The Univalence Axiom

Chris Grossack (they/them)

Many thanks to Jake Park, both for the invitation and their patience while trading emails!

you can find the slides
at my blog:

grossack.site/tags/my-talks

On With The Show

§1 A quick review.

- dependent types
- identity types

§1 A quick review.

- dependent types
- identity types

logical content

§1 A "quick" review.

- dependent types
- identity types

$\Big\}$ → logical content

→ geometric content.

§1 A quick review.

- dependent types
- identity types

→ logical content

→ geometric content.

HoTT is about the interplay between logic & geometry

Dependent Types

Dependent Types

- associates a type $B(a)$ to each $a : A$

Dependent Types

- associates a type $B(a)$ to each $a : A$
- equivalently, $B : A \longrightarrow \mathcal{U}$

Dependent Types

- associates a type $B(a)$ to each $a : A$
- equivalently, $B : A \longrightarrow \mathcal{U}$

  ↳ here $\mathcal{U}$ is a <u>universe</u> of "small" types.

logically

logically

B is a proposition depending on $a : A$

logically

B is a proposition depending on $a : A$

eg

$B : \mathbb{N} \to \mathcal{U}$

$B(x) \triangleq x \geq 4$

eg $B : \mathbb{N} \to \mathcal{U}$

$B(x) \triangleq x \geq 4$

$\hookrightarrow$ $B(5)$ is the collection of proofs that $5 \geq 4$

eg

$B : \mathbb{N} \to \mathcal{U}$

$B(x) \triangleq x \geq 4$

↳ $B(5)$ is the collection of proofs that $5 \geq 4$

↳ $B(x)$ is nonempty iff $x \geq 4$ is provable.

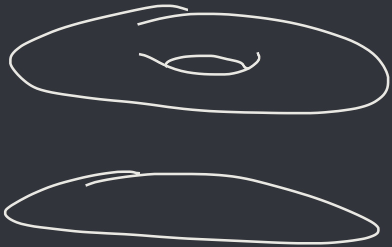
Geometrically

B is a bundle over A

Geometrically

B is a bundle over A

eg.

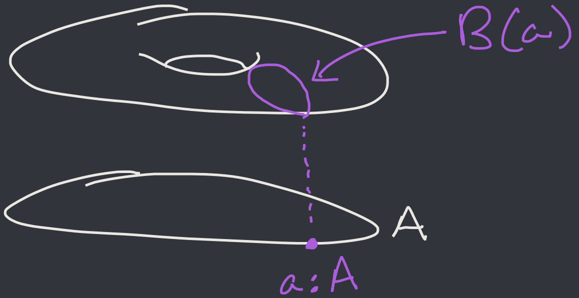


A

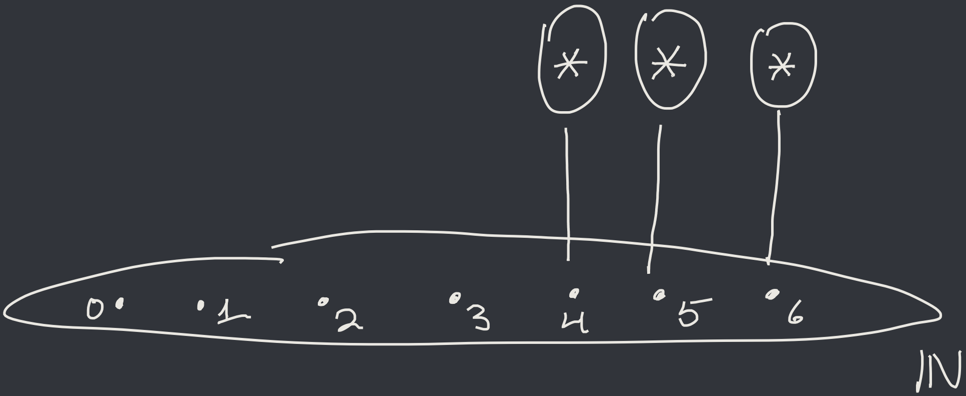
Geometrically

B is a bundle over A

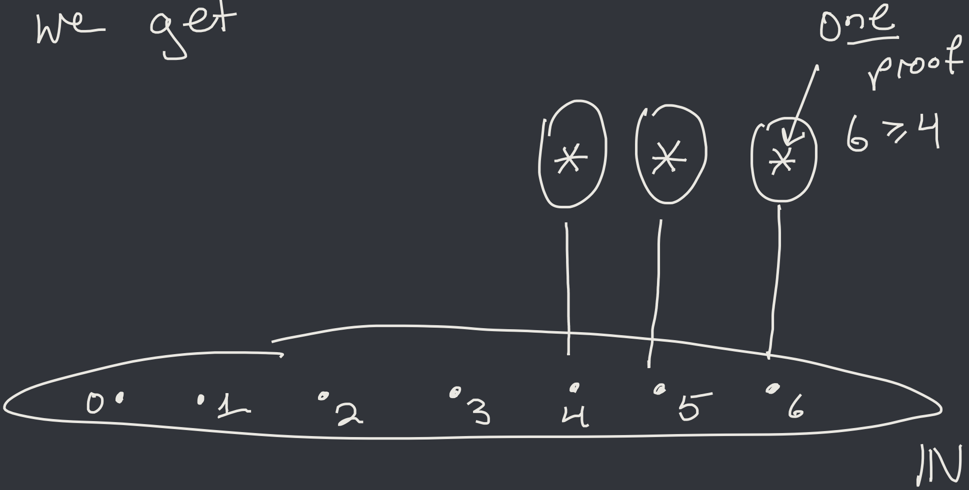
eg.



