Emily Riehl

Johns Hopkins University

Formalizing invisible mathematics:
case studies from higher category theory

iiD Colloquium

An apology

When an idea for the title of this talk popped into my head, I had the good sense to Google it and found the following:

> Andrej Bauer, "Formalizing invisible mathematics," IPAM, February 2023

His very informative talk describes

- the elaboration step that translates between the vernacular language employed by the user of a proof assistant and the core formal language that is verified by the kernel;
- the challenges of interoperability presented by logically equivalent definitions or bi-interpretable foundation systems;
- as well as an aspect of invisible mathematics in the sense that I'll use the term here.

From pen-and-paper to computer formalization

Many unforeseen challenges in translating a pen-and-paper proof to a computer formalized proof arise from the

<div align="center">

invisible mathematics

</div>

that emerges along the way.

For the purposes of this talk, we propose the following definition:

> **Definition.** By invisible mathematics, we refer to any mathematics that is not visible in a pen-and-paper proof but becomes visible in a computer formalized proof.

A pen-and-paper proof

A joint book with Dominic Verity reviews the construction of the reflective embedding of 1-categories into ∞-categories in less than one page:
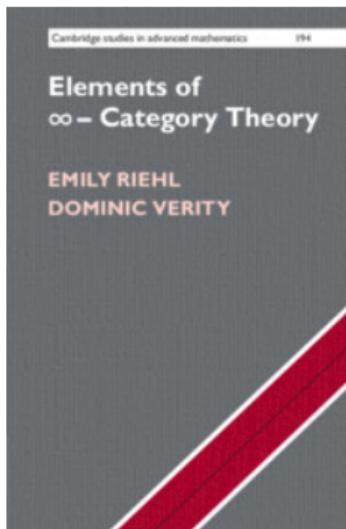
1.1.10. DEFINITION (the homotopy category [44, §2.4]). By 1-truncating, any simplicial set X has an underlying reflexive directed graph with the 0-simplices of X defining the objects and the 1-simplices defining the arrows:

$$X_1 \underset{\xrightarrow{\delta^0}}{\overset{\delta^1}{\underset{\longrightarrow}{\longleftarrow}}} X_0,$$

By convention, the source of an arrow $f \in X_1$ is its 0th face $f \cdot \delta^1$ (the face opposite 1) while the target is its 1st face $f \cdot \delta^0$ (the face opposite 0). The **free category** on this reflexive directed graph has X_0 as its object set, degenerate 1-simplices serving as identity morphisms, and nonidentity morphisms defined to be finite directed paths of nondegenerate 1-simplices. The **homotopy category** hX of X is the quotient of the free category on its underlying reflexive directed graph by the congruence[3] generated by imposing a composition relation $h = g \circ f$ witnessed by 2-simplices

$$\begin{array}{ccc} & x_1 & \\ {}^{f}\nearrow & & \searrow^{g} \\ x_0 & \xrightarrow[h]{} & x_2 \end{array}$$

This relation implies in particular that homotopic 1-simplices represent the same arrow in the homotopy category.

The homotopy category of the nerve of a 1-category is isomorphic to the original category, as the 2-simplices in the nerve witness all of the composition relations satisfied by the arrows in the underlying reflexive directed graph. Indeed, the natural isomorphism $hC \cong C$ forms the counit of an adjunction, embedding Cat as a reflective subcategory of $sSet$.

1.1.11. PROPOSITION. *The nerve embedding admits a left adjoint, namely the functor which sends a simplicial set to its homotopy category:*

$$Cat \underset{\underset{\longleftarrow}{\perp}}{\overset{h}{\longleftarrow}} sSet$$

The adjunction of Proposition 1.1.11 exists for formal reasons (see Exercise 1.1.i), but nevertheless, a direct proof can be enlightening.

PROOF. For any simplicial set X, there is a natural map from X to the nerve of its homotopy category hX; since nerves are 2-coskeletal, it suffices to define the map $sk_2 X \to hX$, and this is given immediately by the construction of Definition 1.1.10. Note that the quotient map $X \to hX$ becomes an isomorphism upon applying the homotopy category functor and is already an isomorphism whenever X is the nerve of a category. Thus the adjointness follows from Lemma B.4.2 or by direct verification of the triangle equalities. □

A formalized proof

It took three months (part time) of joint work with Mario Carneiro to formalize this result in Lean.

It then took another six months for this code, which totalled 2240 lines split across seven PRs to pass the review process to be integrated into Lean's Mathlib.

