Structural Functoriality in Dependent Type Theory

Kenji Maillard[1]

[1]Gallinette Project Team, Inria, Nantes, France

MSFP 2026, Lisboa, 18/07/2026

The Beginning

Polarities: subtyping for datatypes #65

Closed

catalin-hritcu opened on Nov 29, 2014 — Member

`(x:int{x>1} * y:int{y>1})` is not a subtype of `(int * int)`

catalin-hritcu added **kind/bug** on Nov 29, 2014

nikswamy removed **kind/bug** on Dec 5, 2014

Structural Type-Casts

A few examples

Type-cast ::= producing a term of type B from a term of another type A

Definitional Equality

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \equiv B}{\Gamma \vdash t : B}$$

A few examples

Type-cast ::= producing a term of type B from a term of another type A

Subsumptive subtyping

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash t : B}$$

Coercive subtyping

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash \mathsf{coe}^{A}_{B}\ t : B}$$

Gradual Cast

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \langle B \Longleftarrow A \rangle\, t : B}$$

Observational substitution

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash p : A = B}{\Gamma \vdash \mathrm{subst}(p, t) : B}$$

Coercions as adapters

Adapters: **The data along which to cast** (McBride & Nordvall Forsberg)

$$\frac{}{\Gamma \vdash a : A \Rightarrow B} \qquad \frac{\Gamma \vdash a : A \Rightarrow B \qquad \Gamma \vdash t : A}{\Gamma \vdash t\langle a \rangle : B}$$

Coercions as adapters

Adapters: **The data along which to cast** (McBride & Nordvall Forsberg)

$$\frac{}{\Gamma \vdash a : A \Rightarrow B} \qquad \frac{\Gamma \vdash a : A \Rightarrow B \qquad \Gamma \vdash t : A}{\Gamma \vdash t\langle a \rangle : B}$$

Instances	$\Gamma \vdash a : A \Rightarrow B$
Subtyping	$\Gamma \vdash A \leq B$
Gradual	$\top$ (always hold)
Full	$\Gamma, x : A \vdash a : B$
Observational	$\Gamma \vdash p : A = T_1 \times \ldots \times T_n = B$

Coercions as adapters

Adapters: **The data along which to cast** (McBride & Nordvall Forsberg)

$$\frac{}{\Gamma \vdash a : A \Rightarrow B} \qquad \frac{\Gamma \vdash a : A \Rightarrow B \qquad \Gamma \vdash t : A}{\Gamma \vdash t\langle a \rangle : B}$$

Structurality: How do we propagate adapters across type formers?

Propagating Adapters on Pairs

$$\frac{\vdash A \qquad \vdash B}{\vdash A \times B}$$

Propagating Adapters on Pairs

$$\frac{\vdash A \qquad \vdash B}{\vdash A \times B} \qquad\qquad \frac{\vdash A_1 \leq A_2 \qquad \vdash B_1 \leq B_2}{\vdash A_1 \times B_1 \leq A_2 \times B_2}$$

Propagating Adapters on Pairs

$$\frac{\vdash A \qquad \vdash B}{\vdash A \times B} \qquad \frac{\vdash A_1 \leq A_2 \qquad \vdash B_1 \leq B_2}{\vdash A_1 \times B_1 \leq A_2 \times B_2}$$

$$\frac{\vdash f_A : A_1 \Rightarrow A_2 \qquad \vdash f_B : B_1 \Rightarrow B_2}{\vdash f_A \times f_B : A_1 \times B_1 \Rightarrow A_2 \times B_2}$$

Propagating Adapters on Pairs

$$\frac{\vdash A \qquad \vdash B}{\vdash A \times B} \qquad \frac{\vdash A_1 \leq A_2 \qquad \vdash B_1 \leq B_2}{\vdash A_1 \times B_1 \leq A_2 \times B_2}$$

$$\frac{\vdash f_A : A_1 \Rightarrow A_2 \qquad \vdash f_B : B_1 \Rightarrow B_2}{\vdash f_A \times f_B : A_1 \times B_1 \Rightarrow A_2 \times B_2}$$

$$(a, b)\langle f_A \times f_B \rangle \equiv (a\langle f_A \rangle, b\langle f_B \rangle)$$

Propagating Adapters on Dependent Pairs Structural Type-Casts

$$\frac{\vdash A \qquad x : A \vdash B}{\vdash \Sigma(x : A)\, B}$$

Propagating Adapters on Dependent Pairs

Structural Type-Casts

$$\frac{\vdash A \qquad x : A \vdash B}{\vdash \Sigma(x : A)\, B} \qquad\qquad \frac{\vdash A_1 \leq A_2 \qquad x : A_1 \vdash B_1 \leq B_2}{\vdash \Sigma(x : A_1)\, B_1 \leq \Sigma(x : A_2)\, B_2}$$

Propagating Adapters on Dependent Pairs

Structural Type-Casts

$$\frac{\vdash A \qquad x : A \vdash B}{\vdash \Sigma(x : A)\, B} \qquad \frac{\vdash A_1 \leq A_2 \qquad x : A_1 \vdash B_1 \leq B_2}{\vdash \Sigma(x : A_1)\, B_1 \leq \Sigma(x : A_2)\, B_2}$$

$$\frac{\vdash f_A : A_1 \Rightarrow A_2 \qquad x : A_1 \vdash f_B : B_1 \Rightarrow B_2[x\langle f_A \rangle / x]}{\vdash \Sigma(f_A, x.f_B) : \Sigma(x : A_1)\, B_1 \Rightarrow \Sigma(x : A_2)\, B_2}$$

