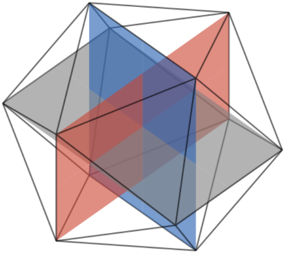
Daily applications of the univalence axiom



Egbert Rijke
University of Ljubljana
egbert.rijke@fmf-uni-lj.si

Part 1: Introduction

Type theory is a language for all mathematics

Structural approach to mathematics
Convenient for expressing mathematical concepts
Can be used for classical and constructive mathematics

Homotopy type theory is the internal language of ∞-toposes

Freudenthal suspension theorem

Van Kampen theorem

Blakers-Massey theorem

Serre spectral sequences

Hurewicz theorem

$\pi_4(S^3) = \mathbb{Z}/2$

Type theory can be used to discover new mathematics

Voevodsky famously discovered the univalence axiom by studying the type theory of Coq.

Many new applications of the univalence axiom are waiting to be discovered

Type theory is used in most big proof assistants

Lean
Coq
Agda
$\vdots$

Why formalizing mathematics?

It's an enjoyable thing to do
Digitization of mathematics
Synthesis of computation and formal reasoning
The computer can help
Instant feedback
Teaching

Anyone can join in!

Libraries for homotopy type theory

Coq-HoTT
https://github.com/HoTT/HoTT

Cubical Agda
https://github.com/agda/cubical

Agda Unimath
https://github.com/UniMath/agda-unimath

Community chat
https://hott.zulipchat.com

The Agda Unimath library

- Univalent mathematics for a general mathematical audience
- Online version with clickable code
- Files have the structure of nlab pages
- We are currently working on group theory, from the univalent perspective

Contributions are very welcome!

- 5 years behind Lean's `mathlib`. But: In the grand scheme of things that's nothing.
- The webpage can improve (to make it actually look like the nlab)
- We need much more algebra, analysis, topology, category theory, combinatorics, number theory, ...

Part 2: The univalent foundations for mathematics

Univalent foundation for mathematics

- Extension of Martin-Löf's dependent type theory with
  - Function extensionality
  - **Univalence axiom**
  - Propositional truncation
- Further extensions can be considered
  - **Propositional resizing**. This is important for the construction of the Dedekind reals.
  - **Higher inductive types**. Those are important for synthetic homotopy theory
  - Modal extensions, such as cohesive homotopy type theory.

The univalent foundations take a **structural** approach to mathematics, in a way that is compatible with both constructive and classical mathematics.

Dependent types

1. Dependent type theory is a language for mathematical structures and constructions.
2. The objects of type theory are types: $A, B, C, \ldots$
3. Types can have elements: $a, b, c, \ldots : A$
4. Types can depend on other types: $A(x_1, \ldots, x_n)$, where each x_i is a typed variable.

Constructions with types

- Given a type A and a family of types $B(x)$ indexed by $x : A$, we can form the **dependent pair type**

  $$\sum_{x:A} B(x).$$

  This type consists of pairs (x, y) where $x : A$ and $y : B(x)$.

- Given a type A and a family of types $B(x)$ indexed by $x : A$, we can form the **dependent function type**

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

  The elements are functions $\lambda x.f(x)$ taking as input $x : A$ and giving an output $f(x) : B(x)$.

The identity type

For any type A and any $x, y : A$, we can form the type of **identifications**

$$x = y.$$

- The identity type is inductively generated by

$$\mathsf{refl} : x = x.$$

- The induction principle asserts that for any family of types $P(y, e)$ indexed by $y : A$ and $e : x = y$, we have a function

$$J : P(x, \mathsf{refl}) \to \prod_{y:A} \prod_{e:x=y} P(y, e)$$

  defined by $J(p, x, \mathsf{refl}) :\equiv p$.

Groupoidal structure of types

- For each identification $p : x = y$ we can construct an inverse $p^{-1} : y = x$, because we have

  $$J : (x = x) \to \prod_{y:A} \prod_{p:x=y} y = x.$$

  Indeed, define $(\text{-})^{-1}$ by $\mathsf{refl}^{-1} := \mathsf{refl}$.

