灼熱のホモトピー型理論入門

関山実
東京大学理学部物理学科 4 年
(東京大学理学系研究科物理学専攻に進学予定)

数物セミナー第 28 回特別講演 3/13

本発表の目的

- ホモトピー型理論 (Homotopy Type Theory) を知ってもらう
- 大まかな気持ちや方向性を感じてもらう

全体の概要

この講演では、

- 型理論
- 計算理論と論理学と幾何学との間の美しい同型
- 普遍性による記述
- ホモトピー型理論を基盤にした数学

をざっくり説明する。

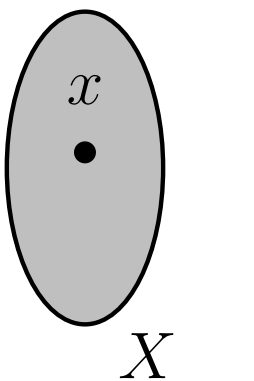
型とは

型理論は、

- 型 X
- 項 x

から構成される。

$$\underbrace{x}_{\text{項}} : \underbrace{X}_{\text{型}}$$



いろいろな型の規則によって項を構成したり計算したりする。
といってもモチベーションが分かりずらい。

プログラミングからのモチベーション

例えば python では

```
5 + 'helloworld'
```

数字と文字列の足し算を実行すると

```
> TypeError: unsupported operand type(s)
> for +: 'int' and 'str'
```

というエラーが出る。
安全性の観点からは、これを実行する前に知りたい。
↓
型理論ではそれが可能

型なしラムダ計算

とても単純なプログラミング言語

1. ラムダ抽象
   x ごとに定まる値 $f(x)$ を、関数にする。

$$f(x) \mapsto \lambda x.f(x)$$

2. 関数適用
   関数 f と値 x をとって、値をかえす。

$$f, x \mapsto f(x)$$

型なしラムダ計算

与えられた式を評価 (β 変換) していくと計算ができる。

$$(\lambda x.f(x))a \Rightarrow f(a)$$

プログラミング言語として表現能力が高い
(チューリング機械と同じクラス)

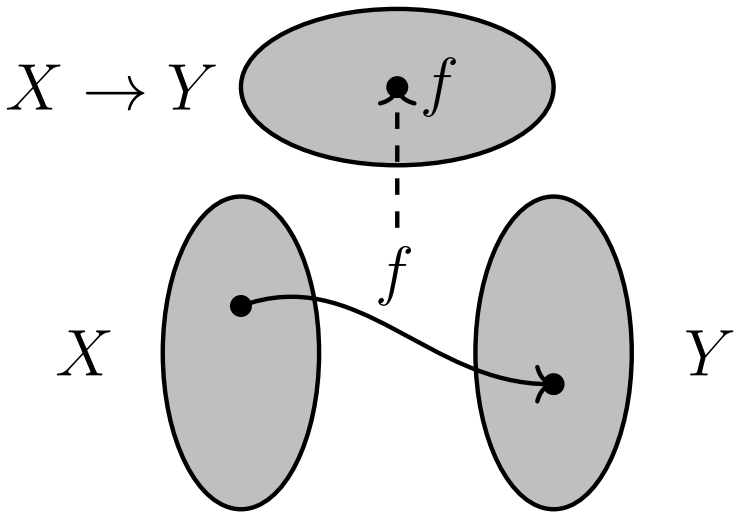
が、いわゆる"バグ"の問題がある。
なので python の場合のように型をつける。

単純型理論

ジャッジメント $\Gamma \vdash \Delta$ で型をチェックしていく。
(Γ は宣言ずみの変数たち、Δ はそれらで構成された式)

1. ラムダ抽象

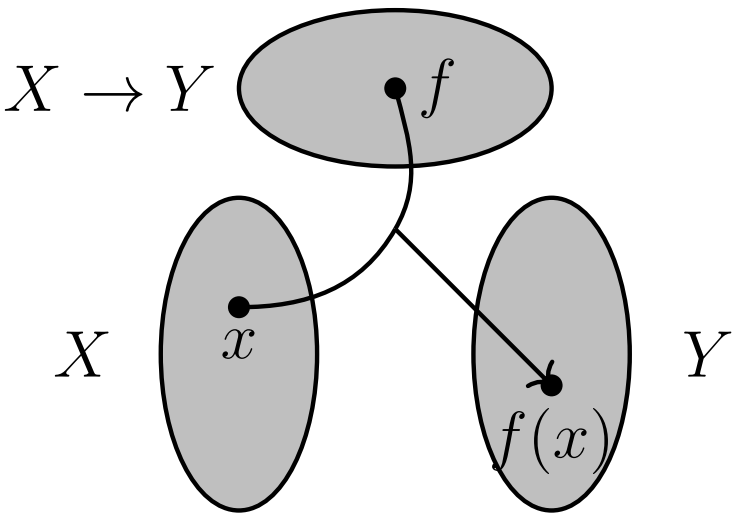
$$\frac{\Gamma, x : X \vdash f(x) : Y}{\Gamma \vdash \lambda x.f(x) : X \to Y}$$



各点に値が定まっているとき、関数を作れる。

2. 関数適用

$$\frac{\Gamma \vdash f : X \to Y \quad \Gamma \vdash x : X}{\Gamma \vdash f(x) : Y}$$



関数と値があるとき、その値での関数の値が得られる。

3. 関数の計算規則
   関数適用したものをラムダ抽象しても変わらない (η-ルール)

$$\lambda x.f(x) \doteq f : X \to Y$$

ラムダ式の簡約 (β-ルール)

$$(\lambda x.f(x))(a) \doteq f(a) : Y$$

単純型理論

型をつけられる項は (計算論的に) いいふるまいをする

- 計算がちゃんと停止する
- 解釈不可能な項が出てこない

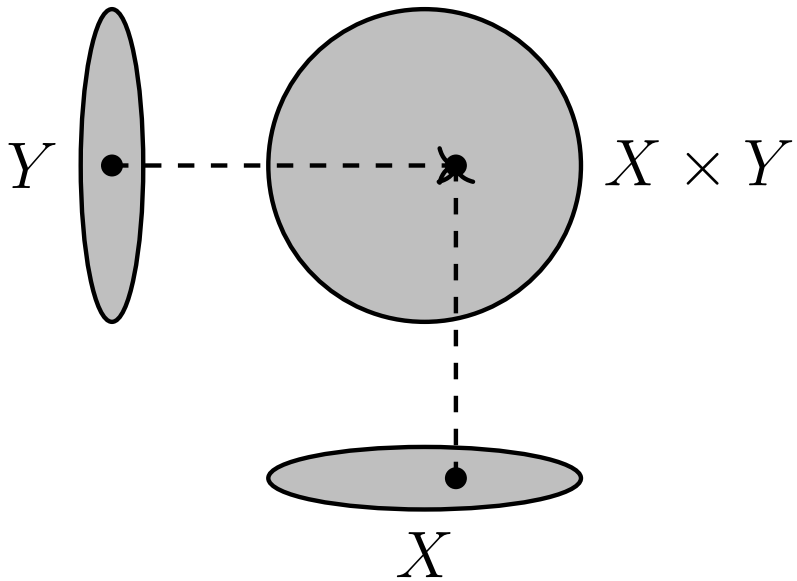
この単純型理論を拡張していくことを考える。

単純型理論

2 つ組の型 (tuple) を考えてみる。

4. 組型

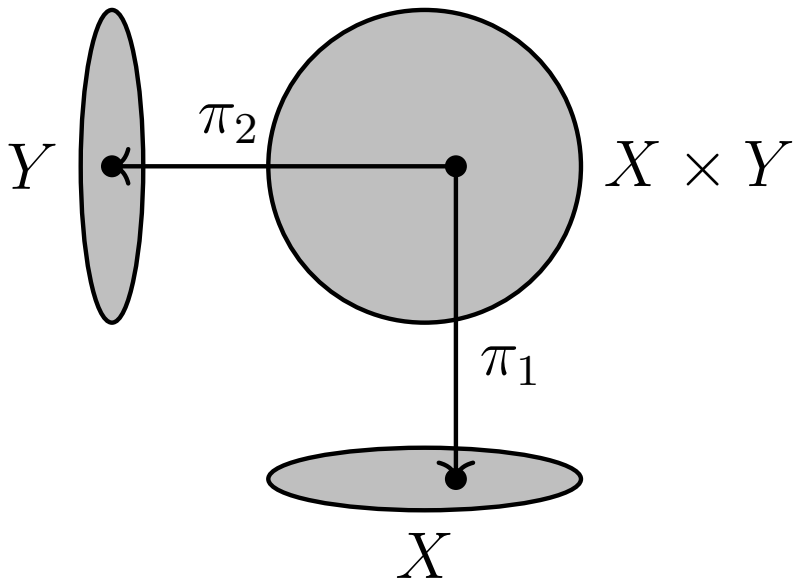
$$\frac{\Gamma \vdash x : X \quad \Gamma \vdash y : Y}{\Gamma \vdash (x, y) : X \times Y}$$