We wrote an 16 page paper to explain the formalization and the challenges we encountered along the way:

Formalizing colimits in $\mathcal{C}at$

Mario Carneiro ✉ 🆔
Chalmers University of Technology, Sweden

Emily Riehl[1] ✉ 🆔
Department of Mathematics, Johns Hopkins University, 3400 N Charles Street, Baltimore, MD, USA

— **Abstract** —
Certain results involving "higher structures" are not currently accessible to computer formalization because the prerequisite ∞-category theory has not been formalized. To support future work on formalizing ∞-category theory in Lean's mathematics library, we formalize some fundamental constructions involving the 1-category of categories. Specifically, we construct the left adjoint to the nerve embedding of categories into simplicial sets, defining the homotopy category functor. We prove further that this adjunction is reflective, which allows us to conclude that $\mathcal{C}at$ has colimits. To our knowledge this is the first formalized proof that the category of categories is cocomplete.

In summary, there was an $16\times$ scaling factor from the original pen-and-paper proof to a pen-and-paper account of the formalization.

What happened here?

Plan

1. A taxonomy of invisible mathematics

2. Prospects for formalizing ∞-category theory

$\left(1 \right)$

A taxonomy of invisible mathematics

Design choices in definitions

Like many mathematical definitions, a category involves:

- some "stuff"
- equipped with "structure"
- satisfying "properties".

DEFINITION 1.1.1. A **category** consists of
- a collection of **objects** $X, Y, Z, \ldots$
- a collection of **morphisms** $f, g, h, \ldots$

so that:
- Each morphism has specified **domain** and **codomain** objects; the notation $f \colon X \to Y$ signifies that f is a morphism with domain X and codomain Y.
- Each object has a designated **identity morphism** $\mathrm{id}_X \colon X \to X$.
- For any **composable pair** of morphisms f, g, with the codomain of f equal to the domain of g, there exists a specified **composite morphism**[4] gf whose domain is equal to the domain of f and whose codomain is equal to the codomain of g, i.e.:

$$f \colon X \to Y, \quad g \colon Y \to Z \qquad \rightsquigarrow \qquad gf \colon X \to Z.$$

This data is subject to the following two axioms:
- For any $f \colon X \to Y$, the composites $\mathrm{id}_Y f$ and $f \mathrm{id}_X$ are both equal to f.
- For any composable triple of morphisms f, g, h, the composites $h(gf)$ and $(hg)f$ are equal and henceforth denoted by hgf.

$$f \colon X \to Y, \quad g \colon Y \to Z, \quad h \colon Z \to W \qquad \rightsquigarrow \qquad hgf \colon X \to W.$$

But there is an essential design choice required to encode this definition in dependent type theory, the formal system used by Lean:

Are the morphisms a closed type or a dependent type over the type of objects?
That is, do we prefer $f : \mathrm{mor}$ or $f : X \to Y$ for $X, Y : \mathrm{ob}$?

Confusions of notation/encoding

Category theory famously eats itself:

- A category C consists of objects A, B, C and arrows $f\colon A \to B$, $g\colon B \to C$ with composition and identities satisfying axioms.
- Categories themselves assemble into a category Cat whose objects are categories and whose morphisms are functors.

In `Mathlib`, categories are typically unbundled, as types with a category structure, while objects of Cat are bundled categories. Functors come with

- two different notations whether they are thought of as functors between categories or arrows in Cat ("$\Rightarrow$" vs "$\longrightarrow$") and
- distinct notations for composition ("$\ggg$" vs "$\gg$") and identities ("$\mathbf{1}$" vs "$\mathbb{1}$") in each setting.

Confusions of notation/encoding

Mario Carneiro and I formalized the free category and underlying reflexive quiver adjunction

$$F$$
$$\text{RefIQuiv} \quad \bot \quad \text{Cat}.$$
$$U$$
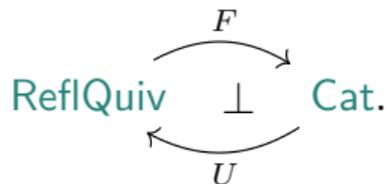
But the construction of the adjunction was repeatedly hindered by confusions about the encoding of categorical data.

```
left_triangle := by
  ext V
  apply Cat.FreeRefl.lift_unique'
  simp only [id_obj, Cat.free_obj, comp_obj, Cat.freeRefl_obj_α, NatTrans.comp_app,
    forget_obj, whiskerRight_app, associator_hom_app, whiskerLeft_app, id_comp,
    NatTrans.id_app']
  rw [Cat.id_eq_id, Cat.comp_eq_comp]
  simp only [Cat.freeRefl_obj_α, Functor.comp_id]
  rw [← Functor.assoc, ← Cat.freeRefl_naturality, Functor.assoc]
  dsimp [Cat.freeRefl]
  rw [adj.counit.component_eq' (Cat.FreeRefl V)]
  conv =>
    enter [1, 1, 2]
    apply (Quiv.comp_eq_comp (X := Quiv.of _) (Y := Quiv.of _) (Z := Quiv.of _) ..).symm
  rw [Cat.free.map_comp]
  show (_ ≫ ((Quiv.forget ≫ Cat.free).map (X := Cat.of _) (Y := Cat.of _)
    (Cat.FreeRefl.quotientFunctor V))) ≫ _ = _
  rw [Functor.assoc, ← Cat.comp_eq_comp]
  conv => enter [1, 2]; apply Quiv.adj.counit.naturality
  rw [Cat.comp_eq_comp, ← Functor.assoc, ← Cat.comp_eq_comp]
  conv => enter [1, 1]; apply Quiv.adj.left_triangle_components V.toQuiv
  exact Functor.id_comp _
```