Propagating Adapters on Dependent Pairs

$$\frac{\vdash A \qquad x : A \vdash B}{\vdash \Sigma(x : A)\, B} \qquad \frac{\vdash A_1 \leq A_2 \qquad x : A_1 \vdash B_1 \leq B_2}{\vdash \Sigma(x : A_1)\, B_1 \leq \Sigma(x : A_2)\, B_2}$$

$$\frac{\vdash f_A : A_1 \Rightarrow A_2 \qquad x : A_1 \vdash f_B : B_1 \Rightarrow B_2[x\langle f_A \rangle / x]}{\vdash \Sigma(f_A, x.f_B) : \Sigma(x : A_1)\, B_1 \Rightarrow \Sigma(x : A_2)\, B_2}$$

$$(a, b)\langle \Sigma(f_A, x.f_B) \rangle \equiv (a\langle f_A \rangle, b\langle f_B[a/x] \rangle)$$

Propagating Adapters on Lists

$$\frac{\vdash A}{\vdash \mathrm{list}\ A}$$

Propagating Adapters on Lists

$$\frac{\vdash A}{\vdash \mathrm{list}\ A} \qquad \frac{\vdash A_1 \leq A_2}{\vdash \mathrm{list}\ A_1 \leq \mathrm{list}\ A_2}$$

Propagating Adapters on Lists

$$\frac{\vdash A}{\vdash \mathrm{list}\, A} \qquad \frac{\vdash A_1 \leq A_2}{\vdash \mathrm{list}\, A_1 \leq \mathrm{list}\, A_2}$$

$$\frac{\vdash f_A : A_1 \Rightarrow A_2}{\vdash \mathrm{list}\, f_A : \mathrm{list}\, A_1 \Rightarrow \mathrm{list}\, A_2}$$

Propagating Adapters on Lists

$$\frac{\vdash A}{\vdash \mathrm{list}\ A} \qquad \frac{\vdash A_1 \leq A_2}{\vdash \mathrm{list}\ A_1 \leq \mathrm{list}\ A_2}$$

$$\frac{\vdash f_A : A_1 \Rightarrow A_2}{\vdash \mathrm{list}\ f_A : \mathrm{list}\ A_1 \Rightarrow \mathrm{list}\ A_2}$$

$$[]\langle \mathrm{list}\ f_A \rangle \equiv []$$

$$(\mathrm{hd} :: \mathrm{tl})\langle \mathrm{list}\ f_A \rangle \equiv \mathrm{hd}\ \langle f_A \rangle :: \mathrm{tl}\ \langle \mathrm{list}\ f_A \rangle$$

Propagating Adapters on Π-types

$$\frac{\vdash A \qquad x : A \vdash B}{\vdash \Pi(x : A)\, B}$$

Propagating Adapters on Π-types

$$\frac{\vdash A \qquad x : A \vdash B}{\vdash \Pi(x : A)\, B} \qquad\qquad \frac{\vdash A_2 \leq A_1 \qquad x : A_2 \vdash B_1 \leq B_2}{\vdash \Pi(x : A_1)\, B_1 \leq \Pi(x : A_2)\, B_2}$$

Propagating Adapters on Π-types

$$\frac{\vdash A \qquad x : A \vdash B}{\vdash \Pi(x : A)\, B} \qquad \frac{\vdash A_2 \leq A_1 \qquad x : A_2 \vdash B_1 \leq B_2}{\vdash \Pi(x : A_1)\, B_1 \leq \Pi(x : A_2)\, B_2}$$

$$\frac{\vdash f_A : A_2 \Rightarrow A_1 \qquad x : A_2 \vdash f_B : B_1[x\langle f_A \rangle / x] \Rightarrow B_2}{\vdash \Pi(f_A, x.f_B) : \Pi(x : A_1)\, B_1 \Rightarrow \Pi(x : A_2)\, B_2}$$

Propagating Adapters on Π-types

$$\frac{\vdash A \qquad x : A \vdash B}{\vdash \Pi(x : A)\, B} \qquad \frac{\vdash A_2 \leq A_1 \qquad x : A_2 \vdash B_1 \leq B_2}{\vdash \Pi(x : A_1)\, B_1 \leq \Pi(x : A_2)\, B_2}$$

$$\frac{\vdash f_A : A_2 \Rightarrow A_1 \qquad x : A_2 \vdash f_B : B_1[x\langle f_A \rangle / x] \Rightarrow B_2}{\vdash \Pi(f_A, x.f_B) : \Pi(x : A_1)\, B_1 \Rightarrow \Pi(x : A_2)\, B_2}$$

$$h\langle \Pi(f_A, x.f_B) \rangle\, a \equiv (h\ a\langle f_A \rangle)\langle f_B[a/x] \rangle$$

$$a\langle f_A \rangle : A_1 \qquad f_B[a/x] : B_1[a\langle f_A \rangle / x] \Rightarrow B_2[a/x]$$

Lawful Adapters

An Issue of Coherence

Assuming subtyping $\mathbb{N} \leq \mathbb{Z}$ and $\mathbb{Z} \leq \mathbb{Q}$, we can derive

$$\mathrm{list}\,\mathbb{N} \leq \mathrm{list}\,\mathbb{Q}$$

in two ways:

