{id} : a = a \text{ (refl)}$

$p^{-1} : b = a \text{ (sym)}$

Spaces as types

a space is a type A



points are
elements

$a : A$

paths are
proofs of equality

$p : a =_A b$

path operations

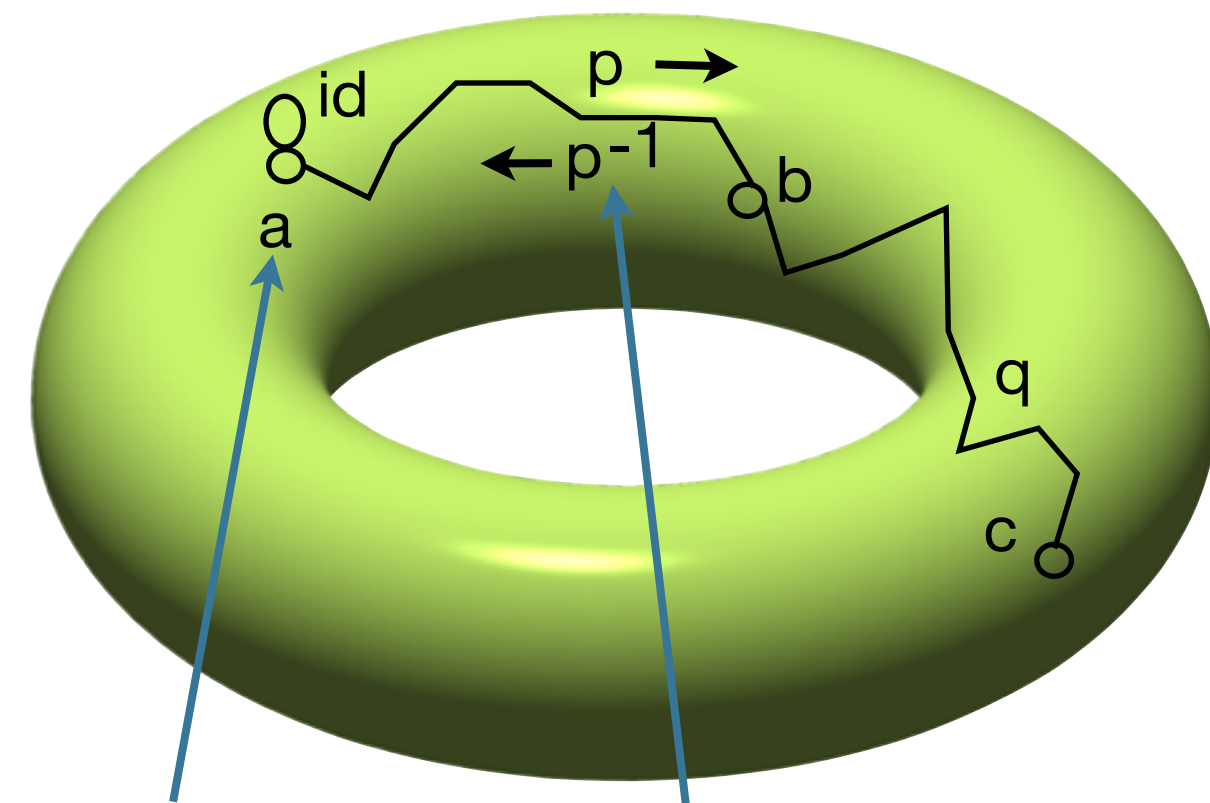
$\text{id} : a = a \text{ (refl)}$

$p^{-1} : b = a \text{ (sym)}$

$q \circ p : a = c \text{ (trans)}$

Spaces as types

a space is a type A



points are
elements

$a : A$

paths are
proofs of equality
 $p : a =_A b$

path operations

$\text{id} : a = a \text{ (refl)}$

$p^{-1} : b = a \text{ (sym)}$

$q \circ p : a = c \text{ (trans)}$

homotopies

$\text{id} \circ p = p$

$p^{-1} \circ p = \text{id}$

$r \circ (q \circ p)$
 $= (r \circ q) \circ p$

Spaces as types

a space is a type A

path operations

$\text{id} : a = a \text{ (refl)}$

$p^{-1} : b = a \text{ (sym)}$

$q \circ p : a = c \text{ (trans)}$

homotopies

$\text{id} \circ p = p$

$p^{-1} \circ p = \text{id}$

$r \circ (q \circ p)$
 $= (r \circ q) \circ p$

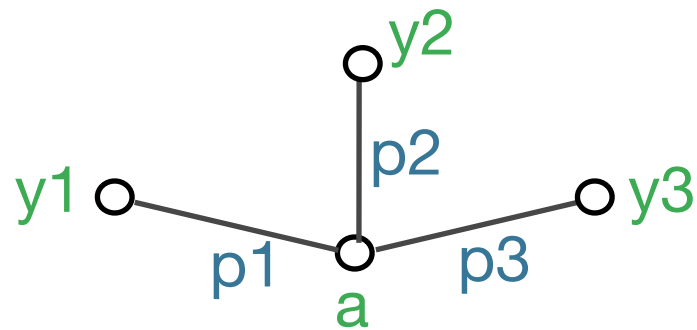
**points are
elements**

$a : A$

**paths are
*proofs of equality***
 $p : a =_A b$

Path induction

**Type of paths
from a to somewhere**



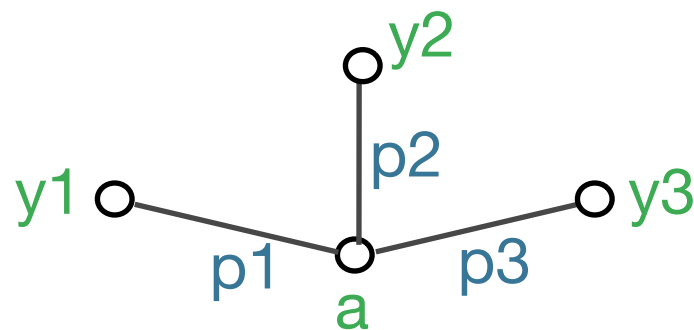
**is inductively
generated by**



Path induction

Type of paths
from a to somewhere

is inductively
generated by

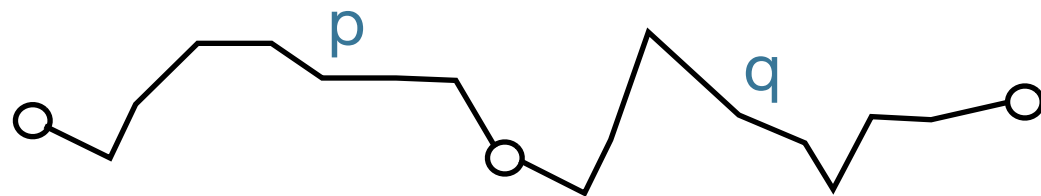


Fix a type A with element $a:A$.

For a family of types $C(y:A, p:a=y)$,
to give an element of

$C(y, p)$ for all y and $p:a=y$,
suffices to give an element of
 $C(a, id)$

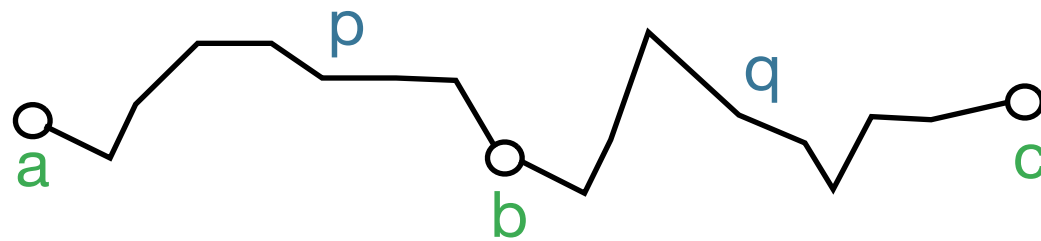
Composition, analytically



Given paths p and $q : [0, 1] \rightarrow X$ where $p(1) = q(0)$ define **composition** by:

$$(q \circ p)(x) = \begin{cases} p(2x) & \text{if } 0 \leq x \leq 1/2 \\ q(2x - 1) & \text{if } 1/2 \leq x \leq 1 \end{cases}$$

Composition, synthetically

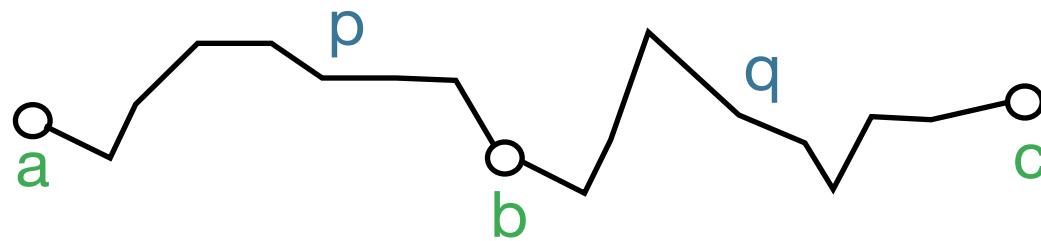


Given paths $p : a = b$ and $q : b = c$

define **composition** ($q \circ p$) by path induction:



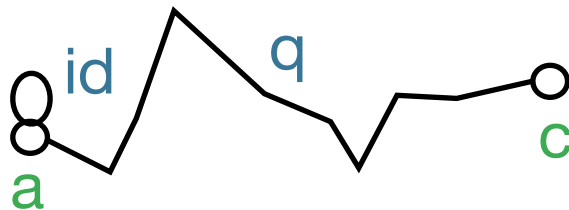
Composition, synthetically



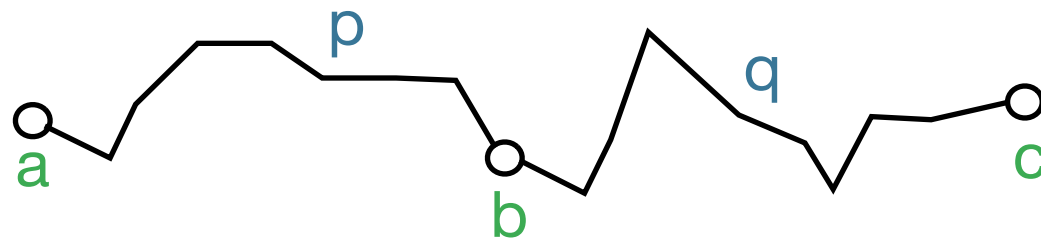
Given paths $p : a = b$ and $q : b = c$

define **composition** ($q \circ p$) by path induction:

✱ Suffices to consider case where b is a , and p is id



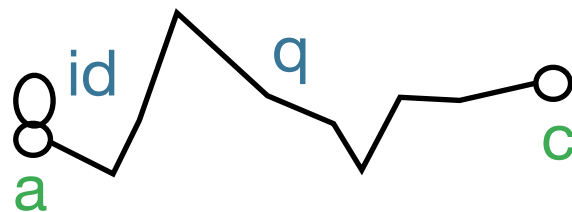
Composition, synthetically



Given paths $p: a=b$ and $q: b=c$

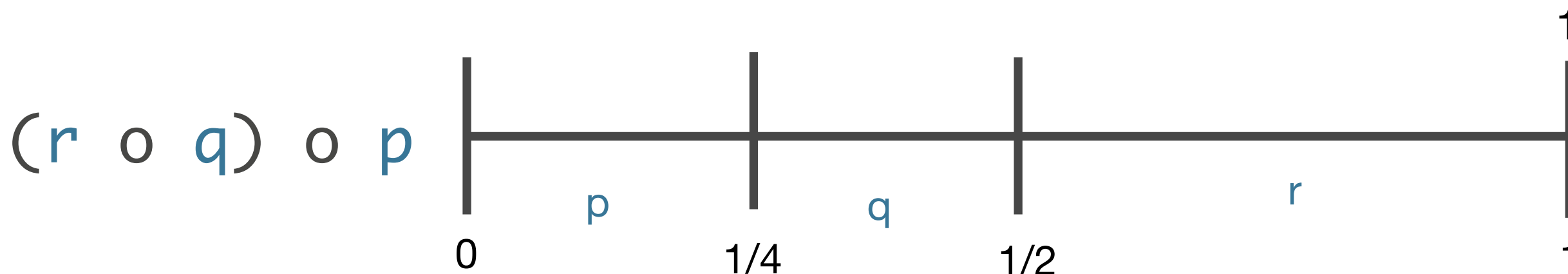
define **composition** ($q \circ p$) by path induction:

- * Suffices to consider case where b is a , and p is id

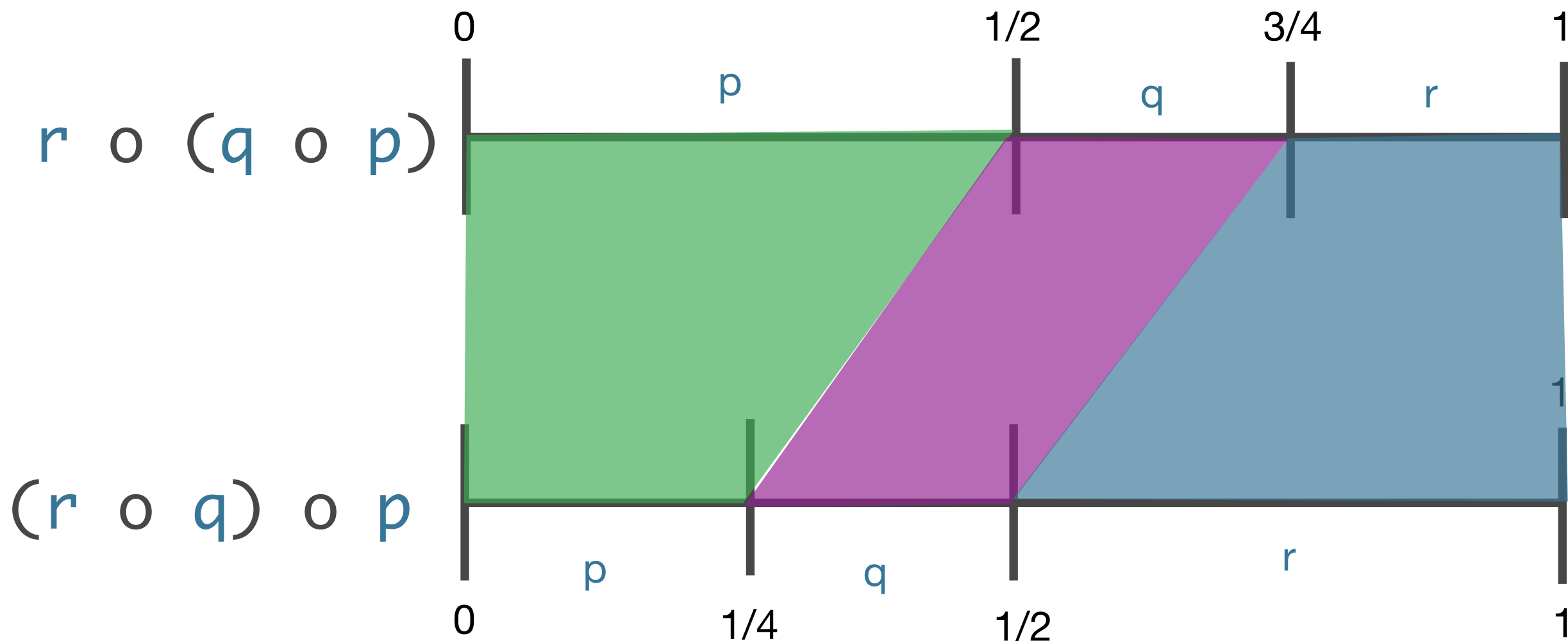
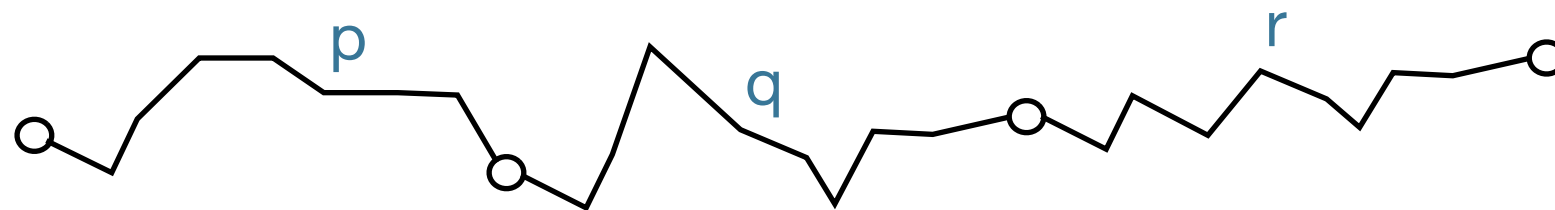


