Homotopy Theory in Type Theory

Guillaume Brunerie, Daniel R. Licata,
and Peter LeFanu Lumsdaine

Joint work with Eric Finster,
Kuen-Bang Hou (Favonia), Michael Shulman

Homotopy Theory

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

[image from wikipedia]

Homotopy

Deformation of one path into another

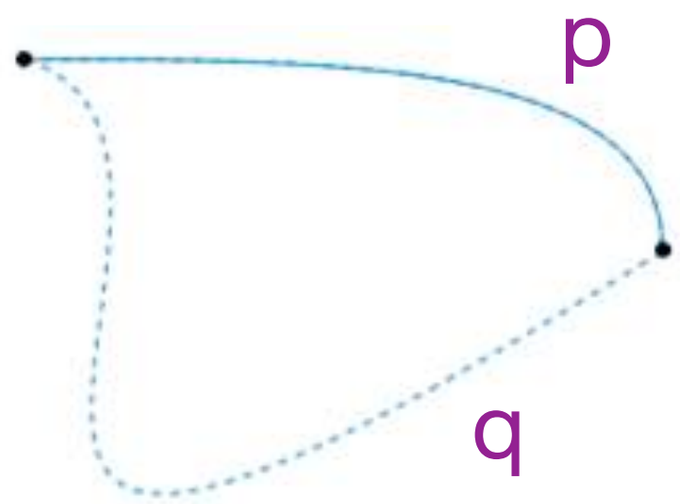
p

q

[image from wikipedia]

Homotopy

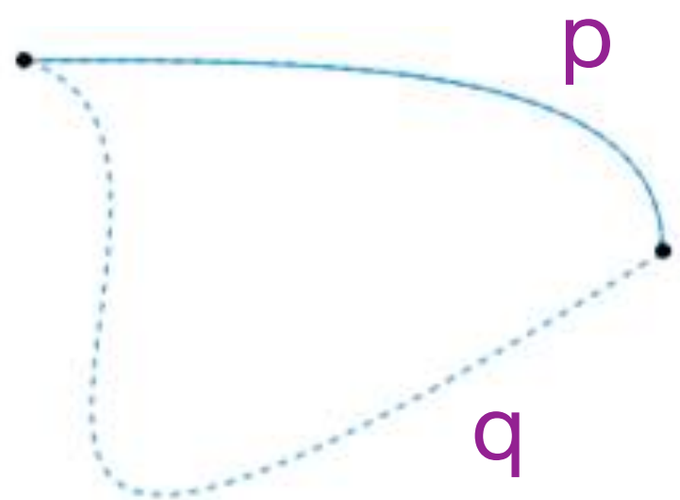
Deformation of one path into another



[image from wikipedia]

Homotopy

Deformation of one path into another

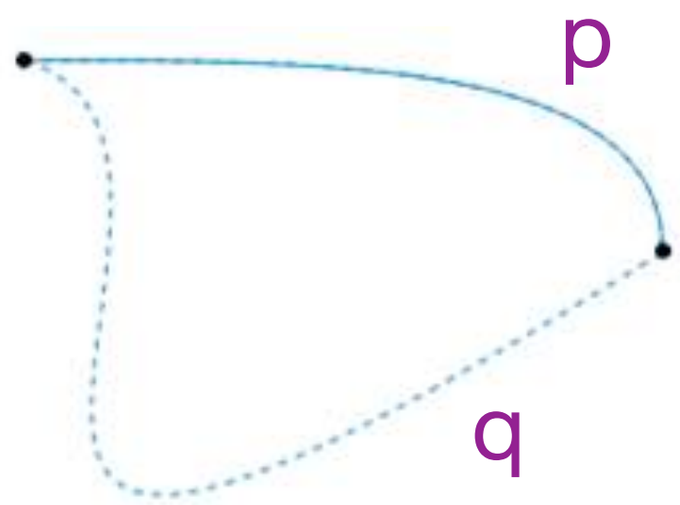


= 2-dimensional *path between paths*

[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]

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]

Type Theory

An alternative to set theory, organized around *types:*

- Basic data types ($\mathbb{N}$, $\mathbb{Z}$, booleans, lists, …)
- Functions

```
double : ℕ → ℕ
double 0 = 0
double (n +1) = double n + 2
```

- Unifies sets and logic

Propositions as Types

1. A proposition is represented by a type
2. A proof is represented by an element of that type

```
∀x: ℕ. double(x) = 2*x
```

type **of proofs of equality**

Propositions as Types

1. A proposition is represented by a type
2. A proof is represented by an element of that type

```
proof : ∀x: ℕ. double(x) = 2*x
```

type **of proofs of equality**

Propositions as Types

1. A proposition is represented by a type
2. A proof is represented by an element of that type

```
proof : ∀x: ℕ. double(x) = 2*x

proof 0 = reflexivity
proof (n +1) = ...
```

Propositions as Types

1. A proposition is represented by a type
2. A proof is represented by an element of that type

```
proof : ∀x: ℕ. double(x) = 2*x

proof 0 = reflexivity
proof (n +1) = ...
```

proof by case analysis represented by a function defined by cases

Type are sets?

Traditional view:

type theory	**set theory**
`<element> : <type>`	`x ∈ S`
$\langle elem_1 \rangle = \langle elem_2 \rangle$	`x = y`

In set theory, an equation is a *proposition*:
we don't ask *why* 1+1=2

Type are sets?

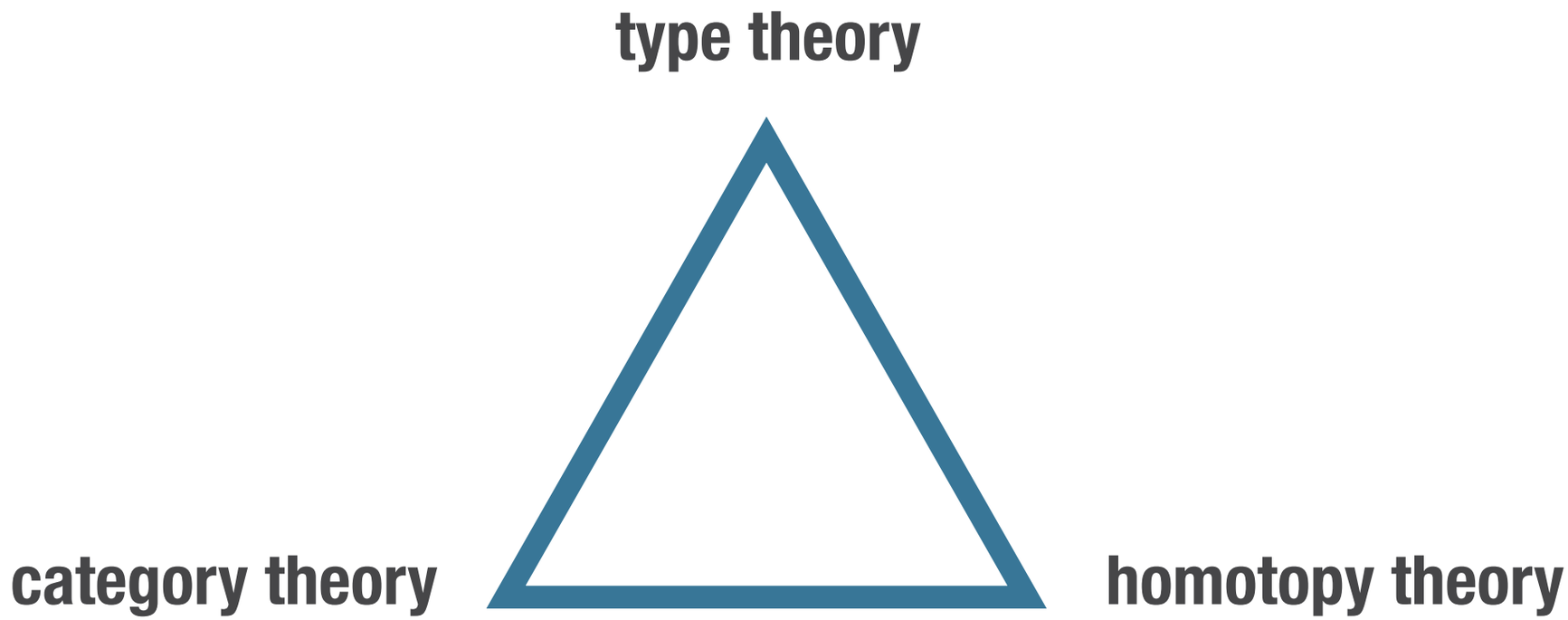
Traditional view:

type theory	**set theory**
`<element> : <type>`	`x ∈ S`
`<proof> : <elem1> = <elem2>`	`x = y`

In set theory, an equation is a *proposition*: we don't ask *why* 1+1=2

In type theory, an equation has a `<proof>`

Homotopy Type Theory



Types are ∞-groupoids

[Hofmann,Streicher,Awodey,Warren,Voevodsky
Lumsdaine,Gambino,Garner,van den Berg]

Types are ∞-groupoids

type theory	**set theory**
`<elem> : <type>`	$x \in S$
`<proof> : <elem`$_1$`> = <elem`$_2$`>`	$x = y$

Types are ∞-groupoids

type theory	**set theory**
`<elem> : <type>`	$x \in S$
`<proof> : <elem`$_1$`> = <elem`$_2$`>`	$x = y$
`<2-proof> : <proof`$_1$`> = <proof`$_2$`>`	

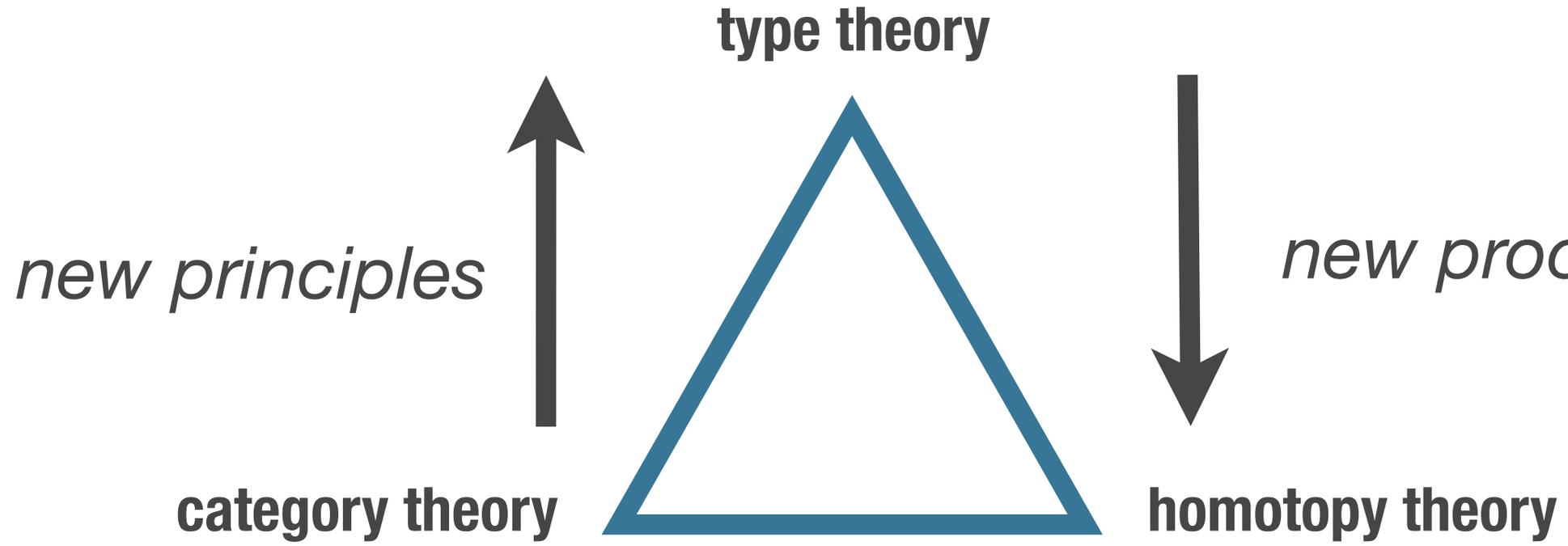
Types are ∞-groupoids

type theory	**set theory**
`<elem> : <type>`	$x \in S$
`<proof> : <elem1> = <elem2>`	$x = y$
`<2-proof> : <proof1> = <proof2>`	
`<3-proof> : <2-proof1> = <2-proof2>`	

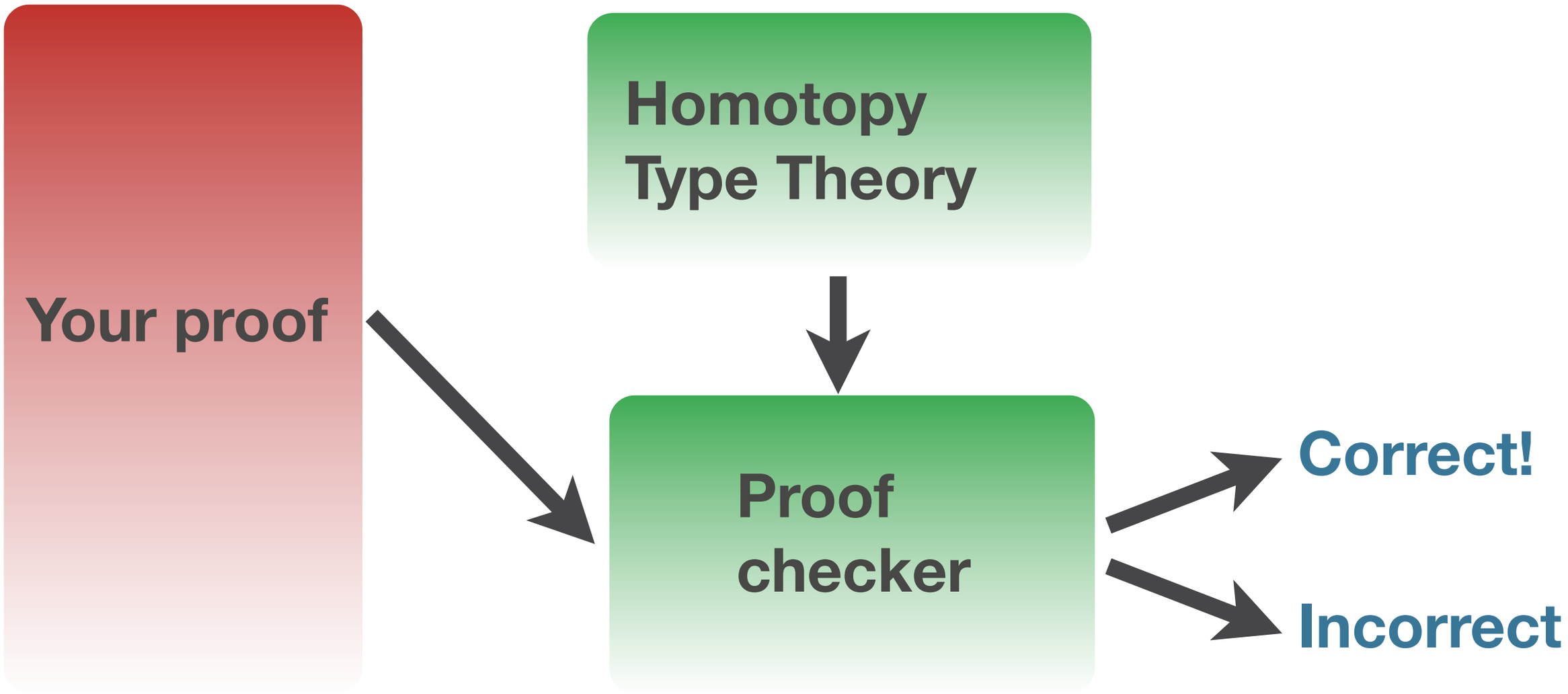
Types are ∞-groupoids

type theory	**set theory**
`<elem> : <type>`	$x \in S$
`<proof> : <elem`$_1$`> = <elem`$_2$`>`	$x = y$
`<2-proof> : <proof`$_1$`> = <proof`$_2$`>`	
`<3-proof> : <2-proof`$_1$`> = <2-proof`$_2$`>`	
⋮	

Homotopy Type Theory



Computer-checked proofs



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.

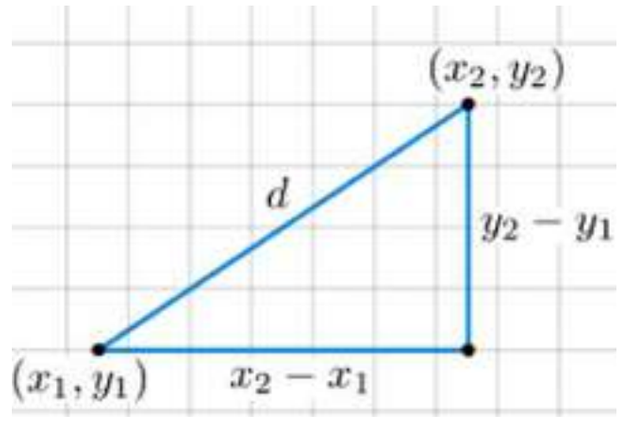
II.

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.

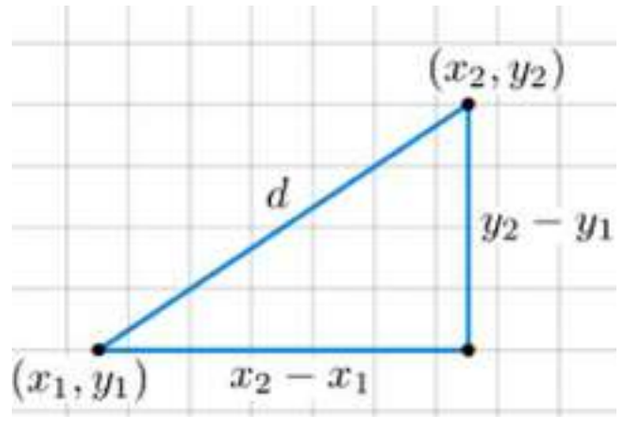
II.