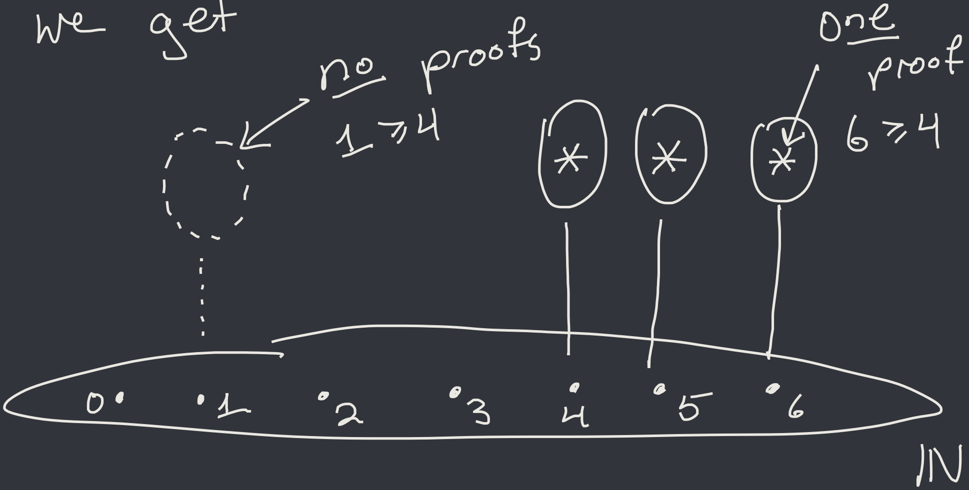
In the previous example,
we get



In the previous example,
we get



In the previous example, we get



Now, there are two natural constructions to consider:

Now, there are two natural constructions to consider:

- Σ-types

Now, there are two natural constructions to consider:

- Σ - types
  $\approx$ existential quantification

Now, there are two natural constructions to consider:

- Σ - types
  $\approx$ existential quantification
  $\approx$ total space

Now, there are two natural constructions to consider:

- Σ - types
  $\approx$ existential quantification
  $\approx$ total space
- Π - types

Now, there are two natural constructions to consider:

- Σ-types
  $\approx$ existential quantification
  $\approx$ total space

- Π-types
  $\approx$ universal quantification
  $\approx$ global sections

$$\sum_{x:A} B(x) \text{ "}=\text{" } \left\{ (x,b) \;\middle|\; \begin{array}{l} x:A \\ b:B(x) \end{array} \right\}$$

$$\sum_{x:A} B(x) \text{ "}=\text{" } \left\{ (x, b) \;\middle|\; \begin{array}{l} x : A \\ b : B(x) \end{array} \right\}$$

(not actually a set)

$$\sum_{x:A} B(x) \text{ `` } = \text{ '' } \left\{ (x, b) \;\middle|\; \begin{matrix} x : A \\ b : B(x) \end{matrix} \right\}$$

$$\sum_{x:A} B(x) \text{ “}=\text{” } \left\{ (x,b) \;\middle|\; \begin{array}{l} x:A \\ b:B(x) \end{array} \right\}$$

$$= \left\{ (x,p) \;\middle|\; \begin{array}{l} p \text{ a } \underline{\text{proof}} \\ \text{of } B(x) \end{array} \right\}$$

$$\sum_{x:A} B(x) \text{ "="} \left\{ (x,b) \;\middle|\; \begin{array}{l} x:A \\ b:B(x) \end{array} \right\}$$

$$= \left\{ (x,p) \;\middle|\; \begin{array}{c} p \text{ a } \underline{\text{proof}} \\ \text{of } B(x) \end{array} \right\}$$

So $\sum_{x:A} B(x)$ is nonempty

iff

$B(x)$ holds <u>for some</u> $x:A$

Note!

Note!

proof relevant

↳ a proof of $\sum_{x:A} B(x)$

necessarily furnishes witnesses

- $x_0 : A$ • $p : B(x_0)$

to the existential quantifier

geometrically?

geometrically?

if $B(x)$ is the fibre over $x : A$, then

$\sum_{x:A} B(x)$ glues the fibres together, viewing them as one space

- $B : A \to \mathcal{U}$ $\approx$ Sheaf on A (stalks)

- $\sum_{x:A} B(x)$ $\approx$ Étale Space over A

$$\begin{array}{cc} \sum_{x:A} B(x) & (x,b) \\ \downarrow & \downarrow \\ A & x \end{array}$$

$$\prod_{x:A} B(x) \quad \text{“}=\text{”} \quad \left\{ f : (x:A) \longrightarrow B(x) \right\}$$

$$\prod_{x:A} B(x) \quad \text{“}=\text{”} \left\{ f : (x:A) \longrightarrow B(x) \right\}$$

$$= \left\{ f \ \text{picking a point from each fibre} \right\}$$

$$\prod_{x:A} B(x) \quad \text{“}=\text{”} \left\{ f : (x:A) \longrightarrow B(x) \right\}$$

$$= \left\{ f \ \ \text{picking a point from each fibre} \right\}$$

$$= \left\{ f : x \longmapsto \text{a proof of } B(x) \right\}$$

$$\prod_{x:A} B(x) \;\text{“}=\text{”}\; \left\{ f : (x:A) \longrightarrow B(x) \right\}$$

$$= \left\{ f \;\; \begin{array}{l}\text{picking a point} \\ \text{from each fibre}\end{array} \right\}$$

$$= \left\{ f : x \longmapsto \begin{array}{l}\text{a proof} \\ \text{of } B(x)\end{array} \right\}$$

So $\prod_{x:A} B(x)$ holds iff

$B(x)$ holds for every x !

So $\prod_{x:A} B(x)$ holds iff

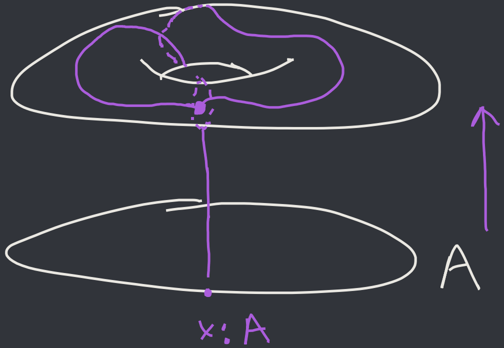
$B(x)$ holds for every x!

↳ again, carries more info than a lowly $\forall$.

↳ gives a uniform assignment of proofs $f(x) : B(x)$ to each $x : A$.

geometrically?

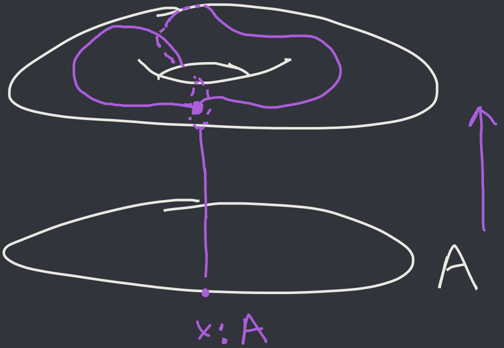
geometrically?



$$f : \prod_{x : A} B(x)$$

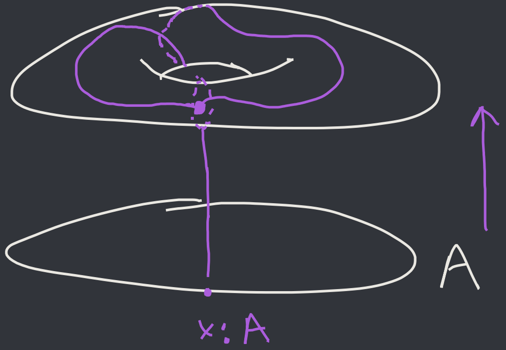
geometrically?

