Set-level mathematics

Joj Ahrens

UniMath School, Birmingham, UK, April 2019

Outline

Outline

Definition of set

$$\mathrm{iscontr}(X) := \sum_{x:X} \prod_{y:X} x = y$$

$$\mathrm{isaprop}(X) := \prod_{x,y:X} x = y$$

$$\mathrm{isaset}(X) := \prod_{x,y:X} \mathrm{isaprop}(x = y)$$

A **set** is a type whose path types are all *propositions*.

Definition of h-Levels

$$\begin{aligned}
\mathsf{isofhlevel}(n, X) &: \mathsf{Nat} \to \mathcal{U} \to \mathsf{hProp} \\
\mathsf{isofhlevel}(0, X) &:= \mathsf{iscontr}(X) \\
\mathsf{isofhlevel}(S(n), X) &:= \prod_{x, x' : X} \mathsf{isofhlevel}(n, x = x')
\end{aligned}$$

Definition of h-Levels

$$\begin{aligned}\text{isofhlevel}(n, X) &: \text{Nat} \to \mathcal{U} \to \text{hProp} \\ \text{isofhlevel}(0, X) &:= \text{iscontr}(X) \\ \text{isofhlevel}(S(n), X) &:= \prod_{x,x':X} \text{isofhlevel}(n, x = x')\end{aligned}$$

Exercise

$$\left(\prod_{X:\mathcal{U}} \text{isaprop}(X)\right) = \text{isofhlevel}(1, X)$$

Definition of h-Levels

$$\begin{aligned} \mathrm{isofhlevel}(n, X) &: \mathrm{Nat} \to \mathcal{U} \to \mathrm{hProp} \\ \mathrm{isofhlevel}(0, X) &:= \mathrm{iscontr}(X) \\ \mathrm{isofhlevel}(S(n), X) &:= \prod_{x, x' : X} \mathrm{isofhlevel}(n, x = x') \end{aligned}$$

Exercise

$$\left(\prod_{X : \mathcal{U}} \mathrm{isaprop}(X) \right) = \mathrm{isofhlevel}(1, X)$$

Hence:

A **set** is a type of hlevel 2.

Topological intuition

Recall:

- Each type X is to be a thought of a *space*
- For two points $a, b : X$, the type $a =_X b$ is the *space of paths* from a to b

Fact (from topology)

A ("nice") space all of whose path spaces are contractible is (homotopy-)equivalent to a discrete space.

Topological intuition

Recall:

- Each type X is to be a thought of a *space*
- For two points $a, b : X$, the type $a =_X b$ is the *space of paths* from a to b

Fact (from topology)

A ("nice") space all of whose path spaces are contractible is (homotopy-)equivalent to a discrete space.

This can be made more precise using the *model in simplicial sets*.

Closure properties

- $\sum_{x:A} B(x)$ is a set if A and all $B(x)$ are
- $A \times B$ is a set if A and B are
- $\prod_{x:A} B(x)$ is a set if all $B(x)$ are
- $A \to B$ is a set if B is

- Any property is a set

Closure properties

- $\sum_{x:A} B(x)$ is a set if A and all $B(x)$ are
- $A \times B$ is a set if A and B are
- $\prod_{x:A} B(x)$ is a set if all $B(x)$ are
- $A \to B$ is a set if B is

- Any property is a set

Exercise

Do you know

- a type that is a set?
- a type for which you don't know (yet) whether it is a set?

What are sets good for?

- Most "traditional" mathematics can be done in UniMath with sets (groups, rings, topological spaces, etc.)

What are sets good for?

- Most "traditional" mathematics can be done in UniMath with sets (groups, rings, topological spaces, etc.)
- Categories or the type of *all* groups (rings, etc.) have h-level 3.

What are sets good for?

- Most “traditional” mathematics can be done in UniMath with sets (groups, rings, topological spaces, etc.)
- Categories or the type of *all* groups (rings, etc.) have h-level 3.
- Higher category theory and synthetic homotopy theory require higher types.