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 map [0,1] → X

[image from wikipedia]

Synthetic homotopy theory

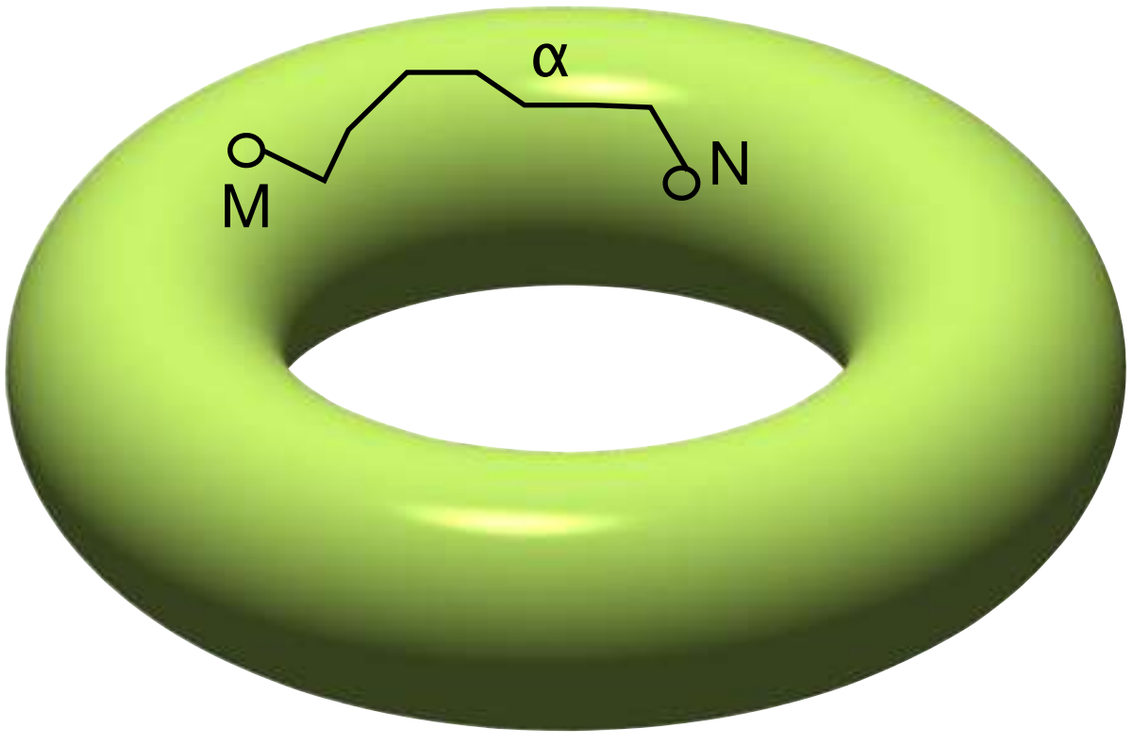
homotopy theory	**type theory**
space	`<type>`
points	`<element> : <type>`
paths	`<proof> : <elem`$_1$`> = <elem`$_2$`>`
homotopies	`<2-proof> : <proof`$_1$`> = <proof`$_2$`>`
⋮	⋮

Synthetic homotopy theory

homotopy theory	**type theory**
space	`<type>`
points	`<element> : <type>`
paths	`<proof> : <elem₁> = <elem₂>`
homotopies	`<2-proof> : <proof₁> = <proof₂>`
⋮	⋮

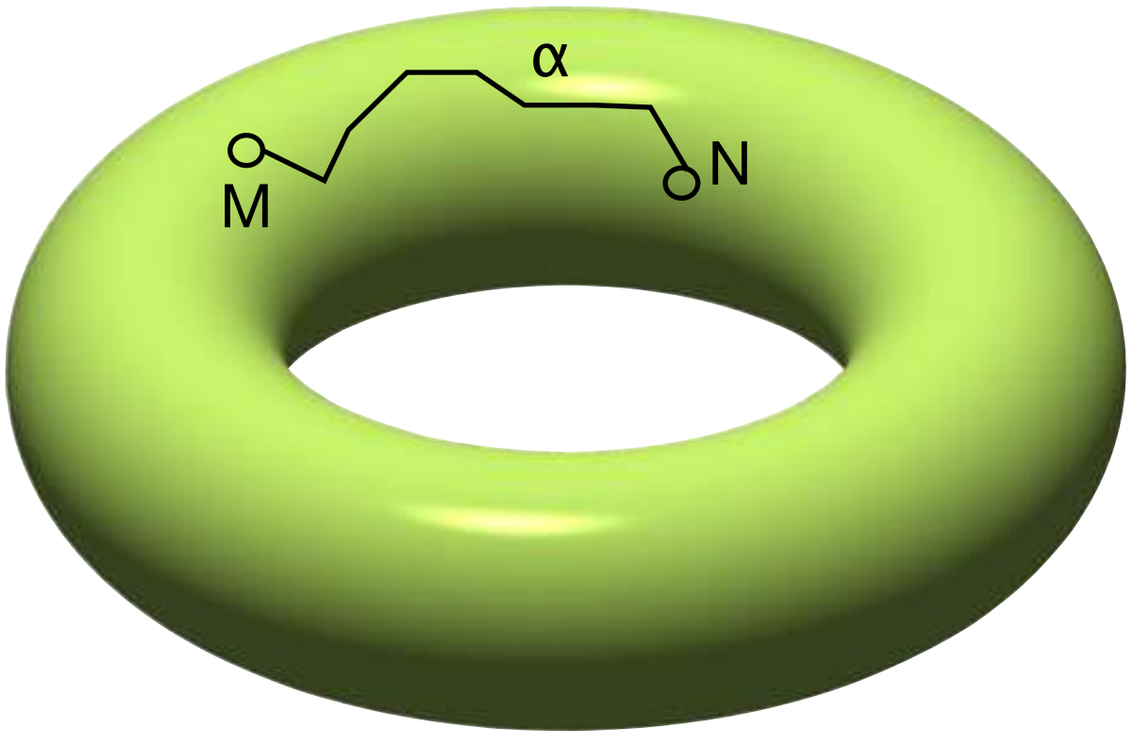
*A path is **not** a map [0,1]→X; it is a basic notion*