An Issue of Coherence

Assuming subtyping $\mathbb{N} \leq \mathbb{Z}$ and $\mathbb{Z} \leq \mathbb{Q}$, we can derive

$$\mathrm{list}\,\mathbb{N} \leq \mathrm{list}\,\mathbb{Q}$$

in two ways:

- Either we use transitivity and then congruence by list

An Issue of Coherence

Assuming subtyping $\mathbb{N} \leq \mathbb{Z}$ and $\mathbb{Z} \leq \mathbb{Q}$, we can derive

$$\mathrm{list}\,\mathbb{N} \leq \mathrm{list}\,\mathbb{Q}$$

in two ways:

- Either we use transitivity and then congruence by list
- Or we use congruence by list and then transitivity

An Issue of Coherence

Assuming subtyping $\mathbb{N} \leq \mathbb{Z}$ and $\mathbb{Z} \leq \mathbb{Q}$, we can derive

$$\mathrm{list}\,\mathbb{N} \leq \mathrm{list}\,\mathbb{Q}$$

in two ways:

- Either we use transitivity and then congruence by list
- Or we use congruence by list and then transitivity

Should $\mathrm{coe}^{\mathrm{list}\,\mathbb{N}}_{\mathrm{list}\,\mathbb{Q}}\ l$ and $\mathrm{coe}^{\mathrm{list}\,\mathbb{Z}}_{\mathrm{list}\,\mathbb{Q}}\left(\mathrm{coe}^{\mathrm{list}\,\mathbb{N}}_{\mathrm{list}\,\mathbb{Z}}\ l\right)$ be the same ?

Functorial Laws

Category structure: Adapters have identities and composition

$$\frac{}{\vdash \mathrm{id}_A : A \Rightarrow A} \qquad \frac{\vdash f : A \Rightarrow B \qquad \vdash g : B \Rightarrow C}{\vdash g \circ f : A \Rightarrow C}$$

$$t\langle \mathrm{id}_A \rangle \equiv t : A \qquad t\langle a' \circ\ a \rangle \equiv t\langle a \rangle \langle a' \rangle : A''$$

Functorial Laws

Category structure: Adapters have identities and composition

$$\frac{}{\vdash \mathrm{id}_A : A \Rightarrow A} \qquad \frac{\vdash f : A \Rightarrow B \qquad \vdash g : B \Rightarrow C}{\vdash g \circ f : A \Rightarrow C}$$

$$t\langle \mathrm{id}_A \rangle \equiv t : A \qquad t\langle a' \circ\ a \rangle \equiv t\langle a \rangle \langle a' \rangle : A''$$

Functoriality: that constructors on adapters need to preserve

$$\mathrm{list}\,\mathrm{id}_A \equiv \mathrm{id}_{\mathrm{list}\,A}$$
$$\mathrm{list}\,(g \circ f) \equiv (\mathrm{list}\,g) \circ (\mathrm{list}\,f)$$

Subsumptive and Coercive Subtyping

$\mathsf{MLTT}_{\mathrm{sub}}$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash t : B}$$

$\mathsf{MLTT}_{\mathrm{coe}}$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash \mathsf{coe}_B^A\ t : B}$$

Subsumptive and Coercive Subtyping

$$\mathsf{MLTT}_{\mathrm{sub}} \qquad \frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash t : B}$$

$$\mathsf{MLTT}_{\mathrm{coe}} \qquad \frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash \mathsf{coe}_B^A\ t : B}$$

Are $\mathsf{MLTT}_{\mathrm{sub}}$ and $\mathsf{MLTT}_{\mathrm{coe}}$ equivalent?

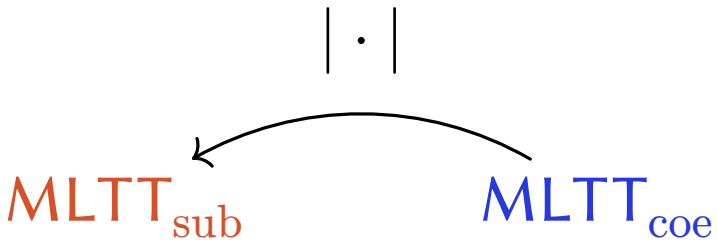
Subsumptive and Coercive Subtyping

$\mathsf{MLTT}_{\mathrm{sub}}$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash t : B}$$

$\mathsf{MLTT}_{\mathrm{coe}}$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash \mathsf{coe}_B^A\ t : B}$$



$$|\mathrm{coe}_B^A\, t| := |t|$$

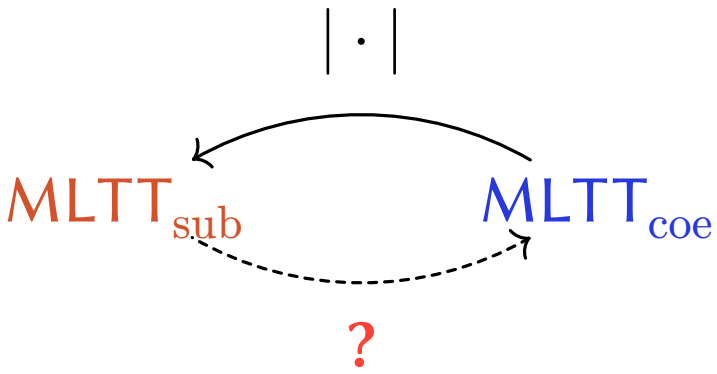
Subsumptive and Coercive Subtyping

$$\mathsf{MLTT}_{\mathrm{sub}}$$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash t : B}$$

$$\mathsf{MLTT}_{\mathrm{coe}}$$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \leq B}{\Gamma \vdash \mathsf{coe}_B^A\ t : B}$$



$$|\mathrm{coe}_B^A\, t| := |t|$$

From Subsumptive to Coercive Subtyping

Necessary conditions for an inverse to $|\cdot|$:

