To depend or not to depend? That is the question

INF513 Project Presentation: Formalizing Categories with Families in Agda

Yee-Jian Tan

March 19, 2024

Advisor: Ambrus Kaposi

Introduction: Dependent Types

- Type theory is a formal system that can replace eg. Set Theory
- Based on giving **types** to λ-calculus **terms**, semantics given by **typing rules** and **reduction rules**.
- Logical power comes from Curry-Howard correspondence:

Type Theory	Logic
Types	Propositions
Terms	Proofs

- Dependent types: powers Agda, Coq, Lean....

  Eg. Calculus of Constructions (CoC), Martin-Löf Type Theory (MLTT)
- Types can depend on terms: vectors of fixed length `Vec n A`
- $(x : \mathbb{N}) \to x \times 0 = 0$ is a dependent type.

Introduction: Semantics

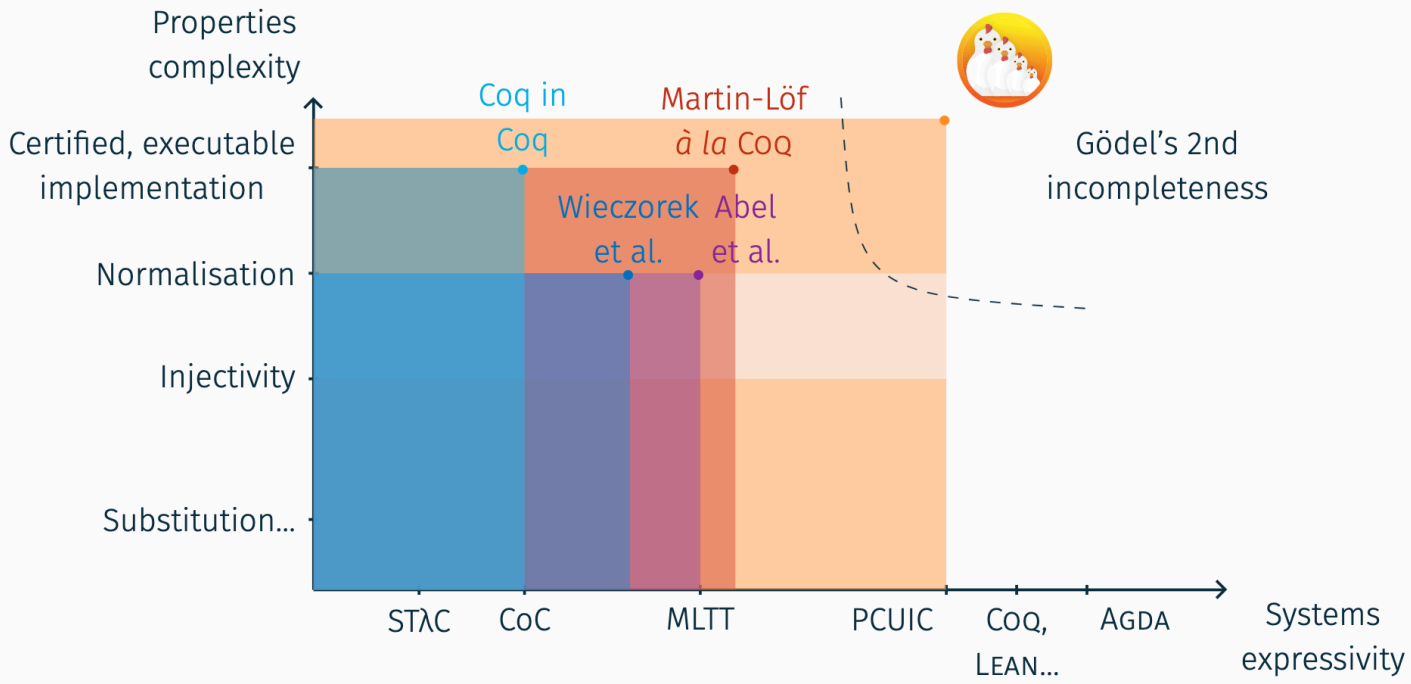
Type theory should eat itself!

We want to (formally) formalize type theory in itself.

