Logical Relations for a Logical Framework

presenting [**RS:logrels:12**] by Florian Rabe and Kristina Sojakova

Navid Roux[1]

KWARC Seminar

Computer Science, FAU Erlangen-Nürnberg

TODO: no date yet

[1] https://orcid.org/0000-0002-8348-2441

What are Logical Relations?

An (informal) class of proof methods used to prove

- strong normalization

  no infinite evaluation path $t_1 \to t_2 \to \ldots$

- type safety

  $t \colon T$ and $t \to^* t' \Rightarrow t' : T$

- program equivalence
  - correctness — e.g. of optimizations
  - theorems-for-free — terms of $\forall\alpha.\ \alpha \to \alpha$ are the identity
  - security-typed languages — $\mathrm{Prg}(s) \approx \mathrm{Prg}(s')$ for “sensitive” s, s'

see appendix for references

Why embed into a Logical Framework?

Theories of logical relations

- have been stated for many formal systems — System F, F_ω, CIC, ...
- have different flavors — syntactic, semantic, reflective
- usually given in meta languages

General Desire: formalize formal systems and their properties

in proof assistants or logical frameworks

Why embed into a Logical Framework?

Theories of logical relations

- have been stated for many formal systems — System F, F_ω, CIC, ...
- have different flavors — syntactic, semantic, reflective
- usually given in meta languages

General Desire: formalize formal systems and their properties

in proof assistants or logical frameworks

Problem: requires formalization of meta languages, too

even more difficult

Why embed into a Logical Framework?

Theories of logical relations

- have been stated for many formal systems — System F, F_ω, CIC, ...
- have different flavors — syntactic, semantic, reflective
- usually given in meta languages

General Desire: formalize formal systems and their properties

in proof assistants or logical frameworks

Problem: requires formalization of meta languages, too

even more difficult

Idea: Is there a general notion of logical relations *usable for formalization* and that *unifies* many other notions?

yes, and this talk gives one

Goals of This Talk

- Building intuition for logical relations and how they can be caputred in MMT/LF
- h
- formalization of *special cases* of logical relations in MMT/LF

[**RS:logrels:12**] gives much more general definitions and theorems than we do.

Prelims: MMT/LF as a Logical Framework

Definition (MMT/LF Grammar)

MMT/LF combines MMT's module system and the dependent type theory LF:

$$
\begin{array}{lcll}
\mathit{Thy} & ::= & T = \{\mathit{Decl}^*\} & \text{theory definition} \\
\mathit{Decl} & ::= & c\colon A\,[= A] \mid \textbf{include } T & \text{declarations in a theory} \\
\mathit{Morph} & ::= & v\colon S \to T = \{\mathit{Ass}^*\} & \text{morphism definition} \\
\mathit{Ass} & ::= & c := A \mid \textbf{include } v & \text{assignments in a morphism} \\
A & ::= & \mathtt{type} \mid c \mid x \mid A\ A \mid & \text{terms} \\
 & & \lambda x\!:\!A.\ A \mid \Pi x\!:\!A.\ A \mid A \to A &
\end{array}
$$

E.g. basic propositional logic (PL):

$$
\textbf{theory } \mathtt{PL} = \left\{
\begin{array}{r}
\mathtt{prop}\colon \mathtt{type} \\
\neg\colon \mathtt{prop} \to \mathtt{prop} \\
\wedge, \vee\colon \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop}
\end{array}
\right\}
$$

Prop. Logic in MMT/LF

Example (PL on the meta-level)

$$
\begin{array}{lcll}
P & ::= & A \mid \neg P \mid P \wedge P \mid P \vee P & \text{propositions} \\
A & ::= & \langle \text{unspecified} \rangle & \text{atoms}
\end{array}
$$

Use variables p, a for derived terms, and $\vdash$ for the usual intuitionistic proof calculus.

Prop. Logic in $\textsc{Mmt}$/LF

Example (PL on the meta-level)

$$\begin{array}{lcll} P & ::= & A \mid \neg P \mid P \wedge P \mid P \vee P & \text{propositions} \\ A & ::= & \langle\text{unspecified}\rangle & \text{atoms} \end{array}$$

Use variables p, a for derived terms, and $\vdash$ for the usual intuitionistic proof calculus.

$$\textbf{theory}\ \texttt{PL} = \left\{ \begin{array}{l} \texttt{prop} : \texttt{type} \\ \texttt{atom} : \texttt{type} \\ \\ \\ \\ \\ \\ \\ \end{array} \right\}$$

Prop. Logic in MMT/LF

Example (PL on the meta-level)

$$\begin{array}{lll} P & ::= & A \mid \neg P \mid P \wedge P \mid P \vee P \quad \text{propositions} \\ A & ::= & \langle\text{unspecified}\rangle \quad \text{atoms} \end{array}$$

Use variables p, a for derived terms, and $\vdash$ for the usual intuitionistic proof calculus.

$$\textbf{theory } \mathtt{PL} = \left\{ \begin{array}{r l} \mathtt{prop} : & \mathtt{type} \\ \mathtt{atom} : & \mathtt{type} \\ \mathtt{inj} : & \mathtt{atom} \to \mathtt{prop} \\ \neg : & \mathtt{prop} \to \mathtt{prop} \\ \wedge, \vee : & \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop} \end{array} \right\}$$

Prop. Logic in MMT/LF

Example (PL on the meta-level)

$$\begin{array}{lcll} P & ::= & A \mid \neg P \mid P \wedge P \mid P \vee P & \text{propositions} \\ A & ::= & \langle \text{unspecified} \rangle & \text{atoms} \end{array}$$

Use variables p, a for derived terms, and $\vdash$ for the usual intuitionistic proof calculus.

$$\textbf{theory } \mathtt{PL} = \left\{ \begin{array}{rl} \mathtt{prop}: & \mathtt{type} \\ \mathtt{atom}: & \mathtt{type} \\ \mathtt{inj}: & \mathtt{atom} \to \mathtt{prop} \\ \neg: & \mathtt{prop} \to \mathtt{prop} \\ \wedge, \vee: & \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop} \\ \vdash: & \mathtt{prop} \to \mathtt{type} \\ \wedge_{\mathtt{I}}: & \Pi p\, q: \mathtt{prop}.\ \vdash p \to \vdash q \to \vdash p \wedge q \\ \wedge_{\mathtt{EL}}: & \Pi p\, q: \mathtt{prop}.\ \vdash p \wedge q \to \vdash p \end{array} \right\}$$

Roadmap

1. $\Rightarrow$ Example 1: unary logical relations — along identity
2. Example 2: binary logical relations — along identity
3. Example 3: unary logical relation along non-trivial morphism $\approx$ Church

   To Curry operator

Example 1: Tertium Non Datur in Prop. Logic

Example (PL on the meta-level)

P	$::=$	$A \mid \neg P \mid P \wedge P \mid P \vee P$	propositions
A	$::=$	$\langle$unspecified$\rangle$	atoms

has become an MMT theory

Example 1: Tertium Non Datur in Prop. Logic

Example (PL on the meta-level)

$$\begin{array}{lcll} P & ::= & A \mid \neg P \mid P \wedge P \mid P \vee P & \text{propositions} \\ A & ::= & \langle\text{unspecified}\rangle & \text{atoms} \end{array}$$

has become an MMT theory

Theorem (Tertium Non Datur in PL)

If $\vdash a \vee \neg a$ for all atoms, then $\vdash p \vee \neg p$ for all propositions.

will become a logical relation

TND as a Meta Proof

Proof. ($\vdash p \lor \neg p$ for all prop. p)

Apply structural induction for the stronger claim $\mathrm{TND}_\_(-)$ where

- $\mathrm{TND}_A(a)$: $\vdash \mathrm{inj}(a) \lor \neg\,\mathrm{inj}(a)$ for atoms a
- $\mathrm{TND}_P(p)$: $\vdash p \lor \neg p$ for propositions p

For every non-terminal τ, $\mathrm{TND}_\tau(-)$ is a unary predicate on terms of τ!

$\rightsquigarrow$ a type-indexed family of relations

TND as a Meta Proof

Proof. ($\vdash p \vee \neg p$ for all prop. p)

Apply structural induction for the stronger claim $\mathrm{TND}\_(-)$ where

- $\mathrm{TND}_A(a)\colon\ \vdash \mathrm{inj}(a) \vee \neg\,\mathrm{inj}(a)$ for atoms a
- $\mathrm{TND}_P(p)\colon\ \vdash p \vee \neg p$ for propositions p

For every non-terminal τ, $\mathrm{TND}_\tau(-)$ is a unary predicate on terms of τ!

$\rightsquigarrow$ a type-indexed family of relations

Cases

- $\mathrm{TND}_A(a)$: by assumption

TND as a Meta Proof

Proof. ($\vdash p \vee \neg p$ for all prop. p)

Apply structural induction for the stronger claim $\mathrm{TND}\_(-)$ where

- $\mathrm{TND}_A(a)$: $\vdash \mathrm{inj}(a) \vee \neg\, \mathrm{inj}(a)$ for atoms a
- $\mathrm{TND}_P(p)$: $\vdash p \vee \neg p$ for propositions p

For every non-terminal τ, $\mathrm{TND}_\tau(-)$ is a unary predicate on terms of τ!