- $|t| = |\mathrm{coe}_A^A\, t| \implies t \equiv \mathrm{coe}_A^A\, t$ in $\mathsf{MLTT}_{\mathrm{coe}}$

From Subsumptive to Coercive Subtyping

Necessary conditions for an inverse to $|\cdot|$:

- $|t| = |\mathrm{coe}_A^A\, t| \implies t \equiv \mathrm{coe}_A^A\, t$ in $\mathrm{MLTT}_{\mathrm{coe}}$
- If $A \leq B$ and $B \leq C$,

$$|\mathrm{coe}_C^B(\mathrm{coe}_B^A\, t)| = |t| = |\mathrm{coe}_C^A\, t| \implies \mathrm{coe}_C^B(\mathrm{coe}_B^A\, t) \equiv \mathrm{coe}_C^A\, t \text{ in } \mathrm{MLTT}_{\mathrm{coe}}$$

From Subsumptive to Coercive Subtyping

Necessary conditions for an inverse to $|\cdot|$:

- $|t| = |\mathrm{coe}_A^A\, t| \implies t \equiv \mathrm{coe}_A^A\, t$ in $\mathrm{MLTT}_{\mathrm{coe}}$
- If $A \leq B$ and $B \leq C$,

$$|\mathrm{coe}_C^B(\mathrm{coe}_B^A\, t)| = |t| = |\mathrm{coe}_C^A\, t| \implies \mathrm{coe}_C^B(\mathrm{coe}_B^A\, t) \equiv \mathrm{coe}_C^A\, t \text{ in } \mathrm{MLTT}_{\mathrm{coe}}$$

And that's **all** we need !

From Subsumptive to Coercive Subtyping

Necessary conditions for an inverse to $|\cdot|$:

- $|t| = |\mathrm{coe}^A_A\, t| \implies t \equiv \mathrm{coe}^A_A\, t$ in $\mathrm{MLTT}_{\mathrm{coe}}$
- If $A \leq B$ and $B \leq C$,

$$|\mathrm{coe}^B_C(\mathrm{coe}^A_B\, t)| = |t| = |\mathrm{coe}^A_C\, t| \implies \mathrm{coe}^B_C(\mathrm{coe}^A_B\, t) \equiv \mathrm{coe}^A_C\, t \text{ in } \mathrm{MLTT}_{\mathrm{coe}}$$

$$\frac{\dfrac{\mathcal{D}_t}{\Gamma \vdash t : A} \quad \dfrac{\mathcal{D}_{\mathrm{sub}}}{\Gamma \vdash A \leq B}}{\Gamma \vdash t : B} \rightsquigarrow \frac{\dfrac{[\![\mathcal{D}_t]\!]}{[\![\Gamma]\!] \vdash [\![t]\!] : [\![A]\!]} \quad \dfrac{[\![\mathcal{D}_{\mathrm{sub}}]\!]}{[\![\Gamma]\!] \vdash [\![A]\!] \leq [\![B]\!]}}{[\![\Gamma]\!] \vdash \mathrm{coe}^A_B\, [\![t]\!] : [\![B]\!]}$$

From Subsumptive to Coercive Subtyping

Necessary conditions for an inverse to $|\cdot|$:

- $|t| = |\mathrm{coe}^A_A\, t| \implies t \equiv \mathrm{coe}^A_A\, t$ in $\mathrm{MLTT}_{\mathrm{coe}}$
- If $A \leq B$ and $B \leq C$,

$$|\mathrm{coe}^B_C(\mathrm{coe}^A_B\, t)| = |t| = |\mathrm{coe}^A_C\, t| \implies \mathrm{coe}^B_C(\mathrm{coe}^A_B\, t) \equiv \mathrm{coe}^A_C\, t \text{ in } \mathrm{MLTT}_{\mathrm{coe}}$$

Elaboration Erasure $|\cdot| : \mathrm{MLTT}_{\mathrm{coe}} \to \mathrm{MLTT}_{\mathrm{sub}}$ has $[\![\cdot]\!]$ as an inverse exactly when the functorial laws hold

$$\mathrm{coe}^A_A\, t \equiv t \qquad\qquad \mathrm{coe}^B_C(\mathrm{coe}^A_B\, t) \equiv \mathrm{coe}^A_C\, t$$

$\mathrm{MLTT}_{\mathrm{map}}$

A theory with definitional functor laws

The Issue with Functorial Laws

$\mathbf{MLTT}_{\mathbf{map}}$

```
Definition map `(f : A -> B) (l : list A) : list B :=
  match l with
  | [] => []
  | hd :: tl => f hd :: map f l
  end.
```

The Issue with Functorial Laws

MLTT$_{\text{map}}$

```
Definition map `(f : A -> B) (l : list A) : list B :=
  match l with
  | [] => []
  | hd :: tl => f hd :: map f l
  end.

Goal forall l, map id l = l.
(* Provable by induction *)
(* However, *not* a conversion *)
Fail reflexivity.
```

