Modelling crisp type theory

Florrie Verity
Australian National University

Manchester category theory seminar
21 July 2023

Firstly

Why crisp type theory?

- relevance to homotopy type theory
  - "Kripke-Joyal forcing for type theory" Awodey, Gambino & Hazratpour, 2021
  - "Internal universes in models of HoTT" Licata, Orton, Pitts & Spitters, 2018
- models aren't very obvious

Plan

1. Modalities and modal type theory
   - the view of crisp type theory from logic and from HoTT

2. Modelling dependent type theory
   - natural models approach

3. Modelling crisp type theory
   - the abstract to the (slightly more) concrete

① Modalities and modal type theory

The "traditional" view from logic

possible world semantics

- modalities are operations on propositions

  e.g. in modal logic

  $\Box A$ "A is necessarily true"

  $\Diamond A$ "A is possibly true"

  also in linear logic: $!A$ and $?A$

- modelled categorically by (co)monads

  e.g. modal logic S4

  (K) $\Box(A \to B) \to (\Box A \to \Box B)$

  (T) $\Box A \to A$

  (4) $\Box A \to \Box\Box A$

the data of a comonad

(i) $\Box : \mathcal{C} \to \mathcal{C}$ a functor

(ii) $\varepsilon : \Box \Rightarrow \mathrm{id}_{\mathcal{C}}$

(iii) $\delta : \Box \Rightarrow \Box\Box$

$\Big\}$ natural transformations

- modal type theories originate in computer science to model "real" programming languages

The view from HoTT

points in a space "hanging together"

"Axiomatic cohesion"
- Lawvere 2007

"Cohesive homotopy type theory"
- Schreiber and Shulman 2012

$$\begin{tikzcd}[column sep=large]
\text{Spaces} \arrow[d, bend right=80, "\Pi"'] \arrow[d, bend left=80, leftarrow, "\nabla"] \arrow[d, bend right=30, leftarrow, "\Delta"'] \arrow[d, "\Gamma"] \\
\text{Sets}
\end{tikzcd}
\qquad \Pi \dashv \Delta \dashv \Gamma \dashv \nabla$$

generalised to $\rightsquigarrow$

$$\begin{tikzcd}[column sep=large]
\text{topological } \infty\text{-groupoids} \arrow[d, bend right=80, "\Pi"'] \arrow[d, bend left=80, leftarrow, "\nabla"] \arrow[d, bend right=30, leftarrow, "\Delta"'] \arrow[d, "\Gamma"] \\
\infty\text{-groupoids}
\end{tikzcd}
\qquad \Pi \dashv \Delta \dashv \Gamma \dashv \nabla$$

types in HoTT

$$
\begin{tikzcd}[row sep=huge]
\text{topological } \infty\text{-groupoids} \arrow[d, bend right=70, "\Pi"'] \arrow[d, "\Gamma"] \\
\infty\text{-groupoids} \arrow[u, bend left=40, "\Delta"] \arrow[u, bend right=70, "\nabla"']
\end{tikzcd}
$$

endofunctor $\rightsquigarrow$ perspective

(since ∇ and Δ are ff)

$$
\begin{tikzcd}[row sep=huge]
\text{topological } \infty\text{-groupoids} \arrow[d, "\flat"] \\
\text{topological } \infty\text{-groupoids} \arrow[u, bend left=50, "\int"] \arrow[u, bend right=50, "\sharp"']
\end{tikzcd}
$$

$$\int = \Delta\Pi$$

$$\text{idempotent} \begin{cases} \flat = \Delta\Gamma & \text{comonad} \\ \sharp = \nabla\Gamma & \text{monad} \end{cases}$$

Modalities are endofunctors on types/propositions.

Modalities

1) The "traditional" view from logic
2) The view from HoTT

How are these views connected?

Case study: crisp type theory

Crisp type theory - overview

	Pfenning & Davies' modal type theory (2001)	Shulman's crisp type theory (2018)
Context structure	"split-contexts" $\Delta \mid \Gamma$ ("modal" variables Δ, regular variables Γ)	"split-contexts" $\Delta \mid \Gamma$ ("crisp" variables Δ, regular variables Γ)
Typing	simply typed	dependently typed
Uses	- new presentation of "lax logic" (used in hardware verification) - staged computation	- allows you to prove, e.g., Brouwer's fixed point thm in HoTT. - internal universe of fibrant types from CCHM cubical sets model.

