Geometric Reasoning in Lean

from algebraic structures to presheaves

Kenji Maillard [1] Yiming Xu [2]

[1] Inria, Nantes, France

[2] Ludwig-Maximilian Universität, München, Germany

June 8, 2025

Motivation

Formula
Proof

interpret →

Varieties of
mathematical contexts

Motivation

Formula Proof	interpret →	Varieties of mathematical contexts
Geometric logic	Formalized in Lean	Presheaf categories (families of sets)

Motivation

Formula Proof	$\xrightarrow{\text{interpret}}$	Varieties of mathematical contexts
Geometric logic	Formalized in Lean	Presheaf categories (families of sets)

- Geometric logic: a variant of first-order logic.
- Expressiveness: semigroups, monoids, division rings, fields, etc.
- Versatile: interprets into lots of categories.
- Transferable: interpretations preserved by enough functors.

Why do this in Lean?

- Actively-developed proof assistant

Why do this in Lean?

- Actively-developed proof assistant
- Dependently typed

Why do this in Lean?

- Actively-developed proof assistant
- Dependently typed
- Mathlib: large library of mathematics, including category theory

Why do this in Lean?

- Actively-developed proof assistant
- Dependently typed
- Mathlib: large library of mathematics, including category theory
- Meta-programming potential

Geometric Logic

- Terms t generated from a finitary monosorted signature Σ
- Formulas ϕ built with
  - atoms $\top$, $\bot$, predicates $P(t_i) \in \Sigma$
  - equalities $t_1 = t_2$
  - binary conjunction $\phi_1 \wedge \phi_2$
  - **possibly-infinite** disjunction $\bigvee_{j \in J} \phi_j$
  - **only** existential quantifier
- A theory is specified by axioms of the form $\phi_1 \vdash \phi_2$.

e.g. semigroup :

- $\Sigma = (\{(*, 2)\}, \emptyset)$
- Terms : a, $a * b$, $(a * b) * b$, $\cdots$
- Axiom : $\top \vdash (a * b) * c = a * (b * c)$

Presheaves

```
def Psh (C:Type) [Category C] := Functor C.op Type
```

A presheaf P on a category C contains

- a family of sets $P(c)$ for $c \in C$
- functions $P(f) : P(c_2) \to P(c_1)$ for $f : c_1 \to c_2$

Examples:

- A presheaf on the terminal category is a set.

  ```
  def Type_equiv_Psh : CategoryTheory.Psh Unit ≅ Type
  ```

- A presheaf P on the category 2

$$
\begin{array}{ccc}
0 & \qquad & P(0) := \mathbb{Z} \\
\downarrow & & \uparrow \\
1 & & P(1) := \mathbb{N}
\end{array}
$$

Set Semantics for Geometric Logic

A set model of a theory T over the signature Σ consists of a set X, with:

- for each n-ary operation $f \in \Sigma$, a function $[\![f]\!] : X^n \to X$
- for each n-ary predicate $P \in \Sigma$, a subset $[\![P]\!] \subseteq X^n$

satisfying the axioms of T

Presheaf Semantics for Geometric Logic

A presheaf model of a theory T over the signature Σ consists of a presheaf X, with:

- for each n-ary operation $f \in \Sigma$, a natural transformation $[\![f]\!] : X^n \to X$
- for each n-ary predicate $P \in \Sigma$, a monomorphism $S \rightarrowtail X^n$

satisfying the axioms of T

Contribution

Our Lean formalization ($\sim$ 2500 lines)

- Geometric syntax
- Proof rules
- Interpretation of geometric formulas in presheaves
- Soundness of the proof system for that semantics
- The collection of presheaf models on any category C forms a category
- **functoriality**

Functoriality of Category of Models

Given

- Categories C and D
- A functor $F : C \to D$
- A model $M \in \mathrm{Mod}(D) \subseteq \mathrm{Psh}(D)$ of a theory T in the signature Σ

Then $\leftarrow$ hard part !

- The pullback presheaf $F^*(M) \in \mathrm{Mod}(C)$ is a model of T
- This defines a functor $F^* : \mathrm{Mod}(D) \Rightarrow \mathrm{Mod}(C)$

Example: Connecting to Mathlib

The categories

- Semigroup: semigroup structures
- Mod(Unit): presheaves models of semigroups on the type Unit

are equivalent

Existing:

```
structure Semigrp : Type (u + 1) where
 carrier : Type u
 str : Semigroup carrier
```

Newly defined:

```
def semigroup_thy: theory where
  sig := semigroup_sig
  axioms := [ assoc ]
def semigroup_set_models :=
    InterpPsh.Mod semigroup_thy Unit
```

$$\texttt{semigroup\_set\_models} \cong \texttt{Semigrp}$$

Next Steps

Immediate Goals

- Refinement using geometric category
- Interesting examples
- Generalization to sheaves

Next Steps

Immediate Goals

- Refinement using geometric category
- Interesting examples
- Generalization to sheaves

Long Term Aims

- Blechschmidt's thesis: application to algebraic geometry
- Deligne's theorem: carrying theorems about set models to other contexts
- Barr's theorem: avoid the Axiom of Choice

Questions

- Comments?
- What would you guess to be difficult, and how would you deal with it?
- What information did you get out of our work? Is there anything surprising to you?

