

Introduction to Homotopy type theory

Andrej Bauer

EPIT 2020¹

Five

~~Six~~ short lectures with exercises:

1. Martin-Löf type theory
2. Identity types
3. Homotopy levels
4. Equivalences
5. Univalence

Background reading:

- Egbert Rijke's textbook - Chapter I
(see #resources on Discord)
- HoTT book - Chapter 1
- Video lectures (see EPIT GitHub repo)

Part 1: Martin-Löf Type Theory

→ Foundational

- It does not rely on logic, set theory or category theory
- Mathematics can be carried out in it

→ A theory of constructions

- How mathematical objects are built & manipulated
- Not about what is the case (that would be logic)
- "Construction" can mean many things
- We take the geometric view:
construction of points & paths in spaces



Basic concepts

Type / Space

A type

Element / Point

$t : A$

Type & point may depend on other points

$x_1 : A_1, x_2 : A_2, \dots, x_n : A_n + \underbrace{B(x_1, \dots, x_n)}$ type
Context Γ
 $x_i : A_i$ parameter of type A_i

Examples from real life:

1. \mathbb{R}^n depends on $n : \mathbb{N}$

$n : \mathbb{N} \vdash \mathbb{R}^n$ type

2. $[a, b]$ depends on $a, b \in \mathbb{R}$ and $a \leq b$

$a : \mathbb{R}, b : \mathbb{R}, p : (a \leq b) \vdash [a, b]$ type

Equality $\Gamma \vdash A \equiv B$

(judgementally) equal types

$\Gamma \vdash s \equiv_A t$

(judgementally) equal points

Type theory as a formal system

Judgement forms

$\Gamma \vdash A \text{ type}$

$\Gamma \vdash t : A$

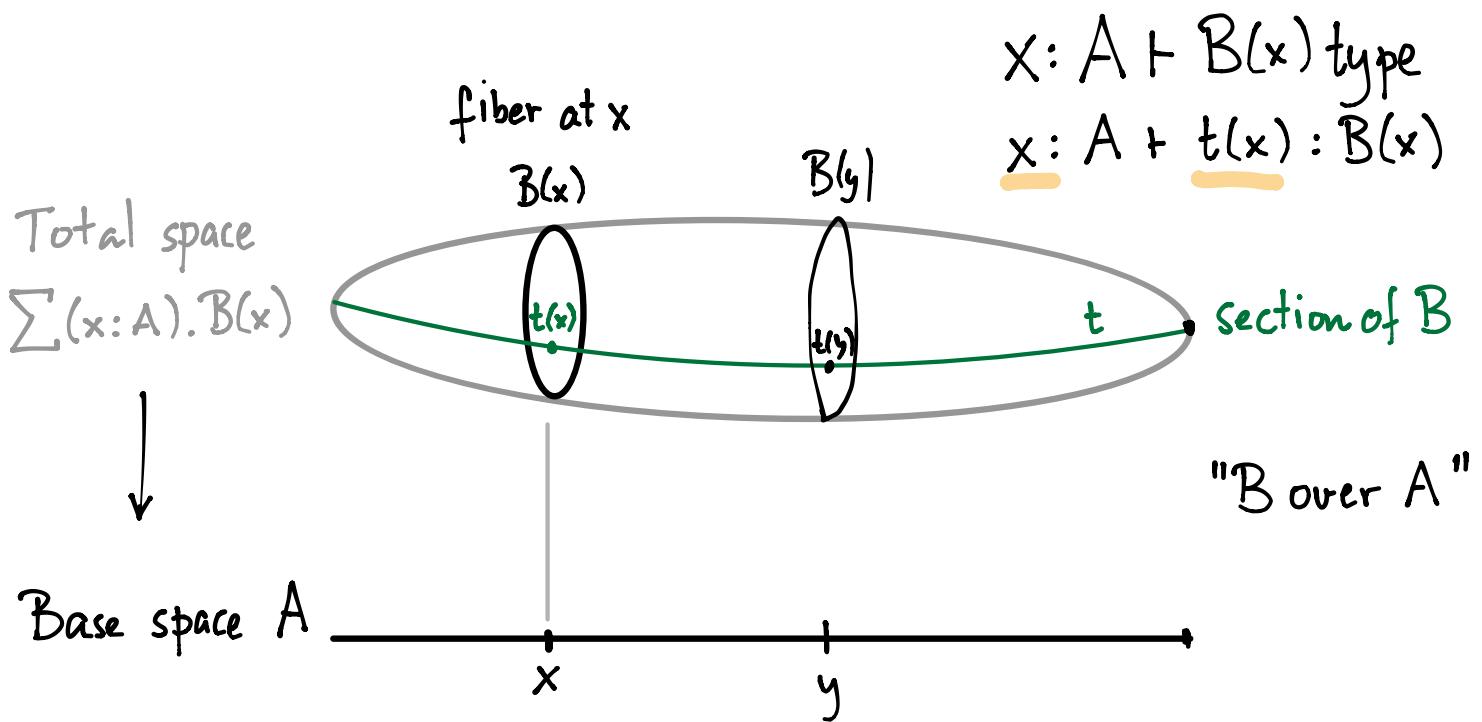
$\Gamma \vdash A \equiv B$

$\Gamma \vdash s \equiv_A t$

Rules of inference

$$\frac{P_1 \quad P_2 \quad \dots \quad P_n}{C} \quad \begin{array}{l} \text{premises} \\ \text{conclusion} \end{array}$$