The Issue with Functorial Laws

$\mathbf{MLTT}_{\mathbf{map}}$

```
Definition map `(f : A -> B) (l : list A) : list B :=
  match l with
  | [] => []
  | hd :: tl => f hd :: map f l
  end.

Goal forall l, map id l = l.
(* Provable by induction *)
(* However, *not* a conversion *)
Fail reflexivity.

Goal forall f g l, map g (map f l) = map (g ∘ f) l.
(* Also not a  conversion *)
```

Challenges of Definitional Functor Laws

$\mathrm{MLTT}_{\mathrm{map}}$

Goal: Extend MLTT with an operator map on lists together with **definitional** functor laws

$$\mathrm{map}\ \ \mathrm{id}\ \ l \equiv l \qquad\qquad \mathrm{map}\ g\ (\mathrm{map}\ f\ l) \equiv \mathrm{map}\ (g \circ f)\ l$$

Does the resulting theory $\mathrm{MLTT}_{\mathrm{map}}$ enjoy:

- consistency?
- canonicity?
- normalization?
- decidability of conversion?

Challenges of Definitional Functor Laws

$\mathrm{MLTT}_{\mathrm{map}}$

Goal: Extend MLTT with an operator map on lists together with **definitional** functor laws

$$\mathrm{map}\ \ \mathrm{id}\ \ l \equiv l \qquad\qquad \mathrm{map}\ g\ (\mathrm{map}\ f\ l) \equiv \mathrm{map}\ (g \circ f)\ l$$

Does the resulting theory $\mathrm{MLTT}_{\mathrm{map}}$ enjoy:

- consistency?
- canonicity?
- normalization?
- decidability of conversion?

A Logical Relation for Normalization

$\mathbf{MLTT}_{\mathrm{map}}$

- Develop a reduction-based logical relation
- Characterizing normal-forms by iterated weak-head reduction
- Mechanized in Rocq
- Yields executable conversion and type checkers for $\mathrm{MLTT}_{\mathrm{map}}$

Key idea: Accumulate and merge map on neutrals

$$\ell^{\mathrm{ne}} \rightsquigarrow \mathrm{map}\ \ \mathrm{id}\ \ \ell^{\mathrm{ne}} \qquad\qquad \mathrm{map}\ g\ (\mathrm{map}\ f\ \ell^{\mathrm{ne}}) \rightsquigarrow \mathrm{map}\ (g \circ f)\ \ell^{\mathrm{ne}}$$

A Logical Relation for Normalization

$\mathrm{MLTT}_{\mathrm{map}}$

- Develop a reduction-based logical relation
- Characterizing normal-forms by iterated weak-head reduction
- Mechanized in Rocq
- Yields executable conversion and type checkers for $\mathrm{MLTT}_{\mathrm{map}}$

Key idea: Accumulate and merge map on neutrals

$$\ell^{\mathrm{ne}} \rightsquigarrow \mathrm{map}\ \ \mathrm{id}\ \ \ell^{\mathrm{ne}} \qquad\qquad \mathrm{map}\ g\ (\mathrm{map}\ f\ \ell^{\mathrm{ne}}) \rightsquigarrow \mathrm{map}\ (g \circ f)\ \ell^{\mathrm{ne}}$$

Divide and Check: Logical Relations, No Algorithms Attached
Thursday 23rd of July, 11:30am, FSCD

Hacking Functorial Map in Rocq

$\mathbf{MLTT}_{\mathbf{map}}$

Using non-linear rewrite rules to obtain a definitionally functorial map

```
Symbol map : forall {A B : Type}, (A -> B) -> list A -> list B.

Rewrite Rules functor_laws_list :=

  | @map ?A ?B ?f nil => @nil ?B
  | @map ?A ?B ?f (cons ?a ?l) => cons (?f ?a) (@map ?A ?B ?f ?l)

  | map (fun x => x) ?l => ?l
  | @map _ ?C ?g (@map ?A _ ?f ?l) =>
      @map ?A ?C (fun x => ?g (?f x)) ?l.
```

AdapTT

A theory with functorial type-casting

Return of the Adapters (AdapTT)

$$\frac{}{\Gamma \vdash a : A \Rightarrow B} \qquad \frac{\Gamma \vdash a : A \Rightarrow B \qquad \Gamma \vdash t : A}{\Gamma \vdash t\langle a \rangle : B}$$

Category structure: Adapters have identities and composition

$$\frac{}{\vdash \mathrm{id}_A : A \Rightarrow A} \qquad \frac{\vdash f : A \Rightarrow B \qquad \vdash g : B \Rightarrow C}{\vdash g \circ f : A \Rightarrow C}$$

$$t\langle \mathrm{id}_A \rangle \equiv t : A \qquad t\langle a' \circ\ a \rangle \equiv t\langle a \rangle \langle a' \rangle : A''$$

Return of the Adapters (AdapTT)

$$\frac{}{\Gamma \vdash a : A \Rightarrow B} \qquad \frac{\Gamma \vdash a : A \Rightarrow B \qquad \Gamma \vdash t : A}{\Gamma \vdash t\langle a \rangle : B}$$

Category structure: Adapters have identities and composition

$$\frac{}{\vdash \mathrm{id}_A : A \Rightarrow A} \qquad \frac{\vdash f : A \Rightarrow B \qquad \vdash g : B \Rightarrow C}{\vdash g \circ f : A \Rightarrow C}$$

$$t\langle \mathrm{id}_A \rangle \equiv t : A \qquad t\langle a' \circ\ a \rangle \equiv t\langle a \rangle \langle a' \rangle : A''$$

