HoTTLean: Formalizing the groupoid model of MLTT

Joseph Hua[1], Steve Awodey, Mario Carneiro, Sina Hazratpour, Wojciech Nawrocki, Spencer Woolfson and Yiming Xu

[1] Carnegie Mellon University

HoTT/UF 2026 | Aarhus, Denmark | June 2026

This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-21-1-0009.

HoTTLean

1. A Martin-Löf Type Theory (MLTT) proof assistant in Lean (e.g. implementing HoTT).
2. Formalization of the abstract semantics of MLTT (in classical Lean).
3. Automated syntax-to-semantics translation (SynthLean) [1].
4. **Formalization of the Hofmann-Streicher model in groupoids**, with Σ, Π, and Id-types.
5. Automated translation of HoTT theorems into Mathlib-style theorems (e.g. about groupoids).

The groupoid model is fully formalised, but many improvements could be made.

The groupoid model

- No UIP (uniqueness of identity principle).
- No general univalence.
- Satisfies set-univalence: equalities of h-sets are the same as equivalences (i.e. bijections) between them.

Semantics of MLTT

We could formalise the groupoid model as a natural model [2] or a category with families [3] or a comprehension category [4] or a category with attributes [5] or a display map category [6]/clan [7] or a category with representables [8] or a contextual category/C-system [9] ... or an **elementary model** or an **algebraic model** [10] ...

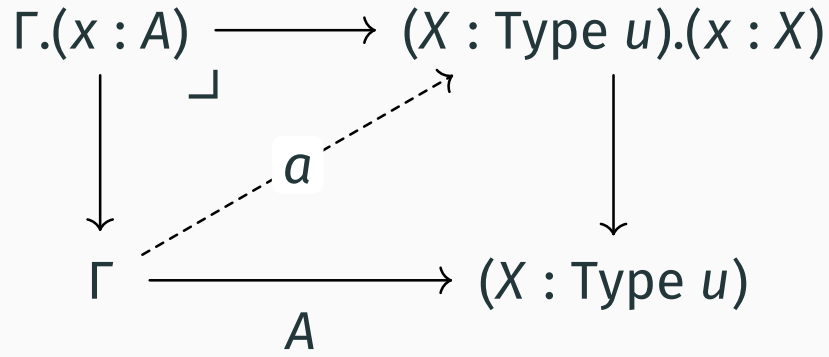
Why?

We want to model a type theory with multiple universes

$$\text{Type } 0 : \text{Type } 1 : \text{Type } 2 : \ldots$$

Let us interpret contexts as objects in a category $\mathcal{C}$, and substitutions as its morphisms. A universe should be interpreted as a substitution $(X : \text{Type } u).(x : X) \to (X : \text{Type } u)$ which is a classifier for u-small type families.

A u-small type $\Gamma \vdash A : \text{Type } u$ is a substitution $A : \Gamma \to (X : \text{Type } u)$, a term is a section.



Why?

No need for an external classifier for all type families $\mathrm{Ty} : \mathcal{C}^{\mathrm{op}} \to \mathrm{Set}$, as is used in natural models and categories with families. I.e., we work **internally** in $\mathcal{C}$.

This made the formalization much nicer.

$$\text{Elementary model} \approx \text{internal category with families}$$

$$\text{Algebraic model} \approx \text{internal natural model}$$

An elementary model is a hierarchy of CwFs, where each $\mathrm{Ty} : \mathcal{C}^{\mathrm{op}} \to \mathrm{Set}$ is a representable presheaf.

The groupoid model

$\mathcal{C} = \mathrm{Grpd}_{\cdot}\{N\}$ is the category of N-small groupoids. For $u < N$,

$$(X : \mathrm{Type}\ u) \coloneqq \mathrm{Core}\left(\mathrm{Grpd}_{\cdot}\{u\}\right)$$

Type u is (the core of) the category of u-small groupoids.

A u-small type $\Gamma \vdash A : \mathrm{Type}\ u$ is a functor $A : \Gamma \to \mathrm{Grpd}_{\cdot}\{u\}$. The classifier for u-small types is the universal u-small (split) isofibration.

$$\begin{array}{ccc} \Gamma.(x : A) & \longrightarrow & (X : \mathrm{Type}\ u).(x : X) \\ \Big\downarrow{\scriptstyle \mathrm{isofib}} & \lrcorner & \Big\downarrow{\scriptstyle \mathrm{isofib}} \\ \Gamma & \xrightarrow[A]{} & (X : \mathrm{Type}\ u) \end{array}$$

The elementary groupoid model

The groupoid model is currently formalised as an elementary model (with Σ, Π, and Id-types).

Σ-formation: for every object Γ, every morphism $A : \Gamma \to \mathrm{Type}\ u$, every morphism $B : \Gamma.A \to \mathrm{Type}\ u$, there is a morphism $\Sigma_A B : \Gamma \to \mathrm{Type}\ u$.

If $A = A' : \Gamma \to \mathrm{Type}\ u$, then we need to consider equalities of groupoids

$$\Gamma.A = \Gamma.A'$$

Currently, the file defining the interpretation of Σ-types in groupoids takes 30 seconds to compile. (This is bad.) These theorems are not modular enough for Mathlib. (This is also bad.)