- Syntactic representation (free family of preterms, then type)
- Semantic model: using category theory, for example **Category with Families** (Dyjber 95)
- Goal: formalize this representation of TT in itself (Agda), which allows proving metatheoretic properties (such as Canonicity/Normalization) formally.

“Setting up the stage” for exciting proofs.

Existing works: Syntactic



Attempts here are all syntactic.

Existing works: Categorical

- Guillaume Brunerie formalized MLTT using **contextual categories** in Agda. Proved initiality.
- Ambrus Kaposi formalized MLTT (**CwF** + Π-types) in Agda. Proved normalization.

$$\begin{gathered}\text{Semantic (lccc)} \to \text{Contextual categories} \to \text{Natural Model} \\ \to \mathbf{CwF} \\ \to \text{E-CwF (unquotiented)} \to \text{Syntactic (preterms + relations)}\end{gathered}$$

Categories with Families is close to the syntax of dependent types.

Achievements in this project

A learning project to

- Understand Category of Families (and the necessary category theory)
- Experience first-hand the difficulties of TT-in-TT (transport hell)

Results:

- Definition of a Category with Families, with $\Pi, \mathbb{0}, \mathbb{1}$
- Fought the battle on transport hell
- A less indexed (dependent) version, with $\Pi, \Sigma, \mathbb{0}, \mathbb{1}, \mathbb{2}$
- A wider perspective on dependent types as a theory

Goal for Today

1. Categories with Families
2. **Transport Hell**: What is it?
3. Less-indexed CwFs: Natural Model
4. Conclusion

Categories with Families

Story: Trying to build CwFs

Imagine me on the first try:

Categories with Families, Part 1

- Category of contexts $\mathcal{C}$ and "substitutions"
- with terminal object $\diamond \in \mathcal{C}$ "empty context"
- Functor $F = (\mathbf{Ty}, \mathbf{Tm}) : \mathcal{C}^{\mathrm{op}} \to \mathbf{Fam}$ of types and terms

Okay, this is just typing out definitions in Agda... Or is it?

Story: Trying to build CwFs

Imagine me on the first try:

Categories with Families, Part 1

- Category of contexts $\mathcal{C}$ and "substitutions"
- with terminal object $\diamond \in \mathcal{C}$ "empty context"
- Functor $F = (\mathbf{Ty}, \mathbf{Tm}) : \mathcal{C}^{\mathrm{op}} \to \mathbf{Fam}$ of types and terms

Okay, this is just typing out definitions in Agda... Or is it?

Agda (expectation)

$$
\begin{aligned}
&\mathrm{Con} && : \mathrm{Set}\\
&\mathrm{Sub} && : \mathrm{Con} \to \mathrm{Con} \to \mathrm{Set}\\
&\mathrm{Ty} && : \mathrm{Con} \to \mathrm{Set}\\
&\mathrm{Tm} && : (\Gamma : \mathrm{Con}) \to \mathrm{Ty}\ \Gamma \to \mathrm{Set}\\
&\mathrm{id} && : \mathrm{Sub}\ \Gamma\ \Gamma\\
&\_ \circ \_ && : \mathrm{Sub}\ \Delta\ \Gamma \to \mathrm{Sub}\ \Gamma\ \Theta \to \mathrm{Sub}\ \Delta\ \Theta\\
&\mathrm{idl} && : \mathrm{id} \circ \sigma = \sigma\\
&\mathrm{idr} && : \sigma \circ \mathrm{id} = \sigma\\
&\mathrm{assoc} && : (\delta \circ \gamma) \circ \theta = \delta \circ (\gamma \circ \theta)
\end{aligned}
$$

$$
\begin{aligned}
&\diamond && : \mathrm{Con}\\
&\varepsilon && : \mathrm{Sub}\ \Gamma\ \diamond\\
&\diamond\,\eta && : (\sigma : \mathrm{Sub}\ \Gamma\ \diamond) = \varepsilon\\
&\_[\_] && : \mathrm{Ty}\ \Gamma \to \mathrm{Sub}\ \Delta\ \Gamma \to \mathrm{Ty}\ \Delta\\
&\_[\_] && : \mathrm{Tm}\ \Gamma\ A \to (\sigma : \mathrm{Sub}\ \Delta\ \Gamma) \to \mathrm{Tm}\ \Delta\ A[\sigma]\\
&[\mathrm{id}] && : A[\mathrm{id}] = A\\
&[\circ] && : A[\sigma \circ \delta] = A[\sigma][\delta]\\
&[\mathrm{id}] && : a[\mathrm{id}] = a\\
&[\circ] && : a[\sigma \circ \delta] = a[\sigma][\delta]
\end{aligned}
$$