Functoriality: Type constructors preserve adapters functorially

Type Formers & Adapters

How do we specify the structural behaviour of **arbitrary** type formers ?

```
Inductive ForallVec (A : Type) (B : A -> Type)
  : forall (n : nat) (v : vec A n), Type :=
  (* ... *)
```

Challenges:

- Type formers may take an arbitrary telescope of parameters
- Parameters may be contra-variant
- Parameters depend on each other

Introducing AdapTT$_2$

Step 1: Package arbitrary parameters as telescopes:

$$\Gamma_{\mathrm{List}} := (X : \mathrm{Type}) \qquad \Gamma_{\times} := (X : \mathrm{Type}), (Y : \mathrm{Type})$$

Step 2: Type formation as instances of telescopes:

$$\frac{\Gamma \vdash A : \mathrm{Type} \qquad \Gamma \vdash B : \mathrm{Type}}{\Gamma \vdash A \times B : \mathrm{Type}} \quad \rightsquigarrow \quad \frac{\Gamma \vdash A : \mathrm{Type} \qquad \Gamma \vdash B : \mathrm{Type}}{\Gamma \vdash \begin{pmatrix} X \mapsto A \\ Y \mapsto B \end{pmatrix} : \Gamma_{\times}}$$

(Yoneda in hiding)

Introducing AdapTT$_2$

Step 1: Package arbitrary parameters as telescopes:

$$\Gamma_{\mathrm{List}} := (X : \mathrm{Type}) \qquad \Gamma_{\times} := (X : \mathrm{Type}), (Y : \mathrm{Type})$$

Step 2: Type formation as instances of telescopes:

$$\frac{\Gamma \vdash A : \mathrm{Type} \qquad \Gamma \vdash B : \mathrm{Type}}{\Gamma \vdash A \times B : \mathrm{Type}} \quad \rightsquigarrow \quad \frac{\Gamma \vdash A : \mathrm{Type} \qquad \Gamma \vdash B : \mathrm{Type}}{\Gamma \vdash \begin{pmatrix} X \mapsto A \\ Y \mapsto B \end{pmatrix} : \Gamma_{\times}}$$

Step 3: To construct adapters, substitutions are related through **transformations**:

$$\frac{\Gamma \vdash a : A_1 \Rightarrow A_2 \qquad \Gamma \vdash b : B_1 \Rightarrow B_2}{\Gamma \vdash a \times b : A_1 \times B_1 \Rightarrow A_2 \times B_2} \quad \rightsquigarrow \quad \frac{\Gamma \vdash a : A_1 \Rightarrow A_2 \qquad \Gamma \vdash b : B_1 \Rightarrow B_2}{\Gamma \vdash \begin{pmatrix} X \mapsto a \\ Y \mapsto b \end{pmatrix} : \begin{pmatrix} X \mapsto A_1 \\ Y \mapsto B_1 \end{pmatrix} \Rightarrow_{\Gamma_{\times}} \begin{pmatrix} X \mapsto A_2 \\ Y \mapsto B_2 \end{pmatrix}}$$

(2-Yoneda in hiding)

Computing functoriality of type formers

Challenges:

- Type formers may take an arbitrary telescope of parameters

  $\Rightarrow$ Instantiate type formers by a substitution

- Parameters may be contra-variant

- Parameters depend on each other

Computing functoriality of type formers

Challenges:

- Type formers may take an arbitrary telescope of parameters

  $\Rightarrow$ Instantiate type formers by a substitution

- Parameters may be contra-variant

  $\Rightarrow$ Add variance information

$$\Gamma_{\to} := (X : \mathrm{Type}_{-}), (Y : \mathrm{Type}_{+})$$

$$\frac{\Gamma \vdash A : \mathrm{Type} \qquad \Gamma \vdash B : \mathrm{Type}}{\Gamma \vdash (A, B) : \Gamma_{\to}} \qquad \frac{\Gamma \vdash a : \boldsymbol{A_2 \Rightarrow A_1} \qquad \Gamma \vdash b : B_1 \Rightarrow B_2}{\Gamma \vdash \begin{pmatrix} X \mapsto a \\ Y \mapsto b \end{pmatrix} : \begin{pmatrix} X \mapsto A_1 \\ Y \mapsto B_1 \end{pmatrix} \Rightarrow_{\Gamma_{\to}} \begin{pmatrix} X \mapsto A_1 \\ Y \mapsto B_2 \end{pmatrix}}$$

- Parameters depend on each other

Computing functoriality of type formers

Challenges:

- Type formers may take an arbitrary telescope of parameters
  $\Rightarrow$ Instantiate type formers by a substitution
- Parameters may be contra-variant
  $\Rightarrow$ Add variance information
- Parameters depend on each other
  $\Rightarrow$ Extend quantification over type families on a telescope

$$\Gamma_\Pi := (X : \mathrm{Type}_-), (Y : \mathbf{X}.\mathrm{Type}_+)$$

$$\frac{\Gamma \vdash A : \mathrm{Type} \quad \Gamma, x : A \vdash B : \mathrm{Type}}{\Gamma \vdash (A, B) : \Gamma_\Pi} \qquad \frac{\Gamma \vdash a : A_2 \Rightarrow A_1 \quad \Gamma, x : A_1 \vdash b : B_1[x := x\langle a \rangle] \Rightarrow B_2}{\Gamma \vdash \begin{pmatrix} X \mapsto a \\ Y \mapsto b \end{pmatrix} : \begin{pmatrix} X \mapsto A_1 \\ Y \mapsto B_1 \end{pmatrix} \Rightarrow_{\Gamma_\Pi} \begin{pmatrix} X \mapsto A_2 \\ Y \mapsto B_2 \end{pmatrix}}$$

Adapters are all you need

AdapTT$_2$ allows us to **formally** compute the functoriality of a given type-former.