$\rightsquigarrow$ a type-indexed family of relations

Cases

- $\mathrm{TND}_A(a)$: by assumption
- $\mathrm{TND}_P(\mathrm{inj}(a))$: immediately by IH $\mathrm{TND}_A(a)$

TND as a Meta Proof

Proof. ($\vdash p \vee \neg p$ for all prop. p)

Apply structural induction for the stronger claim $\mathrm{TND}\_(-)$ where

- $\mathrm{TND}_A(a)$: $\vdash \mathrm{inj}(a) \vee \neg\,\mathrm{inj}(a)$ for atoms a
- $\mathrm{TND}_P(p)$: $\vdash p \vee \neg p$ for propositions p

For every non-terminal τ, $\mathrm{TND}_\tau(-)$ is a unary predicate on terms of τ!

$\rightsquigarrow$ a type-indexed family of relations

Cases

- $\mathrm{TND}_A(a)$: by assumption
- $\mathrm{TND}_P(\mathrm{inj}(a))$: immediately by IH $\mathrm{TND}_A(a)$
- $\mathrm{TND}_P(\neg p)$: $\vdash \neg p \vee \neg\neg p$
  1. by IH we have $\mathrm{TND}_P(p)$: $\vdash p \vee \neg p$
  2. perform $\vee_{\mathtt{E}}$ on IH
     - if $\vdash p$: then $\vdash \neg\neg p$; done by $\vee_{\mathtt{IR}}$.
     - if $\vdash \neg p$: by $\vee_{\mathtt{IL}}$.

TND as a Meta Proof (cont.)

Proof ($\vdash p \vee \neg p$ for all prop. p; cont.).

Cases

- $\mathrm{TND}_P(p_1 \wedge p_2)\colon\ \vdash (p_1 \wedge p_2) \vee \neg(p_1 \wedge p_2)$
  1. by IH we have $\mathrm{TND}_P(p_1)\colon\ \vdash p_1 \vee \neg p_1$ and $\mathrm{TND}_P(p_2)\colon\ \vdash p_2 \vee \neg p_2$
  2. perform $\vee_{\mathtt{E}}$ on both
     - if $\vdash p_1$ and $\vdash p_2$: apply $\vee_{\mathtt{IL}}$, $\wedge_{\mathtt{I}}$, done.
     - if at least one $\vdash \neg p_i$: apply $\vee_{\mathtt{IR}}$, $\neg_{\mathtt{I}}$, $\wedge_{\mathtt{Ei}}$, contradiction.
- $\mathrm{TND}_P(p_1 \vee p_2)\colon\ \vdash p_1 \vee p_2 \vee \neg p_1 \vee p_2$: similar.

$\square$

TND as a Logical Relation

Key Ideas of previous proof:

1. state type-indexed family of predicates $\mathrm{TND}\_(-)$
2. for every type τ, prove predicate is preserved under constructors $c\colon \tau_1 \to \ldots \to \tau_n \to \tau$:

$$\text{if } \mathrm{TND}_{\tau_1}(t_1) \wedge \ldots \wedge \mathrm{TND}_{\tau_n}(t_n), \text{ then } \mathrm{TND}_{\tau}(c\ t_1\ \ldots\ t_n)$$

TND as a Logical Relation

Key Ideas of previous proof:

1. state type-indexed family of predicates $\mathrm{TND}\_(-)$
2. for every type τ, prove predicate is preserved under constructors $c\colon \tau_1 \to \ldots \to \tau_n \to \tau$:

$$\text{if } \mathrm{TND}_{\tau_1}(t_1) \wedge \ldots \wedge \mathrm{TND}_{\tau_n}(t_n), \text{ then } \mathrm{TND}_{\tau}(c\ t_1 \ldots\ t_n)$$

In MMT/LF: mimic that with new syntax!

$$\textbf{relation}\ \texttt{TND}\colon\ id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{$$

$$\}$$

TND as a Logical Relation

Key Ideas of previous proof:

1. state type-indexed family of predicates $\mathrm{TND}\_(-)$
2. for every type τ, prove predicate is preserved under constructors $c\colon \tau_1 \to \ldots \to \tau_n \to \tau$:

$$\text{if } \mathrm{TND}_{\tau_1}(t_1) \wedge \ldots \wedge \mathrm{TND}_{\tau_n}(t_n),\ \text{then } \mathrm{TND}_{\tau}(c\ t_1\ \ldots\ t_n)$$

In MMT/LF: mimic that with new syntax!

Recall: we had `atom: type`, `prop: type`

$$\begin{aligned}
&\textbf{relation}\ \mathtt{TND}\colon\ id_{\mathtt{PL}}\colon \mathtt{PL} \to \mathtt{PL} = \{\\
&\qquad \mathtt{atom} := \lambda a\colon \mathtt{atom}.\ \vdash \mathtt{inj}(a) \vee \neg\mathtt{inj}(a)\\
&\qquad \mathtt{prop} := \underbrace{\lambda p\colon \mathtt{prop}.\ \vdash p \vee \neg p}_{\text{encoding of } \mathtt{TND}_{\mathtt{prop}}(-)}\\
&\qquad\qquad \vdots\\
&\}
\end{aligned}$$

TND as a Logical Relation (cont.)

$$\textbf{relation } \texttt{TND}\colon\ id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{$$

$$\quad \texttt{atom} := \lambda a\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \vee \neg\texttt{inj}(a)$$

$$\quad \texttt{prop} := \lambda p\colon \texttt{prop}.\ \vdash p \vee \neg p$$

TND as a Logical Relation (cont.)

Recall: $\texttt{inj}\colon \texttt{atom} \to \texttt{prop}$; encode a proof of $\mathrm{TND}_P(a)$

$$\textbf{relation}\ \texttt{TND}\colon\ id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{$$

$$\texttt{atom} := \lambda a\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \vee \neg\texttt{inj}(a)$$

$$\texttt{prop} := \lambda p\colon \texttt{prop}.\ \vdash p \vee \neg p$$

$$\texttt{inj} := \lambda a\colon \texttt{atom}.\ \lambda a^*\colon\ \vdash \texttt{inj}(a) \vee \neg\texttt{inj}(a).\ a^*$$

TND as a Logical Relation (cont.)

Recall: $\neg \colon \texttt{prop} \to \texttt{prop}$; encode a proof of $\mathrm{TND}_P(p)$ implying $\mathrm{TND}_P(\neg p)$

$$
\begin{aligned}
&\textbf{relation}\ \texttt{TND} \colon\ id_{\texttt{PL}} \colon \texttt{PL} \to \texttt{PL} = \{\\
&\qquad \texttt{atom} := \lambda a \colon \texttt{atom}.\ \vdash \texttt{inj}(a) \vee \neg\texttt{inj}(a)\\
&\qquad \texttt{prop} := \lambda p \colon \texttt{prop}.\ \vdash p \vee \neg p\\
&\qquad \texttt{inj} := \lambda a \colon \texttt{atom}.\ \lambda a^* \colon\ \vdash \texttt{inj}(a) \vee \neg\texttt{inj}(a).\ a^*\\
&\qquad \neg := \lambda p \colon \texttt{prop}.\ \lambda p^* \colon\ \vdash p \vee \neg p.\\
&\qquad\qquad \ldots \langle \text{of type } \vdash \neg p \vee \neg\neg p \rangle
\end{aligned}
$$

TND as a Logical Relation (cont.)

Recall: $\neg\colon \mathtt{prop} \to \mathtt{prop}$; encode a proof of $\mathrm{TND}_P(p)$ implying $\mathrm{TND}_P(\neg p)$

$$
\begin{aligned}
&\textbf{relation } \mathtt{TND}\colon\ id_{\mathtt{PL}}\colon \mathtt{PL} \to \mathtt{PL} = \{\\
&\qquad \mathtt{atom} := \lambda a\colon \mathtt{atom}.\ \vdash \mathtt{inj}(a) \vee \neg\mathtt{inj}(a)\\
&\qquad \mathtt{prop} := \lambda p\colon \mathtt{prop}.\ \vdash p \vee \neg p\\
&\qquad\ \ \mathtt{inj} := \lambda a\colon \mathtt{atom}.\ \lambda a^*\colon\ \vdash \mathtt{inj}(a) \vee \neg\mathtt{inj}(a).\ a^*\\
&\qquad\quad \neg := \lambda p\colon \mathtt{prop}.\ \lambda p^*\colon\ \vdash p \vee \neg p.\\
&\qquad\qquad\qquad \vee_{\mathtt{E}}\ p^*\ (\lambda p_\top.\ \vee_{\mathtt{IR}}\ (\neg_{\mathtt{I}}\ \lambda p_\bot.\ \neg_{\mathtt{E}}\ p_\top\ p_\bot))\\
&\qquad\qquad\qquad\quad (\lambda p_\bot\colon\ \vdash \neg p.\ \vee_{\mathtt{IL}}\ p_\bot) \qquad \langle\text{of type } \vdash \neg p \vee \neg\neg p\rangle
\end{aligned}
$$

TND as a Logical Relation (cont.)