Spaces as types



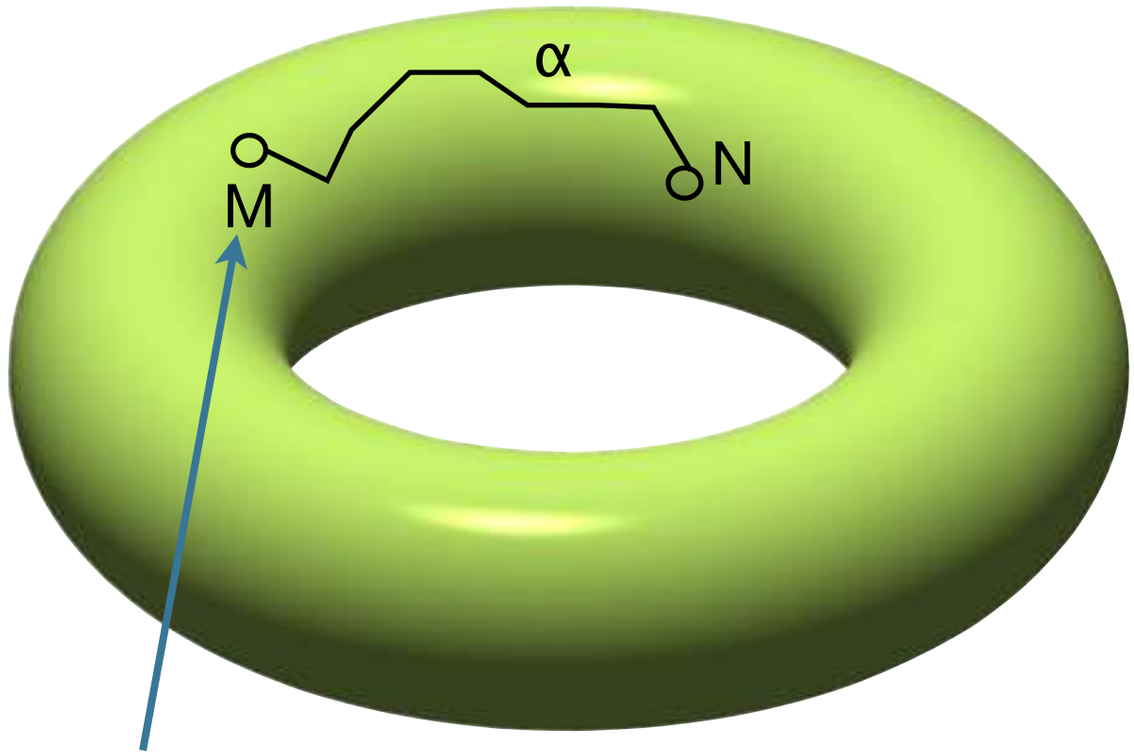
Spaces as types

a space is a type A



Spaces as types

a space is a type A

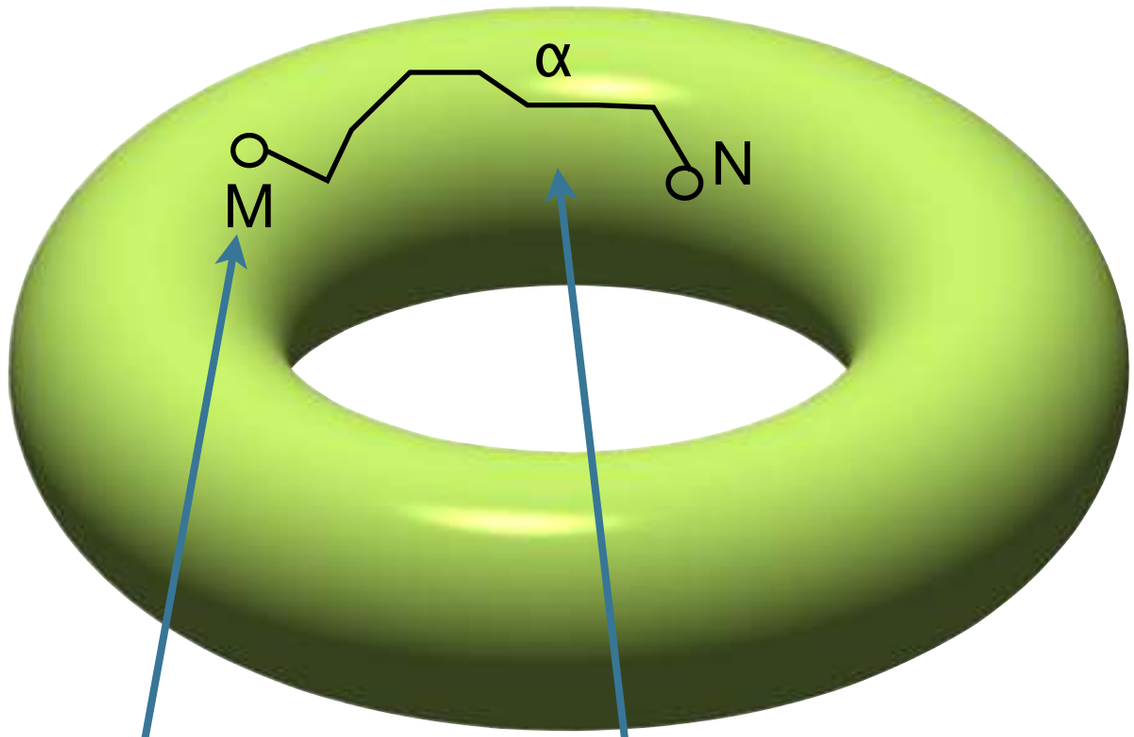


points are elements

M : A

Spaces as types

a space is a type A



points are elements

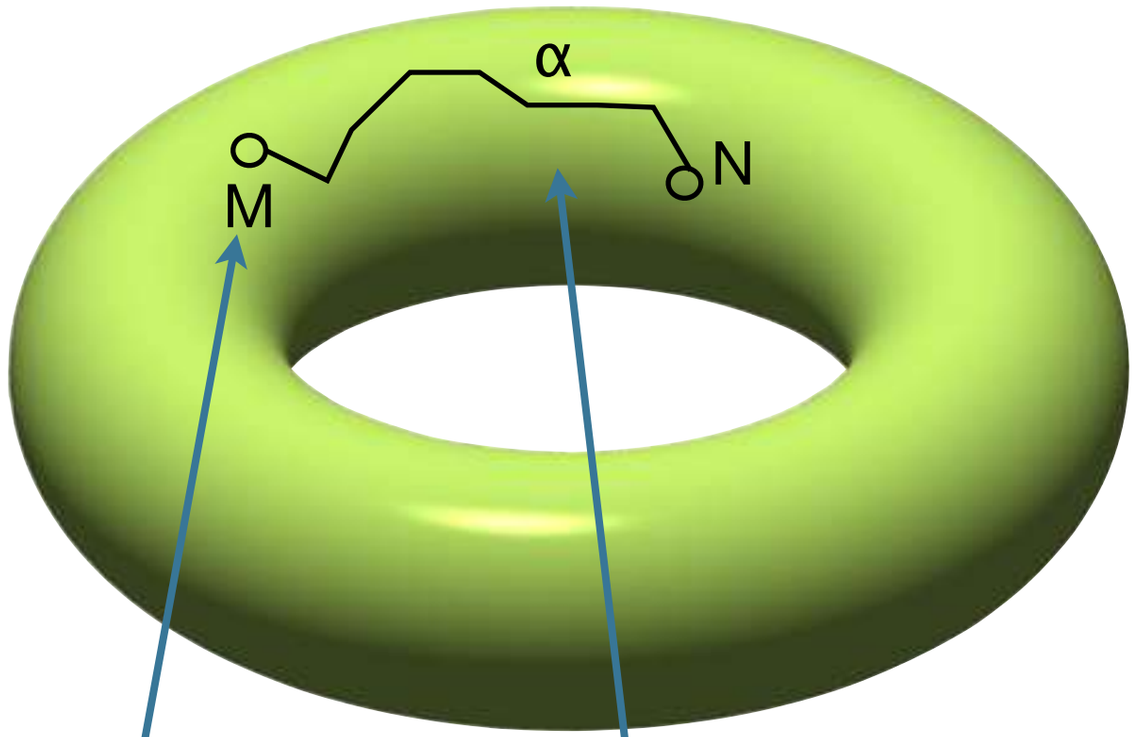
M : A

paths are *proofs of equality*

$\alpha : M =_A N$

Spaces as types

a space is a type A



points are elements

M : A

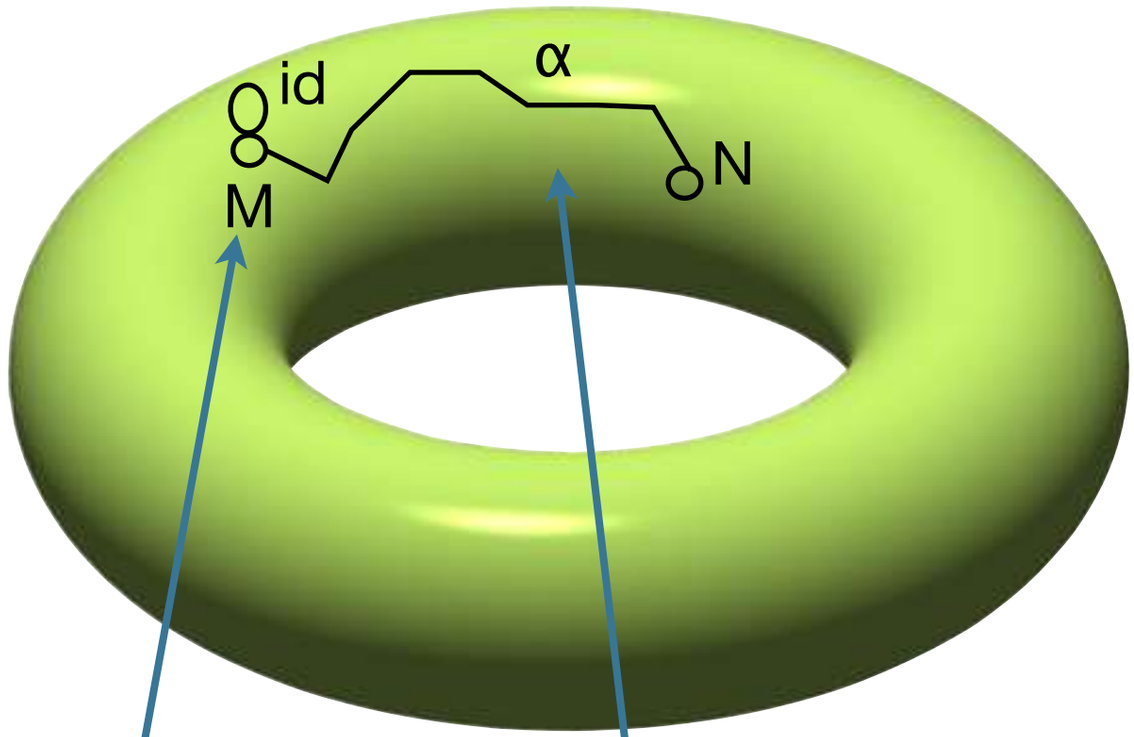
paths are *proofs of equality*

$\alpha : M =_A N$

path operations

Spaces as types

a space is a type A



points are elements

M : A

paths are *proofs of equality*

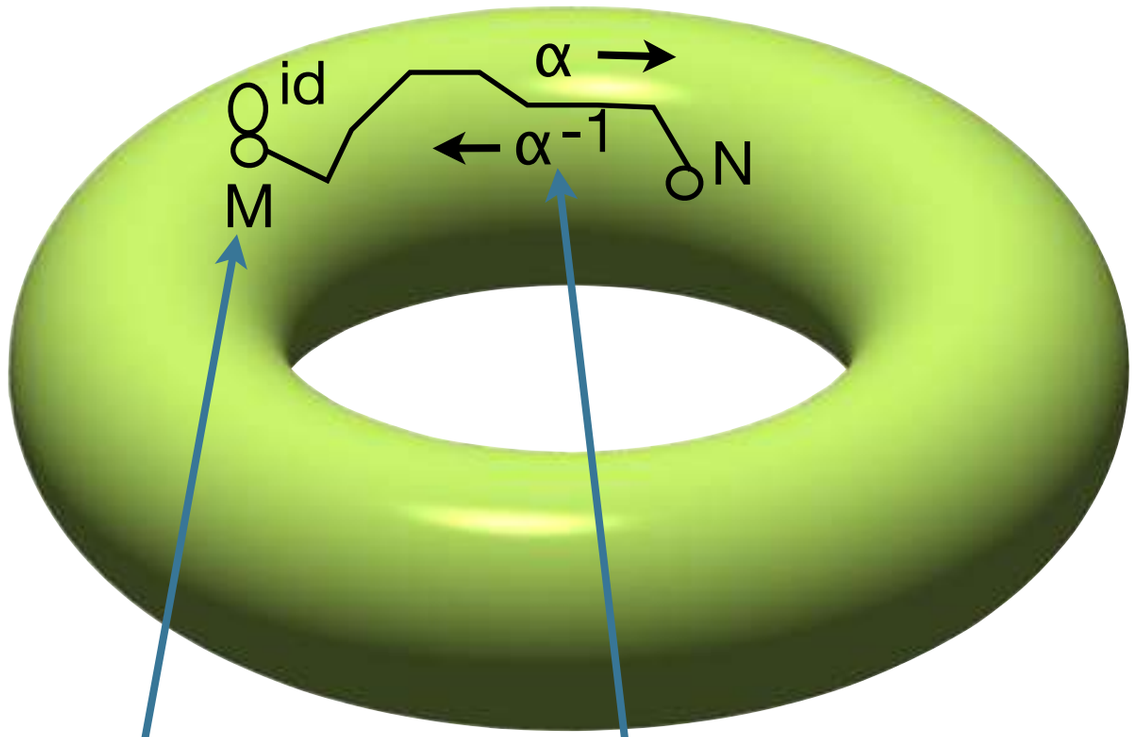
$\alpha : M =_A N$

path operations

```
id      : M = M (refl)
```

Spaces as types

a space is a type A



points are elements

M : A

paths are *proofs of equality*

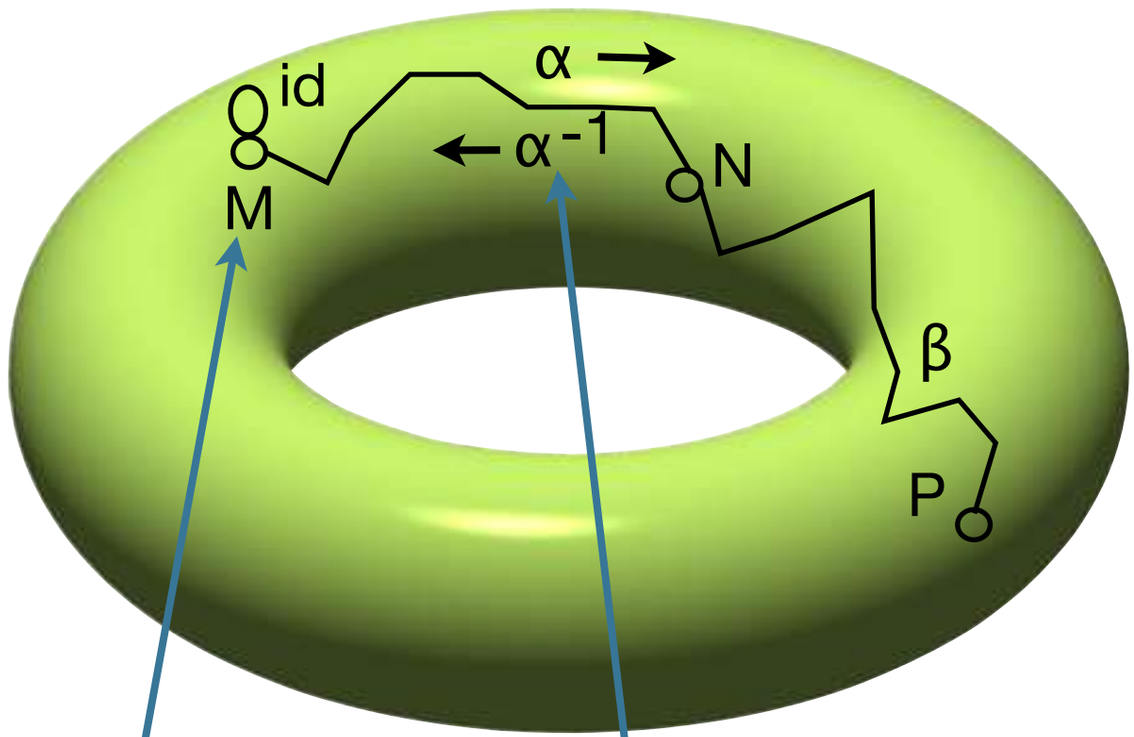
$\alpha : M =_A N$

path operations

```
id      : M = M (refl)
α⁻¹     : N = M (sym)
```

Spaces as types

a space is a type A



points are elements

M : A

paths are *proofs of equality*

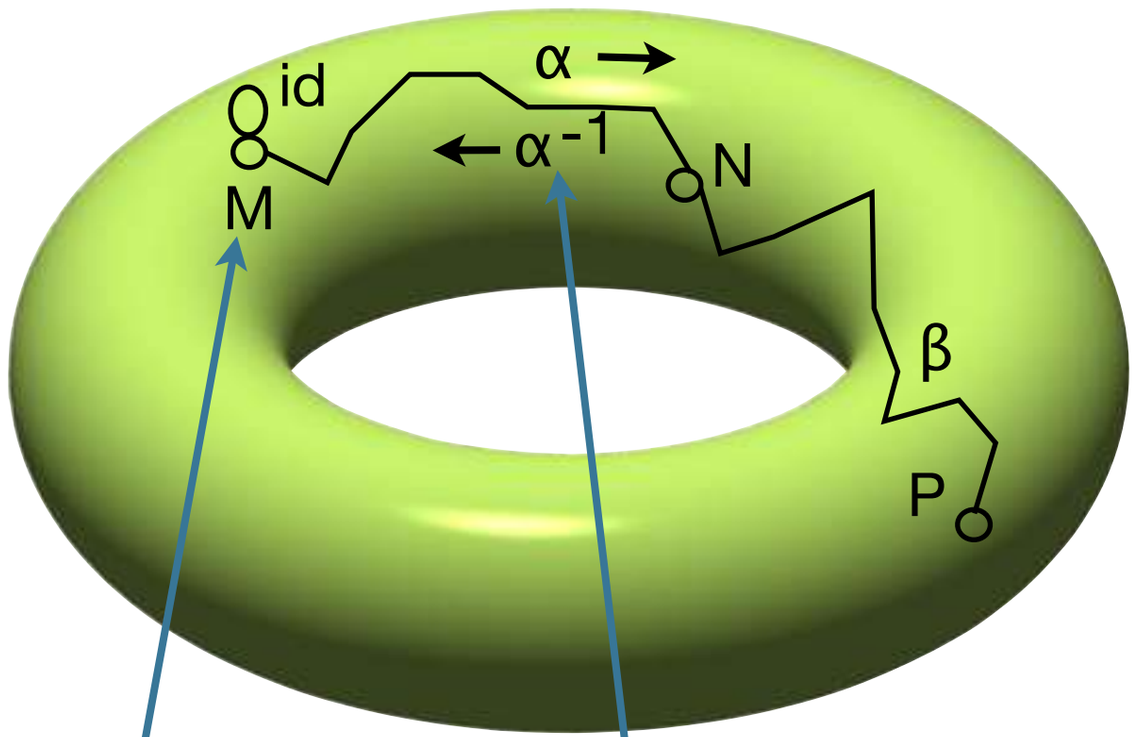
α : M =$_A$ N

path operations

```
id      : M = M (refl)
α⁻¹     : N = M (sym)
β o α   : M = P (trans)
```

Spaces as types

a space is a type A



points are elements

M : A

paths are *proofs of equality*

α : M =$_A$ N

path operations

```
id      : M = M (refl)
α⁻¹     : N = M (sym)
β o α   : M = P (trans)
```

homotopies

```
id o α = α

α⁻¹ o α = id

γ o (β o α)
    = (γ o β) o α
```

Spaces as types

a space is a type A

points are elements

$M : A$

paths are *proofs of equality*

$\alpha : M =_A N$

path operations

$\mathsf{id} : M = M$ (refl)

$\alpha^{-1} : N = M$ (sym)

$\beta \circ \alpha : M = P$ (trans)

homotopies

$\mathsf{id} \circ \alpha = \alpha$

$\alpha^{-1} \circ \alpha = \mathsf{id}$

$\gamma \circ (\beta \circ \alpha) = (\gamma \circ \beta) \circ \alpha$

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

Some results

Homotopy Theoretic

Type Theoretic

Some results

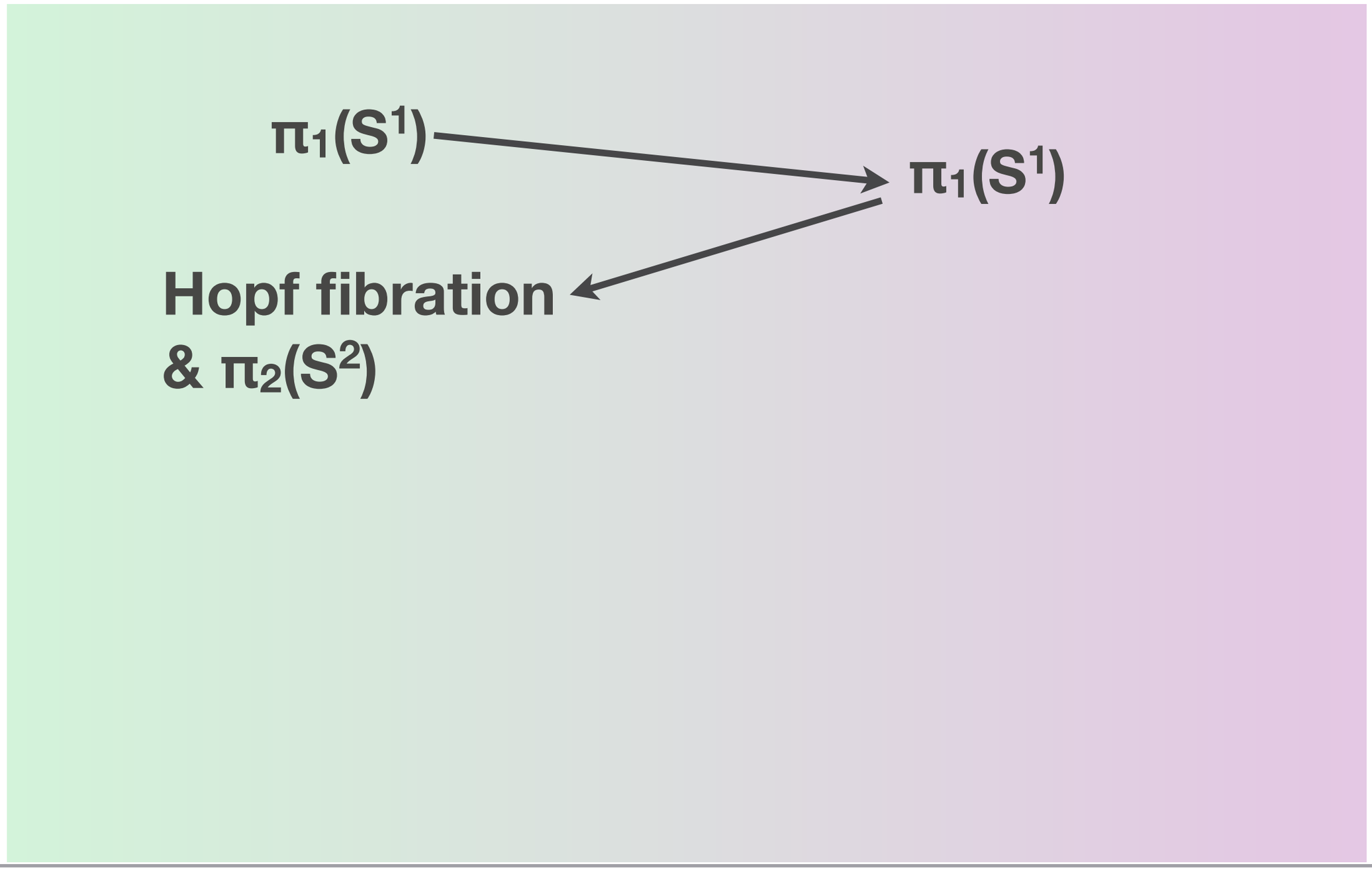
Homotopy Theoretic	Type Theoretic
$\pi_1(S^1)$	