"A judgemental reconstruction of modal logic"

– Pfenning and Davies, 2001

Basic judgements in logic

1) A is a proposition — we know what counts as a verification of A

2) A is true — we know how to verify A

(presupposes A is a proposition)

used in inference rules to explain connectives
e.g. conjunction

$$\frac{A \text{ prop} \quad B \text{ prop}}{A \wedge B \text{ prop}} \text{ Formation}$$

$$\frac{A \text{ true} \quad B \text{ true}}{A \wedge B \text{ true}} \text{ Introduction}$$

$$\left.\begin{array}{c} \dfrac{A \wedge B \text{ true}}{A \text{ true}} \\[2ex] \dfrac{A \wedge B \text{ true}}{B \text{ true}} \end{array}\right\} \text{ Elimination}$$

Hypothetical judgements

- to explain the connective $\Rightarrow$, we need another form of judgement, written:

$$\underbrace{J_1, \ldots, J_n}_{\text{"hypotheses"}} \vdash J$$

J assuming J_1 through J_n

e.g. $A_1 \text{ true}, \ldots, A_n \text{ true} \vdash A \text{ true}$

- this allows us to introduce implications with the rule:

$$\frac{\Gamma, A \text{ true} \vdash B \text{ true}}{\Gamma \vdash A \Rightarrow B \text{ true}}$$

we know how to verify $A \Rightarrow B$ if we know how to verify B under hypothesis "A true"

- we may as well write our other rules in this judgement form, e.g.

$$\frac{A \text{ true} \quad B \text{ true}}{A \wedge B \text{ true}} \rightsquigarrow \frac{\Gamma \vdash A \text{ true} \quad \Gamma \vdash B \text{ true}}{\Gamma \vdash A \wedge B \text{ true}}$$

Rethinking our judgements...

Recall the second basic judgement

2) A is true

we know how to verify A

Let's give names to verifications and replace the above judgement with

$$M : A$$

"M is a proof of proposition A"

"M is a term of type A"

For hypothetical judgements, we name our hypothesised proof/term with a variable:

$$x : A$$

... leads to type theory

Example Conjunction

- Formation rule

$$\frac{A\ \mathrm{prop} \qquad B\ \mathrm{prop}}{A \wedge B\ \mathrm{prop}}$$

- Introduction rule

$$\frac{A\ \mathrm{true} \qquad B\ \mathrm{true}}{A \wedge B\ \mathrm{true}} \rightsquigarrow \frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N \rangle : A \wedge B}$$

- Elimination rule

$$\frac{A \wedge B\ \mathrm{true}}{A\ \mathrm{true}} \rightsquigarrow \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash \mathrm{fst}\, M : A}$$

$$\frac{A \wedge B\ \mathrm{true}}{B\ \mathrm{true}} \rightsquigarrow \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash \mathrm{snd}\, M : B}$$

- Computation rules

$$\mathrm{fst}\langle M, N \rangle \Rightarrow_R M$$

$$\mathrm{snd}\langle M, N \rangle \Rightarrow_R N$$

$$M : A \wedge B \Rightarrow_E \langle \mathrm{fst}\, M, \mathrm{snd}\, M \rangle$$

relate intro and elim rules

Pfenning and Davies' idea.

use this methodology of analysing judgements to incorporate modality in a type theory

Step 1) Introduce a third basic judgement

Definition (Validity)

1) If $\bullet \vdash A \; true$ then $A \; valid$.
2) if $A \; valid$ then $\Gamma \vdash A \; true$.

necessarily true

This may be used in hypothetical judgements

$$B_1 \; valid, \ldots, B_m \; valid \mid A_1 \; true, \ldots, A_n \; true \vdash A \; true,$$

abbreviated

$$\Delta \mid \Gamma \vdash A \; true.$$

Step 2) Internalise this judgement as a proposition

- Formation rule

$$\frac{A\ \mathsf{prop}}{\Box A\ \mathsf{prop}}$$

- Introduction rule

$$\frac{\Delta \mid \cdot \vdash A\ \mathsf{true}}{\Delta \mid \Gamma \vdash \Box A\ \mathsf{true}}$$

(follows from the definition of validity, updated with split contexts.

1) if $\Delta \mid \cdot \vdash A\ \mathsf{true}$ then $A\ \mathsf{valid}$.
2) if $A\ \mathsf{valid}$ then $\Delta \mid \Gamma \vdash A\ \mathsf{true}$.)

- Elimination rule

$$\frac{\Delta \mid \Gamma \vdash \Box A\ \mathsf{true} \qquad \Delta, A\ \mathsf{valid} \mid \Gamma \vdash C\ \mathsf{true}}{\Delta \mid \Gamma \vdash C\ \mathsf{true}}$$

Step 3) Perform the same move as before to "term : type" judgements

- Formation rule

$$\frac{A \text{ prop}}{\Box A \text{ prop}}$$

- Introduction rule

$$\frac{\Delta \mid \cdot \vdash A \text{ true}}{\Delta \mid \Gamma \vdash \Box A \text{ true}} \quad \rightsquigarrow \quad \frac{\Delta \mid \cdot \vdash M : A}{\Delta \mid \Gamma \vdash \text{box } M : \Box A}$$

- Elimination rule

$$\frac{\Delta \mid \Gamma \vdash \Box A \text{ true} \qquad \Delta, A \text{ valid} \mid \Gamma \vdash C \text{ true}}{\Delta \mid \Gamma \vdash C \text{ true}}$$

modal variable

$$\rightsquigarrow \quad \frac{\Delta \mid \Gamma \vdash M : \Box A \qquad \Delta, u :: A \mid \Gamma \vdash N : C}{\Delta \mid \Gamma \vdash \text{let box } u = M \text{ in } N : C}$$

