Normalization for multimodal type theory

Daniel Gratzer

LICS 2022

Aarhus University

A perfunctory slide on modal type theories

We consider type theories extended with *modalities*:

- Our modalities are not fibered; they don't respect substitution
- Lots of different classes of modalities, we focus ((weak) dependent) right adjoints
- Specifically, we focus on MTT; a type theory parameterized a mode theory.

Don't hand-crafting a type theory for each set of modalities, just instantiate MTT!

A perfunctory slide on modal type theories

We consider type theories extended with *modalities*:

- Our modalities are not fibered; they don't respect substitution
- Lots of different classes of modalities, we focus ((weak) dependent) right adjoints
- Specifically, we focus on MTT; a type theory parameterized a mode theory.

Don't hand-crafting a type theory for each set of modalities, just instantiate MTT!

Question

Does MTT normalize regardless of instantiation?

Contributions

We present a normalization proof for MTT, a multimodal dependent type theory.

1. Conversion in MTT is decidable iff the collection of modalities is decidable
2. Type-checking MTT is decidable under the same conditions

The takeaway: MTT can be implemented for every sensible mode theory.

An inadequate summary of MTT

We must begin by recalling some of MTT, a multimodal type theory [Gra+20]

- Start with a mode theory $\mathcal{M}$
- Add a distinct copies of MLTT for each $m : \mathcal{M}$
- Add a modal type for $\mu : n \longrightarrow m$

$$\frac{\Gamma.\{\mu\} \vdash M : A \mathbin{@} n}{\Gamma \vdash \mathsf{mod}_{\mu}(M) : \langle \mu \mid A \rangle \mathbin{@} m}$$

- The elimination rule for modalities is "pattern-matching" style.

Normalization for MTT

What makes normalization challenging?

- Surprisingly, it's not really the modalities!
- The real challenge is the amount of data in play.

Can quickly get buried beneath all the data.

Normalization for MTT

What makes normalization challenging?

- Surprisingly, it's not really the modalities!
- The real challenge is the amount of data in play.

Can quickly get buried beneath all the data.

Enter gluing! [AHS95; Str98; Alt+01; Fio02; AK16; Shu15; Coq19; SA21; Ste21]

Rather than constructing an algorithm, build a model and deduce normalization.

Normalization by Gluing for MTT

It's still quite challenging to construct a gluing model directly.

A few minor adjustments:

- A collection of CwFs becomes a collection of LCCCs.
- Morphisms only preserve some structures up to isomorphism.

We can still obtain normalization for the initial *strict* model of MTT...
but the more flexible structures let us work more abstractly.

The gluing category for mode m

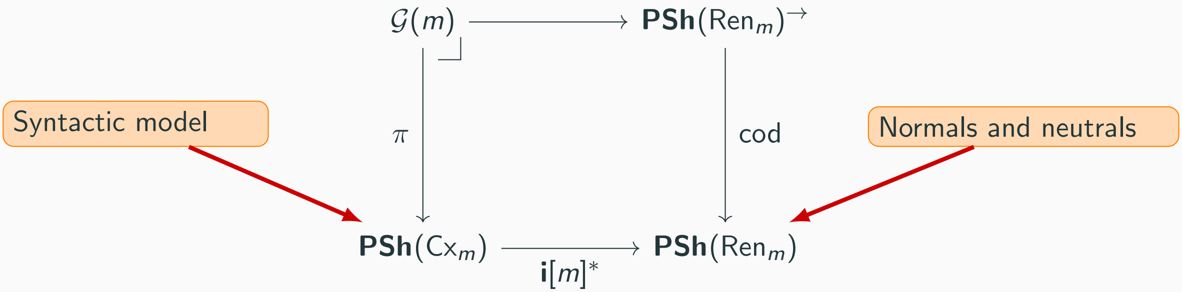
For a mode m, we have a category of contexts and of renamings: $\mathbf{i}[m] : \mathsf{Ren}_m \longrightarrow \mathsf{Cx}_m$

$$
\begin{array}{ccc}
\mathcal{G}(m) & \longrightarrow & \mathbf{PSh}(\mathsf{Ren}_m)^{\rightarrow} \\
{\scriptstyle \pi}\downarrow \lrcorner & & \downarrow{\scriptstyle \mathsf{cod}} \\
\mathbf{PSh}(\mathsf{Cx}_m) & \underset{\mathbf{i}[m]^*}{\longrightarrow} & \mathbf{PSh}(\mathsf{Ren}_m)
\end{array}
$$

Normalization model interprets mode m into $\mathcal{G}(m)$; π is a morphism of models.

The gluing category for mode m

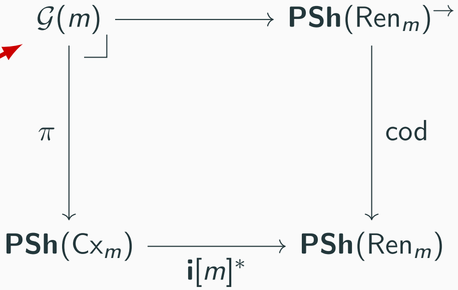
For a mode m, we have a category of contexts and of renamings: $\mathbf{i}[m] : \mathrm{Ren}_m \longrightarrow \mathrm{Cx}_m$



Normalization model interprets mode m into $\mathcal{G}(m)$; π is a morphism of models.

The gluing category for mode m

For a mode m, we have a category of contexts and of renamings: $\mathbf{i}[m] : \mathrm{Ren}_m \longrightarrow \mathrm{Cx}_m$



Proof-relevant predicates on syntax

Normalization model interprets mode m into $\mathcal{G}(m)$; π is a morphism of models.