Universe levels

On paper size issues in category theory are typically addressed in a hand wavy fashion:[1]

REMARK 1.1.5. Russell's paradox implies that there is no set whose elements are "all sets." This is the reason why we have used the vague word "collection" in Definition 1.1.1. Indeed, in each of the examples listed in 1.1.3, the collection of objects is not a set. Eilenberg and Mac Lane address this potential area of concern as follows:

> ... the whole concept of a category is essentially an auxiliary one; our basic concepts are essentially those of a *functor* and of a natural transformation .... The idea of a category is required only by the precept that every function should have a definite class as domain and a definite class as range, for the categories are provided as the domains and ranges of functors. Thus one could drop the category concept altogether and adopt an even more intuitive standpoint, in which a functor such as "Hom" is not defined over the category of "all" groups, but for each particular pair of groups which may be given. [**EM45**]

The set-theoretical issues that confront us while defining the notion of a category will compound as we develop category theory further. For that reason, common practice among category theorists is to work in an extension of the usual Zermelo–Fraenkel axioms of set theory, with new axioms allowing one to distinguish between "small" and "large" sets, or between sets and classes. The search for the most useful set-theoretical foundations for category theory is a fascinating topic that unfortunately would require too long of a digression to explore.[12] Instead, we sweep these foundational issues under the rug, not because these issues are not serious or interesting, but because they distract from the task at hand.[13]

---

[1] Footnote 13 begins "If pressed, let us assume that there exists a countable sequence of inaccessible cardinals, ..."

Universe levels

Of course this doesn't work in proof assistant and over a year's worth of attempts to formalize some category theory in Lean I find myself drawn between two poles:

- On the one hand, I'm embarrassed by how poorly I understand the implications of various categorical constructions on universe levels.
- On the other, I am frustrated by how often universe errors "distract from the task at hand" and feel increasingly drawn to the dark side of type-in-type.

There are two open questions related to practical treatment of universes that I would like to answer:

- What are the best strategies for dealing with universe errors that arise during formalization, both with the aim of resolving them and with the aim of not letting them get in the way of other progress?
- How should universes be addressed in the pen-and-paper literature to better prepare category theorists for formalization, i.e., how should I address this in *Category Theory in Context* volume II?

Invisible inclusions

A simplicial set is a functor $X\colon \Delta^{\mathsf{op}} \Rightarrow \mathsf{Type}$, indexed by a category Δ whose objects are natural numbers.

> Some constructions require only the data of a 2-truncated simplicial set, indexed by the subcategory $\Delta_{\leq 2} \subset \Delta$ spanned by the objects 0, 1, 2.

On paper, $\Delta_{\leq 2}$ has just three objects — the natural numbers 0, 1, and 2 — and then has hom-types inherited from Δ.

In `Mathlib`, objects of $\Delta_{\leq 2}$ are natural numbers m together with a proof that $m \leq 2$.

> This causes all sorts of problems.