Recall: $\wedge, \vee : \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop}$; encode proofs of $\mathrm{TND}_P(p), \mathrm{TND}_P(q)$ implying $\mathrm{TND}_P(p \wedge q)$ and $\mathrm{TND}_P(p \vee q)$

$$
\begin{aligned}
&\textbf{relation } \mathtt{TND} \colon\ id_{\mathtt{PL}} \colon \mathtt{PL} \to \mathtt{PL} = \{\\
&\quad \mathtt{atom} := \lambda a \colon \mathtt{atom}.\ \vdash \mathtt{inj}(a) \vee \neg \mathtt{inj}(a)\\
&\quad \mathtt{prop} := \lambda p \colon \mathtt{prop}.\ \vdash p \vee \neg p\\
&\\
&\quad \mathtt{inj} := \lambda a \colon \mathtt{atom}.\ \lambda a^* \colon \vdash \mathtt{inj}(a) \vee \neg \mathtt{inj}(a).\ a^*\\
&\quad \neg := \lambda p \colon \mathtt{prop}.\ \lambda p^* \colon \vdash p \vee \neg p.\\
&\qquad\qquad \vee_{\mathtt{E}}\ p^*\ (\lambda p_\top.\ \vee_{\mathtt{IR}}\ (\neg_{\mathtt{I}}\ \lambda p_\bot.\ \neg_{\mathtt{E}}\ p_\top\ p_\bot))\\
&\qquad\qquad\quad (\lambda p_\bot \colon \vdash \neg p.\ \vee_{\mathtt{IL}}\ p_\bot) \qquad \langle\text{of type } \vdash \neg p \vee \neg\neg p\rangle\\
&\quad \wedge := \lambda p \colon \mathtt{prop}.\ \lambda p^* \colon \vdash p \vee \neg p.\\
&\qquad\qquad \lambda q \colon \mathtt{prop}.\ \lambda q^* \colon \vdash q \vee \neg q.\ \ldots \langle\text{of type } \vdash p \wedge q \vee \neg p \wedge q\rangle\\
&\quad \vee := \lambda p \colon \mathtt{prop}.\ \lambda p^* \colon \vdash p \vee \neg p.\\
&\qquad\qquad \lambda q \colon \mathtt{prop}.\ \lambda q^* \colon \vdash q \vee \neg q.\ \ldots \langle\text{of type } \vdash p \vee q \vee \neg p \vee q\rangle\\
&\quad \vdots
\end{aligned}
$$

TND as a Logical Relation (cont.)

Final Result:

$$
\begin{aligned}
&\textbf{relation } \texttt{TND}:\ id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{\\
&\qquad \texttt{atom} := \lambda a\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \lor \neg\texttt{inj}(a)\\
&\qquad \texttt{prop} := \lambda p\colon \texttt{prop}.\ \vdash p \lor \neg p\\
&\qquad \texttt{inj} := \lambda a\colon \texttt{atom}.\ \lambda a^*\colon\ \vdash \texttt{inj}(a) \lor \neg\texttt{inj}(a).\ a^*\\
&\qquad \neg := \lambda p\colon \texttt{prop}.\ \lambda p^*\colon\ \vdash p \lor \neg p.\\
&\qquad\qquad \lor_{\texttt{E}}\ p^*\ (\lambda p_\top.\ \lor_{\texttt{IR}}\ (\neg_{\texttt{I}}\ \lambda p_\perp.\ \neg_{\texttt{E}}\ p_\top\ p_\perp))\\
&\qquad\qquad\quad (\lambda p_\perp\colon\ \vdash \neg p.\ \lor_{\texttt{IL}}\ p_\perp) \qquad \langle\text{of type } \vdash \neg p \lor \neg\neg p\rangle\\
&\qquad \land := \lambda p\colon \texttt{prop}.\ \lambda p^*\colon\ \vdash p \lor \neg p.\\
&\qquad\qquad \lambda q\colon \texttt{prop}.\ \lambda q^*\colon\ \vdash q \lor \neg q.\ \ldots \langle\text{of type } \vdash p \land q \lor \neg p \land q\rangle\\
&\qquad \lor := \lambda p\colon \texttt{prop}.\ \lambda p^*\colon\ \vdash p \lor \neg p.\\
&\qquad\qquad \lambda q\colon \texttt{prop}.\ \lambda q^*\colon\ \vdash q \lor \neg q.\ \ldots \langle\text{of type } \vdash p \lor q \lor \neg p \lor q\rangle\\
&\}
\end{aligned}
$$

Consequences of TND as a Log. Rel.

So far only a piece of syntax:

$$
\begin{array}{l}
\textbf{relation}\ \texttt{TND}\colon\ id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{ \\
\qquad \texttt{atom} := \lambda a\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \vee \neg\texttt{inj}(a) \\
\qquad \texttt{prop} := \lambda p\colon \texttt{prop}.\ \vdash p \vee \neg p \\
\qquad\qquad \vdots \\
\}
\end{array}
$$

Consequences of TND as a Log. Rel.

So far only a piece of syntax:

$$
\begin{array}{l}
\textbf{relation}\ \texttt{TND}\colon\ id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{ \\
\qquad \texttt{atom} := \lambda a\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \vee \neg\texttt{inj}(a) \\
\qquad \texttt{prop} := \underbrace{\lambda p\colon \texttt{prop}.\ \vdash p \vee \neg p}_{=:\texttt{TND}_{\texttt{prop}}} \\
\qquad\qquad \vdots \\
\}
\end{array}
$$

Consequences of $\texttt{TND}$ as a Log. Rel.

So far only a piece of syntax:

$$\begin{aligned}
&\textbf{relation}\ \texttt{TND}\colon\ id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{\\
&\qquad \texttt{atom} := \lambda a\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \lor \neg\texttt{inj}(a)\\
&\qquad \texttt{prop} := \underbrace{\lambda p\colon \texttt{prop}.\ \vdash p \lor \neg p}_{=:\texttt{TND}_{\texttt{prop}}}\\
&\qquad\qquad \vdots\\
&\}
\end{aligned}$$

But meta theory gives: if $r\colon\ id_T\colon T \to T$ is a logical relation (i.e. well-typed), then

Theorem (Basic Lemma)

If $\vdash_T t\colon \tau$, *then* $\vdash_T r(t)\colon r(\tau)\ t$.

Consequences of $\mathtt{TND}$ as a Log. Rel.

So far only a piece of syntax:

$$\begin{aligned}
&\textbf{relation}\ \mathtt{TND}\colon\ id_{\mathtt{PL}}\colon \mathtt{PL} \to \mathtt{PL} = \{\\
&\qquad \mathtt{atom} := \lambda a\colon \mathtt{atom}.\ \vdash \mathtt{inj}(a) \vee \neg\mathtt{inj}(a)\\
&\qquad \mathtt{prop} := \underbrace{\lambda p\colon \mathtt{prop}.\ \vdash p \vee \neg p}_{=:\mathtt{TND}_{\mathtt{prop}}}\\
&\qquad\qquad \vdots\\
&\}
\end{aligned}$$

But meta theory gives: if $r\colon\ id_T\colon T \to T$ is a logical relation (i.e. well-typed), then

Theorem (Basic Lemma)

If $\vdash_T t\colon \tau$, *then* $\vdash_T r(t)\colon r(\tau)\ t$.

Corollary

If $\vdash_{\mathtt{PL}} p\colon \mathtt{prop}$ *is a proposition, then* $\vdash_{\mathtt{PL}} r(p)\colon \mathtt{TND}_{\mathtt{prop}}\ p$, *i.e.* $\vdash_{\mathtt{PL}} p \vee \neg p$ *is provable.*

Logical Relations: A Definition

Definition

Let T be a theory and $(r_c)_{c \in T}$ a family of T-expressions. Define

$$
\begin{aligned}
r(\mathtt{type}) &= \lambda A\colon \mathtt{type}.\ A \to \mathtt{type} \\
r(\Pi x\colon A.\ B) &= \lambda f\colon (\Pi x\colon A.\ B).\ \Pi x\colon A.\ \Pi x^*\colon r(A)\ x.\ r(B)\ (f\ x) \\
r(A\ B) &= r(A)\ B\ r(B) \\
r(\lambda x\colon A.\ B) &= \lambda x\colon A.\ \lambda x^*\colon r(A)\ x.\ r(B) \\
r(c) &= r_c \\
r(x) &= x^*
\end{aligned}
$$

We call r a **unary logical relation on** id_T if $\vdash_T r_c\colon r(\tau)\ c$ for all constants $c\colon \tau$ in T.

- e.g. $\mathtt{prop}\colon \mathtt{type}$, hence $r_{\mathtt{prop}}\colon \mathtt{prop} \to \mathtt{type}$ $\quad r_{\mathtt{prop}} := \lambda p.\ \vdash p \vee \neg p$
- e.g. $\neg\colon \mathtt{prop} \to \mathtt{prop}$, hence
  $r_\neg\colon \Pi p\colon \mathtt{prop}.\ \Pi p^*\colon r(\mathtt{prop})\ p.\ r(\mathtt{prop})\ (\neg p)$ $\quad r_\neg := \langle \text{our proof} \rangle$

Logical Relations: A Definition

Definition

Let T be a theory and $(r_c)_{c\in T}$ a family of T-expressions. Define

$$
\begin{aligned}
r(\mathtt{type}) &= \lambda A\colon \mathtt{type}.\ A \to \mathtt{type}\\
r(\Pi x\colon A.\ B) &= \lambda f\colon (\Pi x\colon A.\ B).\ \Pi x\colon A.\ \Pi x^*\colon r(A)\ x.\ r(B)\ (f\ x)\\
r(A\ B) &= r(A)\ B\ r(B)\\
r(\lambda x\colon A.\ B) &= \lambda x\colon A.\ \lambda x^*\colon r(A)\ x.\ r(B)\\
r(c) &= r_c\\
r(x) &= x^*
\end{aligned}
$$

We call r a **unary logical relation on** id_T if $\vdash_T r_c\colon r(\tau)\ c$ for all constants $c\colon \tau$ in T.

