Two-dimensional semantics of homotopy type theory

Michael Shulman

May 27, 2021
Topos Institute Colloquium

Outline

Dependent type theory

Martin-Löf dependent type theory is both:

1. An internal language for statements that are true in any topos.
2. A programming language for certifiably correct programs.

Example

```
(++) : List(A) -> List(A) -> List(A)
assoc : forall (l1 l2 l3 : List(A)),
          l1 ++ (l2 ++ l3) = (l1 ++ l2) ++ l3
```

This is both

1. A theorem about list objects in any topos.
2. A verified property of a program acting on lists.

Making the language richer

In both cases we have parallel problems that:

1. Many interesting categories are not toposes.
2. Many programming languages are not dependently typed.

Indeed, simply typed programming languages correspond closely to less structured categories, such as cartesian closed ones.

So the rich structure of dependent types is not directly available.

Native type theory

A solution is the Yoneda embedding $よ : \mathcal{C} \hookrightarrow \mathcal{PC}$.

Theorem (D.S. Scott, 1980)

For any category $\mathcal{C}$, the embedding $よ$ into its presheaf topos $\mathcal{PC}$ is fully faithful and preserves any limits and exponentials that exist. Using sheaves $\mathcal{S}h(\mathcal{C})$ instead, it also preserves "good" colimits.

Thus, we can use dependent type theory to reason about arbitrary categories and programming languages, by

1. Embedding them in a topos,
2. Reasoning in the topos,
3. Using full-faithfulness and preservation to reflect conclusions.

For programming, this has recently been advocated under the name native type theory by Williams and Stay (arXiv:2102.04672).

The problem of universes

This works well for all the structure of dependent type theory except universes.

- Universes in a general topos require inaccessible cardinals.
- General universes in $\mathcal{PC}$ are not related to $\mathcal{C}$.
- Universes in a topos impose an unnatural "equality of objects".

And yet, in both category theory and programming, we certainly want to be polymorphic over objects/types:

```
(++): forall (A:Type), List(A) -> List(A) -> List(A)
```

The universe of representables

Idea

There should be a universe $\mathcal{U}$ in $\mathcal{PC}$ whose elements "are" the objects of $\mathcal{C}$.

$$\mathcal{U}(I) \cong \mathcal{PC}(\text{よ}I, \mathcal{U}) = \text{“}I\text{-indexed families of objects of } \mathcal{C}\text{”}$$

$$\cong ob(\mathcal{C}/I)$$

(From $(A_i)_{i:I}$ in Set, get $\sum_i A_i \to I$ in Set$/I$.)

Problem

This is not functorial! The reindexing action should be by pullback, but that is not strictly functorial.

Higher presheaves

Solution

The collection of arrows $A \to I$ in $\mathcal{C}$ should not be regarded as a set, but as a category: the slice category $\mathcal{C}/I$.

Now $I \mapsto \mathcal{C}/I$ is pseudo-functorial.

$$\begin{tikzcd} g^*f^*A \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner"] & f^*A \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner"] & A \arrow[d] \\ K \arrow[r, "g"'] & J \arrow[r, "f"'] & I \end{tikzcd} \qquad \cong \qquad \begin{tikzcd} (fg)^*A \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner"] & A \arrow[d] \\ K \arrow[r, "fg"'] & I \end{tikzcd}$$

For technical reasons (later) we use instead the groupoid $\mathrm{core}(\mathcal{C}/I)$, containing only the isomorphisms in $\mathcal{C}/I$.

Univalent extensions

Given a category (or language) $\mathcal{C}$, we give:

- A dependent type theory with:
  - Σ-types, Π-types, sum types, and quotient types
  - One universe $\mathcal{U}$ that is univalent, meaning isomorphic types are indistinguishable.
- An interpretation of this type theory in a $(2, 1)$-topos of (pre)sheaves of groupoids on $\mathcal{C}$.

This is a simplified version of homotopy type theory and its semantics in $(\infty, 1)$-toposes (sheaves of ∞-groupoids). Ordinary groupoids are much easier than ∞-groupoids, and often sufficient.

Outline

Stacks

Example (For the rest of this talk)

Let $\mathcal{C}$ be a cartesian closed category with finite coproducts that are disjoint and stable.