The geometric picture



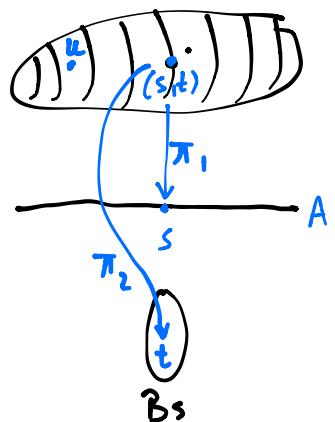
Basic constructions

Presupposition A type

$x:A \vdash B(x)$ type

Dependent sum $\sum(x:A). B(x)$

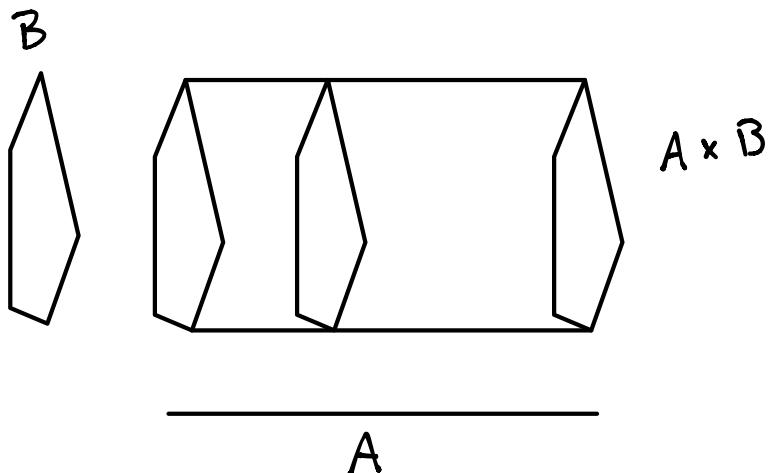
- elements: ordered pairs (s, t) where $s:A, t:B(s)$
- operations: projections π_1 & π_2
- equations $\pi_1(s, t) \equiv_A s$
 $\pi_2(s, t) \equiv_{B(s)} t$
 $(\pi_1 u, \pi_2 u) \equiv_{\sum(x:A). B(x)} u$



Binary product $A \times B := \sum(x:A). B$

$$3 \times 4 = 4 + 4 + 4$$

(B does not depend on A)



Why "sum"?

$B_a \quad B_b \quad B_c$
 $\vdots \quad \vdots \quad \vdots$

$\sum(x:A) B(x)$

$\vdots \quad \vdots \quad \vdots$
 $a \quad b \quad c$

Dependent product $\prod(x:A).B(x)$

- elements: functions $\lambda x:A.e^{:\text{B}(x)}$ "x maps to e"
 $x \mapsto e$

- operation: application

$$\frac{\Gamma \vdash s : \prod(x:A).B(x) \quad \Gamma \vdash t : A}{\Gamma \vdash st : B(t)}$$

- equations:

$$(\lambda x:A.e) t \equiv_{B(t)} e[x:=t]$$

→ substitution

$$\underline{\lambda x:A.sx} \equiv_{\prod(x:A).Bx} \underline{s}$$

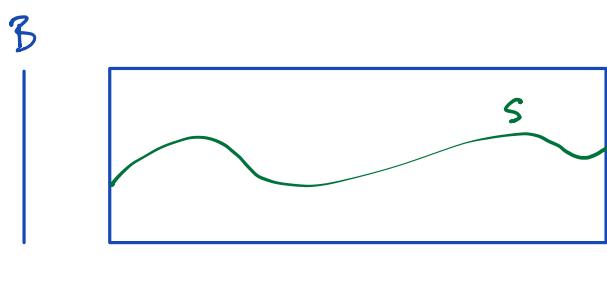
"η-rule"
 ↑ inter-
 ↓ derivable

$$\frac{x:A + fx \equiv_{Bx} gx}{f \equiv_{\prod(x:A).Bx} g}$$

extensionality rule

Function space $A \rightarrow B := \prod(x:A).B$

(B does not depend on A)



$$4^3 = 4 \times 4 \times 4$$

Natural numbers \mathbb{N}

$(\lambda x, s(s(x))) (s0) : \mathbb{N}$

$s^{III} (s(s(s0)))$

- elements: $0 : \mathbb{N}$ and $\frac{n : \mathbb{N}}{S n : \mathbb{N}}$

- induction principle:

Given a type P over \mathbb{N}

$x : \mathbb{N} \vdash \text{exfalso } x : \mathbb{N}$

$$\text{ind}_{\mathbb{N}} : P_0 \xrightarrow{\text{base}} (\prod(k : \mathbb{N}). P_k \rightarrow P(Sk)) \rightarrow \prod(n : \mathbb{N}). P_n$$

- equations

$$\text{ind}_{\mathbb{N}} s t 0 \equiv_{P_0} s$$

$$\text{ind}_{\mathbb{N}} s t (Sn) \equiv_{P(Sn)} t n (\text{ind}_{\mathbb{N}} s t n)$$

Finite types

① empty type: $\frac{}{\vdash t : \emptyset \quad \vdash A \text{ type}} \vdash \text{exfalso} : A$

1 unit type: element $() : \mathbb{1}$
equation $s \equiv_{\mathbb{1}} t$

2 booleans: $\text{false}, \text{true} : \mathbb{2}$

Construction by cases: given type P over $\mathbb{2}$

$$\frac{}{\vdash b : \mathbb{2} \quad \vdash s : P(\text{true}) \quad \vdash t : P(\text{false})}$$

$$\vdash \text{case}(b, s, t) : P(b)$$

if b then s else t

$$\text{case}(\text{true}, s, t) \equiv_{P(\text{true})} s$$

$$\text{case}(\text{false}, s, t) \equiv_{P(\text{false})} t$$

Universe \mathcal{U} - a type of types

- elements are types (not all of them)
- closed under Σ, Π and identity types

We can have many universes: $\mathcal{U}_0, \mathcal{U}_1, \mathcal{U}_2, \dots$

Universes may be cummulative: if $A : \mathcal{U}_n$ then $A : \mathcal{U}_{n+1}$

Observation: A dependent type B over A is a map

$$B : A \rightarrow \mathcal{U}$$

$$B(x) : \mathcal{U}$$

$$\mathcal{U} : \mathbb{N} \rightarrow \mathcal{U}_\infty ?$$

Exercises

① Construct a point of

$$(\prod(x:A). \sum(y:B). C \times y) \rightarrow \sum(f:A \rightarrow B). \prod(x:A). C \times (fx)$$

where A and B are types and $C:A \rightarrow B \rightarrow U$.

② Use ind_N to define $\text{double}: N \rightarrow N$

which doubles a number, e.g.: $\text{double}(S(S0)) \equiv S(S(S(S0)))$.

③ Define $\text{halve}: N \rightarrow N$

which halves a number (rounded down).

Part 2: Identity types

Given a type A and points $s, t : A$, we form the type

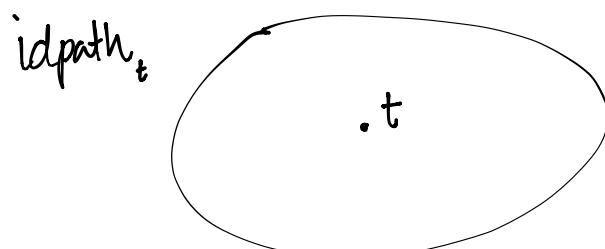
$\text{Id}_A(s, t)$ the identity type
(path type)

We think of $p : \text{Id}_A(s, t)$ as a path from s to t . We shall also use the notation

$s =_A t$ or just $s = t$
(not to be confused with $s \equiv_A t$)

For any $t : A$ there is

$\text{idpath}_A t$: $t =_A t$ also: $\text{refl}_A t$: $t =_A t$



Next we would like to have

Homotopy invariance

If there is a path $p: s =_A t$ then any construction that uses s can be transformed along p to use t instead.

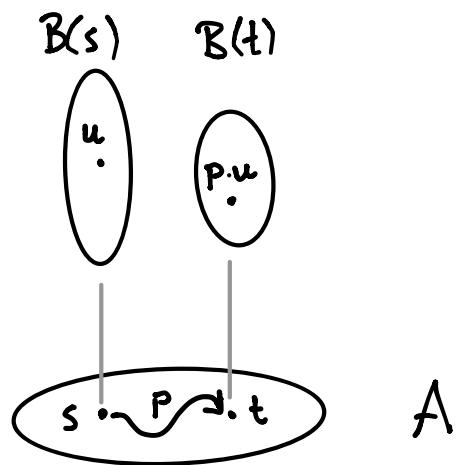
First attempt:

Given $B: A \rightarrow U$

$s, t: A$

$p: s =_A t$

$u: B(s)$



we form the transport

$p.u: B(t)$

Problem: B may depend on x, y and p , not just x .

Solution: path induction

Suppose we would like to construct
a point of

$$\text{TT}(\underline{x : A})(\underline{y : A})(\underline{p : x =_A y}). \underline{B(x, y, p)}$$

Then it suffices to construct a point of

$$d : \text{TT}(\underline{z : A}). B(z, z, \text{idpath } z).$$

We write

$$J(d, \underline{x, y, p}) : B(x, y, p)$$

for the point so constructed.