- e.g. $\mathtt{prop}\colon \mathtt{type}$, hence $r_{\mathtt{prop}}\colon \mathtt{prop} \to \mathtt{type}$ $\quad r_{\mathtt{prop}} := \lambda p.\ \vdash p \vee \neg p$
- e.g. $\neg\colon \mathtt{prop} \to \mathtt{prop}$, hence $r_\neg\colon \Pi p\colon \mathtt{prop}.\ \Pi p^*\colon\ \vdash p \vee \neg p.\ \vdash \neg p \vee \neg\neg p$ $\quad r_\neg := \langle$our proof$\rangle$

Logical Relations: A Definition

Definition

Let T be a theory and $(r_c)_{c \in T}$ a family of T-expressions. Define

$$
\begin{aligned}
r(\mathtt{type}) &= \lambda A\colon \mathtt{type}.\ A \to \mathtt{type} \\
r(\Pi x\colon A.\ B) &= \lambda f\colon (\Pi x\colon A.\ B).\ \Pi x\colon A.\ \Pi x^*\colon r(A)\ x.\ r(B)\ (f\ x) \\
r(A\ B) &= r(A)\ B\ r(B) \\
r(\lambda x\colon A.\ B) &= \lambda x\colon A.\ \lambda x^*\colon r(A)\ x.\ r(B) \\
r(c) &= r_c \\
r(x) &= x^*
\end{aligned}
$$

We call r a **unary logical relation on** id_T if $\vdash_T r_c\colon r(\tau)\ c$ for all constants $c\colon \tau$ in T.

Theorem (Basic Lemma)

If $\Gamma \vdash_T t\colon \tau$, *then* $r(\Gamma) \vdash_T r(t)\colon r(\tau)\ t$.

Example 1: Summary

Summary:

- Logical relations often occur in a type theory, and there
  - associate to every type τ a meta proposition $C_\tau(-)$
  - "the basic lemma" then states: if $t\colon \tau$, then $C_\tau(t)$
- In LF, we can
  - map every type to a function giving us the meta proposition

    e.g. $\texttt{prop} := \lambda p.\ \vdash p \vee \neg p$
  - prove for every term constructor $f\colon \tau_1 \to \ldots \to \tau_n \to \tau$ the preservation of $C_\tau(-)$ given $C_{\tau_1}(-)$, ..., $C_{\tau_n}(-)$

    e.g. $\mathrm{TND}_P(p)$ must imply $\mathrm{TND}_P(\neg p)$

    "unary congruences"
- As such, they subsume (some) inductive arguments

Next: Binary Logical Relations

Towards Binary Logical Relations

Key Ideas of Unary Relations: already seen

1. state type-indexed family of predicates $\mathrm{TND}\_(-)$
2. for every type τ, prove predicate is preserved under constructors $c\colon \tau_1 \to \ldots \to \tau_n \to \tau$:

$$\text{if } \mathrm{TND}_{\tau_1}(t_1) \wedge \ldots \wedge \mathrm{TND}_{\tau_n}(t_n), \text{ then } \mathrm{TND}_{\tau}(c\ t_1 \ldots\ t_n)$$

How about $C_P(p, p') :=\ \vdash p \Leftrightarrow p'$?

Towards Binary Logical Relations

Key Ideas of Unary Relations: already seen

1. state type-indexed family of predicates $\mathrm{TND}\_(-)$
2. for every type τ, prove predicate is preserved under constructors $c\colon \tau_1 \to \ldots \to \tau_n \to \tau$:

$$\text{if } \mathrm{TND}_{\tau_1}(t_1) \wedge \ldots \wedge \mathrm{TND}_{\tau_n}(t_n), \text{ then } \mathrm{TND}_{\tau}(c\ t_1\ \ldots\ t_n)$$

How about $C_P(p, p') := \vdash p \Leftrightarrow p'$?

1. forms a type-indexed family of binary predicates

   together with $C_A(a, b) := \vdash \mathrm{inj}(a) \Leftrightarrow \mathrm{inj}(b)$

2. e.g. is preserved under $\wedge$:
   - given $\vdash p \Leftrightarrow p'$ and $\vdash q \Leftrightarrow q'$
   - we can derive $\vdash (p \wedge q) \Leftrightarrow (p' \wedge q')$

Example 2: $\Leftrightarrow$ congruence in Prop. Logic

Example (PL Extended)

We extend PL's grammar by

$$\begin{array}{lll l} P & ::= & A \mid \neg P \mid P \wedge P \mid P \vee P \mid & \text{propositions} \\ & & P \Leftrightarrow P & \\ A & ::= & \langle\text{unspecified}\rangle & \text{atoms} \end{array}$$

and its proof calculus accordingly.

Theorem (Leftrightarrow (LRA) Congruence)

$\Leftrightarrow$ is a congruence on P. That is, if $\vdash p \Leftrightarrow p'$ and $\vdash q \Leftrightarrow q'$, then

- $\vdash \neg p \Leftrightarrow \neg p'$
- $\vdash p \wedge q \Leftrightarrow p' \wedge q'$
- $\vdash p \vee q \Leftrightarrow p' \vee q'$
- $\vdash (p \Leftrightarrow q) \Leftrightarrow (p' \Leftrightarrow q')$,

and, trivially, if $\vdash \mathrm{inj}(a) \Leftrightarrow \mathrm{inj}(a')$, then $\vdash \mathrm{inj}(a) \Leftrightarrow \mathrm{inj}(a')$.

LRA as a Logical Relation

relation $\texttt{LRA:}\ id_{\mathrm{PL}} \times id_{\mathrm{PL}} \colon \mathtt{PL} \to \mathtt{PL} = \{$

LRA as a Logical Relation

Recall: `atom`: `type`, map to the desired binary relation on atoms

$\mathbf{relation}$ $\mathtt{LRA}\colon\ id_{\mathtt{PL}} \times id_{\mathtt{PL}}\colon \mathtt{PL} \to \mathtt{PL} = \{$

$\quad \mathtt{atom} := \lambda a\colon \mathtt{atom}.\ \lambda a'\colon \mathtt{atom}.\ \vdash \mathtt{inj}(a) \Leftrightarrow \mathtt{inj}(a')$

LRA as a Logical Relation

Recall: `prop`: type, map to the desired binary relation on propositions

$$\textbf{relation}\ \texttt{LRA}\colon\ \mathit{id}_{\texttt{PL}} \times \mathit{id}_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{$$

$$\texttt{atom} := \lambda a\colon \texttt{atom}.\ \lambda a'\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \Leftrightarrow \texttt{inj}(a')$$

$$\texttt{prop} := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \vdash p \Leftrightarrow p'$$

LRA as a Logical Relation

Recall: $\texttt{inj}\colon \texttt{atom} \to \texttt{prop}$, map to appropriate proof

$$\textbf{relation}\ \texttt{LRA}\colon\ id_{\texttt{PL}} \times id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{$$

$$\texttt{atom} := \lambda a\colon \texttt{atom}.\ \lambda a'\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \Leftrightarrow \texttt{inj}(a')$$

$$\texttt{prop} := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \vdash p \Leftrightarrow p'$$

$$\texttt{inj} := \lambda a\colon \texttt{atom}.\ \lambda a'\colon \texttt{atom}.\ \lambda a^*\colon\ \vdash \texttt{inj}(a) \Leftrightarrow \texttt{inj}(a').\ \ a^*$$

LRA as a Logical Relation

Recall: $\neg$: prop $\to$ prop, map to appropriate proof

$$\textbf{relation}\ \mathtt{LRA}\colon\ id_{\mathtt{PL}} \times id_{\mathtt{PL}} \colon \mathtt{PL} \to \mathtt{PL} = \{$$

$$\mathtt{atom} := \lambda a\colon \mathtt{atom}.\ \lambda a'\colon \mathtt{atom}.\ \vdash \mathtt{inj}(a) \Leftrightarrow \mathtt{inj}(a')$$

$$\mathtt{prop} := \lambda p\colon \mathtt{prop}.\ \lambda p'\colon \mathtt{prop}.\ \vdash p \Leftrightarrow p'$$

$$\mathtt{inj} := \lambda a\colon \mathtt{atom}.\ \lambda a'\colon \mathtt{atom}.\ \lambda a^{*}\colon\ \vdash \mathtt{inj}(a) \Leftrightarrow \mathtt{inj}(a').\ \ a^{*}$$

$$\neg := \lambda p\colon \mathtt{prop}.\ \lambda p'\colon \mathtt{prop}.\ \lambda p^{*}\colon\ \vdash p \Leftrightarrow p'.$$

$$\ldots\ \langle \text{of type}\ \vdash \neg p \Leftrightarrow \neg p' \rangle$$

LRA as a Logical Relation

Recall: $\wedge, \vee \colon \texttt{prop} \to \texttt{prop} \to \texttt{prop}$, map to appropriate proofs

$$\textbf{relation}\ \texttt{LRA}\colon\ id_{\texttt{PL}} \times id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{$$

$$\texttt{atom} := \lambda a\colon \texttt{atom}.\ \lambda a'\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \Leftrightarrow \texttt{inj}(a')$$

$$\texttt{prop} := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \vdash p \Leftrightarrow p'$$

$$\texttt{inj} := \lambda a\colon \texttt{atom}.\ \lambda a'\colon \texttt{atom}.\ \lambda a^*\colon \vdash \texttt{inj}(a) \Leftrightarrow \texttt{inj}(a').\ \ a^*$$

$$\neg := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \lambda p^*\colon \vdash p \Leftrightarrow p'.$$

$$\ldots \langle \text{of type } \vdash \neg p \Leftrightarrow \neg p' \rangle$$

$$\wedge := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \lambda p^*\colon \vdash p \Leftrightarrow p'. \qquad \text{// first arg of } \wedge$$

$$\lambda q\colon \texttt{prop}.\ \lambda q'\colon \texttt{prop}.\ \lambda q^*\colon \vdash q \Leftrightarrow q'. \qquad \text{// second arg of } \wedge$$

$$\ldots \langle \text{of type } \vdash p \wedge q \Leftrightarrow p' \wedge q' \rangle$$