Invisible inclusions

Proofs involving $\Delta_{\leq 2}$ often require rewriting along an equality between arrows that has been proven for Δ. Since `rw` typically fails, we are forced to duplicated infrastructure:

```
/-- Abbreviations for face maps in the 2-truncated simplex category. -/
abbrev δ₂ {n} (i : Fin (n + 2)) (hn := by decide) (hn' := by decide) :
    (([n], hn) : SimplexCategory.Truncated 2) ⟶ ([n + 1], hn') := SimplexCategory.δ i

/-- Abbreviations for degeneracy maps in the 2-truncated simplex category. -/
abbrev σ₂ {n} (i : Fin (n + 1)) (hn := by decide) (hn' := by decide) :
    (([n+1], hn) : SimplexCategory.Truncated 2) ⟶ ([n], hn') := SimplexCategory.σ i

@[reassoc (attr := simp)]
lemma δ₂_zero_comp_σ₂_zero {n} (hn := by decide) (hn' := by decide) :
    δ₂ (n := n) 0 hn hn' ≫ σ₂ 0 hn' hn = 𝟙 _ := SimplexCategory.δ_comp_σ_self

@[reassoc]
lemma δ₂_zero_comp_σ₂_one : δ₂ (0 : Fin 3) ≫ σ₂ 1 = σ₂ 0 ≫ δ₂ 0 :=
    SimplexCategory.δ_comp_σ_of_le (i := 0) (j := 0) (Fin.zero_le _)

@[reassoc (attr := simp)]
lemma δ₂_one_comp_σ₂_zero {n} (hn := by decide) (hn' := by decide) :
    δ₂ (n := n) 1 hn hn' ≫ σ₂ 0 hn' hn = 𝟙 _ := SimplexCategory.δ_comp_σ_succ

@[reassoc (attr := simp)]
lemma δ₂_two_comp_σ₂_one : δ₂ (2 : Fin 3) ≫ σ₂ 1 = 𝟙 _ := SimplexCategory.δ_comp_σ_succ' (by decide)

@[reassoc]
lemma δ₂_two_comp_σ₂_zero : δ₂ (2 : Fin 3) ≫ σ₂ 0 = σ₂ 0 ≫ δ₂ 1 :=
    SimplexCategory.δ_comp_σ_of_gt' (by decide)
```

Dependent equality and "evil"

> Universal properties characterize objects of a category only up to (unique) isomorphism. Categorical statements that refer to an equality between objects are known as "evil."

Equality between parallel functors $F, G \colon \mathsf{C} \Rightarrow \mathsf{D}$ is certainly evil, though strangely the "evil" equality between objects is the easiest part to formalize.

On paper, $F = G$ just when

- $FX = GX$ for all objects X in C, and
- $Ff = Gf$ for all arrows $f \colon X \to Y$ in C.

But `Mathlib`, the second statement does not type check: it is not meaningful to ask whether Ff equals Gf because $Ff \colon FX \to FY$ and $Gf \colon GX \to GY$. Even if we have proofs $hX \colon FX = GX$ and $hY \colon FY = GY$, these are different types.

Dependent equality and "evil"

Using an evil hypothesis h_obj : ∀X, F.obj X = G.obj X involving parallel functors F, G: C ⇒ D, there are various ways to conclude that $F = G$:

```
/-- Proving equality between functors. This isn't an extensionality lemma,
  because usually you don't really want to do this. -/
theorem ext {F G : C ⇒ D} (h_obj : ∀ X, F.obj X = G.obj X)
    (h_map : ∀ X Y f,
      F.map f = eqToHom (h_obj X) ≫ G.map f ≫ eqToHom (h_obj Y).symm := by aesop_cat) :
    F = G := by
```

```
/-- Proving equality between functors using heterogeneous equality. -/
theorem hext {F G : C ⇒ D} (h_obj : ∀ X, F.obj X = G.obj X)
    (h_map : ∀ (X Y) (f : X → Y), HEq (F.map f) (G.map f)) : F = G :=
```

```
lemma ext_of_iso {F G : C ⇒ D} (e : F ≅ G) (hobj : ∀ X, F.obj X = G.obj X)
    (happ : ∀ X, e.hom.app X = eqToHom (hobj X)) : F = G :=
```

None of these are particularly fun to use,
as they involve arguments that are totally invisible on paper.

Coherence theorems

On paper, a coherence theorem tells us that 2-cells in the 2-category Cat compose by pasting, generalization the primitive operations of vertical composition and whiskering:

In `Mathlib`, the 2-cells displayed here belong to dependent types (over their boundary 1-cells and objects).

Depending on parenthesizations, 2-cells that are composable on paper are not composable in `Lean` as the 1-cells along their common boundary are not definitionally equal:

$$\text{e.g., is } R_3 H_2 L_2 \eta_2 G_1 R_1 \text{ composable with } R_3 H_2 \epsilon_2 L_2 G_1 R_1 ?$$

Coherence theorems

is $R_3 H_2 L_2 \eta_2 G_1 R_1$
composable
with $R_3 H_2 \epsilon_2 L_2 G_1 R_1$?