Geometric Syntax

```
inductive tm (m : monosig) (n : Nat) where
  | var : Fin n → tm m n
  | op : (o : m.ops) → (Fin (m.arity_ops o) → tm m n) → tm m n

class SmallUniverse where
 U : Type
 El : U → Type
inductive fml [SmallUniverse] (m : monosig) : RenCtx → Type where
  | pred : (p : m.preds) → (Fin (m.arity_preds p) → tm m n) → fml m n
  | true : fml m n
  | false : fml m n
  | conj : fml m n → fml m n → fml m n
  | disj : fml m n → fml m n → fml m n
  | infdisj : (a : SmallUniverse.U) → (SmallUniverse.El a → fml m n) →
    fml m n
  | eq : tm m n → tm m n → fml m n
  | existsQ : fml m (n + 1) → fml m n
```

Proof Rules

```
inductive proof [SmallUniverse] {T : theory} :
 {n : RenCtx} → fml T.sig n → fml T.sig n → Prop where
  | axiom : s ∈ T.axioms → proof (s.premise.subst σ) (s.concl.subst σ)
  | cut : proof ϕ τ → proof τ ψ → proof ϕ ψ
  | var : proof ϕ ϕ
  | true_intro : proof ϕ .true
  | false_elim : proof ϕ .false → proof ϕ ψ
  | conj_intro : proof ν ϕ → proof ν ψ → proof ν (.conj ϕ ψ)
  | conj_elim_l : proof (.conj ϕ  ψ) ϕ
  | conj_elim_r : proof (.conj ϕ  ψ) ψ
  | disj_intro_l : proof ϕ (.disj ϕ ψ)
  | disj_intro_r : proof ψ (.disj ϕ ψ)
  | disj_elim : proof δ (.disj ϕ ψ) →
    proof (ϕ.conj δ) ξ → proof (ψ.conj δ) ξ → proof δ ξ
  | infdisj_intro (k : SmallUniverse.El a) : proof (ϕ k) (.infdisj a ϕ)
  | infdisj_elim : proof δ (.infdisj a ϕ) →
    (forall k, proof (.conj (ϕ k) δ) ξ) → proof δ ξ
 ...
```

Interpretation of Geometric Syntax on a Presheaf

```
structure Str (S : monosig) (C : Type) [Category C] where
  carrier : Psh C
  interp_ops : forall (o : S.ops), npow carrier (S.arity_ops o) →
    carrier
  interp_preds : forall (p : S.preds), npow carrier (S.arity_preds p)
    → prop

def interp_fml {S : monosig} {n} (L : Str S C) : fml S n → (npow
    L.carrier n → prop)
  | .pred p k ↦ L.interp_subst k ≫ L.interp_preds p
  | .true ↦ ⊤
  | .conj φ ψ ↦ L.interp_fml φ ⊓ L.interp_fml ψ
  | .infdisj a φ ↦ ⨆ i: SmallUniverse.El a, interp_fml L (φ i)
  | .existsQ φ ↦ existπ (L.interp_fml φ)
  ...
```

Soundness

```
def model {S : monosig} (L : Str S C) (s : sequent S) : Prop :=
 L.interp_fml s.premise ≤ L.interp_fml s.concl

theorem soundness {T : theory} {n : RenCtx} (M:Mod T D) (φ ψ: fml T.sig n)
 (h:proof φ ψ): model M.str (sequent.mk _ φ ψ)
```

Pullback of a Model is a Model

```
structure Mod [SmallUniverse] (T : theory) (C : Type) [Category C]
   where
       str : Str T.sig C
       valid : forall s, s ∈ T.axioms → str.model s

def pb_prod_iso (X : Psh D) (n : Nat) :
 F.op ⋙ npow X n ≅ npow (F.op ⋙ X) n := ...

theorem pb_obj_interp_ops (L : Str T.sig D) (o: T.sig.ops):
 whiskerLeft F.op (L.interp_ops o) =
 (pb_prod_iso F L.carrier (T.sig.arity_ops o)).hom ≫
 (pb_obj F T L).interp_ops o := by ...

def pb_prop_preserves_interp
  (L : Str T.sig D) (s : sequent T.sig) (h : L.model s) : (pb_obj F T L).model s
def pullback_Mod : Mod T D ⥤ Mod T C
```