- For any two identifcations $p : x = y$ and $q : y = z$ we can construct the concatenation $p \cdot q$, because we have

  $$J : ((x = z) \to (x = z)) \to \prod_{y:A} \prod_{p:x=y} (y = z) \to (x = z)$$

  This lets us define $(\text{-}) \cdot q$ by $\mathsf{refl} \cdot q := q$.

- Using the induction principle of identifications, we can prove the groupoid laws, and higher coherences.

Homotopies and equivalences

- Given two functions $f : A \to B$, we define the type of homotopies
$$f \sim g := \prod_{x:A} f(x) = g(x).$$
- Give a map $f : A \to B$, we define the type of sections of f by
$$\mathrm{sec}(f) := \sum_{g:B\to A} f \circ g \sim \mathrm{id},$$
and we define the type of retractions of f by
$$\mathrm{retr}(f) := \sum_{g:B\to A} g \circ f \sim \mathrm{id}.$$
- We define is-equiv$(f) := \mathrm{sec}(f) \times \mathrm{retr}(f)$, and we define
$$A \simeq B := \sum_{f:A\to B} \text{is-equiv}(f)$$

Universes

In type theory, there are universes $\mathcal{U}$ that are closed under

$$\prod, \sum, =, \times, +, 0, 1, \mathbb{N}, \ldots$$

Universes are useful to

- Define dependent types over A, as maps $A \to \mathcal{U}$.
- Define types equipped with structure, such as
    - Pointed types
    - Groups
    - Rings
    - Categories
    - Spectra
    - $\ldots$

The univalence axiom

By identification elimination, we can also show that there is a map

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

for every $A, B : \mathcal{U}$.

The univalence axiom asserts that $\mathsf{equiv\text{-}eq}$ **is an equivalence, for every** $A, B : \mathcal{U}$

Contractible types

A type A is said to be **contractible** if it comes equipped with an element of type

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

Theorem

A map $f : A \to B$ is an equivalence if and only if for each $y : B$ the type

$$\text{fib}_f(b) := \sum_{x:A} f(x) = b$$

is contractible.

The fundamental theorem of identity types

Theorem

Consider a type A and an element $a : A$, and consider a type family B over A and an element $b : B(a)$. Then the following are equivalent:

1. *Any family of maps (in particular the canonical family of maps)*

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

   is a family of equivalences.

2. *The total space*

   $$\sum_{x:A} B(x)$$

   is contractible.

Propositions

A type A is said to be a **proposition** if it comes equipped with an element

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

We also define

$$\text{Prop} := \sum_{X:\mathcal{U}} \text{is-prop}(X).$$

1. is-equiv(f) is a proposition for any $f : A \to B$
2. is-contr(A) is a proposition for any type A.
3. is-prop(A) is a proposition for any type A.

Truncated types

Definition

A type A is said to be ***n*-truncated**, for $n \geq -2$ if it comes equipped with an element of type $\text{is-trunc}_n(A)$, which is defined recursively by

$$\begin{aligned} \text{is-trunc}_{-2}(A) &:= \text{is-contr}(A) \\ \text{is-trunc}_{n+1}(A) &:= \prod_{x,y:A} \text{is-trunc}_n(x = y). \end{aligned}$$

We say that A is a **set** if it is 0-truncated, i.e., if its identity types are propositions.

(Propositional) truncations

For any type A there is an n-truncated type $\|A\|_n$ equipped with a map $\eta : A \to \|A\|_n$, called the ***n*-truncation** of A, such that

$$(\|A\|_n \to X) \to (A \to X)$$

is an equivalence for every n-truncated type X.

$$\begin{tikzcd} A \arrow[d, "\eta"'] \arrow[dr, "f"] & \\ \|A\|_n \arrow[r, dashed] & X \end{tikzcd}$$

The **propositional truncation** of A is its (-1)-truncation. We often write $\|A\|$ for $\|A\|_{-1}$. The proposition $\|A\|$ expresses that A is inhabited.