- * In this case the composite is q

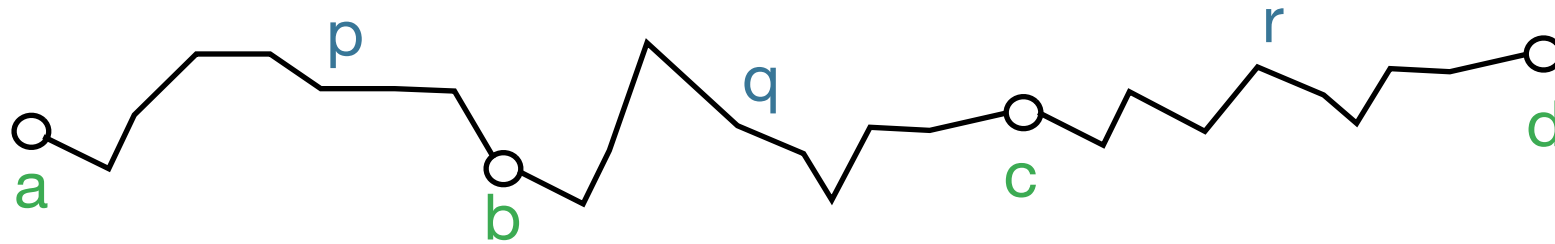
Associativity, analytically



Associativity, analytically

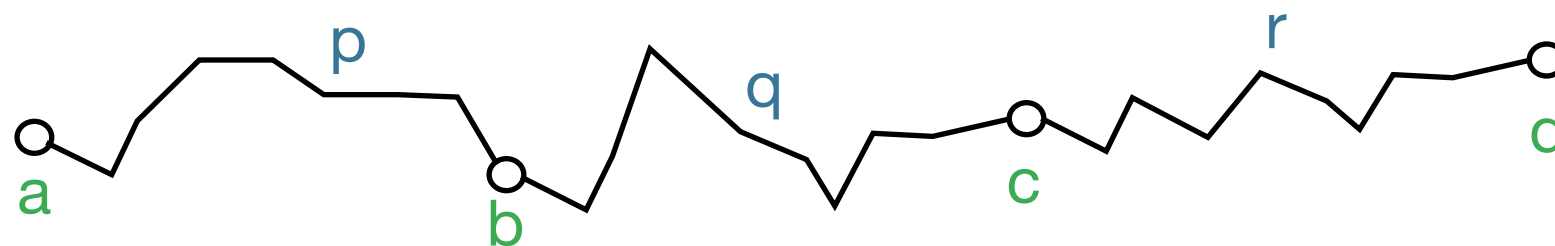


Associativity, synthetically



$$\forall a, b, c, d, p : a = b, q : b = c, r : c = d. \\ r \circ (q \circ p) = (r \circ q) \circ p$$

Associativity, synthetically

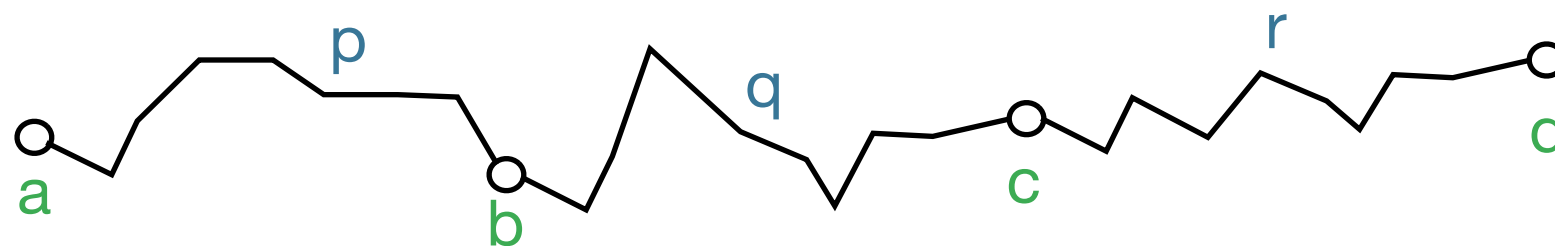


$$\forall a, b, c, d, p: a=b, q: b=c, r: c=d.$$
$$r \circ (q \circ p) = (r \circ q) \circ p$$

- * By path induction, suffices to consider case where all points are a and all paths are id :

$$id \circ (id \circ id) = (id \circ id) \circ id$$

Associativity, synthetically



$$\forall a, b, c, d, p: a=b, q: b=c, r: c=d.$$

$$r \circ (q \circ p) = (r \circ q) \circ p$$

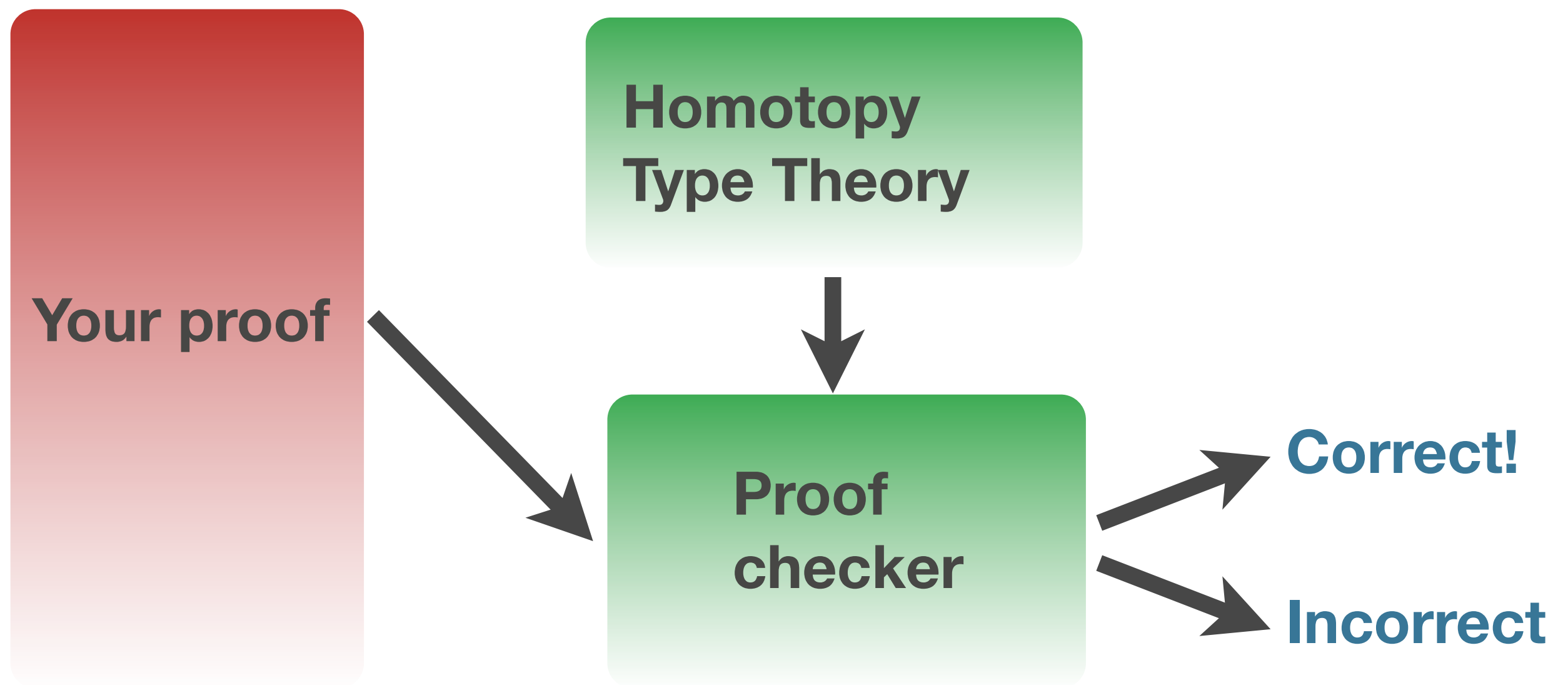
- * By path induction, suffices to consider case where all points are a and all paths are id :

$$id \circ (id \circ id) = (id \circ id) \circ id$$

- * By definition of \circ , both sides equal id

Type theory is
a **logic** of
homotopy theory

Computer-checked proofs



Computer-checked proofs

```
_◦_   : {A : Type} {a b c : A}
        → Path b c → Path a b → Path a c
q ◦ id = q
```

```
◦-assoc : {A : Type} {a b c d : A}
          (p : Path a b) (q : Path b c) (r : Path c d)
          → Path (r ◦ (q ◦ p)) ((r ◦ q) ◦ p)
◦-assoc id id id = id
```

We can do
computer-checked proofs
in **synthetic** homotopy theory

We can do
computer-checked proofs
in **synthetic** homotopy theory

* Proofs are constructive*: can run them

We can do **computer-checked proofs** in **synthetic** homotopy theory

- * Proofs are constructive*: can run them
- * Results apply in a variety of settings,
from simplicial sets (hence topological spaces)
to Quillen model categories and ∞ -topoi*

We can do **computer-checked proofs** in **synthetic** homotopy theory

- * Proofs are constructive*: can run them
- * Results apply in a variety of settings,
from simplicial sets (hence topological spaces)
to Quillen model categories and ∞ -topoi*
- * New type-theoretic proofs/methods

We can do **computer-checked proofs** in **synthetic** homotopy theory

- * Proofs are constructive*: can run them
- * Results apply in a variety of settings,
from simplicial sets (hence topological spaces)
to Quillen model categories and ∞ -topoi*
- * New type-theoretic proofs/methods

*work in progress

Homotopy in HoTT

$$\pi_1(S^1) = \mathbb{Z}$$

Freudenthal

Van Kampen

$$\pi_{k < n}(S^n) = 0$$

$$\pi_n(S^n) = \mathbb{Z}$$

Covering spaces

Hopf fibration

$K(G, n)$

**Whitehead
for n-types**

$$\pi_2(S^2) = \mathbb{Z}$$

Cohomology
axioms

$$\pi_3(S^2) = \mathbb{Z}$$

James

Blakers-Massey

Construction

$$\pi_4(S^3) = \mathbb{Z}?$$

**[Brunerie, Finster, Hou,
Licata, Lumsdaine, Shulman]**

Homotopy in HoTT


$$\pi_1(S^1) = \mathbb{Z}$$

$$\pi_{k < n}(S^n) = 0$$

Hopf fibration

$$\pi_2(S^2) = \mathbb{Z}$$

$$\pi_3(S^2) = \mathbb{Z}$$

James
Construction

$$\pi_4(S^3) = \mathbb{Z}?$$

Freudenthal

$$\pi_n(S^n) = \mathbb{Z}$$

K(G,n)

Cohomology
axioms

Blakers-Massey

Van Kampen

Covering spaces

**Whitehead
for n-types**

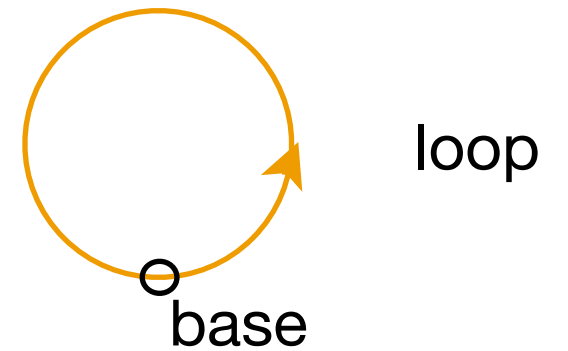
**[Brunerie, Finster, Hou,
Licata, Lumsdaine, Shulman]**

Example:

The Fundamental Group of the Circle

The Circle

Circle S^1 is *inductively generated* by

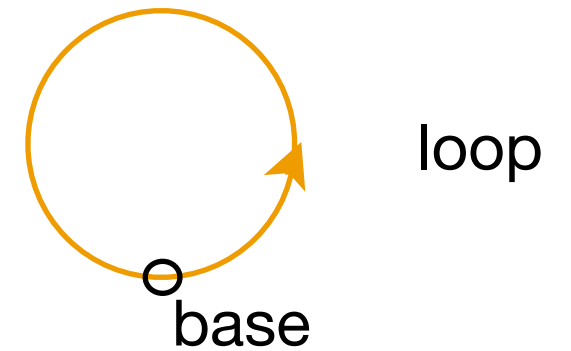


The Circle

Circle S^1 is *inductively generated* by

base : S^1

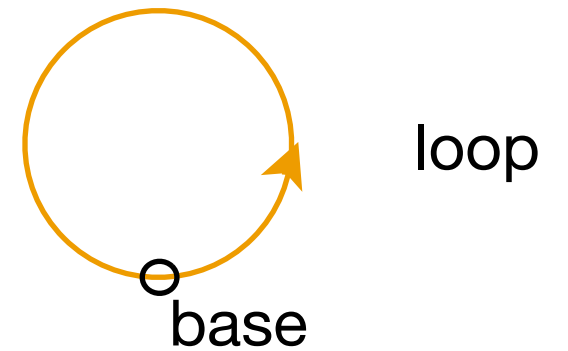
loop : base = base



The Circle

Circle S^1 is *inductively generated* by

point $\text{base} : S^1$
 $\text{loop} : \text{base} = \text{base}$

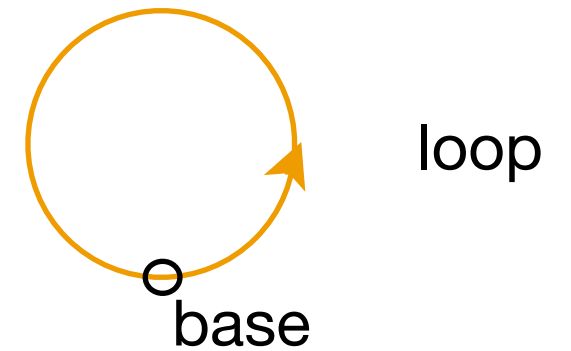


The Circle

Circle S^1 is *inductively generated* by

point $\text{base} : S^1$

path $\text{loop} : \text{base} = \text{base}$

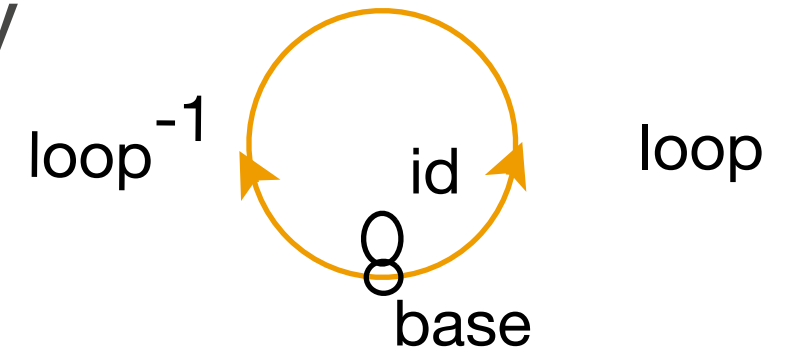


The Circle

Circle S^1 is *inductively generated* by

point $\text{base} : S^1$

path $\text{loop} : \text{base} = \text{base}$



Free type: equipped with structure by path induction

id $\text{inv} : \text{loop} \circ \text{loop}^{-1} = \text{id}$

loop^{-1} \dots

$\text{loop} \circ \text{loop}$

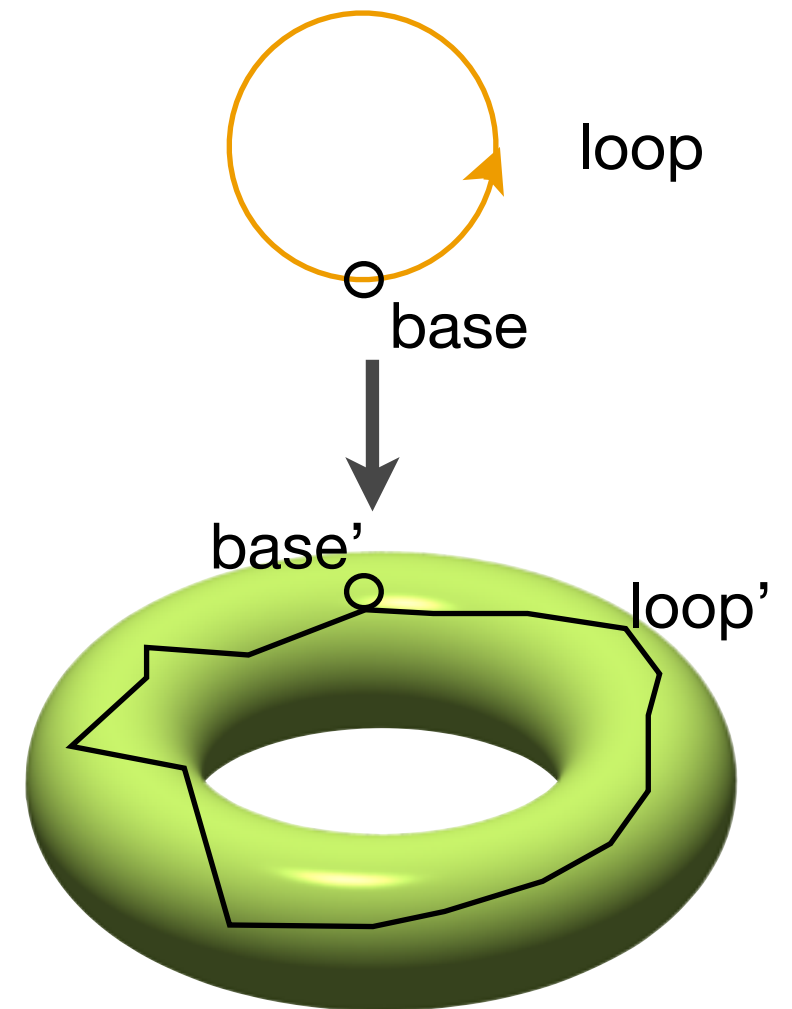
The Circle

Circle recursion:

function $S^1 \rightarrow X$ determined by

$\text{base}' : X$

$\text{loop}' : \text{base}' = \text{base}'$



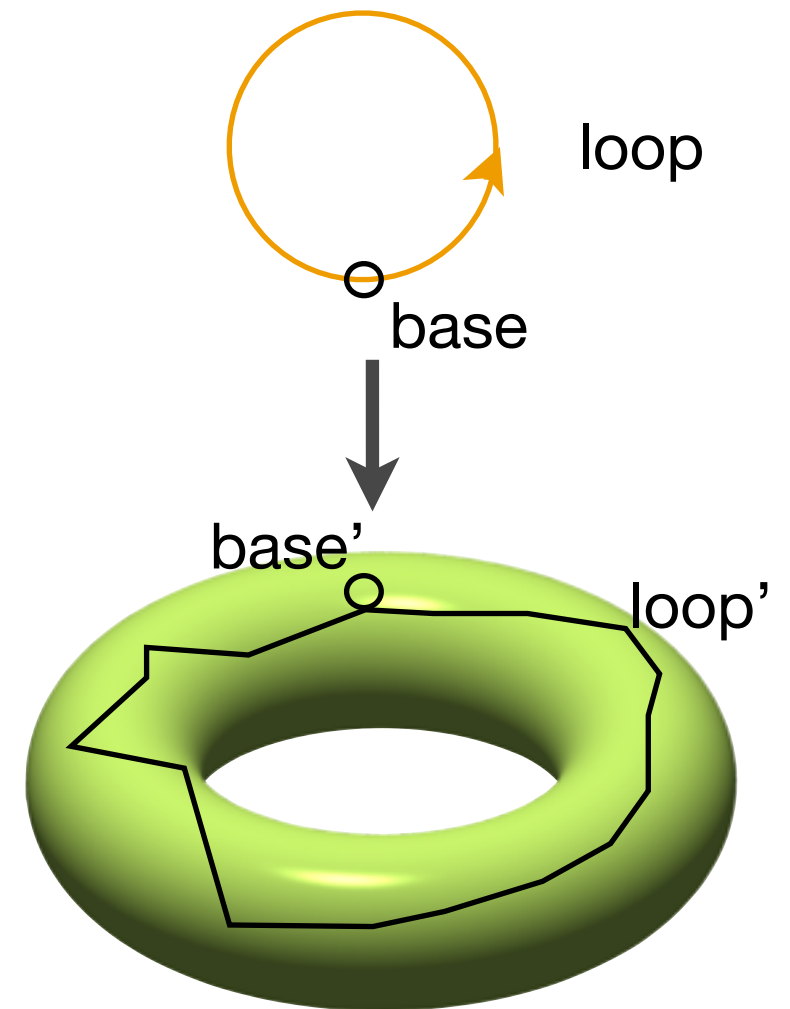
The Circle

Circle recursion:

function $S^1 \rightarrow X$ determined by

$\text{base}' : X$

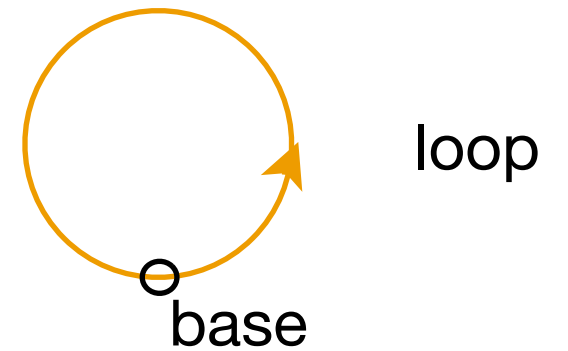
$\text{loop}' : \text{base}' = \text{base}'$



Circle induction: To prove a predicate P for all points on the circle, suffices to prove $P(\text{base})$, continuously in the loop

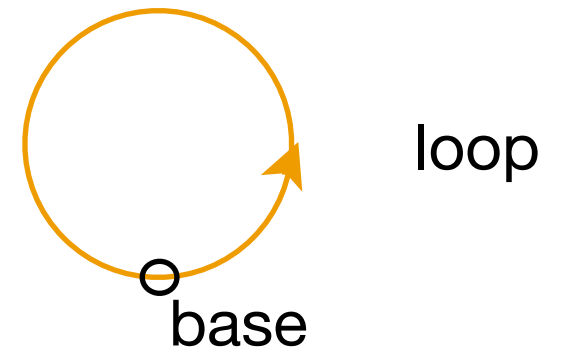
Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



Fundamental group of circle

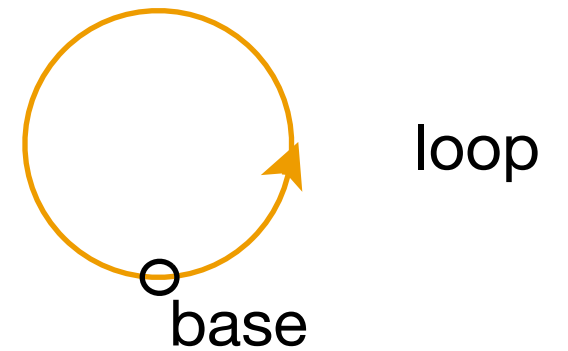
How many different loops are there on the circle, up to homotopy?



id

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?

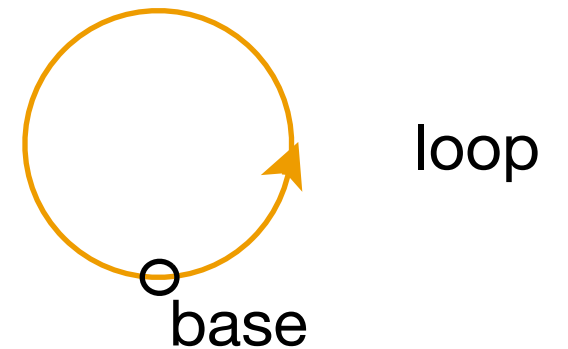


id

loop

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



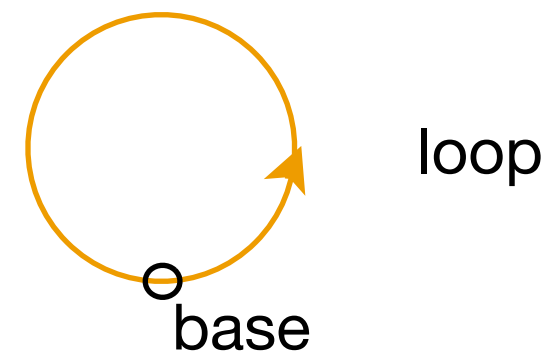
id

loop

loop^{-1}

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



id

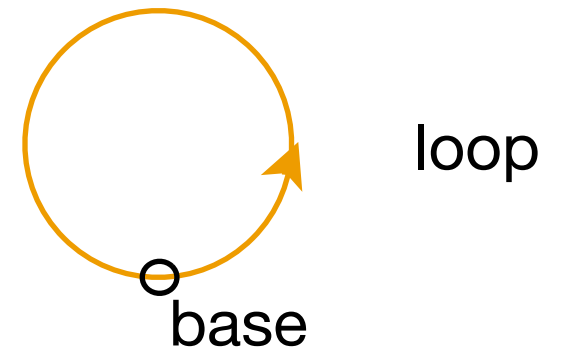
loop

loop^{-1}

$\text{loop} \circ \text{loop}$

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



id

loop

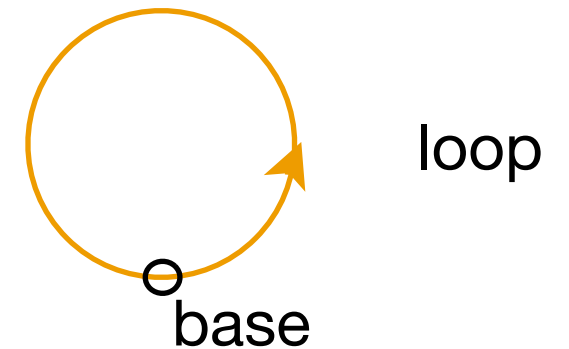
loop^{-1}

$\text{loop} \circ \text{loop}$

$\text{loop}^{-1} \circ \text{loop}^{-1}$

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



id

loop

loop^{-1}

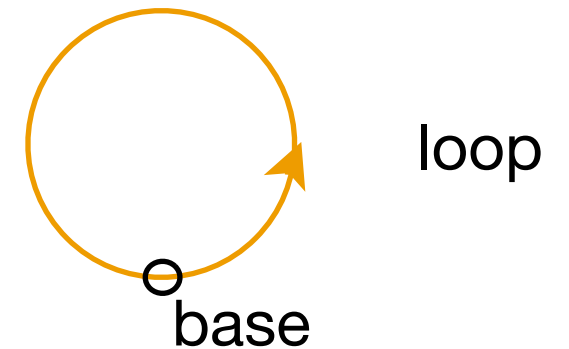
$\text{loop} \circ \text{loop}$

$\text{loop}^{-1} \circ \text{loop}^{-1}$

$\text{loop} \circ \text{loop}^{-1}$

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



id

loop

loop^{-1}

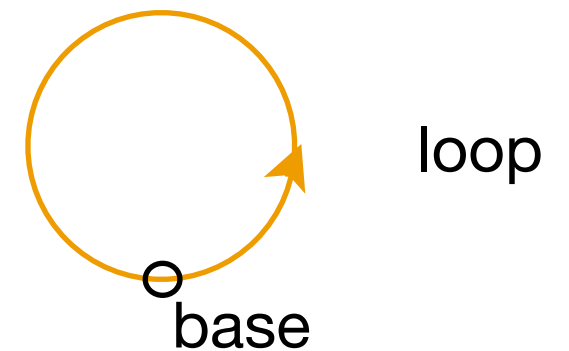
$\text{loop} \circ \text{loop}$

$\text{loop}^{-1} \circ \text{loop}^{-1}$

$\text{loop} \circ \text{loop}^{-1} = \text{id}$

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



id \emptyset

loop

loop^{-1}

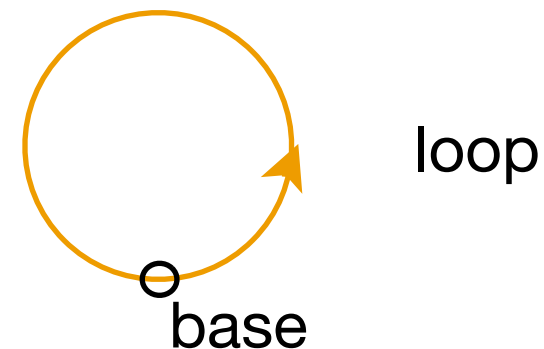
$\text{loop} \circ \text{loop}$

$\text{loop}^{-1} \circ \text{loop}^{-1}$

$\text{loop} \circ \text{loop}^{-1} = \text{id}$

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



id 0

loop 1

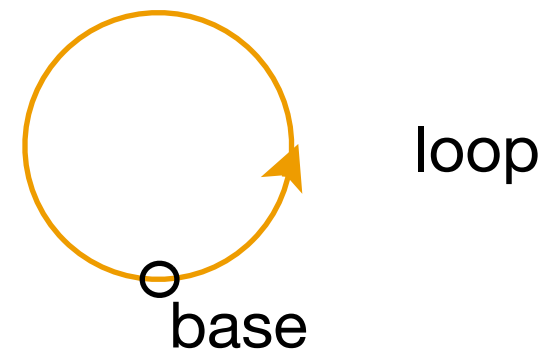
$$\text{loop}^{-1}$$

loop o loop

$$\text{loop}^{-1} \circ \text{loop}^{-1}$$
$$\text{loop} \circ \text{loop}^{-1} = \text{id}$$

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



id 0

loop 1

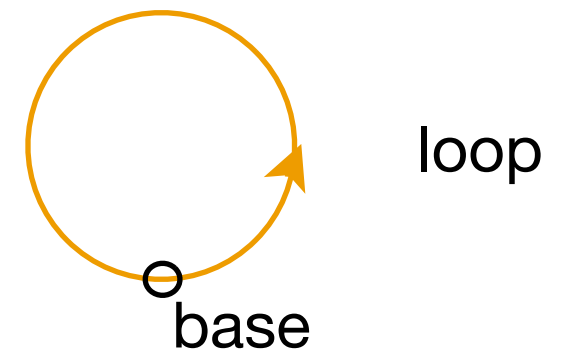
$$\text{loop}^{-1} \quad -1$$

loop o loop

$$\text{loop}^{-1} \circ \text{loop}^{-1}$$
$$\text{loop} \circ \text{loop}^{-1} = \text{id}$$

Fundamental group of circle

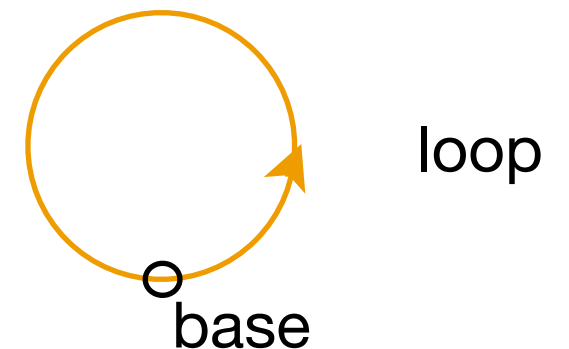
How many different loops are there on the circle, up to homotopy?



id	0
loop	1
loop ⁻¹	-1
loop o loop	2
loop ⁻¹ o loop ⁻¹	
loop o loop ⁻¹	= id

Fundamental group of circle

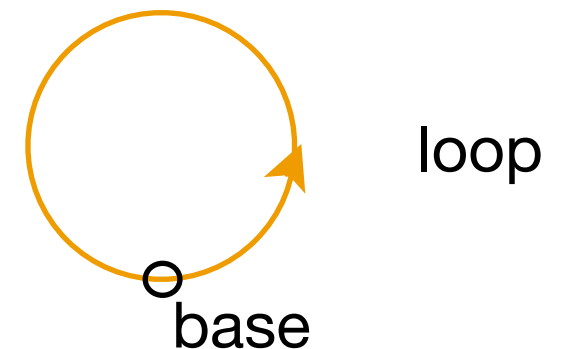
How many different loops are there on the circle, up to homotopy?



id	0
loop	1
loop ⁻¹	-1
loop o loop	2
loop ⁻¹ o loop ⁻¹	-2
loop o loop ⁻¹	= id

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



id	\emptyset
----	-------------

loop	1
------	---

loop^{-1}	-1
--------------------	----

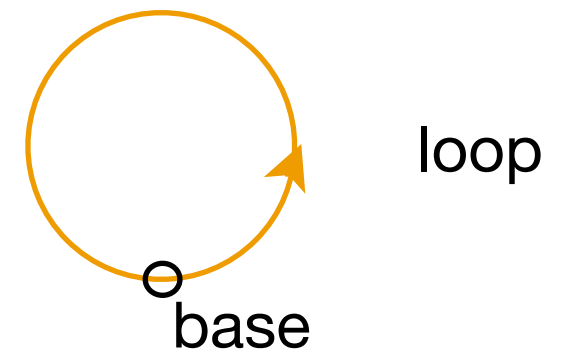
loop o loop	2
-------------	---

$\text{loop}^{-1} \text{ o } \text{loop}^{-1}$	-2
--	----

$\text{loop o } \text{loop}^{-1} = \text{id}$	\emptyset
---	-------------

Fundamental group of circle

How many different loops are there on the circle, up to homotopy?



id 0

loop 1

loop^{-1} -1

loop o loop 2

loop^{-1} o loop^{-1} -2

loop o loop^{-1} = id 0

**integers are “codes”
for paths on the
circle**

Fundamental group of circle

Definition. $\Omega(S^1)$ is the **type** of loops at base
i.e. the type (base = base)

Fundamental group of circle

Definition. $\Omega(S^1)$ is the **type** of loops at base
i.e. the type (base = base)

Theorem. $\Omega(S^1)$ is equivalent to \mathbb{Z} ,
by a map that sends 0 to +

Fundamental group of circle

Definition. $\Omega(S^1)$ is the **type** of loops at base
i.e. the type (base = base)

Theorem. $\Omega(S^1)$ is equivalent to \mathbb{Z} ,
by a map that sends 0 to +

Corollary: Fundamental group
of the circle is isomorphic to \mathbb{Z}

Fundamental group of circle

Definition. $\Omega(S^1)$ is the **type** of loops at base
i.e. the type (base = base)

Theorem. $\Omega(S^1)$ is equivalent to \mathbb{Z} ,
by a map that sends 0 to +

Corollary: Fundamental group
of the circle is isomorphic to \mathbb{Z}

**0-truncation (set of connected components)
of $\Omega(S^1)$**



Fundamental group of circle

Theorem. $\Omega(S^1)$ is equivalent to \mathbb{Z}

Proof: two mutually inverse functions

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

$$\text{loop}^- : \mathbb{Z} \rightarrow \Omega(S^1)$$

Fundamental group of circle

Theorem. $\Omega(S^1)$ is equivalent to \mathbb{Z}

Proof: two mutually inverse functions

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

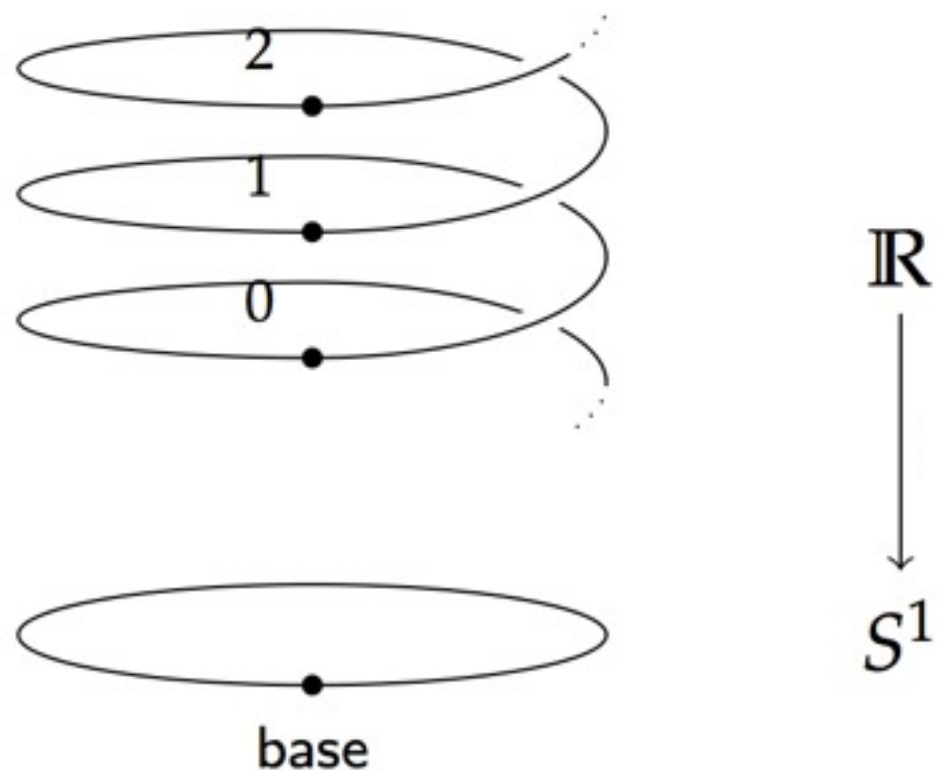
$$\text{loop}^- : \mathbb{Z} \rightarrow \Omega(S^1)$$

$$\text{loop}^0 = \text{id}$$

$$\text{loop}^{+n} = \text{loop} \circ \text{loop} \circ \dots \circ \text{loop} \quad (n \text{ times})$$

$$\text{loop}^{-n} = \text{loop}^{-1} \circ \text{loop}^{-1} \circ \dots \circ \text{loop}^{-1} \quad (n \text{ times})$$