Category with Families: definition

Categories with Families, Part 2: Context extension

Given $\Gamma \in \mathcal{C}$, $A \in \mathrm{Ty}(\Gamma)$, there exists

$$\begin{aligned} \Gamma.A &\in \mathcal{C}, \\ p :\ \Gamma.A &\to \Gamma, \\ q &\in \mathrm{Tm}(\Gamma.A, A[p]) \end{aligned}$$

such that for any substitution $\sigma : \Delta \to \Gamma$, $a \in \mathrm{Tm}(\Delta, A[\sigma])$, we have a unique "extension"

$$\langle \sigma, a \rangle : \Delta \to \Gamma.A.$$

In human language: we can extend contexts with terms, from Γ to $\Gamma.A$, with a term $a : A$.

Story: Trying to build CwFs

Agda (expectation)

$$
\begin{array}{ll}
(\_ \triangleright \_) : (\Gamma : \mathrm{Con}) \to \mathrm{Ty}\ \Gamma \to \mathrm{Con} & \triangleright\beta_1 : p \circ (\sigma, t) = \sigma \\
(\_, \_) \quad : \sigma : \mathrm{Sub}\ \Gamma\ \Delta \to \mathrm{Tm}\ \Gamma\ A \to \mathrm{Sub}\ \Gamma\ (\Delta \triangleright A) & \triangleright\beta_2 : q \circ (\sigma, t) = t \\
p \quad : \mathrm{Sub}\ (\Gamma \triangleright A)\ \Gamma & ,\circ \quad : (\sigma, t) \circ \gamma = (\sigma \circ \gamma, t[\gamma]) \\
q \quad : \mathrm{Tm}\ (\Gamma \triangleright A)\ A[p] & ,\eta \quad : (p, q) = \mathrm{id}
\end{array}
$$

Let's go!

Category with Families: Agda, let's go!

```
23 record CwF {l k} : Set (lsuc (l ⊔ k)) where
24   field
25     -- Con is a category with a terminal object.
26     Con   : Set l
27     Sub   : Con → Con → Set k
28
29     _∘_   : ∀{Γ1 Γ2 Γ3} → Sub Γ2 Γ3 → Sub Γ1 Γ2 → Sub Γ1 Γ3
30     id    : {Γ : Con} → Sub Γ Γ
31     idl   : ∀{A B} → (f : Sub A B) → id ∘ f ≡ f
32     idr   : ∀{A B} → (f : Sub A B) → f ∘ id ≡ f
33     assoc : ∀{A B C D} → (f : Sub A B) → (g : Sub B C) → (h : Sub C D) →
34             ((h ∘ g) ∘ f) ≡ (h ∘ (g ∘ f))
```

Easy!

Category with Families: Agda, let's go!

Human language: substitution changes **types**...

```
44      -- substitution in types & terms
45      _[_] : ∀{Γ Δ} → Ty Γ → Sub Δ Γ → Ty Δ
46      [id] : ∀{Γ} → {A : Ty Γ} → A [ id ] ≡ A
47      [∘]  : ∀{Γ1 Γ2 Γ3} → (A : Ty Γ3) → (f : Sub Γ1 Γ2) → (g : Sub Γ2 Γ3) →
48             A [ (g ∘ f) ] ≡ A [ g ] [ f ]
```

Human language: substitution changes **terms** too...

```
50      _t[_] : ∀{Δ Γ A} → Tm Γ A → (σ : Sub Δ Γ) → Tm Δ (A [ σ ])
51      t[id] : ∀{Γ A} → (t : Tm Γ A) → t ≡ transp (Tm Γ) [id] (t t[ id ])
52      t[∘]  : ∀{Γ1 Γ2 Γ3 A} → (a : Tm Γ3 A)
53              → (f : Sub Γ1 Γ2) → (g : Sub Γ2 Γ3)
54              → a t[ g ∘ f ] ≡ transp (Tm Γ1) (sym ([∘] A f g)) ((a t[ g ]) t[ f ])
```

Some transport, but still doable...