- Computation rules

$$\text{let box } u = \text{box } M \text{ in } N \Longrightarrow_R N[\![M/u]\!]$$

replace all instances of u in N with M

$$M : \Box A \Longrightarrow_E \text{let box } u = M \text{ in } (\text{box } u)$$

Moving to Crisp type theory

$x : A \vdash B(x)$ type

- Crisp type theory is dependently-typed

  i.e. $x_1 : A_1, \ldots, x_n : A_n \vdash$ really means

  $x_1 : A_1, x_2 : A_2(x_1), x_3 : A_3(x_1, x_2), \ldots, x_n : A_n(x_1, \ldots, x_{n-1}) \vdash$

  – Substitution is a meta-operation on expressions (types & terms)

  $\phi[N/x]$ replace all instances of x in ϕ with N

  N.B. substitution is strictly functorial

- Terminology changes

  box modality $\Box A$ $\rightsquigarrow$ flat modality $\flat A$

  validity hypotheses $u :: A$ $\rightsquigarrow$ "crisp" hypotheses

"crisp context / context of crisp variables" $\rightsquigarrow$ $\Delta \mid \Gamma$ $\leftarrow$ "non-crisp context, context of non-crisp variables"

2 Modelling dependent type theory

Modelling dependent type theory

Let $\mathcal{C}$ be a category with a class of display maps $D \subseteq mor(\mathcal{C})$.

(all pullbacks of members of D exist and belong to D)

Ingredients of a type theory	$(\mathcal{C}, D)$
contexts Γ, Δ, Θ	objects in $\mathcal{C}$ Γ, Δ, Θ
types-in-context $\Gamma \vdash \alpha \text{ type}$	display maps $\alpha \downarrow \Gamma$
terms-in-context $\Gamma \vdash s : \alpha$	sections of display maps $\begin{tikzcd} \alpha \arrow[d] \\ \Gamma \arrow[u, bend left, "s"] \end{tikzcd}$

Substitution

1) of a term into another term

$$\frac{x:\alpha \vdash s:\beta \qquad y:\gamma \vdash t:\alpha}{y:\gamma \vdash s[t]:\beta}$$

which is functorial:

$$\frac{x:\alpha \vdash s:\beta \quad y:\gamma \vdash t:\alpha \quad z:\delta \vdash r:\gamma}{z:\delta \vdash s[t][r] = s[t\circ r]:\beta}$$

2) of a term into a type

$$\frac{x:\alpha \vdash \beta(x)\ \text{type} \qquad y:\gamma \vdash t:\alpha}{y:\gamma \vdash \beta(t)\ \text{type}}$$

which is functorial:

$$\frac{x:\alpha \vdash \beta(x)\ \text{type} \quad y:\gamma \vdash t:\alpha \quad z:\delta \vdash r:\gamma}{z:\delta \vdash \beta(t)(r) = \beta(t\circ r)\ \text{type}}$$

composition $s\circ t$

$$\begin{tikzcd} & \beta \arrow[d] \\ \gamma \arrow[r, "t"'] & \alpha \arrow[u, bend right, "s"'] \end{tikzcd}$$

$$\begin{tikzcd} & & \beta \arrow[d] \\ \delta \arrow[r, "r"'] & \gamma \arrow[r, "t"'] & \alpha \arrow[u, bend right, "s"'] \end{tikzcd} = \begin{tikzcd} & \beta \arrow[d] \\ \delta \arrow[r, "t\circ r"'] & \alpha \arrow[u, bend right, "s"'] \end{tikzcd}$$