picking a point in each fibre gives a <u>section</u> of $pr : \sum_{x:A} B(x) \to A$



$f : \prod_{x:A} B(x)$

geometrically?



picking a point in each fibre gives a section of $pr : \sum_{x:A} B(x) \to A$

(Automatically continuous!)

ex (*)

find a term of type

$$A \times \prod_{a:A} P(a) \longrightarrow \sum_{a:A} P(a)$$

↳ a souped up version of

$$A \neq \emptyset \wedge \forall a . P(a) \Longrightarrow \exists a . P(a)$$

Identity Types

Identity Types (path types)

Identity Types (path types)

- remember
  - types ≈ propositions
  - programs ≈ proofs

Identity Types (path types)

- remember - types $\approx$ propositions
  - programs $\approx$ proofs

- if $a, b : A$, we can ask if $a = b$. so it's a proposition! So it's a type!

- if $a, b : A$ then $a = b$ is a type

- if $a, b : A$ then $a = b$ is a type

↳ $p : a = b$ is a proof of equality.

- if $a, b : A$ then $a = b$ is a type

↳ $p : a = b$ is a proof of equality.

↳ $\mathrm{refl}_a : a = a$ is the canonical witness of reflexivity

↳ $\mathrm{refl}_a : a = a$ is the canonical witness of reflexivity

⚠ it is consistent with ¬Univalence that these are the only proofs of equality (cf. Axiom K)

↳ this is (imo) part of why identity types / path induction / etc are confusing.

↳ thankfully there has been a lot of great research lately on computational content for these "nonstandard" paths! (eg: "Computing with univalence" [Licata])

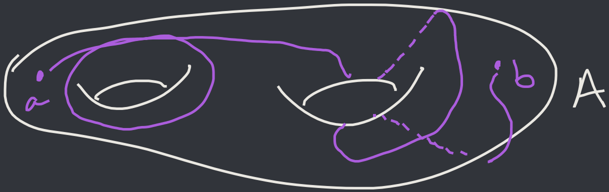
geometrically?

geometrically?

$p : a = b$ is a path from a to b in A.

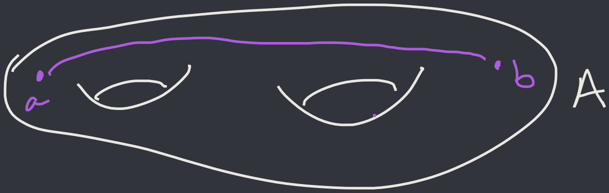
geometrically?

$p : a = b$ is a path from a to b in A.



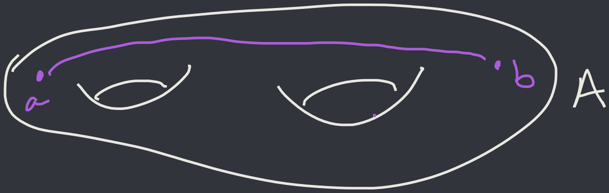
geometrically?

$p : a = b$ is a path from a to b in A.



geometrically?

$p : a = b$ is a path from a to b in A.



↳ realizes the intuitive idea in algebraic topology that we can contract two points along a path.

geometrically?

$p : a = b$ is a path from a to b in A.

↳ realizes the intuitive idea in algebraic topology that we can contract two points along a path.

But why stop there?

But why stop there?

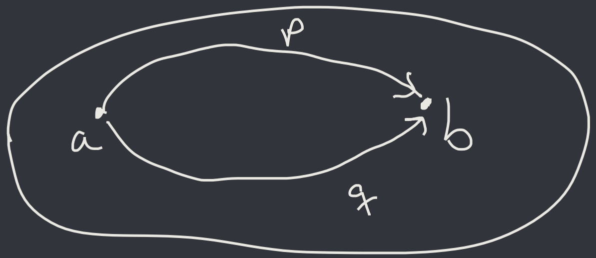
↳ if $p, q : a = b$, what is $H : p = q$?

But why stop there?

$\hookrightarrow$ if $p, q : a = b$, what is $H : p = q$?

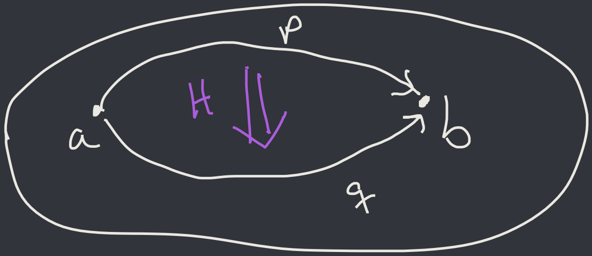
But why stop there?

↳ if $p, q : a = b$, what is $H : p = q$?



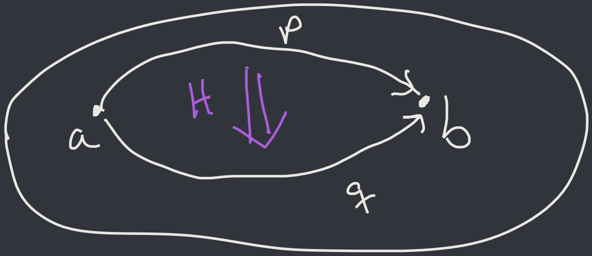
But why stop there?

↳ if $p, q : a = b$, what is $H : p = q$?



But why stop there?

↳ if $p, q : a = b$, what is $H : p = q$?



H is a <u>homotopy</u> from p to q.

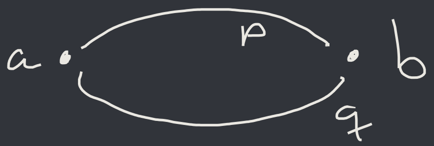
↳ "fills in the circle"

But why stop <u>there</u>!?

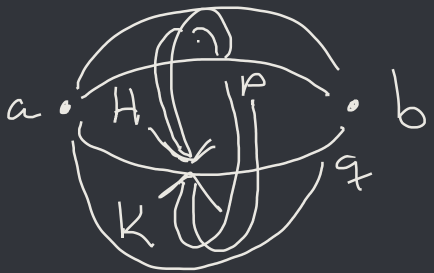
But why stop there!?

a •   • b

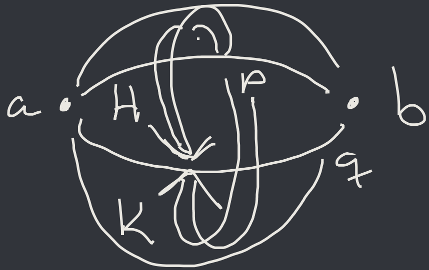
But why stop <u>there</u>!?