- This is a reasonable first approximation to the semantics of a simple functional programming language.
- The same methods also work for categories with more or less structure of a similar sort.

Definition

A stack is a *pseudo* functor $F : \mathcal{C}^{\mathrm{op}} \rightsquigarrow \mathcal{G}pd$ such that the canonical maps are equivalences

$$F(0) \simeq 1 \qquad\qquad F(A + B) \simeq F(A) \times F(B).$$

Examples of stacks

Example

For any $A \in \mathcal{C}$, the representable functor

$$よA(X) = \mathcal{C}(X, A)$$

(regarded as a discrete groupoid) is a stack.

Example

If $\mathcal{C}$ has all pullbacks, the universe of representables

$$\mathcal{U}(X) = \mathrm{core}(\mathcal{C}/X)$$

is a stack, with functorial action by pullback.

Otherwise, we can restrict to the full subcategory $\mathcal{C}/\!\!/X$ of $\mathcal{C}/X$ whose objects are the product projections $A \times X \to X$.

"Type theory" in stacks

Let $\mathcal{S}t(\mathcal{C})$ denote the 2-category of stacks and pseudonatural transformations. Informally, we expect:

- A type A is interpreted by a stack.
- A family of types B_x indexed by $x : A$ is interpreted by a morphism of stacks $B \to A$.
- Substitution (reindexing) $\{B_{f(x)}\}_{x:C}$ is interpreted by the pullback of $B \to A$ along $f : C \to A$.
- The dependent sum $\sum_{x:A} B_x$ and dependent product $\prod_{x:A} B_x$ are interpreted by left and right (pseudo) adjoints to pullback.

$$\begin{tikzcd}
\mathcal{S}t(\mathcal{C})/S \arrow[r, bend left, "\Sigma_f"] \arrow[r, bend right, "\Pi_f"'] & \mathcal{S}t(\mathcal{C})/T \arrow[l, "\Delta_f"' description]
\end{tikzcd}$$

NB

This doesn't quite work yet, but for now let's assume it does.

Representables in stacks

An object $A \in \mathcal{C}$ corresponds to the representable $よA \in \mathcal{S}t(\mathcal{C})$.

Fact

The Yoneda embedding $よ : \mathcal{C} \hookrightarrow \mathcal{S}t(\mathcal{C})$ preserves limits, exponentials, and sums (the latter because we used stacks).

Thus, these type operations in $\mathcal{S}t(\mathcal{C})$ applied to representables reduce to those of $\mathcal{C}$.

$$よA \times よB \;=\; \sum_{x:よA} よB \;\cong\; よ(A \times B)$$

$$よA \to よB \;=\; \prod_{x:よA} よB \;\cong\; よ(A \to B)$$

$$よA + よB \;\cong\; よ(A + B)$$

Inside the "type theory" of $\mathcal{S}t(\mathcal{C})$ we have a copy of $\mathcal{C}$.

Identity and isomorphism types

The identity type $a = b$, for $a, b : A$, is interpreted by the stack

$$A^{\cong}(X) = \left\{ (x_1, x_2, \xi) \;\middle|\; x_1 \in A(X), x_2 \in A(x), \xi : x_1 \cong x_2 \right\}$$

of isomorphisms in A, with projection to $A \times A$.

- A type A is contractible if $\sum_{x:A} \prod_{y:A} x = y$: there is an object x that is *naturally* isomorphic to every other object.
- A is a proposition if each $x = y$ is contractible, and a set if each $x = y$ is a proposition. The sets in the type theory of $\mathcal{S}t(\mathcal{C})$ are the *sheaves* on $\mathcal{C}$.
- Since $\mathcal{C}$ is a 1-category and よ preserves identity types, all representables $\text{よ}A$ are sets.
- Since $\mathcal{S}t(\mathcal{C})$ is a 2-category (not an ∞-category), all identity types $x = y$ are sets, i.e. all types are 1-types.

Predicates on representables

Definition

A predicate on $A \in \mathcal{S}t(\mathcal{C})$ is a fully faithful inclusion $P \hookrightarrow A$, i.e. a family $\{P_x\}_{x:A}$ of propositions.