Part 3: Univalent combinatorics

Finite types

Definition

The **standard finite types** $\mathrm{Fin}(n)$ are defined inductively on $n : \mathbb{N}$ by

$$\mathrm{Fin}(0) := \emptyset$$
$$\mathrm{Fin}(n+1) := \mathrm{Fin}(n) + \mathbf{1}$$

However, the **concept** of finite types is defined differently:

Definition

A type A is said to be finite if

$$\text{is-finite}(A) := \sum_{n:\mathbb{N}} \| Fin(n) \simeq A \|.$$

We define $\mathbb{F} := \sum X : \mathcal{U} \text{is-finite}(X)$.

n-element types

A type A has n elements if it is merely equivalent to $\mathrm{Fin}(n)$, i.e., if

$$\| Fin(n) \simeq A \|.$$

We define BS_n by

$$BS_n := \sum_{X:\mathcal{U}} \| Fin(n) \simeq X \|.$$

Theorem

For any $n : \mathbb{N}$ we have

$$\Omega BS_n \simeq (\mathrm{Fin}(n) \simeq \mathrm{Fin}(n)) =: \mathrm{Aut}(\mathrm{Fin}(n)).$$

Here $\Omega A := (a = a)$, for any pointed type A with $a : A$.

Proof.

If we prove that the type

$$\sum_{(X,H):\sum_{X:\mathcal{U}}\|\mathsf{Fin}(n)\simeq X\|} \mathsf{Fin}(n)\simeq X$$

is contractible, then we're done by the fundamental theorem. This type is equivalent to

$$\sum_{(X,e):\sum_{X:\mathcal{U}}\mathsf{Fin}(n)\simeq X} \|\mathsf{Fin}(n)\simeq X\|$$

The type of pairs (X, e) is contractible by the univalence axiom, and given $e:\mathsf{Fin}(n)\simeq X$ it follows that $\|\mathsf{Fin}(n)\simeq X\|$ is a true proposition, hence a contractible type. □

Pointed $(n+1)$-element types

Theorem (Buchholtz)

There is an equivalence

$$\left(\sum_{(X,H):BS_{n+1}} X\right) \simeq BS_n$$

Proof.

Let $(X, H) : BS_{n+1}$. For any $x : X$ we have

$$\{x\}^c := \sum_{y:X} x \neq y.$$

This is an n-element type. It follows that X is the unique pointed type (pointed by $x : X$), equipped with a pointed equivalence $X \simeq \{x\}^c + \mathbf{1}$. □

Corollary

We have an equivalence

$$\mathbb{F} \simeq \sum_{X:\mathbb{F}} X$$

In other words, the groupoid of finite sets is equivalent to the groupoid of pointed finite sets.

Corollary

The type

$$\sum_{X:BS_2} X$$

is contractible, so for any 2-element type X we have

$$(\mathrm{Fin}(2) \simeq X) \simeq X.$$

Corollary

There is **no** *dependent function*

$$\prod_{X:BS_2} X.$$

Proof.

If we had such a function, then we would obtain

$$\prod_{X:BS_2} \mathrm{Fin}(2) \simeq X.$$

By univalence, this would imply that BS_2 is contractible. However, identity types of contractible types are contractible, and the identity types of BS_2 are 2-element types. □

The previous corollary implies that we have to be somewhat careful about assuming the axiom of choice in univalent mathematics.

- **Inconsistent attempt.** For every family B of inhabited types over **any** type A, the type of sections of B is inhabited:

  $$\prod_{x:A} \|B(x)\| \to \left\| \prod_{x:A} B(x) \right\|$$

  This formulation of the axiom of choice is inconsistent by the example of BS_2.
- **Consistent formulation.** Restrict the inconsistent formulation to families of sets indexed by sets.

Decidable types

A type A is said to be **decidable** if it comes equipped with an element

$$\text{is-decidable}(A) := A + (A \to \emptyset)$$

Theorem

There is **no** *dependent function*

$$\prod_{(X:U)} X + (X \to \emptyset)$$

Proof.