On negative types (e.g Π, Σ): η-expansion provides the computational rules for free:

$$h\langle \Pi f_A . f_B \rangle \; a \equiv (h \; a\langle f_A \rangle)\langle f_B \rangle$$

On positive types (e.g List, W):

- A **theory of signatures**, in the style of Kaposi, Kovács et al.
- For parametrised indexed inductive types with variance.
- Derives the action of adapters on constructors.

AdapTT & AdapTT$_2$ at a glance

Adapters for type-casting

Functorial type formers

Functorial inductives

Categorical Models

From AdapTT$_2$ back to AdapTT

AdapTTing the models

Categorically, building on Natural models (MLTT):

- A **category** $\mathbf{Ctx}$ of contexts, substitutions
- A presheaf $\mathrm{Ty} : \mathbf{Ctx}^{\mathrm{op}} \to \mathbf{Set}$
  - Types over a context + action of substitutions
- A presheaf $\mathrm{Tm} : \mathbf{Ctx}^{\mathrm{op}} \to \mathbf{Set}$
  - Terms over a type and a context + action of substitutions
- A natural transformation $\mathrm{ty} : \mathrm{Tm} \to \mathrm{Ty}$
  - To each term in a context gives its type + compatibility with substitution
- Representability
  - Term variables, context extensions, ...

AdapTTing the models

Categorically, building on Natural models (MLTT):

- A **category** $\mathbf{Ctx}$ of contexts, substitutions
- A presheaf $\mathrm{Ty} : \mathbf{Ctx}^{\mathrm{op}} \to \mathbf{Cat}$
  - Types **and adapters** over a context + action of substitutions
- A presheaf $\mathrm{Tm} : \mathbf{Ctx}^{\mathrm{op}} \to \mathbf{Cat}$
  - Terms over a type and a context $+$ **action of adapters** + action of substitutions
- A natural transformation $\mathrm{ty} : \mathrm{Tm} \to \mathrm{Ty}$ + a discrete opfibration
  - To each term in a context gives its type + compatibility with substitution
- Representability
  - Term variables, context extensions, ...

AdapTTing the models

Categorically, building on Natural models (MLTT):

- A **category** $\mathbf{Ctx}$ of contexts, substitutions
- A presheaf $\mathrm{Ty} : \mathbf{Ctx}^{\mathrm{op}} \to \mathbf{Cat}$
  - Types **and adapters** over a context + action of substitutions
- A presheaf $\mathrm{Tm} : \mathbf{Ctx}^{\mathrm{op}} \to \mathbf{Cat}$
  - Terms over a type and a context $+$ **action of adapters**
- A natural transformation $\mathrm{ty} : \mathrm{Tm} \to \mathrm{Ty}$ + a discrete opf
  - To each term in a context gives its type + compatibility
- Representability
  - Term variables, context extensions, ...

We aren't the only ones !

Equivalent constructions to

- Split Comprehension Categories (Najmaei et al.)
- Generalized CwFs (Coraglia & Emmenegger)

AdapTTing the models, now in dimension 2! Categorical Models

A model of **AdapTT$_2$** consist of:

- A **2-category** $\mathbf{Ctx}$ of contexts, substitutions, **transformations**
- A presheaf $\mathrm{Ty} : \mathbf{Ctx}^{\mathrm{op}} \to \mathbf{Cat}$
  - Types **and adapters** over a context + action of substitutions
- A presheaf $\mathrm{Tm} : \mathbf{Ctx}^{\mathrm{op}} \to \mathbf{Cat}$
  - Terms over a type and context + action of substitutions
- A natural transformation $\mathrm{ty} : \mathrm{Tm} \to \mathrm{Ty}$ + a discrete opfibration
  - To each term in a context gives its type + compatibility with substitution
- Representability
  - Term variables, type variables, context extensions, ...

The SOGAT PoV on models of AdapTT

SOGAT = Second-Order Generalized Algebraic Theory

$$\mathsf{Ty} : \mathsf{Sort} \qquad \mathsf{Tm} : \mathsf{Ty} \to \mathsf{RepSort}$$

$$\mathsf{Ad} : \mathsf{Ty} \to \mathsf{Ty} \to \mathsf{Sort} \qquad \cdot\langle\cdot\rangle : \{A\,B{:}\,\mathsf{Ty}\} \to \mathsf{Ad}\ A\,B \to \mathsf{Tm}\ A \to \mathsf{Tm}\ B$$

$$\mathsf{id} : \{A{:}\,\mathsf{Ty}\} \to \mathsf{Ad}\ A\,A \qquad \circ : \{A\,B\,C{:}\,\mathsf{Ty}\} \to \mathsf{Ad}\ B\,C \to \mathsf{Ad}\ A\,B \to \mathsf{Ad}\ A\,C$$