Category with Families: wait...wait...

```
56      -- context extension (by a type)
57      _▹_ : (Γ : Con) → Ty Γ → Con -- Γ.σ
58      p   : ∀{Γ A} → Sub (Γ ▹ A) Γ
59      v   : ∀{Γ A} → Tm (Γ ▹ A) (A [ p ])
60      -- context morphism extension (by a term)
61      _,_ : ∀{Γ Δ A} → (σ : Sub Δ Γ) → Tm Δ (A [ σ ]) → Sub Δ (Γ ▹ A)
62      -- satisfying axioms
63      consl   : ∀{Γ Δ} → (f : Sub Γ Δ) → (A : Ty Δ) → (a : Tm Γ (A [ f ]))
64                → p ∘ (f , a) ≡ f
65      consr   : ∀{Γ Δ} → (f : Sub Γ Δ) → (A : Ty Δ) → (a : Tm Γ (A [ f ]))
66                → transp (Tm Γ) (trans (sym ([∘] A (f , a) p)) (cong (_[_] A) (consl f A a))) (v t[ (f , a) ]) ≡ a
67      consnat : ∀{Γ Δ Θ} → (f : Sub Γ Δ) → (A : Ty Δ) → (a : Tm Γ (A [ f ]))
68                → (g : Sub Θ Γ)
69                → (f , a) ∘ g ≡ (f ∘ g) , transp (Tm Θ) (sym ([∘] A g f)) (a t[ g ])
70      consid  : ∀{Δ}{A : Ty Δ} → p {Δ} {A} , v ≡ id
```

Writing definitions gets difficult!

Category with Families: Stop!

Wait! What happened?

Category with Families: Stop!

Wait! What happened?

Transport Hell: What is it?

What is a transport?

Transport rule

$\mathrm{transp} : P \to a = b \to P\ a \to P\ b.$

```
51      t[id] : ∀{Γ A} → (t : Tm Γ A) → t ≡ transp (Tm Γ) [id] (t t[ id ])
```

Example in our context

For any $a \in \mathrm{Tm}\ \Gamma\ A$, clearly $a = a[\mathrm{id}]$ by definition of the identity substitution.

However, It is *impossible* to state $a = a[\mathrm{id}]$ directly in Agda because equality is only between terms of the same type. LHS has type A, and RHS $A[\mathrm{id}]$.

Solution: **transport** a from type A to $A[\mathrm{id}]$ along $A = A[\mathrm{id}]$, or vice versa.

Transport: another example from CwFs

This is the Π-type (dependent function type), and its constructor lam (dependent functions).

```
77      Π : ∀{Γ} → (A : Ty Γ) → Ty (Γ ▹ A) → Ty Γ
78      lam : ∀{Γ A B} → Tm (Γ ▹ A) B → Tm Γ (Π A B)
79      Π[] : ∀{Γ A B Δ} → (σ : Sub Δ Γ) → (Π A B) [ σ ]
80        ≡ (Π (A [ σ ]) (B [ (σ ∘ p)
81           , transp (Tm (Δ ▹ (A [ σ ]))) (sym ([∘] A p σ)) (v {Δ} {A [ σ ]}) ]))
82      lam[] : ∀{Γ A B Δ}{b : Tm (Γ ▹ A) B} → (σ : Sub Δ Γ) →
83        (lam b) t[ σ ] ≡ transp (Tm Δ) (sym (Π[] σ))
84        (lam (b t[ (σ ∘ p) , transp (Tm (Δ ▹ (A [ σ ]))) (sym ([∘] A p σ)) (v {Δ} {A [ σ ]})]))
```

This is difficult to even formulate, becase one needs to combine equalities (via transitivity) to reach the desired types.

These stack up in the future as the terms get more and more **opaque**.

Transport: How bad is it?

Question: This (applying equalities) seems to be something that Agda can infer easily, no?

Transport: How bad is it?

Question: This (applying equalities) seems to be something that Agda can infer easily, no?