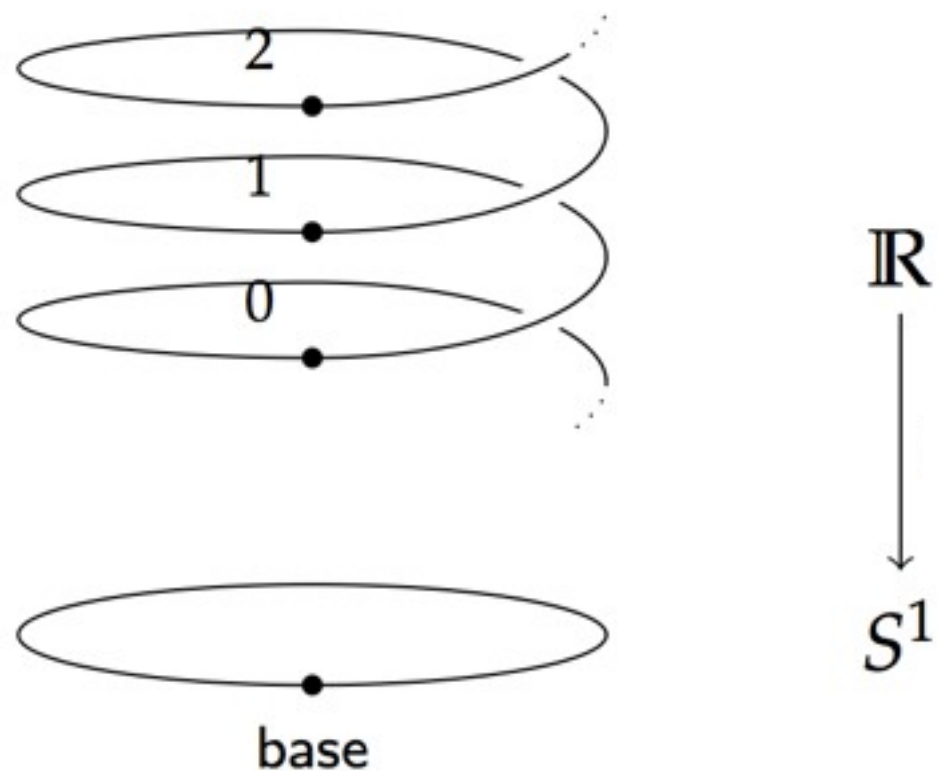
Universal Cover



$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

defined by **lifting** a loop to the cover, and giving the other endpoint of 0

Universal Cover

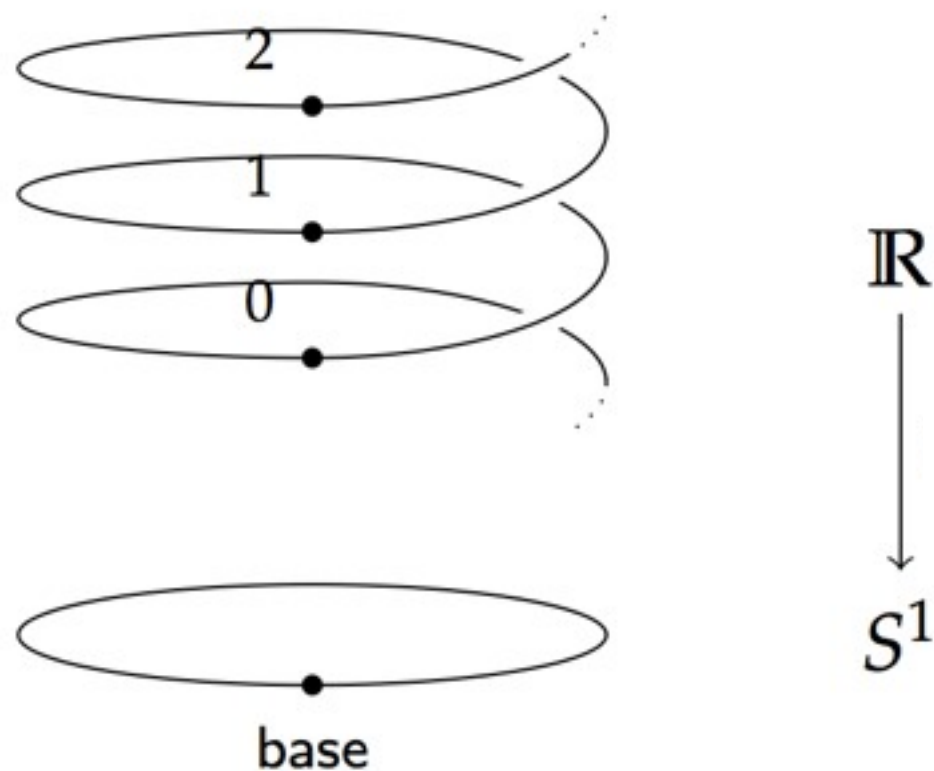


$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

defined by **lifting** a loop to the cover, and giving the other endpoint of 0

lifting is functorial

Universal Cover



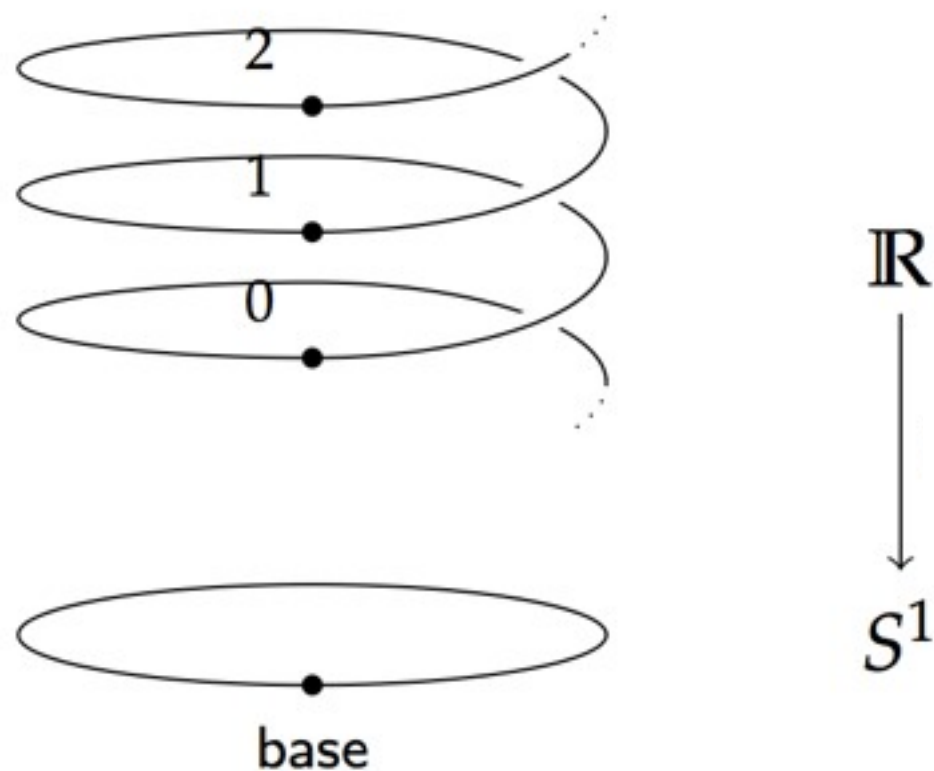
$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

defined by **lifting** a loop to the cover, and giving the other endpoint of 0

lifting is functorial

lifting loop adds 1

Universal Cover



$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

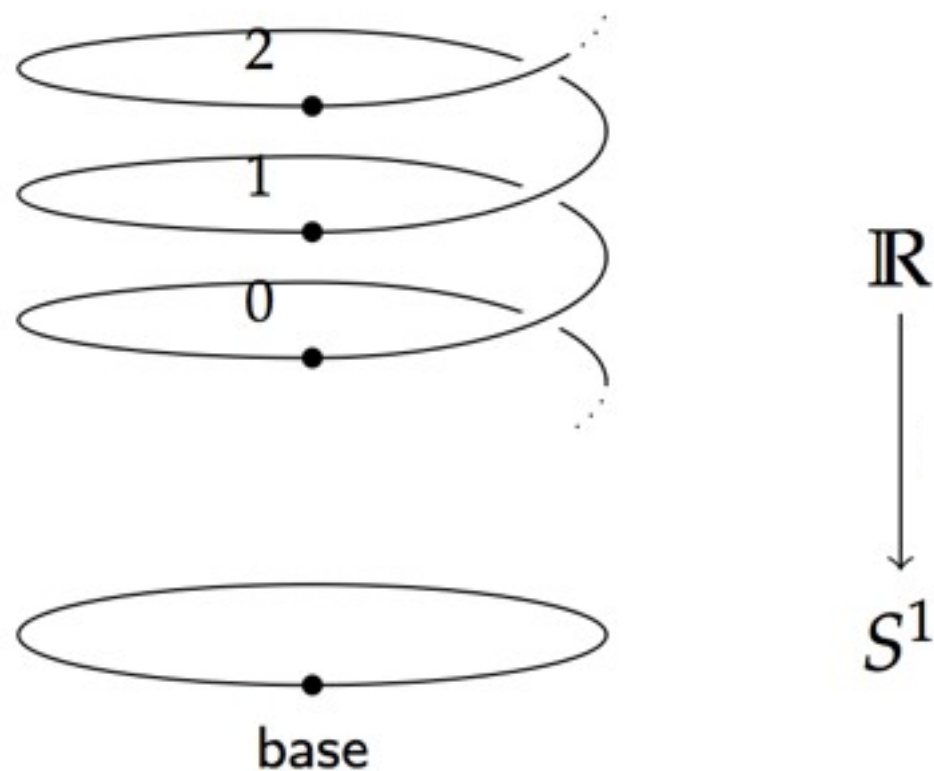
defined by **lifting** a loop
to the cover, and giving
the other endpoint of 0

lifting is functorial

lifting loop adds 1

lifting loop^{-1} subtracts 1

Universal Cover



$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

defined by **lifting** a loop to the cover, and giving the other endpoint of 0

Example:

$$\begin{aligned} & \text{wind}(\text{loop} \circ \text{loop}^{-1}) \\ &= 0 + 1 - 1 \\ &= 0 \end{aligned}$$

lifting is functorial

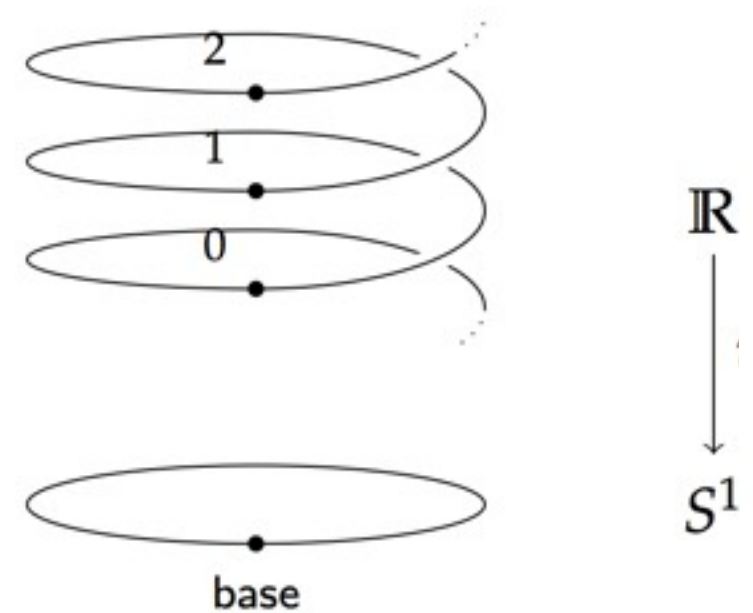
lifting loop adds 1

lifting loop^{-1} subtracts 1

Fibration = Family of types

Fibration (classically):

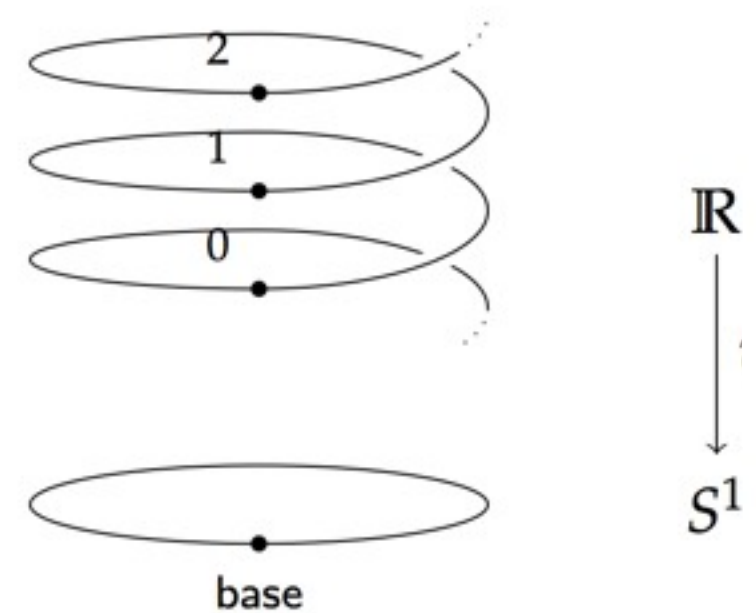
map $p: E \rightarrow B$ such that
any path from $p(e)$ to y
lifts to a path in E from e
to some point in $p^{-1}(y)$



Fibration = Family of types

Fibration (classically):

map $p: E \rightarrow B$ such that
any path from $p(e)$ to y
lifts to a path in E from e
to some point in $p^{-1}(y)$



Family of types $(E(x))_{x:B}$

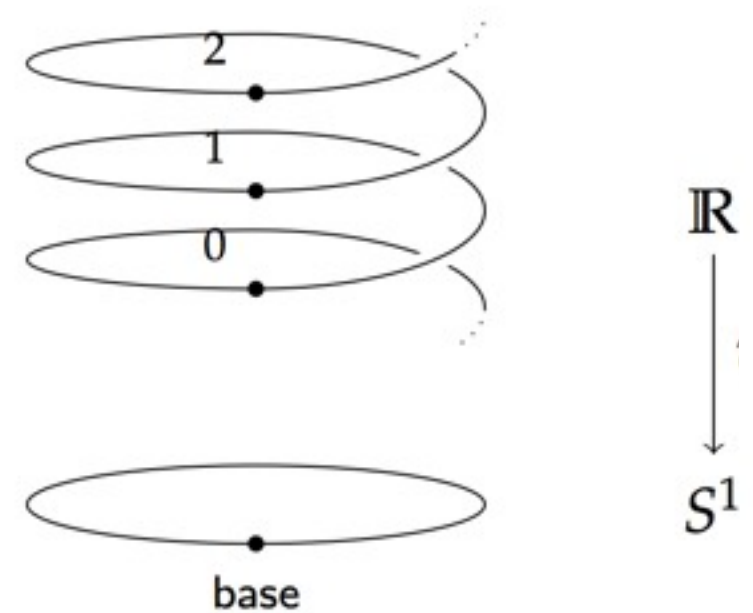
✱ *Fibers:* $E(b)$ is a type for all $b:B$

✱ *transport:* equivalence $E(b_1) \simeq E(b_2)$ for all $p:b_1 =_B b_2$

Fibration = Family of types

Fibration (classically):

map $p: E \rightarrow B$ such that
any path from $p(e)$ to y
lifts to a path in E from e
to some point in $p^{-1}(y)$