Lean has a clever composition operation for 2-cells in a bicategory:

```
/-- Construct an isomorphism between two objects in a bicategorical category
out of unitors and associators. -/
abbrev bicategoricalIso (f g : a ⟶ b) [BicategoricalCoherence f g] : f ≅ g :=
  ⊗𝟙


/-- Compose two morphisms in a bicategorical category,
inserting unitors and associators between as necessary. -/
def bicategoricalComp {f g h i : a ⟶ b} [BicategoricalCoherence g h]
    (η : f ⟶ g) (θ : h ⟶ i) : f ⟶ i :=
  η ≫ ⊗𝟙.hom ≫ θ
```

but no normal form for whiskered 2-cells or pasting diagram composites.

Coherence theorems

```
/-- The mates equivalence commutes with vertical composition. -/
theorem mateEquiv_vcomp (α : g₁ ≫ l₂ ⟶ l₁ ≫ h₁) (β : g₂ ≫ l₃ ⟶ l₂ ≫ h₂) :
    mateEquiv adj₁ adj₃ (leftAdjointSquare.vcomp α β) =
      rightAdjointSquare.vcomp (mateEquiv adj₁ adj₂ α) (mateEquiv adj₂ adj₃ β) := by
  dsimp only [leftAdjointSquare.vcomp, mateEquiv_apply, rightAdjointSquare.vcomp]
  symm
  calc
    _ = 𝟙 _ ⊛≫ r₁ ◁ g₁ ◁ adj₂.unit ▷ g₂ ⊛≫ r₁ ◁ α ▷ r₂ ▷ g₂ ⊛≫
        ((adj₁.counit ▷ (h₁ ≫ r₂ ≫ g₂ ≫ 𝟙 e)) ≫ 𝟙 b ◁ (h₁ ◁ r₂ ◁ g₂ ◁ adj₃.unit)) ⊛≫
        h₁ ◁ r₂ ◁ β ▷ r₃ ⊛≫ h₁ ◁ adj₂.counit ▷ h₂ ▷ r₃ ⊛≫ 𝟙 _ := by
      bicategory
    _ = 𝟙 _ ⊛≫ r₁ ◁ g₁ ◁ adj₂.unit ▷ g₂ ⊛≫
        (r₁ ◁ (α ▷ (r₂ ≫ g₂ ≫ 𝟙 e) ≫ (l₁ ≫ h₁) ◁ r₂ ◁ g₂ ◁ adj₃.unit)) ⊛≫
        ((adj₁.counit ▷ (h₁ ≫ r₂) ▷ (g₂ ≫ l₃) ≫ (𝟙 b ≫ h₁ ≫ r₂) ◁ β) ▷ r₃) ⊛≫
        h₁ ◁ adj₂.counit ▷ h₂ ▷ r₃ ⊛≫ 𝟙 _ := by
      rw [← whisker_exchange]
      bicategory
    _ = 𝟙 _ ⊛≫ r₁ ◁ g₁ ◁ (adj₂.unit ▷ (g₂ ≫ 𝟙 e) ≫ (l₂ ≫ r₂) ◁ g₂ ◁ adj₃.unit) ⊛≫
        (r₁ ◁ (α ▷ (r₂ ≫ g₂ ≫ l₃) ≫ (l₁ ≫ h₁) ◁ r₂ ◁ β) ▷ r₃) ⊛≫
        (adj₁.counit ▷ h₁ ▷ (r₂ ≫ l₂) ≫ (𝟙 b ≫ h₁) ◁ adj₂.counit) ▷ h₂ ▷ r₃ ⊛≫ 𝟙 _ := by
      rw [← whisker_exchange, ← whisker_exchange]
      bicategory
    _ = 𝟙 _ ⊛≫ r₁ ◁ g₁ ◁ g₂ ◁ adj₃.unit ⊛≫
        r₁ ◁ g₁ ◁ (adj₂.unit ▷ (g₂ ≫ l₃) ≫ (l₂ ≫ r₂) ◁ β) ▷ r₃ ⊛≫
        r₁ ◁ (α ▷ (r₂ ≫ l₂) ≫ (l₁ ≫ h₁) ◁ adj₂.counit) ▷ h₂ ▷ r₃ ⊛≫
        adj₁.counit ▷ h₁ ▷ h₂ ▷ r₃ ⊛≫ 𝟙 _ := by
      rw [← whisker_exchange, ← whisker_exchange, ← whisker_exchange]
      bicategory
    _ = 𝟙 _ ⊛≫ r₁ ◁ g₁ ◁ g₂ ◁ adj₃.unit ⊛≫ r₁ ◁ g₁ ◁ β ▷ r₃ ⊛≫
        ((r₁ ≫ g₁) ◁ leftZigzag adj₂.unit adj₂.counit ▷ (h₂ ≫ r₃)) ⊛≫
        r₁ ◁ α ▷ h₂ ▷ r₃ ⊛≫ adj₁.counit ▷ h₁ ▷ h₂ ▷ r₃ ⊛≫ 𝟙 _ := by
      rw [← whisker_exchange, ← whisker_exchange]
      bicategory
    _ = _ := by
      rw [adj₂.left_triangle]
      bicategory
```

A formal proof by Yuma Mizuno leveraged his bicategory tactic to prove an equality between the previous pasting composite and a reduced form (with the whiskered composite of η_2 and ϵ_2 replaced by an identity).