But why stop there!?



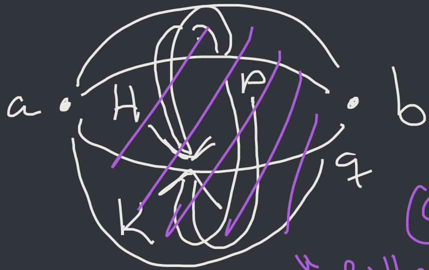
But why stop ~~there~~ there!?



H, K : $p = q$

↳ looks like S^2

But why stop ~~there~~ !?



H, K : p = q

↳ looks like S^2

⊙ : H = K

"fills in the ball"

(now it's D^3)

But why stop there!?

But why stop there!?

A
because I can't draw
4D pictures

But why stop there!?

A (better)

don't!

But why stop there!?

A (better) don't!

Types are ∞-groupoids!

Types are ∞-groupoids!

§2

What the @#!% does <u>that</u> mean??.

recall, a <u>groupoid</u> is a category where every arrow is an isomorphism

<u>eg</u>

↳ this is an algebraic representation of all the ≤ 1 dimensional information about a (nice) topological space.

↳ this is an algebraic representation of all the ≤ 1 dimensional information about a (nice) topological space.

↳ eg: if X is a space,

$\Pi_1 X$ is the category with
- objects = points of X
- $\hom(a,b) = \{\text{paths } a \to b \text{ in } X\}$

↳ Similarly, given a groupoid, we can build a (nice) space

- add a 0-cell for each object
- add a 1-cell for each arrow

↳ Similarly, given a groupoid, we can build a (nice) space

- add a 0-cell for each object
- add a 1-cell for each arrow

eg

↳ Similarly, given a groupoid, we can build a (nice) space

- add a 0-cell for each object
- add a 1-cell for each arrow

eg

is this a picture of the groupoid? or it's associated space?

↳ Similarly, given a groupoid, we can build a (nice) space

- add a 0-cell for each object
- add a 1-cell for each arrow

eg

is this a picture of the groupoid? or it's associated space?
(it's both)

By analogy:

↳ a <u>set</u> is a <u>discrete</u> groupoid
(only identity arrows)

By analogy:

↳ a <u>set</u> is a <u>discrete</u> groupoid
(only identity arrows)
(we call this a 0-groupoid)

By analogy:

↳ a <u>set</u> is a 0-groupoid

↳ a <u>groupoid</u> is a <u>1-groupoid</u>

(we allow 1-dimensional arrows)

By analogy:

↳ a <u>set</u> is a 0-groupoid

↳ a <u>groupoid</u> is a <u>1-groupoid</u>

↳ a <u>2-groupoid</u> has 2-isomorphisms between arrows

By analogy:

↳ a set is a 0-groupoid

↳ a groupoid is a 1-groupoid

↳ a 2-groupoid has 2-isomorphisms between arrows

eg

$$\begin{tikzcd} a \arrow[r, bend left, "p"] \arrow[r, bend right, "q"'] \arrow[r, phantom, "\Downarrow H"] & b \end{tikzcd}$$

By analogy:

↳ a <u>set</u> is a 0-groupoid

↳ a <u>groupoid</u> is a <u>1-groupoid</u>

↳ a <u>2-groupoid</u> has

2-isomorphisms between arrows

eg
$$\begin{tikzcd} a \arrow[r, bend left, "p"{name=U}] \arrow[r, bend right, "q"'{name=D}] & b \arrow[from=U, to=D, phantom, "\Downarrow H"] \end{tikzcd}$$

(look familiar?)

An ∞-groupoid allows $n+1$-isomorphisms between n-isomorphisms for all n

An ∞-groupoid allows
$n+1$ - isomorphisms between
n - isomorphisms <u>for all n</u>

↳ Compare with a
<u>cell complex</u> in topology.

But ∞-groupoids are <u>algebraic</u> gadgets...

But ∞-groupoids are
<u>algebraic</u> gadgets...

↳ so there should be a "free" ∞-groupoid on some generators...

But ∞-groupoids are <u>algebraic</u> gadgets...

↳ so there should be a "free" ∞-groupoid on some generators...

§3

equivalences.

logically, we would like

$$a = b \longrightarrow P(a) = P(b)$$

"indiscernability of identicals"

logically, we would like

$$a = b \longrightarrow P(a) = P(b)$$

"indiscernability of identicals"

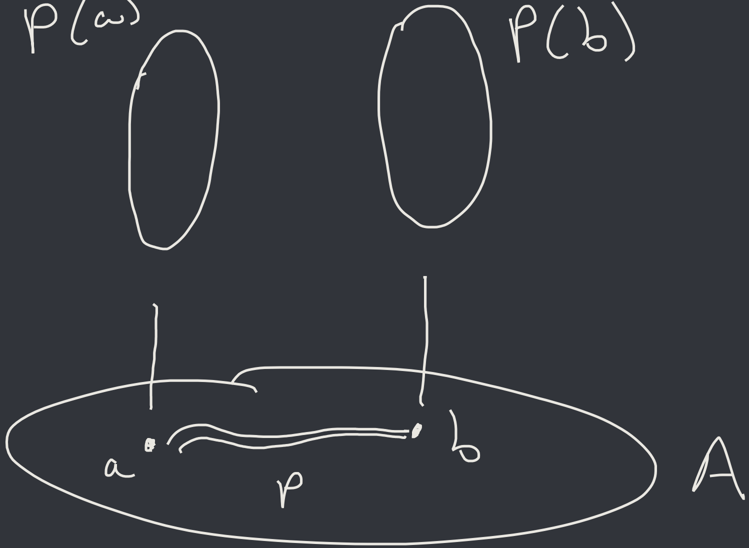
↳ this is <u>true</u>, but we need to use univalence to prove it!

let's start easier:

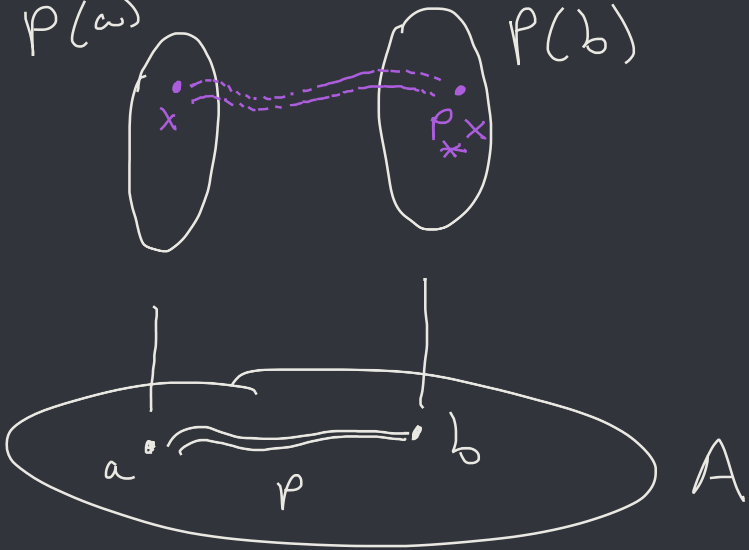
$$a = b \longrightarrow \big(P(a) \longrightarrow P(b) \big)$$

↳ can we prove this?