Family of types $(E(x))_{x:B}$

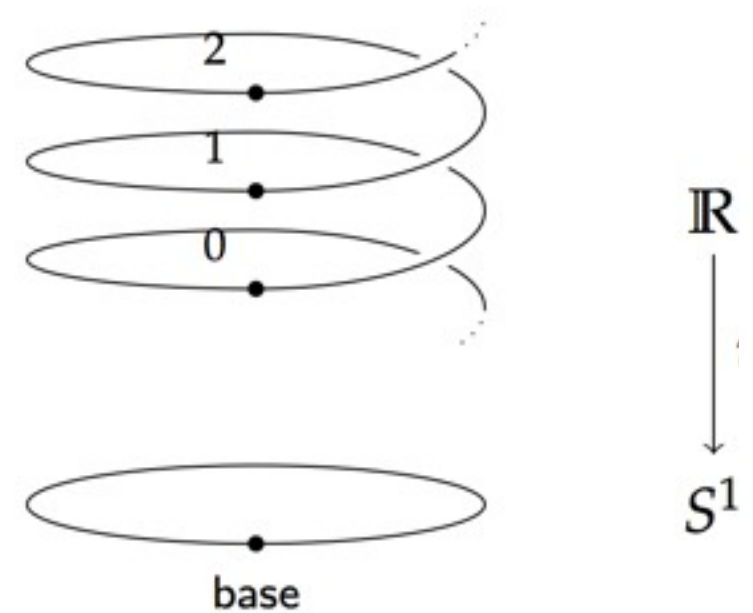
✱ *Fibers:* $E(b)$ is a type for all $b:B$

✱ *transport:* equivalence $E(b_1) \simeq E(b_2)$ for all $p:b_1=Bb_2$

Fibration = Family of types

Fibration (classically):

map $p: E \rightarrow B$ such that
any path from $p(e)$ to y
lifts to a path in E from e
to some point in $p^{-1}(y)$



Family of types $(E(x))_{x:B}$

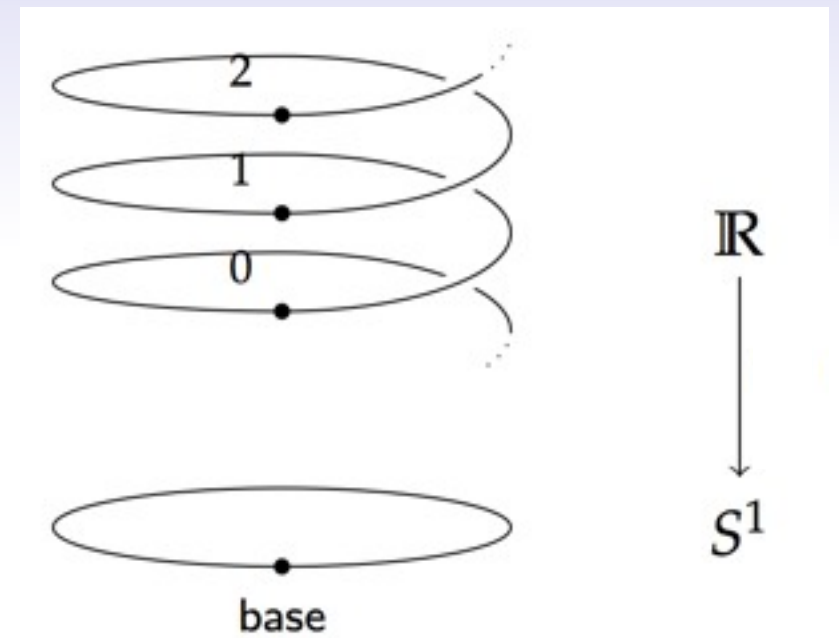
* *Fibers*: $E(b)$ is a type for all $b:B$

* *transport*: equivalence $E(b_1) \simeq E(b_2)$ for all $p: b_1 =_B b_2$

sends $e \in E(x)$ to other endpoint of lifting of p

Universal Cover

family of types $(\text{Cover}(x))_{x:S^1}$

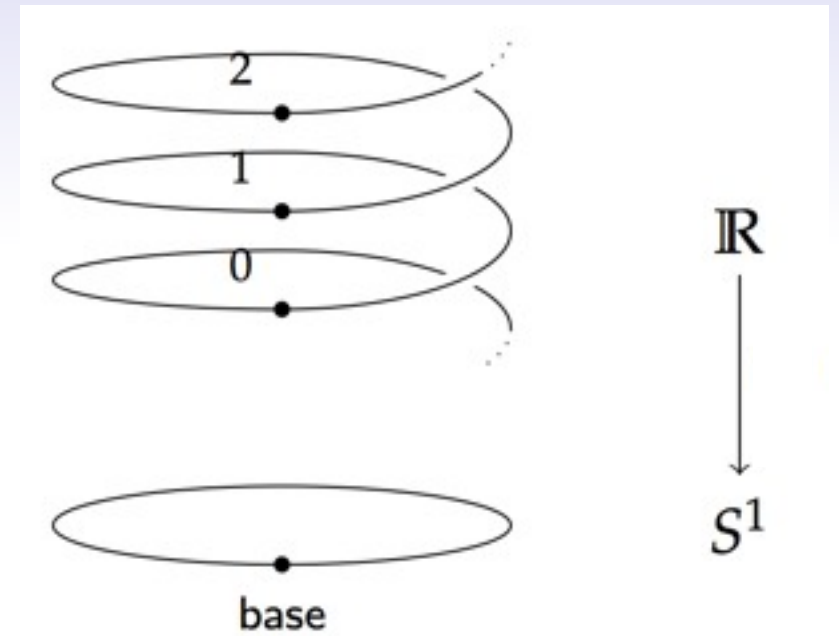


Universal Cover

family of types $(\text{Cover}(x))_{x:S^1}$

By circle recursion, it suffices to give

- * Fiber over base: the type \mathbb{Z}
- * Equivalence $\mathbb{Z} \rightarrow \mathbb{Z}$ as lifting of loop:
successor

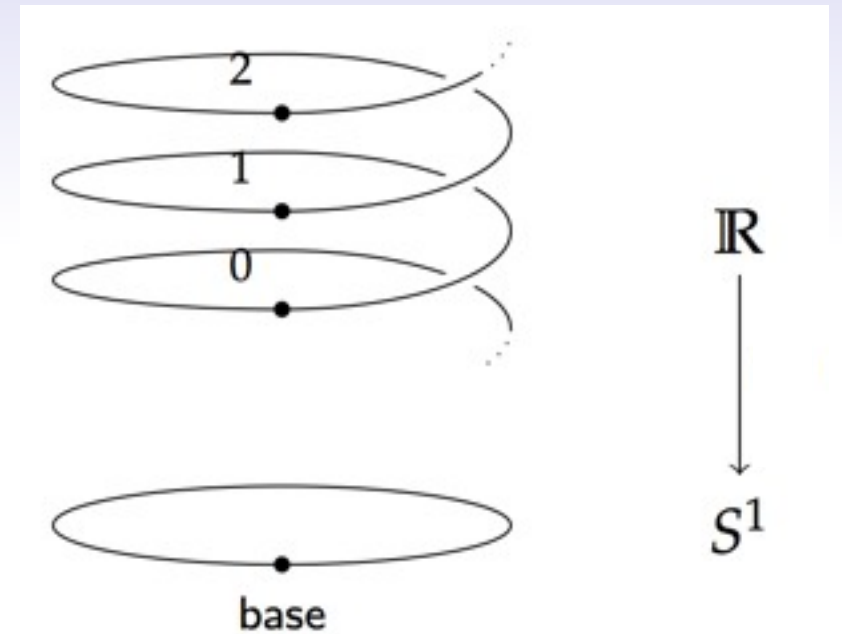


Universal Cover

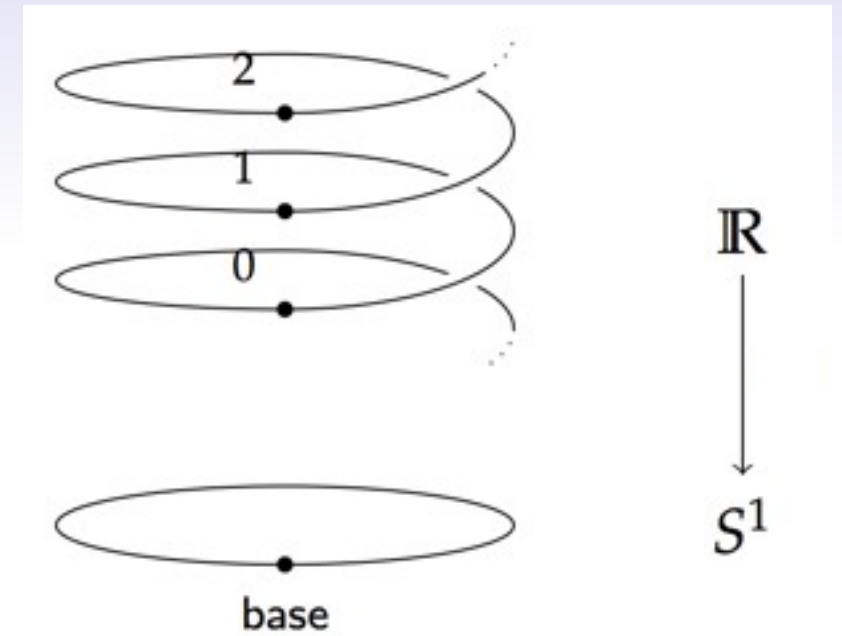
family of types $(\text{Cover}(x))_{x:S^1}$

By circle recursion, it suffices to give

- * Fiber over base: the type \mathbb{Z}
- * Equivalence $\mathbb{Z} \rightarrow \mathbb{Z}$ as lifting of loop: **uses *univalence***
successor



Universal Cover



family of types $(\text{Cover}(x))_{x:S^1}$

By circle recursion, it suffices to give

- * Fiber over base: the type \mathbb{Z}
- * Equivalence $\mathbb{Z} \simeq \mathbb{Z}$ as lifting of loop: **uses *univalence***
successor

Defining equations:

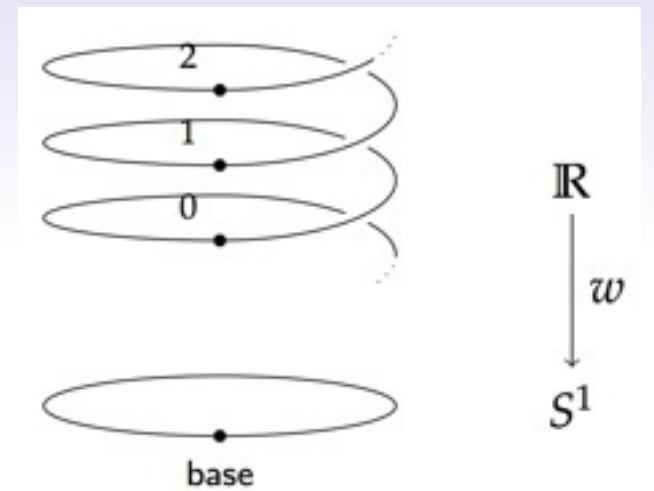
$\text{Cover}(\text{base}) := \mathbb{Z}$

$\text{transport}_{\text{Cover}}(\text{loop}) := \text{successor}$

Winding number

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

$$\text{wind}(p) = \text{transport}_{\text{cover}}(p, 0)$$



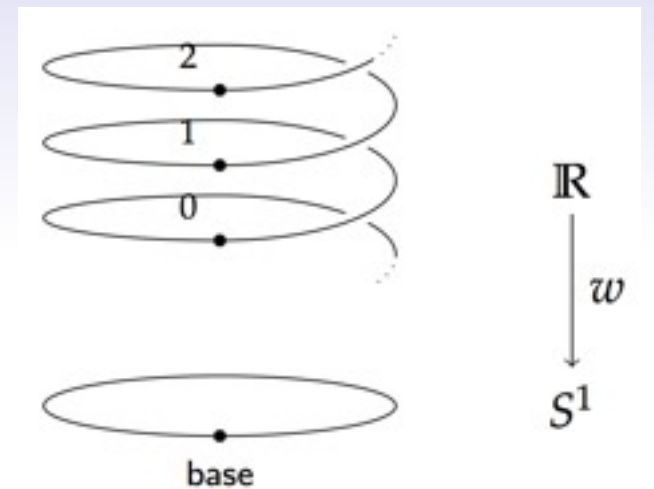
**lift p to cover,
starting at 0**

Winding number

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

$$\text{wind}(p) = \text{transport}_{\text{cover}}(p, 0)$$

$$\text{wind}(\text{loop}^{-1} \circ \text{loop})$$



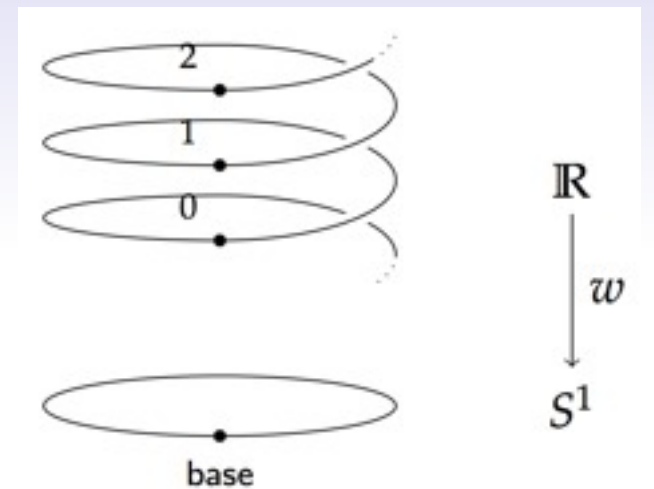
**lift p to cover,
starting at 0**

Winding number

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

$$\text{wind}(p) = \text{transport}_{\text{cover}}(p, 0)$$

$$\begin{aligned} & \text{wind}(\text{loop}^{-1} \circ \text{loop}) \\ &= \text{transport}_{\text{cover}}(\text{loop}^{-1} \circ \text{loop}, 0) \end{aligned}$$

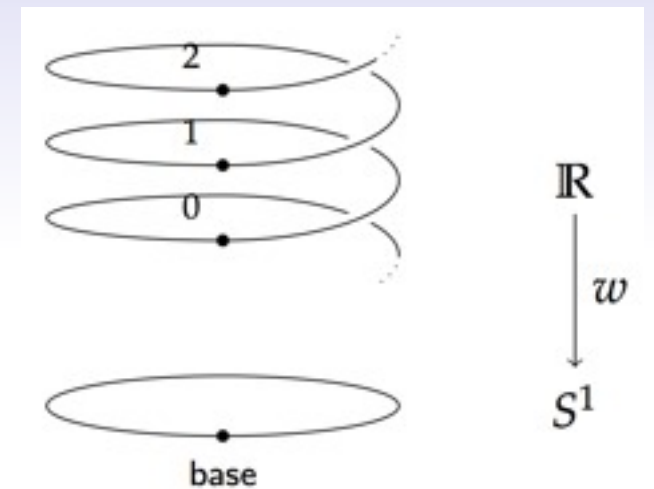


**lift p to cover,
starting at 0**

Winding number

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

$$\text{wind}(p) = \text{transport}_{\text{Cover}}(p, 0)$$



**lift p to cover,
starting at 0**

$$\text{wind}(\text{loop}^{-1} \circ \text{loop})$$

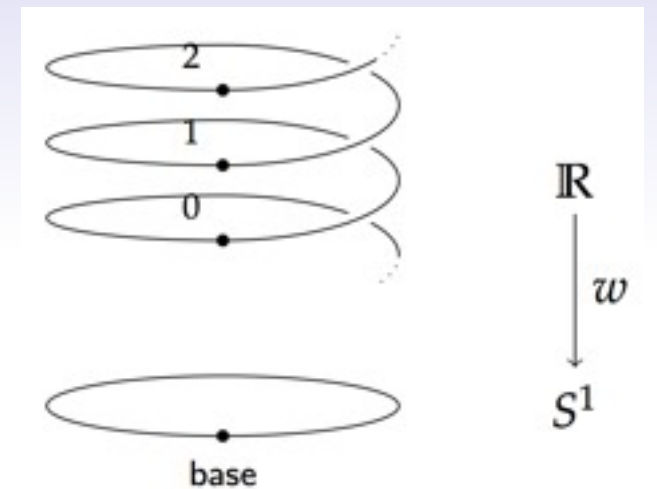
$$= \text{transport}_{\text{Cover}}(\text{loop}^{-1} \circ \text{loop}, 0)$$

$$= \text{transport}_{\text{Cover}}(\text{loop}^{-1}, \text{transport}_{\text{Cover}}(\text{loop}, 0))$$

Winding number

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

$$\text{wind}(p) = \text{transport}_{\text{Cover}}(p, 0)$$



**lift p to cover,
starting at 0**

$$\text{wind}(\text{loop}^{-1} \circ \text{loop})$$

$$= \text{transport}_{\text{Cover}}(\text{loop}^{-1} \circ \text{loop}, 0)$$

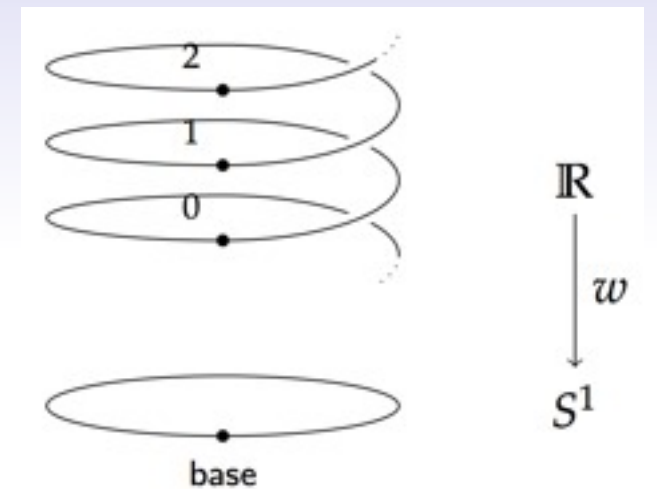
$$= \text{transport}_{\text{Cover}}(\text{loop}^{-1}, \text{transport}_{\text{Cover}}(\text{loop}, 0))$$

$$= \text{transport}_{\text{Cover}}(\text{loop}^{-1}, 1)$$

Winding number

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

$$\text{wind}(p) = \text{transport}_{\text{Cover}}(p, 0)$$



**lift p to cover,
starting at 0**

$$\text{wind}(\text{loop}^{-1} \circ \text{loop})$$

$$= \text{transport}_{\text{Cover}}(\text{loop}^{-1} \circ \text{loop}, 0)$$

$$= \text{transport}_{\text{Cover}}(\text{loop}^{-1}, \text{transport}_{\text{Cover}}(\text{loop}, 0))$$

$$= \text{transport}_{\text{Cover}}(\text{loop}^{-1}, 1)$$

$$= 0$$

So far

Theorem. $\Omega(S^1)$ is equivalent to \mathbb{Z}

Proof: two functions

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

$$\text{wind}(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$$

$$\text{loop}^- : \mathbb{Z} \rightarrow \Omega(S^1)$$

$$\text{loop}^0 = \text{id}$$

$$\text{loop}^{+n} = \text{loop} \circ \text{loop} \circ \dots \circ \text{loop} \quad (\text{n times})$$

$$\text{loop}^{-n} = \text{loop}^{-1} \circ \text{loop}^{-1} \circ \dots \circ \text{loop}^{-1} \quad (\text{n times})$$

So far

Theorem. $\Omega(S^1)$ is equivalent to \mathbb{Z}

Proof: two **mutually inverse** functions

$$\text{wind} : \Omega(S^1) \rightarrow \mathbb{Z}$$

$$\text{wind}(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$$

$$\text{loop}^- : \mathbb{Z} \rightarrow \Omega(S^1)$$

$$\text{loop}^0 = \text{id}$$

$$\text{loop}^{+n} = \text{loop} \circ \text{loop} \circ \dots \circ \text{loop} \quad (\text{n times})$$

$$\text{loop}^{-n} = \text{loop}^{-1} \circ \text{loop}^{-1} \circ \dots \circ \text{loop}^{-1} \quad (\text{n times})$$

Composite #1

Lemma. $\forall n. \text{wind}(\text{loop}^n) = n$

Proof: induction on n . E.g.