Consider $X : BS_2$. Then the type $X \to \emptyset$ is empty. Indeed, we have the equivalences

$$(X \to \emptyset) \simeq (\|X\| \to \emptyset) \simeq \emptyset.$$

It follows that $(X + (X \to \emptyset)) \simeq X$, but there is no dependent function $\prod_{(X:BS_2)} X$. □

Isolated points

An element $x : X$ is said to be **isolated** if it comes equipped with a dependent function

$$\text{is-isolated}(x) := \prod_{y:X} (x = y) + (x \neq y).$$

Being isolated is a proposition, even if X is not a set. We write $\lfloor X \rfloor$ for the type of isolated points of X, i.e.,

$$\lfloor X \rfloor := \sum_{x:X} \text{is-isolated}(x)$$

Theorem

For any type A, we have an equivalence

$$\sum_{X:\mathcal{U}_{A+\mathbf{1}}} \lfloor X \rfloor \simeq \mathcal{U}_A.$$

Decidable embeddings

- A map $f : A \to B$ is said to be an **embedding** if it induces an equivalence
  $$(x = y) \simeq (f(x) \simeq f(y))$$
  for every $x, y : A$. We write $A \hookrightarrow B$ for the embeddings from A to B.
- A map is an embedding iff its fibers are propositions. Indeed, it is an embedding if and only if
  $$\mathrm{fib}_f(f(y)) \doteq \sum_{x:A} f(x) = f(y)$$
  is contractible for every $y : A$.
- A map $f : A \to B$ is said to be **decidable** if its fibers are decidable, i.e., if
  $$\mathrm{fib}_f(y)$$
  is decidable for every $y : B$. We write $A \hookrightarrow_d B$ for the decidable embeddings from A to B.

The binomial types

Definition

For any type $B : \mathcal{U}$, we define the **connected component** of $\mathcal{U}$ at B by

$$\mathcal{U}_B := \sum_{X:\mathcal{U}} \|B \simeq X\|.$$

Definition

For any two types A and B, we define the binomial type $\binom{A}{B}$ by

$$\binom{A}{B} := \sum_{X:\mathcal{U}_B} X \hookrightarrow_d A$$

The recursive laws of binomial types

Theorem

For any two types A and B, we have equivalences

$$\binom{\emptyset}{\emptyset} \simeq \mathbf{1} \qquad \binom{A+\mathbf{1}}{\emptyset} \simeq \mathbf{1}$$

$$\binom{\emptyset}{B+\mathbf{1}} \simeq \emptyset \qquad \binom{A+\mathbf{1}}{B+\mathbf{1}} \simeq \binom{A}{B} + \binom{A}{B+\mathbf{1}}.$$

Proof.

For any $f : X \hookrightarrow_d A + \mathbf{1}$ the type $\mathsf{fib}_f(*) + \neg\mathsf{fib}_f(*)$ is a true decidable proposition, so it is contractible. Therefore, we have equivalences

$$\begin{aligned}\binom{A+\mathbf{1}}{B+\mathbf{1}} &\simeq \sum_{X:\mathcal{U}_{B+\mathbf{1}}} \sum_{f:X\hookrightarrow_d A+\mathbf{1}} \mathsf{fib}_f(*) + \neg\mathsf{fib}_f(*) \\ &\simeq \left(\sum_{X:\mathcal{U}_{B+\mathbf{1}}} \sum_{f:X\hookrightarrow_d A+\mathbf{1}} \mathsf{fib}_f(*)\right) + \left(\sum_{X:\mathcal{U}_{B+\mathbf{1}}} \sum_{f:X\hookrightarrow A+\mathbf{1}} \neg\mathsf{fib}_f(*)\right)\end{aligned}$$

The right summand is $\binom{A}{B+\mathbf{1}}$. To compute the left summand, note that any $x : X$ such that $f(x) = *$ is an isolated point in X. Therefore, the left summand is equivalent to

$$\sum_{X:\mathcal{U}_B} \sum_{f:X+\mathbf{1}\hookrightarrow_d A+\mathbf{1}} f(*) = *,$$

which is equivalent to $\binom{A}{B}$. □

Corollary

If A and B are finite types with n and m elements, then $\binom{A}{B}$ is a finite type $\binom{n}{m}$ elements.

Proof.

By recursion on n and m. □

Part 4: Synthetic homotopy theory

Another way to think about BS_2 is as $\mathbb{R}\mathrm{P}^\infty$. The canonical family $(X, H) \mapsto X$ is the canonical line bundle over $\mathbb{R}\mathrm{P}^\infty$.