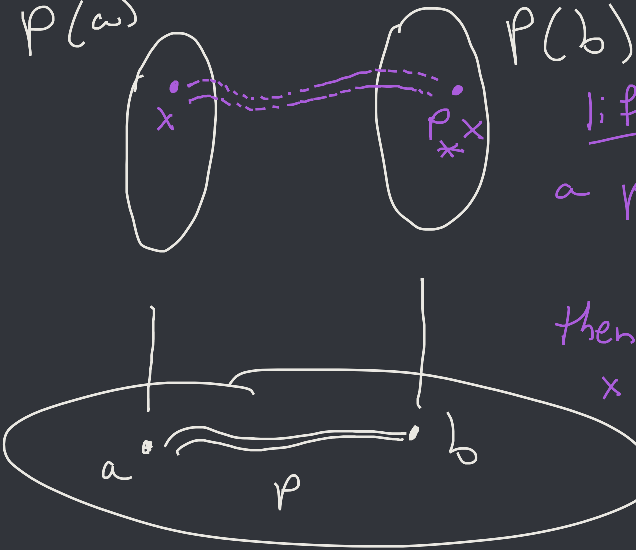
↳ what does it mean geometrically?



p(a)
p(b)
a
b
p
A



P(a)
P(b)
x
p
x
*
a
b
p
A



lift p to a path in $\sum_{a:A} P(a)$ then transport x along that path.

↳ the fact that $p : a = b$ lifts to a path in the total space means type families are <u>fibrations</u>.

↳ the fact that $p : a = b$ lifts to a path in the total space means type families are <u>fibrations</u>.

↳ we prove this via path induction. it suffices to build a map

$P(a) \longrightarrow P(a)$, and the identity works.

So if $p : a = b$, we have maps

- $p_* : P(a) \longrightarrow P(b)$
- $(p^{-1})_* : P(b) \longrightarrow P(a)$

which are easily checked to be inverses.

That is the data of an <u>equivalence</u> $P(a) \simeq P(b)$!

That is the data of an <u>equivalence</u> $P(a) \simeq P(b)$!

(ok, for technical reasons we formally define $\mathrm{isEquiv}(p_*)$ differently, but this data is enough to guarantee p_* <u>is</u> an equivalence.
Cf. Ch 4 of the HoTT book.)

Intuitively, an equivalence

$$A \simeq B$$

says that A and B have "the same information".

Intuitively, an equivalence

$$A \simeq B$$

says that A and B have "the same information".

↳ we can convert back and forth without forgetting anything.

eg $\mathbb{1}+\mathbb{1} \simeq \mathbb{2}$

$$f: \begin{array}{l} \mathrm{inl}\ * \longmapsto T \\ \mathrm{inr}\ * \longmapsto F \end{array}$$

$$\begin{array}{l} \mathrm{inl}\ * \longleftarrow\!\shortmid\ T \\ \mathrm{inr}\ * \longleftarrow\!\shortmid\ F \end{array} : g$$

eg (harder)

$$\mathbb{Z}_1 \triangleq \mathbb{N} + \mathbb{1} + \mathbb{N}$$

$$\mathbb{Z}_2 \triangleq \mathbb{N} \times \mathbb{N} / \sim$$

eg (harder)

$$\mathbb{Z}_1 \triangleq \underset{\text{negatives}}{\mathbb{N}} + \underset{0}{\mathbb{1}} + \underset{\text{positives}}{\mathbb{N}}$$

$$\mathbb{Z}_2 \triangleq \mathbb{N} \times \mathbb{N} / \sim$$

"classical" definition

eg (harder)

- in fact, we can define $+$, $\times$, $\leq$, etc. on both $\mathbb{Z}_1$, and $\mathbb{Z}_2$.
- these structures should be the same
- $\mathbb{N} + \mathbb{1} + \mathbb{N}$ has nice normal forms, but defining $+$, $\times$, $\leq$ etc is annoying.
- $\mathbb{N} \times \mathbb{N} / \sim$ has easy operations, but annoying normal forms.

eg (harder)

- well, if we show $e : \mathbb{Z}_1 \simeq \mathbb{Z}_2$ then we can transport structures, proofs, etc. between them:

eg (harder)

- well, if we show $e : \mathbb{Z}_1 \simeq \mathbb{Z}_2$ then we can transport structures, proofs, etc. between them:

$$n + m \triangleq e^{-1}(en + em)$$

eg (harder)

- well, if we show $e : \mathbb{Z}_1 \simeq \mathbb{Z}_2$

then we can transport structures, proofs, etc. between them:

$$n + m \triangleq e^{-1}(en + em)$$

ex (☆☆)

$$\prod_{n,m : \mathbb{Z}_1} n + m = m + n \ ?$$

meta-theoretically it's clear that we can always do this.

meta-theoretically it's clear that we can always do this.

↳ is there a way to tell the logic about this so we don't need to do it by hand every time?

meta-theoretically it's clear that we can always do this.

↳ is there a way to tell the logic about this so we don't need to do it by hand every time?

↳ A yes! (drumroll please...)

§4

The Univalence Axiom

There is an obvious map

$$A = B \longrightarrow A \simeq B$$

for all $A, B : \mathcal{U}$

There is an obvious map

$$A = B \longrightarrow A \simeq B$$

for all $A, B : \mathcal{U}$

(ex ($\star$) build it!)

There is an obvious map

$$A = B \longrightarrow A \simeq B$$

$$A \simeq B \xrightarrow{ua} A = B$$

for all $A, B : \mathcal{U}$

(ex (☆) build it!)

the univalence axiom says this obvious map has a section.