Composite #1

Lemma. $\forall n. \text{wind}(\text{loop}^n) = n$

Proof: induction on n . E.g.

$\text{wind}(\text{loop}^{n+1})$

Composite #1

Lemma. $\forall n. \text{wind}(\text{loop}^n) = n$

Proof: induction on n . E.g.

$$\begin{aligned} & \text{wind}(\text{loop}^{n+1}) \\ = & \text{wind}(\text{loop} \circ \text{loop}^n) \end{aligned} \quad [\text{def. loop}]$$

Composite #1

Lemma. $\forall n. \text{wind}(\text{loop}^n) = n$

Proof: induction on n . E.g.

$$\begin{aligned} & \text{wind}(\text{loop}^{n+1}) \\ = & \text{wind}(\text{loop} \circ \text{loop}^n) && [\text{def. loop}] \\ = & \text{transport}_{\text{Cover}}(\text{loop} \circ \text{loop}^n, 0) && [\text{def. wind}] \end{aligned}$$

Composite #1

Lemma. $\forall n. \text{wind}(\text{loop}^n) = n$

Proof: induction on n . E.g.

$$\begin{aligned} & \text{wind}(\text{loop}^{n+1}) \\ = & \text{wind}(\text{loop} \circ \text{loop}^n) && [\text{def. loop}] \\ = & \text{transport}_{\text{Cover}}(\text{loop} \circ \text{loop}^n, 0) && [\text{def. wind}] \\ = & \text{transport}_{\text{Cover}}(\text{loop}, \\ & \quad \text{transport}_{\text{Cover}}(\text{loop}^n, 0)) && [\text{functorial}] \end{aligned}$$

Composite #1

Lemma. $\forall n. \text{wind}(\text{loop}^n) = n$

Proof: induction on n . E.g.

$$\begin{aligned} & \text{wind}(\text{loop}^{n+1}) \\ = & \text{wind}(\text{loop} \circ \text{loop}^n) && [\text{def. loop}] \\ = & \text{transport}_{\text{Cover}}(\text{loop} \circ \text{loop}^n, 0) && [\text{def. wind}] \\ = & \text{transport}_{\text{Cover}}(\text{loop}, \\ & \quad \text{transport}_{\text{Cover}}(\text{loop}^n, 0)) && [\text{functorial}] \\ = & \text{transport}_{\text{Cover}}(\text{loop}, n) && [\text{IH}] \end{aligned}$$

Composite #1

Lemma. $\forall n. \text{wind}(\text{loop}^n) = n$

Proof: induction on n . E.g.

$$\begin{aligned} & \text{wind}(\text{loop}^{n+1}) \\ = & \text{wind}(\text{loop} \circ \text{loop}^n) && [\text{def. loop}] \\ = & \text{transport}_{\text{Cover}}(\text{loop} \circ \text{loop}^n, 0) && [\text{def. wind}] \\ = & \text{transport}_{\text{Cover}}(\text{loop}, && \\ & \quad \text{transport}_{\text{Cover}}(\text{loop}^n, 0)) && [\text{functorial}] \\ = & \text{transport}_{\text{Cover}}(\text{loop}, n) && [\text{IH}] \\ = & n+1 && [\text{def. Cover}] \end{aligned}$$

Composite #1

```

wind-loop^ : (n : Int) → Path (wind (loop^ n)) n
wind-loop^ Zero = id
wind-loop^ (Pos One) = ap≈ transport-Cover-loop
wind-loop^ (Pos (S n)) =
  transport Cover (loop · loop^ (Pos n)) Zero           =< ap≈ (transport-· Cover loop (loop^ (Pos n))) >
  transport Cover loop                                     =< ap (transport Cover loop) (wind-loop^ (Pos n)) >
    (transport Cover (loop^ (Pos n)) Zero)               =< ap≈ transport-Cover-loop >
  transport Cover loop (Pos n)
  succ (Pos n) ■
wind-loop^ (Neg One) = ap≈ transport-Cover-!loop
wind-loop^ (Neg (S n)) =
  transport Cover (! loop · loop^ (Neg n)) Zero           =< ap≈ (transport-· Cover (! loop) (loop^ (Neg n))) >
  transport Cover (! loop)
    (transport Cover (loop^ (Neg n)) Zero)               =< ap≈ transport-Cover-!loop >
  pred (transport Cover (loop^ (Neg n)) Zero)            =< ap pred (wind-loop^ (Neg n)) >
  pred (Neg n) ■

```

Composite #2

Lem. $\forall p: \text{base} = \text{base}. \text{loop}^{\text{wind}(p)} = p$

Composite #2

Lem. $\forall p: \text{base} = \text{base}. \text{loop}^{\text{wind}(p)} = p$

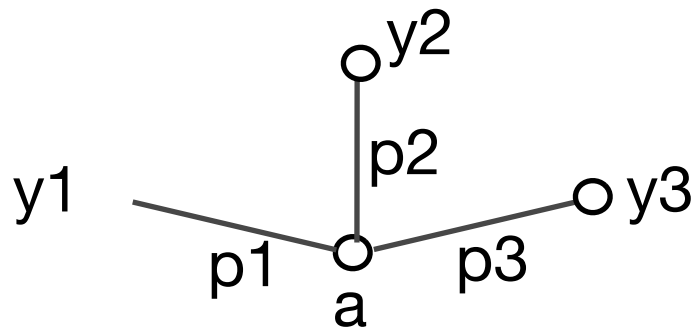
Proof: want to apply path induction
but path induction does not apply to loops

Composite #2

Lem. $\forall p: \text{base} = \text{base}. \text{loop}^{\text{wind}(p)} = p$

Proof: want to apply path induction
but path induction does not apply to loops

**Type of paths
from a to somewhere**



**is inductively
generated by**



Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{loop}^{\text{wind}(p)} = p$

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{loop}^{\text{wind}(p)} = p$

Proof:

$\text{wind} : \text{base}=\text{base} \rightarrow \mathbb{Z}$

$\text{wind}(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$

$\text{loop}^- : \mathbb{Z} \rightarrow \Omega(S^1)$

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{loop}^{\text{wind}(p)} = p$

Proof: need to generalize wind

$\text{wind} : \text{base}=\text{base} \rightarrow \mathbb{Z}$

$\text{wind}(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$

$\text{loop}^- : \mathbb{Z} \rightarrow \Omega(S^1)$

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{loop}^{\text{wind}(p)} = p$

Proof:

$\text{encode} : \forall y:S^1. \text{base}=y \rightarrow \text{Cover}(y)$

$\text{encode}(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$

$\text{loop}^- : \mathbb{Z} \rightarrow \Omega(S^1)$

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{loop}^{\text{wind}(p)} = p$

Proof: need to generalize loop^-

$\text{encode} : \forall y:S^1. \text{base}=y \rightarrow \text{Cover}(y)$

$\text{encode}(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$

$\text{loop}^- : \mathbb{Z} \rightarrow \Omega(S^1)$

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{loop}^{\text{wind}(p)} = p$

Proof: need to generalize wind and loop-

encode : $\forall y:S^1. \text{base}=y \rightarrow \text{Cover}(y)$

encode(p) = $\text{transport}_{\text{Cover}}(p, \emptyset)$

decode : $\forall y:S^1. \text{Cover}(y) \rightarrow \text{base}=y$

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{decode}_y(\text{encode}_y(p)) = p$

Proof: need to generalize wind and loop-

$\text{encode} : \forall y:S^1. \text{base}=y \rightarrow \text{Cover}(y)$

$\text{encode}(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$

$\text{decode} : \forall y:S^1. \text{Cover}(y) \rightarrow \text{base}=y$

Decode

decode : $\forall y:S^1. \text{Cover}(y) \rightarrow \text{base}=y$

Defined by circle induction:

decode(base) := loop⁻
decode(loop) := ...

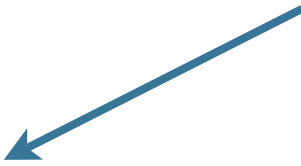
Decode

decode : $\forall y:S^1. \text{Cover}(y) \rightarrow \text{base}=y$

Defined by circle induction:

decode(base) := loop⁻
decode(loop) := ...

Cover(base) \rightarrow base=base



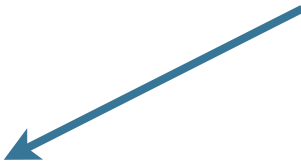
Decode

decode : $\forall y : S^1. \text{Cover}(y) \rightarrow \text{base}=y$

Defined by circle induction:

decode(base) := loop⁻
decode(loop) := ...

Cover(base) → base=base



“loop⁻ is invariant under going around the loop
in the fibration $\text{Cover}(y) \rightarrow \text{base}=y$ ”

Decode

```

decode : {x : S1} → Cover x → Path base x
decode {x} =
  S1-induction
  (λ x' → Cover x' → Path base x')
  loop^
  loop^-respects-loop
x where
  loop^-respects-loop : transport (λ x' → Cover x' → Path base x') loop loop^ = (λ n → loop^ n)
  loop^-respects-loop =
    (transport (λ x' → Cover x' → Path base x') loop loop^ ≈< transport→ Cover (Path base) loop loop^ >

      transport (λ x' → Path base x') loop
    o loop^
    o transport Cover (! loop)

      (λ p → loop · p)
    o loop^
    o transport Cover (! loop)

      (λ p → loop · p)
    o loop^
    o pred

    (λ n → loop · (loop^ (pred n)))

    (λ n → loop^ n)
  ■)

≈< λ≈ (λ y → transport-Path-right loop (loop^ (transport Cover (! loop) y))) >

≈< λ≈ (λ y → ap (λ x' → loop · loop^ x') (ap≈ transport-Cover-!loop)) >

≈< id >

≈< λ≈ (λ y → move-left-! _ loop (loop^ y) (loop^-preserves-pred y)) >

```

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{decode}_y(\text{encode}_y(p)) = p$

$\text{encode} : \forall y:S^1. \text{base}=y \rightarrow \text{Cover}(y)$

$\text{encode}_y(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$

$\text{decode} : \forall y:S^1. \text{Cover}(y) \rightarrow \text{base}=y$

$\text{decode}_{\text{base}}(n) = \text{loop}^n$

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{decode}_y(\text{encode}_y(p)) = p$

Proof: By **path induction**, suffices to show

$$\text{decode}_{\text{base}}(\text{encode}_{\text{base}}(\text{id}))$$

$$= \text{id}$$

$\text{encode} : \forall y:S^1. \text{base}=y \rightarrow \text{Cover}(y)$

$\text{encode}_y(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$

$\text{decode} : \forall y:S^1. \text{Cover}(y) \rightarrow \text{base}=y$

$\text{decode}_{\text{base}}(n) = \text{loop}^n$

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{decode}_y(\text{encode}_y(p)) = p$

Proof: By **path induction**, suffices to show

$$\begin{aligned} & \text{decode}_{\text{base}}(\text{encode}_{\text{base}}(\text{id})) \\ &= \text{decode}_{\text{base}}(0) \\ &= \text{id} \end{aligned}$$

$\text{encode} : \forall y:S^1. \text{base}=y \rightarrow \text{Cover}(y)$

$\text{encode}_y(p) = \text{transport}_{\text{Cover}}(p, 0)$

$\text{decode} : \forall y:S^1. \text{Cover}(y) \rightarrow \text{base}=y$

$\text{decode}_{\text{base}}(n) = \text{loop}^n$

Composite #2

Lem. $\forall y:S^1, p:\text{base}=y. \text{decode}_y(\text{encode}_y(p)) = p$

Proof: By **path induction**, suffices to show

$$\begin{aligned} & \text{decode}_{\text{base}}(\text{encode}_{\text{base}}(\text{id})) \\ &= \text{decode}_{\text{base}}(\emptyset) \\ &= \text{loop}^{\emptyset} \\ &= \text{id} \end{aligned}$$

$\text{encode} : \forall y:S^1. \text{base}=y \rightarrow \text{Cover}(y)$

$\text{encode}_y(p) = \text{transport}_{\text{Cover}}(p, \emptyset)$

$\text{decode} : \forall y:S^1. \text{Cover}(y) \rightarrow \text{base}=y$

$\text{decode}_{\text{base}}(n) = \text{loop}^n$

Composite #2

`decode-encode` : $\{x : S^1\} (\alpha : \text{Path base } x)$
 $\rightarrow \text{Path } (\text{decode } (\text{encode } \alpha)) \alpha$
`decode-encode id = id`

Fundamental group of the circle

The book

Computer-checked

7.2.1.1 Encode/decode proof

By definition, $\Omega(S^1)$ is base \Rightarrow base. If we attempt to prove that $\Omega(S^1) = \mathbb{Z}$ by directly constructing an equivalence, we will get stuck, because type theory gives you little leverage for working with loops. Instead, we generalize the theorem statement to the path fibration, and analyze the whole fibration.

$$P(x : S^1) := \{\text{base} \Rightarrow x\}$$

with one end-point free.

We show that $P(x)$ is equal to another fibration, which gives a more explicit description of the paths—we call this other fibration “codes”, because its elements are data that act as codes for paths on the circle. In this case, the codes fibration is the universal cover of the circle.

Definition 7.2.1 (Universal Cover of S^1). Define $\text{code}(x : S^1) : \Omega$ by circle-recursion, with

$$\begin{aligned}\text{code}(\text{base}) &:= \mathbb{Z} \\ \text{code}(\text{loop}) &:= \text{us}(\text{succ})\end{aligned}$$

where succ is the equivalence $\mathbb{Z} \simeq \mathbb{Z}$ given by adding one, which by univalence determines a path from \mathbb{Z} to \mathbb{Z} in Ω .

To define a function by circle recursion, we need to find a point and a loop in the target. In this case, the target is Ω , and the point we choose is \mathbb{Z} , corresponding to our expectation that the fiber of the universal cover should be the integers. The loop we choose is the successor/predecessor isomorphism on \mathbb{Z} , which corresponds to the fact that going around the loop in the base goes up one level on the helix. Univalence is necessary for this part of the proof, because we need a non-trivial equivalence on \mathbb{Z} .

From this definition, it is simple to calculate that transporting with code takes loop to the successor function, and loop^{-1} to the predecessor function:

Lemma 7.2.2. $\text{transport}^{\text{code}}(\text{loop}, x) = x + 1$ and $\text{transport}^{\text{code}}(\text{loop}^{-1}, x) = x - 1$

Proof. For the first, we calculate as follows:

$$\begin{aligned}\text{transport}^{\text{code}}(\text{loop}, x) &= \text{transport}^{\text{code}}(\text{code}(\text{loop}), x) && \text{associativity} \\ &= \text{transport}^{\text{code}}(\text{us}(\text{succ}), x) && \text{reduction for circle-recursion} \\ &= x + 1 && \text{reduction for us}\end{aligned}$$

The second follows from the first, because $\text{transport}^p p$ and $\text{transport}^p p^{-1}$ are always inverses, so $\text{transport}^{\text{code}} \text{loop}^{-1} = \text{must be the inverse of the } \rightarrow + 1$. \square

In the remainder of the proof, we will show that P and code are equivalent.