Constructing a model in $\mathcal{G}(-)$

We now must interpret types, terms, etc. into $\mathcal{G}(-)$. To do this, we use MTT.

Theorem

There is a model of extensional MTT in $\mathcal{G}(-)$; modalities precompose with $-.\{\mu\}$.

Not the normalization model.

An internal language for the *network* of categories $\mathcal{G}(m)$.

Building the normalization model

We extend this internal language and define the normalization model internally.

- Interpretations of terms/types and the reify/reflect are done internally
- We extend *synthetic Tait computability* [SH21; SG20; Ste21] to the modal setting.

The complicated bookkeeping is now handled by the internal language!

Theorem

There exists a model of MTT in $\mathcal{G}(-)$ lying over the syntactic model in $\mathbf{PSh}(\mathsf{Cx}_{-})$.

Main results

From this model, we extract the main results

Theorem

A term in MTT with any mode theory has a unique normal form; there is a computable bijection between terms modulo definitional equality and normal forms.

Corollary

Conversion in MTT is decidable if and only if the mode theory is decidable.

Corollary

Every closed boolean in MTT is convertible to tt *or* ff.

How robust is the proof?

The proof is surprisingly extensible!

- We also consider an extension of MTT with a “crisp induction” principle [Shu18].
- Only local changes are required to the normalization model.
- We obtain the same set of theorems for this extended version of MTT.

Summary

In summary

- We define a normalization algorithm by building a particular model of MTT
- We crucially leverage MTT as the internal language of the gluing categories
- This approach extends modern gluing techniques to multimodal theories
- MTT is implementable for a wide class of mode theories.

Experimental implementation available by Stassen et al:
https://github.com/logsem/mitten_preorder

Questions?

The elimination rule for $\langle \mu \mid - \rangle$

The most complex part of MTT is the elimination principle for $\langle \mu \mid - \rangle$. Two perspectives...

1. An induction principle with the base case $\mathsf{mod}_\mu(M)$
2. The types are *weakly right orthogonal* to $\Gamma.(\nu \circ \mu \mid A) \longrightarrow \Gamma.(\nu \mid \langle \mu \mid A \rangle)$

Rules for modalities

$$\frac{\Gamma\ \mathsf{cx} \mathbin{@} m}{\Gamma.\{\mu\}\ \mathsf{cx} \mathbin{@} n} \qquad \frac{\Gamma.\{\mu\} \vdash A \mathbin{@} n}{\Gamma \vdash \langle \mu \mid A \rangle \mathbin{@} m}$$

$$\frac{\Gamma\ \mathsf{cx} \mathbin{@} m \qquad \Gamma.\{\mu\} \vdash A \mathbin{@} n}{\Gamma.(\mu \mid A).\{\mu\} \vdash \mathbf{v}_0 : A[\uparrow.\{\mu\}] \mathbin{@} n} \qquad \frac{\Gamma.\{\mu\} \vdash M : A \mathbin{@} n}{\Gamma \vdash \mathsf{mod}_\mu(M) : \langle \mu \mid A \rangle \mathbin{@} m}$$

$$\frac{\begin{array}{c} \nu : o \longrightarrow n \qquad \mu : n \longrightarrow m \\ \Gamma\ \mathsf{cx} \mathbin{@} m \qquad \Gamma.\{\mu\}.\{\nu\} \vdash A \mathbin{@} o \qquad \Gamma.(\mu \mid \langle \nu \mid A \rangle) \vdash B \mathbin{@} m \\ \Gamma.\{\mu\} \vdash M_0 : \langle \nu \mid A \rangle \mathbin{@} n \qquad \Gamma.(\mu \circ \nu \mid A) \vdash M_1 : B[\uparrow.\mathsf{mod}_\nu(\mathbf{v}_0)] \mathbin{@} m \end{array}}{\Gamma \vdash \mathsf{let}_\mu\ \mathsf{mod}_\nu(\_) \leftarrow M_0\ \mathsf{in}\ M_1 : B[\mathsf{id}.M_0] \mathbin{@} m}$$

$$\mathsf{let}_\mu\ \mathsf{mod}_\nu(\_) \leftarrow \mathsf{mod}_\nu(M_0)\ \mathsf{in}\ M_1 = M_1[\mathsf{id}.M_0]$$

Crisp induction principles for identity types

We force the following equivalence:

$$\mathsf{Id}_{\langle\mu|A\rangle}(\mathsf{mod}_\mu(M_0), \mathsf{mod}_\mu(M_1)) \simeq \langle \mu \mid \mathsf{Id}_A(M_0, M_1)\rangle$$

The key is the strengthened induction principle:

$$\frac{\begin{array}{c}\Gamma.(\mu \mid A).(\mu \mid A[\uparrow]).(\mu \mid \mathsf{Id}_{A[\uparrow^2]}(\mathbf{v}_1, \mathbf{v}_0)) \vdash B \mathbin{@} m \\ \Gamma.(\mu \mid A) \vdash M : B[\uparrow.\mathbf{v}_0.\mathbf{v}_0.\mathsf{refl}(\mathbf{v}_0)] \mathbin{@} m \\ \Gamma.\{\mu\} \vdash N_0, N_1 : A \mathbin{@} n \qquad \Gamma.\{\mu\} \vdash P : \mathsf{Id}_A(N_0, N_1) \mathbin{@} n\end{array}}{\Gamma \vdash \mathsf{J}^\mu(B, M, P) : B[\mathsf{id}.N_0.N_1.P] \mathbin{@} m}$$

$$\mathsf{J}^\mu(B, M, \mathsf{refl}(N)) = M[\mathsf{id}.N]$$