Fundamental Theorems for Free

Logical Relations & Parametricity for Substructural Type Theory and Beyond

Corinthia Beatrix Aberlé (she/her)

February 14, 2025

What is a Logical Relation?

LOGICAL RELATIONS are a powerful proof technique that can be used to prove properties of type systems such as normalization, canonicity, and parametricity, that typically evade straightforward proof by induction.

What is a Logical Relation?

Logical relations are a powerful proof technique that can be used to prove properties of type systems such as normalization, canonicity, and parametricity, that typically evade straightforward proof by induction.

But what is a logical relation?

What is a Logical Relation?

Logical relations are a powerful proof technique that can be used to prove properties of type systems such as normalization, canonicity, and parametricity, that typically evade straightforward proof by induction.

But what is a logical relation?

The usual way of introducing logical relations is by example, with the term "logical relations" being applied to anything that sufficiently resembles other logical relations arguments, rather than having a precise definition.

What is a Logical Relation?

Logical relations are a powerful proof technique that can be used to prove properties of type systems such as normalization, canonicity, and parametricity, that typically evade straightforward proof by induction.

But what is a logical relation?

The usual way of introducing logical relations is by example, with the term “logical relations” being applied to anything that sufficiently resembles other logical relations arguments, rather than having a precise definition.

This lack of precision makes for difficulty in extending logical relations arguments to novel type systems, since there is not a clear standard by which to judge whether the relations one defines are the “right” ones.

What is a Logical Relation?

LOGICAL RELATIONS are a powerful proof technique that can be used to prove properties of type systems such as normalization, canonicity, and parametricity, that typically evade straightforward proof by induction.

But what is a logical relation?

The usual way of introducing logical relations is by example, with the term "logical relations" being applied to anything that sufficiently resembles other logical relations arguments, rather than having a precise definition.

This lack of precision makes for difficulty in extending logical relations arguments to novel type systems, since there is not a clear standard by which to judge whether the relations one defines are the "right" ones.

In this talk, I will outline some of my own recent work on developing logical relations to prove parametricity theorems for substructural type systems, using this as a jumping-off point to discuss a more general categorical *recipe* for logical relations, that can be used to derive these and other examples.

Simply-Typed λ-Calculus

Simply-Typed λ-Calculus (STLC) is the internal language of Cartesian Closed Categories (CCCs), i.e. there is an equivalence of categories:

$$\mathbf{STLC} \underset{\mathcal{L}}{\overset{\mathsf{Syn}}{\rightleftarrows}} \mathbf{CCC} \quad (\simeq)$$

where

Simply-Typed λ-Calculus

Simply-Typed λ-Calculus (STLC) is the internal language of Cartesian Closed Categories (CCCs), i.e. there is an equivalence of categories:

$$\mathbf{STLC} \underset{\mathcal{L}}{\overset{\mathsf{Syn}}{\underset{\simeq}{\rightleftarrows}}} \mathbf{CCC}$$

where

- **STLC** is the category of theories in STLC (aka λ-theories) and translations between them.

Simply-Typed λ-Calculus

Simply-Typed λ**-Calculus (STLC)** is the internal language of Cartesian Closed Categories (CCCs), i.e. there is an equivalence of categories:

$$\mathbf{STLC} \underset{\mathcal{L}}{\overset{\mathsf{Syn}}{\underset{\simeq}{\rightleftarrows}}} \mathbf{CCC}$$

where

- **STLC** is the category of theories in STLC (aka λ-theories) and translations between them.
- **CCC** is the category of Cartesian Closed Categories and functors that (strictly) preserve finite products and exponentials between them.

Simply-Typed λ-Calculus

Simply-Typed λ-Calculus (STLC) is the internal language of Cartesian Closed Categories (CCCs), i.e. there is an equivalence of categories:

$$\mathbf{STLC} \underset{\mathcal{L}}{\overset{\mathsf{Syn}}{\underset{\simeq}{\rightleftarrows}}} \mathbf{CCC}$$

where

- **STLC** is the category of theories in STLC (aka λ-theories) and translations between them.
- **CCC** is the category of Cartesian Closed Categories and functors that (strictly) preserve finite products and exponentials between them.
- Syn constructs a “syntactic category” for each theory in STLC.

Simply-Typed λ-Calculus

Simply-Typed λ-Calculus (STLC) is the internal language of Cartesian Closed Categories (CCCs), i.e. there is an equivalence of categories:

$$\mathbf{STLC} \underset{\mathcal{L}}{\overset{\mathsf{Syn}}{\rightleftarrows}} \mathbf{CCC} \quad (\simeq)$$

where

- **STLC** is the category of theories in STLC (aka λ-theories) and translations between them.
- **CCC** is the category of Cartesian Closed Categories and functors that (strictly) preserve finite products and exponentials between them.
- Syn constructs a "syntactic category" for each theory in STLC.
- $\mathcal{L}$ defines a theory in STLC—the *internal language* of $\mathcal{C}$—for each Cartesian Closed Category $\mathcal{C}$.