[DRAFT OF MARCH 19, 2013]

7.2.1.1.1 Encoding Next, we define a function encode that maps paths to codes:

Definition 7.2.3. Define $\text{encode} : \prod (x : S^1) \rightarrow P(x)$ by

$$\text{encode } p := \text{transport}^{\text{code}}(p, 0)$$

(we leave the argument x implicit).

encode is defined by lifting a path into the universal cover, which determines an equivalence, and then applying the resulting equivalence to 0. The interesting thing about this function is that it computes a concrete number from a loop on the circle, when this loop is represented using the abstract groupoidal framework of HoTT. To gain an intuition for how it does this, observe that by the above lemmas, $\text{transport}^{\text{code}}(\text{loop}, x)$ is $x + 1$ and $\text{transport}^{\text{code}} \text{loop}^{-1} x$ is $x - 1$. Further, transport is functorial (chapter 2), so $\text{transport}^{\text{code}} \text{loop} \circ \text{loop}$ is $(\text{transport}^{\text{code}} \text{loop}) \circ (\text{transport}^{\text{code}} \text{loop})$, etc. Thus, when p is a composition like

$$\text{loop} \circ \text{loop}^{-1} \circ \text{loop} \circ \dots$$

$\text{transport}^{\text{code}} p$ will compute a composition of functions like

$$\{ \rightarrow + 1 \} \circ \{ \rightarrow - 1 \} \circ \{ \rightarrow + 1 \} \circ \dots$$

Applying this composition of functions to 0 will compute the winding number of the path—how many times it goes around the circle, with orientation marked by whether it is positive or negative, after inverses have been canceled. Thus, the computational behavior of encode follows from the reduction rules for higher-inductive types and univalence, and the action of transport on compositions and inverses.

Note that the instance $\text{encode}' := \text{encode}_{\text{base}}$ has type $\text{base} = \text{base} \rightarrow \mathbb{Z}$, which will be one half of the equivalence between $\text{base} = \text{base}$ and \mathbb{Z} .

7.2.1.1.2 Decoding Decoding an integer as a path is defined by recursion:

Definition 7.2.4. Define $\text{loop}'' : \mathbb{Z} \rightarrow \text{base} = \text{base}$ by

$$\text{loop}'' = \begin{cases} \text{loop} \circ \text{loop} \circ \dots \circ \text{loop} \text{ (n times)} & \text{for positive } n \\ \text{loop}^{-1} \circ \text{loop}^{-1} \circ \dots \circ \text{loop}^{-1} \text{ (n times)} & \text{for negative } n \\ \text{refl} & \text{for } 0 \end{cases}$$

Since what we want overall is an equivalence between $\text{base} = \text{base}$ and \mathbb{Z} , we might expect to be able to prove that encode' and loop'' give an equivalence. The problem comes in trying to prove the “decode after encode” direction, where we would need to show that $\text{loop}^{\text{code}}(p) = p$ for all p . We would like to apply path induction, but path induction

[DRAFT OF MARCH 19, 2013]

does not apply to loops like a with both endpoints fixed! The way to solve this problem is to generalize the theorem to show that $\text{loop}^{\text{code}, p} = p$ for all $x : S^1$ and $p : \text{base} = x$. However, this does not make sense as is, because loop'' is defined only for $\text{base} = \text{base}$, whereas here it is applied to a $\text{base} = x$. Thus, we generalize loop as follows:

Definition 7.2.5. Define $\text{decode} : \prod (x : S^1) \prod [\text{code}(x) \rightarrow P(x)]$, by circle induction on x . It suffices to give a function $\text{code}(\text{base}) \rightarrow P(\text{base})$, for which we use loop'' , and to show that loop'' respects the loop.

Proof. To show that loop'' respects the loop, it suffices to give a path from loop'' to itself that lies over loop . Formally, this means a path from $\text{transport}^{(\text{code} \circ \text{loop})}(\text{loop}, \text{loop})$ to loop'' . We define such a path as follows:

$$\begin{aligned}\text{transport}^{(\text{code} \circ \text{loop})}(\text{loop}, \text{loop}) &= \text{transport}^{\text{loop} \circ \text{loop}''} \circ \text{transport}^{\text{code}} \text{loop}^{-1} \\ &= \{ \rightarrow + \text{loop} \} \circ \{ \text{loop}'' \} \circ \text{transport}^{\text{code}} \text{loop}^{-1} \\ &= \{ \rightarrow + \text{loop} \} \circ \{ \text{loop}'' \} \circ \{ \rightarrow - 1 \} \\ &= \{ n \mapsto \text{loop}^{n-1} \circ \text{loop} \}\end{aligned}$$

From line 1 to line 2, we apply the definition of transport when the outer connective of the fibration is \rightarrow , which reduces the transport to pre- and post-composition with transport at the domain and range types. From line 2 to line 3, we apply the definition of transport when the type family is $\text{base} = x$, which is post-composition of paths. From line 3 to line 4, we use the action of code on loop^{-1} defined in Lemma 7.2.2. From line 4 to line 5, we simply reduce the function composition. Thus, it suffices to show that for all n , $\text{loop}^{n-1} \circ \text{loop} = \text{loop}''$, which is an easy induction, using the groupoid laws. \square

7.2.1.1.3 Decoding after encoding

Lemma 7.2.6. For all p for all $x : S^1$ and $p : \text{base} = x$, $\text{decode}_x(\text{encode}_x(p)) = p$.

Proof. By path induction, it suffices to show that $\text{decode}_{\text{base}}(\text{encode}_{\text{base}}(\text{refl}_{\text{base}})) = \text{refl}_{\text{base}}$. But $\text{encode}_{\text{base}}(\text{refl}_{\text{base}}) = \text{transport}^{\text{code}}(\text{refl}_{\text{base}}, 0) = 0$, and $\text{decode}_{\text{base}}(0) = \text{loop}'' = \text{refl}_{\text{base}}$. \square

7.2.1.1.4 Encoding after decoding

Lemma 7.2.7. For all p for all $x : S^1$ and $c : \text{code}(x)$, $\text{encode}_x(\text{decode}_x(c)) = c$.

Proof. The proof is by circle induction. It suffices to show the case for base, because the case for loop is a path between paths in \mathbb{Z} , which can be given by appealing to the fact that \mathbb{Z} is a set.

Thus, it suffices to show, for all $n : \mathbb{Z}$, that

$$\text{encode}'(\text{loop}'' n) = n$$

The proof is by induction, with cases for 0, -1 , $x + 1$, and $x - 1$.

- In the case for 0, the result is true by definition.
- In the case for 1, $\text{encode}'(\text{loop}'')$ reduces to $\text{transport}^{\text{code}}(\text{loop}, 0)$, which by Lemma 7.2.2 is $0 + 1 = 1$.
- In the case for $n + 1$,

$$\begin{aligned}\text{encode}'(\text{loop}^{n+1}) &= \text{encode}'(\text{loop}^n \circ \text{loop}) \\ &= \text{transport}^{\text{code}}(\text{loop}^n \circ \text{loop}, 0) \\ &= \text{transport}^{\text{code}}(\text{loop}, \text{transport}^{\text{code}}(\text{loop}^n, 0)) && \text{by functoriality} \\ &= (\text{transport}^{\text{code}}(\text{loop}^n, 0)) + 1 && \text{by Lemma 7.2.2} \\ &= n + 1 && \text{by the IH}\end{aligned}$$

- The cases for negatives are analogous. \square

7.2.1.1.5 Tying it all together

Theorem 7.2.8. There is a family of equivalences $\prod (x : S^1) \prod [P(x) \simeq \text{code}(x)]$.

Proof. The maps encode and decode are mutually inverse by Lemmas 7.2.6 and 7.2.7, and this can be improved to an equivalence. \square

Instantiating at base gives

Corollary 7.2.9. $(\text{base} = \text{base}) \simeq \mathbb{Z}$

A simple induction shows that this equivalence takes addition to composition, so $\Omega(S^1) = \mathbb{Z}$ as groups.

Corollary 7.2.10. $\pi_k(S^1) = \mathbb{Z}$ if $k = 1$ and 0 otherwise.

Proof. For $k = 1$, we sketched the proof from Corollary 7.2.9 above. For $k > 1$, $\|\Omega^{k+1}(S^1)\|_0 = \|\Omega^k(\Omega(S^1))\|_0 = \|\Omega^k(\mathbb{Z})\|_0$, which is 1 because \mathbb{Z} is a set and π_n of a set is trivial (PIDM lemmas to cite!). \square

[DRAFT OF MARCH 19, 2013]

Cover $x = S^1 \rightarrow \text{rec } \text{Int} \text{ (us succEquiv) } x$

```
transport-Cover-loop : Path (transport Cover loop) succ
transport-Cover-loop =
  transport Cover loop
  <= transport-ap-assoc Cover loop >
  transport (λ x → x) (ap Cover loop)
  <= ap (transport (λ x → x)) (ap (λ Cover loop)
    (gloop/rec Int (us succEquiv))) >
  transport (λ x → x) (us succEquiv)
  <= typeβ _ >
  succ *
```

```
transport-Cover-ll-loop : Path (transport Cover (l loop)) pred
transport-Cover-ll-loop =
  transport Cover (l loop)
  <= transport-ap-assoc Cover (l loop) >
  transport (λ x → x) (ap Cover (l loop))
  <= ap (transport (λ x → x)) (ap (l Cover loop)) >
  transport (λ x → x) (l (ap Cover loop))
  <= ap (λ y → transport (λ x → x) (l y))
    (gloop/rec Int (us succEquiv)) >
  transport (λ x → x) (l (us succEquiv))
  <= ap (transport (λ x → x)) (l (us succEquiv)) >
  transport (λ x → x) (us (l equiv succEquiv))
  <= typeβ _ >
  pred *
```

```
encode : (x : S1) → Path base x → Cover x
encode a = transport Cover a Zero
```

```
encode' : Path base base → Int
encode' a = encode (base) a
```

```
loopA : Int → Path base base
loopA Zero = id
loopA (Pos One) = loop
loopA (Pos (S n)) = loop · loopA (Pos n)
loopA (Neg One) = l loop
loopA (Neg (S n)) = l loop · loopA (Neg n)
```

```
loopA-preserves-pred
: (n : Int) → Path (loopA (pred n)) (l loop · loopA n)
loopA-preserves-pred (Pos One) = l (l-inv-1 loop)
loopA-preserves-pred (Pos (S y)) =
  l (l-assoc (l loop) loop (loopA (Pos y)))
  · l (ap (λ x → x · loopA (Pos y)) (l-inv-1 loop))
  · l (l-unit-1 (loopA (Pos y)))
loopA-preserves-pred Zero = id
loopA-preserves-pred (Neg One) = id
loopA-preserves-pred (Neg (S y)) = id
```

```
decode : (x : S1) → Cover x → Path base x
```

```
decode (x) =
```

```
 $S^1$ -induction
```

```
(λ x' → Cover x' → Path base x')
```

```
loopA
```

```
 $x$  where
```

```
abstract -- prevent Agda from normalizing
```

```
loopA-respects-loop : transport (λ x' → Cover x' → Path base x') loop loopA = (λ n → loopA n)
loopA-respects-loop =
  (transport (λ x' → Cover x' → Path base x') loop loopA
  <= transport-→ Cover (Path base) loop loopA >
  transport (λ x' → Path base x') loop
  <= loopA
  <= transport Cover (l loop)
  <= λ y → transport-Path-right loop (loopA (transport Cover (l loop) y))) >
  (λ p → loop · p)
  <= loopA
  <= transport Cover (l loop)
  <= λ y → ap (λ x' → loop · loopA x') (ap= transport-Cover-ll-loop)) >
  (λ p → loop · p)
  <= loopA
  <= pred
  <= id >
  (λ n → loop · (loopA (pred n)))
  <= λ y → move-left-l _ loop (loopA y) (loopA-preserves-pred y)) >
  (λ n → loopA n)
  >
```

```
abstract -- prevent Agda from normalizing
```

```
encode-loopA : (n : Int) → Path (encode (loopA n)) n
encode-loopA Zero = id
encode-loopA (Pos One) = ap= transport-Cover-loop
encode-loopA (Pos (S n)) =
  encode (loopA (Pos (S n)))
  <= id >
  transport Cover (loop · loopA (Pos n)) Zero
  <= ap= (transport-→ Cover loop (loopA (Pos n))) >
  transport Cover loop
  (transport Cover (loopA (Pos n)) Zero)
  <= ap= transport-Cover-loop >
  succ (transport Cover (loopA (Pos n)) Zero)
  <= id >
  succ (encode (loopA (Pos n)))
  <= ap succ (encode-loopA (Pos n)) >
  succ (Pos n) *
encode-loopA (Neg One) = ap= transport-Cover-ll-loop
encode-loopA (Neg (S n)) =
  transport Cover (l loop · loopA (Neg n)) Zero
  <= ap= (transport-→ Cover (l loop) (loopA (Neg n))) >
  transport Cover (l loop) (transport Cover (loopA (Neg n)) Zero)
  <= ap= transport-Cover-ll-loop >
  pred (transport Cover (loopA (Neg n)) Zero)
  <= ap pred (encode-loopA (Neg n)) >
  pred (Neg n) *
```

```
encode-decode : (x : S1) → (c : Cover x)
  → Path (encode (decode (x) c)) c
encode-decode (x) =  $S^1$ -induction
  (λ (x : S1) → (c : Cover x)
    → Path (encode (x) (decode (x) c)) c)
  encode-loopA (λ x' → fst (use-level (use-level HSet-Int _ _) _ _)) x
```

```
decode-encode : (x : S1) (a : Path base x)
  → Path (decode (encode a)) a
decode-encode (x) a =
  path-induction
  (λ (x' : S1) (a' : Path base x')
    → Path (decode (encode a')) a')
  id a
```

```
Ω[S1]-Equiv-Int : Equiv (Path base base) Int
Ω[S1]-Equiv-Int =
  improve (hequiv encode decode decode-encode encode-loopA)
```

```
Ω[S1]-is-Int : (Path base base) = Int
Ω[S1]-is-Int = ua Ω[S1]-Equiv-Int
```

```
n[S1]-is-Int : a One S1 base = Int
n[S1]-is-Int = UnTrunc.path _ _ HSet-Int · ap (Trunc (λ l 0)) Ω[S1]-is-Int
```


Conclusion

We can do **computer-checked proofs** in **synthetic** homotopy theory

- * Proofs are constructive*: can run them
- * Results apply in a variety of settings,
from simplicial sets (hence topological spaces)
to Quillen model categories and ∞ -topoi*
- * New type-theoretic proofs/methods

*work in progress

Homotopy in HoTT

$$\pi_1(S^1) = \mathbb{Z}$$

Freudenthal

Van Kampen

$$\pi_{k < n}(S^n) = 0$$

$$\pi_n(S^n) = \mathbb{Z}$$

Covering spaces

Hopf fibration

$K(G, n)$

**Whitehead
for n-types**

$$\pi_2(S^2) = \mathbb{Z}$$

Cohomology
axioms

$$\pi_3(S^2) = \mathbb{Z}$$

Blakers-Massey

James

Construction

$$\pi_4(S^3) = \mathbb{Z}?$$

**[Brunerie, Finster, Hou,
Licata, Lumsdaine, Shulman]**

Homotopy in HoTT


$$\pi_1(S^1) = \mathbb{Z}$$

$$\pi_{k < n}(S^n) = 0$$

Hopf fibration

$$\pi_2(S^2) = \mathbb{Z}$$

$$\pi_3(S^2) = \mathbb{Z}$$

James
Construction

$$\pi_4(S^3) = \mathbb{Z}?$$

Freudenthal

$$\pi_n(S^n) = \mathbb{Z}$$

K(G,n)

