Lessons from Formalizing (Higher) Category Theory in UniMath

Benedikt Ahrens

jww Niels van der Weide

Coq Workshop 2024

2024–09–14

Summary

What are we(*) doing?

1. Develop a library of category theory
2. In the UniMath library based on Coq
3. Based on univalent foundations

Challenges and observations

1. Modularity: how to reuse stuff?
2. The (un)importance of strictness
3. Transporting results along equivalences

(*) Besides Ahrens and Van der Weide, several others have been involved: Frumin, Lafont, Van der Leer, Lumsdaine, Maggesi, Matthes, Mörtberg, Wullaert,...

Outline

What is a category?

A category C consists of

1. A type C_0 of objects
2. For any $a, b : \mathsf{C}_0$, a type $\mathsf{C}(a, b)$ of morphisms from a to b
3. Composition $\mathsf{C}(a, b) \to \mathsf{C}(b, c) \to \mathsf{C}(a, c)$
4. Identity $\mathsf{C}(a, a)$
5. Laws: unitality and associativity of composition

Examples

1. Sets and functions
2. Groups and group homomorphisms
3. Types and terms of STLC

Outline

Reusing stuff

Reusing stuff in mathematics

“Composition of group homomorphisms is given by composition of the underlying functions.”

Reusing stuff in computer formalization

- Algebraic hierarchies
- Add fields to extend to a structure with more operations or properties

Question

In category theory, we consider collections of all things and morphisms between them. What is a suitable mathematical structure for “adding fields” to objects and morphisms?

Displayed categories

Definition (Displayed category)

Let C be a category. A displayed category D over C is given by

1. for every object $x : \mathsf{C}_0$, a type D_x of displayed objects over x
2. for every morphism $f : \mathsf{C}(x, y)$ and objects $\bar{x} : D_x$ and $\bar{y} : D_y$, a type $\mathsf{D}(\bar{x}, f, \bar{y})$ of displayed morphisms over f from $\bar{x}$ to $\bar{y}$
3. composition and identity of displayed morphisms
4. laws

Definition (Total category)

Any displayed category D induces a total category $\int \mathsf{D}$ and a functor $\int \mathsf{D} \to \mathsf{C}$.

Example: displayed category of groups

The displayed category of group structures over the category of sets:

- Objects over set X are group structures on X
- Morphisms over $f : X \to Y$ from G_X to G_Y are proofs that f is a homomorphism from G_X to G_Y

Total category

is the category of groups, with a forgetful functor to sets.

Example: displayed category of topologies

The displayed category of topologies over the category of sets:

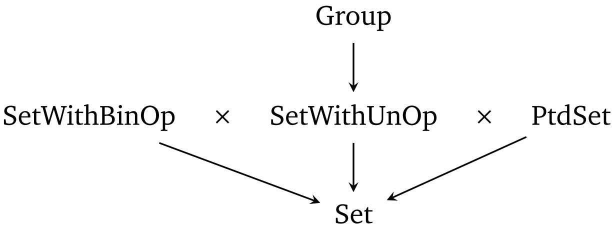
- Objects over set X are topologies on X
- Morphisms over $f : X \to Y$ from T_X to T_Y are proofs that f is a continuous map from T_X to T_Y

Total category

is the category of topological spaces, with a forgetful functor to sets.

Displayed categories, layered

Using displayed categories, we can construct categories in a modular way:



Summary: displayed categories

1. We use displayed categories for modular constructions of categories, by layering many displayed categories
2. Structure on total category can be obtained from structure on base and “displayed” structure on displayed category
3. Same principle works for *higher* categories, such as bicategories
4. Literature:
   - *Displayed categories*, Ahrens, Lumsdaine
   - *Bicategories in univalent foundations*, Ahrens, Frumin, Maggesi, Veltri, Van der Weide
   - *Univalent monoidal categories*, Ahrens, Matthes, Wullaert

Outline

What do “strict” and “weak” mean?

A categorical structure is

strict when it preserves objects up to **equality**.

weak when it preserves objects up to **isomorphism**.

Example

A monoidal category C has a binary operation $\otimes : \mathsf{C} \times \mathsf{C} \to \mathsf{C}$ and a unit $I : \mathsf{C}_o$. It is called

strict when $X \otimes I = X = I \otimes X$

weak when $X \otimes I \cong X \cong I \otimes X$

Strictness versus weakness

In set-theoretic mathematics

- Strict structures are **more convenient**, because they are easier to use
- One does not lose generality: weak structures are equivalent to strict ones in many cases

Suppose we have

$$X \xrightarrow{f} Y \otimes I \quad Y \xrightarrow{g} Z$$

Can we write their composition $f \cdot g$?