X と Y とで項があるとき、その組が得られる。

5. 射影

$$\frac{\Gamma \vdash t : X \times Y}{\Gamma \vdash \pi_1(t) : X}, \quad \frac{\Gamma \vdash t : X \times Y}{\Gamma \vdash \pi_2(t) : Y}$$



組型から、一方の値のみをかえす。

6. 組型の計算規則

$$\pi_1(x, y) \doteq x : X, \quad \pi_2(x, y) \doteq y : Y$$

組にしてから射影しても変わらない。

型と論理

型付けのルールは、論理における推論のルールと似ている。

例えば、論理における推論ルールは

$\dfrac{\begin{array}{c}[A]\\ \vdots\\ B\end{array}}{A \to B}$	$\dfrac{A \to B \quad A}{B}$	$\dfrac{A \quad B}{A \wedge B}$	$\dfrac{A \wedge B}{A}$
$\to$ の導入則	$\to$ の除去則	$\wedge$の導入則	$\wedge$の除去則

実は、単純型理論と、特定の論理体系とは、きれいに対応する。

直観主義論理 intuitive logic

「命題の真偽は、直接的に構成された証明 (証拠) で判断する」という論理体系。

例えば、

- 排中律 $A \vee \neg A$ や二重否定除去 $\neg\neg A \to A$ が一般には成り立たない。

これは A の直接的な証明 (証拠) を与えていないので直観主義的ではない。

Curry-Howard 同型対応

直観主義論理において、ある命題が証明できる

$\Updownarrow$

その命題に対応する型が項をもつ

Curry-Howard 同型対応

型＝命題の同一視

型理論	直観主義論理
型 X	命題 X
項 x	証明
型 X が項をもつ	命題 X が証明できる
関数型 $X \to Y$	$X \to Y$(ならば)
組型 $X \times Y$	$X \wedge Y$(かつ)

Curry-Howard 同型対応の例

例えば

$$A \to B, B \to C \vdash A \to C$$

は直観主義論理で証明できる。

型理論では、

$$f : A \to B, \quad g : B \to C$$

が与えられたとき、関数の合成によって

$$g \circ f : A \to C$$

を得る。たしかに Curry-Howard 対応はなりたっている。

単純型理論の拡張を考える

Curry-Howard 対応の観点から、型理論を拡張することを考える。

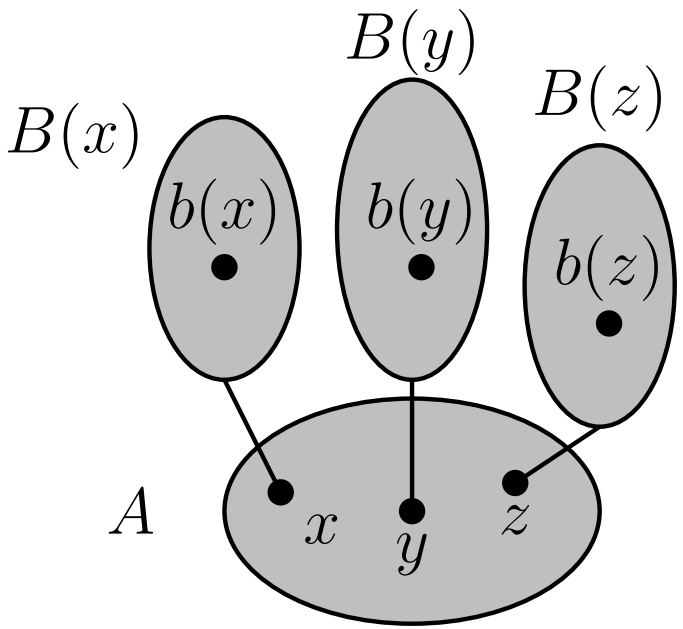
- 命題論理の拡張として、述語論理を考える。
- 型が項に依存するような体系を考えたい。

MartinLöf 依存型理論 MLTT

項ごとに異なる型を考える。

$$\Gamma, x : A \vdash b(x) : B(x)$$

型 A の項 x ごとに型 $B(x)$ の項 $b(x)$ が対応している。

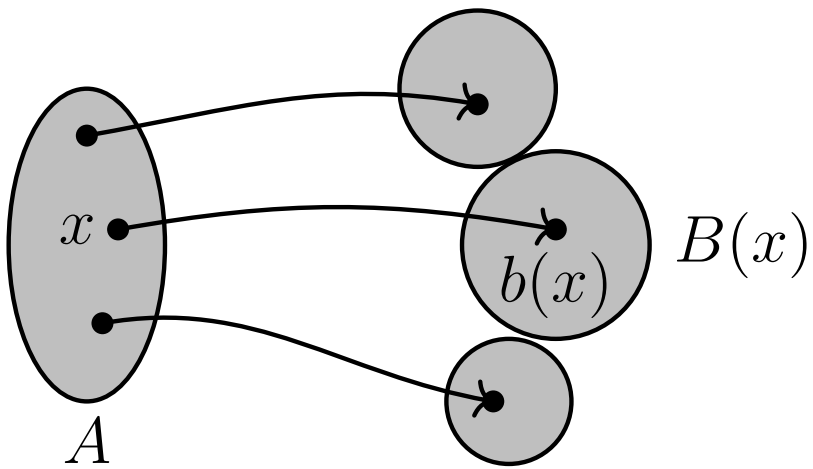


A 上の型の族

依存関数型 dependent function type

要素ごとに行先の型が異なってもよいような関数を考える。

$$\frac{\Gamma, x : A \vdash b(x) : B(x)}{\Gamma \vdash \lambda x.b(x) : \prod_{x:A} B(x)}$$