A predicate on a representable $よA$ is a closed sieve on $A \in \mathcal{C}$: a set P of morphisms $X \to A$ such that

1. If $f \in P$ then $fg \in P$.
2. The unique map $0 \to A$ is in P.
3. If $f : X \to A$ and $g : Y \to A$ are in P, so is their copairing $[f, g] : X + Y \to A$.

Example

For $h : A \to B$ in $\mathcal{C}$, the predicate $\{\exists_{a:よA} h(a) = b\}_{b:よB}$ is the sieve of all $f : X \to B$ that factor through h: the image of $よh$ in $\mathcal{S}t(\mathcal{C})$.

Aside: why groupoids?

Question

Why do we use stacks of groupoids instead of categories?

1. For categories, we would want the stack $A^{\to}$ of morphisms instead of $A^{\cong}$ of isomorphisms. But:
   - $A^{\to}$ doesn't obey the ordinary rules of identity types.
   - We can write down rules for it (Licata-Harper), but they're not as pretty or convenient, and hard to generalize.
2. Pullback $f^* : \mathcal{C}at/B \to \mathcal{C}at/A$ doesn't generally have even a right pseudo adjoint.
   - It does if f is a fibration or opfibration...
   - So we might try to introduce "types with variance"...
   - But plenty of important dependent types, like sieves, don't have either variance!

With groupoids, we can use ordinary identity types and Π-types.

The universe of representables

The universe type $\mathcal{U}$ is the universe of representables, $\mathcal{U}(X) = \mathrm{core}(\mathcal{C}/\!\!/X)$. Essentially by definition this gives:

The univalence axiom

For $A, B : \mathcal{U}$, we have $(A = B) \cong (A \cong B)$ canonically.

In particular, the type $\mathcal{U}$ is not generally a "set", since two objects of $\mathcal{C}$ can be isomorphic in more than one way.

Isomorphism invariance

Since everything is invariant under *equality*, univalence implies everything we can say about $A : \mathcal{U}$ is invariant under isomorphism: we treat $\mathcal{C}$ truly category-theoretically.

Parametric polymorphism

By the Yoneda lemma, $\mathrm{Hom}_{\mathcal{S}t(\mathcal{C})}(よA, \mathcal{U}) \simeq \mathcal{U}(A)$. Thus:

- A representable type family $B : よA \to \mathcal{U}$ is an object of $\mathcal{C}$.
- An equality $よB = よC$ of such families over $よA$ is an isomorphism $A \times B \cong A \times C$ in $\mathcal{C}/A$.

Example

A morphism $\mathcal{U} \to \mathcal{U}$ (or $\mathcal{U} \to \mathcal{U} \to \mathcal{U}$, etc.) is an operation on objects of $\mathcal{C}$ that is functorial on isomorphisms *in all slices*:

$$\begin{aligned} (+) &: \mathcal{U} \to \mathcal{U} \to \mathcal{U} \\ \texttt{List} &: \mathcal{U} \to \mathcal{U} \end{aligned}$$

Parametric polymorphism

Example

A function parametrized by $A : \mathcal{U}$ is a family of morphisms in $\mathcal{C}$ that is natural with respect to isomorphisms in all slices. E.g.

$$\texttt{inl} : \prod_{A,B:\mathcal{U}} \big(A \to A + B\big)$$

$$\texttt{inr} : \prod_{A,B:\mathcal{U}} \big(B \to A + B\big)$$

$$\texttt{(++)} : \prod_{A:\mathcal{U}} \Big(\texttt{List}(A) \to \texttt{List}(A) \to \texttt{List}(A)\Big)$$

Outline

Type theory is 1-categorical

We can't actually interpret type theory directly in $\mathcal{S}t(\mathcal{C})$.

Problem

The rules of type theory give strict 1-categorical universal properties, not the 2-categorical "up to equivalence" ones in $\mathcal{S}t(\mathcal{C})$.

Example

The rules for pairs:

$$\pi_1(a, b) \equiv a \qquad \pi_2(a, b) \equiv b \qquad s \equiv (\pi_1(s), \pi_2(s))$$

say that maps $X \to A \times B$ are bijective with pairs of maps $X \to A$ and $X \to B$, not merely equivalent as hom-groupoids.