But his proof required a lot of intermediate calculation — specifying a particular sequence of presentations of the pasted composite as a vertical composite of whiskered 2-cells — that ideally would be automated.

Equivalent definitions

> …la Mathématique est l'art de donner le même nom à des choses différentes.
> …mathematics is the art of giving the same name to different things.
>
> — Henri Poincaré, "L'avenir des mathématiques," *Science et Méthode*

To an Australian category theorist at least, there is no difference between the following three concepts found in `Mathlib`:

- a Cat-enriched category, meaning
- a Cat-enriched ordinary category, meaning
- a strict bicategory, meaning

Our PR constructing a strict bicategory (and thus also a bicategory) from a Cat-enriched category was delayed because we had not also formalized the corresponding result for Cat-enriched ordinary categories, nor the inverse constructions.

A 2-simplex in a simplicial set
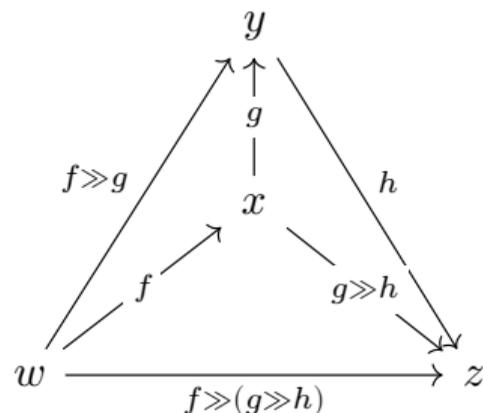
$$\begin{array}{c} y \\ f \nearrow \quad \searrow g \\ \sigma \\ x \xrightarrow{\ f \gg g\ } z \end{array}$$
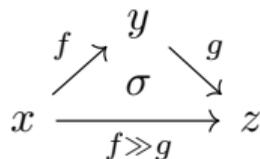
witnesses that the diagonal edge is a composite of the edges along its spine.

In a quasi-category, inner horns can be filled to form simplices, which witness the associativity of composition:

There is an inner horn built from the three faces witnessing the composites $f \gg g$, $g \gg h$, and $f \gg (g \gg h)$. In the resulting simplex, the back face witnesses that the diagonal edge $f \gg (g \gg h)$ also defines a composite $(f \gg g) \gg h$.

Invisible details
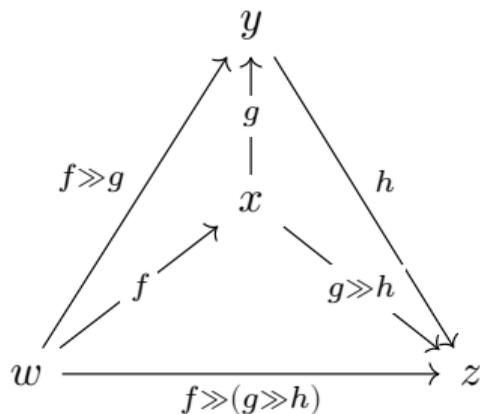
Technically, the inner horn is the subcomplex of the 3-simplex defined by gluing the three 2-simplices along their four common vertices and three common edges.

This data requires almost 200 lines to specify (written by Julian Komaromy) and requires three open PRs (written by Joël Riou) to demonstrate the universal property of such diagrams, that a multicofork is a multicoequalizer.

Thus we have no proof of associativity of composition in a quasi-category
— and the full statement and proof is more complicated than this.

Invisible expertise

Because the simplex category Δ is a strict Reedy category, there is a well understood procedure for extending a map $f \colon X \to Y$ of $n-1$-truncated simplicial sets to a map of n-truncated simplicial sets:

(i) define the new component: $f_n \colon X_n \to Y_n$, a function carrying n-simplices of X to n-simplices of Y, and

(ii) check that the map f_n respects degeneracies and faces.

If desired, each of the checks mentioned in (ii) can be encoded as a single commutative square, one involving a colimit forming the latching object of degenerate simplices and the other involving a limit forming the matching object of simplex face data.

These conditions are both necessary and sufficient.