Definition

We construct $\mathbb{R}\mathrm{P}^n$ equipped with its line bundle $\gamma^n : \mathbb{R}\mathrm{P}^n \to \mathbb{R}\mathrm{P}^\infty$ inductively by $\mathbb{R}\mathrm{P}^0 := \mathbf{1}$ and we define $\mathbb{R}\mathrm{P}^{n+1}$ as a pushout:

$$\begin{tikzcd}
\sum_{(X:\mathbb{R}\mathrm{P}^n)} \gamma^n(X) \arrow[r] \arrow[d] & \sum_{(X:\mathbb{R}\mathrm{P}^\infty)} X \arrow[d] \arrow[ddr, bend left] & \\
\mathbb{R}\mathrm{P}^n \arrow[r] \arrow[drr, bend right, "\gamma^n"'] & \mathbb{R}\mathrm{P}^{n+1} \arrow[dr, dashed, "\gamma^{n+1}"'] & \\
 & & \mathbb{R}\mathrm{P}^\infty
\end{tikzcd}$$

The descent theorem

Theorem

Consider a commuting cube

$$\begin{tikzcd}
 & S' \arrow[dl] \arrow[d] \arrow[dr] & \\
A' \arrow[d] \arrow[dr] & S \arrow[dl] \arrow[dr] & B' \arrow[dl] \arrow[d] \\
A \arrow[dr] & X' \arrow[d] & B \arrow[dl] \\
 & X &
\end{tikzcd}$$

in which the vertical squares in the back are pullback and the bottom square is a pushout. Then the following are equivalent:

1. *The top square is a pushout.*
2. *The vertical squares in the front are pullback.*

Descent is false in the category of sets

Consider the commuting cube

$$\begin{tikzcd}
 & \mathbf{2}\times\mathbf{2} \arrow[dl, "\pi_1"'] \arrow[d, "\mu"'] \arrow[dr, "\pi_2"] & \\
\mathbf{2} \arrow[d] \arrow[dr] & \mathbf{2} \arrow[dl] \arrow[dr] & \mathbf{2} \arrow[dl] \arrow[d] \\
\mathbf{1} \arrow[dr] & \mathbf{1} \arrow[d] & \mathbf{1} \arrow[dl] \\
 & \mathbf{1} &
\end{tikzcd}$$

Where μ is the group operation on $\mathbf{2}$. Then the squares in the back are pullback and the top and bottom squares are pushout, but the squares in the front are not pullback.

Theorem

The type $\sum_{X:\mathbb{R}\mathrm{P}^n} \gamma^n(X)$ is equivalent to S^n. Consequently, we obtain a fiber sequence

$$\mathrm{Fin}(2) \hookrightarrow S^n \twoheadrightarrow \mathbb{R}\mathrm{P}^n.$$

Proof.

$\mathbb{R}\mathrm{P}^0 = \mathbf{1}$, so the base case follows. For the inductive step, consider the commuting cube

$$\begin{tikzcd}
 & & \left(\sum_{X:\mathbb{R}\mathrm{P}^n} \gamma^n(X)\right) \times \mathbf{2} \arrow[dll] \arrow[d] \arrow[drr] & & \\
\sum_{X:\mathbb{R}\mathrm{P}^n} \gamma^n(X) \arrow[d] \arrow[drr, crossing over] & & \sum_{X:\mathbb{R}\mathrm{P}^n} \gamma^n(X) \arrow[dll] \arrow[drr] & & \mathbf{2} \arrow[dll, crossing over] \arrow[d] \\
\mathbb{R}\mathrm{P}^n \arrow[drr] & & \sum_{X:\mathbb{R}\mathrm{P}^{n+1}} \gamma^{n+1}(X) \arrow[d] & & \mathbf{1} \arrow[dll] \\
 & & \mathbb{R}\mathrm{P}^{n+1} & &
\end{tikzcd}$$

The bottom square is pushout and the front two squares are pullback. The square on the back left is also pullback, so the remaining square in the back is pullback. (The proof is continued on the blackboard)

The construction of $\mathbb{R}\mathrm{P}^n$ is an instance of a more general construction: For any two maps $f : A \to X$ and $g : B \to X$, we can define a map $f * g : A *_X B \to X$ as a pushout of the pullback:

$$\begin{tikzcd}
A \times_X B \arrow[r] \arrow[d] & B \arrow[d] \arrow[ddr, bend left] & \\
A \arrow[r] \arrow[drr, bend right] & A *_X B \arrow[dr, dashed, "f*g"'] & \\
 & & X
\end{tikzcd}$$

Theorem

For each $x : X$ we have $\mathrm{fib}_{f*g}(x) \simeq \mathrm{fib}_f(x) * \mathrm{fib}_g(x)$.