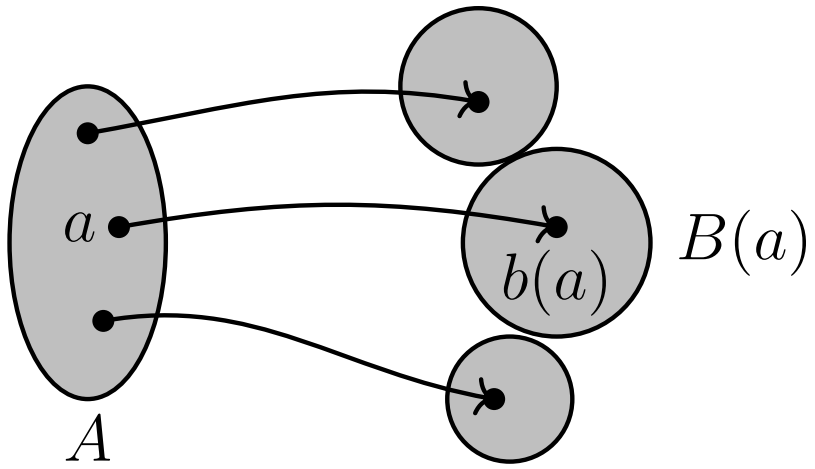


A 上の型の族

依存関数型 dependent function type

Curry-Howard 対応でみると、全称量化子 $\forall$ に対応している。

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



$x \in A$ に依存する命題

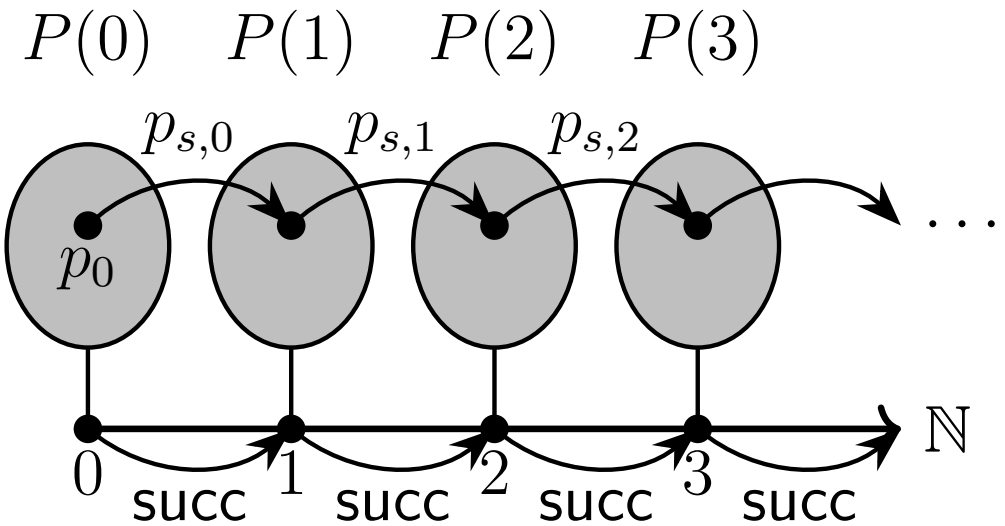
$\forall x \in A(B(x))$ の証明が与えられているとき、A の項 a を指定すると、それに対応する命題 $B(a)$ の証明 $b(a)$ が得られる。

自然数型

$\mathbb{N}$ 型は $0 : \mathbb{N}$ と $\mathsf{succ} : \mathbb{N} \to \mathbb{N}, n \mapsto n + 1$ から生成される。

$\mathbb{N}$ 上の型の族 P を考える。

- $p_0 : P(0)$
- $p_s : \prod_{n:\mathbb{N}} P(n) \to P(n+1)$



この構成によって関数 $\prod_{n:\mathbb{N}} P(n)$ を得る。

帰納型 inductive type

数学的帰納法 (の一般化) を型理論でもやりたい

重要なのは、

- 生成子のふるまいから、生成子で張られる型のふるまいが分かる
- 関数を構成するためには、生成子に対してどうふるまうかを知る必要がある
- 計算規則によって、どのように関数が実装されるのかが分かる

帰納型の例

いろいろな帰納型を考えることができる (割愛)。

- 和型
  $A + B$(集合における非交和)
- $\mathbf{1}$ 型
  ただ 1 つの項をもつ型
- $\emptyset$ 型
  1 つも項を持たない型
- **依存組型**
- **等式型**

帰納型の表現能力は高い。

依存組型 dependent pair type

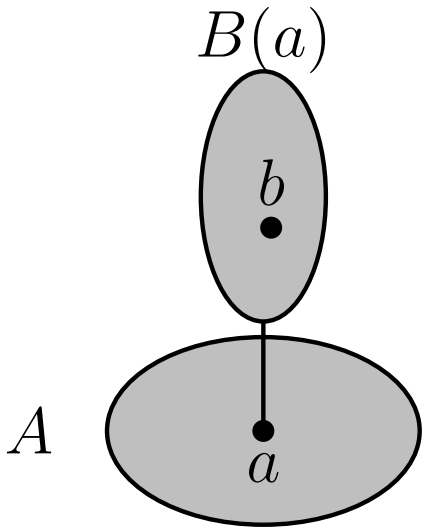
組型だが、２つめの型が１つめに依存する。

$$(a, b) : \sum_{x:A} B(x)$$

生成子は $a : A, b : B(a)$

$$\mathsf{pair} : \prod_{x:A} \left(B(x) \to \sum_{y:A} B(y) \right), \quad a, b \mapsto (a, b)$$