LRA as a Logical Relation

Recall: $\wedge, \vee\colon \texttt{prop} \to \texttt{prop} \to \texttt{prop}$, map to

$$\textbf{relation}\ \texttt{LRA}\colon\ id_{\texttt{PL}} \times id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{$$

$$\texttt{atom} := \lambda a\colon \texttt{atom}.\ \lambda a'\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \Leftrightarrow \texttt{inj}(a')$$

$$\texttt{prop} := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \vdash p \Leftrightarrow p'$$

$$\texttt{inj} := \lambda a\colon \texttt{atom}.\ \lambda a'\colon \texttt{atom}.\ \lambda a^*\colon\ \vdash \texttt{inj}(a) \Leftrightarrow \texttt{inj}(a').\ \ a^*$$

$$\neg := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \lambda p^*\colon\ \vdash p \Leftrightarrow p'.$$

... ⟨of type $\vdash \neg p \Leftrightarrow \neg p'$⟩

$$\wedge := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \lambda p^*\colon\ \vdash p \Leftrightarrow p'.$$ // first arg of $\wedge$

$$\lambda q\colon \texttt{prop}.\ \lambda q'\colon \texttt{prop}.\ \lambda q^*\colon\ \vdash q \Leftrightarrow q'.$$ // second arg of $\wedge$

... ⟨of type $\vdash p \wedge q \Leftrightarrow p' \wedge q'$⟩

$$\vee := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \lambda p^*\colon\ \vdash p \Leftrightarrow p'.$$ // first arg of $\vee$

$$\lambda q\colon \texttt{prop}.\ \lambda q'\colon \texttt{prop}.\ \lambda q^*\colon\ \vdash q \Leftrightarrow q'.$$ // second arg of $\vee$

... ⟨of type $\vdash p \vee q \Leftrightarrow p' \vee q'$⟩

LRA as a Logical Relation

$$
\begin{aligned}
&\textbf{relation}\ \texttt{LRA}\colon\ id_{\texttt{PL}} \times id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL} = \{\\
&\quad \texttt{atom} := \lambda a\colon \texttt{atom}.\ \lambda a'\colon \texttt{atom}.\ \vdash \texttt{inj}(a) \Leftrightarrow \texttt{inj}(a')\\
&\quad \texttt{prop} := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \vdash p \Leftrightarrow p'\\
&\quad\ \ \texttt{inj} := \lambda a\colon \texttt{atom}.\ \lambda a'\colon \texttt{atom}.\ \lambda a^*\colon\ \vdash \texttt{inj}(a) \Leftrightarrow \texttt{inj}(a').\ \ a^*\\
&\quad\quad \neg := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \lambda p^*\colon\ \vdash p \Leftrightarrow p'.\\
&\quad\quad\quad\quad \ldots \langle \text{of type } \vdash \neg p \Leftrightarrow \neg p' \rangle\\
&\quad\quad \wedge := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \lambda p^*\colon\ \vdash p \Leftrightarrow p'. \quad\quad \text{// first arg of } \wedge\\
&\quad\quad\quad\quad \lambda q\colon \texttt{prop}.\ \lambda q'\colon \texttt{prop}.\ \lambda q^*\colon\ \vdash q \Leftrightarrow q'. \quad\quad \text{// second arg of } \wedge\\
&\quad\quad\quad\quad\quad \ldots \langle \text{of type } \vdash p \wedge q \Leftrightarrow p' \wedge q' \rangle\\
&\quad\quad \vee := \lambda p\colon \texttt{prop}.\ \lambda p'\colon \texttt{prop}.\ \lambda p^*\colon\ \vdash p \Leftrightarrow p'. \quad\quad \text{// first arg of } \vee\\
&\quad\quad\quad\quad \lambda q\colon \texttt{prop}.\ \lambda q'\colon \texttt{prop}.\ \lambda q^*\colon\ \vdash q \Leftrightarrow q'. \quad\quad \text{// second arg of } \vee\\
&\quad\quad\quad\quad\quad \ldots \langle \text{of type } \vdash p \vee q \Leftrightarrow p' \vee q' \rangle\\
&\}
\end{aligned}
$$

Logical Relations: A Second Definition

Definition

Let T be a theory and $(r_c)_{c \in T}$ a family of T-expressions. Define

$$
\begin{aligned}
r(\mathtt{type}) &= \lambda A\colon \mathtt{type}.\ \lambda A'\colon \mathtt{type}.\ A \to A' \to \mathtt{type} \\
r(\Pi a\colon A.\ B) &= \lambda f\colon (\Pi a\colon A.\ B).\ \lambda f'\colon (\Pi a'\colon A'.\ B'). \\
&\qquad\quad \Pi x\colon A.\ \Pi x'\colon A'.\ \Pi x^*\colon r(A)\ x\ x'. \\
&\qquad\qquad r(B)\ (f\ x)\ (f'\ x') \\
r(A\ B) &= r(A)\ B\ B'\ r(B) \\
r(\lambda x\colon A.\ B) &= \lambda x\colon A.\ \lambda x'\colon A'.\ \lambda x^*\colon r(A)\ x\ x'.\ r(B) \\
r(c) &= r_c \\
r(x) &= x^*
\end{aligned}
$$

where A' denotes systematic priming. We call r a **binary logical relation on** $id_T \times id_T$ if $\vdash_T r_c\colon r(\tau)\ c\ c$ for all constants $c\colon \tau$ in T.

- $\mathtt{prop}\colon \mathtt{type}$, hence $r_{\mathtt{prop}}\colon \mathtt{prop} \to \mathtt{prop} \to \mathtt{type}$

$r_{\mathtt{prop}} := \lambda p.\ \lambda p'.\ \vdash p \Leftrightarrow p'$

Logical Relations: A Second Definition

Definition

Let T be a theory and $(r_c)_{c \in T}$ a family of T-expressions. Define

$$
\begin{aligned}
r(\mathtt{type}) &= \lambda A\colon \mathtt{type}.\ \lambda A'\colon \mathtt{type}.\ A \to A' \to \mathtt{type} \\
r(\Pi a\colon A.\ B) &= \lambda f\colon (\Pi a\colon A.\ B).\ \lambda f'\colon (\Pi a'\colon A'.\ B'). \\
&\qquad \Pi x\colon A.\ \Pi x'\colon A'.\ \Pi x^*\colon r(A)\ x\ x'. \\
&\qquad\quad r(B)\ (f\ x)\ (f'\ x') \\
r(A\ B) &= r(A)\ B\ B'\ r(B) \\
r(\lambda x\colon A.\ B) &= \lambda x\colon A.\ \lambda x'\colon A'.\ \lambda x^*\colon r(A)\ x\ x'.\ r(B) \\
r(c) &= r_c \\
r(x) &= x^*
\end{aligned}
$$

where A' denotes systematic priming. We call r a **binary logical relation on** $id_T \times id_T$ if $\vdash_T r_c\colon r(\tau)\ c\ c$ for all constants $c\colon \tau$ in T.

- $\neg\colon \mathtt{prop} \to \mathtt{prop}$, hence
  $r_\neg\colon \Pi p\, p'\colon \mathtt{prop}.\ \Pi p^*\colon\ \vdash p \Leftrightarrow p'.\ \vdash \neg p \Leftrightarrow \neg p'$

Logical Relations: A Second Definition

Definition

Let T be a theory and $(r_c)_{c \in T}$ a family of T-expressions. Define

$$
\begin{aligned}
r(\mathtt{type}) &= \lambda A\colon \mathtt{type}.\ \lambda A'\colon \mathtt{type}.\ A \to A' \to \mathtt{type} \\
r(\Pi a\colon A.\ B) &= \lambda f\colon (\Pi a\colon A.\ B).\ \lambda f'\colon (\Pi a'\colon A'.\ B'). \\
&\qquad \Pi x\colon A.\ \Pi x'\colon A'.\ \Pi x^*\colon r(A)\ x\ x'. \\
&\qquad\quad r(B)\ (f\ x)\ (f'\ x') \\
r(A\ B) &= r(A)\ B\ B'\ r(B) \\
r(\lambda x\colon A.\ B) &= \lambda x\colon A.\ \lambda x'\colon A'.\ \lambda x^*\colon r(A)\ x\ x'.\ r(B) \\
r(c) &= r_c \\
r(x) &= x^*
\end{aligned}
$$

where A' denotes systematic priming. We call r a **binary logical relation on** $id_T \times id_T$ if $\vdash_T r_c\colon r(\tau)\ c\ c$ for all constants $c\colon \tau$ in T.

Theorem (Basic Lemma). If $\vdash_T t\colon \tau$, then $\vdash_T r(t)\colon r(\tau)\ t\ t'$.