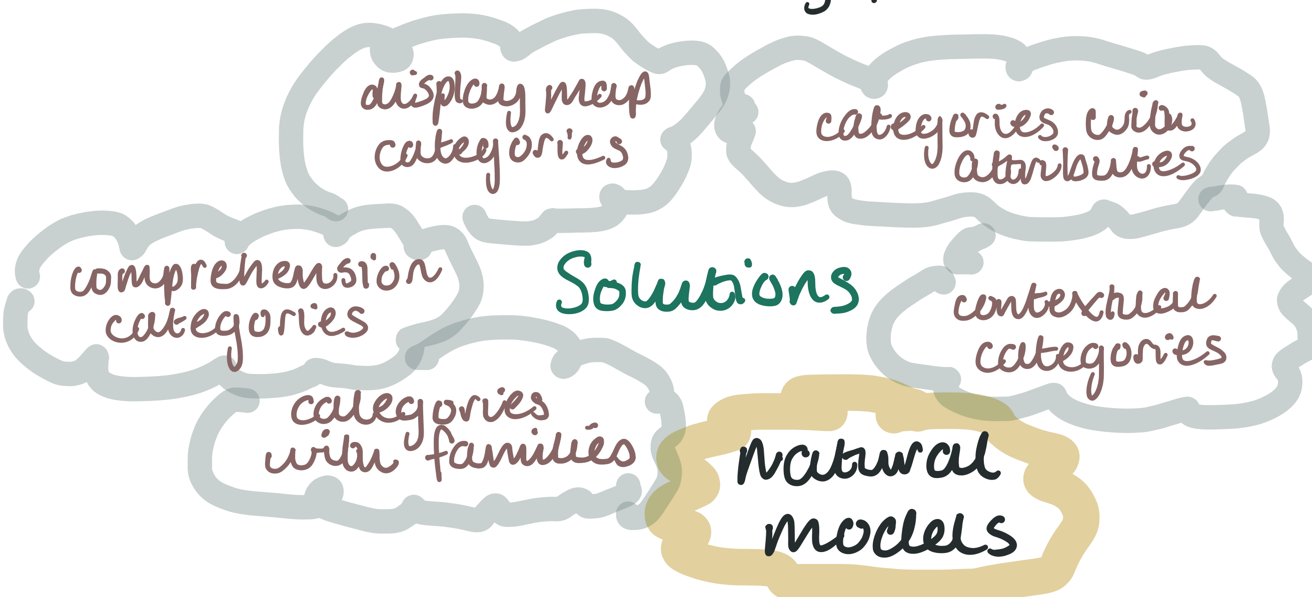
pullback

$$\begin{tikzcd} \beta(t) \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \beta \arrow[d] \\ \gamma \arrow[r, "t"'] & \alpha \end{tikzcd}$$

$$\begin{tikzcd} \beta[t][r] \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \beta[t] \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \beta \arrow[d] \\ \delta \arrow[r, "r"'] & \gamma \arrow[r, "t"'] & \alpha \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} \beta[t\circ r] \arrow[r] \arrow[d] \arrow[dr, phantom, "\lrcorner", very near start] & \beta \arrow[d] \\ \delta \arrow[r, "t\circ r"'] & \alpha \end{tikzcd}$$

$$\beta[t][r] \cong \beta[t\circ r]$$

Substitution is strictly functorial,
while pullback is only pseudofunctorial.



Advantages of natural models (Awodey, 2016)

- smaller distance between the syntax and the categorical model
- distinguishes between a type in context and extension by a single type
- type constructors are given by operations on a "universe" of types / terms

Definition A natural model is a category $\mathcal{C}$ with

objects $\Gamma, \Delta, \ldots$

morphisms $s : \Delta \to \Gamma$

contexts "$\Gamma \vdash$", substitutions

and

(i) a specified terminal object $1_{\mathcal{C}}$

empty context "$\bullet \vdash$"

(ii) presheaves $\mathcal{U}, \tilde{\mathcal{U}}$ over $\mathcal{C}$

$\mathcal{U}(\Gamma)$ set of types in context Γ

$\tilde{\mathcal{U}}(\Gamma)$ set of terms in context Γ

(iii) a natural transformation $ty : \tilde{\mathcal{U}} \longrightarrow \mathcal{U}$

$ty_{\Gamma} : \tilde{\mathcal{U}}(\Gamma) \longrightarrow \mathcal{U}(\Gamma)$

sends a term to its unique type

Observation

Given $\mathcal{U}, \tilde{\mathcal{U}} \in [\mathcal{C}^{op}, \mathrm{Set}]$ and $ty : \tilde{\mathcal{U}} \longrightarrow \mathcal{U}$,

by Yoneda we have

$$\frac{\alpha \in \mathcal{U}(\Gamma)}{\Gamma = よ_\Gamma \xrightarrow[\alpha]{} \mathcal{U}} \quad , \quad \frac{a \in \tilde{\mathcal{U}}(\Gamma)}{\Gamma = よ_\Gamma \xrightarrow[a]{} \tilde{\mathcal{U}}}$$

so "typing" corresponds to a commutative triangle

$$\begin{tikzcd} & \tilde{\mathcal{U}} \arrow[d, "ty"] \\ \Gamma \arrow[ur, "a"] \arrow[r, "\alpha"'] & \mathcal{U} \end{tikzcd}$$

$\Gamma \vdash a : \alpha$

(iv) specified represented pullbacks, i.e.

for each object Γ in $\mathcal{C}$ and each $\alpha \in \mathcal{U}(\Gamma)$, there is a specified pullback

$$
\begin{tikzcd}
\text{よ}_{\Gamma.\alpha} \arrow[r, "q_\alpha"] \arrow[d, "\text{よ} p_\alpha"'] \arrow[dr, phantom, "\lrcorner", very near start] & \tilde{\mathcal{U}} \arrow[d, "ty"] \\
\text{よ}_\Gamma \arrow[r, "\alpha"'] & \mathcal{U}
\end{tikzcd}
$$

$\Gamma, \alpha \vdash q_\alpha : \alpha[p_\alpha]$

in $\hat{\mathcal{C}}$, so we have $\Gamma.\alpha \xrightarrow{p} \Gamma$ in $\mathcal{C}$.

