Strictifying Categories with Families

Ambrus Kaposi, Loïc Pujet

19 november 2024

Today's menu

Formally proving normalisation for dependent type theory

Today's menu

Formally proving normalisation for dependent type theory
$\sim$ with a side of categorical gluing $\sim$

I.

Gluing? Quésaco?

The PL theorist's approach

The PL theorist's approach

- First, we define a calculus of untyped terms:

$$\Lambda \quad := \quad x \mid t\,u \mid \lambda x.t \mid \ldots$$

The PL theorist's approach

- First, we define a calculus of untyped terms:

$$\Lambda \quad := \quad x \mid t\,u \mid \lambda x.t \mid \dots$$

- Then we define a family of typing judgments

$$\Gamma \vdash t : A \qquad \Gamma \vdash t \equiv u : A \qquad \Gamma \vdash t \rightsquigarrow u : A \qquad \dots$$

The PL theorist's approach

- First, we define a calculus of untyped terms:

$$\Lambda \quad := \quad x \mid t\,u \mid \lambda x.t \mid \ldots$$

- Then we define a family of typing judgments

$$\Gamma \vdash t : A \qquad \Gamma \vdash t \equiv u : A \qquad \Gamma \vdash t \rightsquigarrow u : A \qquad \ldots$$

which are inductively generated by typing rules:

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : \Pi(x : A).B} \qquad \frac{\Gamma \vdash t \equiv u : A}{\Gamma \vdash u \equiv t : A} \qquad \ldots$$

The PL theorist's approach

- If we want to prove a metatheoretical property of well-typed terms, our only option is induction on typing derivations

The PL theorist's approach

- If we want to prove a metatheoretical property of well-typed terms, our only option is induction on typing derivations
- For complex properties, a naive induction will not go through: we must strengthen the induction hypothesis

The PL theorist's approach

- If we want to prove a metatheoretical property of well-typed terms, our only option is induction on typing derivations
- For complex properties, a naive induction will not go through: we must strengthen the induction hypothesis
- The standard tool for this is logical relations
  - We interpret every well-formed type as a partial equivalence relation (PER) on terms, such that any term in the PER satisfies the desired property
  - Then we prove that every well-typed term is in the PER associated to its type, using induction on derivations
  - The choice of PERs is not always easy with dependent types!

The algebraist's approach

The algebraist's approach

Instead of reasoning on syntax, we shift our focus to a well-behaved category of models. For dependent type theory, a common option is Categories with Families (CwF)

The algebraist's approach

Instead of reasoning on syntax, we shift our focus to a well-behaved category of models. For dependent type theory, a common option is Categories with Families (CwF)

A category with families is the data of:

The algebraist's approach

Instead of reasoning on syntax, we shift our focus to a well-behaved category of models. For dependent type theory, a common option is Categories with Families (CwF)

A category with families is the data of:

- A category of contexts and subtitutions
- For every context Γ, a set of types $\mathrm{Ty}\ \Gamma$
- For every subst. $\sigma : \Delta \to \Gamma$, a function $\mathrm{Ty}\ \Gamma \to \mathrm{Ty}\ \Delta$
- For every context Γ and type A, a set of terms $\mathrm{Tm}\ \Gamma\ A$
- For every subst. $\sigma : \Delta \to \Gamma$, a function $\mathrm{Tm}\ \Gamma\ A \to \mathrm{Tm}\ \ \Delta\ \ A[\sigma]$
- Context extensions $\Gamma \triangleright A$, context projections $\mathrm{wk} : \Gamma \triangleright A \to \Gamma$ and $\mathrm{var}_0 : \mathrm{Tm}\ \ (\Gamma \triangleright A)\ \ (A[\mathrm{wk}])$

The algebraist's approach

Instead of reasoning on syntax, we shift our focus to a well-behaved category of models. For dependent type theory, a common option is Categories with Families (CwF)

A category with families is the data of:

- A category of contexts and subtitutions
- For every context Γ, a set of types $\mathsf{Ty}\ \Gamma$
- For every subst. $\sigma : \Delta \to \Gamma$, a function $\mathsf{Ty}\ \Gamma \to \mathsf{Ty}\ \Delta$
- For every context Γ and type A, a set of terms $\mathsf{Tm}\ \Gamma\ A$
- For every subst. $\sigma : \Delta \to \Gamma$, a function $\mathsf{Tm}\ \Gamma\ A \to \mathsf{Tm}\ \ \Delta\ \ A[\sigma]$
- Context extensions $\Gamma \triangleright A$, context projections $\mathsf{wk} : \Gamma \triangleright A \to \Gamma$ and $\mathsf{var}_0 : \mathsf{Tm}\ \ (\Gamma \triangleright A)\ \ (A[\mathsf{wk}])$
- and more...

The algebraist's approach

The algebraist's approach

The point is, CwFs are presented by an algebraic theory with sorts, terms and equations.

The algebraist's approach

The point is, CwFs are presented by an algebraic theory with sorts, terms and equations.

General theorems ensure the existence of an initial CwF
This initial model is the syntax!

The algebraist's approach

The point is, CwFs are presented by an algebraic theory with sorts, terms and equations.

General theorems ensure the existence of an initial CwF
This initial model is the "syntax"

(intrinsically well-typed terms quotiented by conversion.)

The algebraist's approach

We can reformulate most meta-theoretical properties to be about the initial CwF, without needing to mention raw terms.

The algebraist's approach

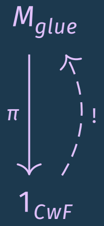
We can reformulate most meta-theoretical properties to be about the initial CwF, without needing to mention raw terms.

In order to prove them, we ditch logical relations for a newer and fancier tool: gluing

The algebraist's approach

We can reformulate most meta-theoretical properties to be about the initial CwF, without needing to mention raw terms.

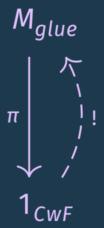
In order to prove them, we ditch logical relations for a newer and fancier tool: gluing



The algebraist's approach

We can reformulate most meta-theoretical properties to be about the initial CwF, without needing to mention raw terms.

In order to prove them, we ditch logical relations for a newer and fancier tool: gluing

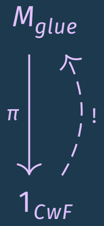


In the end, this is still a form of induction.

The algebraist's approach

We can reformulate most meta-theoretical properties to be about the initial CwF, without needing to mention raw terms.

In order to prove them, we ditch logical relations for a newer and fancier tool: gluing



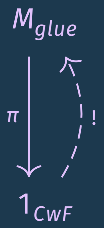
In the end, this is still a form of induction.

But instead of using PERs on raw syntax, we use proof-relevant predicates on well-typed syntax quotiented by conversion.

The algebraist's approach

We can reformulate most meta-theoretical properties to be about the initial CwF, without needing to mention raw terms.

In order to prove them, we ditch logical relations for a newer and fancier tool: gluing



In the end, this is still a form of induction.

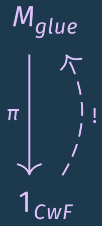
But instead of using PERs on raw syntax, we use proof-relevant predicates on well-typed syntax quotiented by conversion.

The resulting proof is arguably more principled and cleaner

The algebraist's approach

We can reformulate most meta-theoretical properties to be about the initial CwF, without needing to mention raw terms.

In order to prove them, we ditch logical relations for a newer and fancier tool: gluing



In the end, this is still a form of induction.

But instead of using PERs on raw syntax, we use proof-relevant predicates on well-typed syntax quotiented by conversion.

The resulting proof is arguably more principled and cleaner...at least on paper!

II.

Gluing in a proof assistant

Gluing in a proof assistant

Gluing in a proof assistant

First obstacle:

In order to formalise a proof of normalisation by gluing, we need quotient types, and function extensionality...

Gluing in a proof assistant

First obstacle:

In order to formalise a proof of normalisation by gluing, we need quotient types, and function extensionality...

...both of which are problematic in dependent type theory.