Equality:

$$J(d, s, s, \text{idpath } s) \equiv d(s).$$

Example: path concatenation

Given $p : x =_A y$ and $q : y =_A z$, construct

$$p \cdot q : x =_A z$$

$$\dots : \prod(x:A)(y:A)(p:x=y) \prod(z:A)(q:y=z) . \underline{x = z}$$

$\underbrace{\hspace{10em}}_{B(x,y,p)}$

We apply path induction:

$$\prod(x:A) B(x,x,\text{idpath } x)$$

$$\prod(x:A) \prod(z:A) (q : x = z) . x = z$$

$\lambda x z q \cdot q$

Answer :

$$p \cdot q := J((\lambda x z q \cdot q), x, y, p) \stackrel{z}{=} q$$

Equation :

$$\text{idpath}_x \cdot q \stackrel{\uparrow}{=} J((\lambda x z q \cdot q), x, x, \text{idpath } x) \stackrel{z}{=} q \stackrel{\uparrow}{=}$$

Transport:

$$\prod(x, y : A)(\varphi : x =_A y) . \ B(x) \xrightarrow{u} B(y)$$

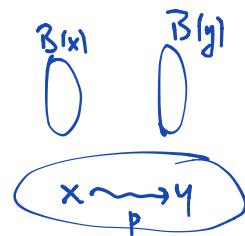
Path induction:

$$\rightarrow \prod(z : A) . B(z) \rightarrow B(z)$$

$$\lambda z \ u' . \ u'$$

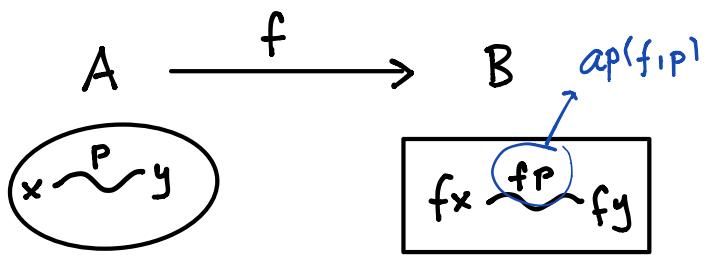
$$\text{Transport: } p.u := J(\lambda z u'. u', x, y, p) u$$

$$\text{idpath}_x . u \equiv u$$



Action on paths

Given $f: A \rightarrow B$



$$f(-) : \prod(x, y : A)(p : x =_A y). f x =_B f y$$

By path induction:

$$\prod(z : A). f z =_B f z$$

$$\lambda z. \text{idpath}_{fz}$$

$$\text{Answer: } f(p) = J(\lambda z. \text{idpath}_{fz}, x, y, p)$$

$$f(\text{idpath } x) = \text{idpath } (fx)$$

Exercises:

① Construct a path between paths:

$$p \cdot \text{idpath}_y =_{x=_A y} p$$

where $x, y : A$ and $p : x =_A y$

② Given a path $p : x =_A y$ construct $\tilde{p}^{-1} : y =_A x$
and show that $\text{idpath}_x^{-1} = \text{idpath}_y$.

③ Suppose $u, v : \sum(x:A). B(x)$

$$p : \pi_1 u =_A \pi_1 v$$

$$q : p \cdot (\pi_2 u) =_{B(\pi_1 v)} (\pi_2 v)$$

are given. Construct a path $u =_{\sum(x:A). B(x)} v$

Part 3: Homotopy levels

Higher-dimensional paths:

$t : A$ point $\cdot t$
 $p : s =_A t$ path $s \xrightarrow{p} t$
 $\alpha : p =_{s =_A t} q$ 2-path $s \xrightarrow{\alpha} t$
(and so on)

Sometimes the higher-dimensional paths collapse above dimension n .

We start with the collapse of points:

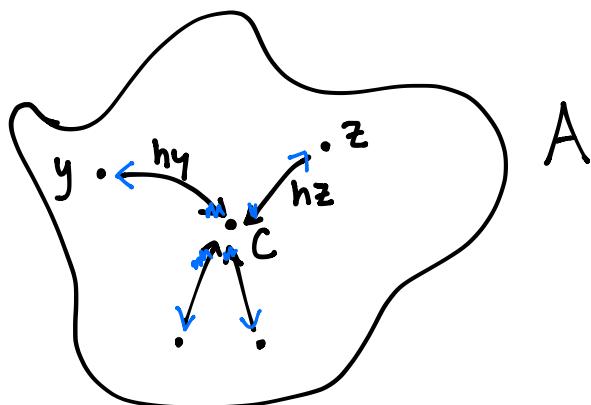
Definition: A type A is contractible when
isContr $A := \sum_{\text{c}} (x : A) . \prod_{y : A} . x =_A y$

has a point.