Strictness versus weakness

In set-theoretic mathematics

- Strict structures are **more convenient**, because they are easier to use
- One does not lose generality: weak structures are equivalent to strict ones in many cases

Suppose we have

$$X \xrightarrow{f} Y \otimes I \quad Y \xrightarrow{g} Z$$

Can we write their composition $f \cdot g$?

- Set theory: sure, just write $f \cdot g$
- Type theory: **no**, $Y \otimes I$ and Y would need to be **convertible**

This is a consequence of intensional equality.

Monoidal categories in intensional foundations

Instead you would write something like

$$X \xrightarrow{\quad f \quad} Y \otimes I \xrightarrow{\quad \rho \quad} Y \xrightarrow{\quad g \quad} Z$$

where ρ shows that $Y \otimes I = Y$
So:

- In essence, we are working with weak structures
- The advantages of strict structures evaporate

Reflecting on weakness versus strictness

- In intensional type theory, strict structures do not offer simplifications compared to strict structures
- It is natural to work with weak structures: bicategories instead of 2-categories, weak monoidal categories instead of strict ones,. . .
- In concrete examples, the additional bureaucracy is often trivial (i.e., given by identity isos)

Outline

Introduction to univalent foundations

Key features of univalent foundations

- Identity types are interpreted as ∞-groupoid structure
- Univalence axiom: identity of types is equivalence of types
- Univalence principle: identity of structures is the same as equivalence of structures

Example (Univalence principle for groups)

$$(G_1 = G_2) \simeq (G_1 \cong G_2) \quad \text{for groups } G_1 \text{ and } G_2$$

Example (Univalence principles for categories)

$$(\mathsf{C}_1 = \mathsf{C}_2) \simeq (\mathsf{C}_1 \cong \mathsf{C}_2) \quad \text{for set-categories } \mathsf{C}_1 \text{ and } \mathsf{C}_2$$
$$(\mathsf{C}_1 = \mathsf{C}_2) \simeq (\mathsf{C}_1 \simeq \mathsf{C}_2) \quad \text{for univalent categories } \mathsf{C}_1 \text{ and } \mathsf{C}_2$$

Univalence principle for set-categories

Isomorphism of categories

An isomorphism from C_1 to C_2 consists of functors $F : \mathsf{C}_1 \to \mathsf{C}_2$ and $G : \mathsf{C}_2 \to \mathsf{C}_1$ such that $F \cdot G = \mathsf{id}$ and $G \cdot F = \mathsf{id}$ (**strict**)

Set-categories

In a setcategory, the type of objects is a set (identity types are subsingletons).

Theorem

$$(\mathsf{C}_1 = \mathsf{C}_2) \simeq (\mathsf{C}_1 \cong \mathsf{C}_2) \quad \textit{for set-categories } \mathsf{C}_1 \textit{ and } \mathsf{C}_2$$

Examples: syntactic categories of type theories

Univalence principle for univalent categories

Equivalence of categories

Equivalence from C_1 to C_2 consists of functors $F : \mathsf{C}_1 \to \mathsf{C}_2$ and $G : \mathsf{C}_2 \to \mathsf{C}_1$ such that $F \cdot G \simeq \mathsf{id}$ and $G \cdot F \simeq \mathsf{id}$ (**weak**)

Univalent categories

In a univalent category, identity of objects $a = b$ is the same as isomorphism $a \cong b$.

Theorem

$$(\mathsf{C}_1 = \mathsf{C}_2) \simeq (\mathsf{C}_1 \simeq \mathsf{C}_2) \quad \textit{for univalent categories } \mathsf{C}_1 \textit{ and } \mathsf{C}_2$$

Examples: the categories of sets, groups, rings

Transport of structure along sameness of categories

Transporting along isomorphisms

Given **setcategories** C_1 and C_2 and an **isomorphism** between them, every structure on C_1 can be transported to C_2.

Transporting along adjoint equivalences

Given **univalent categories** C_1 and C_2 and an **adjoint equivalence** between them, every structure on C_1 can be transported to C_2.

E.g., have an easy proof that adjoint equivalence preserves being Cartesian closed.

Summary: formalizing category theory in (univalent) type theory

1. Displayed categories can provide **modular** constructions
2. Strict categorical structures are not as useful as in set theory; it is more natural to work with **weak** categorical structures
3. Univalent foundations give us tools to reason formally modulo equivalence of categories

Thanks to the Coq team for support and patience!

Thanks to you for listening!