Answer: Unfortunately, there are too many “combinations” of equalities that can occur.

In fact, it is **undecidable** even for rewriting equalities in Generalized Algebraic Theories (includes CwFs)!

Million-dollar question

Is there a way to escape transport hell? Are there better representations?

Dependence in terms is the main culprit

```
44      -- substitution in types & terms
45      _[_] : ∀{Γ Δ} → Ty Γ → Sub Δ Γ → Ty Δ
46      [id] : ∀{Γ} → {A : Ty Γ} → A [ id ] ≡ A
47      [∘]  : ∀{Γ1 Γ2 Γ3} → (A : Ty Γ3) → (f : Sub Γ1 Γ2) → (g : Sub Γ2 Γ3) →
48             A [ (g ∘ f) ] ≡ A [ g ] [ f ]
49
50      _t[_] : ∀{Δ Γ A} → Tm Γ A → (σ : Sub Δ Γ) → Tm Δ (A [ σ ])
51      t[id] : ∀{Γ A} → (t : Tm Γ A) → t ≡ transp (Tm Γ) [id] (t t[ id ])
52      t[∘]  : ∀{Γ1 Γ2 Γ3 A} → (a : Tm Γ3 A)
53             → (f : Sub Γ1 Γ2) → (g : Sub Γ2 Γ3)
54             → a t[ g ∘ f ] ≡ transp (Tm Γ1) (sym ([∘] A f g)) ((a t[ g ]) t[ f ])
```

The equation on **terms** depends on the equations on **types**. What if we separate them?

Less-indexed CwFs: Natural Model

Indexing

What is indexing?

Vectors

$$\mathrm{List} : \mathrm{Set} \to \mathrm{Set} \qquad \mathrm{Vec} : A \to \mathbb{N} \to \mathrm{Set}$$
$$\mathrm{len} \ : \mathrm{List}\ A \to \mathbb{N}$$

We say lists are **less-indexed** than vectors.

Steve Awodey's natural model is a model for homotopy type theory (HoTT). It is less-indexed than traditional CwFs.

Agda (expectation)

$$
\begin{aligned}
&\mathrm{Con} && : \mathrm{Set}\\
&\mathrm{Sub} && : \mathrm{Con} \to \mathrm{Con} \to \mathrm{Set}\\
&\mathrm{Ty} && : \mathrm{Con} \to \mathrm{Set}\\
&\mathrm{Tm} && : (\Gamma : \mathrm{Con}) \to \mathrm{Ty}\ \Gamma \to \mathrm{Set}\\
&\mathrm{id} && : \mathrm{Sub}\ \Gamma\ \Gamma\\
&\_ \circ \_ && : \mathrm{Sub}\ \Delta\ \Gamma \to \mathrm{Sub}\ \Gamma\ \Theta \to \mathrm{Sub}\ \Delta\ \Theta\\
&\mathrm{idl} && : \mathrm{id} \circ \sigma = \sigma\\
&\mathrm{idr} && : \sigma \circ \mathrm{id} = \sigma\\
&\mathrm{assoc} && : (\delta \circ \gamma) \circ \theta = \delta \circ (\gamma \circ \theta)
\end{aligned}
$$

$$
\begin{aligned}
&\diamond && : \mathrm{Con}\\
&\varepsilon && : \mathrm{Sub}\ \Gamma\ \diamond\\
&\diamond\,\eta && : (\sigma : \mathrm{Sub}\ \Gamma\ \diamond) = \varepsilon\\
&\_[\_] && : \mathrm{Ty}\ \Gamma \to \mathrm{Sub}\ \Delta\ \Gamma \to \mathrm{Ty}\ \Delta\\
&\_[\_] && : \mathrm{Tm}\ \Gamma\ A \to (\sigma : \mathrm{Sub}\ \Delta\ \Gamma) \to \mathrm{Tm}\ \Delta\ A[\sigma]\\
&[\mathrm{id}] && : A[\mathrm{id}] = A\\
&[\circ] && : A[\sigma \circ \delta] = A[\sigma][\delta]\\
&[\mathrm{id}] && : a[\mathrm{id}] = a\\
&[\circ] && : a[\sigma \circ \delta] = a[\sigma][\delta]
\end{aligned}
$$