When $(c, h) : \text{isContr } A$:

- $c : A$ is the center of contraction
- $h : \prod_{y : A} . c =_A y$ is a homotopy from const_c to id_A

(observe that $h : \prod_{x : A} . \text{const}_c x =_A \text{id}_A x$)



Example: $\mathbb{1}$ is contractible

center: ()

homotopy: $\text{TT}(x:\mathbb{1}).() =_{\mathbb{1}} x$ is given by $\lambda x:\mathbb{1}.\text{idpath}_{()}(x)$

(We used the fact that $\text{idpath}_{()}(x) =_{\mathbb{1}} x$ because $(x) \equiv_{\mathbb{1}} x$.)

$$\begin{array}{c} \cdot x \\ \vdots \\ () \end{array}$$

We now generalize to all levels.

Definition

For $n \geq -2$ define the notion of n -type:

(-2) -type $A := \text{isContr } A$

$(n+1)$ -type $A := \text{TT}(x,y:A). n\text{-type}(x =_A y)$

Fact: $n\text{-type } A \rightarrow (n+1)\text{-type } A$

(HoTT book 12.4.3)
Egbert

Special cases of interest:

→ (-1)-types are called propositions:

- equivalent formulation: has at most one point

$$\text{isProp } A := \prod(x, y : A). x =_A y$$

- any two points are equal (connected by a path)
- all higher identity types are contractible
- examples: $\mathbb{0}$, $\mathbb{1}$

$$\begin{array}{c} \downarrow \\ \text{False} \end{array} \quad \begin{array}{c} \downarrow \\ \text{true} \end{array}$$

→ 0-types are called ssets:

- equivalent formulation:

$$\text{isSet } A := \prod(x, y : A) \prod(p, q : x =_A y). p = q$$

- parallel paths are homotopic
- examples: $\mathbb{0}, \mathbb{1}, \mathbb{2}, \mathbb{N}, \mathbb{N} \rightarrow \mathbb{N}$

→ 1-types are called groupoids

- no interesting 2-paths

Property vs. structure

We say that $P:A \rightarrow \mathcal{U}$ is a property on A when

$\prod(x:A). \text{isProp}(P_x)$.

(In general, P is a structure on A.)

$$\begin{array}{ccc}
 P_x & & P_y \quad P_z \\
 \bullet & & \bullet \quad \bullet \\
 \xrightarrow{x} & & \xrightarrow{y} : A
 \end{array}$$

Example:

- The structure of a monoid on a set M:
S

Monoid S :=

$$\sum(m:S \times S \rightarrow S)(e:S).$$

S:U
_ : isSet S

associativity $(\prod(x,y,z:S). m(m(x,y),z) = m(x,m(y,z)) \times$

unit laws $(\prod(x:S). (m(x,e) = x) \times (m(e,x) = x))$

- The property of a monoid being a group:

$\text{isGroup}(S, m, e, \alpha, \beta) :=$

$$\sum_{\text{inverse}}(i:S \rightarrow S). \prod(x:S). (m(x, ix) = e) \times (m(ix, x) = e)$$

- The structure of a group on a set S

Group S := $\sum(M: \text{Monoid } S). \text{isGroup}(M)$

Truncation:

An operation which turns a type A into an n -type.

$$\begin{array}{c} \rightarrow \\ \text{A type} \\ \hline \text{|| } A \text{ ||}_n \text{ type} \end{array}$$

$$\frac{t : A}{\text{|| } t \text{ ||}_n : \text{|| } A \text{ ||}_n}$$

$$\frac{\text{A type}}{\text{n-trunc } A : \text{n-type } \text{|| } A \text{ ||}_n}$$

Universal property of $\text{|| } A \text{ ||}_n$:

every map $f : A \rightarrow B$ into an n -type B factors uniquely through $\text{|| } A \text{ ||}_n$

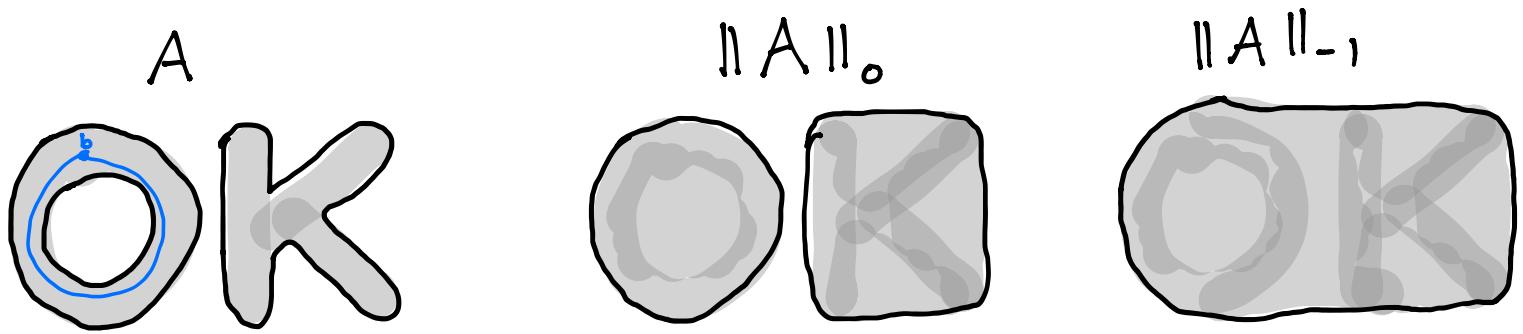
$$\begin{array}{ccc} A & \xrightarrow{\text{|| } \text{I} \text{ ||}_n} & \text{|| } A \text{ ||}_n \\ & \searrow f & \downarrow \\ & B & \text{n-type} \end{array}$$