Invisible expertise

`Mathlib` does not know any Reedy category theory. Do we digress to formalize this general abstract nonsense or instead just check that the given family of maps

$$f_0 \colon X_0 \to Y_0, \quad f_1 \colon X_1 \to Y_1, \quad ..., \quad f_{n-1} \colon X_{n-1} \to Y_{n-1}, \quad f_n \colon X_n \to Y_n$$

defines a natural transformation in the case of interest, checking a lot of unnecessary naturality conditions?

In order to define the unit of the adjunction
between the nerve functor and the homotopy category functor,
we had to extend a map of 1-truncated simplicial sets
to a map of 2-truncated simplicial sets,
which requires 31 naturality checks.
Mario Carneiro and I essentially did this by hand,
splitting into 9 cases, parametrized by pairs of objects.[2]

---

[2]While integrating this into `Mathlib`, Joël Riou provided infrastructure that allowed us to reduce to four cases, though there should have just been two.

Imprecise statements and muddy thinking

Finally, some complications with paper proofs are "invisible" not because they ought to be, but because the proof writer is not thinking clearly.

In this case, formalization — as a proxy for explaining the proof carefully to a critical human listener — is extremely helpful in clearing through the mud and cobwebs.

See Sophie Morel's talk "Lean for the Research Mathematician" for an excellent description of this.

A taxonomy of invisible mathematics?

Should the microscopic details concerning

- design choices,
- notation/encoding,
- universe levels,
- inclusions and coercions,
- dependent equality,
- coherence theorems,
- equivalent definitions,
- technical constructions,
- mathematical theories, or
- precise statements

remain invisible or be brought out into the light?

Opinions vary, depending on who is being asked ...

2

Prospects for formalizing ∞-category theory

What does invisible mathematics mean for big proofs?

Invisible mathematics causes:

- annoyances for domain experts who don't know how to handle the invisible formalization infrastructure;
- high barriers to entry for non-domain experts because of all of the invisible mathematical details that are seldom explained in the literature.

To the latter point, it is challenging to write a blueprint that (i) anticipates infrastructural challenges and (ii) makes all of the mathematical details visible.

> What does this mean for large formalization projects?

> Is there another way?

The ∞-cosmos project: overview

The ∞-cosmos project aims to use a convenient abstraction boundary to formalize some core category theory of ∞-categories for immediate use in `Mathlib`.

`Mathlib` knows definitions of

- ∞-groupoid, that is a weak infinite-dimensional category with all morphisms invertible, modeled as Kan complexes
- ∞-category, that is a weak infinite-dimensional category with non-invertible morphisms only allowed in dimension one, modeled as quasi-categories

But very few theorems have been formalized about these notions. In particular, earlier this year, Joël Riou noticed that the definition of Kan complexes was wrong!

The ∞-cosmos project: status

A paper I co-wrote in summer 2023 included some fighting words:

> In addition to homological algebra, Lean's mathematics library `mathlib` [80] contains standard results from number theory, representation theory, general topology, linear algebra including Banach and Hilbert spaces, measures and integral calculus, random variables, basic algebraic geometry, model theory, and category theory, among other topics. Despite all these achievements, as of the writing of this article, Lean's `mathlib` does not contain any ∞-category theory, and

Nikolai Kudasov, Emily Riehl, and Jonathan Weinberger. 2024. Formalizing the ∞-Categorical Yoneda Lemma. In *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP '24), January 15–16, 2024, London, UK*. ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3636501.3636945

Two plus years of work later, `Mathlib` now has one theorem about quasi-categories:

Construction: There is a strict bicategory of ∞-categories, ∞-functors, and ∞-natural transformations (using quasi-categories).

Corollary: Existing `Mathlib` results about adjunctions in a bicategory specialize to provide a theory of adjunctions between ∞-categories.

A much harder result — that the internal hom into a quasi-category is a quasi-category — was formalized by Jack McKoen and is currently being PR-ed.

Is there another way?

The Origins and Motivations of Univalent Foundations

A Personal Mission to Develop Computer Proof Verification to Avoid Mathematical Mistakes

By Vladimir Voevodsky • Published 2014

"A technical argument by a trusted author, which is hard to check and looks similar to arguments known to be correct, is hardly ever checked in detail."

Voevodsky's solution to the problem of invisible mathematics was to create a new foundation system to facilitate efficient synthetic reasoning about higher structures.

Simplicial HoTT and synthetic ∞-categories

> **ℹ Info**
>
> This project originated as a fork of ○ emilyriehl/yoneda.

This is a formalization library for simplicial Homotopy Type Theory (sHoTT) with the aim of proving resulting in synthetic ∞-category theory, starting with the results from the following papers:

- "A type theory for synthetic ∞-categories" [1]
- "Synthetic fibered (∞,1)-category theory" [2]
- "Limits and colimits of synthetic ∞-categories" [3]

This formalization project follows the philosophy laid out in the article "Could ∞-category theory be taught to undergraduates?" [4].