Gluing in a proof assistant

First obstacle:

In order to formalise a proof of normalisation by gluing, we need quotient types, and function extensionality...

...both of which are problematic in dependent type theory.

Good news: in 2024, this is not an insurmountable problem anymore. We have several solutions:

- Cubical type theory (available in Agda)
- Observational type theory (available in Coq and Agda)

Gluing in a proof assistant

The first step is a definition of the initial CwF
Most natural option: some form of inductive type

Gluing in a proof assistant

The first step is a definition of the initial CwF
Most natural option: some form of inductive type

More specifically, the initial CwF is best described as a quotient inductive-inductive type

Gluing in a proof assistant

The first step is a definition of the initial CwF
Most natural option: some form of inductive type

More specifically, the initial CwF is best described as a quotient inductive-inductive type

```
Con    : Set i
Sub    : Con → Con → Set j
_∘_    : ∀{Γ Δ} → Sub Δ Γ → ∀{Θ} → Sub Θ Δ → Sub Θ Γ
ass    : ∀{Γ Δ}{γ : Sub Δ Γ}{Θ}{δ : Sub Θ Δ}{Ξ}{θ : Sub Ξ Θ} → ((γ ∘ δ) ∘ θ) ~ (γ ∘ (δ ∘ θ))
id     : ∀{Γ} → Sub Γ Γ
idl    : ∀{Γ Δ}{γ : Sub Δ Γ} → (id ∘ γ) ~ γ
idr    : ∀{Γ Δ}{γ : Sub Δ Γ} → (γ ∘ id) ~ γ
◇      : Con
ε      : ∀{Γ} → Sub Γ ◇
◇η     : ∀{Γ}{σ : Sub Γ ◇} → σ ~ (ε {Γ})

Ty     : Con → Set k
_[_]T  : ∀{Γ} → Ty Γ → ∀{Δ} → Sub Δ Γ → Ty Δ
[∘]T   : ∀{Γ}{A : Ty Γ}{Δ}{γ : Sub Δ Γ}{Θ}{δ : Sub Θ Δ} → A [ γ ∘ δ ]T ~ A [ γ ]T [ δ ]T
[id]T  : ∀{Γ}{A : Ty Γ} → A [ id ]T ~ A

Tm     : (Γ : Con) → Ty Γ → Set l
_[_]t  : ∀{Γ}{A : Ty Γ} → Tm Γ A → ∀{Δ}(γ : Sub Δ Γ) → Tm Δ (A [ γ ]T)
[∘]t   : ∀{Γ}{A : Ty Γ}{a : Tm Γ A}{Δ}{γ : Sub Δ Γ}{Θ}{δ : Sub Θ Δ} → a [ γ ∘ δ ]t ~ a [ γ ]t [ δ ]t
[id]t  : ∀{Γ}{A : Ty Γ}{a : Tm Γ A} → a [ id ]t ~ a

_▹_    : (Γ : Con) → Ty Γ → Con
_,[_]_: ∀{Γ Δ}(γ : Sub Δ Γ) → ∀ {A A'} → A [ γ ]T ~ A' → Tm Δ A' → Sub Δ (Γ ▹ A)
p      : ∀{Γ A} → Sub (Γ ▹ A) Γ
q      : ∀{Γ A} → Tm (Γ ▹ A) (A [ p ]T)
▹β₁    : ∀{Γ Δ}{γ : Sub Δ Γ}{A}{a : Tm Δ (A [ γ ]T)} → p ∘ (γ ,[ ~refl ] a) ~ γ
▹β₂    : ∀{Γ Δ}{γ : Sub Δ Γ}{A}{a : Tm Δ (A [ γ ]T)} → q [ γ ,[ ~refl ] a ]t ~ a
▹η     : ∀{Γ Δ A}{γa : Sub Δ (Γ ▹ A)} → ((p ∘ γa) ,[ [∘]T ] (q [ γa ]t)) ~ γa
```

Gluing in a proof assistant

Then, we want to define indexed CwFs over the initial CwF
→ another, even more complex, list of fields

Gluing in a proof assistant

Then, we want to define indexed CwFs over the initial CwF
→ another, even more complex, list of fields

Finally, we want to define the glued model as an indexed CwF
→ welcome to transport hell!