Some results

Homotopy Theoretic	Type Theoretic
$\pi_1(S^1)$	$\pi_1(S^1)$

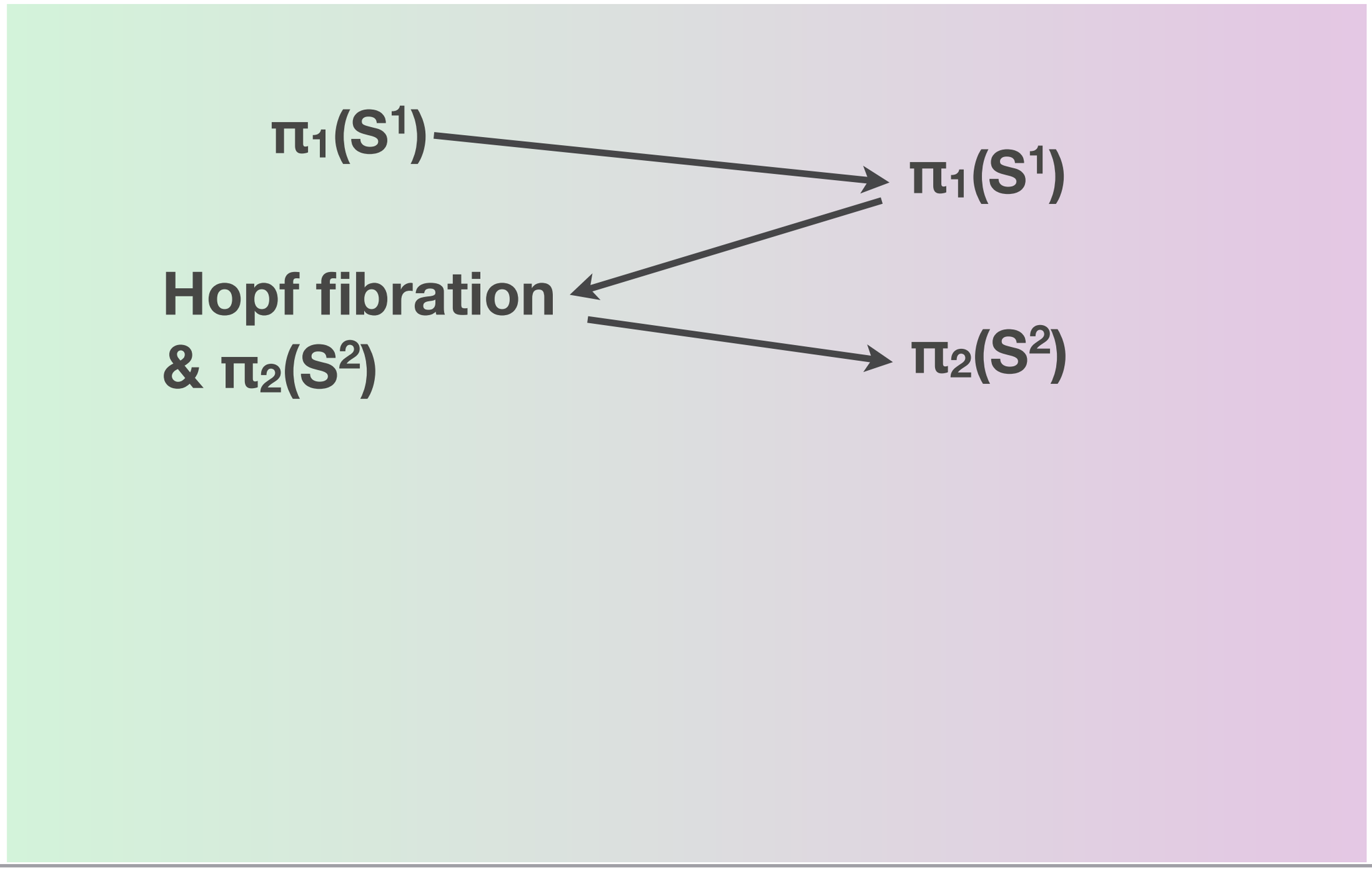
Some results

Homotopy Theoretic **Type Theoretic**



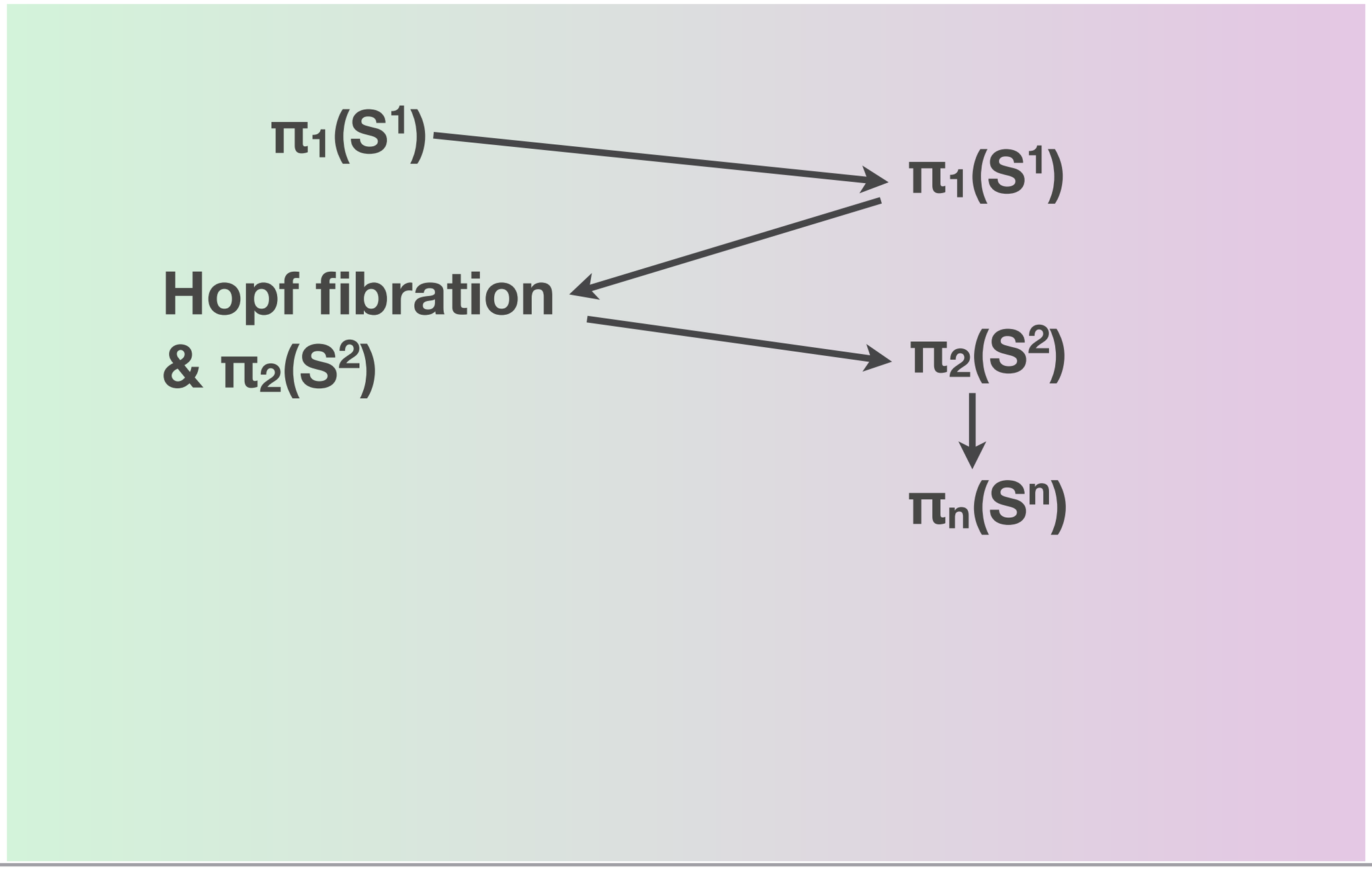
Some results

Homotopy Theoretic **Type Theoretic**



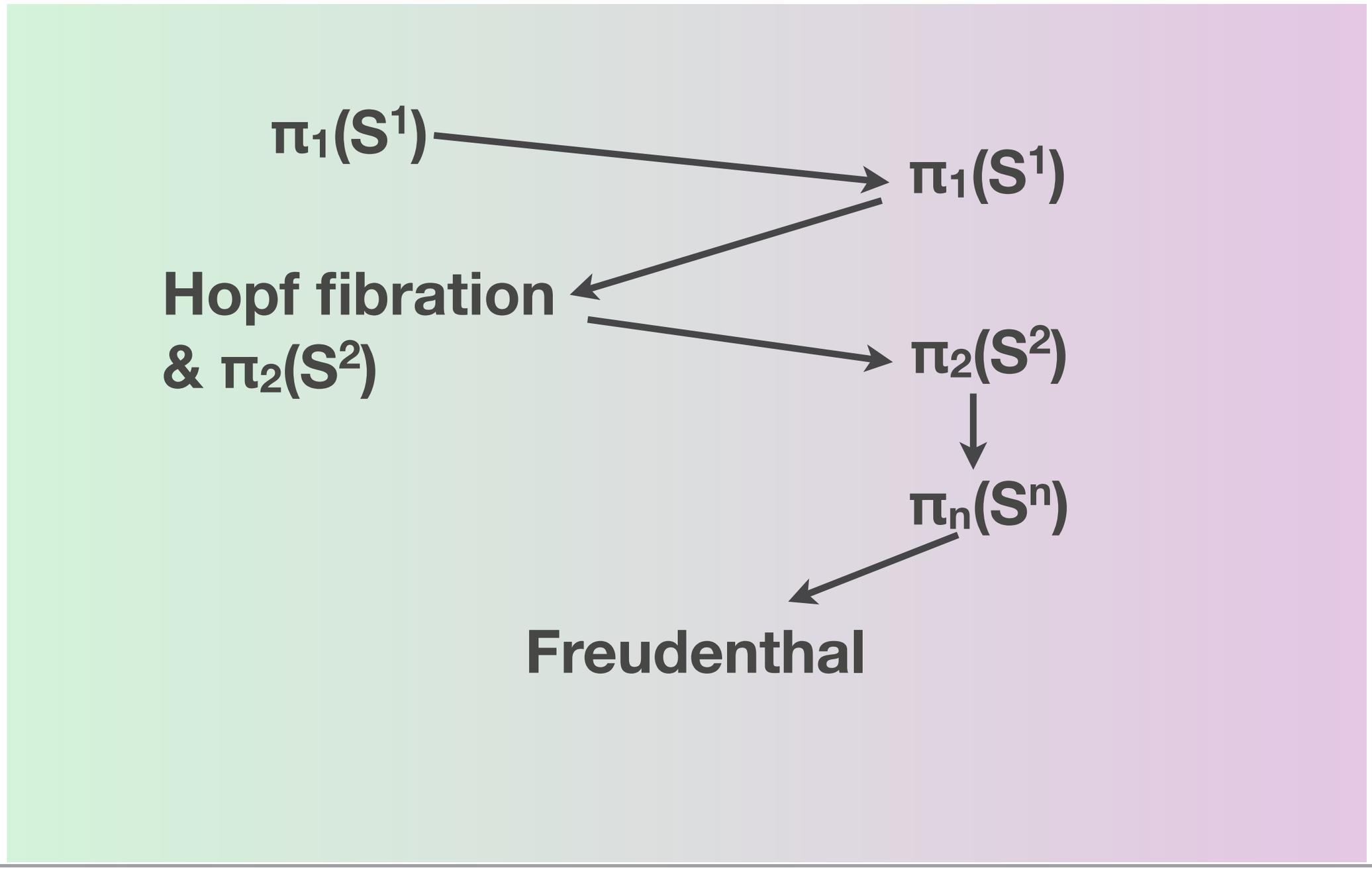
Some results

Homotopy Theoretic **Type Theoretic**



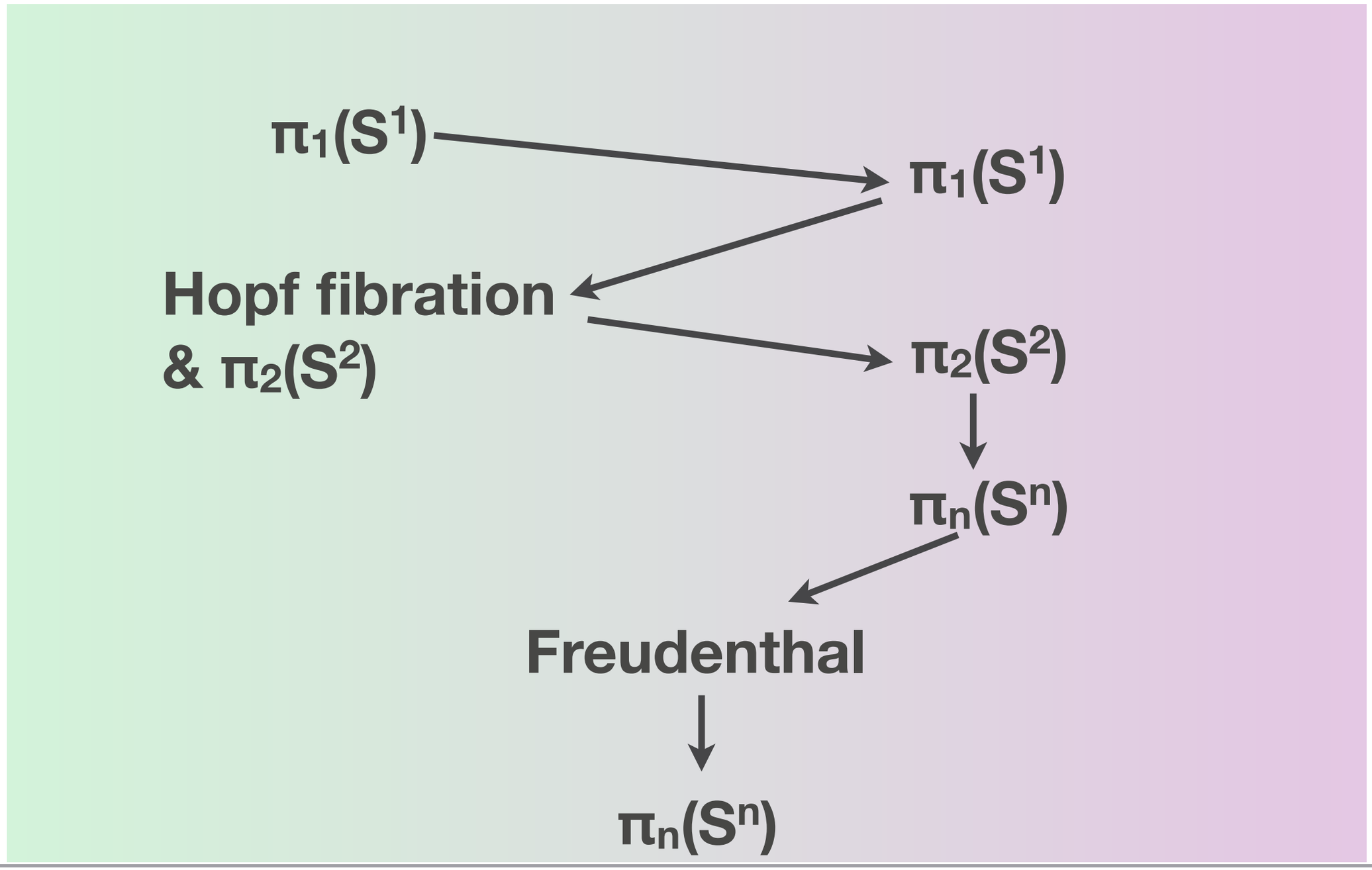
Some results

Homotopy Theoretic **Type Theoretic**



Some results

Homotopy Theoretic **Type Theoretic**



Outline

1. $\pi_1(S^1) = \mathbb{Z}$
2. The Hopf fibration
3. Connectedness and Freudenthal Suspension

Outline

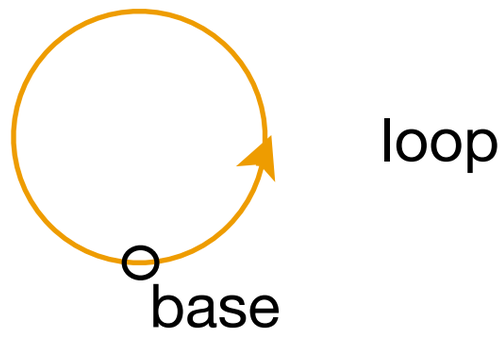
1.$\pi_1(S^1) = \mathbb{Z}$

2.The Hopf fibration

3.Connectedness and Freudenthal Suspension

Higher inductive types

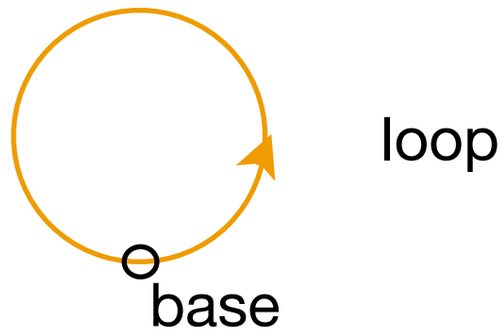
Circle is ***inductively generated*** by



Higher inductive types

Circle is ***inductively generated*** by

```
base : Circle
loop : base = base
```

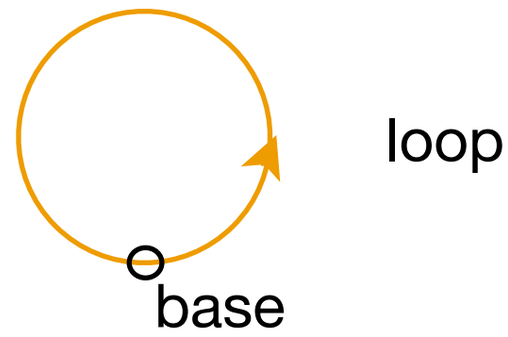


Higher inductive types

Circle is ***inductively generated*** by

point `base : Circle`

`loop : base = base`

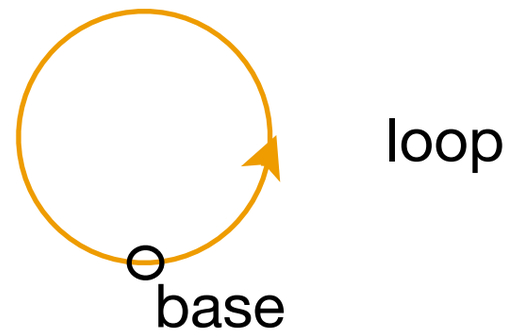


Higher inductive types

Circle is ***inductively generated*** by

point `base : Circle`

path `loop : base = base`

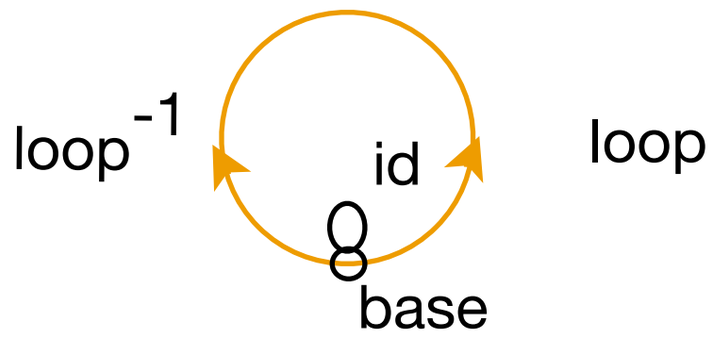


Higher inductive types

Circle is ***inductively generated*** by

point `base : Circle`

path `loop : base = base`



Free ∞-groupoid with these generators

`id` `inv : loop o loop^-1 = id`

`loop^-1` `...`

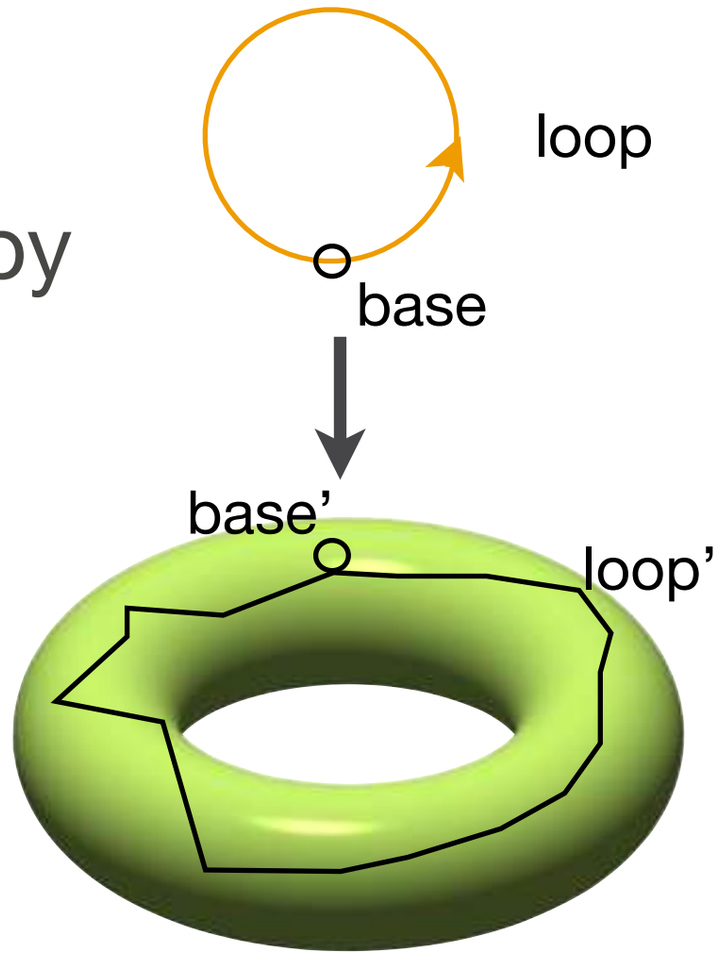
`loop o loop`

Higher inductive types

Circle recursion:

function `Circle → X` determined by

```
base' : X
loop' : base' = base'
```

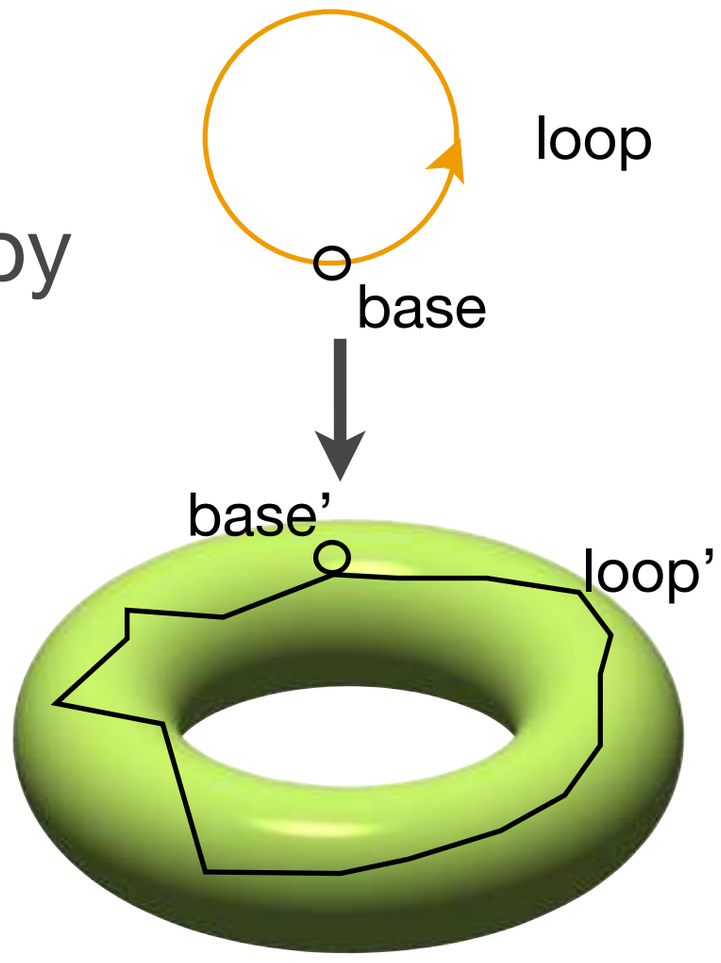


Higher inductive types

Circle recursion:

function `Circle → X` determined by

```
base' : X
loop' : base' = base'
```



Circle induction: To prove a predicate `P` for all points on the circle, suffices to prove `P(base)`, continuously in the loop

Fundamental group of circle

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

Fundamental group of circle

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

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

Fundamental group of circle

Definition. $\Omega(S^1)$ is the **space** 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 +

Proof: two mutually inverse functions

$$winding : \Omega(S^1) \to \mathbb{Z}$$

$$loop^n : \mathbb{Z} \to \Omega(S^1)$$

Fundamental group of circle

Definition. $\Omega(S^1)$ is the **space** 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 +

Proof: two mutually inverse functions

$$winding : \Omega(S^1) \to \mathbb{Z}$$

$$loop^n : \mathbb{Z} \to \Omega(S^1)$$

Corollary: $\pi_1(S^1)$ is isomorphic to $\mathbb{Z}$

Fundamental group of circle

Definition. $\Omega(S^1)$ is the **space** 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 +

Proof: two mutually inverse functions

$$winding : \Omega(S^1) \to \mathbb{Z}$$

$$loop^n : \mathbb{Z} \to \Omega(S^1)$$

Corollary: $\pi_1(S^1)$ is isomorphic to $\mathbb{Z}$

0-truncation (set of connected components) of Ω(S1)

Fundamental group of circle

Definition. $\Omega(S^1)$ is the **space** 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 +

Proof: two mutually inverse functions

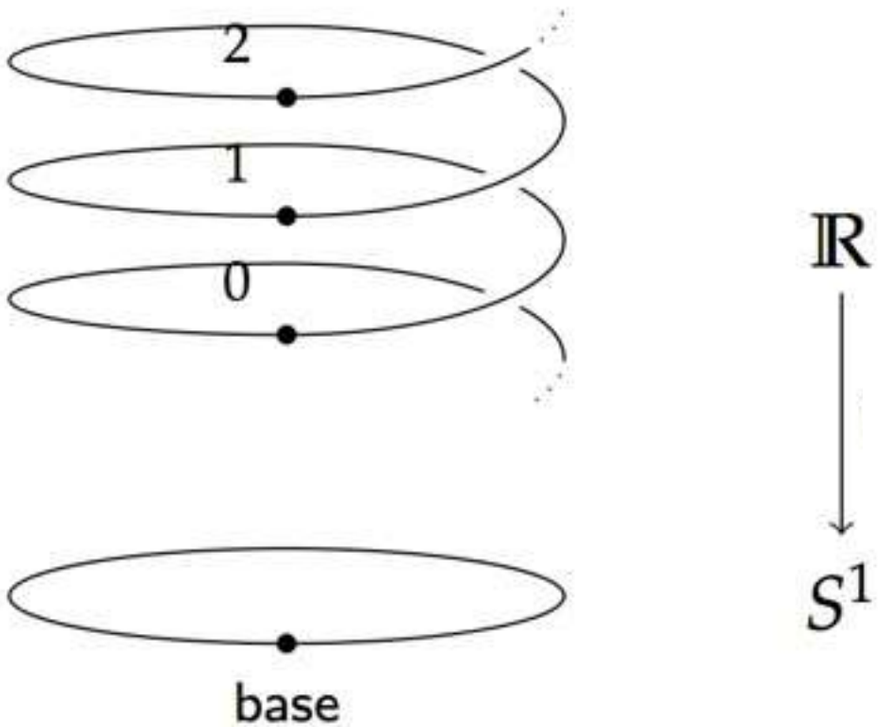
$$winding : \Omega(S^1) \to \mathbb{Z}$$

$$loop^n : \mathbb{Z} \to \Omega(S^1)$$

Corollary: $\pi_1(S^1)$ is isomorphic to $\mathbb{Z}$
$\pi_k(S^1)$ trivial otherwise

0-truncation (set of connected components) of Ω(S1)