The complex projective spaces

The infinite complex projective space $\mathbb{C}\mathrm{P}^\infty$ can be defined as

$$\mathbb{C}\mathrm{P}^\infty :\equiv \|S^2\|_2.$$

The type family $x \mapsto \mathrm{pt} = x$ is a bundle of (oriented) circles over $\mathbb{C}\mathrm{P}^\infty$. This is its tautological bundle.

Definition

We define $\mathbb{C}\mathrm{P}^n$ equipped with a map $\gamma^n : \mathbb{C}\mathrm{P}^n \to \mathbb{C}\mathrm{P}^\infty$ inductively by taking $\mathbb{C}\mathrm{P}^0 :\equiv \mathbf{1}$, and

$$\mathbb{C}\mathrm{P}^{n+1} :\equiv \mathbb{C}\mathrm{P}^n *_{\mathbb{C}\mathrm{P}^\infty} \mathbf{1}$$

and $\gamma^{n+1} :\equiv \gamma^n * \mathrm{pt}$.

We obtain a fiber sequence

$$\begin{tikzcd} S^1 \arrow[r, hook] & S^{2n+1} \arrow[r, two heads] & \mathbb{C}\mathrm{P}^n \end{tikzcd}$$

We don't know how to define the quaternionic projective spaces

Open problem: To deloop the 3-sphere in HoTT.

Part 5: Abstract and concrete groups

Concrete groups

We have seen that the type of all n-element sets encodes the group of all symmetries of n-element sets via

$$\Omega BS_n \simeq S_n.$$

The idea is now that the group of symmetries of an object of kind X is the type of all objects of kind X:

1. The dihedral groups are presented by the types of all polygons,
2. Hyperoctahedral groups are presented by the types of all n-cubes,
3. The Rubik's cube group is the type of all (valid configurations of) Rubik's cube,
4. ...and so on.

All of these describe **pointed connected 1-types**.

Definition

A type A is said to be **connected** if its set truncation $\|A\|_0$ is contractible.

Definition

- A **(concrete) ∞-group** is a pointed connected type.
- A **(concrete) *n*-group** is a pointed connected n-type.
- A **(concrete) group homomorphism** is a pointed map.

Given an n-group/∞-group BG, we define $G :\equiv \Omega BG$. This has the structure of an (abstract) n-group/∞-group by the operations on paths.

Our goal is to construct a concrete 1-group from an abstract 1-group, i.e., to construct the Eilenberg-Mac Lane space $K(G, 1)$

Concrete group actions

Given an ∞-group BG, a ***G*-type** is just a map

$$X : BG \to \mathcal{U}$$

The type being acted on is $X(*)$, and the action on $X(*)$ by an element $g : G := \Omega BG$ is given by transport:

$$\mathrm{tr}_X(g) : X(*) \simeq X(*).$$

We often abuse notation and write X for $X(*)$.

Example

For any type B recall that $\mathrm{BAut}(B) := \mathcal{U}_B$, the connected component of $\mathcal{U}$ at B. For any two types A and B, we saw the $\mathrm{Aut}(B)$-set

$$(X : \mathcal{U}_B) \mapsto (X \hookrightarrow_d A).$$

Given a G-type $X : BG \to \mathcal{U}$, we define

$$X/\!/G := \sum_{g:G} X(g)$$

Note that for $g : G := \Omega BG$ and $x : X(*)$ we have an identification

$$(*, x) = (*, gx)$$

by the path lifting property. So we see that elements in the same orbit of the G-action on X are identified in $X/\!/G$.

Example

For the $\mathrm{Aut}(B)$-set $B \hookrightarrow_d A$ we find that

$$\binom{A}{B} := (B \hookrightarrow_d A)/\!/\mathrm{Aut}(B).$$

Two embeddings $B \hookrightarrow_d A$ are identified in $\binom{A}{B}$ if they are the same up to permutation of elements in B.