How do we express the above in type theory?

Special cases of interest

Set truncation $\|A\|_0$ the space of connected components of A

A is connected when $\|A\|_0$ is a ~~proposition~~^{contractible}



Propositional truncation $\|A\|_{-1}$

We use it to define logic

$$\top := \mathbb{1}$$

$$\perp := \emptyset$$

$$\neg P := P \rightarrow \perp$$

$$P \wedge Q := P \times Q$$

$$P \Rightarrow Q := P \rightarrow Q$$

$$P \vee Q := \|P + Q\|_{-1} \quad (\text{where } A+B \text{ is disjoint sum})$$

$$\forall x:A, P(x) := \prod(x:A). P(x)$$

$$\exists x:A, P(x) := \|\sum(x:A). P(x)\|_{-1}$$

These satisfy the rules of intuitionistic logic.

Caveat:

In order to show that \rightarrow and \prod preserve propositions we need function extensionality.

Importance of truncation for mathematics

→ Surjection vs. retraction: $f : A \rightarrow B$

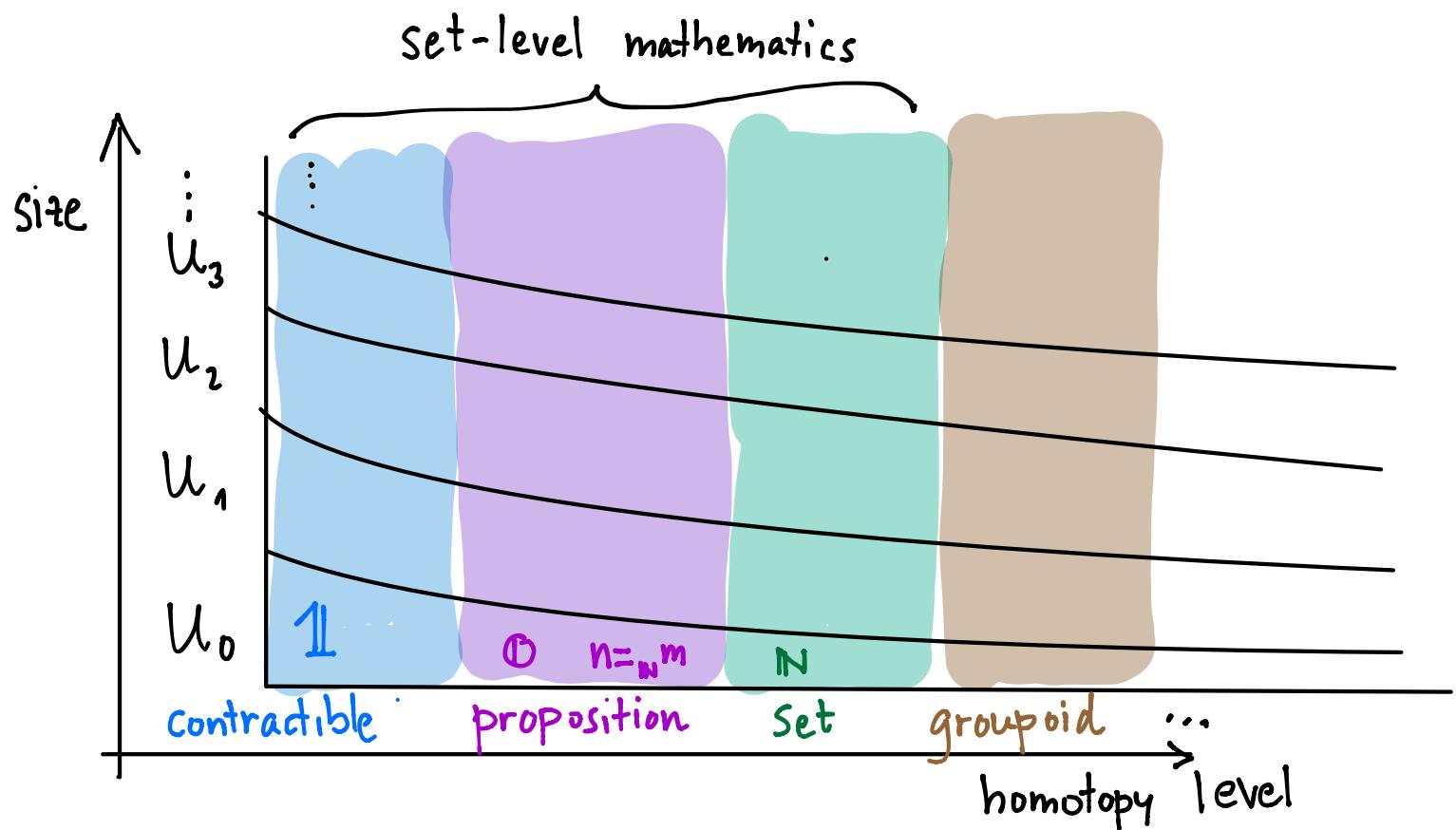
$$\text{TT}(y:B) . \sum(x:A) . f x =_B y \quad \text{"f has a right inverse"}$$