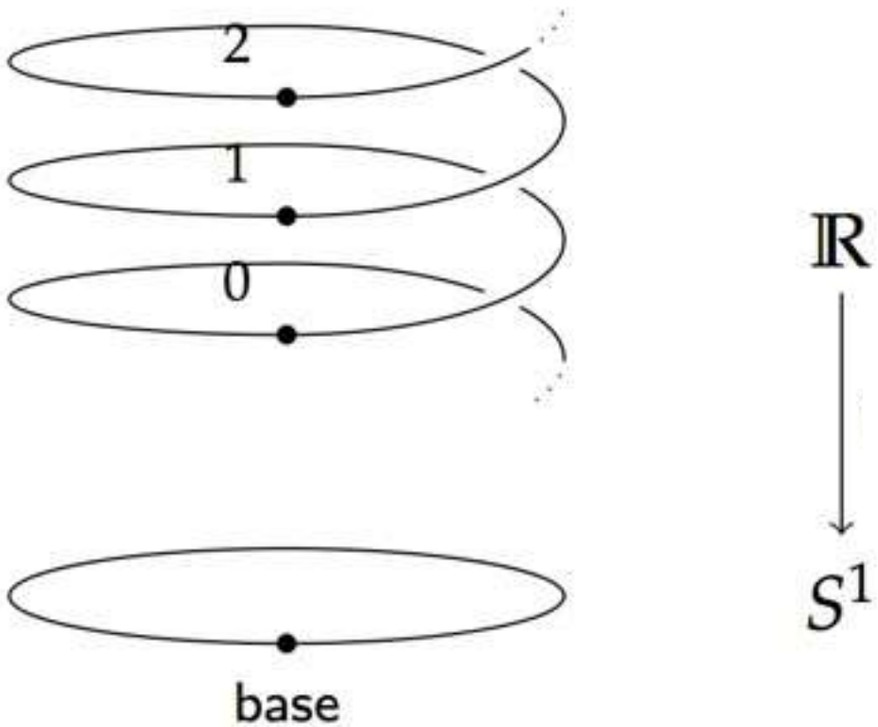
Universal Cover



$w : \Omega(S^1) \to \mathbb{Z}$

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

Universal Cover

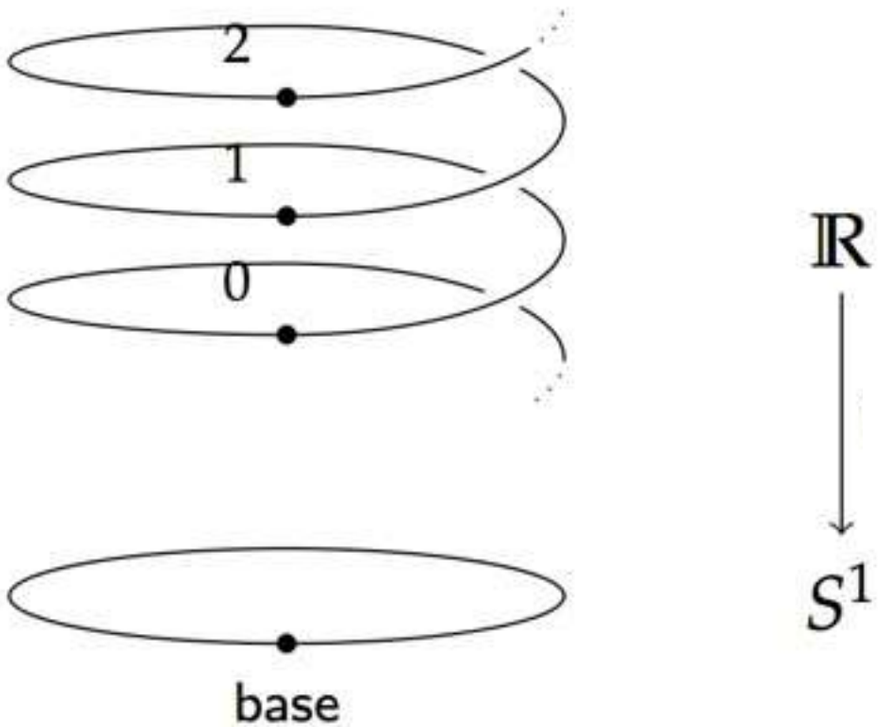


$w : \Omega(S^1) \to \mathbb{Z}$

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

lifting is functorial

Universal Cover



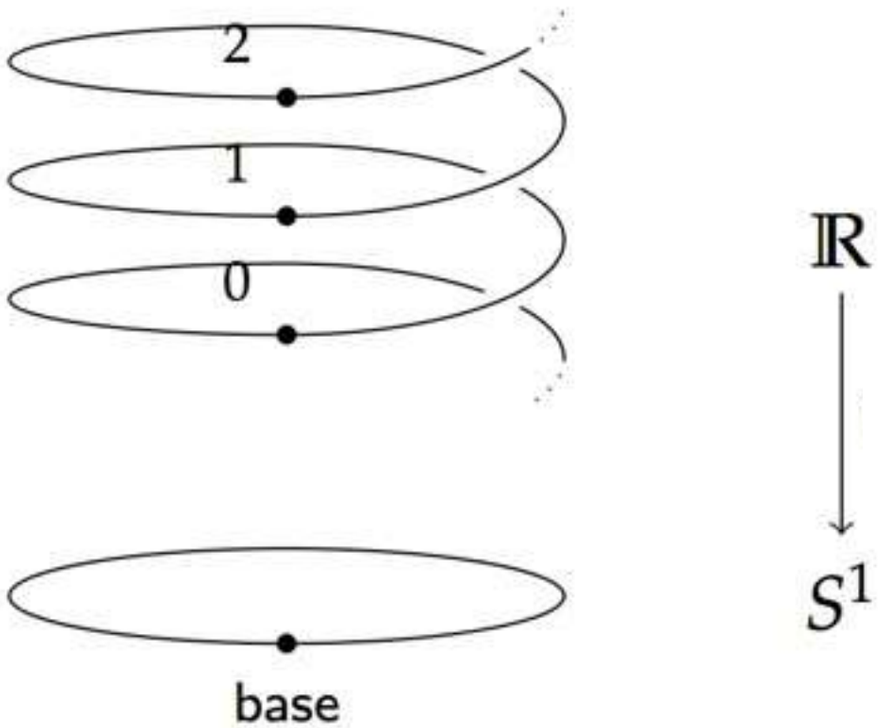
$w : \Omega(S^1) \to \mathbb{Z}$

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

lifting is functorial

lifting `loop` adds 1

Universal Cover



$w : \Omega(S^1) \to \mathbb{Z}$

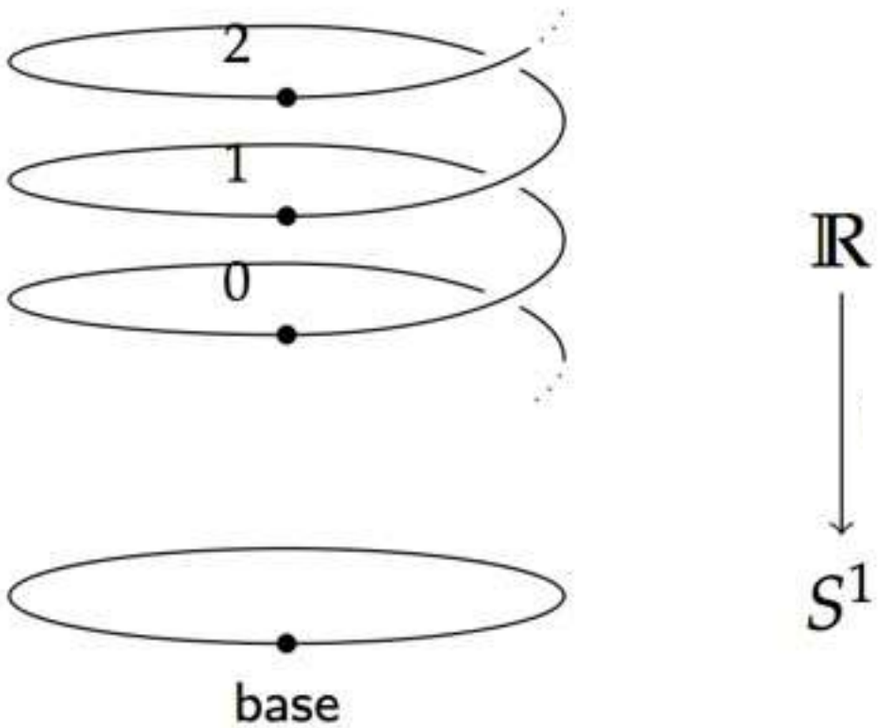
defined by **lifting** a loop to the cover, and giving the other endpoint of 0

lifting is functorial

lifting `loop` adds 1

lifting $\texttt{loop}^{-1}$ subtracts 1

Universal Cover



$w : \Omega(S^1) \to \mathbb{Z}$

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

lifting is functorial

lifting `loop` adds 1

lifting $\texttt{loop}^{-1}$ subtracts 1

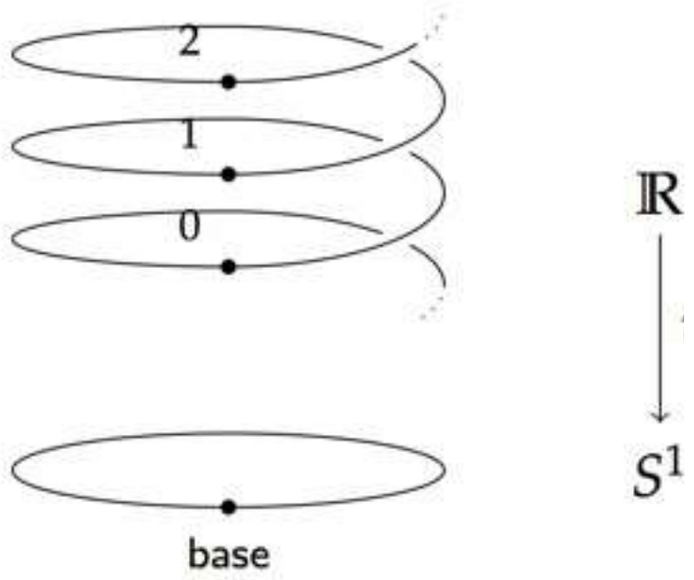
Example:

$w(\texttt{loop} \circ \texttt{loop}^{-1})$
$= 0 + 1 - 1$
$= 0$

Fibration = Family of types

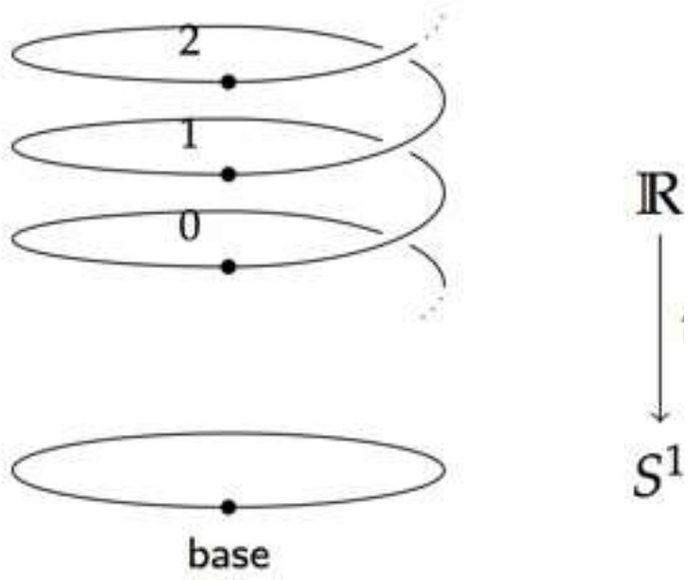
Fibration (classically):

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



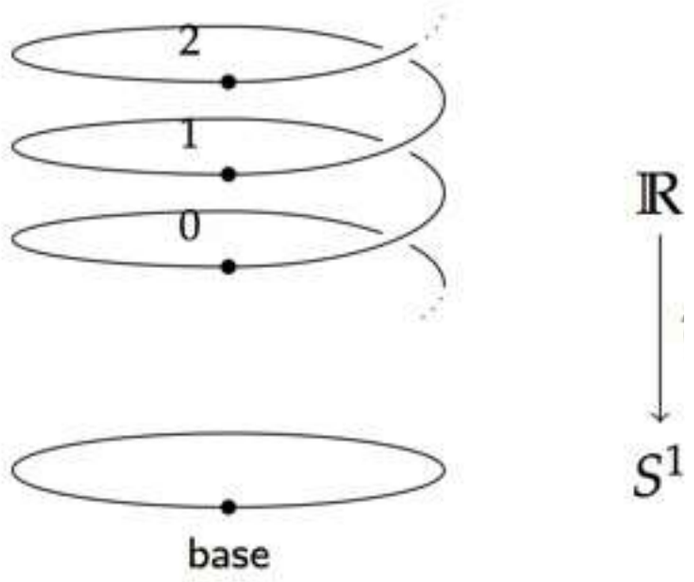
Family of types $(E(x))_{x:B}$

- *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 $p: E \to B$ 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 $(E(x))_{x:B}$

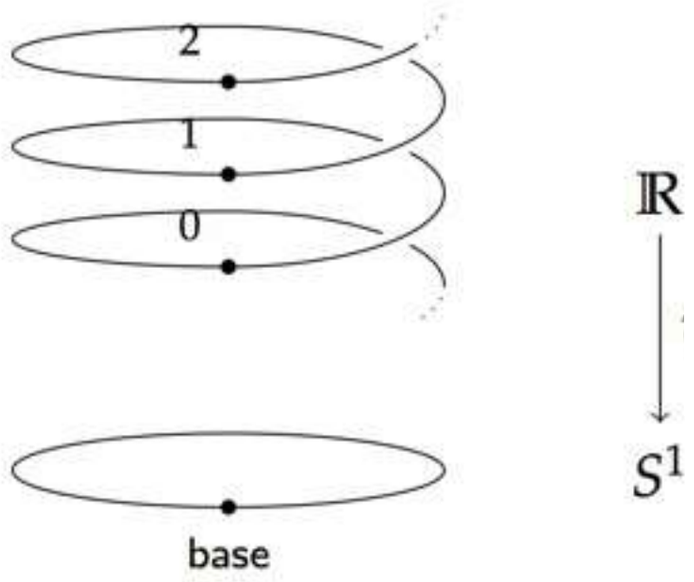
$p^{-1}(b)$

- *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 $p: E \to B$ 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 $(E(x))_{x:B}$

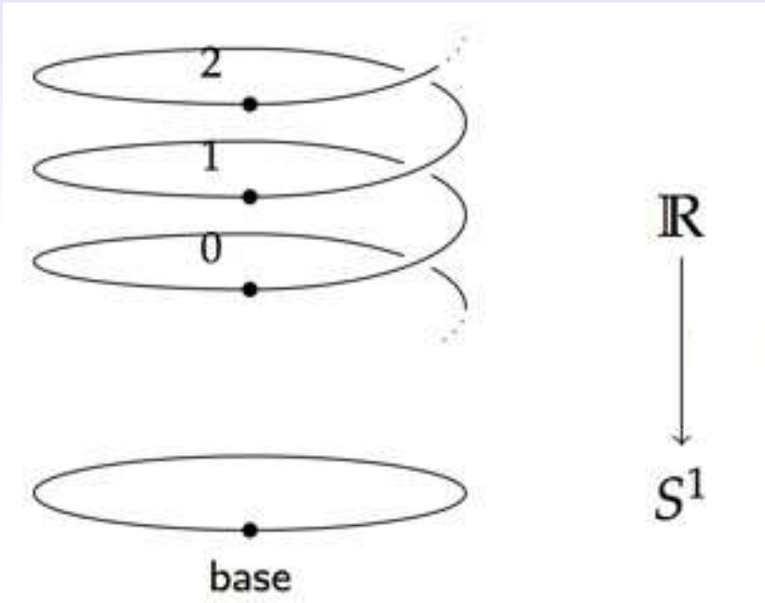
- *Fibers:* $E(b)$ is a type for all $b:B$
- *transport:* equivalence $E(b_1) \simeq E(b_2)$ for all $p: b_1 =_B b_2$

$p^{-1}(b)$

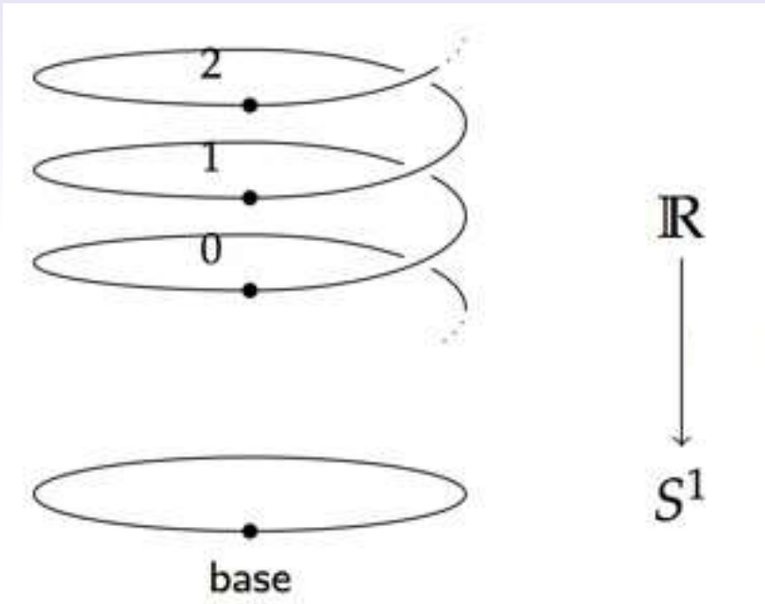
sends $e \in E(x)$ to other endpoint of lifting of p

Universal Cover

family of types $(\mathtt{Cover}(x))_{x:S1}$



Universal Cover

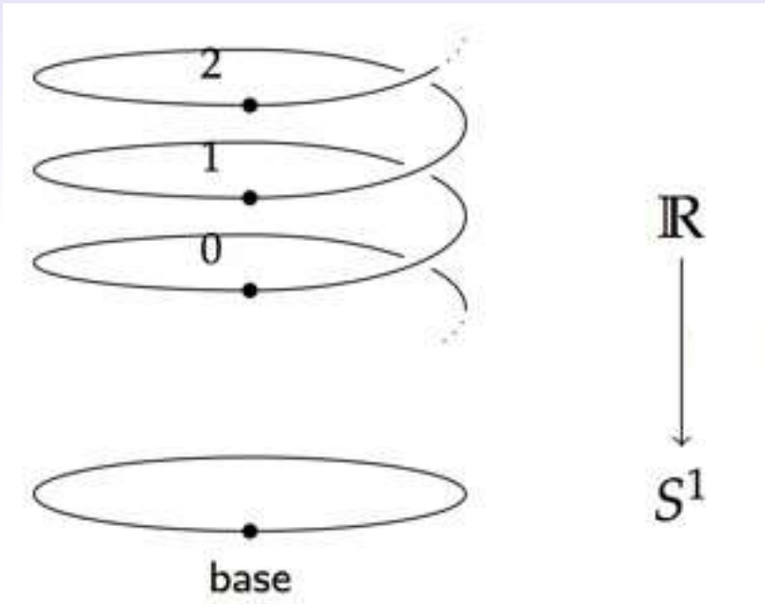


family of types $(\mathtt{Cover}(x))_{x:S1}$

By circle recursion, it suffices to give

- Fiber over $\mathtt{base}$: $\mathbb{Z}$
- Equivalence $\mathbb{Z} \simeq \mathbb{Z}$ as lifting of $\mathtt{loop}$: $\mathtt{successor}$

Universal Cover



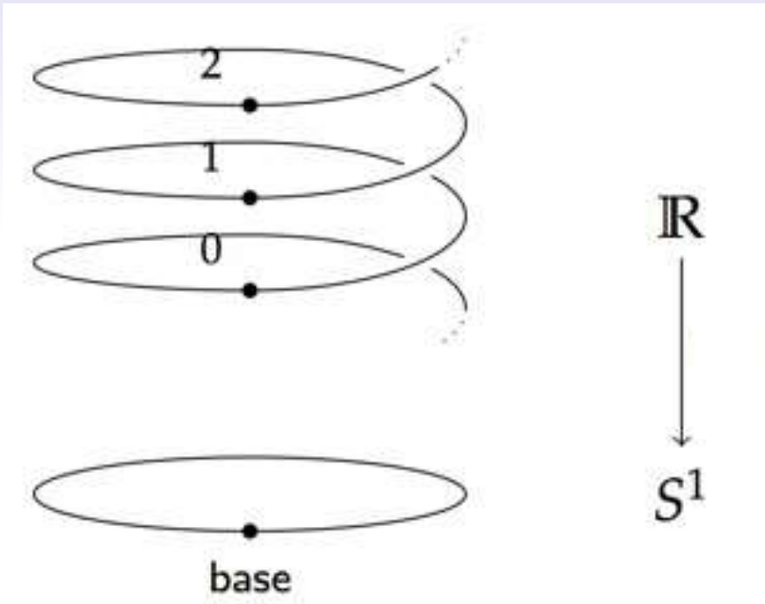
family of types $(\mathtt{Cover}(x))_{x:S1}$

By circle recursion, it suffices to give

- Fiber over $\mathtt{base}$: $\mathbb{Z}$
- Equivalence $\mathbb{Z} \simeq \mathbb{Z}$ as lifting of $\mathtt{loop}$: $\mathtt{successor}$

uses *univalence*

Universal Cover



family of types $(\mathtt{Cover}(x))_{x:S1}$

By circle recursion, it suffices to give

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

Defining equations:

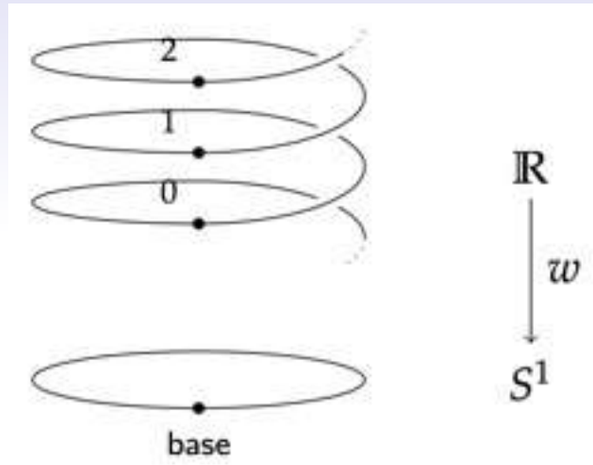
$$\mathtt{Cover}(\mathtt{base}) := \mathbb{Z}$$

$$\mathtt{transport}^{\mathtt{Cover}}(\mathtt{loop}) := \mathtt{successor}$$

Winding number

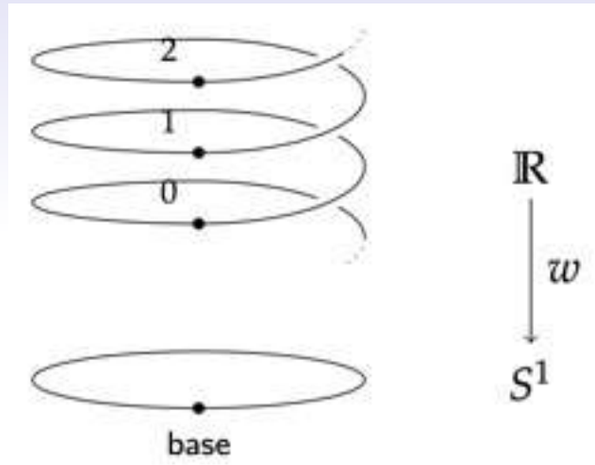
$w : \Omega(S^1) \to \mathbb{Z}$

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



lift p to cover, starting at 0

Winding number



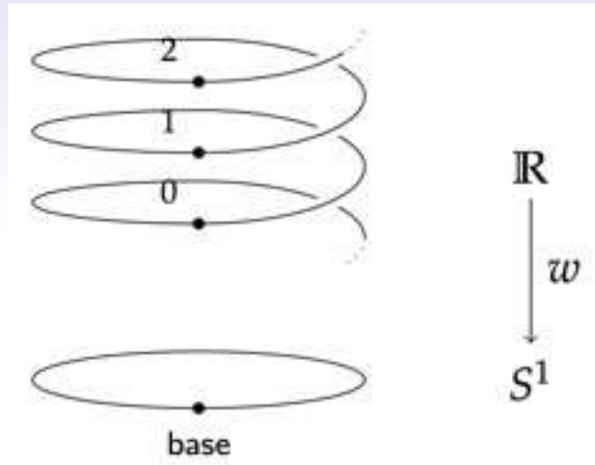
$w : \Omega(S^1) \to \mathbb{Z}$

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

lift p to cover, starting at 0

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

Winding number



$w : \Omega(S^1) \to \mathbb{Z}$

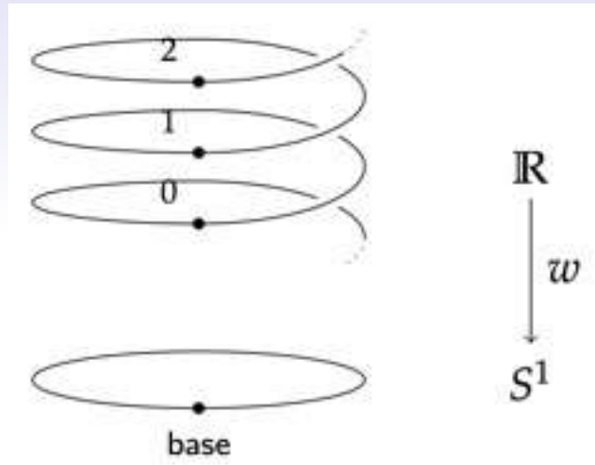
$w(p) = \text{transport}_{\text{Cover}}(p, 0)$

lift p to cover, starting at 0

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

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

Winding number



$w : \Omega(S^1) \to \mathbb{Z}$

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

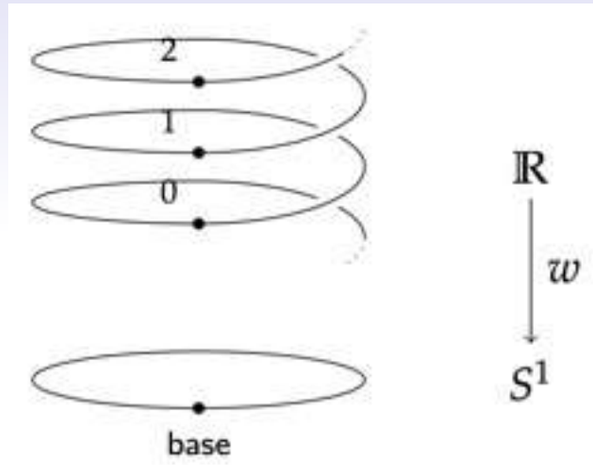
lift p to cover, starting at 0

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

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

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

Winding number



$w : \Omega(S^1) \to \mathbb{Z}$

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

lift p to cover, starting at 0

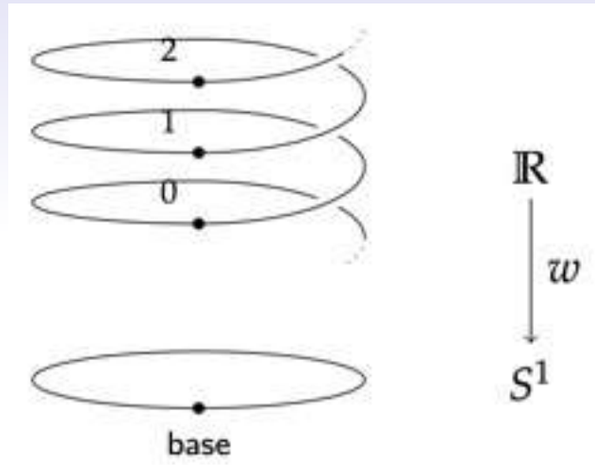
$w(\text{loop}^{-1} \circ \text{loop})$

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

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

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

Winding number



$w : \Omega(S^1) \to \mathbb{Z}$

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

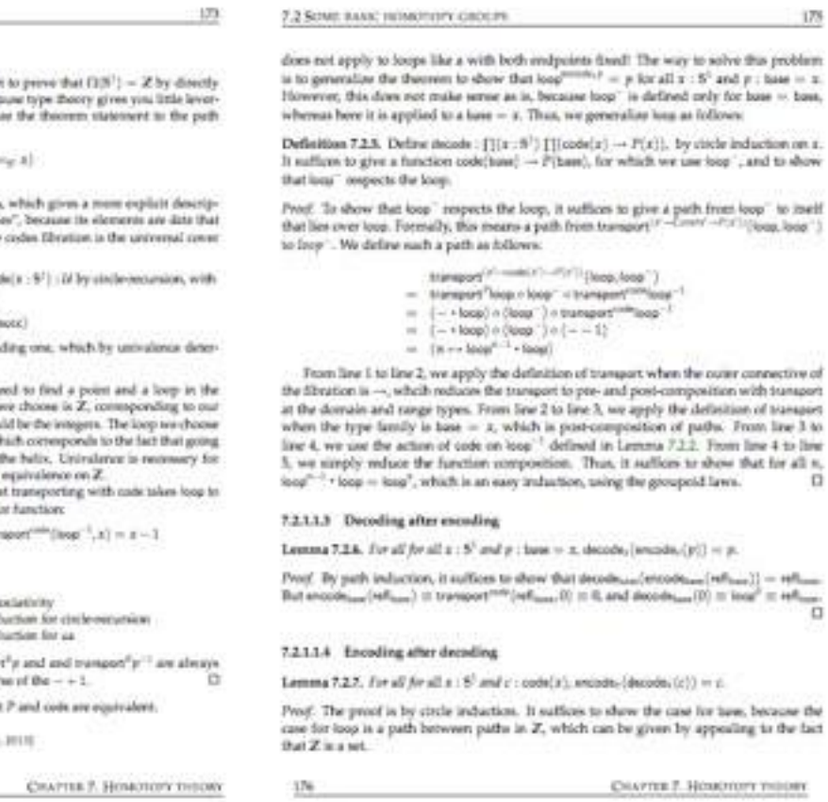
lift p to cover, starting at 0

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

Fundamental group of the circle

The book

Computer-checked



7.2 Some basic homotopy groups 173

7.2.1.1 Encode/decode proof

By definition, $\Omega(\mathbb{S}^1)$ is base $=_{\mathbb{S}^1}$ base. If we attempt to prove that $\Omega(\mathbb{S}^1) = \mathbb{Z}$ 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:

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

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 $\mathbb{S}^1$). Define $\mathsf{code}(x : \mathbb{S}^1) : \mathcal{U}$ by circle-recursion, with

$$\mathsf{code}(\mathsf{base}) := \mathbb{Z}$$
$$\mathsf{code}(\mathsf{loop}) := \mathsf{ua}(\mathsf{succ})$$

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 $\mathcal{U}$.

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 $\mathcal{U}$, 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. $\mathsf{transport}^{\mathsf{code}(x)}(\mathsf{loop}, z) = z + 1$ and $\mathsf{transport}^{\mathsf{code}(x)}(\mathsf{loop}^{-1}, z) = z - 1$.

Proof. For the first, we calculate as follows:

$$\begin{aligned}
&\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}, z) \\
=\ &\mathsf{transport}^{A \to A}(\langle \mathsf{code}\ \mathsf{loop} \rangle_1, z) && \text{associativity} \\
=\ &\mathsf{transport}^{A \to A}(\mathsf{ua}(\mathsf{succ}), z) && \text{reduction for circle-recursion} \\
=\ &z + 1 && \text{reduction for ua}
\end{aligned}$$

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

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

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

Definition 7.2.3. Define $\mathsf{encode} : \prod_{(x:\mathbb{S}^1)} (P(x) \to \mathsf{code}(x))$ by

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

(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, $\mathsf{transport}^{\mathsf{code}}(\mathsf{loop}, x)$ is $- + 1$ and $\mathsf{transport}^{\mathsf{code}}\mathsf{loop}^{-1} x$ is $x - 1$. Further, transport is functorial (chapter 2), so $\mathsf{transport}^{\mathsf{code}}\mathsf{loop} \cdot \mathsf{loop}$ is $(\mathsf{transport}^{\mathsf{code}}\mathsf{loop}) \circ (\mathsf{transport}^{\mathsf{code}}\mathsf{loop})$, etc. Thus, when p is a composition like

$$\mathsf{loop} \cdot \mathsf{loop}^{-1} \cdot \mathsf{loop} \cdot \ldots$$

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

$$(- + 1) \circ (- - 1) \circ (- + 1) \circ \ldots$$

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 $\mathsf{encode}' := \mathsf{encode}_{\mathsf{base}}$ has type $\mathsf{base} = \mathsf{base} \to \mathbb{Z}$, which will be one half of the equivalence between $\mathsf{base} = \mathsf{base}$ and $\mathbb{Z}$.

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

Definition 7.2.4. Define $\mathsf{loop}^{-} : \mathbb{Z} \to \mathsf{base} = \mathsf{base}$ by

$$\mathsf{loop}^n = \begin{cases} \mathsf{loop} \cdot \mathsf{loop} \cdot \ldots \cdot \mathsf{loop}\ (n \text{ times}) & \text{for positive } n \\ \mathsf{loop}^{-1} \cdot \mathsf{loop}^{-1} \cdot \ldots \cdot \mathsf{loop}^{-1}\ (n \text{ times}) & \text{for negative } n \\ \mathsf{refl} & \text{for } 0 \end{cases}$$

Since what we want overall is an equivalence between $\mathsf{base} = \mathsf{base}$ 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 $\mathsf{loop}^{\mathsf{encode}\, p} = p$ for all p. We would like to apply path induction, but path induction

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 $\mathsf{loop}^{\mathsf{encode}_x\, p} = p$ for all $x : \mathbb{S}^1$ and $p : \mathsf{base} = x$. However, this does not make sense as is, because loop^{-} is defined only for base = base, whereas here it is applied to a base = x. Thus, we generalize loop as follows:

Definition 7.2.5. Define $\mathsf{decode} : \prod_{(x:\mathbb{S}^1)} \prod_{(c:\mathsf{code}(x))} P(x)$, by circle induction on x. It suffices to give a function $\mathsf{code}(\mathsf{base}) \to P(\mathsf{base})$, 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 $\mathsf{transport}^{x \mapsto \mathsf{code}(x) \to P(x)}(\mathsf{loop}, \mathsf{loop}^{-})$ to loop^{-}. We define such a path as follows:

$$\begin{aligned}
&\mathsf{transport}^{x \mapsto \mathsf{code}(x) \to P(x)}(\mathsf{loop}, \mathsf{loop}^{-}) \\
=\ &\mathsf{transport}^{\mathsf{base} = \mathsf{base}}(\mathsf{loop}, \mathsf{loop}^{-} \circ \mathsf{transport}^{\mathsf{code}}\mathsf{loop}^{-1}) \\
=\ &(- \cdot \mathsf{loop}) \circ (\mathsf{loop}^{-}) \circ \mathsf{transport}^{\mathsf{code}}\mathsf{loop}^{-1} \\
=\ &(- \cdot \mathsf{loop}) \circ (\mathsf{loop}^{-}) \circ (- - 1) \\
=\ &(n \mapsto \mathsf{loop}^{n-1} \cdot \mathsf{loop})
\end{aligned}$$

From line 1 to line 2, we apply the definition of transport when the outer connective of the fibration is $\to$, 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 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, $\mathsf{loop}^{n-1} \cdot \mathsf{loop} = \mathsf{loop}^{n}$, which is an easy induction, using the groupoid laws. □

7.2.1.1.3 Decoding after encoding

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

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

7.2.1.1.4 Encoding after decoding

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

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

$$\mathsf{encode}(\mathsf{loop}^n) = n$$

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

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

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

- The cases for negatives are analogous.

□

7.2.1.1.5 Tying it all together

Theorem 7.2.8. *There is a family of equivalences $\prod_{(x:\mathbb{S}^1)} \prod_{(P(x))} (\mathsf{code}(x))$.*

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. □

Instantiating at base gives

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

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

Corollary 7.2.10. $\pi_1(\mathbb{S}^1) = \mathbb{Z}$ *if* $k = 1$ *and* 1 *otherwise.*

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


```
Cover : S¹ → Type
Cover x = S¹-rec Int (ua 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))
           (βloop/rec Int (ua succEquiv)) ⟩
  transport (λ x → x) (ua succEquiv)
    =⟨ type-β _ ⟩
  succ ∎

transport-Cover-!loop : Path (transport Cover (! loop)) pred
transport-Cover-!loop =
  transport Cover (! loop)
    =⟨ transport-ap-assoc Cover (! loop) ⟩
  transport (λ x → x) (ap Cover (! loop))
    =⟨ ap (transport (λ x → x)) (ap-! Cover loop)⟩
  transport (λ x → x) (! (ap Cover loop))
    =⟨ ap  (λ y → transport (λ x → x) (! y))
           (βloop/rec Int (ua succEquiv)) ⟩
  transport (λ x → x) (! (ua succEquiv))
    =⟨ ap (transport (λ x → x)) (!-ua succEquiv) ⟩
  transport (λ x → x) (ua (!equiv succEquiv))
    =⟨ type-β _ ⟩
  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

loop^ : Int → Path base base
loop^ zero     = id
loop^ (pos one)  = loop
loop^ (pos (S n)) = loop · loop^ (pos n)
loop^ (neg one)  = ! loop
loop^ (neg (S n)) = ! loop · loop^ (neg n)

loop^-preserves-pred
  : (n : Int) → Path (loop^ (pred n)) (! loop · loop^ n)
loop^-preserves-pred (pos one) = ! (!-inv-l loop)
loop^-preserves-pred (pos (S y)) =
    (--assoc (! loop) loop (loop^ (pos y)))
  ∙ (ap (λ x → x · loop^ (pos y)) (!-inv-l loop))
  ∙ (--unit-l (loop^ (pos y)))
loop^-preserves-pred zero = id
loop^-preserves-pred (neg one) = id
loop^-preserves-pred (neg (S y)) = id

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
  abstract -- prevent Agda from normalizing
    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
          ≡⟨ loop^
          ≡ transport Cover (! loop)
          =⟨ i≡ (λ y → transport-Path-right loop (loop^ (transport Cover (! loop) y))) ⟩
            (λ p → loop · p)
          ≡⟨ loop^
          ≡ transport Cover (! loop)
          =⟨ i≡ (λ y → ap (λ x' → loop · loop^ x') (ap≡ transport-Cover-(!loop))) ⟩
            (λ p → loop · p)
          ≡⟨ loop^
          ≡ pred
          =⟨ id ⟩
          (λ n → loop · (loop^ (pred n)))
          =⟨ i≡ (λ y → move-left-l _ loop (loop^ y) (loop^-preserves-pred y)) ⟩
          (λ n → loop^ n)
          ∎)
```

```
abstract -- prevent Agda from normalizing
  encode-loop^ : (n : Int) → Path (encode (loop^ n)) n
  encode-loop^ zero = id
  encode-loop^ (pos one) = ap≡ transport-Cover-loop
  encode-loop^ (pos (S n)) =
    encode (loop^ (pos (S n)))
      =⟨ id ⟩
    transport Cover (loop · loop^ (pos n)) zero
      =⟨ ap≡ (transport→ Cover loop (loop^ (pos n))) ⟩
    transport  (Cover loop
               (transport Cover (loop^ (pos n)) zero)
      =⟨ ap≡ transport-Cover-loop ⟩
    succ (transport Cover (loop^ (pos n)) zero)
      =⟨ id ⟩
    succ (encode (loop^ (pos n)))
      =⟨ ap succ (encode-loop^ (pos n)) ⟩
    succ (pos n) ∎
  encode-loop^ (neg one) = ap≡ transport-Cover-!loop
  encode-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 (encode-loop^ (neg n)) ⟩
    pred (neg n) ∎

encode-decode  : (x : S¹) → (c : Cover x)
               → Path (encode (decode[x] c)) c
encode-decode (x) = S¹-induction
  (λ (x : S¹) →  (c : Cover x)
                 → Path (encode[x] (decode[x] c)) c)
  encode-loop^ (i≡ (λ x' → fst (use-level (use-level (use-level hSet-Int _ _) _ _)))) x

decode-encode  : (x : S¹) (a : Path base x)
               → Path (decode (encode x)) 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-loop^)

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

π[S¹]-is-Int : π One S¹ base = Int
π[S¹]-is-Int = UnTrunc.path _ _ hSet-Int · ap (Trunc (tl 0)) Ω[S¹]-is-Int
```