で生成されるもの。

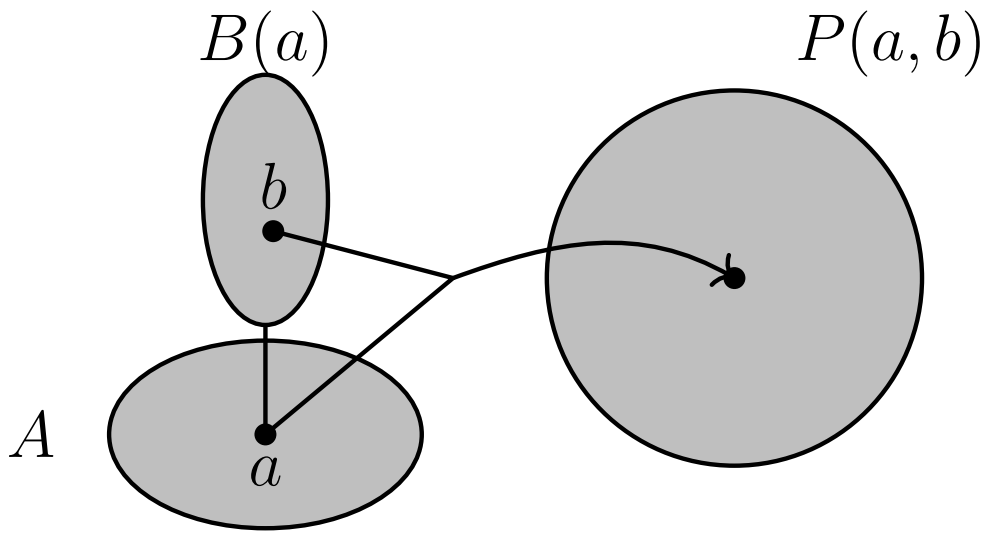


依存組型 dependent pair type

依存組型の帰納規則は

- 任意の $x : A$
- それに対応する型 $B(x)$ の任意の要素 y

からの対応が分かれば依存関数を構成できると主張。

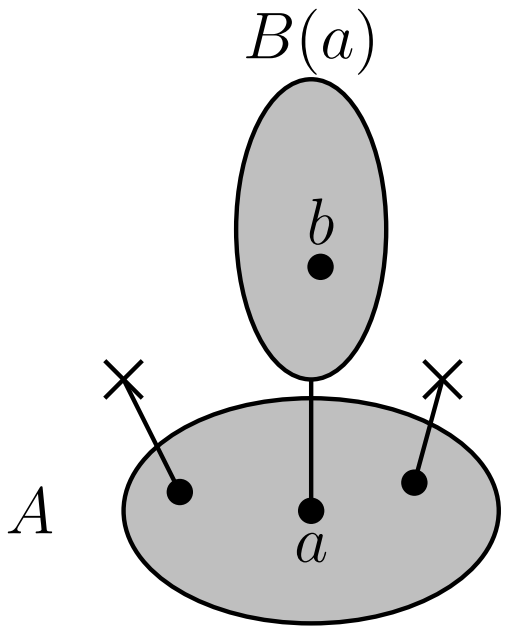


依存組型 dependent pair type

Curry-Howard 対応で見ると、存在量化子 $\exists$ に対応している。

$$(a, b) : \sum_{x:A} B(x)$$

$x : A$ であって、命題 $B(x)$ をみたすようなものが存在する。



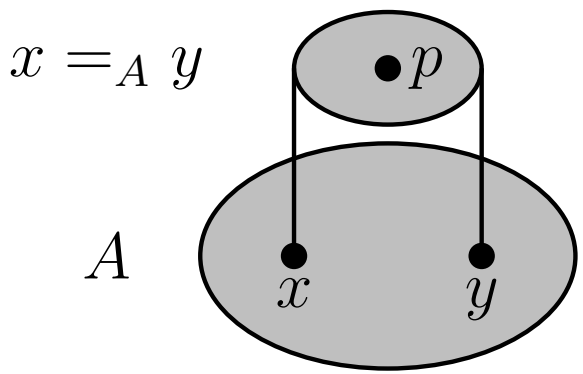
等式型 identity type

$x, y : A$ について等式の型を考える。

$$x =_A y$$

要素は x と y が等しいことの証明に対応する。

$$p : x =_A y$$



等式型 identity type

等式型に依存する型は、

$$P(x, y, p), \quad x, y : A, p : x =_A y$$

のように 3 変数に依存している。

等式型の生成子は、反射律に対応する。

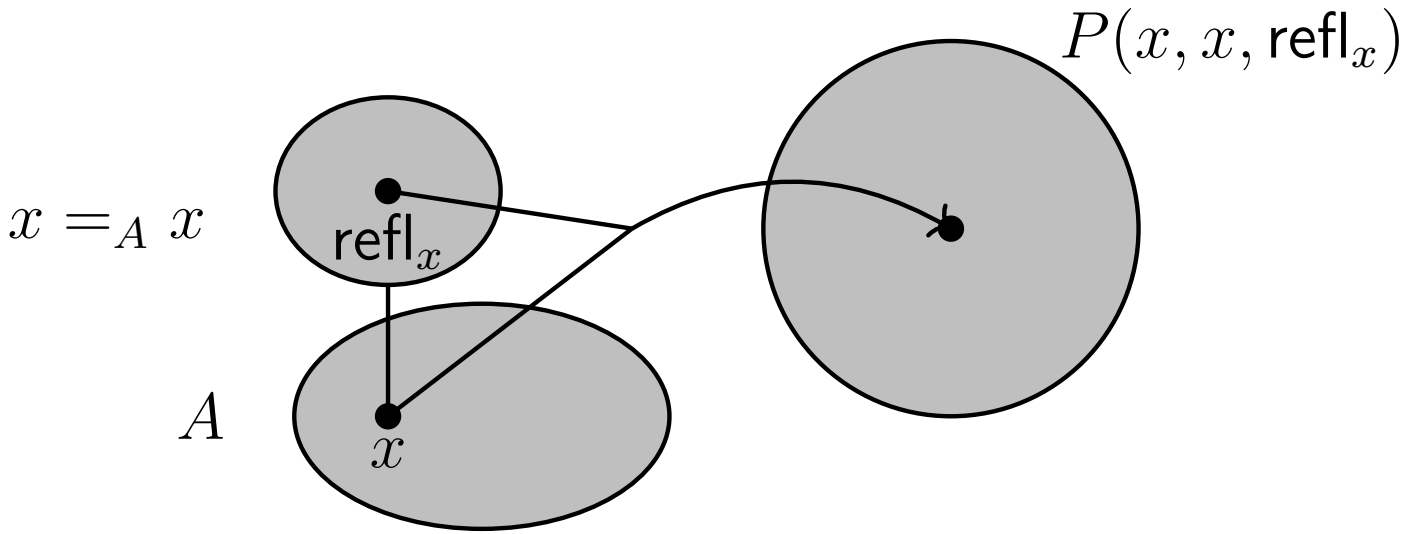
$$\mathsf{refl}_x : x =_A x$$

道帰納法 path induction

等式型からの関数は、

- $x \doteq y, p \doteq \mathsf{refl}_x$ の場合に項を構成できる

ならば構成できる (path induction)。



あまり直観的ではないが、この定義でうまくいく。

等式型の計算

等式の性質を証明 (項を構成) できる。

- 対称律 $x = y \to y = x$
- 推移律 $x = y \to (y = z \to x = z)$

例えば対称律の場合には、

$$\mathsf{inv} : (x =_A y) \to (y =_A x), \quad p \mapsto \mathsf{inv}(p)$$

を構成すればいいが、$x \doteq y, p \doteq \mathsf{refl}_x$ の場合のみ考えて $x =_A x$ の項を指定する。

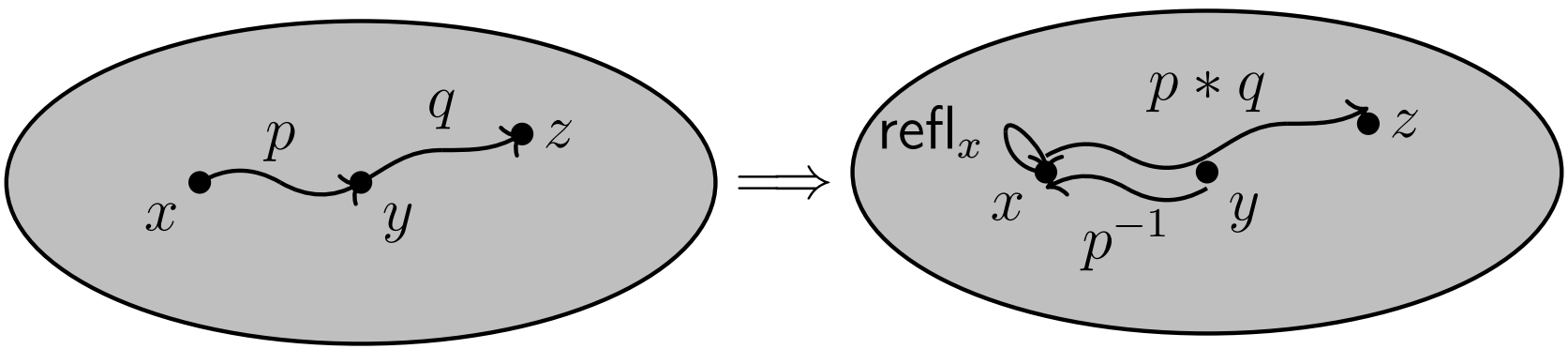
$$\mathsf{inv}(\mathsf{refl}_x) = \mathsf{refl}_x$$

とすればいい。あとは path induction をする。

等式は道である

$$p : x =_A y \Longleftrightarrow \text{点 } x \text{ と点 } y \text{ を結ぶ道 } p$$

- 反射律 $\mathsf{refl}_x : x =_A x$
- 対称律 $\mathsf{inv} : (x =_A y) \to (y =_A x)$
- 推移律 $\mathsf{concat} : (x =_A y) \to (y =_A z) \to (x =_A z)$



ホモトピー解釈

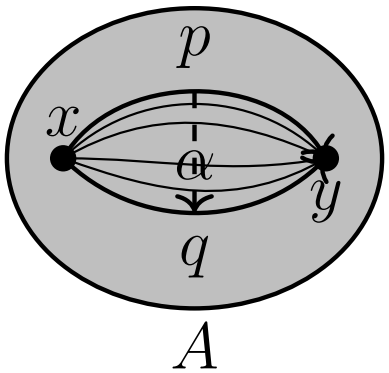
型＝空間の同一視

型理論	ホモトピー論
型 A	空間 A
項 $x : A$	A 上の点 x
等式型 $x =_A y$	x から y への道全体
$p : x =_A y$	x から y への道 p
refl_x	x から x への自明な道
p^{-1}	p を逆にたどった道
$p * q$	p と q をつなげた道

道の道

道の間の道 (高階の道) を考えることができる。

$$\alpha : p =_{x=_A y} q, \quad p, q : x =_A y$$



道の道の道,$\cdots$ としてどんどん高階の道を考えることができる。

型は高階の亜群である

$$x \overset{p}{=} y \overset{q}{=} z \overset{r}{=} w$$

について

$$\mathsf{assoc}_{p,q,r} : (p * q) * r = p * (q * r)$$

を構成できる。

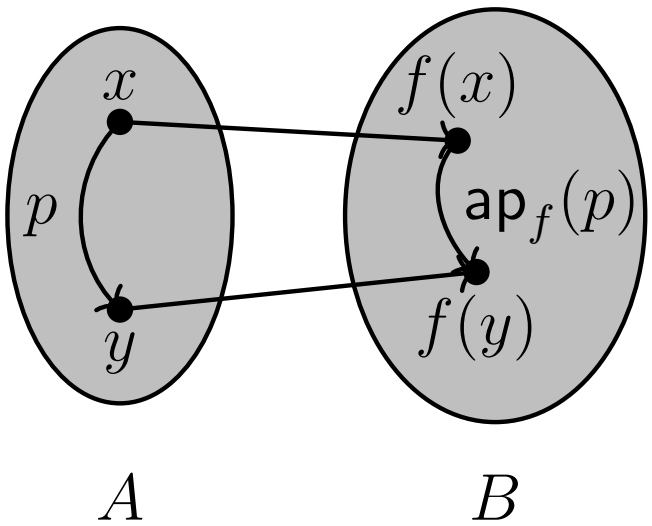
等式型	群
項 p	元 p
$p * q$	合成
refl_x	単位元
p^{-1}	逆元
assoc	結合法則