$$\text{TT}(y:B) . \parallel \sum(x:A) . f x =_B y \parallel \quad \text{"f is surjective"}$$

→ Loop space vs. fundamental group

$$\Omega(A, a) := (a =_A a) \quad \text{loop space at base point } a : A$$

$$\pi_1(A, a) := \parallel \Omega(a, a) \parallel_0 \quad \text{fundamental group}$$



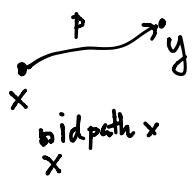
Exercises:

- ① $\text{isContr } A \rightarrow \text{TT}(x, y : A) . \text{isContr}(x =_A y)$
- ② Is contractibility a property or a structure?
- ③ $\text{TT}(x : A) . \text{isContr}(\sum(y : A) . x =_A y)$

Solution: Suppose $x : A$.

We seek an element of $\text{isContr}(\sum(y : A) . x =_A y)$

$\nearrow x . \dots$

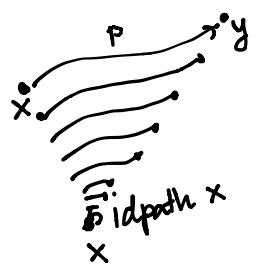


- center $c = (x, \text{idpath } x)$
- homotopy: given any (y, p) where $p : x =_A y$, construct $(x, \text{idpath } x) = (y, p)$

$$\text{TT}(x : A)(y : A)(p : x = y) . (x, \text{idpath } x) = (y, p)$$

Path induction: $\text{TT}(z : A) . z \text{ idpath } z = (z, \text{idpath } z)$

Why does path induction work?



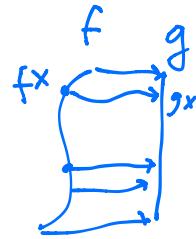
Part 4: Equivalences

Function extensionality

Given maps $f, g : \prod(x:A). B(x)$ define

$$f \sim g := \prod(x:A). f x =_{Bx} g x$$

pointwise equal



We have:

$$f =_{\prod(x:A). Bx} g \rightarrow f \sim g \quad (\text{exercise})$$

How about the other way?

Function extensionality:

$$f \sim g \rightarrow f =_{\prod(x:A). Bx} g$$

It turns out that univalence implies function extensionality.

Given a map $f: A \rightarrow B$, how do we say that it is an equivalence, or an isomorphism?

(1) "f has a two-sided inverse"

$$\text{isIso } f := \sum_g (g: B \rightarrow A). (f \circ g \sim \text{id}_B) \times (g \circ f \sim \text{id}_A)$$

unique?

(2) "Every $y: B$ has exactly one preimage"

$$\text{isEquiv } f := \underline{\text{TT}(y: B)} \cdot \text{isContr} \left(\sum_{\substack{x: A \\ \text{center}}} (x: A). f x =_B y \right)$$

Homotopy fiber of f at y

(3) "f has a left- and right- sided inverses"

$$\text{isBiinv } f := (\sum (g: B \rightarrow A). f \circ g \sim \text{id}_B) \times (\sum (h: B \rightarrow A). h \circ f \sim \text{id}_A)$$

Observations:

→ isEquiv is a property

→ isIso is a structure in general

→ isBiinv is a property & equivalent to isEquiv

Define :

$$A \cong B := \sum (f: A \rightarrow B) \cdot \text{isIso } f$$

structure

structure on f

$$A \simeq B := \sum (f: A \rightarrow B) \cdot \text{isEquiv } f$$

structure

property

We prefer \simeq because isEquiv is a property.