(NB. we will omit the "よ", as in:

$$
\begin{tikzcd}
\Gamma.\alpha \arrow[r, "q_\alpha"] \arrow[d, "p_\alpha"'] \arrow[dr, phantom, "\lrcorner", very near start] & \tilde{\mathcal{U}} \arrow[d, "ty"] \\
\Gamma \arrow[r, "\alpha"'] & \mathcal{U}
\end{tikzcd}
$$

)

Remarks

- (ii)–(iv) abbreviated by "$ty : \tilde{\mathcal{U}} \to \mathcal{U}$ is locally representable"
- substitution into a type is now given by composition, which is strictly functorial!

$$\begin{tikzcd}
 & & \tilde{\mathcal{U}} \arrow[d, "ty"] \\
\Delta \arrow[r, "s"'] & \Gamma \arrow[ur, "a"] \arrow[r, "\alpha"'] & \mathcal{U}
\end{tikzcd}$$

explicit substitution

$$\frac{\Gamma \vdash a : \alpha \qquad s : \Delta \to \Gamma}{\Delta \vdash a[s] : \alpha[s]}$$

- We can define structure-preserving maps between categories with natural model structure, so we have a category

NMCat objects - natural model categories
arrows - natural model functors

- We won't look at type constructors

Example - presheaf topos

Proposition Suppose $\mathcal{C}$ has a class of display maps $D \subseteq mor(\mathcal{C})$. Then there is a representable natural transformation

$$ty : \tilde{\mathcal{U}} \longrightarrow \mathcal{U}$$

in $\widehat{\mathcal{C}}$ defined as follows:

$$\begin{array}{c}
\tilde{\mathcal{U}}(\Gamma) := \left\{ \begin{tikzcd} & \Theta \arrow[d, "\alpha"] \\ \Gamma \arrow[ur, "a"] \arrow[r] & \Delta \end{tikzcd} \;\middle|\; \alpha \in D \right\} \\
\Big\downarrow \\
\mathcal{U}(\Gamma) := \left\{ \begin{tikzcd} & \Theta \arrow[d, "\alpha"] \\ \Gamma \arrow[r, "s"'] & \Delta \end{tikzcd} \;\middle|\; \alpha \in D \right\}
\end{array}$$

where the action of the presheaves is by precomposition.

3 Modelling crisp type theory

On modelling crisp type theory

– Licata, Orton, Pitts, Spitters 2018, referencing Shulman 2018

"very little is required of a category $\mathcal{C}$ for the presheaf topos $\hat{\mathcal{C}}$ to soundly interpret [crisp type theory] using the comonad $\flat$ ... Although the details remain to be worked out, it appears that ... the only additional condition needed is that this comonad is idempotent"

$$\begin{tikzcd} \text{constant presheaf on } B & \hat{A} \arrow[loop above, "\flat"] \arrow[d, bend left, "\Gamma"] & X \arrow[d, mapsto] \\ B \arrow[u, mapsto] & \mathrm{Set} \arrow[u, bend left, "\Delta"] \arrow[u, phantom, "\dashv"] & \{1 \xrightarrow{x} X\} \text{ global sections} \end{tikzcd}$$

Is it obvious how this is a model?

Zooming out

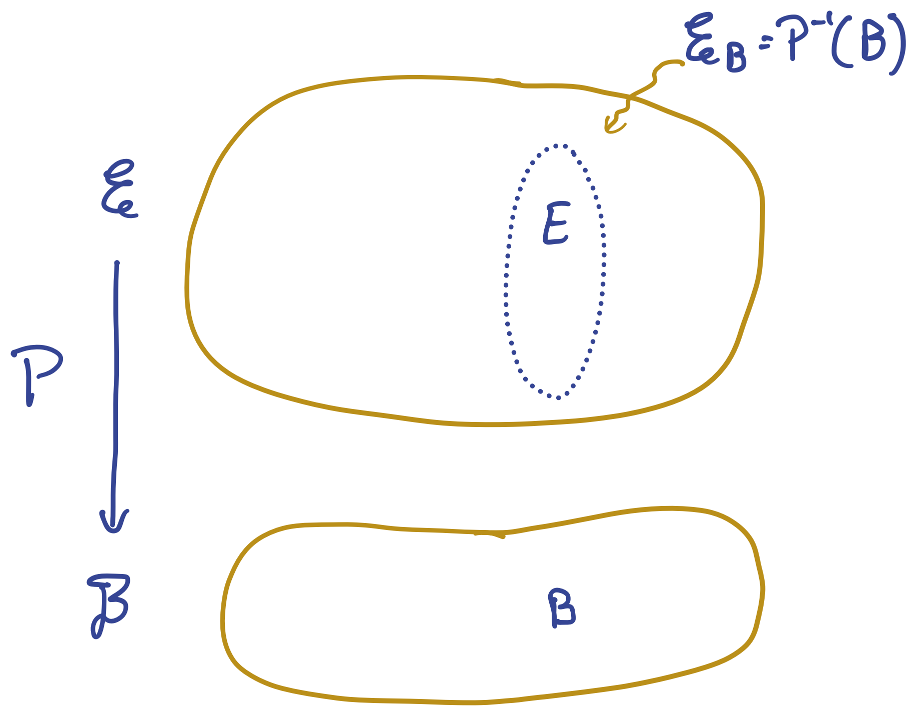
Question what are the features of the language and how might we model them more abstractly?