型を亜群 (groupoid) として扱うこともできる。

道の作用 action on path

A の道を関数 $f : A \to B$ によって B の道に送ることができる。

- $\mathsf{ap}_f : x =_A y \to f(x) =_B f(y)$

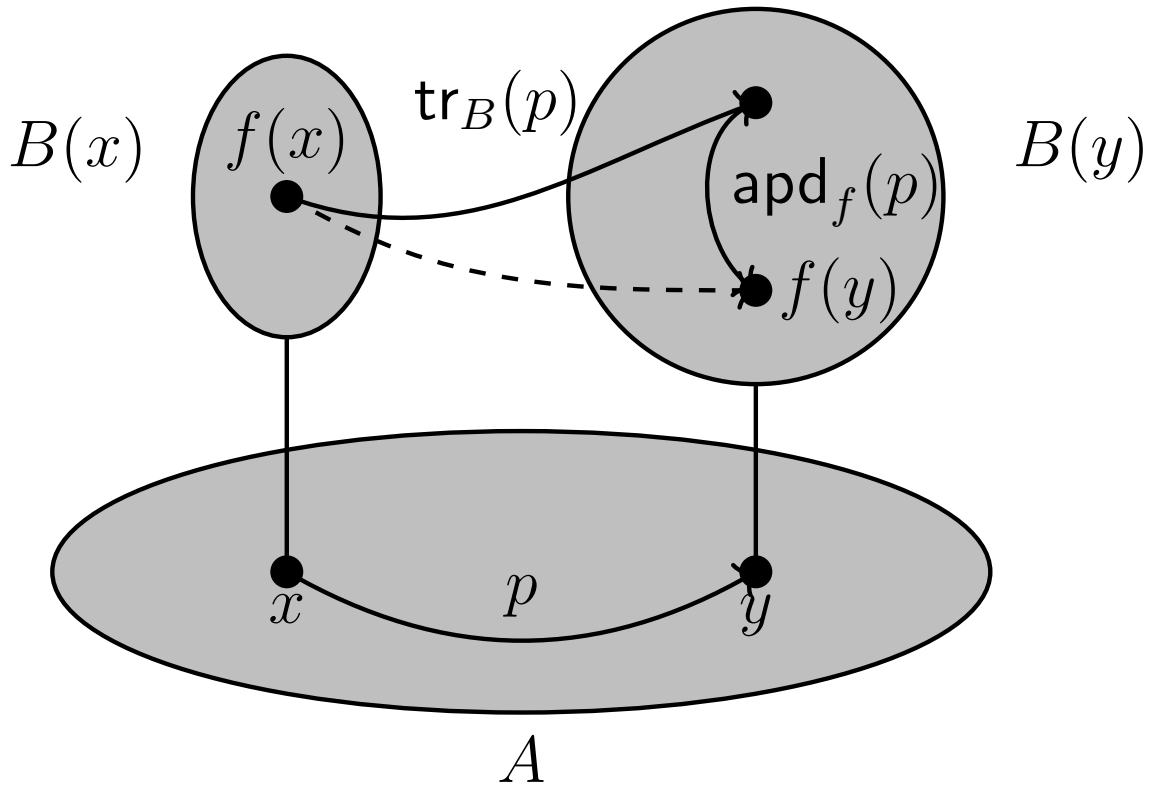


ap は等式型に対して関手的にふるまう。

依存関数版の道の作用 action on dependent path

依存型 $B(x)$ および依存関数 $f : \prod_{x:A} B(x)$ についても同じようなことが言える。

- $\mathsf{tr}_B : x =_A y \to (B(x) \to B(y))$
- $\mathsf{apd}_f(p) : \mathrm{tr}_B(p, f(x)) = f(y)$



帰納型のまとめ

- 生成子のふるまいが重要
- 関数は生成子と同調して (矛盾なく) ふるまう必要がある
- 帰納型からの依存関数をどう構成するかに興味がある

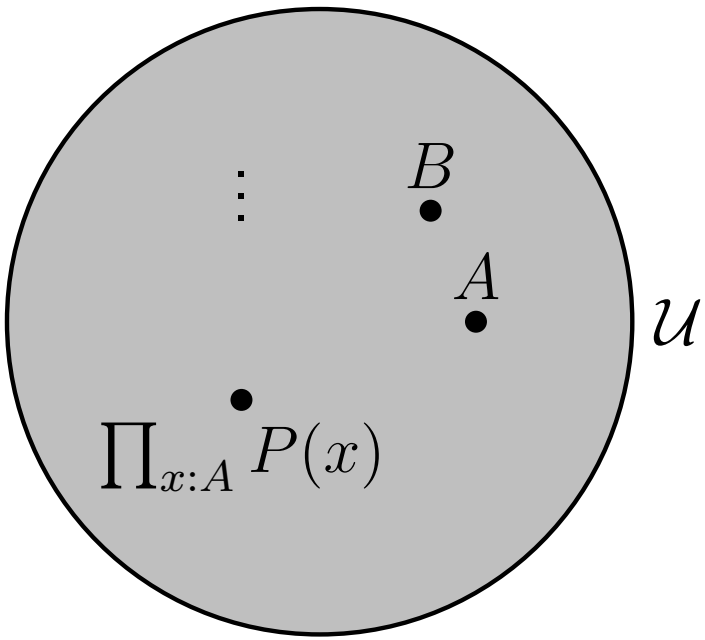
あとは、型の族をとる方法を知りたい。

宇宙

型の型を考える。

$$A : \mathcal{U}$$

型の構成に関して閉じている。



型の族は型から宇宙への関数だと思える。

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

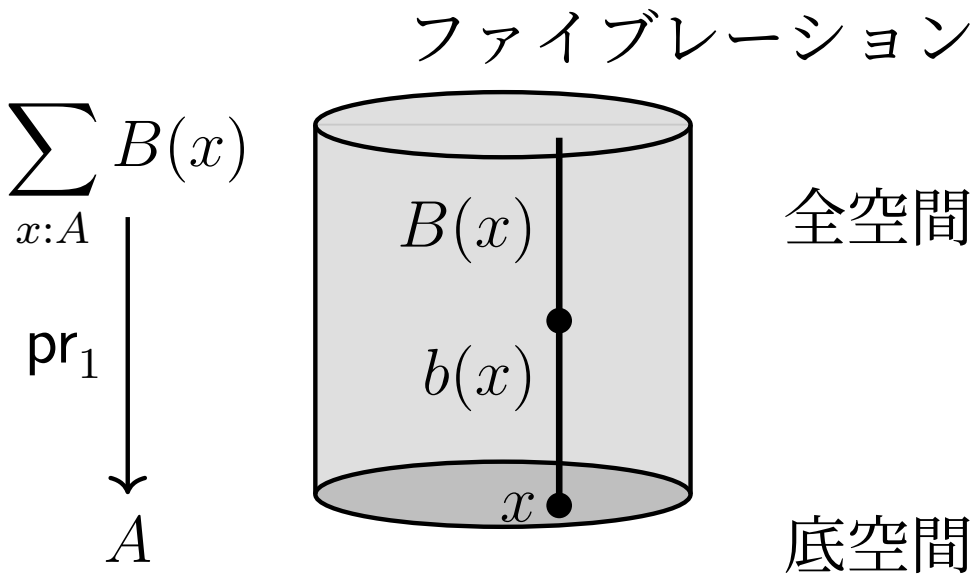
MLTT のまとめ

MartinLöf 依存型理論の主な構成要素:

- 依存関数型
- 帰納型
  - 依存組型
  - 等式型
- 宇宙

幾何的な意味論がある

幾何からの視点



等式型とあわせて道の持ち上げができる。

型理論の内部で幾何学を展開することができる (ホモトピー型理論)