Gluing in a proof assistant

The transport hell is much worse than what we are used to:

Gluing in a proof assistant

The transport hell is much worse than what we are used to:

In traditional proofs, terms are a first order object and substitutions are defined by recursion on terms.
Most substitution laws become definitional equalities

$$(\Pi\ A\ B)[\sigma] \equiv \Pi\ (A[\sigma])\ (B[\sigma \uparrow])$$

Gluing in a proof assistant

The transport hell is much worse than what we are used to:

In traditional proofs, terms are a first order object and substitutions are defined by recursion on terms.
Most substitution laws become definitional equalities

$$(\Pi\ A\ B)[\sigma] \equiv \Pi\ (A[\sigma])\ (B[\sigma \uparrow])$$

But in our QIIT formulation, substitutions are part of the algebra signature, and we only get propositional equalities

$$(\Pi\ A\ B)[\sigma] = \Pi\ (A[\sigma])\ (B[\sigma \uparrow])$$

Gluing in a proof assistant

In conclusion, normalisation by gluing is even less tractable than old fashioned normalisation proofs.

III.

Strictification

From propositional to definitional

Point of today's talk:
give an alternative definition of the initial CwF, for which almost all of the administrative equations become definitional equalities.

Strictifying groups

Suppose G is a group:

$$
\begin{array}{lll lll}
G & : & Set & unit_l & : & \forall x,\ e \times x = x \\
\_ \times \_ & : & G \to G \to G & unit_r & : & \forall x,\ x \times e = x \\
inv & : & G \to G & inv_l & : & \forall x,\ (inv\ x) \times x = e \\
e & : & G & inv_r & : & \forall x,\ x \times (inv\ x) = e
\end{array}
$$

$$assoc : \forall x\ y\ z,\ (x \times y) \times z = x \times (y \times z)$$

Strictifying groups

Suppose G is a group:

$$\begin{array}{lll} G & : & Set \\ \_\times\_ & : & G \to G \to G \\ inv & : & G \to G \\ e & : & G \end{array} \qquad \begin{array}{lll} unit_l & : & \forall x,\ e \times x = x \\ unit_r & : & \forall x,\ x \times e = x \\ inv_l & : & \forall x,\ (inv\ x) \times x = e \\ inv_r & : & \forall x,\ x \times (inv\ x) = e \end{array}$$

$$assoc : \forall x\ y\ z,\ (x \times y) \times z = x \times (y \times z)$$

Then G embeds in the group of permutations of G (Cayley's theorem)

$$Perm(G) := \{\ f : G \to G \ \mid\ \text{isBijective}\ f\ \}$$

Strictifying groups

Suppose G is a group:

$$
\begin{array}{lcl}
G & : & Set \\
\_ \times \_ & : & G \to G \to G \\
inv & : & G \to G \\
e & : & G \\
assoc & : & \forall x\ y\ z,\ (x \times y) \times z = x \times (y \times z)
\end{array}
\qquad
\begin{array}{lcl}
unit_l & : & \forall x,\ e \times x = x \\
unit_r & : & \forall x,\ x \times e = x \\
inv_l & : & \forall x,\ (inv\ x) \times x = e \\
inv_r & : & \forall x,\ x \times (inv\ x) = e
\end{array}
$$

Then G embeds in the group of permutations of G (Cayley's theorem)

$$Perm(G) := \{\ f : G \to G \ \mid\ \text{isBijective}\ f\ \}$$

Essential point: the group law on Perm(G) is given by function composition, which is definitionally associative and unital!