Feature 1: Split context

Grothendieck fibration

For a context $\Delta | \Gamma$, want to capture the dependency of Γ on Δ.

Ask for a functor, viewed as a display family of categories



$E \in \mathcal{E}_B \rightsquigarrow \Delta | \Gamma$

$B \in \mathcal{B} \rightsquigarrow \Delta$

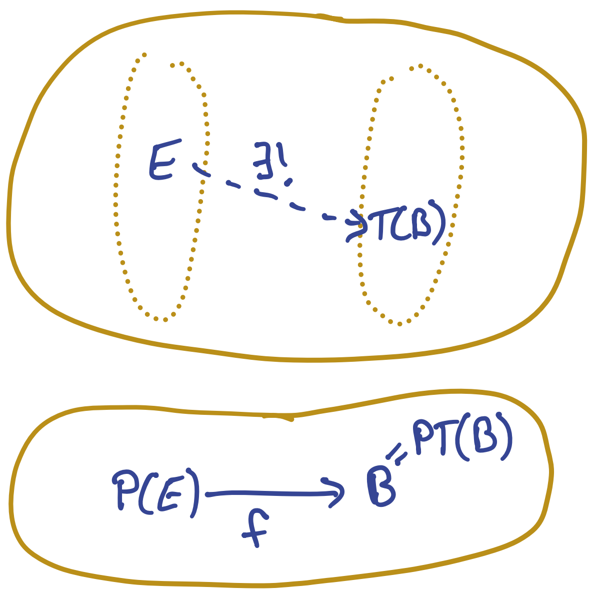
Feature 2: Empty contexts

The context may have no non-crisp variables.

$$\Delta \mid \cdot$$

Ask for a right adjoint right inverse to P.

$$\begin{tikzcd} \mathcal{E} \arrow[d, "P"', shift right=2] \arrow[d, phantom, "\dashv"] \\ \mathcal{B} \arrow[u, "T"', shift right=2, bend right] \end{tikzcd}$$



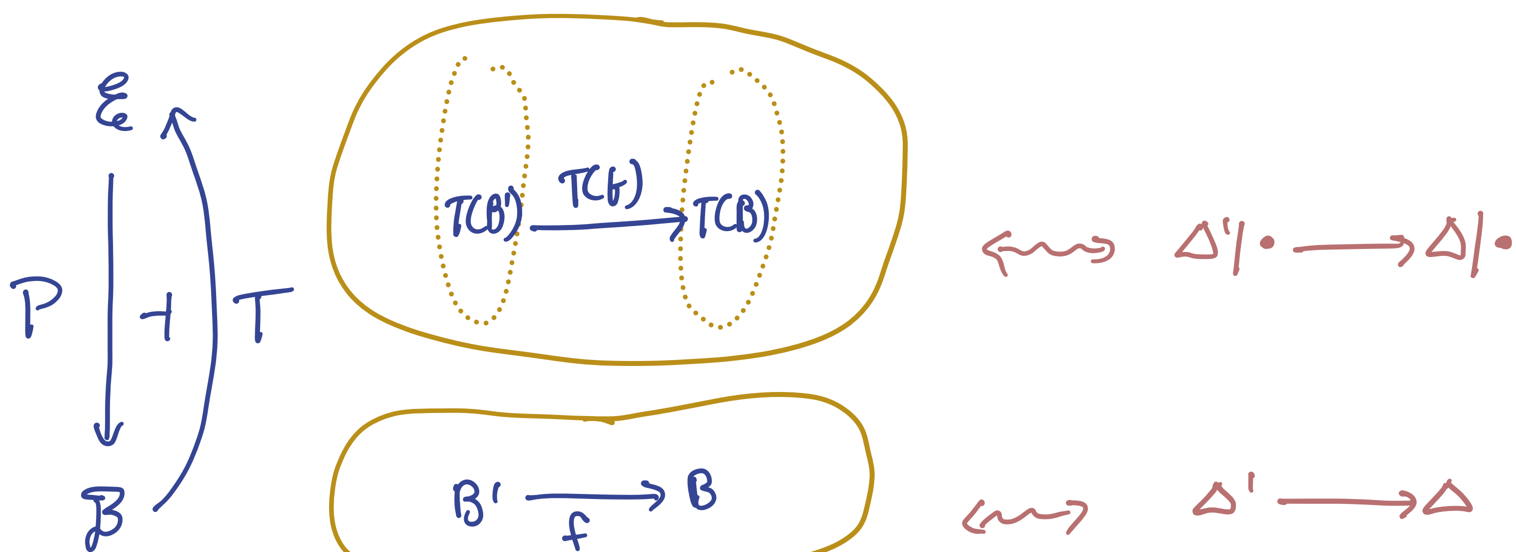
$$\iff \Delta' \mid \Gamma \longrightarrow \Delta \mid \cdot$$

$$\iff \Delta' \xrightarrow{f} \Delta$$

Consequence 1: each fibre $\mathcal{E}_B$ has a terminal object, $T(B)$

Consequence 2: fibrewise terminal objects are stable under reindexing

i.e. $T(f)$ is a cartesian lift of f



Feature 2: Empty contexts (continued)