Outline

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

2. The Hopf fibration

3. Connectedness and Freudenthal Suspension

The Hopf fibration

The Hopf fibration is a fibration with

- base $\mathbb{S}^2$
- fiber $\mathbb{S}^1$
- total space $\mathbb{S}^3$

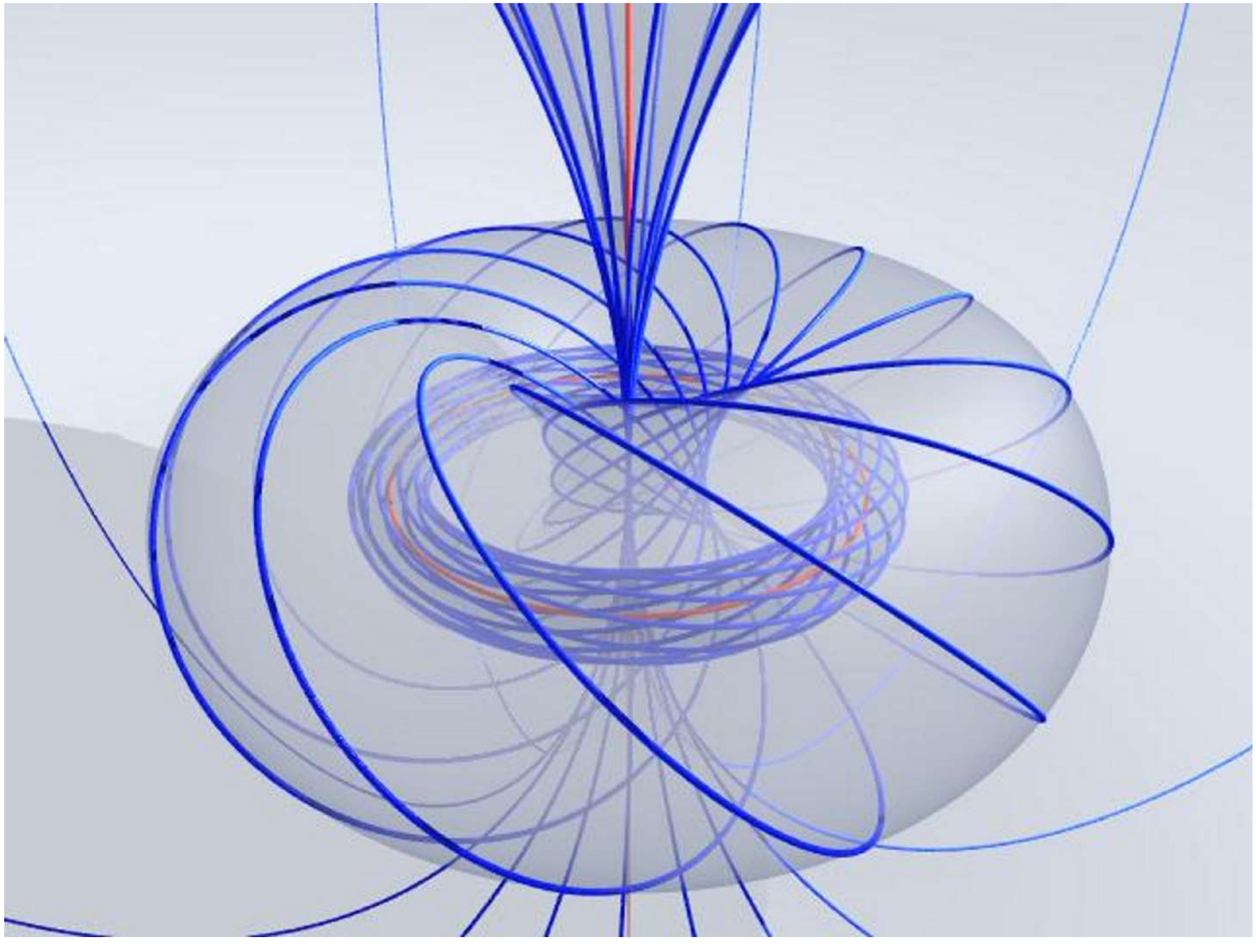
The Hopf fibration

The Hopf fibration is a fibration with

- base $\mathbb{S}^2$
- fiber $\mathbb{S}^1$
- total space $\mathbb{S}^3$

The Hopf fibration is a family of circles, parametrized by $\mathbb{S}^2$ and whose "union" is $\mathbb{S}^3$.

Picture



©Benoît R. Kloeckner CC-BY-NC

The spheres

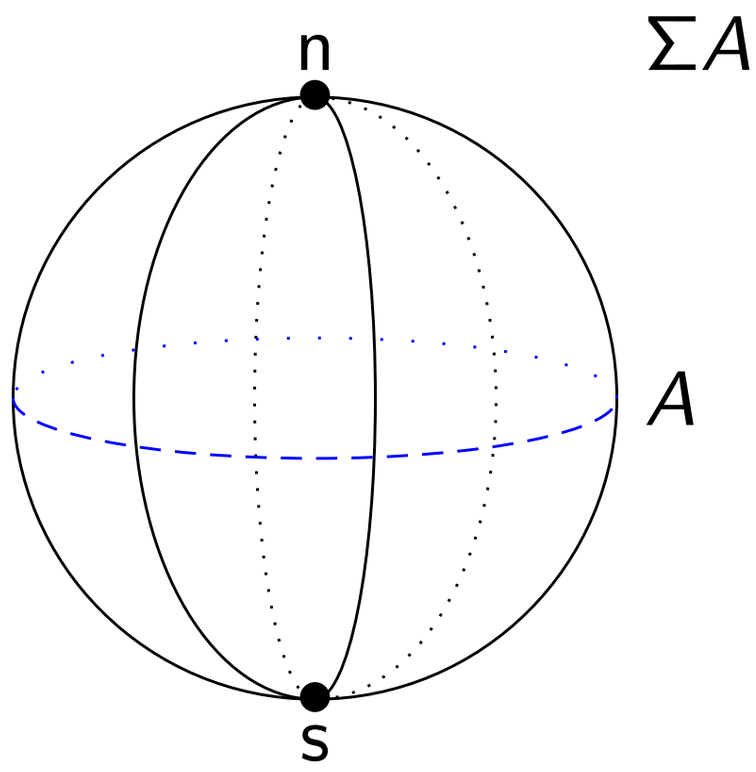
Definition

The suspension of a space A (denoted ΣA) is generated by

- Two points $\mathsf{n}, \mathsf{s} : \Sigma A$
- For every $a : A$, a path $\mathsf{m}(a) : \mathsf{n} =_{\Sigma A} \mathsf{s}$

Definition

$$\mathbb{S}^{n+1} := \Sigma \mathbb{S}^n$$



Fibrations over $\mathbb{S}^2$

A fibration over $\mathbb{S}^2$ is given by

- a space A (over n)

Fibrations over $\mathbb{S}^2$

A fibration over $\mathbb{S}^2$ is given by

- a space A (over n)
- a space B (over s)

Fibrations over $\mathbb{S}^2$

A fibration over $\mathbb{S}^2$ is given by

- a space A (over n)
- a space B (over s)
- a "circle of equivalences" between A and B (over m)
  $\iff$ a function $e : \mathbb{S}^1 \to (A \simeq B)$
  $\iff$ for every $x : \mathbb{S}^1$, an equivalence $e_x : A \simeq B$

The Hopf fibration in HoTT

A fibration over $\mathbb{S}^2$ with fiber $\mathbb{S}^1$ and total space $\mathbb{S}^3$?

The Hopf fibration in HoTT

A fibration over $\mathbb{S}^2$ with fiber $\mathbb{S}^1$ and total space $\mathbb{S}^3$?

- $\mathbb{S}^1$ over n
- $\mathbb{S}^1$ over s
- for $x : \mathbb{S}^1$, the equivalence $e_x : \mathbb{S}^1 \simeq \mathbb{S}^1$ is the "rotation of angle" x

The Hopf fibration in HoTT

A fibration over $\mathbb{S}^2$ with fiber $\mathbb{S}^1$ and total space $\mathbb{S}^3$?

- $\mathbb{S}^1$ over n
- $\mathbb{S}^1$ over s
- for $x : \mathbb{S}^1$, the equivalence $e_x : \mathbb{S}^1 \simeq \mathbb{S}^1$ is the "rotation of angle" x

Left to do:

- Define the rotation of angle x
- Prove that the total space is $\mathbb{S}^3$

Rotations of $\mathbb{S}^1$

We want

$$e : \mathbb{S}^1 \to (\mathbb{S}^1 \simeq \mathbb{S}^1)$$

Rotations of $\mathbb{S}^1$

We want

$$e : \mathbb{S}^1 \to (\mathbb{S}^1 \simeq \mathbb{S}^1)$$

By definition of $\mathbb{S}^1$, we need

- an equivalence $e_{\mathsf{base}} : \mathbb{S}^1 \simeq \mathbb{S}^1$
- a homotopy $e(\mathsf{loop}) : e_{\mathsf{base}} = e_{\mathsf{base}}$

Rotations of $\mathbb{S}^1$

We want

$$e : \mathbb{S}^1 \to (\mathbb{S}^1 \simeq \mathbb{S}^1)$$

By definition of $\mathbb{S}^1$, we need

- an equivalence $\mathrm{id}_{\mathbb{S}^1} : \mathbb{S}^1 \simeq \mathbb{S}^1$
- a homotopy $e(\mathrm{loop}) : \mathrm{id}_{\mathbb{S}^1} = \mathrm{id}_{\mathbb{S}^1}$

Rotations of $\mathbb{S}^1$

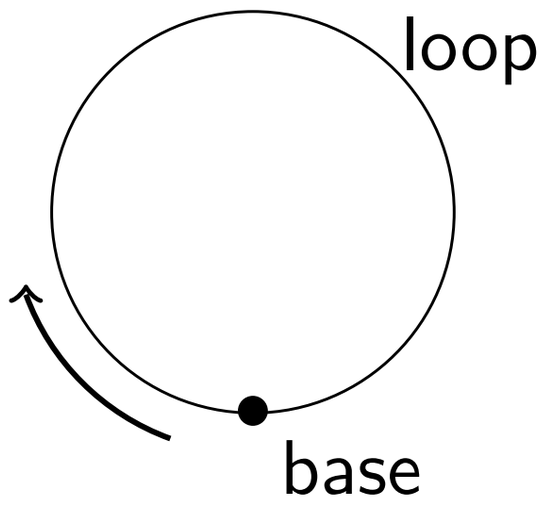
We want

$$e : \mathbb{S}^1 \to (\mathbb{S}^1 \simeq \mathbb{S}^1)$$

By definition of $\mathbb{S}^1$, we need

- an equivalence $\mathrm{id}_{\mathbb{S}^1} : \mathbb{S}^1 \simeq \mathbb{S}^1$
- a homotopy $e(\mathsf{loop}) : \mathrm{id}_{\mathbb{S}^1} = \mathrm{id}_{\mathbb{S}^1}$

$e(\mathsf{loop})$ is the homotopy "turning once around the circle".



Homotopy turning once around the circle

A homotopy $\mathrm{id}_{\mathbb{S}^1} = \mathrm{id}_{\mathbb{S}^1}$ $\iff$ for every $x : \mathbb{S}^1$, a path $x = x$

Homotopy turning once around the circle

A homotopy $\mathrm{id}_{\mathbb{S}^1} = \mathrm{id}_{\mathbb{S}^1}$ $\iff$ for every $x : \mathbb{S}^1$, a path $x = x$

We need:

- a path

$$p : \mathsf{base} = \mathsf{base}$$

- a (2-dimensional) path

$$q : p \cdot \mathsf{loop} = \mathsf{loop} \cdot p$$

Homotopy turning once around the circle

A homotopy $\mathrm{id}_{\mathbb{S}^1} = \mathrm{id}_{\mathbb{S}^1}$ $\iff$ for every $x : \mathbb{S}^1$, a path $x = x$

We need:

- a path

$$\mathsf{loop} : \mathsf{base} = \mathsf{base}$$

- a (2-dimensional) path

$$q : \mathsf{loop} \cdot \mathsf{loop} = \mathsf{loop} \cdot \mathsf{loop}$$

Homotopy turning once around the circle

A homotopy $\mathrm{id}_{\mathbb{S}^1} = \mathrm{id}_{\mathbb{S}^1}$ $\iff$ for every $x : \mathbb{S}^1$, a path $x = x$

We need:

- a path

$$\mathsf{loop} : \mathsf{base} = \mathsf{base}$$

- a (2-dimensional) path

$$\mathsf{refl}_{\mathsf{loop} \cdot \mathsf{loop}} : \mathsf{loop} \cdot \mathsf{loop} = \mathsf{loop} \cdot \mathsf{loop}$$

Total space

We just constructed a fibration with

- base $\mathbb{S}^2$
- fiber $\mathbb{S}^1$

What is the total space?

Homotopy pushouts

Given a span

$$Y \xleftarrow{f} X \xrightarrow{g} Z$$

Definition

The homotopy pushout $Y \sqcup^X Z$ is the space generated by

- For all $y : Y$, a point $\mathsf{l}(y) : Y \sqcup^X Z$
- For all $z : Z$, a point $\mathsf{r}(z) : Y \sqcup^X Z$
- For all $x : X$, a path $\mathsf{g}(x) : \mathsf{l}(f(x)) = \mathsf{r}(g(x))$

The suspension of A is the homotopy pushout of

$$1 \longleftarrow A \longrightarrow 1$$

Total space

By gluing/descent/flattening, the total space is the homotopy pushout of:

$$\mathbb{S}^1 \xleftarrow{\mathsf{e}} \mathbb{S}^1 \times \mathbb{S}^1 \xrightarrow{p_2} \mathbb{S}^1$$

Total space

By gluing/descent/flattening, the total space is the homotopy pushout of:

$$\mathbb{S}^1 \xleftarrow{\mathsf{e}} \mathbb{S}^1 \times \mathbb{S}^1 \xrightarrow{p_2} \mathbb{S}^1$$

This span is equivalent to the following:

$$\mathbb{S}^1 \xleftarrow{p_1} \mathbb{S}^1 \times \mathbb{S}^1 \xrightarrow{p_2} \mathbb{S}^1$$

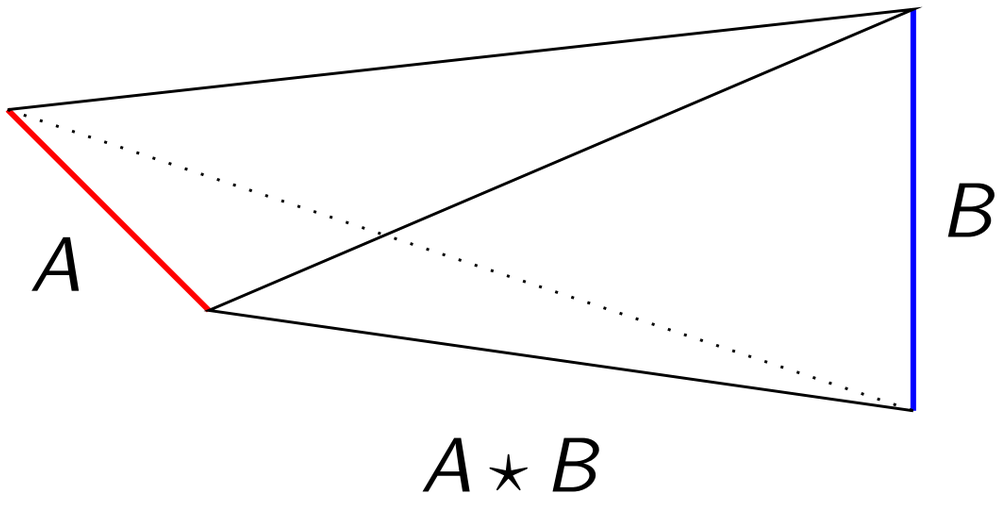
whose total space is $\mathbb{S}^1 \star \mathbb{S}^1$

Join

Definition

The join of A and B is the homotopy pushout of

$$A \xleftarrow{p_1} A \times B \xrightarrow{p_2} B$$

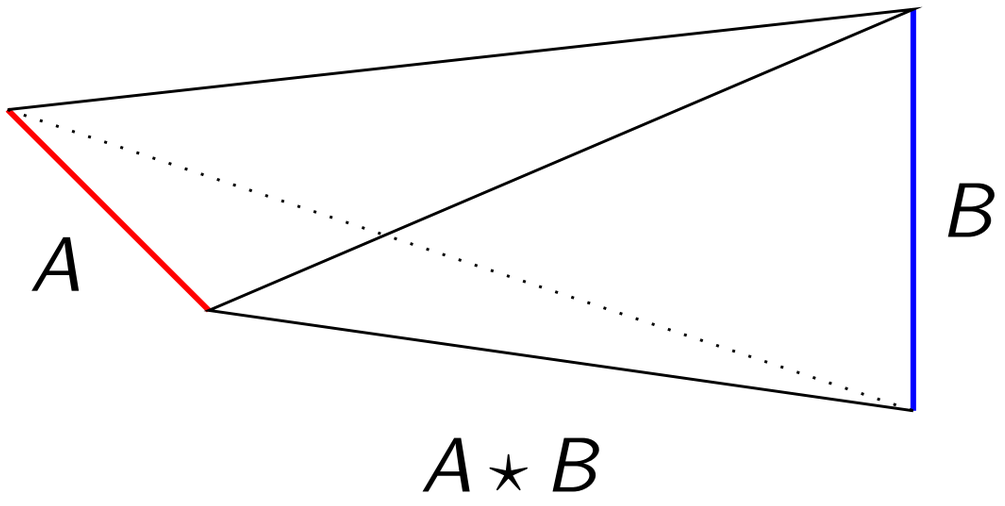


Join

Definition

The join of A and B is the homotopy pushout of

$$A \xleftarrow{p_1} A \times B \xrightarrow{p_2} B$$



$A \star B$

We have

$$\mathbb{S}^0 \star A = \Sigma A$$

$$(A \star B) \star C = A \star (B \star C)$$

Total space

$$\begin{aligned}
\mathbb{S}^1 \star \mathbb{S}^1 &= (\Sigma\mathbb{S}^0) \star \mathbb{S}^1 \\
&= (\mathbb{S}^0 \star \mathbb{S}^0) \star \mathbb{S}^1 \\
&= \mathbb{S}^0 \star (\mathbb{S}^0 \star \mathbb{S}^1) \\
&= \Sigma(\Sigma\mathbb{S}^1) \\
&= \mathbb{S}^3
\end{aligned}$$

Total space

$$\begin{aligned}
\mathbb{S}^1 \star \mathbb{S}^1 &= (\Sigma\mathbb{S}^0) \star \mathbb{S}^1 \\
&= (\mathbb{S}^0 \star \mathbb{S}^0) \star \mathbb{S}^1 \\
&= \mathbb{S}^0 \star (\mathbb{S}^0 \star \mathbb{S}^1) \\
&= \Sigma(\Sigma\mathbb{S}^1) \\
&= \mathbb{S}^3
\end{aligned}$$

We have the Hopf fibration in homotopy type theory.