Outline

Decidable equality

Definition

A type X is **decidable** if we can write a term of type

$$X + \neg X$$

Definition

A type X has **decidable path-equality** if we can write a term of type

$$\prod_{x, x' : A} (x = x') + \neg(x = x')$$

(that is, if all its paths types are decidable)

Hedberg's theorem

Theorem

If a type X has decidable equality, then it is a set.

In the problem session, we will show that Bool and Nat are sets.

Hedberg's theorem

Theorem

If a type X has decidable equality, then it is a set.

In the problem session, we will show that Bool and Nat are sets.

Note

Hedberg's theorem is *hard*. There is also an easier proof that Bool and Nat are sets.

Are all types sets?

Is there a type that is not a set?

Great question! It depends:

- In "spartan" type theory some types can't be shown to be sets.
- Assuming univalence, some types can be shown not to be sets.

Another set

Theorem

The type

$$\mathsf{hProp}_{\mathcal{U}} :\equiv \sum_{X:\mathcal{U}} \mathsf{isaprop}(X)$$

is a set.

The proof relies on the univalence axiom for the unviverse $\mathcal{U}$.

Exercise

How would you generalize the above statement to any h-level?
How would you attempt proving it?

Types that are **not** sets

Exercise

Let $\mathcal{U}$ be a univalent universe that contains the type Bool. Why is $\mathcal{U}$ not a set?

Which property of Bool does the proof of the above result exploit?

Outline

Sets and propositions

Types representing properties of sets are usually propositions.

Sets and propositions

Types representing properties of sets are usually propositions.

Example

Given $f : X \to Y$,

$$\mathrm{isInjective}(f) :\equiv \prod_{x,x':X} f(x) = f(x') \to x = x'$$

is not a proposition in general, but it is if X and Y are sets.

Predicates on types

A **subtype** A on a type X is a map

$$A : X \to \mathsf{hProp}_U$$

Exercise

Show that the type of subtypes of any type X is a set.

Predicates on types

A **subtype** A on a type X is a map

$$A : X \to \mathrm{hProp}_U$$

Exercise

Show that the type of subtypes of any type X is a set.

The **carrier** of a subtype A is the type of elements satisfying A:

$$\mathrm{carrier}(A) := \sum_{x:X} A(x)$$

Predicates and injections

There is a canonical map $\mathrm{incl}_A : \mathrm{carrier}(A) \to X$

Predicates and injections

There is a canonical map $\mathsf{incl}_A : \mathsf{carrier}(A) \to X$, namely

$$pr1 : \sum_{x:X} A(X) \to X$$

Predicates and injections

There is a canonical map $\mathrm{incl}_A : \mathrm{carrier}(A) \to X$, namely

$$pr1 : \sum_{x:X} A(X) \to X$$

Exercise

$$\mathrm{isaset}(X) \to \mathrm{isInjective}(\mathrm{incl}_A)$$

Predicates and injections

Conversely, given a map $f : A \to X$, we can form the function $\chi_f : X \to \mathcal{U}$ given by

$$\chi_f(x) :\equiv \sum_{a:A} f(a) = x$$

Predicates and injections

Conversely, given a map $f : A \to X$, we can form the function $\chi_f : X \to \mathcal{U}$ given by

$$\chi_f(x) :\equiv \sum_{a:A} f(a) = x$$

Exercise

$$\mathsf{isaset}(X) \to \mathsf{isInjective}(f) \to \prod_{x:X} \mathsf{isaprop}(\chi_f(x))$$

Predicates and injections

Conversely, given a map $f : A \to X$, we can form the function $\chi_f : X \to \mathcal{U}$ given by

$$\chi_f(x) :\equiv \sum_{a:A} f(a) = x$$

Exercise

$$\mathsf{isaset}(X) \to \mathsf{isInjective}(f) \to \prod_{x:X} \mathsf{isaprop}(\chi_f(x))$$

(Hard) exercise

ξ_f and incl_A establish an isomorphism between $X \to \mathsf{hProp}_{\mathcal{U}}$ and $\mathsf{injections}(X) :\equiv \sum_{A:\mathcal{U}} \mathsf{isaset}(A) \times \sum_{f:A\to X} \mathsf{isInjective}(f)$

Relations on a type

A **binary relation** R on a type X is a map

$$R : X \to X \to \mathsf{hProp}_U$$

Relations on a type

A **binary relation** R on a type X is a map

$$R : X \to X \to \mathsf{hProp}_U$$

Exercise

Show that the type of binary relations on X is a set.