The context may be empty:

$\cdot \mid \cdot$

Ask for a terminal object in $\mathcal{B}$

Consequence: $\mathcal{E}$ has a terminal object

Feature 3: Extension of the non-crisp context

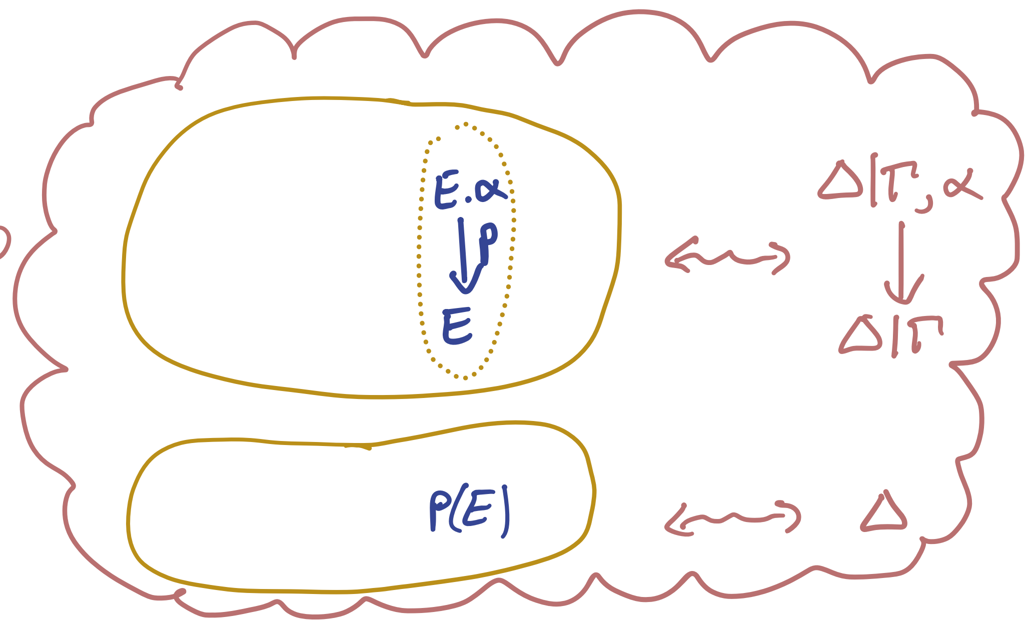
$$\frac{\Delta \mid \Gamma \vdash \alpha \ \text{type}}{\Delta \mid \Gamma, \alpha \ \text{context}}$$

type extension in $\mathcal{E}$

Ask for a locally representable map $ty_{\mathcal{E}} : \tilde{\mathcal{U}}_{\mathcal{E}} \longrightarrow \mathcal{U}_{\mathcal{E}}$ in $\hat{\mathcal{E}}$ with "fibrewise local representatives": in the specified pullback along an $E \xrightarrow{\alpha} \mathcal{U}_{\mathcal{E}}$,

$$\begin{tikzcd}
E.\alpha \arrow[r] \arrow[d, "p"'] \arrow[dr, phantom, "\lrcorner", very near start] & \tilde{\mathcal{U}}_{\mathcal{E}} \arrow[d, "ty_{\mathcal{E}}"] \\
E \arrow[r, "\alpha"'] & \mathcal{U}_{\mathcal{E}}
\end{tikzcd}$$

$E.\alpha \xrightarrow{p} E$ in $\mathcal{E}$ lies in the fibre $\mathcal{E}_{P(E)}$.



Consequences

1) the fibres are natural model categories:

In the fibre $\mathcal{E}_B$ over B there is

- a specified terminal object TB
- a locally representable map

$$\tilde{\mathcal{U}}|_{\mathcal{E}_B} \xrightarrow{ty} \mathcal{U}|_{\mathcal{E}_B}$$

2) The natural model structure is preserved between the fibres

Feature 4: extension of the crisp context

$$\frac{\Delta \mid \cdot \vdash \alpha \ \mathrm{type}}{\Delta, \alpha \mid \cdot \ \mathrm{context}}$$

type extension in $\mathcal{B}$

Ask that the following map defined using $ty_{\mathcal{E}}$ in $\widehat{\mathcal{E}}$ is locally representable in $\widehat{\mathcal{B}}$:

$$\begin{array}{ll}
\widetilde{\mathcal{U}}_{\mathcal{B}} := \widetilde{\mathcal{U}}_{\mathcal{E}} \circ T^{op} & \left(\mathcal{B}^{op} \xrightarrow{T^{op}} \mathcal{E}^{op} \xrightarrow{\widetilde{\mathcal{U}}_{\mathcal{E}}} \mathrm{Set}\right) \\
\Big\downarrow {\scriptstyle ty \circ T^{op}} & \\
\mathcal{U}_{\mathcal{B}} := \mathcal{U}_{\mathcal{E}} \circ T^{op} & \left(\mathcal{B}^{op} \xrightarrow{T^{op}} \mathcal{E}^{op} \xrightarrow{\mathcal{U}_{\mathcal{E}}} \mathrm{Set}\right)
\end{array}$$