The formalizations are implemented using `rzk`, an experimental proof assistant for a variant of type theory with shapes.

Formalizing the ∞-categorical Yoneda lemma

The ∞-categorical Yoneda lemma is out of scope of the ∞-cosmos project for the foreseeable future, but we have formalized it in `Rzk`.

```
#def Contra-yoneda-lemma uses (funext)
  ( A : U)
  ( is-pre-∞-category-A : Is-pre-∞-category A)
  ( a b : A)
  : is-equiv ((z : A) → Hom A z a → Hom A z b) (Hom A a b) (Contra-evid A a b)
  :=
    ( ( ( Contra-yon A is-pre-∞-category-A a b)
      , ( Contra-yon-evid A is-pre-∞-category-A a b))
    , ( ( Contra-yon A is-pre-∞-category-A a b)
      , ( Contra-evid-yon A is-pre-∞-category-A a b)))
```

Given two terms $a, b : A$ in a pre-∞-category, the type Hom $A\ a\ b$ of arrows from a to b in A is equivalent to the type of natural transformations between the contravariant representable functors at a and b.

In the synthetic language of sHoTT used here, any fiberwise function $\phi : ((z : A) \to \text{Hom } A\ z\ a \to \text{Hom } A\ z\ b)$ is automatically a natural transformation!

Formalizing the ∞-categorical Yoneda lemma

In sHoTT, the ∞-categorical Yoneda asserts that two easily definable functors are inverse equivalences:

```
#def Contra-evid
  ( A : U)
  ( a b : A)
  : ( ( z : A) → Hom A z a → Hom A z b) → Hom A a b
  := \ φ → φ a (Id-hom A a)
```

```
#def Contra-yon
  ( A : U)
  ( is-pre-∞-category-A : Is-pre-∞-category A)
  ( a b : A)
  : Hom A a b → ((z : A) → Hom A z a → Hom A z b)
  := \ v z f → Comp-is-pre-∞-category A is-pre-∞-category-A z a b f v
```

By one of the identity laws — a theorem for, rather than an axiom about pre-∞-categories — contra-yon followed by contra-evid is homotopic to the identity:

```
#def Contra-evid-yon
  ( A : U)
  ( is-pre-∞-category-A : Is-pre-∞-category A)
  ( a b : A)
  ( v : Hom A a b)
  : Contra-evid A a b (Contra-yon A is-pre-∞-category-A a b v) = v
  :=
    Id-comp-is-pre-∞-category A is-pre-∞-category-A a b v
```

The other composite carries `φ` to an a priori distinct natural transformation. We first show that these are pointwise equal at all `x : A` and `f : Hom A x a` in two steps.

```
#section contra-yon-evid

#variable A : U
#variable is-pre-∞-category-A : Is-pre-∞-category A
#variables a b : A
```

The composite `Contra-yon-evid` of `φ` equals `φ` at all `x : A` and `f : Hom A x a`.

To show that contra-evid followed by contra-yon is homotopic to the identity, we compare a natural transformation

$$\phi : ((z : A) \to \text{Hom } A\ z\ a \to \text{Hom } A\ z\ b)$$

with its image under the composite of these functions and verify that both natural transformations define the same function given $x : A$ and $f : \text{Hom } A\ x\ a$.

Formalizing the ∞-categorical Yoneda lemma

```
#def Contra-yon-evid-twice-pointwise
  ( φ : (z : A) → Hom A z a → Hom A z b)
  ( x : A)
  ( f : Hom A x a)
  : ( ( Contra-yon A is-pre-∞-category-A a b)
        ( ( Contra-evid A a b) φ)) x f = φ x f
  :=
    concat
      ( Hom A x b)
      ( ( ( Contra-yon A is-pre-∞-category-A a b)
            ( ( Contra-evid A a b) φ)) x f)
      ( φ x (Comp-is-pre-∞-category A is-pre-∞-category-A x a a f (Id-hom A a)))
      ( φ x f)
      ( Naturality-contravariant-fiberwise-representable-transformation
          A is-pre-∞-category-A a b x a f (Id-hom A a) φ)
      ( ap
        ( Hom A x a)
        ( Hom A x b)
        ( Comp-is-pre-∞-category A is-pre-∞-category-A x a a
          f (Id-hom A a))
        ( f)
        ( φ x)
        ( Comp-id-is-pre-∞-category A is-pre-∞-category-A x a f))
```

By function extensionality, this completes the proof of the Yoneda lemma.