Relations on a type

A **binary relation** R on a type X is a map

$$R : X \to X \to \mathrm{hProp}_U$$

Exercise

Show that the type of binary relations on X is a set.

Properties of such relations are defined as usual, e.g.,

$$\mathrm{reflexive}(R) := \prod_{x:X} R(x)(x)$$

Relations on a type

A **binary relation** R on a type X is a map

$$R : X \to X \to \mathrm{hProp}_U$$

Exercise

Show that the type of binary relations on X is a set.

Properties of such relations are defined as usual, e.g.,

$$\mathrm{reflexive}(R) := \prod_{x:X} R(x)(x)$$

Exercise

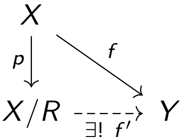
Formulate the properties of being symmetric, transitive, an equivalence relation.

Set-level quotient

Given a set X and relation R on X, the **quotient**

$$X \xrightarrow{p} X/R$$

is defined by the property that any compatible map f **into a set** Y factors uniquely through p:



Exercise

Formulate this condition precisely.

The quotient set

We can define the quotient X/R of a set by an equivalence relation as the set of equivalence classes.

The quotient set

We can define the quotient X/R of a set by an equivalence relation as the set of equivalence classes.
First we define for a subtype $A : X \to \mathrm{hProp}_U$

$$\begin{aligned}\mathrm{iseqclass}(A) := &\ \|\mathrm{carrier}(A)\| \\ &\times \prod_{x,y:A} Rxy \to Ax \to Ay \\ &\times \prod_{x,y:A} Ax \to Ay \to Rxy\end{aligned}$$

The quotient set

We can define the quotient X/R of a set by an equivalence relation as the set of equivalence classes.
First we define for a subtype $A : X \to \mathrm{hProp}_U$

$$\begin{aligned}\mathrm{iseqclass}(A) :=\, & \|\mathrm{carrier}(A)\| \\ & \times \prod_{x,y:A} Rxy \to Ax \to Ay \\ & \times \prod_{x,y:A} Ax \to Ay \to Rxy\end{aligned}$$

Then we define

$$X/R := \sum_{A:X \to \mathrm{hProp}_U} \mathrm{iseqclass}(A)$$

Outline

Paths between pairs

Given $B : A \to \mathcal{U}$ and $a, a' : A$ and $b : B(a)$ and $b' : B(a')$,

$$(a, b) = (a', b') \quad \simeq \quad \sum_{p : a = a'} \mathsf{transport}^B\Big(p,\ b\Big) = b'$$

If $B(x)$ is a proposition for any $x : A$, then this simplifies to

$$(a, b) = (a', b') \quad \simeq \quad a = a'$$

Paths between pairs

Given $B : A \to \mathcal{U}$ and $a, a' : A$ and $b : B(a)$ and $b' : B(a')$,

$$(a, b) = (a', b') \quad \simeq \quad \sum_{p:a=a'} \mathsf{transport}^B\Big(p,\ b\Big) = b'$$

If $B(x)$ is a proposition for any $x : A$, then this simplifies to

$$(a, b) = (a', b') \quad \simeq \quad a = a'$$

Exercise

Why?

Example: natural numbers

An **even natural number** is a pair consisting of a natural number and a proof of its evenness.

$$\mathsf{iseven}(n) :\equiv \sum_{k:\mathsf{Nat}} k + k = n \qquad \mathsf{evennat} :\equiv \sum_{n:\mathsf{Nat}} \mathsf{iseven}(n)$$

Example: natural numbers

An **even natural number** is a pair consisting of a natural number and a proof of its evenness.

$$\mathrm{iseven}(n) :\equiv \sum_{k:\mathrm{Nat}} k + k = n \qquad \mathrm{evennat} :\equiv \sum_{n:\mathrm{Nat}} \mathrm{iseven}(n)$$