There is a map $A \cong B \rightarrow A \simeq B$
(every iso can be rectified to an equivalence)

Equivalence is equivalence:

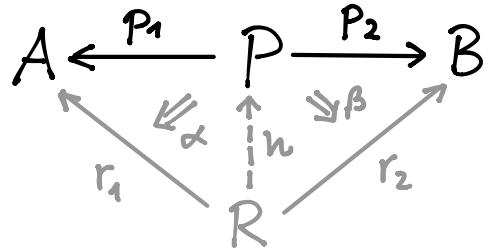
$$A \simeq A \quad \text{id}: A \rightarrow A \text{ is an equivalence}$$

$$A \overset{e}{\simeq} B \rightarrow B \overset{e^{-1}}{\simeq} A$$

$$A \simeq B \rightarrow B \simeq C \rightarrow A \simeq C$$

Equivalences & universal properties

Recall the notion of a product diagram



How to express its universal property?

$$e_R: (R \rightarrow P) \rightarrow (R \rightarrow A) \times (R \rightarrow B)$$

$$e_R(h) := (p_1 \circ h, p_2 \circ h)$$

$$g, \quad \delta$$

$$e_R(h)$$

$$\text{isEquiv}(e_R) = \prod_{(r_1, r_2)} \text{isContr} \left(\sum_h (h: R \rightarrow P) . (p_1 \circ h, p_2 \circ h) = (r_1, r_2) \right)$$

The inverse map e_R^{-1} computes h from (r_1, r_2) and also

$$\alpha: p_1 \circ h = r_1$$

$$\beta: p_2 \circ h = r_2$$

Uniqueness comes from the rest.

Exercises:

① (a) If P and Q are propositions then

$$(P \rightarrow Q) \times (Q \rightarrow P) \rightarrow P \simeq Q$$

(b) If P and Q are propositions then the map you defined in (a) is an equivalence.

(c) Homework: If X and Y are sets then

$$(X \simeq Y) \simeq (X \cong Y)$$

② Use equivalences to express the fact that

$$I_{\perp_1}: A \longrightarrow \|\mathcal{A}\|_{\perp_1}$$

is the universal map from A to propositions:

$$\begin{array}{ccc} A & \xrightarrow{I_{\perp_1}} & \|\mathcal{A}\|_{\perp_1}, \\ & \searrow f & \downarrow \tilde{f} \\ & & P \end{array}$$

For any proposition P and $f: A \rightarrow P$ there is a unique $\tilde{f}: \|\mathcal{A}\|_{\perp_1} \rightarrow P$ such that $\tilde{f} \circ I_{\perp_1} = f$.

Answer: the map $\lambda(g: \|\mathcal{A}\|_{\perp_1} \rightarrow P). g \circ I_{\perp_1}$ is an equivalence

Part 5: Univalence

Function extensionality:

$$(f =_{A \rightarrow B} g) \simeq \prod(x:A). \underline{fx =_B gx}$$

Equality on $A \times B$:

$$(u =_{A \times B} v) \simeq (\pi_1 u =_A \pi_1 v) \times (\pi_2 u =_B \pi_2 v)$$

These are extensionality principles:

"Two objects are equal when their parts (extensions) are equal."

Do we have such a principle for universes?

There is a map

$$\prod(p:A=_u B). A \simeq B$$

$$\text{idtoequiv} : \prod(A, B : \mathcal{U}). \underline{A =_u B} \rightarrow A \simeq B$$

How about $A \cong B$?

Defined by path induction

$$c \simeq c$$

$$\text{idtoequiv } C \in \text{idpath}_c := (\text{id}_c, \dots)$$

$\uparrow \text{istequiv}(\text{id}_c)$

Definition:

A universe \mathcal{U} is univalent if
 idtoequiv is an equivalence.

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

"Identity is equivalent to equivalence."

Axiom of univalence: Every universe is univalent.

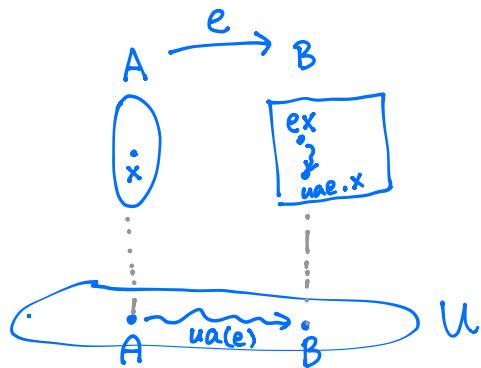
In practice we use the inverse of idtoequiv :

$$ua : (A \simeq B) \longrightarrow A =_u B$$

Given an equivalence $e : A \simeq B$ we get $ua(e) : A =_u B$,
and we know

$$ua(e).x =_B e(x)$$

↑
transport



Some consequences of univalence:

- function extensionality
- it allows us to compute homotopy groups
- it implies that

$$\text{Set}_U := \sum(A : U) . \text{isSet } A$$

is a groupoid (and not a set).

- it implies that isomorphic groups/rings/modules are equal ("Structure identity principle")
- many other mathematically desirable facts

Exercises

- ① Show that the type of true propositions
 $\sum(A : U) . \text{isProp } A \times A$
is contractible.
- ② Show that $\sum(A : U) . \text{isSet } A$ is not a set.
Hint: $(2 \simeq 2) \simeq 2$.