Strictifying groups

If we have access to a sort of proof-irrelevant propositions, we can define a group that is isomorphic to G:

$$G' := \{\, f : G \to G \mid \exists (g : G), f = \tau_g \,\}$$

With G' being definitionally associative and unital:

$$((f, f_\varepsilon) \circ (g, g_\varepsilon)) \circ (h, h_\varepsilon) \equiv (f, f_\varepsilon) \circ ((g, g_\varepsilon) \circ (h, h_\varepsilon))$$
$$(f, f_\varepsilon) \circ (id, id_\varepsilon) \equiv (f, f_\varepsilon)$$
$$(id, id_\varepsilon) \circ (f, f_\varepsilon) \equiv (f, f_\varepsilon)$$

Strictifying CwFs, first attempt

Cayley's theorem is an instance of the Yoneda lemma:
any category C embeds into the category $\hat{C} := \mathrm{Hom}(C^{op}, \mathrm{Set})$.

Strictifying CwFs, first attempt

Cayley's theorem is an instance of the Yoneda lemma:
any category $\mathcal{C}$ embeds into the category $\hat{\mathcal{C}} := \mathrm{Hom}(\mathcal{C}^{op}, \mathrm{Set})$.

The Yoneda generalises to categories with families:

Given a CwF $\mathcal{C}$, the presheaf category $\hat{\mathcal{C}}$ is naturally equipped with a CwF structure inherited from Set. Additionally, there is an embedding of CwFs $\mathcal{C} \to \hat{\mathcal{C}}$.

Strictifying CwFs, first attempt

Cayley's theorem is an instance of the Yoneda lemma:
any category C embeds into the category $\hat{C} := \mathrm{Hom}(C^{op}, \mathrm{Set})$.

The Yoneda generalises to categories with families:

Given a CwF C, the presheaf category $\hat{C}$ is naturally equipped with a CwF structure inherited from Set. Additionally, there is an embedding of CwFs $C \to \hat{C}$.

We can thus try the same trick: define C' to be the image of C under the embedding.

Strictifying CwFs, first attempt

$\mathcal{C}'$ is thus isomorphic to $\mathcal{C}$, and enjoys more definitional eqs:

- substitutions are definitionally associative
- substitutions are definitionally unital
- wk and var_0 satisfy their equations definitionally

Strictifying CwFs, first attempt

$\mathcal{C}'$ is thus isomorphic to $\mathcal{C}$, and enjoys more definitional eqs:

- substitutions are definitionally associative
- substitutions are definitionally unital
- wk and var_0 satisfy their equations definitionally

...BUT

The commutation of substitutions with binders is not definitional

$$(\Pi\ A\ B)[\sigma] \not\equiv \Pi\ \ (A[\sigma])\ \ (B[\sigma\uparrow])$$

Prefascist sets

Unfolding the computations, the reason why substitutions do not commute with binders boils down to natural transformations not being definitional

$$F_y \, (a \,|_f) \not\equiv (F_x \, a) \,|_f$$

Prefascist sets

Unfolding the computations, the reason why substitutions do not commute with binders boils down to natural transformations not being definitional

$$F_y\ (a\,|_f) \not\equiv (F_x\ a)\,|_f$$

In "Russian constructivism in a prefascist theory" (2020), Pédrot introduces prefascist sets, an alternative definition of presheaves that is strictly natural.

Strictifying CwFs, second attempt

If we reproduce our strictification construction using prefascist sets, we obtain a new CwF $\mathcal{C}''$, in which all* the administrative equalities are definitional.

Strictifying CwFs, second attempt

If we reproduce our strictification construction using prefascist sets, we obtain a new CwF $\mathcal{C}''$, in which all* the administrative equalities are definitional.

We formalised the construction of $\mathcal{C}''$ and its isomorphism with $\mathcal{C}$ in Agda.

Strictifying CwFs, second attempt

If we reproduce our strictification construction using prefascist sets, we obtain a new CwF $\mathcal{C}''$, in which all* the administrative equalities are definitional.

We formalised the construction of $\mathcal{C}''$ and its isomorphism with $\mathcal{C}$ in Agda. Surprisingly doable, even when the CwF is equipped with dependent products and booleans!

Back to our original goal

Applying our strictification construction to the initial CwF, it becomes much easier to construct gluing models. We were able to define a canonicity model (which computes normal forms for closed terms) in about 200 lines!

(the strictification construction is about 4000 lines)

Thank you!