Definition: Natural Model

Less-indexed CwFs

$$\mathrm{Con} : \mathrm{Set}$$
$$\mathrm{Sub} : \mathrm{Con} \to \mathrm{Con} \to \mathrm{Set}$$

$$\mathrm{Ty} : \mathrm{Con} \to \mathrm{Set}$$
$$\mathrm{Tm} : (\Gamma : \mathrm{Con}) \to \mathrm{Ty}\ \Gamma \to \mathrm{Set}$$

$$\_[\_] : \mathrm{Ty}\ \Gamma \to \mathrm{Sub}\ \Delta\ \Gamma \to \mathrm{Ty}\ \Delta$$
$$\_[\_] : \mathrm{Tm}\ \Gamma\ A \to (\sigma : \mathrm{Sub}\ \Delta\ \Gamma) \to \mathrm{Tm}\ \Delta\ A[\sigma]$$

$$\mathrm{Con} : \mathrm{Set}$$
$$\mathrm{Sub} : \mathrm{Con} \to \mathrm{Con} \to \mathrm{Set}$$

$$\mathrm{Ty} : \mathrm{Con} \to \mathrm{Set}$$
$$\mathrm{Tm} : \mathrm{Con} \to \mathrm{Set}$$
$$\mathrm{ty} : \mathrm{Tm}\ \Gamma \to \mathrm{Ty}\ \Gamma$$

$$\_[\_] : \mathrm{Ty}\ \Gamma \to \mathrm{Sub}\ \Delta\ \Gamma \to \mathrm{Ty}\ \Delta$$
$$\_[\_] : \mathrm{Tm}\ A \to \mathrm{Sub}\ \Delta\ \Gamma \to \mathrm{Tm}\ \Delta$$
$$\mathrm{ty}\_[\_] : \mathrm{ty}\ a[\sigma] = (\mathrm{ty}\ a)[\sigma]$$

Natural Model at work

Types are the same.

```
45      -- Ty, Tm, ty, and action by substitution
46      Ty   : Con → Set k
47      _[_] : ∀{Γ Δ} → Ty Γ → Sub Δ Γ → Ty Δ
48      [id] : ∀{Γ}{A : Ty Γ} → A ≡ A [ id ]
49      [∘]  : ∀{Γ1 Γ2 Γ3}{A : Ty Γ1}{σ21 : Sub Γ2 Γ1}{σ32 : Sub Γ3 Γ2}
50               → A [ σ21 ] [ σ32 ] ≡ A [ σ21 ∘ σ32 ]
```

Terms are slightly different:

```
52      Tm    : Con → Set l
53      ty    : {Γ : Con} → Tm Γ → Ty Γ
54      _[_]t : ∀{Γ Δ} → Tm Γ → (γ : Sub Δ Γ) → Tm Δ
55      ty[]t : ∀{Γ Δ} → (γ : Sub Δ Γ) → (a : Tm Γ) → ty (a [ γ ]t) ≡ (ty a) [ γ ]
```

Natural Model at work

Now the transports disappear!

```
57      [id]t   : ∀{Γ}{a : Tm Γ} → a [ id ]t ≡ a
58      ty[id]t : ∀{Γ}{a : Tm Γ} → ty (a [ id ]t) ≡ (ty a) [ id ]
59
60      [∘]t    : ∀{Γ1 Γ2 Γ3}{σ21 : Sub Γ2 Γ1}{σ32 : Sub Γ3 Γ2}{a : Tm Γ1}
61                → a [ σ21 ∘ σ32 ]t ≡ a [ σ21 ]t [ σ32 ]t
62      ty[∘]t  : ∀{Γ1 Γ2 Γ3}{σ21 : Sub Γ2 Γ1}{σ32 : Sub Γ3 Γ2}{a : Tm Γ1}
63                → ty (a [ σ21 ∘ σ32 ]t) ≡ ty a [ σ21 ∘ σ32 ]
```

Beautiful!