$$\mathsf{idl} : \{A\,B{:}\,\mathsf{Ty}\}(a{:}\,\mathsf{Ad}\ A\,B) \to \mathsf{id} \circ a \equiv a \qquad \mathsf{idr} : \{A\,B{:}\,\mathsf{Ty}\}(a{:}\,\mathsf{Ad}\ A\,B) \to a \circ \mathsf{id} \equiv a$$

$$\mathsf{assoc} : \{A\,B\,C\,D{:}\,\mathsf{Ty}\}(a{:}\,\mathsf{Ad}\ A\,B)(b{:}\,\mathsf{Ad}\ B\,C)(c{:}\,\mathsf{Ad}\ C\,D) \to c \circ (b \circ a) \equiv (c \circ b) \circ a$$

$$\mathsf{adapt/id} : \{A{:}\,\mathsf{Ty}\}(t{:}\,\mathsf{Tm}\ A) \to t\langle\mathsf{id}\rangle \equiv t$$

$$\mathsf{adapt}/\circ : \{A\,B\,C{:}\,\mathsf{Ty}\}(a{:}\,\mathsf{Ad}\ A\,B)(b{:}\,\mathsf{Ad}\ B\,C)(t{:}\,\mathsf{Tm}\ A) \to t\langle b \circ a\rangle \equiv t\langle a\rangle\langle b\rangle$$

Morally working internally to presheaves over $\mathbf{Ctx}$

Relating AdapTT & AdapTT$_2$

Theorem Let $\mathcal{C}$ be a model of AdapTT, then $\mathrm{Psh}(\mathcal{C})$ can be equipped with a model of AdapTT$_2$.

Key Idea: From the SOGAT PoV

- Ty is an internal category in $\mathrm{Psh}(\mathcal{C})$
- Tm is an internal presheaf
- the internal dual $\mathrm{Ty}^{\mathrm{op}}$ models contravariance
- $\mathrm{Tm}\ A \to \mathrm{Ty}$ provides family variables

A marriage of **2-level type theory** with **directed type theory**.

Thank You !

Structural Subtyping in MLTT

Definitional Functoriality for Dependent (Sub)Types

AdapTT: Functoriality for Dependent Type Casts

Bonus slides

A Theory of Signatures for Inductive Types

A **parametrised indexed** inductive type Ind is:

- A context Γ of **parameters**
- A telescope $\Theta_I : \mathrm{Tel}_+(\Gamma)$ of **indices**
- A list $\vec{C}$ of **constructors**

A constructor C is:

- A telescope $\Theta_{\mathrm{norec}} : \mathrm{Tel}(\Gamma)$ of **non-recursive arguments**
- A list of **recursive arguments**
- An instantiation Θ_I of **indices** in $\Gamma \triangleright \Theta_{\mathrm{norec}}$

A recursive argument $(a_1 : A_1) \to \ldots \to (a_n : A_n) \to \mathrm{Ind}\ \vec{P}\ \vec{I}$ is:

- A telescope $\Theta_{\mathrm{rec}} := \varepsilon \triangleright A_1 \triangleright \ldots \triangleright A_n : \mathrm{Tel}_-(\Gamma \triangleright \Theta_{\mathrm{norec}})$ of **arity**
- An instantiation Θ_I of **indices** in $\Gamma \triangleright \Theta_{\mathrm{norec}} \triangleright \Theta_{\mathrm{rec}}$

A Theory of Signatures for Inductive Types

Example : Bounded W-types

- Parameters :
  $\Gamma_{\text{Ind}} := (A : \mathsf{Ty}^+), (B : (A.\mathsf{Ty}^-)$
- Indices : $\Theta_I := (n : \mathbb{N})$
- Constructor:
  - Non-recursive fields:
    $\Theta_{\text{norec}} := (n : \mathbb{N}), (a : A)$
  - Recursive field:
    - Telescope $\Theta_{\text{rec}} := (b : B\ a)$
    - Index instantiation : n
  - Index instantiation : $n + 1$

```
data BW (A : 𝒰) (B : A → 𝒰) : ℕ → 𝒰
where
  sup : (n : ℕ)
      → (a : A)
      → ((b : B a) → BW A B n)
      → BW A B (n+1)
```

Bibliography

[1] C. McBride and F. Nordvall Forsberg, “Functorial Adapters,” in *27th International Conference on Types for Proofs and Programs*, 2021.

[2] N. Najmaei, N. van der Weide, B. Ahrens, and P. R. North, “A Type Theory for Comprehension Categories with Applications to Subtyping.” 2025. doi: 10.48550/arXiv.2503.10868.

[3] G. Coraglia and J. Emmenegger, “Categorical models of subtyping.” 2023. doi: 10.48550/arxiv.2312.14600.

[4] A. Kaposi and A. Kovács, “Signatures and Induction Principles for Higher Inductive-Inductive Types,” *Logical Methods in Computer Science*, 2020, doi: 10.23638/LMCS-16(1:10)2020.

[6] A. Kovács, “Type-Theoretic Signatures for AlgebraicTheories and Inductive Types,” phdthesis, 2022. doi: 10.15476/ELTE.2022.070.

[5] A. Kaposi and J. von Raumer, “A Syntax for Mutual Inductive Families,” in *5th International Conference on Formal Structures for Computation and Deduction (FSCD 2020)*, Z. M. Ariola, Ed., in Leibniz International Proceedings in Informatics (LIPIcs), vol. 167. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020, pp. 23:1–23:21. doi: 10.4230/LIPIcs.FSCD.2020.23.