Consequence: a type $\alpha \in \mathcal{U}_{\mathcal{E}}(T(B))$ corresponds to both

1) a map $\alpha : B \longrightarrow \mathcal{U}_{\mathcal{E}} \circ T^{op}$ in $\widehat{\mathcal{B}}$ $\quad \leftrightsquigarrow \Delta \vdash \alpha \ \mathrm{type}$

2) a map $\alpha : T(B) \longrightarrow \mathcal{U}_{\mathcal{E}}$ in $\widehat{\mathcal{E}}$ $\quad \leftrightsquigarrow \Delta \mid \cdot \vdash \alpha \ \mathrm{type}$

Summary

Let $P : \mathcal{E} \to \mathcal{B}$ be a functor.

Axioms

1) P has a right adjoint right inverse, T.
2) $\mathcal{B}$ has a specified terminal object.
3) There is a locally representable map

$$ty : \tilde{\mathcal{U}}_{\mathcal{E}} \to \mathcal{U}_{\mathcal{E}} \text{ in } \widehat{\mathcal{E}}$$

whose local representatives are given fibrewise.

4) $\tilde{\mathcal{U}}_{\mathcal{E}} \circ T^{op} \longrightarrow \mathcal{U}_{\mathcal{E}} \circ T^{op}$ in $\widehat{\mathcal{B}}$ is locally representable.

(+ ask for cartesian lifts of display maps in $\mathcal{B}$)

Claim: This models the context in crisp type theory.

Zooming back in

Let $\mathscr{C}$ be a category with

1) a terminal object
2) a class of display maps $\mathcal{D}$
3) an idempotent comonad $(\flat, \varepsilon_C : \flat C \to C)$ where $\flat$ preserves
   - the terminal object
   - display maps and their pullbacks

Theorem The above category possesses the structure of our abstract model.

Proof sketch

Let $\mathcal{B} = \mathcal{C}_b \hookrightarrow \mathcal{C}$

$\mathcal{E} = \mathcal{C} \downarrow \mathcal{C}_b$

$P = \mathrm{cod} : \mathcal{C} \downarrow \mathcal{C}_b \longrightarrow \mathcal{C}_b$

$T : \mathcal{C}^b \longrightarrow \mathcal{C} \downarrow \mathcal{C}_b$

full subcategory of objects c in $\mathcal{C}_0$ with $\varepsilon_c = id_c$

1) Show T is a right adjoint right inverse to P
↳ (a general result about comma categories)

2) Show that $\mathcal{C}_b$ has a terminal object

3) Define a locally representable map $\tilde{\mathcal{U}} \xrightarrow{t_y} \mathcal{U}$ in $\mathcal{C} \downarrow \mathcal{C}_b$

4) Show that the restriction of t_y to T is locally representable

$$\tilde{\mathcal{U}} \circ T^{op} \longrightarrow \mathcal{U} \circ T^{op}$$

Our conjectured model provides us with an example of such a category:

Let $\mathcal{C} = \hat{A}$

$b = \Delta\Gamma$

$$\begin{tikzcd} \hat{A} \arrow[loop right, "b"] \arrow[d, bend left, "\Gamma"] \\ Set \arrow[u, bend left, "\Delta"] \arrow[u, phantom, "\dashv"] \end{tikzcd}$$

$\mathcal{C}_b \simeq Set$

And we have:

$$\begin{tikzcd} \hat{A} \downarrow Set \arrow[d, "cod"'] \\ Set \arrow[u, bend right, "T"'] \arrow[u, phantom, "\dashv"] \end{tikzcd}$$

$$\begin{tikzcd} \mathcal{C} \downarrow \mathcal{C}_b \arrow[d, "cod"'] \\ \mathcal{C}_b \arrow[u, bend right, "T"'] \end{tikzcd}$$

Ongoing work

- Adding the modality $\square$ and the "let" constructor in the elimination rule

$$\begin{tikzcd} \tilde{\mathcal{U}} \circ T^{op} \arrow[r, "\square"] \arrow[d, "ty \circ T^{op}"'] & \tilde{\mathcal{U}} \circ T^{op} \arrow[d, "ty \circ T^{op}"] \\ \mathcal{U} \circ T^{op} \arrow[r, "\square"'] & \mathcal{U} \circ T^{op} \end{tikzcd} \quad \text{in } \widehat{\mathcal{B}}$$

- Formalising the abstract model as semantics

- Extending the Kripke-Joyal semantics in [Awodey, Gambino, Hazratpour, 2021] to crisp type theory.

Thanks