We could replace strict equality $\equiv$ by something weaker, but we would lose computational behavior and become harder to use.

Strictification

Solution

Use strict objects to represent weak ones.

Consider the case of presheaves first.

- Let $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$ be the 2-category of strict functors $\mathcal{C}^{\mathrm{op}} \to \mathcal{G}pd$ and strict natural transformations.
- Let $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}$ be the 2-category of pseudofunctors $\mathcal{C}^{\mathrm{op}} \rightsquigarrow \mathcal{G}pd$ and pseudonatural transformations.

Theorem (Blackwell-Kelly-Power, Lack, . . .)

The "inclusion" $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd] \to [\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}$ has a left adjoint $\mathfrak{Q}$ and a right adjoint $\mathfrak{R}$.

We have a strict unit $A \to \mathfrak{R}A$ (for strict A) and a pseudo counit $\mathfrak{R}B \rightsquigarrow B$ (for pseudo B): inverse equivalences in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}$.

Fibrant presheaves

The functor $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd] \to [\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}$ is not a (bi)equivalence, but becomes so when we restrict its domain to a full sub-2-category.

Definition

A strict functor $A : \mathcal{C}^{\mathrm{op}} \to \mathcal{G}pd$ is fibrant (a.k.a. coflexible) if the unit $A \to \mathfrak{R}A$ has a retraction in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$.

Let $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathfrak{R}} \subseteq [\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$ consist of the fibrant objects.

Theorem (Blackwell-Kelly-Power, Lack, ...)

The composite $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathfrak{R}} \hookrightarrow [\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd] \to [\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}$ *is a (bi)equivalence, with inverse* $\mathfrak{R}$*.*

Fibrations of presheaves

- $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathfrak{R}}$ is a CCC, but not finitely complete or LCCC.
- It interprets strict product types and exponentials, but not substitution of type families or Π-types.

Solution

Interpret type families $(B_x)_{x \in A}$ as "special" morphisms $B \to A$.

$$\begin{tikzcd}
B \arrow[dr] \arrow[drr, bend left] \arrow[ddr, bend right, "f"'] & & \\
& \mathfrak{R}_A B \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \mathfrak{R} B \arrow[d, "\mathfrak{R} f"] \\
& A \arrow[r] & \mathfrak{R} A
\end{tikzcd}$$

Definition

$f : B \to A$ is a fibration if each $f_c : B_c \to A_c$ is a fibration of groupoids and $B \to \mathfrak{R}_A f$ has a retraction over A.

Facts about fibrations

1. A is fibrant iff $A \to 1$ is a fibration.
2. If $B \to A$ is a fibration and A is fibrant, so is B.
3. Fibrations are closed under pullback in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$.
4. Fibrations are closed under pushforward in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$.
5. If $B \to A$ and $C \to A$ are fibrations, so is $B + C \to A$.
6. $A^{\cong} \to A \times A$ is a fibration.
7. Every representable is fibrant.
8. The universal map over $\mathfrak{R}\mathcal{U}$ is a fibration, and $\mathfrak{R}\mathcal{U}$ is fibrant.

This is enough to actually interpret type theory in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathfrak{R}}$.

Fibrations for stacks

To do something similar in the 2-category $\mathcal{S}t(\mathcal{C})$ of stacks, we hope for a similar notion of local fibration such that:

1. A is a fibrant stack iff $A \to 1$ is a local fibration.
2. If $B \to A$ is a local fibration and A is a fibrant stack, so is B.
3. Local fibrations are closed under pullback in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$.
4. Local fibrations are closed under pushforward in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$.
5. If $B \to A$ and $C \to A$ are local fibrations, so is $B + C \to A$.
6. $A^{\cong} \to A \times A$ is a local fibration.
7. Every representable is a fibrant stack.
8. There is a universal local fibration over a fibrant stack.

Fibrations for stacks

To do something similar in the 2-category $\mathcal{S}t(\mathcal{C})$ of stacks, we hope for a similar notion of local fibration such that:

1. A is a fibrant stack iff $A \to 1$ is a local fibration.
2. If $B \to A$ is a local fibration and A is a fibrant stack, so is B.
3. Local fibrations are closed under pullback in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$.
4. Local fibrations are closed under pushforward in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$.
5. ~~If $B \to A$ and $C \to A$ are local fibrations, so is $B + C \to A$.~~
6. $A^{\cong} \to A \times A$ is a local fibration.
7. Every representable is a fibrant stack.
8. There is a universal local fibration over a fibrant stack.