Example 2: Summary

Summary: we have n-ary logical relations of the form

$$\textbf{relation}\ r\colon\ id_T \times \cdots \times id_T\colon T \to T = \{\ldots\}$$

- they associate to every type τ an n-ary relation $C_\tau(-, \ldots, -)$
- their well-typedness expresses that these relations are congruences
- "the basic lemma" states: if $t\colon \tau$, then $C_\tau(t, \ldots, t)$

reflexivity? TODO

Next: Logical Relations along a non-trivial morphism

Logical Relations of a Single Morphism

theory PL

$\mathtt{prop} \colon \mathtt{type}$	$\lambda p \colon \mathtt{prop}.\ \vdash p \vee \neg p$
$\mathtt{atom} \colon \mathtt{type}$	$\lambda a \colon \mathtt{atom}.\ \vdash \mathtt{inj}(a) \vee \neg\mathtt{inj}(a)$
$\mathtt{inj} \colon \mathtt{atom} \to \mathtt{prop}$	for $a \colon \mathtt{atom}$: $\mathtt{inj}(a)$
$\wedge \colon \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop}$	for $p\, q \colon \mathtt{prop}$: $p \wedge q$

$$\textbf{theory}\ \mathtt{PL} \xrightarrow{id_{\mathtt{PL}}} \textbf{theory}\ \mathtt{PL}$$

$\mathtt{prop} \colon \mathtt{type}$	$\lambda p \colon id_{\mathtt{PL}}(\mathtt{prop}).\ \vdash p \vee \neg p$
$\mathtt{atom} \colon \mathtt{type}$	$\lambda a \colon id_{\mathtt{PL}}(\mathtt{atom}).\ \vdash \mathtt{inj}(a) \vee \neg\mathtt{inj}(a)$
$\mathtt{inj} \colon \mathtt{atom} \to \mathtt{prop}$	for $a \colon id_{\mathtt{PL}}(\mathtt{atom})$: $id_{\mathtt{PL}}(\mathtt{inj})(a)$
$\wedge \colon \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop}$	for $p\, q \colon id_{\mathtt{PL}}(\mathtt{prop})$: $p\, id_{\mathtt{PL}}(\wedge) q$
"structure space"	"assertion/proof space"

Logical Relations of a Single Morphism

theory $\mathtt{PL}$

$\mathtt{prop} \colon \mathtt{type}$	$\lambda p \colon \mathtt{prop}.\ \vdash p \vee \neg p$
$\mathtt{atom} \colon \mathtt{type}$	$\lambda a \colon \mathtt{atom}.\ \vdash \mathtt{inj}(a) \vee \neg\mathtt{inj}(a)$
$\mathtt{inj} \colon \mathtt{atom} \to \mathtt{prop}$	for $a \colon \mathtt{atom}$: $\mathtt{inj}(a)$
$\wedge \colon \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop}$	for $p\ q \colon \mathtt{prop}$: $p \wedge q$

$$\textbf{theory}\ \mathtt{PL} \xrightarrow{\quad v \quad} \textbf{theory}\ T$$

$\mathtt{prop} \colon \mathtt{type}$	$\lambda p \colon v(\mathtt{prop}).\ \vdash p \vee \neg p$
$\mathtt{atom} \colon \mathtt{type}$	$\lambda a \colon v(\mathtt{atom}).\ \vdash \mathtt{inj}(a) \vee \neg\mathtt{inj}(a)$
$\mathtt{inj} \colon \mathtt{atom} \to \mathtt{prop}$	for $a \colon v(\mathtt{atom})$: $v(\mathtt{inj})(a)$
$\wedge \colon \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop}$	for $p\ q \colon v(\mathtt{prop})$: $p v(\wedge) q$
"structure space"	"assertion/proof space"

Logical Relations of a Single Morphism

theory $\mathtt{PL}$

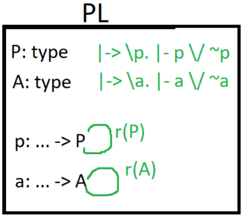
$\mathtt{prop} : \mathtt{type}$	$\lambda p : \mathtt{prop}.\ \vdash p \vee \neg p$
$\mathtt{atom} : \mathtt{type}$	$\lambda a : \mathtt{atom}.\ \vdash \mathtt{inj}(a) \vee \neg\mathtt{inj}(a)$
$\mathtt{inj} : \mathtt{atom} \to \mathtt{prop}$	for $a : \mathtt{atom}$: $\mathtt{inj}(a)$
$\wedge : \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop}$	for $p\,q : \mathtt{prop}$: $p \wedge q$

$$\textbf{theory } \mathtt{PL} \overset{v}{\underset{w}{\rightrightarrows}} \textbf{theory } T$$

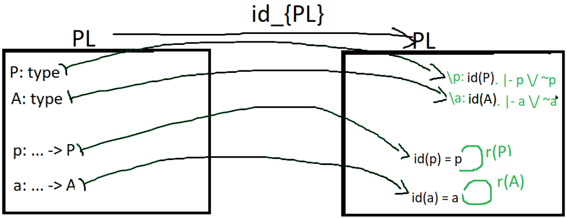
$\mathtt{prop} : \mathtt{type}$	$\lambda p : v(\mathtt{prop}).\ \lambda p' : w(\mathtt{prop}).\ \ldots$
$\mathtt{atom} : \mathtt{type}$	$\lambda a : v(\mathtt{atom}).\ \lambda a' : w(\mathtt{atom}).\ \ldots$
$\mathtt{inj} : \mathtt{atom} \to \mathtt{prop}$	for $a : v(\mathtt{atom})$, $a' : w(\mathtt{atom})$: $v(\mathtt{inj})(a)$
$\wedge : \mathtt{prop} \to \mathtt{prop} \to \mathtt{prop}$	for $p\,q : v(\mathtt{prop})$: $p v(\wedge) q$
"structure space"	"assertion/proof space"

Logical Relations of a Single Morphism

So far: **relation** $\mathtt{TND}$: $id_{\mathrm{PL}} \colon \mathrm{PL} \to \mathrm{PL} = \{\ldots\}$:



What does id_{PL} stand for?



Towards n-ary Logical Relations

Intuition:

- have a single identity morphism id_T

  idea: generalize to $\mu : S \to T$

- relation asserts a unary predicate $C_\tau(t) = C_\tau(id_T(t))$ for every $t \colon \tau$ of the domain

  idea: generalize to n-ary predicates with n morphisms $S \to T$

- domain is “structure” space
- codomain is “assertion and proof” space

  idea: need not be the same

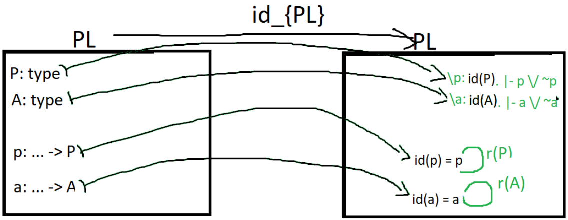
n-ary Logical Relations

- let $\mu_1, \ldots, \mu_n \colon S \to T$ be morphisms

  on common (co)domain theories

- introduce construct

$$\textbf{relation}\ r\colon\ \mu_1 \times \cdots \times \mu_n\colon S \to T = \{c_1 := e_1, \ldots\}$$



Example 3: Church To Curry

Example (Intrinsic vs. Extrinsic Typing)

A type theory is

- intrinsically typed if its terms carry their types.

  pair constructor takes type arguments *and* typed terms

  $(a, b)^{A \times B}$ is a term

- extrinsically typed if its terms are untyped, but "external" typing judgements exist.

  pair constructor takes untyped terms

  (a, b) is a term

  $(a, b) :: A \times B$ a typing judgement

also "Church vs. Curry"

Church and Curry Products

$$\textbf{theory}\ \ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm} \colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a,b) :: (A \end{array}\right.$$

$$\textbf{view}\ \ \texttt{TypeEras} \colon \ \texttt{ChurchProd} \to \texttt{CurryProd} =$$

$$\begin{array}{ll} \quad \texttt{tp} & := \texttt{tp} \\ \quad \texttt{tm} & := \lambda\_.\ \texttt{tm} \\ \quad \_ \times \_ & := \_ \times \_ \\ \quad (\_,\_)^{\_\times\_} & := \lambda\_.\ \lambda\_.\ (\_,\_) \end{array}$$

$\}$

e.g. $\texttt{TypeEras}((a,b)^{A\times B}) = (a,b)$

type information lost!

Capturing Type Preservation

Desired property: to be captured *within* the formalization

Theorem (Type Preservation)

If $\vdash_{\texttt{ChurchProd}} t \colon \texttt{tm}\, A$, *then*

$$\vdash_{\texttt{CurryProd}} \texttt{TypeEras}(t) :: A.$$

Capturing Type Preservation

Desired property: to be captured *within* the formalization

Theorem (Type Preservation)

If $\vdash_{\texttt{ChurchProd}} t \colon \texttt{tm}\, A$, *then*

$$\vdash_{\texttt{CurryProd}} \texttt{TypeEras}(t) :: \texttt{TypeEras}(A).$$

$\texttt{TypeEras}$ is identity on types $A \colon \texttt{tp}$

Capturing Type Preservation

Desired property: to be captured *within* the formalization

Theorem (Type Preservation)

If $\vdash_{\texttt{ChurchProd}} t \colon \texttt{tm}\, A$, *there is* wit *st.*

$$\vdash_{\texttt{CurryProd}} \mathrm{wit} \colon \texttt{TypeEras}(t) :: \texttt{TypeEras}(A).$$

Capturing Type Preservation

Desired property: to be captured *within* the formalization

Theorem (Type Preservation)

If $\vdash_{\texttt{ChurchProd}} t : \texttt{tm}\, A$, *there is* wit *st.*

$$\vdash_{\texttt{CurryProd}} \mathrm{wit} : \texttt{TypeEras}(t) :: \texttt{TypeEras}(A).$$

Capturing Type Preservation

Desired property: to be captured *within* the formalization

Theorem (Type Preservation)

If $\vdash_{\texttt{ChurchProd}} t \colon \texttt{tm}\, A$, *there is* wit *st.*

$$\vdash_{\texttt{CurryProd}} \mathrm{wit} \colon \texttt{TypeEras}(t) :: \texttt{TypeEras}(A).$$

Resembles Basic Lemma: let $r\colon\ \mu\colon S \to T$ be a logical relation over $\mu\colon S \to T$, then

Theorem (Basic Lemma)

If $\vdash_S t\colon \tau$, *then* $\vdash_T r(t) \colon r(\tau)\ \mu(t)$.