Natural Model at work

Works for Π-types as well!

```
 97      Π : {Γ : Con} → (A : Ty Γ) → Ty (Γ ▹ A) → Ty Γ
 98      Π[] : ∀{Γ Δ}{A : Ty Γ}{B : Ty (Γ ▹ A)}{σ : Sub Δ Γ} →
 99        (Π A B) [ σ ] ≡ Π (A [ σ ]) (B [ σ ^ ])
100
101     lam : ∀{Γ A B}(b : Tm (Γ ▹ A)) → ty b ≡ B → Tm Γ
102     tylam : ∀{Γ A B}{b : Tm (Γ ▹ A)}{e : ty b ≡ B} → ty (lam b e) ≡ Π A B
103     lam[] : ∀{Γ A B Δ}{b : Tm (Γ ▹ A)}{e : ty b ≡ B}(σ : Sub Δ Γ) →
104       (lam b e) [ σ ]t ≡ lam (b [ σ ^ ]t)
105         (ty[]t (σ ^) b)
```

Beautiful! Also note that now we have more arguments than the syntax. A tradeoff!

Improvements over CwF

With less opaque terms, formalization is much more pleasant.

Formalized booleans 2 and Σ-types (dependent pairs).

```
117     Σ : {Γ : Con} → (A : Ty Γ) → Ty (Γ ▹ A) → Ty Γ
118     Σ[] : ∀{Γ Δ}{A : Ty Γ}{B : Ty (Γ ▹ A)}{σ : Sub Δ Γ} →
119       (Σ A B) [ σ ] ≡ Σ (A [ σ ]) (B [ σ ^ ])
120
121     pair : ∀{Γ A B} → (a : Tm Γ) → (b : Tm (Γ ▹ A)) → ty a ≡ A → ty b ≡ B → Tm Γ
122     typair : ∀{Γ A B}{a : Tm Γ}{ea : ty a ≡ A}{b : Tm (Γ ▹ A)}{eb : ty b ≡ B}
123       → ty (pair a b ea eb) ≡ Σ A B
124     pair[] : ∀{Γ A B Δ}{a : Tm Γ}{ea : ty a ≡ A}{b : Tm (Γ ▹ A)}{eb : ty b ≡ B}{σ : Sub Δ Γ}
125       → (pair a b ea eb) [ σ ]t ≡ pair (a [ σ ]t) (b [ σ ^ ]t)
126             (ty[]t σ a ▪ cong (λ t → t [ σ ]) ea)
127             (ty[]t (σ ^) b)
```

Transport is now equational reasoning!

Natural Model: Comparison

CwF	Less indexed
$\mathrm{Ty}\ \Gamma$	$\mathrm{Ty}\ \Gamma$
$\mathrm{Tm}\ G\ A$	$\mathrm{Tm}\ \Gamma, \mathrm{ty} : \mathrm{Tm}\ \Gamma \to \mathrm{Ty}\ \Gamma$
$\mathrm{Tm}\ \Gamma := \Sigma(A : \mathrm{Ty}\ G).\ \mathrm{Tm}\ \Gamma\ A$	$\mathrm{Tm}\ \Gamma\ A := \Sigma(a : \mathrm{Tm}\ \Gamma).(\mathrm{ty}\ a = A)$

Note that

$$\mathrm{Tm}\ \Gamma \quad \mapsto \quad A \to \Sigma(a : \mathrm{Tm}\ \Gamma).(\mathrm{ty}\ a = A) \quad \mapsto \quad \Sigma(A : \mathrm{Ty}\ \Gamma).\Sigma(a : \mathrm{Tm}\ \Gamma).(\mathrm{ty}\ a = A)$$

so the two notions (categories) are **equivalent**, but not isomorphic.

Comparison

CwF	Less indexed
transport hell	equational reasoning
close to syntax: no extra argument	extra arguments compared to the presyntax
less line of code	more lines of code
Generalized Algebraic Theory (indexed sorts)	Essentialy Algebraic Theory (partial functions)

Conclusion

What I have done/learnt in this project

- Categories with Families
- Explained Transport Hell
- Natural Model

Possible future work

Extending the current project:

- Formulate (and prove) metatheoretic properties
- Show formally that CwFs and Natural Model are equivalent

Related topics:

- Is one way better than the other? Is there disappearing work? Or is it purely syntactic sugar?
- Observational Type Theory (can decrease transport hell, but only some)
- Defining substitution recursively (poses other challenges)

Thank you!

Questions?