In a Slogan:

In a Slogan:

$$(A = B) \simeq (A \simeq B)$$

In a Slogan:

$$(A = B) \simeq (A \simeq B)$$

(moreover, it tells us *what* the equivalence is, cf the previous slide)

Ok. But what does this buy us?

$\boxed{\mathrm{I}}$

if $p : a = b$, then

$$p_* : P(a) \simeq P(b).$$

so univalence says

$$ua(p_*) : P(a) = P(b)$$

$\boxed{\mathrm{II}}$

$$e : \mathbb{N} + \mathbb{1} + \mathbb{N} \xrightarrow{\sim} \mathbb{N} \times \mathbb{N} / \sim$$

so

$$\mathrm{ua}(e) : \mathbb{N} + \mathbb{1} + \mathbb{N} = \mathbb{N} \times \mathbb{N} / \sim$$

$\boxed{\text{II.}}$

$$e : \mathbb{N} + \mathbb{1} + \mathbb{N} \xrightarrow{\sim} \mathbb{N} \times \mathbb{N} / \sim$$

so

$$\mathrm{ua}(e) : \mathbb{N} + \mathbb{1} + \mathbb{N} = \mathbb{N} \times \mathbb{N} / \sim$$

so for <u>any</u> construction P on $\mathbb{Z}$,

$$\mathrm{ua}(e)_* : P(\mathbb{Z}_1) \simeq P(\mathbb{Z}_2)$$

write

$$\mathrm{Group} \triangleq \sum_{X:\mathcal{U}} \sum_{e:X} \sum_{m:X^2\to X} \sum_{i:X\to X} \left(\text{group axioms}\right)$$

$\boxed{\text{III}}$ write

$$\text{Group} \triangleq \sum_{X:\mathcal{U}} \sum_{e:X} \sum_{m:X^2\to X} \sum_{i:X\to X} \begin{pmatrix}\text{group}\\ \text{axioms}\end{pmatrix}$$

then $(G, e_G, m_G, i_G) \simeq (H, e_H, m_H, i_H)$

is exactly an isomorphism of groups!

$\boxed{\mathrm{III}}$ So if $\varphi : G \simeq H$,

$$\mathrm{ua}(\varphi) : G =_{\mathrm{Group}} H$$

So <u>every</u> proposition

$$P : \mathrm{Group} \longrightarrow \mathcal{U}$$

agrees on G and H!

III

Questions that don't respect isomorphism are not even expressible!

$\boxed{\text{IV}}$ Univalence decides K.

↳ <u>not</u> every type is a set

↳ intuitively, because we have a new way to get paths now (that isn't refl)

$\boxed{\text{IV}}$ ex (☆☆☆)

$$\mathrm{Set} \triangleq \sum_{x:\mathcal{U}} \mathrm{isSet}(x)$$

is not a set.

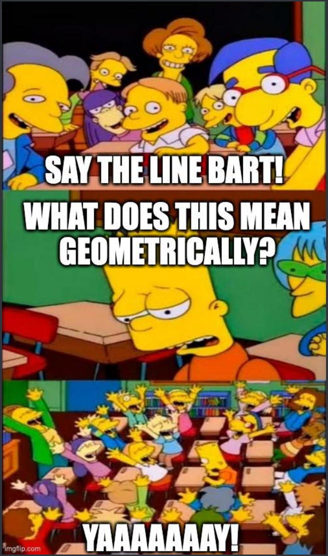
$\boxed{\mathrm{IV}}$ ex ($\star\star\star$)

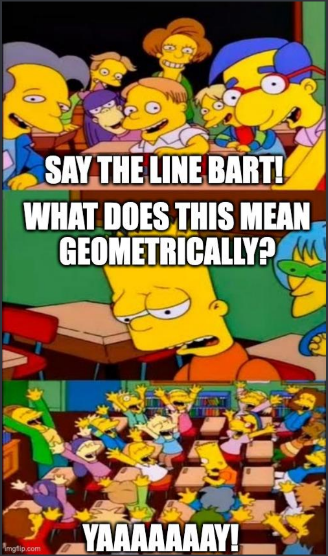
$$\mathrm{Set} \triangleq \sum_{x:\mathcal{U}} \mathrm{isSet}(x)$$

is not a set.

hint: $\ulcorner$ $2 : \mathrm{Set}$, $\neg : 2 \simeq 2$

So $\mathrm{ua}(\neg)$ is a nontrivial path in Set $\lrcorner$


SAY THE LINE BART!
WHAT DOES THIS MEAN GEOMETRICALLY?
YAAAAAAAY!
imgflip.com



to say what univalence buys us geometrically, we're going to need types that are more "obviously" geometric.

§5 Higher Inductive Types.

How do we define $\mathbb{N}$?

How do we define $\mathbb{N}$?

- $0 : \mathbb{N}$

How do we define $\mathbb{N}$?

- $0 : \mathbb{N}$
- $S : \mathbb{N} \to \mathbb{N}$

How do we define $\mathbb{N}$?

- $0 : \mathbb{N}$
- $S : \mathbb{N} \to \mathbb{N}$
- "do this freely"

How do we define $\mathbb{N}$.

- $0 : \mathbb{N}$
- $S : \mathbb{N} \to \mathbb{N}$
- "do this freely"

if $x_0 : X$,

and $\sigma : X \to X$,

$\exists!\ f : \mathbb{N} \to X$

with
$$f0 = x_0$$
$$f(Sn) = \sigma(fn)$$

↳ in general, inductive types give us a way to define new sets in $\mathcal{U}$.

↳ in general, inductive types give us a way to define new <u>sets</u> in $\mathcal{U}$.

↳ is there an analogous way to define types with higher homotopy structure?

↳ in general, inductive types give us a way to define new <u>sets</u> in $\mathcal{U}$.

↳ is there an analogous way to define types with higher homotopy structure?

remember me?

Definition by example

Definition by example

base: S^1

Definition by example

base : S^1

loop : base = base

Definition by example

base : S^1

loop : base $=$ base

be freely generated by these.

Definition by example

base : S^1

loop : base = base

be freely generated by these.

eg not only do we have loop, we also get loop^2, loop^3, loop^{-1}, etc.

↳ all <u>distinct</u> because this object is free!

Definition by example

base : S^1
loop : base $=$ base

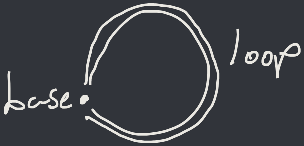
be freely generated by these.

if $x : X$
and $p : x = x$
then $\exists!$
$f : S^1 \to X$
w/ f base $= x$
$\mathrm{apd}_f(\mathrm{loop}) : x = x$

Definition by example

base : S^1
loop : base $=$ base

be freely generated by these.



this is S^1 in HoTT!