ホモトピー

関数の外延性を記述する。$f, g : A \to B$ に対し

$$f \sim g \doteq \prod_{x:A} f(x) =_B g(x)$$

MLTT では $f \sim g \to f = g$ は必ずしも成り立たない。

ホモトピーは関数で記述される。

$$H : \prod_{x:A} f(x) = g(x), \quad H(x) : f(x) =_B g(x)$$

逆射 inverse

$f : A \to B$ の section(前から合成)

$$\mathsf{sec}(f) \doteq \sum_{g:B\to A} f \circ g \sim \mathsf{id}_A$$

$f : A \to B$ の retraction(後ろから合成)

$$\mathsf{retr}(f) \doteq \sum_{g:B\to A} g \circ f \sim \mathsf{id}_B$$

同値射 equivalence

これらを同時にもつとき、f は同値射

$$\mathsf{is\text{-}equiv}(f) = \mathsf{sec}(f) \times \mathsf{retr}(f)$$

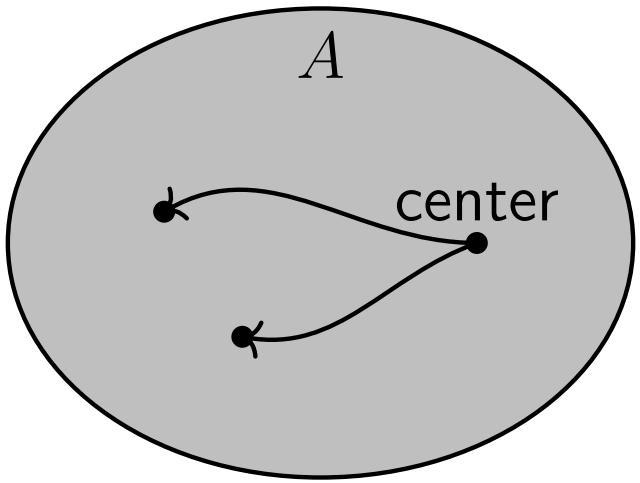
A と B の間に同値射があるとき $A \simeq B$

$$A \simeq B \doteq \sum_{f:A \to B} \mathsf{is\text{-}equiv}(f)$$

可縮 contractible

すべての点が、ある 1 点と道で結ばれるような型。

$$\mathsf{is\text{-}contr}(A) = \sum_{c:A} \prod_{x:A} c =_A x$$

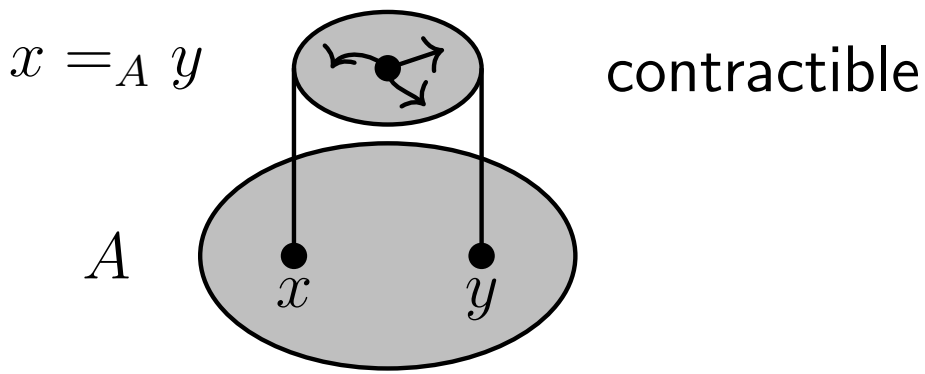


可縮な型は本質的には 1 つの要素をもつ。
(-2)-type とも言われる。

命題 proposition type

任意の等式型が可縮であるような型。

$$\mathsf{is\text{-}prop}(A) = \prod_{x,y:A} \mathsf{is\text{-}contr}(x =_A y)$$



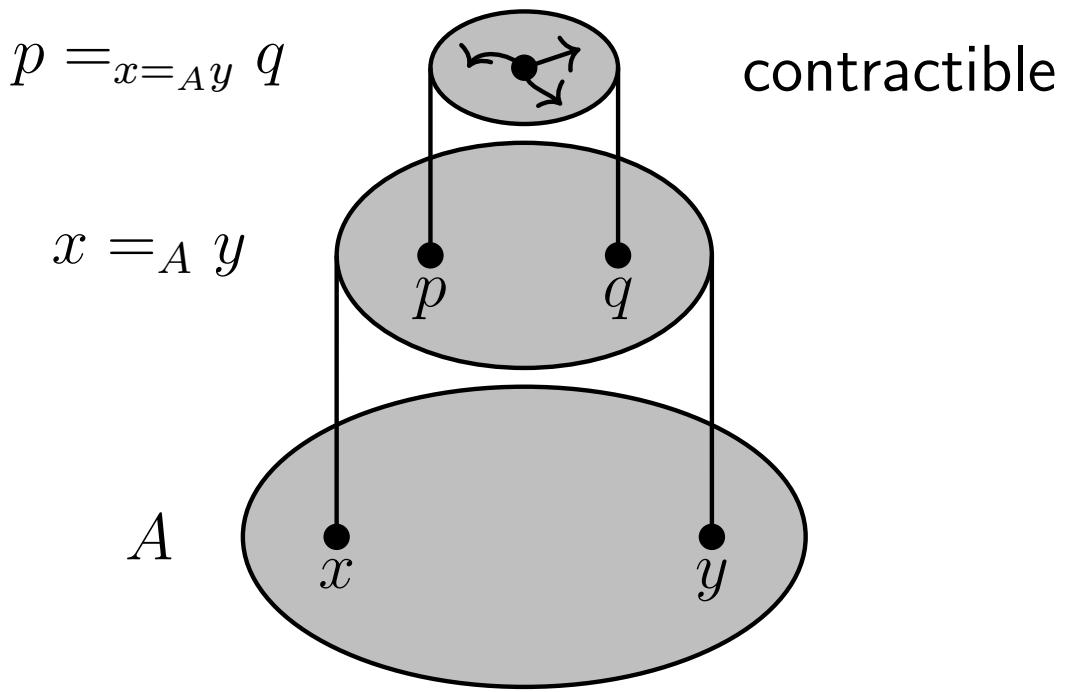
命題の性質

- 命題の要素はすべて等しい。
- 命題は要素を 1 個か 0 個もつ。
- (-1)-type とも言われる。

集合 set type

任意の等式型が命題であるような型。

$$\mathsf{is\text{-}set}(A) = \prod_{x,y:A} \mathsf{is\text{-}prop}(x =_A y)$$



高次の truncated type

これを繰り返していくと、

$$\begin{aligned}\mathsf{is\text{-}trunc}_{-2}(A) &\doteq \mathsf{is\text{-}contr}(A)\\ \mathsf{is\text{-}trunc}_{k+1}(A) &\doteq \textstyle\prod_{x,y:A} \mathsf{is\text{-}trunc}_k(x =_A y)\end{aligned}$$

高次の truncated type が得られる。

A の k-type への丸め込みを $||A||_k$ と書く。

Univalence 公理

同値射の存在を公理的に要請。

$$\mathsf{equiv\text{-}eq} : (A =_{\mathcal{U}} B) \simeq (A \simeq B)$$

以下が同値。

- 型として同値か
- 型として等しいか

いろいろな応用がある便利な公理。
例えば関数の外延性なども示せる。

等式型の基本定理

点付きの型 $a : A$ および A 上の型の族 $B(x)$ および $b : B(a)$ にたいして、

$$f : \prod_{x:A} (a =_A x) \to B(x)$$

が $f(a, \mathsf{refl}_a) = b$ をみたすとする。

このとき次の条件が同値

- f が各 x で同値射
- 全空間 $\sum_{x:A} B(x)$ が可縮

HoTT のまとめ

ホモトピー型理論の主な構成要素

- MLTT
- ホモトピー同値
- 可縮性
- Univalence 公理

実際に幾何学 (synthetic homotopy theory) を展開してみよう

synthetic homotopy theory

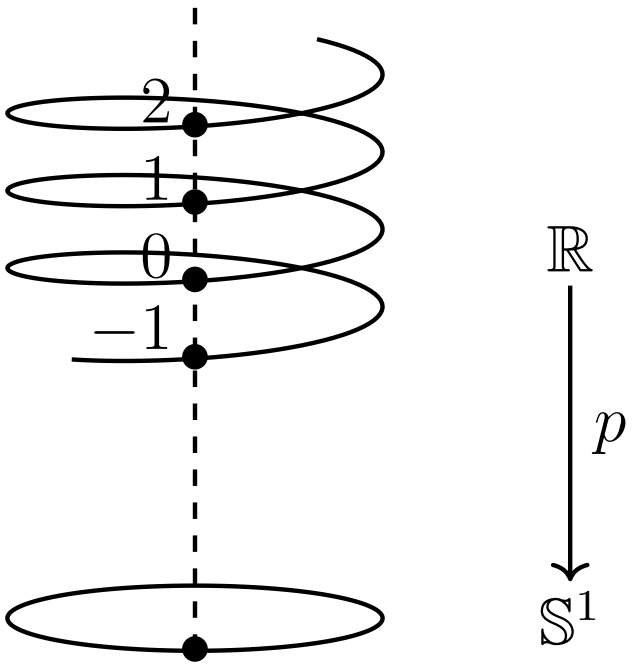
classical $\Longleftrightarrow$ **synthetic**