The elementary groupoid model

How can we mitigate these issues?

Instead of constructing Σ-types by hand, we would like to just show that isofibrations are closed under composition and call it a day. (And more generally for an arbitrary model.)

Algebraic models (as yet not fully formalized) make this possible.

The algebraic groupoid model

Proposition. The class of isofibrations in the category of groupoids forms a π-clan:

1. (Substitution) Pullbacks of isofibrations exist and are isofibrations.
2. (Unit) Isomorphisms are isofibrations.
3. (Σ) Composition of isofibrations are isofibrations.
4. (Π) Pushforwards of a isofibrations along a isofibrations exist and are isofibrations.
5. The unique map to the terminal groupoid is an isofibration.

The algebraic groupoid model

To construct an algebraic groupoid model with Unit, Σ and Π-types, we only need to prove the following facts about groupoids [10, proposition 5.12].

1. Isofibrations form a π-clan in groupoids.
2. u-small isofibrations form a π-preclan in groupoids (the map to the terminal object is not always u-small).
3. u-small isofibrations are classified by a universe.

This is good:

- Nice to formalise, Mathlib-friendly.
- No equalities of groupoids appear in this construction.
- Equalities of types still appear, but they are now someone else's problem.

The algebraic groupoid model

The “someone else’s problem” is resolved using polynomial functors (a.k.a containers) in π-clans in [10]. In this way, algebraic models (like elementary models) are **strict**, meaning there is no coherence problem.

What about algebraic Id-types?

Algebraic identity types

Let $I = \{\star \cong \star\}$ be the walking isomorphism. For any groupoid A, taking endpoints

$$(\mathrm{src}, \mathrm{trg}) : A^I \to A \times A$$

is an isofibration. More generally, for any isofibration $A \to X$,

$$(\mathrm{src}, \mathrm{trg}) : A^{I_X} \to A \times_X A$$

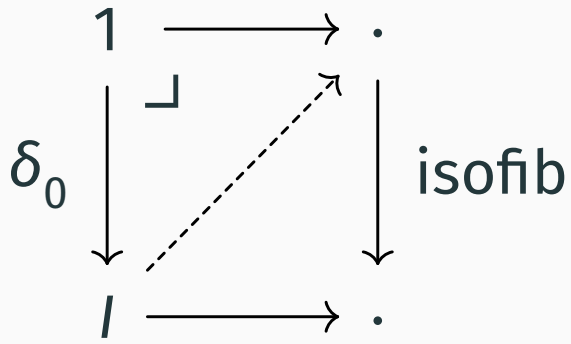
is an isofibration in the slice over X.

"Isofibrations are closed under taking pathobjects."

Algebraic identity types

To construct an algebraic groupoid model with Id-types, we can use path types [2]. We just need to show that:

1. u-small isofibrations are closed under taking pathobjects.
2. Isofibrations right-lift against the endpoint inclusion $\delta_0 : 1 \to I$.



The current formalization uses path types in an elementary model.

Our repository: github.com/sinhp/HoTTLean

[1] W. Nawrocki et al., “A Certifying Proof Assistant for Synthetic Mathematics in Lean,” in Proceedings of the 15th ACM SIGPLAN International Conference on Certified Programs and Proofs, in CPP '26. Rennes, France: Association for Computing Machinery, 2026, pp. 88–103. doi: 10.1145/3779031.3779087.

[2] S. Awodey and J. Hua, “Path Types in Algebraic Type Theory.” [Online]. Available: https://arxiv.org/abs/2601.06567

[3] P. Clairambault and P. Dybjer, “The biequivalence of locally cartesian closed categories and Martin-Löf type theories,” in International Conference on Typed Lambda Calculi and Applications, 2011, pp. 91–106.

[4] B. Jacobs, “Comprehension categories and the semantics of type dependency,” Theoretical Computer Science, vol. 107, no. 2, pp. 169–207, 1993, doi: https://doi.org/10.1016/0304-3975(93)90169-T.

[5] K. Kapulkin and P. L. Lumsdaine, “The homotopy theory of type theories,” Advances in Mathematics, vol. 337, pp. 1–38, 2018, doi: https://doi.org/10.1016/j.aim.2018.08.003.

[6] P. Taylor, “Recursive domains, indexed category theory and polymorphism,” Doctoral dissertation, 1987.

[7] A. Joyal, “Notes on Clans and Tribes.” [Online]. Available: https://arxiv.org/abs/1710.10238

[8] T. Uemura, “A general framework for the semantics of type theory,” Mathematical Structures in Computer Science, vol. 33, no. 3, pp. 134–179, 2023, doi: 10.1017/S0960129523000208.

[9] J. Cartmell, “Generalised algebraic theories and contextual categories,” Annals of Pure and Applied Logic, vol. 32, pp. 209–243, 1986, doi: https://doi.org/10.1016/0168-0072(86)90053-9.

[10] J. Hua and Y. Xu, “Polynomial functors in pi-clans for the semantics of type theory.” [Online]. Available: https://arxiv.org/abs/2602.05689