$\Rightarrow$ create relation $\texttt{TypePres}\colon\ \texttt{TypeEras}\colon \texttt{ChurchProd} \to \texttt{CurryProd}$ mapping $\texttt{tm}\, -$ to $\_ :: \_$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{l} \texttt{tp}, \texttt{tm} \colon \texttt{type} \\ \qquad \begin{array}{ll} \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ (\_,\_)^{T,\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \qquad a :: A \to b :: B \to (a, b) :: (A \end{array} \end{array}\right.$$

$$\begin{array}{llll} \textbf{relation} & \texttt{TypePres} \colon\ \texttt{TypeEras} \colon\ \texttt{ChurchProd} & \to & \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{TypeEras}(tp).\ \texttt{Unit} & & \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \begin{cases} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{cases}$$

$$\textbf{theory}\ \texttt{CurryProd} = \begin{cases} \texttt{tp}, \texttt{tm} \colon \texttt{type} \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a,b) :: (A \end{cases}$$

$$\textbf{relation}\ \ \texttt{TypePres} \colon\ \texttt{TypeEras} \colon\ \texttt{ChurchProd} \to \texttt{CurryProd}$$

$$\texttt{tp} \quad := \lambda A \colon \texttt{tp}.\ \texttt{Unit}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{ \begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array} \right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{ \begin{array}{lll} \texttt{tp}, \texttt{tm} \colon \texttt{type} & & \\ & \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ & \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ & (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ & (\_,\_)^{T,\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & & \quad a :: A \to b :: B \to (a, b) :: (A \end{array} \right.$$

$$\begin{array}{lllll} \textbf{relation} & \texttt{TypePres} \colon & \texttt{TypeEras} \colon \texttt{ChurchProd} & \to & \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} & & & \\ \quad \texttt{tm} & := \lambda A \colon \texttt{TypeEras}(tp). & & & \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{l} \texttt{tp}, \texttt{tm} \colon \texttt{type} \\ \quad \begin{array}{ll} \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ (\_,\_)^{T,\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \qquad a :: A \to b :: B \to (a, b) :: (A \end{array} \end{array}\right.$$

$$\begin{array}{lll} \textbf{relation} & \texttt{TypePres} \colon\ \texttt{TypeEras} \colon\ \texttt{ChurchProd} \to \texttt{CurryProd} & = \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} & \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}. & \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm} \colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a,b) :: (A \end{array}\right.$$

$$\begin{array}{lll} \textbf{relation} & \texttt{TypePres} \colon \ \texttt{TypeEras} \colon \ \texttt{ChurchProd} & \to \quad \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} & \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{TypeEras}(\texttt{tm}\, A). & \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \begin{cases} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_, \_)^{\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{cases}$$

$$\textbf{theory}\ \texttt{CurryProd} = \begin{cases} \texttt{tp}, \texttt{tm} \colon \texttt{type} \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_, \_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_, \_)^{T, \_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a, b) :: (A \end{cases}$$

$$\begin{array}{llllll} \textbf{relation} & \texttt{TypePres} \colon & \texttt{TypeEras} \colon & \texttt{ChurchProd} & \to & \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}. \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm} \colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,\_\times\_} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \qquad a :: A \to b :: B \to (a,b) :: (A \end{array}\right.$$

$$\begin{array}{lllll} \textbf{relation} & \texttt{TypePres} \colon & \texttt{TypeEras} \colon\ \texttt{ChurchProd} & \to & \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} & & & \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ \ t :: A & & & \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \mathtt{ChurchProd} = \left\{\begin{array}{ll} \mathtt{tp} & : \mathtt{type} \\ \mathtt{tm} & : \mathtt{tp} \to \mathtt{type} \\ \_ \times \_ & : \mathtt{tp} \to \mathtt{tp} \to \mathtt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \mathtt{tp}.\ \mathtt{tm}\, A \to \mathtt{tm}\, B \to \mathtt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \mathtt{CurryProd} = \left\{\begin{array}{l} \mathtt{tp}, \mathtt{tm} \colon \mathtt{type} \\ \begin{array}{ll} \_ :: \_ & : \mathtt{tm} \to \mathtt{tp} \to \mathtt{type} \\ \_ \times \_ & : \mathtt{tp} \to \mathtt{tp} \to \mathtt{tp} \\ (\_,\_) & : \mathtt{tm} \to \mathtt{tm} \to \mathtt{tm} \\ (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \mathtt{tp}.\ \Pi a\, b \colon \mathtt{tm}. \\ & \quad a :: A \to b :: B \to (a,b) :: (A \end{array} \end{array}\right.$$

$$\begin{array}{lll} \textbf{relation} & \mathtt{TypePres} \colon\ \mathtt{TypeEras} \colon\ \mathtt{ChurchProd} \to \mathtt{CurryProd} & = \\ \quad \mathtt{tp} & := \lambda A \colon \mathtt{tp}.\ \mathtt{Unit} & \\ \quad \mathtt{tm} & := \lambda A \colon \mathtt{tp}.\ \lambda t \colon \mathtt{tm}.\ t :: A & \\ \quad \_ \times \_ & := \ldots \mathtt{unit} & \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm} \colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a, b) :: (A \end{array}\right.$$

$$\begin{array}{ll} \textbf{relation} & \texttt{TypePres} \colon\ \texttt{TypeEras} \colon\ \texttt{ChurchProd} \to \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ \ t :: A \\ \quad \_ \times \_ & := \ldots \texttt{unit} \\ \quad (\_,\_)^{-\times-} & := \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \begin{Bmatrix} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{Bmatrix}$$

$$\textbf{theory}\ \texttt{CurryProd} = \begin{cases} \texttt{tp}, \texttt{tm} \colon \texttt{type} \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \qquad a :: A \to b :: B \to (a,b) :: (A \end{cases}$$

$$\begin{array}{ll} \textbf{relation} & \texttt{TypePres} \colon\ \texttt{TypeEras} \colon\ \texttt{ChurchProd} \to \texttt{CurryProd} = \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ \ t :: A \\ \quad \_ \times \_ & := \ldots \texttt{unit} \\ \quad (\_,\_)^{-\times-} & := \lambda A \colon \texttt{TypeEras}(\texttt{tp}).\ \lambda A^{*}. \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B\colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm}\colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,-\times-} & : \Pi A\, B\colon \texttt{tp}.\ \Pi a\, b\colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a,b) :: (A \end{array}\right.$$

$$\begin{array}{ll} \textbf{relation} & \texttt{TypePres}\colon\ \texttt{TypeEras}\colon\ \texttt{ChurchProd} \to \texttt{CurryProd} = \\ \quad \texttt{tp} & := \lambda A\colon \texttt{tp}.\ \texttt{Unit} \\ \quad \texttt{tm} & := \lambda A\colon \texttt{tp}.\ \lambda t\colon \texttt{tm}.\ \ t :: A \\ \quad \_ \times \_ & := \ldots \texttt{unit} \\ \quad (\_,\_)^{-\times-} & := \lambda A\colon \texttt{tp}.\ \lambda A^*. \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm} \colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \qquad a :: A \to b :: B \to (a,b) :: (A \end{array}\right.$$

$$\begin{array}{ll} \textbf{relation} & \texttt{TypePres} \colon\ \texttt{TypeEras} \colon\ \texttt{ChurchProd} \to \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ \ t :: A \\ \quad \_ \times \_ & := \ldots \texttt{unit} \\ \quad (\_,\_)^{-\times-} & := \lambda A \colon \texttt{tp}.\ \lambda A^*.\ \lambda B \colon \texttt{TypeEras}(\texttt{tp}).\ \lambda B^*. \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm} \colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a,b) :: (A \end{array}\right.$$

$$\textbf{relation}\quad \texttt{TypePres} \colon\ \texttt{TypeEras} \colon\ \texttt{ChurchProd} \to \texttt{CurryProd}$$

$$\begin{array}{ll} \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} \\ \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ \ t :: A \\ \_ \times \_ & := \ldots \texttt{unit} \\ (\_,\_)^{-\times-} & := \lambda A \colon \texttt{tp}.\ \lambda A^*.\ \lambda B \colon \texttt{tp}.\ \lambda B^*. \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll}\texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll}\texttt{tp}, \texttt{tm} \colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a,b) :: (A \end{array}\right.$$

$$\begin{array}{ll} \textbf{relation} & \texttt{TypePres} \colon \ \texttt{TypeEras} \colon \ \texttt{ChurchProd} \to \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ \ t :: A \\ \quad \_ \times \_ & := \ldots \texttt{unit} \\ \quad (\_,\_)^{-\times-} & := \lambda A \colon \texttt{tp}.\ \lambda A^*.\ \lambda B \colon \texttt{tp}.\ \lambda B^*. \\ & \quad \lambda a \colon \texttt{TypeEras}(\texttt{tm}\, A). \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm} \colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a,b) :: (A \end{array}\right.$$