Cohomology
axioms

Blakers-Massey

Van Kampen

Covering spaces

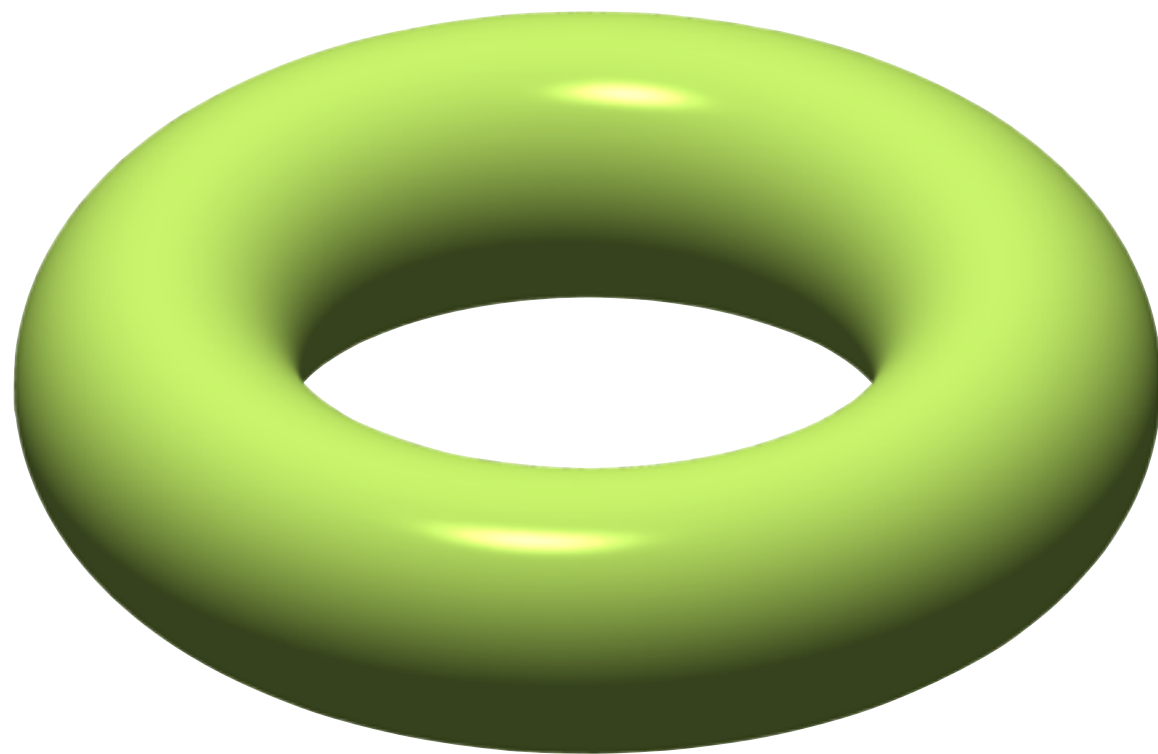
**Whitehead
for n-types**

**[Brunerie, Finster, Hou,
Licata, Lumsdaine, Shulman]**

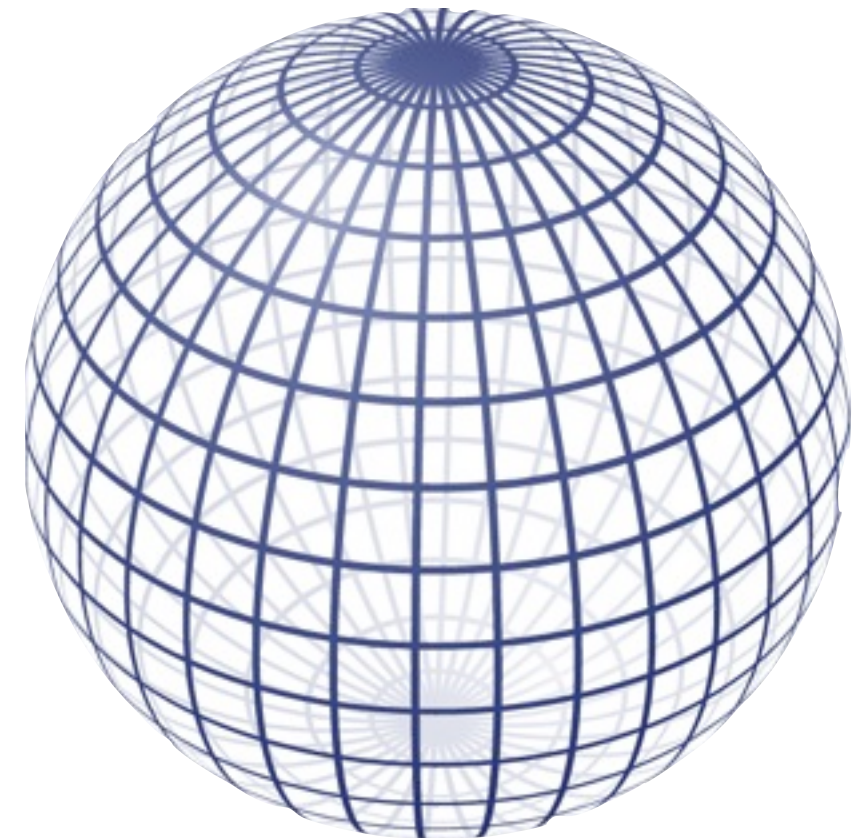
Reading list

- * $\pi_1(S^1) = \mathbb{Z}$ [Licata and Shulman, LICS'13]
- * Other results: forthcoming
Homotopy Type Theory book
- * Blog: homotopytypetheory.org
- * Formalizations:
github.com/dlicata335/hott-agda
github.com/hott/hott-agda
github.com/hott/hott [Coq]

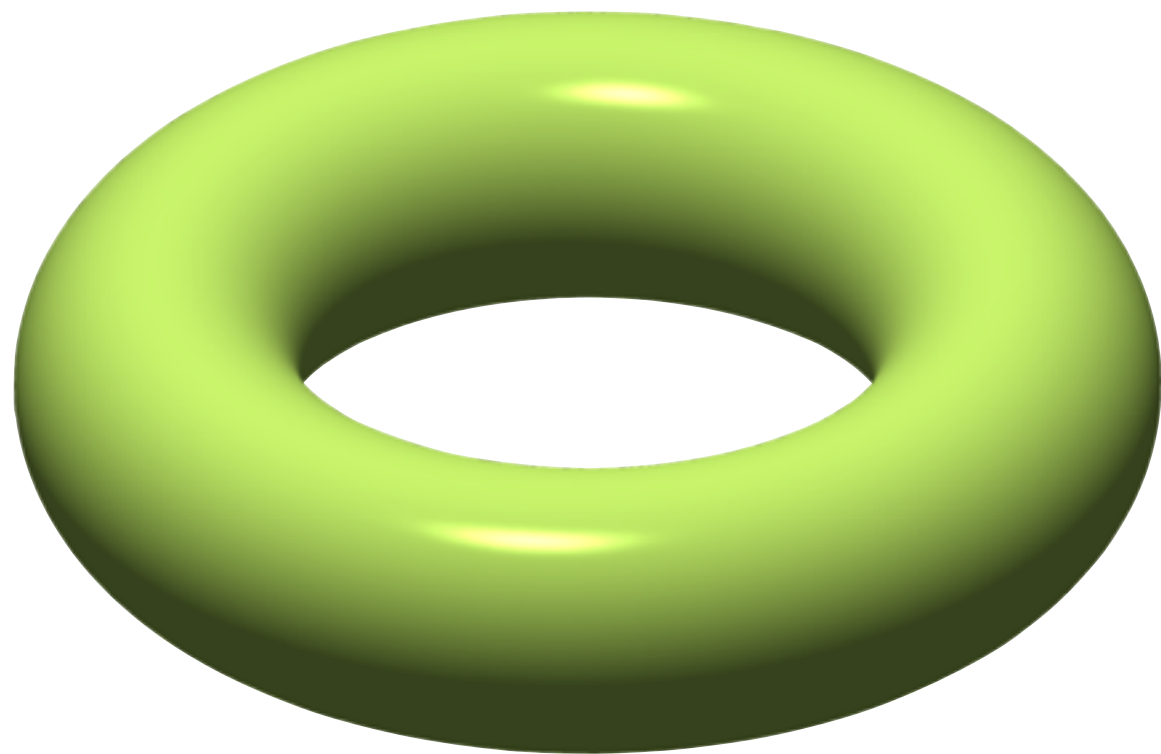
Algebraic invariants



\neq

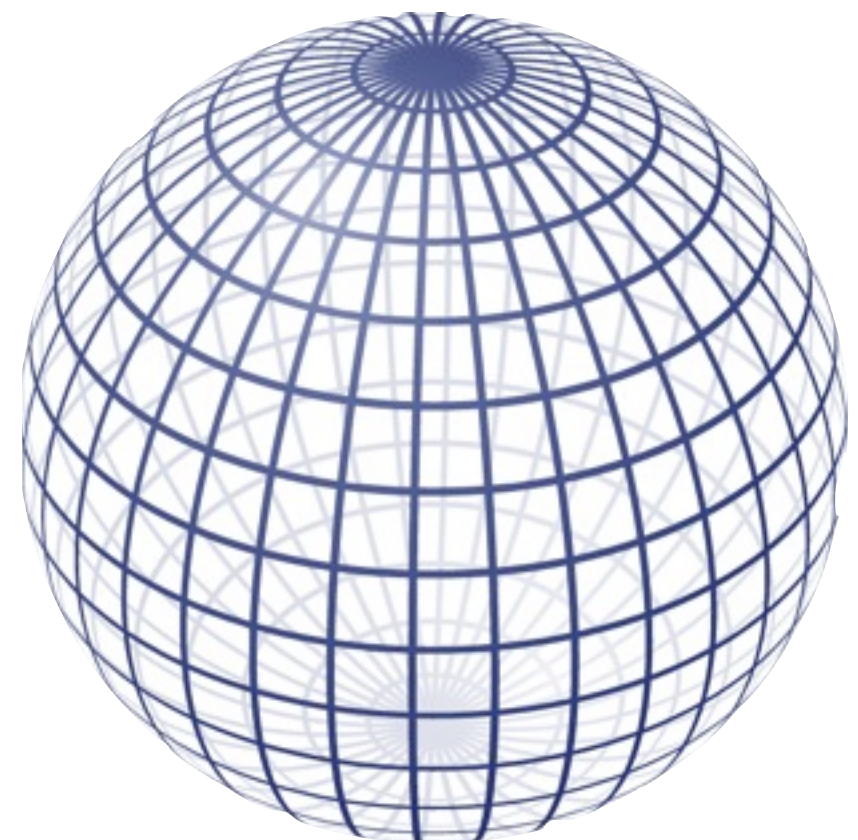


Algebraic invariants



fundamental group
is non-trivial ($\mathbb{Z} \times \mathbb{Z}$)

\neq



fundamental group
is trivial

Homotopy groups

Homotopy groups

- * Fundamental group π_1 : group of loops

Homotopy groups

- * Fundamental group π_1 : group of loops
- * π_2 : group of homotopies

Homotopy groups

- * Fundamental group π_1 : group of loops
- * π_2 : group of homotopies
- * π_3 : group of homotopies between homotopies

Homotopy groups

- * Fundamental group π_1 : group of loops
- * π_2 : group of homotopies
- * π_3 : group of homotopies between homotopies
- * ...

Homotopy groups

k^{th} homotopy group

n-dimensional sphere

	π_1	π_2	π_3	π_4	π_5	π_6	π_7	π_8	π_9	π_{10}	π_{11}	π_{12}	π_{13}	π_{14}	π_{15}
S^0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^1	\mathbb{Z}	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^2	0	\mathbb{Z}	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{12}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^2$	\mathbb{Z}_2^2
S^3	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{12}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^2$	\mathbb{Z}_2^2
S^4	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	$\mathbb{Z} \times \mathbb{Z}_{12}$	\mathbb{Z}_2^2	\mathbb{Z}_2^2	$\mathbb{Z}_{24} \times \mathbb{Z}_3$	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^3	$\mathbb{Z}_{120} \times \mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^5$
S^5	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{24}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{30}	\mathbb{Z}_2	\mathbb{Z}_2^3	$\mathbb{Z}_{72} \times \mathbb{Z}_2$
S^6	0	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{24}	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_{60}	$\mathbb{Z}_{24} \times \mathbb{Z}_2$	\mathbb{Z}_2^3
S^7	0	0	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{24}	0	0	\mathbb{Z}_2	\mathbb{Z}_{120}	\mathbb{Z}_2^3
S^8	0	0	0	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{24}	0	0	\mathbb{Z}_2	$\mathbb{Z} \times \mathbb{Z}_{120}$

[image from wikipedia]

$\pi_k(S^n)$ in HoTT

k^{th} homotopy group

n-dimensional sphere

	π_1	π_2	π_3	π_4	π_5	π_6	π_7	π_8	π_9	π_{10}	π_{11}	π_{12}	π_{13}	π_{14}	π_{15}
S^0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^1	\mathbb{Z}	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^2	0	\mathbb{Z}	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{12}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^2$	\mathbb{Z}_2^2
S^3	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{12}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^2$	\mathbb{Z}_2^2
S^4	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2									
S^5	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{24}							
S^6	0	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{24}	0					
S^7	0	0	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{24}	0	0			
S^8	0	0	0	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{24}	0	0	\mathbb{Z}_2	

[image from wikipedia]

$\pi_k(S^n)$ in HoTT

k^{th} homotopy group

n-dimensional sphere

	π_1	π_2	π_3	π_4	π_5	π_6	π_7	π_8	π_9	π_{10}	π_{11}	π_{12}	π_{13}	π_{14}	π_{15}
S^0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^1	\mathbb{Z}	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S^2	0	\mathbb{Z}	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{12}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^2$	\mathbb{Z}_2^2
S^3	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_{12}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^2$	\mathbb{Z}_2^2
S^4	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^2$	\mathbb{Z}_2^2
S^5	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^2$	\mathbb{Z}_2^2
S^6	0	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	$\mathbb{Z}_{84} \times \mathbb{Z}_2^2$	\mathbb{Z}_2^2
S^7	0	0	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	$\mathbb{Z}_{12} \times \mathbb{Z}_2$	\mathbb{Z}_2^2
S^8	0	0	0	0	0	0	0	\mathbb{Z}	\mathbb{Z}_2	\mathbb{Z}_2	\mathbb{Z}_3	\mathbb{Z}_{15}	\mathbb{Z}_2	\mathbb{Z}_2^2	\mathbb{Z}_2^2

[image from wikipedia]

$$\pi_n(S^n) = \mathbb{Z} \text{ for } n \geq 1$$

Proof: Induction on n

* Base case: $\pi_1(S^1) = \mathbb{Z}$

* Inductive step: $\pi_{n+1}(S^{n+1}) = \pi_n(S^n)$

$$\pi_n(S^n) = \mathbb{Z} \text{ for } n \geq 1$$

Proof: Induction on n

* Base case: $\pi_1(S^1) = \mathbb{Z}$

* Inductive step: $\pi_{n+1}(S^{n+1}) = \pi_n(S^n)$

Key lemma: $\tau_n(S^n) = \tau_n(\Omega(S^{n+1}))$

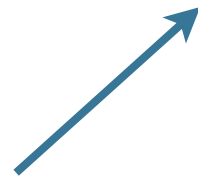
$$\pi_n(S^n) = \mathbb{Z} \text{ for } n \geq 1$$

Proof: Induction on n

* Base case: $\pi_1(S^1) = \mathbb{Z}$

* Inductive step: $\pi_{n+1}(S^{n+1}) = \pi_n(S^n)$

Key lemma: $\tau_n(S^n) = \tau_n(\Omega(S^{n+1}))$



n-truncation:

**best approximation of a type such
that all $(n+1)$ -paths are equal**

$$\pi_n(S^n) = \mathbb{Z} \text{ for } n \geq 1$$

Proof: Induction on n

* Base case: $\pi_1(S^1) = \mathbb{Z}$

* Inductive step: $\pi_{n+1}(S^{n+1}) = \pi_n(S^n)$

Key lemma: $\tau_n(S^n) = \tau_n(\Omega(S^{n+1}))$

n-truncation:
best approximation of a type such
that all $(n+1)$ -paths are equal

**higher inductive type
generated by**
 $\text{base}_n : S^n$
 $\text{loop}_n : \Omega^n(S^n)$

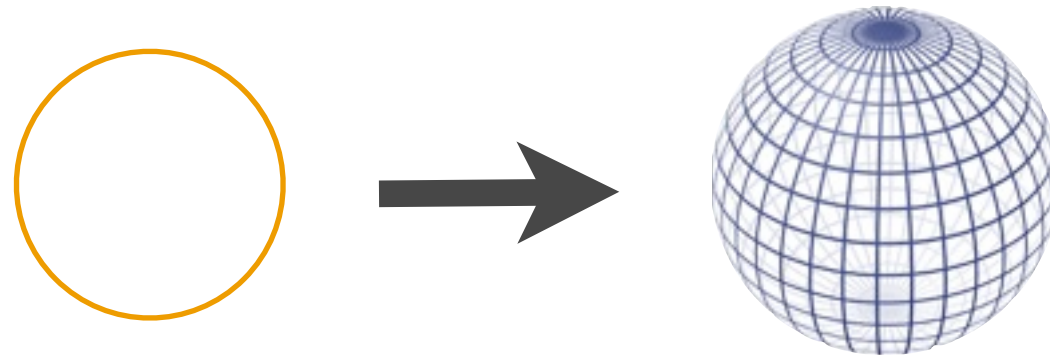
$$\tau_n(S^n) = \tau_n(\Omega(S^{n+1}))$$

n -truncation of S^n is the type of “codes” for loops on S^{n+1}

$$\tau_n(S^n) = \tau_n(\Omega(S^{n+1}))$$

n -truncation of S^n is the type of “codes” for loops on S^{n+1}

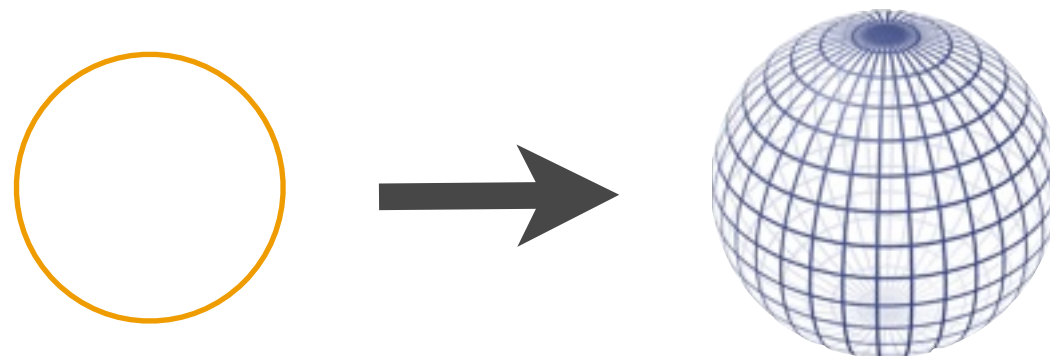
- * Decode: promote n -dimensional loop on S^n to $n+1$ -dimensional loop on S^{n+1}



$$\tau_n(S^n) = \tau_n(\Omega(S^{n+1}))$$

n -truncation of S^n is the type of “codes” for loops on S^{n+1}

- * Decode: promote n -dimensional loop on S^n to $n+1$ -dimensional loop on S^{n+1}



- * Encode: define fibration $\text{Code}(x : S^{n+1})$ with
 $\text{Code}(\text{base}_{n+1}) := \tau_n(S^n)$
 $\text{Code}(\text{loop}_{n+1}) := \text{equivalence } \tau_n(S^n) \xrightarrow{\sim} \tau_n(S^n)$
“rotating by loop_n ”

$\pi_2(S^2)$: Hopf fibration