Long exact sequence

Long exact sequence of homotopy groups of the Hopf fibration:

$$\vdots$$

$$\pi_4(\mathbb{S}^1) \longrightarrow \pi_4(\mathbb{S}^3) \longrightarrow \pi_4(\mathbb{S}^2)$$

$$\pi_3(\mathbb{S}^1) \longrightarrow \pi_3(\mathbb{S}^3) \longrightarrow \pi_3(\mathbb{S}^2)$$

$$\pi_2(\mathbb{S}^1) \longrightarrow \pi_2(\mathbb{S}^3) \longrightarrow \pi_2(\mathbb{S}^2)$$

$$\pi_1(\mathbb{S}^1) \longrightarrow \pi_1(\mathbb{S}^3) \longrightarrow \pi_1(\mathbb{S}^2)$$

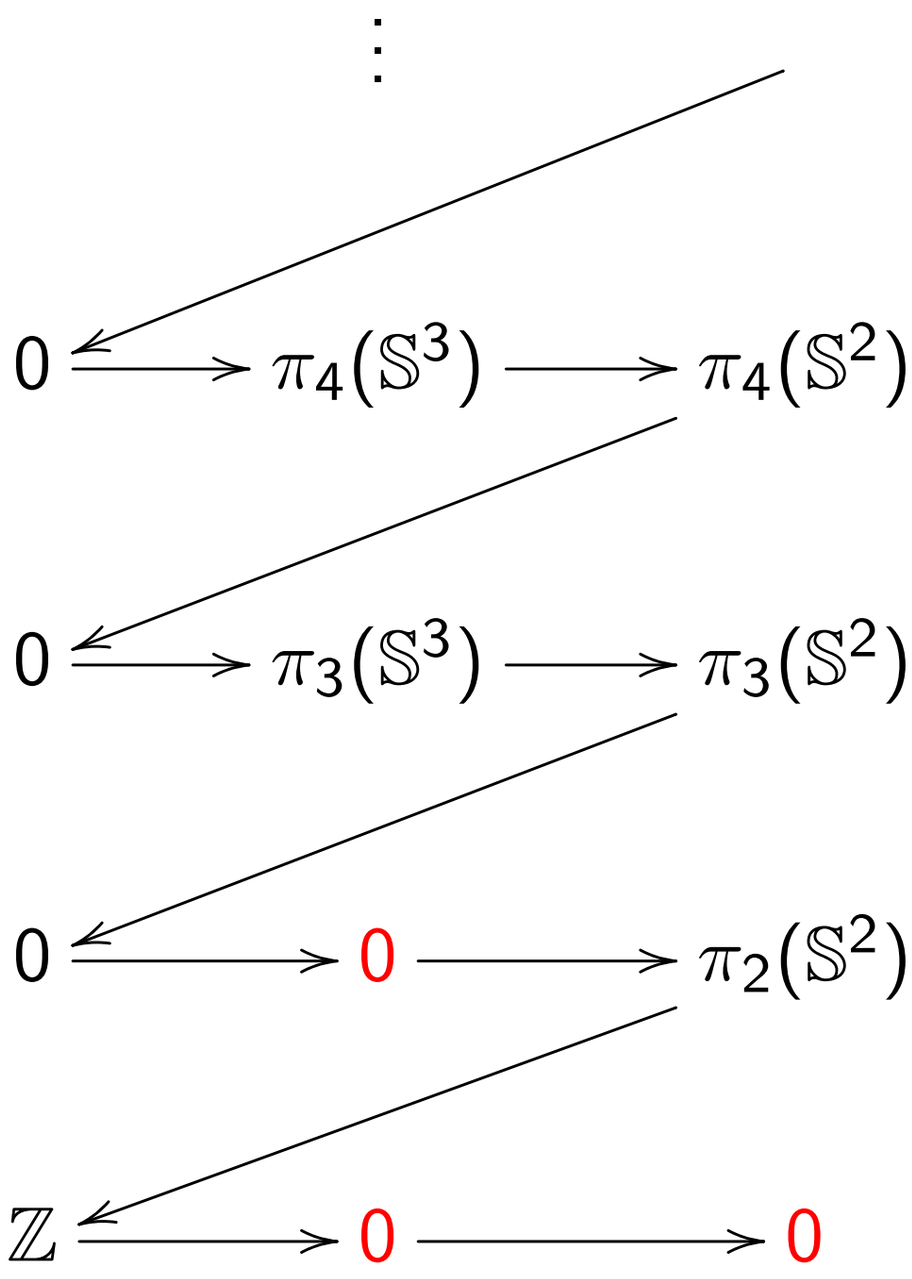
Long exact sequence

Long exact sequence of homotopy groups of the Hopf fibration:

$$
\begin{array}{ccccc}
 & & \vdots & & \\
 & \swarrow & & & \\
0 & \longrightarrow & \pi_4(\mathbb{S}^3) & \longrightarrow & \pi_4(\mathbb{S}^2) \\
 & \swarrow & & & \\
0 & \longrightarrow & \pi_3(\mathbb{S}^3) & \longrightarrow & \pi_3(\mathbb{S}^2) \\
 & \swarrow & & & \\
0 & \longrightarrow & \pi_2(\mathbb{S}^3) & \longrightarrow & \pi_2(\mathbb{S}^2) \\
 & \swarrow & & & \\
\mathbb{Z} & \longrightarrow & \pi_1(\mathbb{S}^3) & \longrightarrow & \pi_1(\mathbb{S}^2)
\end{array}
$$

Long exact sequence

Long exact sequence of homotopy groups of the Hopf fibration:



Long exact sequence

Long exact sequence of homotopy groups of the Hopf fibration:

$$
\begin{array}{ccccc}
 & & \vdots & & \\
 & \swarrow & & & \\
0 & \longrightarrow & \pi_4(\mathbb{S}^3) & \xrightarrow{\sim} & \pi_4(\mathbb{S}^2) \\
 & \swarrow & & & \\
0 & \longrightarrow & \pi_3(\mathbb{S}^3) & \xrightarrow{\sim} & \pi_3(\mathbb{S}^2) \\
 & \swarrow & & & \\
0 & \longrightarrow & 0 & \longrightarrow & \pi_2(\mathbb{S}^2) \\
 & & \sim & & \\
 & \swarrow & & & \\
\mathbb{Z} & \longrightarrow & 0 & \longrightarrow & 0
\end{array}
$$

Homotopy groups

Theorem

We have

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

$$\pi_k(\mathbb{S}^2) = \pi_k(\mathbb{S}^3) \text{ for } k \geq 3$$

In particular

Theorem

Assuming $\pi_3(\mathbb{S}^3) = \mathbb{Z}$

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

$\pi_4(\mathbb{S}^3)$

Theorem

There exists a natural number n such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$.

$\pi_4(\mathbb{S}^3)$

Theorem

There exists a natural number n such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$.

- Classical mathematics: cannot compute n, unless the proof is nice enough

$\pi_4(\mathbb{S}^3)$

Theorem

There exists a natural number n such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$.

- Classical mathematics: cannot compute n, unless the proof is nice enough
- Constructive mathematics: disallow the axiom of choice and excluded middle $\Longrightarrow$ every proof is nice enough

$\pi_4(\mathbb{S}^3)$

Theorem

There exists a natural number n such that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/n\mathbb{Z}$.

- Classical mathematics: cannot compute n, unless the proof is nice enough
- Constructive mathematics: disallow the axiom of choice and excluded middle $\implies$ every proof is nice enough

In this case we can compute the value of n and get 2*

*work in progress

Outline

1. $\pi_1(S^1) = \mathbb{Z}$
2. The Hopf fibration
3. **Connectedness and Freudenthal Suspension**

Part III: Freudenthal and friends

1. Truncatedness

2. Connectedness

3. Freudenthal Suspension Theorem

Truncatedness

Definition

A type X is n-truncated (or an n-type) if, by induction on $n \geq -2$:

- $n = -2$: if X is contractible, i.e. $X \simeq 1$;
- $n > -2$: if each path space $(x =_X x')$ of X is $(n-1)$-truncated.

Proposition

Suppose X is n-truncated, for $n \geq -1$. Then $\pi_k(X, x_0) \simeq 1$, for all $k > n$ and $x_0 : X$.

[In **Top** and **SSet**, the converse holds; but not in all classical settings, cf. Whitehead's theorem and hypercompleteness.]

Truncations

Definition

For any type X, and $n \geq -1$, the *n*-truncation $\tau_n X$ is the higher inductive type generated by:

- for $x : X$, an element $[x]_n : \tau_n X$;
- for $f : \mathbb{S}^{n+1} \to \tau_n X$, and $t : \mathbb{S}^{n+1}$, a path $f(t) = f(0)$.

Proposition

$\tau_n X$ is the free n-truncated type on X: any $f : X \to Y$, with Y n-truncated, factors uniquely through $\tau_n X$.

[Classically: iteratively glue cells on to X to kill homotopy in dimensions $> n$.]

Connectedness (of types)

Definition

X is *n-connected* if $\tau_{n+1}X$ is contractible.

Proposition

TFAE:

- *X is n-connected;*
- *every map from X to an n-type is constant;*
- *(when $n \geq 0$) $\pi_k(X, x_0) \simeq 1$, for all $k \leq n$ and $x_0 : X$.*

Connectedness (trivial low homotopy groups) is dual to truncatedness (trivial high homotopy groups).

Connectedness (of maps)

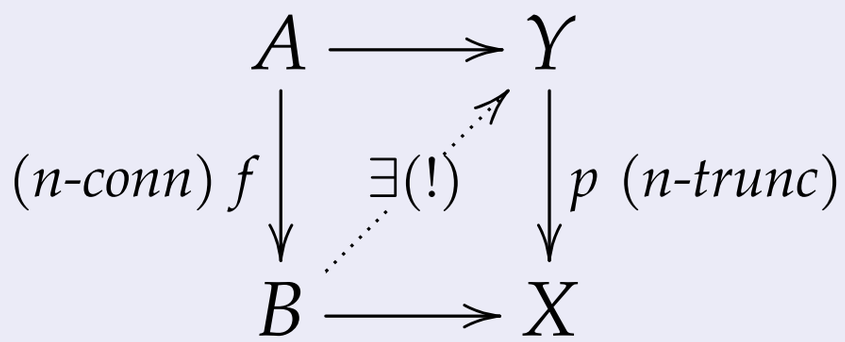
Definition

$f : A \to B$ is *n*-connected if each (homotopy) fiber $f^{-1}(b)$ is n-connected. (Warning: indexing conventions vary by ± 1.)

Proposition

TFAE:

- *f is n-connected;*
- *f is weakly (or strongly) orthogonal to maps with n-truncated fibers;*

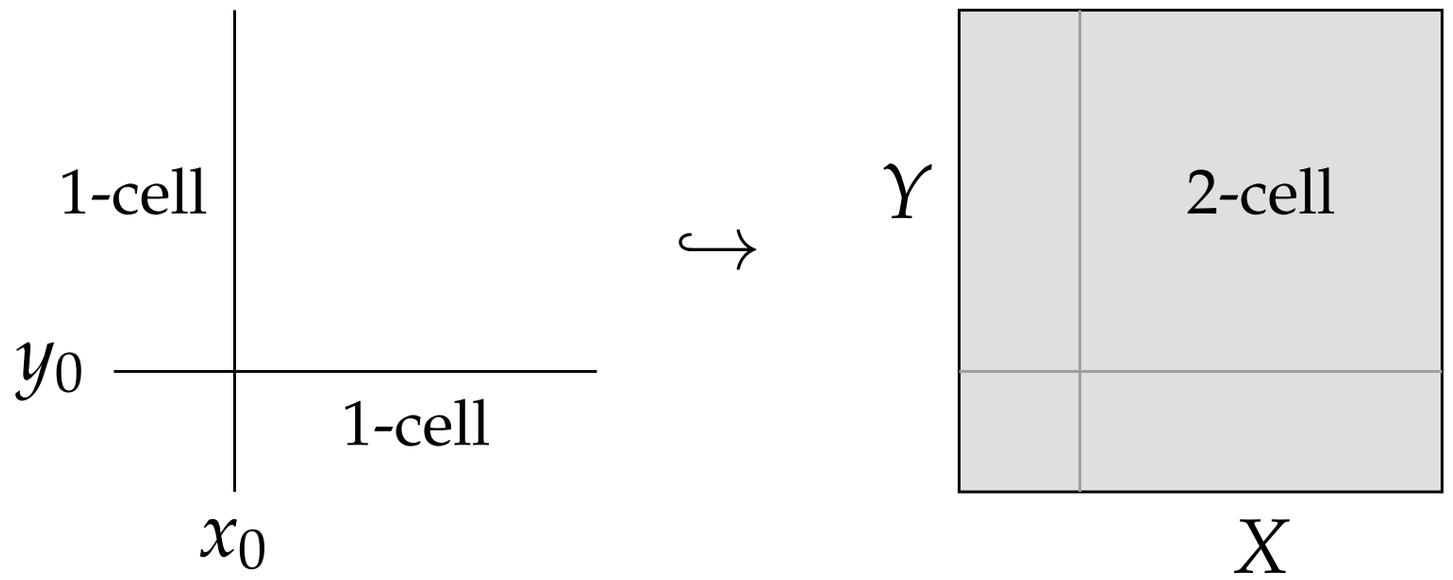


- *f is equivalent to the inclusion of A into some extension by cells of dimensions $> n$.*

Additivity of connectedness

Lemma (Wedge-product connectedness)

Suppose (X, x_0) *is* i*-connected,* (Y, y_0) *is* j*-connected. Then the inclusion* $X \sqcup_1 Y \hookrightarrow X \times Y$ *is* $(i+j)$*-connected.*



Type-theoretically: to define a function of two variables $f(x, y)$ into an $(i+j)$-type, enough to define in the cases $f(x_0, y)$ and $f(x, y_0)$, agreeing in the case $f(x_0, y_0)$.

Freudenthal

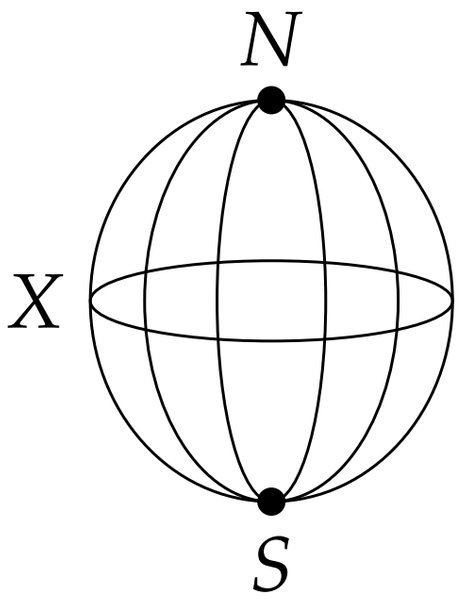
Definition

Recall: the suspension ΣX is generated by

- $N, S : \Sigma X$;
- for each $x : X$, a path $m(x) : N =_{\Sigma X} S$.

Theorem (Freudenthal Suspension Theorem)

Suppose (X, x_0) is n-connected. Then the canonical map $X \to \Omega(\Sigma X, N)$ is $2n$-connected.



Idea: want $X \to \Omega(\Sigma X, N)$ to be an equivalence. Generally (e.g. for $\Sigma\mathbb{S}^1 \simeq \mathbb{S}^2$) it isn't; but within a certain dimension range, it is.

Important application: stable homotopy groups of spheres.

Proof: weak Freudenthal

For now, prove a weaker statement. (Same approach, with more work, yields full FST.)

Theorem (Weak Freudenthal)

Suppose (X, x_0) *is n-connected. Then the canonical map* $\tau_{2n}(X) \to \tau_{2n}\Omega(\Sigma X, N)$ *is an equivalence.*

Proof.

Heuristic: to prove a result of the form $X \approx \Omega(Y, y_0)$, generalise X to a dependent type $\bar{X}_y$ over $y : Y$, with $\bar{X}_{y_0} \simeq X$, and prove $\bar{X}_y \approx (y_0 =_Y y)$ for all $y : Y$.

So: define type $\bar{X}_y$ depending on $y : \Sigma X$, and maps $\bar{m}_y : \bar{X}_y \to \tau_{2n}(N = y)$, using universal property of ΣX.

Weak Freudenthal, cont'd

Proof.

To give $\bar{X}_y$, $\bar{m}_y$ for all $y : \Sigma X$, need:

- types and maps $\bar{m}_N : \bar{X}_N \to \tau_{2n}(N = N)$, and $\bar{m}_S : \bar{X}_S \to \tau_{2n}(N = S)$;
- transport equivalences $\mathsf{transport}_{\bar{X}} m(x_1) : \bar{X}_N \to \bar{X}_S$, for each $x_1 : X$, commuting with $\bar{m}_N$, $\bar{m}_S$.

$$\text{over } S: \qquad \bar{m}_S := \tau_{2n}(m) : \tau_{2n}(X) \to \tau_{2n}(N = S)$$

$$\text{over } N: \quad \bar{m}_N := \tau_{2n}(x \mapsto m(x) \circ m(x_0)^{-1}) : \tau_{2n}(X) \to \tau_{2n}(N = N)$$

and over $m(x)$, need to define for each $x_1 : X$ the action $\mathsf{transport}_{\bar{X}}(m(x), -) : \bar{X}_N \to \bar{X}_S$.

Weak Freudenthal, cont'd

Proof.

…transport over $m(x_1)$: need to give, for each $x_1 : X$ and $z : \bar{X}_N = \tau_{2n}(X)$, some element of $\bar{X}_S = \tau_{2n}(X)$.

Since RHS is $2n$-truncated, may assume z is of form $[x_2]$, some $x_2 : X$. Also, by wedge-product connectedness lemma, enough to assume one of x_1, x_2 is x_0. So: when $x_1 = x_0$, return $[x_2]$. When $x_2 = x_0$, return $[x_1]$. (Check: when $x_1 = x_2 = x_0$, these agree)

(Roughly: defining a multiplication $X \times \tau_{2n}(X) \to \tau_{2n}(X)$, with x_0 as a two-sided unit.)

So: have $\bar{m}_y : \bar{X}_y \to (N = y)$, for all $y : \Sigma X$.

Define converse $\bar{n}_y : (N = y) \to \bar{X}_y$ by $n_y(p) := \mathsf{transport}_{\bar{X}}[x_0]$. Not hard to prove $\bar{m}, \bar{n}$ mutually inverse; so, each $\bar{m}_y$ is an equivalence, as desired. □

Consequences

From (weak) Freudenthal, immediately have:

Corollary (Homotopy groups of spheres stabilise)

$\pi_{n+k}(\mathbb{S}^n) \simeq \pi_{n+1+k}(\mathbb{S}^{n+1})$, *for* $n \geq k+2$.

In particular,

Corollary

$\pi_n(\mathbb{S}^n) \simeq \mathbb{Z}$, *for all* $n \geq 1$.

Proof.

- $n = 1$: by universal cover.
- $n = 2$: by LES of Hopf fibration.
- $n \geq 2$: by stabilisation. □

$\pi_k(S^n)$ 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]

More results

James construction

Refinement of Freudenthal: describes $\Omega(\Sigma X)$ precisely, via a filtration.

Theorem

Suppose (X, x_0) *is n-connected, for* $n \geq 0$. *There is a sequence*

$$1 \longrightarrow X \longrightarrow J_2(X) \longrightarrow J_3(X) \longrightarrow J_4(X) \longrightarrow \cdots$$

with the maps having respective connectivities $(n-1)$, $2n$, $(3n+1)$, *. . . , and such that* $J_\infty(X) := \varinjlim_n J_n(X) \simeq \Omega(\Sigma X)$.

Conceptually, $J_\infty(X)$ is the free monoid on X; as X is connected, this is the free group on X.

Blakers–Massey

Generalization of Freudenthal: describes path spaces in pushouts.

Theorem (Blakers–Massey theorem)

Suppose given maps f, g as below, with f i-connected, g j-connected.

$$\begin{array}{ccc} Z & \xrightarrow{g} & Y \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \mathsf{inr}} \\ X & \xrightarrow{\mathsf{inl}} & X \sqcup_Z Y \end{array}$$

Then for all $x : X$, $y : Y$, the canonical map $Z_{x,y} \to (\mathsf{inl}\ x = \mathsf{inr}\ y)$ is $(i + j)$-connected.

van Kampen

Another tool for pushouts of types:

Theorem (van Kampen theorem)

For any pointed maps $f : Z \to X$ and $g : Z \to Y$, with Z 0-connected, the fundamental group of the pushout of f and g is the amalgamated free product (pushout of groups) of $\pi_1(X)$ and $\pi_1(Y)$ over $\pi_1(Z)$:

$$\pi_1(X \sqcup_Z Y) \simeq \pi_1(X) *_{\pi_1(Z)} \pi_1(Y).$$

Can also be generalised to non-connected Z.

Covering spaces

The (beautiful) classical theory of covering spaces transfers straightforwardly. In particular:

Definition

A covering space of a connected type X is a dependent family of 0-types over X.

Theorem

Covering spaces of X correspond to sets with an action of $\pi_1(X)$.

Eilenberg–Mac Lane spaces; cohomology

Eilenberg–Mac Lane spaces of Abelian groups can be constructed as HIT's:

Theorem

For any (n-truncated) Abelian group G and natural number $n > 0$, there is a type $K(G, n)$ such that $\pi_n(K(G, n)) \simeq G$, and $\pi_n(K(G, n)) \simeq 1$ for $k \neq n$.

These (and other spectra) can be used to define cohomology of types.

Conclusion

We can do

computer-checked proofs

in **synthetic** homotopy theory

January 14, 2013

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

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

April 11, 2013

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

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

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