$$\begin{array}{lll} \textbf{relation} & \texttt{TypePres} \colon\ \texttt{TypeEras} \colon\ \texttt{ChurchProd} & \to \quad \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} & \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ \ t :: A & \\ \quad \_ \times \_ & := \ldots \texttt{unit} & \\ \quad (\_,\_)^{-\times-} & := \lambda A \colon \texttt{tp}.\ \lambda A^*.\ \lambda B \colon \texttt{tp}.\ \lambda B^*. & \\ & \quad \lambda a \colon \texttt{tm}. & \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{l} \texttt{tp}, \texttt{tm} \colon \texttt{type} \\ \quad \begin{array}{ll} \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a, b) :: (A \end{array} \end{array}\right.$$

$$\begin{array}{lllll} \textbf{relation} & \texttt{TypePres} \colon & \texttt{TypeEras} \colon\ \texttt{ChurchProd} & \to & \texttt{CurryProd} \end{array}$$

$$\begin{array}{ll} \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} \\ \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ \ t :: A \\ \_ \times \_ & := \ldots \texttt{unit} \\ (\_,\_)^{-\times-} & := \lambda A \colon \texttt{tp}.\ \lambda A^*.\ \lambda B \colon \texttt{tp}.\ \lambda B^*. \\ & \quad \lambda a \colon \texttt{tm}.\ \lambda a^* \colon a :: A. \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{l} \texttt{tp}, \texttt{tm} \colon \texttt{type} \\ \quad \begin{array}{ll} \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_,\_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ (\_,\_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \qquad a :: A \to b :: B \to (a, b) :: (A \end{array} \end{array}\right.$$

$$\begin{array}{ll} \textbf{relation} & \texttt{TypePres} \colon\ \texttt{TypeEras} \colon\ \texttt{ChurchProd} \to \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ t :: A \\ \quad \_ \times \_ & := \ldots \texttt{unit} \\ \quad (\_,\_)^{-\times-} & := \lambda A \colon \texttt{tp}.\ \lambda A^{*}.\ \lambda B \colon \texttt{tp}.\ \lambda B^{*}. \\ & \qquad \lambda a \colon \texttt{tm}.\ \lambda a^{*} \colon a :: A. \\ & \qquad\quad \lambda b \colon \texttt{TypeEras}(\texttt{tm}\, B).\ \lambda b^{*} \colon b :: B. \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_, \_)^{-\times-} & : \Pi A\, B\colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm}\colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_, \_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_, \_)^{T,-\times-} & : \Pi A\, B\colon \texttt{tp}.\ \Pi a\, b\colon \texttt{tm}. \\ & \quad a :: A \to b :: B \to (a, b) :: (A \end{array}\right.$$

$$\begin{array}{ll} \textbf{relation} & \texttt{TypePres}\colon\ \texttt{TypeEras}\colon\ \texttt{ChurchProd} \to \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A\colon \texttt{tp}.\ \texttt{Unit} \\ \quad \texttt{tm} & := \lambda A\colon \texttt{tp}.\ \lambda t\colon \texttt{tm}.\ \ t :: A \\ \quad \_ \times \_ & := \ldots \texttt{unit} \\ \quad (\_, \_)^{-\times-} & := \lambda A\colon \texttt{tp}.\ \lambda A^*.\ \lambda B\colon \texttt{tp}.\ \lambda B^*. \\ & \qquad \lambda a\colon \texttt{tm}.\ \lambda a^*\colon a :: A. \\ & \qquad\quad \lambda b\colon \texttt{tm}.\ \lambda b^*\colon b :: B. \end{array}$$

Type Preservation as a Logical Relation

$$\textbf{theory}\ \texttt{ChurchProd} = \left\{\begin{array}{ll} \texttt{tp} & : \texttt{type} \\ \texttt{tm} & : \texttt{tp} \to \texttt{type} \\ \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ (\_, \_)^{-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \texttt{tm}\, A \to \texttt{tm}\, B \to \texttt{tm}\, A \times B \end{array}\right\}$$

$$\textbf{theory}\ \texttt{CurryProd} = \left\{\begin{array}{ll} \texttt{tp}, \texttt{tm} \colon \texttt{type} & \\ \quad \_ :: \_ & : \texttt{tm} \to \texttt{tp} \to \texttt{type} \\ \quad \_ \times \_ & : \texttt{tp} \to \texttt{tp} \to \texttt{tp} \\ \quad (\_, \_) & : \texttt{tm} \to \texttt{tm} \to \texttt{tm} \\ \quad (\_, \_)^{T,-\times-} & : \Pi A\, B \colon \texttt{tp}.\ \Pi a\, b \colon \texttt{tm}. \\ & \qquad a :: A \to b :: B \to (a, b) :: (A \end{array}\right.$$

$$\begin{array}{lllll} \textbf{relation} & \texttt{TypePres} \colon & \texttt{TypeEras} \colon & \texttt{ChurchProd} \to & \texttt{CurryProd} \\ \quad \texttt{tp} & := \lambda A \colon \texttt{tp}.\ \texttt{Unit} & & & \\ \quad \texttt{tm} & := \lambda A \colon \texttt{tp}.\ \lambda t \colon \texttt{tm}.\ \ t :: A & & & \\ \quad \_ \times \_ & := \ldots \texttt{unit} & & & \\ \quad (\_, \_)^{-\times-} & := \lambda A \colon \texttt{tp}.\ \lambda A^*.\ \lambda B \colon \texttt{tp}.\ \lambda B^*. & & & \\ & \qquad \lambda a \colon \texttt{tm}.\ \lambda a^* \colon a :: A. & & & \\ & \qquad\quad \lambda b \colon \texttt{tm}.\ \lambda b^* \colon b :: B.\quad (a, b)^{T, A \times B}\ a^*\ b^* & & & \end{array}$$

Example 3: Summary

- formalized two flavors of type theories: `ChurchProd`, `CurryProd`

  extrinsic vs. intrinsic

  $\mathtt{tm} \colon \mathtt{tp} \to \mathtt{type}$ vs. $\mathtt{tm} \colon \mathtt{type}$

- saw type erasure as a "lossy" translation

$$\begin{array}{l} \textbf{view}\ \mathtt{TypeEras} \colon \mathtt{ChurchProd} \to \mathtt{CurryProd} = \{ \\ \qquad \mathtt{tm} := \lambda\_.\ \mathtt{tm} \\ \qquad \vdots \\ \} \end{array}$$

- used a logical relation to complement the morphism to recover a meta theorem (within the system!)

$$\begin{array}{l} \textbf{relation}\ \mathtt{TypePres} \colon\ \mathtt{TypeEras} \colon \mathtt{ChurchProd} \to \mathtt{CurryProd} = \{ \\ \qquad \mathtt{tm} := \lambda A \colon \mathtt{tp}.\ \lambda t \colon \mathtt{tm}.\ \ t :: A \\ \qquad \vdots \\ \} \end{array}$$

Overall Summary

Logical Relations:

- many type theories admit this proof technique:
  1. map every type τ to an n-ary relation $C_\tau(-)$
  2. prove that every constructor $f\colon \tau_1 \to \ldots \to \tau_n \to \tau$ preserves the relation $C_\tau(-)$
- "the basic lemma": if $t\colon \tau$, then $C_\tau(t)$

Concretely in MMT/LF: for morphisms $\mu_1, \ldots, \mu_n \colon S \to T$

$$\textbf{relation}\ r\colon\ \mu_1 \times \cdots \times \mu_n\colon S \to T$$

- special case for morphisms being identities:
  - $\texttt{TND}\colon\ id_{PL}\colon \texttt{PL} \to \texttt{PL}$: unary congruence proving $p\colon \texttt{prop} \Longrightarrow\ \vdash p \vee \neg p$
  - $\texttt{LRA}\colon\ id_{PL} \times id_{\texttt{PL}}\colon \texttt{PL} \to \texttt{PL}$: (binary) congruence whose well-typedness shows $\Leftrightarrow$ being a congruence
- special case

$$\textbf{relation}\ \texttt{TypePres}\colon\ \texttt{TypeEras}\colon \texttt{ChurchProd} \to \texttt{CurryProd}$$

  - complements lossy translation TypeEras
  - $\vdash_{\texttt{ChurchProd}} t\colon \texttt{tm}\, A \ \Longrightarrow\ \vdash_{\texttt{CurryProd}} \mathrm{wit}\colon \texttt{TypeEras}(t) :: \texttt{TypeEras}(A)$

Future Work

...

TODO

log relations incomplete wrt mapping to natural deduction calculus things
why do we need systematic renaming by priming?
def and basic lemma for binary congruence correct? how do congruence properties follow from basic lemma?
TODO: Choose other letter than r for relation variables on previous slide!
Warning: slightly misrepresented logical relations as nothing more than inductions

Further Pointers I

- Logical Relations:
  - refer to [**ahmed_log_rel**][2] for an intro showing strong normalization and type safety for STLC
  - some interesting thoughts on relations with automata simulations: [**stackexchange_log_rel_and_simulations**]
- OMDoc/Mmt Language and Representation: [**rabe:howto:14**; **RabMue:WADT18**]
- Mmt System: [**MueRab:rpfsm19**; **ShaRab:dcm19**; **Rabe:MAGMS13**]
- Old LF papers: [**HarperEtAl:affdl93**]
- Diagram Operators: TODO insert WADT paper [**ShaRab:dcm19**]

[2] available as recorded videos; notes can be found at [**skorstengaard_log_rel**]