Stirling types of the second kind

Consider two types A and B, and consider the $\mathrm{Aut}(B)$-type

$$X \mapsto (A \twoheadrightarrow X),$$

where $A \twoheadrightarrow X$ is the type of surjective maps from A to X. Then we define the **Stirling type of the second kind** by

$$\left\{ \begin{matrix} A \\ B \end{matrix} \right\} := (A \twoheadrightarrow B)/\!/\mathrm{Aut}(B)$$

Theorem
If A and B are finite types with n and m elements, then the Stirling type of the second kind $\left\{ {A \atop B} \right\}$ is a finite type with $\left\{ {n \atop m} \right\}$ elements.

Abstract group actions

Recall that for an abstract group G, a G-set consists of a set X equipped with a group homomorphism

$$\mu : \mathrm{Grp}(G, \mathrm{Aut}(X)).$$

A morphism of G-sets from X to Y consists of a map $f : X \to Y$ such that for every $g : G$ the square

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, "\mu_G(g)"'] & Y \arrow[d, "\mu_Y(g)"] \\
X \arrow[r, "f"'] & Y
\end{tikzcd}$$

commutes. A morphism $f : X \to Y$ of G-sets is an equivalence if the underlying map $f : X \to Y$ is an equivalence.

Univalence for abstract group actions

Theorem

For any two G-sets (X, μ_X) and (Y, μ_Y), the map

$$(X, \mu_X) = (Y, \mu_Y) \xrightarrow{\alpha} (X, \mu_X) \simeq_G (Y, \mu_Y)$$

is an equivalence. Furthermore, for any two identifications

$$(X, \mu_X) \overset{p}{=\!=\!=\!=} (Y, \mu_Y) \overset{q}{=\!=\!=\!=} (Z, \mu_Z)$$

we have $\alpha(p \cdot q) \sim \alpha(q) \circ \alpha(p)$.

Proof.

By the fundamental theorem, it suffices to show that

$$\sum_{(Y,\mu_Y):G\text{-set}} (X,\mu_X) \simeq_G (Y,\mu_Y)$$

is contractible. Note that $\sum_{Y:\mathcal{U}} X \simeq Y$ is contractible by univalence, and the type

$$\sum_{(\mu_Y:\mathrm{Grp}(G,\mathrm{Aut}\,X))} \prod_{(g:G)} \mathrm{id}\circ\mu_G \sim \mu_Y \circ \mathrm{id}$$

is contractible by function extensionality.

The last claim follows since $\alpha(\mathrm{refl}) := (\mathrm{id}, \mathrm{refl})$, and moreover we have $\mathrm{refl} \cdot q := q$, and composition of equivalences satisfies the right unit law. □

Delooping abstract groups

Definition

The **principal G-set** P_G on G is the left action of G on itself. We define

$$BG := G\text{-torsor} := \sum_{X : G\text{-set}} \| P_G \simeq_G X \|.$$

Theorem

For any group G, we have a group isomorphism $\Omega BG \cong G$.

Proof.

By the previous theorem we already know that

$$\Omega BG \cong (P_G \simeq_G P_G).$$

Therefore it suffices to show that $(P_G \simeq_G P_G) \cong G$.

Proof (Continued).

We claim that the map $(P_G \simeq_G P_G) \to G$ given by $(e, H) \mapsto e(1)$ is an equivalence.

- The inverse $G \to (P_B \simeq_G P_G)$ sends g to the equivalence $- \cdot g$, which preserves the G-action by associativity of G.
- Any (e, H) is of this form because

$$e(g) \xlongequal{\quad} e(g \cdot 1) \xlongequal{H(g,1)} g \cdot e(1).$$

- Finally, composites are preserved because

$$(f \circ e)(1) = f(e(1)) = f(e(1) \cdot 1) = e(1) \cdot f(1).$$

for any $(e, H), (f, K) : P_G \simeq_G P_G$.