- 帰納型の推論ルールは普遍性を記述していた
- 普遍性による記述は具体的な構成によらないので便利
- **証明が正しいかどうかをコンピュータで検証できる**
  - Coq や Agda、LEAN などの証明支援系
  - Univalent Foundation Program(UFP)

今回の目標:HoTT をつかって $\pi_1(\mathbb{S}^1) \cong \mathbb{Z}$ であることを示す。

古典的なホモトピー論

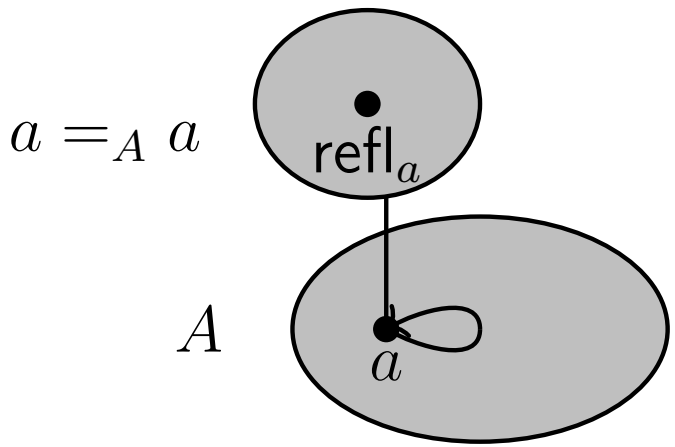
普遍被覆の方法による。



$$\pi_1(\mathbb{S}^1) \cong \pi_1(\mathbb{S}^1, \bullet) \cong p^{-1}(\bullet) \cong \mathbb{Z}$$

基点つきループ空間

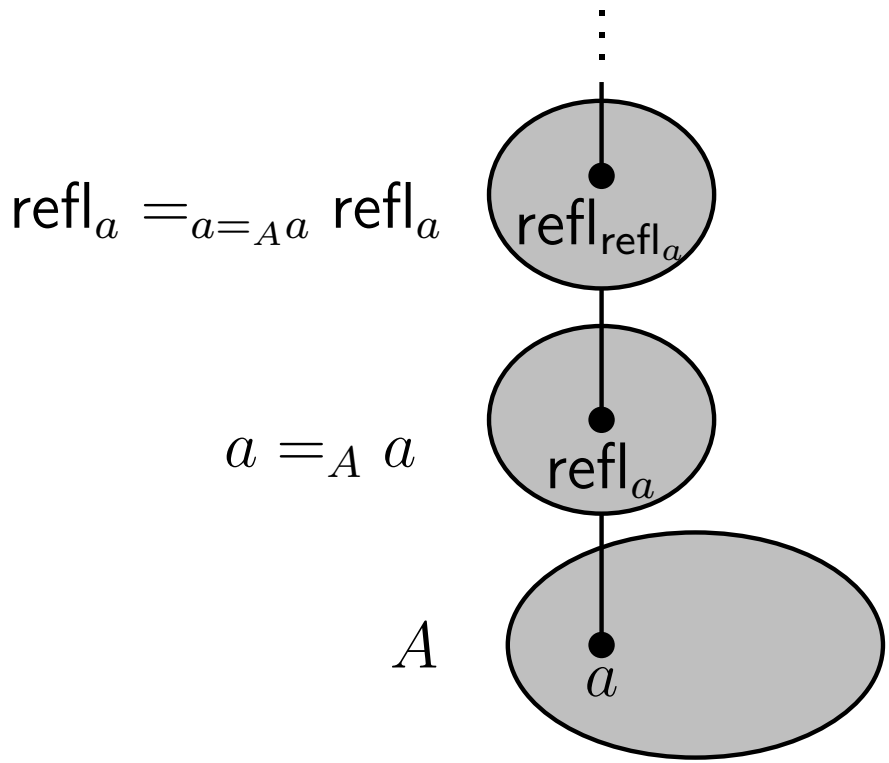
$$\begin{array}{rccc} \Omega: & \mathcal{U}_* & \longrightarrow & \mathcal{U}_* \\ & \vdots & & \vdots \\ & (A, a) & \longmapsto & (a =_A a, \mathsf{refl}_a) \end{array}$$



幾何的には、A の a 上のループ空間に対応する。

基点つきループ空間

$$\Omega^{n+1}(A, a) \doteq \Omega(\Omega^n(A, a))$$



高次のループ空間を考えることができる。

基点つきホモトピー群

- 基点付きループ空間は群の公理を満足する
- ただし集合とは限らない
- 群は集合である必要がある

集合に丸め込むことでホモトピー群を得る

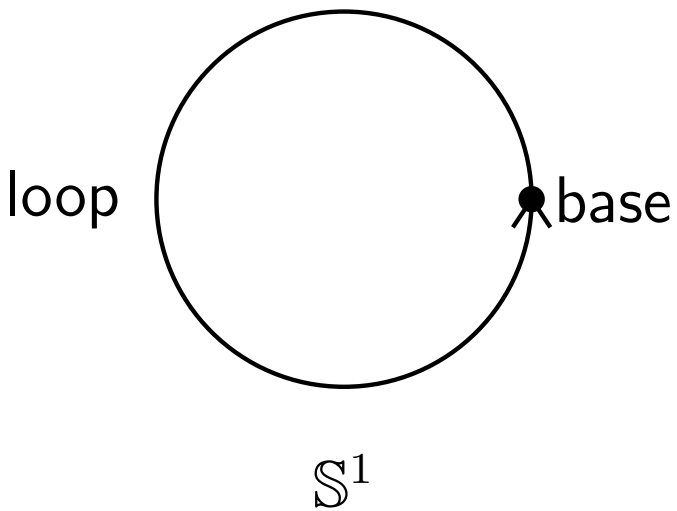
$$\pi_n(A) \cong \pi_n(A, a) \doteq ||\Omega^n(A, a)||_0$$