Let's see another example:

Let's see another example:

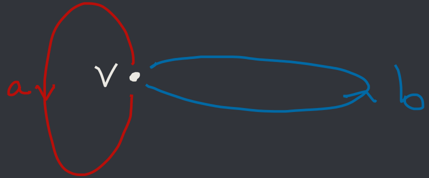
$V : \prod$ $\qquad$ $v \cdot$

Let's see another example:

$V : \mathbb{T}$

$a : V = V$

$b : V = V$



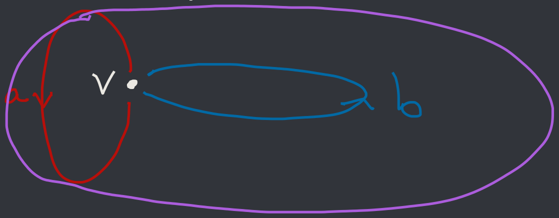
Let's see another example:

$v : \mathbb{T}$

$a : v = v$

$b : v = v$

$T : ab = ba$



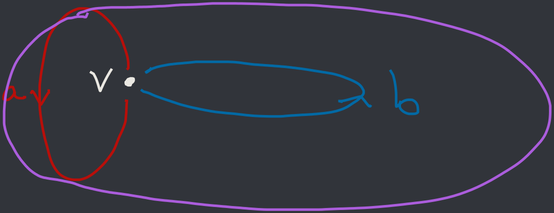
Let's see another example:

$v : \mathbb{T}$

$a : v = v$

$b : v = v$

$T : ab = ba$



maybe this is more clearly drawn as

$$\begin{tikzcd} v \arrow[r, "a"] \arrow[d, "b"'] \arrow[dr, phantom, "\Swarrow T"] & v \arrow[d, "b"] \\ v \arrow[r, "a"'] & v \end{tikzcd}$$

§ 6

Wait ... what does this have to do with univalence?

let's start small.

let's start small.

↳ how do we know $\text{loop} \neq \text{refl}_{\text{base}}$?

let's start small.

↳ how do we know $loop \neq refl_{base}$?

↳ it shouldn't be, but can we prove it?

let's start small.

↳ how do we know $\text{loop} \neq \text{refl}_{\text{base}}$?

↳ it shouldn't be, but can we prove it?

↳ yes! by blending logic and geometry!

recall $\neg : 2 \simeq 2$ So $ua(\neg) : 2 = 2$.

$$\begin{aligned} T &\longmapsto F \\ F &\longmapsto T \end{aligned}$$

recall $\neg : \mathbb{2} \simeq \mathbb{2}$ so $ua(\neg) : \mathbb{2} = \mathbb{2}$

$$\begin{aligned} T &\longmapsto F \\ F &\longmapsto T \end{aligned}$$

now define $P : S^1 \to \mathcal{U}$ by

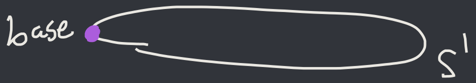
- $P(\text{base}) = \mathbb{2}$
- $\text{apd}_P(\text{loop}) = ua(\neg)$

recall $\neg : \mathbb{2} \simeq \mathbb{2}$ so $ua(\neg) : \mathbb{2} = \mathbb{2}$

$$T \mapsto F$$
$$F \mapsto T$$

now define $P : S^1 \to \mathcal{U}$ by

- $P(\text{base}) = \mathbb{2}$
- $\text{apd}_P(\text{loop}) = ua(\neg)$



recall $\neg : \mathbf{2} \simeq \mathbf{2}$ so $ua(\neg) : \mathbf{2} = \mathbf{2}$

$$\begin{aligned} T &\mapsto F \\ F &\mapsto T \end{aligned}$$

now define $P : S^1 \to \mathcal{U}$ by

- $P(\text{base}) = \mathbf{2}$
- $\text{apd}_P(\text{loop}) = ua(\neg)$

recall $\neg : \mathbf{2} \simeq \mathbf{2}$ so $\mathrm{ua}(\neg) : \mathbf{2} = \mathbf{2}$

$$\begin{aligned} T &\longmapsto F \\ F &\longmapsto T \end{aligned}$$

now define $P : S^1 \to \mathcal{U}$ by

- $P(\mathrm{base}) = \mathbf{2}$
- $\mathrm{apd}_P(\mathrm{loop}) = \mathrm{ua}(\neg)$

now,

$\mathrm{apd}(\mathrm{refl}_{base}) : \mathbf{2} = \mathbf{2}$

$\mathrm{apd}(\mathrm{loop}) : \mathbf{2} = \mathbf{2}$

but if $B : \mathbf{2} \to \mathcal{U}$, then

- $\mathrm{apd}(\mathrm{refl})_* : B(x) \longrightarrow B(x)$
- $\mathrm{apd}(\mathrm{loop})_* : B(x) \longrightarrow B(\neg x)$

now,

$\mathrm{apd}(\mathrm{refl}_{base}) : \mathbf{2} = \mathbf{2}$

$\mathrm{apd}(\mathrm{loop}) : \mathbf{2} = \mathbf{2}$

but if $B : \mathbf{2} \to \mathcal{U}$, then

- $\mathrm{apd}(\mathrm{refl})_* : B(x) \longrightarrow B(x)$ → this is the definition of apd_f
- $\mathrm{apd}(\mathrm{loop})_* : B(x) \longrightarrow B(\neg x)$

now,

$\mathrm{apd}(\mathrm{refl}_{base}) : \mathbf{2} = \mathbf{2}$

$\mathrm{apd}(\mathrm{loop}) : \mathbf{2} = \mathbf{2}$

but if $B : \mathbf{2} \to \mathcal{U}$, then

- $\mathrm{apd}(\mathrm{refl})_* : B(x) \longrightarrow B(x)$
- $\mathrm{apd}(\mathrm{loop})_* : B(x) \longrightarrow B(\neg x)$

this is the computation rule for ua

ex (★★★)

flesh out this proof that

$$\neg\left(\mathrm{loop} = \mathrm{refl}_{\mathrm{base}}\right)$$

ex (★★★)

flesh out this proof that

$$\neg\,(\mathrm{loop} = \mathrm{refl}_{\mathrm{base}})$$

~ bonus ~ (★★★★)

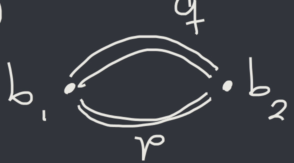
is this the same proof as the one in the HoTT book?