If A and B are stacks, $A + B$ may not be: we need to stackify it.
This has a universal property for stacks C:

$$[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}(\mathsf{st}(A+B), C) \simeq [\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}(A+B, C)$$

$$\begin{tikzcd}
A \arrow[dr] \arrow[ddrrr, bend left] & & & \\
& A+B \arrow[ddr] \arrow[drr] & & \\
& & \cong & C \\
& & \mathrm{st}(A+B) \arrow[ur] & \\
B \arrow[uuur] \arrow[uurrr, bend right] & & &
\end{tikzcd}$$

But the type-theoretic rules for $A + B$ are also strict!

Weak factorization systems

Definition

A weak factorization system (a.k.a. wfs) in a category is a pair of classes of maps $(\mathcal{L}, \mathcal{R})$ such that

1. Every morphism factors as an $\mathcal{L}$-map followed by an $\mathcal{R}$-map.
2. Every square from an $\mathcal{L}$-map to an $\mathcal{R}$-map has a (strict) filler:

$$\begin{tikzcd} A \arrow[r] \arrow[d, "\in\mathcal{L}"'] & C \arrow[d, "\in\mathcal{R}"] \\ B \arrow[r] \arrow[ur, dashed, "\exists"] & D \end{tikzcd}$$

3. This property characterizes $\mathcal{L}$ in terms of $\mathcal{R}$, and vice versa.

A wfs for stacks?

If we had a wfs $(\mathcal{L}, \mathcal{R})$ where $\mathcal{R}$ is the "local fibrations" for stacks, it would solve the problem of $A + B$:

$$
\begin{tikzcd}
 & A+B \arrow[r] \arrow[d, "\in\mathcal{L}"'] & C & \text{stack} \arrow[l] \\
\text{stack} \arrow[r] & st(A+B) \arrow[r] \arrow[ur, dashed, "\exists"] \arrow[d, "\in\mathcal{R}"'] & 1 \arrow[from=u, "\in\mathcal{R}"] & \\
 & 1 & &
\end{tikzcd}
$$

Outline

Model categories

Definition

A Quillen model category is a bicomplete category $\mathcal{M}$ with:

- Three classes of maps
  - $\mathcal{C}of$ = cofibrations, $\rightarrowtail$
  - $\mathcal{F}ib$ = fibrations, $\twoheadrightarrow$
  - $\mathcal{W}$ = weak equivalences, $\xrightarrow{\sim}$
- $\mathcal{W}$ is closed under retracts and 2-out-of-3 (if two of f, g, and gf are in $\mathcal{W}$, so is the third).
- $(\mathcal{C}of \cap \mathcal{W}, \mathcal{F}ib)$ and $(\mathcal{C}of, \mathcal{F}ib \cap \mathcal{W})$ are wfs.

- $\mathcal{F}ib \cap \mathcal{W}$ = acyclic fibrations or trivial fibrations.
- $\mathcal{C}of \cap \mathcal{W}$ = acyclic cofibrations or trivial cofibrations.
- A is fibrant if $A \to 1$ is a fibration.
- A is cofibrant if $0 \to A$ is a cofibration.

The model category of groupoids

In general, constructing model categories is hard and abstract.
But our examples are very explicit.

Example (The "canonical" or "folk" model structure)

In the category $\mathcal{G}pd$ of groupoids:

- A cofibration is a functor injective on objects.
- A fibration is a Grothendieck fibration: $p : B \to A$ such that for $\varphi : p(b) \cong a$, there exists $\overline{\varphi} : b \cong b'$ with $p(\overline{\varphi}) = \varphi$.
- A weak equivalence is an equivalence of groupoids.
- All objects are fibrant and cofibrant.
- The acyclic fibrations are the retract equivalences, $f : A \to B$ with $s : B \to A$ such that $fs = 1_B$ and $sf \cong 1_A$.
- The acyclic cofibrations are the coretract equivalences, $f : A \to B$ with $r : B \to A$ such that $fr \cong 1_B$ and $rf = 1_A$.