When comparing two even natural numbers, we want to compare them as numbers:

$$(n, p) = (n', p') \quad \simeq \quad n = n'$$

Example: natural numbers

An **even natural number** is a pair consisting of a natural number and a proof of its evenness.

$$\mathrm{iseven}(n) :\equiv \sum_{k:\mathrm{Nat}} k + k = n \qquad \mathrm{evennat} :\equiv \sum_{n:\mathrm{Nat}} \mathrm{iseven}(n)$$

When comparing two even natural numbers, we want to compare them as numbers:

$$(n, p) = (n', p') \quad \simeq \quad n = n'$$

The type $\mathrm{iseven}(n)$ hence should be a proposition.

Example: natural numbers

An **even natural number** is a pair consisting of a natural number and a proof of its evenness.

$$\mathrm{iseven}(n) :\equiv \sum_{k:\mathrm{Nat}} k + k = n \qquad \mathrm{evennat} :\equiv \sum_{n:\mathrm{Nat}} \mathrm{iseven}(n)$$

When comparing two even natural numbers, we want to compare them as numbers:

$$(n, p) = (n', p') \quad \simeq \quad n = n'$$

The type iseven(n) hence should be a proposition.

Exercise

It is!

Groups

Traditionally (in set theory), a group is a quadruple (G, m, e, i) of

- a set G
- a multiplication $m : G \times G \to G$
- a unit $e \in G$
- an inverse $i : G \to G$

subject to the usual axioms.

Groups in type theory

In type theory, a group is a (dependent) pair $(data, proof)$ where

- *data* is a quadruple (G, m, e, i) as above
- p is a proof that these satisfy the usual axioms.

Groups in type theory

In type theory, a group is a (dependent) pair $(data, proof)$ where

- $data$ is a quadruple (G, m, e, i) as above
- p is a proof that these satisfy the usual axioms.

We want to regard two groups $(data, proof)$ and $(data', proof')$ as being the same if $data$ is the same as $data'$.

Groups in type theory

In type theory, a group is a (dependent) pair $(data, proof)$ where

- *data* is a quadruple (G, m, e, i) as above
- p is a proof that these satisfy the usual axioms.

We want to regard two groups $(data, proof)$ and $(data', proof')$ as being the same if *data* is the same as $data'$.

This requires that the type encoding the group axioms be a *proposition*.

Groups in type theory

In type theory, a group is a (dependent) pair $(data, proof)$ where

- *data* is a quadruple (G, m, e, i) as above
- p is a proof that these satisfy the usual axioms.

We want to regard two groups $(data, proof)$ and $(data', proof')$ as being the same if *data* is the same as $data'$.

This requires that the type encoding the group axioms be a *proposition*.

This is in turn guaranteed as long as the underlying type G is required to be a *set*.

Exercise

Why?

Group isomorphisms

The type of groups is

$$\mathsf{Grp} := \sum_{X:\mathsf{hSet}} \mathsf{GrpStructure}(X)$$

A **group isomorphism** $G \to G'$ is

- a bijective function on the underlying sets $X \to X'$
- compatible with the group structures S and S' on X and X'.

Identity is isomorphism for groups

$$
\begin{aligned}
G = G' &\simeq (X, S) = (X', S') \\
&\simeq \sum_{p : X = X'} \mathrm{transport}^{\mathrm{GrpStructure}}(p, S) = S' \\
&\simeq \sum_{p : X = X'} (\mathrm{transport}^{Y \mapsto (Y \times Y \to Y)}(p, m) = m') \\
&\qquad\quad \times (\mathrm{transport}^{Y \mapsto (Y \to Y)}(p, i) = i') \\
&\qquad\quad \times (\mathrm{transport}^{Y \mapsto (1 \to Y)}(p, e) = e') \\
&\simeq \sum_{f : X \simeq X'} \left(f \circ m \circ (f^{-1} \times f^{-1}) = m'\right) \\
&\qquad\quad \times \left(f \circ i \circ f^{-1} = i'\right) \\
&\qquad\quad \times (f \circ e = e') \\
&\simeq (G \cong G')
\end{aligned}
$$