We conclude that $(P_G \simeq_G P_G) \cong G$. □

Delooping abstract group homomorphisms (1)

To figure out how to deloop abstract group homomorphisms, we first look at the concrete case: For any pointed map $Bf : BG \to_* BH$ we have a map

$$Bf_! : (BG \to \mathcal{U}) \to (BH \to \mathcal{U})$$

given by

$$Bf_!(X, y) := \sum_{x : BG} (Bf(x) = y) \times X(x)$$

In the type $Bf_!(X, *)$, the identifications $(*, h, x) = (*, h', x')$ are equivalently described as triples

$$(g, p, q)$$

consisting of $g : G$, $p : f(g)h = h'$ and $q : gx = x'$.

Delooping abstract group homomorphisms (2)

Given a G-torsor X we therefore define the H-set

$$f_! X := (H, X)/\sim$$

where $(h, x) \sim (h', x')$ if and only if there exists an element $g : G$ such that $f(g)h = h'$ and $gx = x'$. Since X is a G-torsor, there is a most one such g. The H-action is given by $h[(y, x)] := [(hy, x)]$.

Theorem

The H-set $f_! X$ is an H-torsor.

Proof.

Being an H-torsor is the proposition $\| P_H \simeq_H f_! X \|$. By the universal property of propositional truncation applied to the assumption $\| P_G \simeq_G X \|$, it suffices to show that $f_! P_G$ is a principal H-torsor. We claim that

$$f_! P_G \simeq_H P_H.$$

Proof (Continued).

To see that $(H \times G)/\sim \simeq H$, note that the elements of the form $(h, 1)$ are canonical representatives of the $\sim$-equivalence classes. Indeed, for each (h, g) we have

$$(h, g) \sim (f(g^{-1})h, 1)$$

and H acts on $f_! P_G$ simply by multiplication on the left. □

We have therefore constructed a map

$$Bf : G\text{-torsor} \to H\text{-torsor}.$$

Take-away: **The category of abstract groups is equivalent to the category of concrete groups, i.e., the category of pointed connected 1-types.**

Part 6: The number of groups of order n, the univalent way

- Up to isomorphism, there are only finitely many groups of order n.
- However, those groups may have nontrivial automorphisms, so the type of groups of order n is not expect to be finite. Finite types are sets, and the type of groups of order n is a 1-type.
- How can univalence help us to count up to isomorphism?
- How can homotopy theory help?

Definition

A type A is said to be **π_n-finite** if it comes equipped with an element of type is-π_n-finite(A), which is defined recursively by

$$\begin{aligned} \text{is-}\pi_0\text{-finite}(A) &:= \text{is-finite}\|A\|_0 \\ \text{is-}\pi_{n+1}\text{-finite}(A) &:= \text{is-finite}\|A\|_0 \times \prod_{x,y:A} \text{is-}\pi_n\text{-finite}(x = y) \end{aligned}$$

In other words: a type is π_n-finite if it has finitely many connected components and all its homotopy groups $\pi_i(A, x)$ are finite, for every $i \leq n$ and every $x : A$.

Theorem

Consider a family B of π_n-finite types over a finite type A, then the product

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

is π_n-finite.

Lemma

Consider a family B of π_0-finite types over a connected π_1-finite type A. Then the type

$$\textstyle\sum_{(x:A)} B(x)$$

is π_0-finite.

Proof.

- We're proving a property and we assumed that A is connected. Therefore we may assume that A is pointed by $a : A$.
- Since A is assumed to be connected, the fiber inclusion

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

  is surjective.

Proof.

- Set truncation preserves surjectiveness, so

  $$\|B(a)\|_0 \to \|\sum_{(x:A)} B(x)\|_0$$

  is surjective.
- This is a surjective map out of a finite type. Classically it follows that the codomain is finite, and we would not need the π_1-finiteness of A. But for us it remains to **show that the codomain has decidable equality**.
- Proof continues on the blackboard.

We now prove π_n-finiteness and get rid of the connectedness assumption.

Theorem

Consider a family B of π_n-finite types indexed by a π_{n+1}-finite type A. Then the type

$$\sum_{x:A} B(x)$$

is π_n-finite.

Note: Classically we would only need that A is π_n-finite.

Proof.

- By induction on the number of connected components.
- By induction on n. □

Theorem

The type of groups of order n is π_0-finite.

Proof.

- The type of n-element types is π_1-finite.
- The type of group structures on an n-element type is finite.

Therefore the previous theorem applies. □