ex (★★)

- let S_1 be defined by
  - $b_1 : S_1$, $b_2 : S_1$
  - $p : b_1 = b_2$, $q : b_2 = b_1$

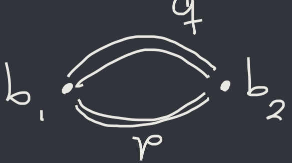
ex (☆☆)

- let S_1 be defined by
  - $b_1 : S_1$, $b_2 : S_1$
  - $p : b_1 = b_2$, $q : b_2 = b_1$



ex (☆☆)

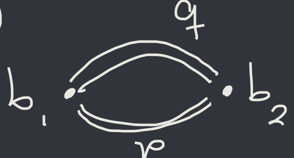
- let S_1 be defined by
  - $b_1 : S_1$, $b_2 : S_1$
  - $p : b_1 = b_2$, $q : b_2 = b_1$



$\boxed{\text{I}}$ prove $S_1 \simeq S^1$

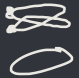
ex (☆☆)

- let S_1 be defined by
  - $b_1 : S_1$, $b_2 : S_1$
  - $p : b_1 = b_2$, $q : b_2 = b_1$



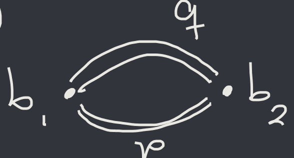
$\boxed{\mathrm{I}}$ prove $S_1 \simeq S^1$

$\boxed{\mathrm{II}}$ prove $\sum_{x:S^1} P(x) \simeq S_1$

(where P is again the double cover )

ex (★★)

- let S_1 be defined by
  - $b_1 : S_1$, $b_2 : S_1$
  - $p : b_1 = b_2$, $q : b_2 = b_1$



$\boxed{\mathrm{I}}$ prove $S_1 \simeq S^1$

$\boxed{\mathrm{II}}$ prove $\sum_{x:S^1} P(x) \simeq S_1$

$\boxed{\mathrm{III}}$ conclude the twisted double cover is again S^1.

ex (☆☆)

Contrast this with

$Q : S^1 \to \mathcal{U}$

$\text{base} \mapsto \mathbf{2}$

$\text{loop} \mapsto \text{refl}_{\mathbf{2}}$,

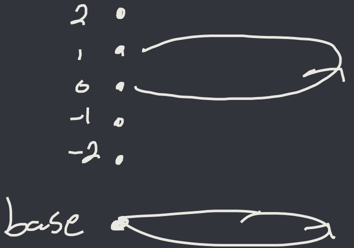
Prove $\sum_{x:S^1} Q(x) \simeq S^1 \times \mathbf{2}$

$(\simeq S^1 + S^1)$

ok, let's kick things up a notch!

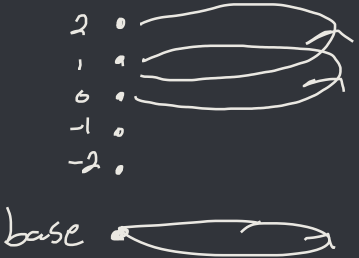
ok, let's kick things up a notch!

- Helix : $S^1 \to \mathcal{U}$
- Helix(base) $= \mathbb{Z}$
- Helix(loop) $= \mathrm{ua}(n \mapsto n+1 : \mathbb{Z} \simeq \mathbb{Z})$



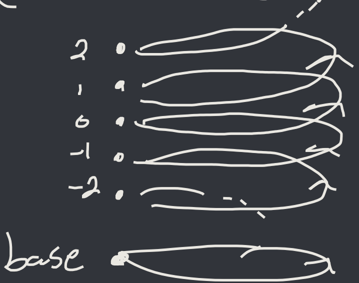
ok, let's kick things up a notch!

- $\mathrm{Helix} : S^1 \to \mathcal{U}$
- $\mathrm{Helix}(\mathrm{base}) = \mathbb{Z}$
- $\mathrm{Helix}(\mathrm{loop}) = \mathrm{ua}(n \mapsto n+1 : \mathbb{Z} \simeq \mathbb{Z})$



ok, let's kick things up a notch!

- $\mathrm{Helix} : S^1 \to \mathcal{U}$
- $\mathrm{Helix}(\mathrm{base}) = \mathbb{Z}$
- $\mathrm{Helix}(\mathrm{loop}) = \mathrm{ua}(n \mapsto n+1 : \mathbb{Z} \simeq \mathbb{Z})$



This is how we compute

$\pi_1 S^1 \cong \mathbb{Z}$ in HoTT!

(see Ch 8 of the HoTT book for more)

§ 7

Conclusion.

From here, we can start doing homotopy theory!

From here, we can start doing homotopy theory!

↳ use HITs to define cell complexes

From here, we can start doing homotopy theory!

↳ use HITs to define cell complexes

↳ define Σ (suspension)

Ω (loop space)

etc

And moreover, everything we do here will have <u>computational</u> <u>content</u>.

And moreover, everything we do here will have <u>computational</u> <u>content</u>.

eg: Brunerie's Number

And moreover, everything we do here will have <u>computational</u> <u>content</u>.

eg: Brunerie's Number

↳ showed "there is a k so that

$$\pi_4(S^3) = \mathbb{Z}/k\mathbb{Z}"$$

eg: Brunerie's Number

↳ showed "there is a k so that $\pi_4(S^3) = \mathbb{Z}/k\mathbb{Z}$"

↳ we can run this proof as a program to learn $k=2$

(since all $\exists$ statements in HoTT come with witnesses!)

eg: Brunerie's Number

↳ showed "there is a k so that $\pi_4(S^3) = \mathbb{Z}/k\mathbb{Z}$"

↳ we can run this proof as a program to learn $k=2$

↳ ... but current implementations are too inefficient :(

Thankfully there are lots of people studying:

- HoTT as foundations (Joyal, etc)

Thankfully there are lots of people studying:

- HoTT as foundations (Joyal, etc)
- HoTT for homotopy (Riehl, etc)

Thankfully there are lots of people studying:

- HoTT as foundations (Joyal, etc)
- HoTT for homotopy (Riehl, etc)
- efficient implementations of HoTT (Licata, etc)

Thankfully there are lots of people studying:

- HoTT as foundations (Joyal, etc)
- HoTT for homotopy (Riehl, etc)
- efficient implementations of HoTT (Licata, etc)
- and much more!

Thankfully there are lots of people studying:

- HoTT as foundations (Joyal, etc)
- HoTT for homotopy (Riehl, etc)
- efficient implementations of HoTT (Licata, etc)
- and much more!

:)

Thank

You

^_^