The injective model structure

Example (The injective model structure)

In the category $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$:

- A cofibration $f : A \to B$ is a pointwise cofibration: each $f_c : A_c \to B_c$ is a cofibration in $\mathcal{G}pd$.
- Similarly, a weak equivalence is a pointwise equivalence. (*Not* internal equivalences in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$, but in $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}$.)
- A fibration is as previously: a pointwise fibration such that $A \to \mathfrak{R}_B f$ has a retraction over B.
- All objects are cofibrant; the fibrant objects are as previously.

- There is a dual *projective* model structure using $\mathfrak{Q}$ instead.
- The 2-category of fibrant objects is $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathfrak{R}}$, which is (bi)equivalent to $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}$.

The injective stack model structure

Example (The injective model structure for stacks)

A different model structure on the same category $[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]$:

- The cofibrations are the same: the pointwise cofibrations.
- $f : A \to B$ is a weak equivalences if for any stack C,

$$[\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}(B, C) \to [\mathcal{C}^{\mathrm{op}}, \mathcal{G}pd]_{\mathsf{ps}}(A, C)$$

  is an equivalence of groupoids.
- The fibrations are the pullbacks of injective fibrations between fibrant stacks.

The 2-category of fibrant objects is (bi)equivalent to $\mathcal{S}t(\mathcal{C})$.

Outline

Type theory in a model category

Theorem (Awodey-Warren)

In any model category, we can interpret type theory:

- *A type A is a fibrant object.*
- *A type family $(B_x)_{x:A}$ is a fibration $B \twoheadrightarrow A$.*
- *The identity type is a path object: a factorization of the diagonal $A \overset{\sim}{\rightarrowtail} PA \twoheadrightarrow A \times A$.* *(Modulo coherence; later.)*

In particular, the lifting property of $A \overset{\sim}{\rightarrowtail} PA$ is precisely the Martin-Löf elimination rule for identity types.

The Frobenius property and Π-types

Σ-types always exist (compose fibrations). For Π-types we need:

Lemma

If pushforward f_* along $f : B \to A$ exists, it preserves fibrations iff pullback f^* preserves acyclic cofibrations.

$$\begin{tikzcd}
f^*X \arrow[r] \arrow[d, tail, "\sim", "\text{acyc. cof.}"'] & C \arrow[d, two heads, "\text{fib.}"] & \\
f^*Y \arrow[r] \arrow[ur, dashed] & B \arrow[dr, "f"'] & \\
 & & A
\end{tikzcd}
\qquad
\begin{tikzcd}
X \arrow[r] \arrow[d, tail, "\sim", "\text{acyc. cof.}"'] & f_*C \arrow[d] \\
Y \arrow[r] \arrow[ur, dashed] & A
\end{tikzcd}$$

We can verify this in examples, generally when f is a fibration.

Coherence

We still need to do something to make all the operations strictly preserved by substitution (pullback).

Coherence theorem (Lumsdaine–Warren)

Represent a type family $(B_x)_{x:A}$ by a fibration $B \twoheadrightarrow V_B$ together with a map $b : A \to V_B$, standing in for the pullback $b^* B$.

Then $(B_{f(y)})_{y:C}$ consists of $B \twoheadrightarrow V_B$ (same V_B!) with $bf : C \to V_B$, and composition is strictly associative.

Conclusion

Theorem

There is a model category presenting the 2-category $\mathcal{S}t(\mathcal{C})$ of stacks of groupoids on any site $\mathcal{C}$, in which we can interpret type theory with Σ-types, Π-types, and identity types. Moreover:

- *Any limits, exponentials, and "good" colimits in $\mathcal{C}$ are preserved in $\mathcal{S}t(\mathcal{C})$.*
- *Any pullback-stable class of maps in $\mathcal{C}$ satisfying descent yields a universe of representables $\mathcal{U}$.*

Some choices of $\mathcal{C}$ that have "good" colimits include:

- Any small category with trivial topology.
- An extensive category with its extensive topology.
- An exact category with its regular topology.
- A (pre)topos with its coherent topology.