synthetic $\mathbb{S}^1$

base : $\mathbb{S}^1$

loop : base $=_{\mathbb{S}^1}$ base

非自明な loop をもつ



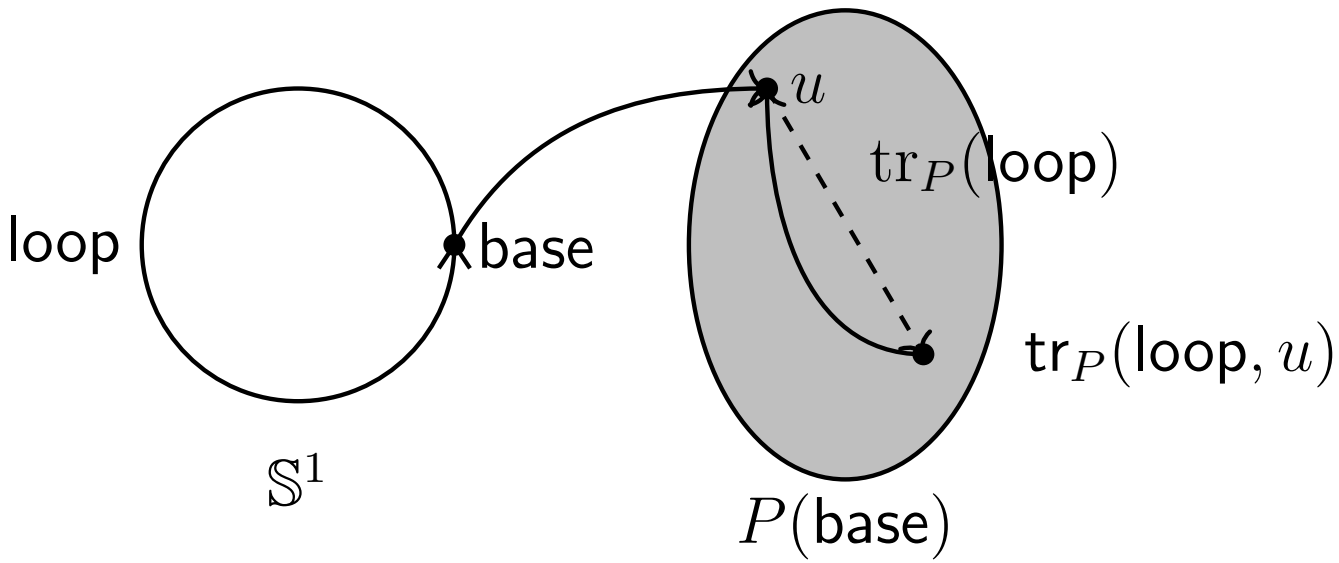
synthetic $\mathbb{S}^1$ からの関数

$\mathbb{S}^1$ からの依存関数を定めるには、

- base の行先
- loop による tr と identity

$$\sum_{u:P(\mathsf{base})} \mathrm{tr}_P(\mathsf{loop}, u) = u$$

があればよい。



synthetic $\mathbb{S}^1$ の普遍性

$$\begin{array}{rccc} \mathsf{gen}_{\mathbb{S}^1} : & (\mathbb{S}^1 \to X) & \xrightarrow{\simeq} & \left(\sum_{x:X} x =_X x\right) \\ & \vdots & & \vdots \\ & f & \mapsto & (f(\mathsf{base}), \mathsf{ap}_f(\mathsf{loop})) \end{array}$$

は同値射になっている。

$\mathbb{S}^1$ からの関数の情報は、すべて等式型で表現できる。

synthetic 被覆

$\mathbb{S}^1$ 上の各点に空間を対応させる操作は、$\mathbb{S}$ 上の型の族をとる操作

$$\mathbb{S}^1 \to \mathcal{U}$$

$\mathbb{S}^1$ の普遍性と、Univalence 公理から、

$$(\mathbb{S}^1 \to \mathcal{U}) \simeq \left(\sum_{X:\mathcal{U}} X =_{\mathcal{U}} X\right) \simeq \left(\sum_{X:\mathcal{U}} X \simeq X\right)$$

より型 X と同値射 $X \simeq X$ を定めると、$\mathbb{S}^1$ の被覆を取ることができる。

encode-decode 法による $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$ の証明

$(\mathbb{Z}, \mathsf{succ}_{\mathbb{Z}})$ に対応する $\mathbb{S}^1 \to \mathcal{U}$ を

$$\begin{aligned} &\mathsf{code}(\mathsf{base}) \doteq \mathbb{Z} \\ &\mathsf{ap}_{\mathsf{code}}(\mathsf{loop}) \doteq \mathsf{eq\text{-}equiv}(\mathsf{succ}_{\mathbb{Z}}) \end{aligned}$$

で定める。情報を $\mathbb{Z}$ に code している。
元に戻すには

$$\begin{array}{rccc} \mathsf{decode}: & \mathbb{Z} & \longrightarrow & \mathsf{base} = \mathsf{base} \\ & \vdots & & \vdots \\ & n & \mapsto & \mathsf{loop}^n \end{array}$$

ループ空間についての encode-decode 法

点付きの型 (A, a) とファイブレーション $\mathsf{code} : A \to \mathcal{U}$ に対して、

1. $c : \mathsf{code}(a)$
2. $\mathsf{decode} : \prod_{x:A} \mathsf{code}(x) \to (a =_A x)$
   - $y : \mathsf{code}(a)$ に対して $\mathsf{tr}_{\mathsf{code}}(\mathsf{decode}(a, y), c) = y$
   - $\mathsf{decode}(a, c) = \mathsf{refl}_a$

をみたすとき、

$$\prod_{x:A} a =_A x \simeq \mathsf{code}(x)$$

が成り立つ。

証明は等式型の基本定理による。

encode-decode 法による $\pi_1(\mathbb{S}^1) \cong \mathbb{Z}$ の証明

ということで

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

がわかる。

$$\pi_1(\mathbb{S}^1) \doteq ||\Omega(\mathbb{S}^1, \mathsf{base})||_0 \doteq ||\mathsf{base} = \mathsf{base}||_0 \cong ||\mathbb{Z}||_0 \doteq \mathbb{Z}$$

synthetic homotopy theory まとめ

- encode-decode 法は、普遍被覆の理論を用いていない。
- 実数を用いていない。
- 帰納型の性質 (普遍性による記述) のみを用いている。
- 証明をコンピュータ上で検証可能

まとめ

ホモトピー型理論 HoTT は激アツ (灼熱)

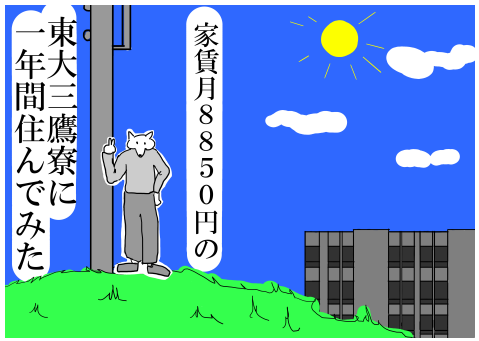
どうアツいか:

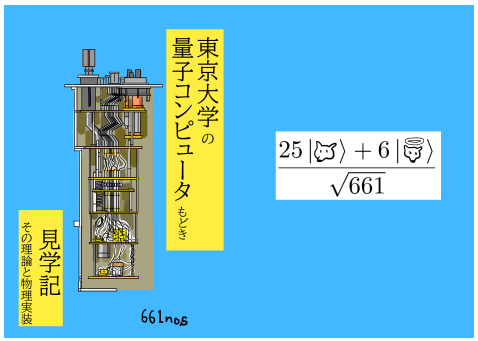
- 型理論は**論理 計算 幾何**の間にある美しい同型を記述する
- 帰納型+Univalence 公理は新しい数学を生み出す
- 普遍性による記述 (圏論) は同一視をするときに強力
- 証明をコンピュータ上で検証することができる

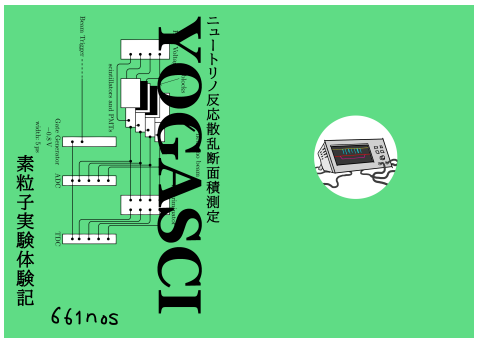
宣伝

年に１回のペースで漫画を書いて、イベントで頒布しています。
これまでに書いた内容:

1. 三鷹寮 (東大の寮)
2. 量子コンピュータの物理実装
3. 学生実験でやった素粒子実験







サークル名:661nos
次の COMITIA152 に、HoTT の解説漫画を出したい。
2025 年 6 月 1 日に東京ビッグサイトで会いましょう。

参考文献 I

この講演の内容は、主に [1] および [2] による。

[1] Egbert Rijke.
Introduction to Homotopy Type Theory.
arXiv:2212.11082, 2022.

[2] The Univalent Foundations Program.
Homotopy Type Theory: Univalent Foundations of Mathematics.
https://homotopytypetheory.org/book, Institute for Advanced Study, 2013.