Natural Number Objects & System T

A **natural number object** in a CCC $\mathcal{C}$, if one exists, is the universal object $\mathbb{N} \in \mathcal{C}$ equipped with morphisms

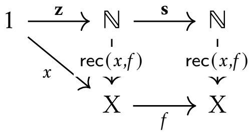
$$\mathbf{z} : 1 \to \mathbb{N} \quad \text{and} \quad \mathbf{s} : \mathbb{N} \to \mathbb{N}$$

Natural Number Objects & System T

A **natural number object** in a CCC $\mathcal{C}$, if one exists, is the universal object $\mathbb{N} \in \mathcal{C}$ equipped with morphisms

$$\mathbf{z} : 1 \to \mathbb{N} \quad \text{and} \quad \mathbf{s} : \mathbb{N} \to \mathbb{N}$$

i.e. such that for any object X equipped with $x : 1 \to \mathrm{X}$ and $f : \mathrm{X} \to \mathrm{X}$ there is a unique morphism $\mathsf{rec}(x, f) : \mathbb{N} \to \mathrm{X}$ that makes the following commute:

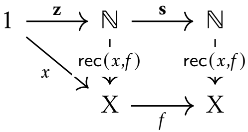


Natural Number Objects & System T

A **natural number object** in a CCC $\mathcal{C}$, if one exists, is the universal object $\mathbb{N} \in \mathcal{C}$ equipped with morphisms

$$\mathbf{z} : 1 \to \mathbb{N} \quad \text{and} \quad \mathbf{s} : \mathbb{N} \to \mathbb{N}$$

i.e. such that for any object X equipped with $x : 1 \to \mathrm{X}$ and $f : \mathrm{X} \to \mathrm{X}$ there is a unique morphism $\mathsf{rec}(x, f) : \mathbb{N} \to \mathrm{X}$ that makes the following commute:



System T (Gödel, 1958) adds to STLC a type of natural numbers $\mathbb{N}$, which makes $\mathsf{Syn}(\textbf{SysT})$ the initial object in the category $\mathbf{CCC}_{\mathbb{N}}$ of CCCs equipped with natural number objects and functors between them that (strictly) preserve finite products, exponentials, and natural number objects.

Canonicity for System T

For each (metatheoretic) natural number m, let $\overline{m}$ be defined by

$$\overline{0} = \mathbf{z} \quad \text{and} \quad \overline{n+1} = \mathbf{s}(\overline{n})$$

Canonicity for System T

For each (metatheoretic) natural number m, let $\overline{m}$ be defined by

$$\overline{0} = \mathbf{z} \quad \text{and} \quad \overline{n+1} = \mathbf{s}(\overline{n})$$

Theorem (Canonicity): every closed term $n : \mathbb{N}$ in System T is judgmentally equal to $\overline{m}$ for some natural number m.

Canonicity for System T

For each (metatheoretic) natural number m, let $\overline{m}$ be defined by

$$\overline{0} = \mathbf{z} \quad \text{and} \quad \overline{n+1} = \mathbf{s}(\overline{n})$$

Theorem (Canonicity): every closed term $n : \mathbb{N}$ in System T is judgmentally equal to $\overline{m}$ for some natural number m.
Proof: by a logical relations argument.

Logical Relations for System T

For each type τ in System T, let $[\![\tau]\!]$ be the set of closed terms of type τ, quotiented up to judgmental equality. Equivalently, this is $\mathrm{Hom}_{\mathsf{Syn}(\mathbf{SysT})}(1, \tau)$.

Logical Relations for System T

For each type τ in System T, let $[\![\tau]\!]$ be the set of closed terms of type τ, quotiented up to judgmental equality. Equivalently, this is $\mathrm{Hom}_{\mathsf{Syn}(\mathbf{SysT})}(1, \tau)$.

Then for each type τ in System T, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau]\!]$, as follows:

Logical Relations for System T

For each type τ in System T, let $[\![\tau]\!]$ be the set of closed terms of type τ, quotiented up to judgmental equality. Equivalently, this is $\mathrm{Hom}_{\mathsf{Syn}(\mathbf{SysT})}(1, \tau)$.

Then for each type τ in System T, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau]\!]$, as follows:

$$\mathbb{P}_{\mathbf{1}}(u) \quad \Longleftrightarrow \quad u \equiv \langle\rangle$$

Logical Relations for System T

For each type τ in System T, let $[\![\tau]\!]$ be the set of closed terms of type τ, quotiented up to judgmental equality. Equivalently, this is $\mathrm{Hom}_{\mathsf{Syn}(\mathbf{SysT})}(1, \tau)$.

Then for each type τ in System T, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau]\!]$, as follows:

$$\begin{array}{lcl} \mathbb{P}_{\mathbf{1}}(u) & \Longleftrightarrow & u \equiv \langle\rangle \\ \mathbb{P}_{\mathbb{N}}(n) & \Longleftrightarrow & \exists m \text{ s.t. } n \equiv \overline{m} \end{array}$$

Logical Relations for System T

For each type τ in System T, let $[\![\tau]\!]$ be the set of closed terms of type τ, quotiented up to judgmental equality. Equivalently, this is $\mathrm{Hom}_{\mathsf{Syn}(\mathbf{SysT})}(1, \tau)$.

Then for each type τ in System T, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau]\!]$, as follows:

$$
\begin{array}{rcl}
\mathbb{P}_{\mathbf{1}}(u) & \iff & u \equiv \langle\rangle \\
\mathbb{P}_{\mathbb{N}}(n) & \iff & \exists m \text{ s.t. } n \equiv \overline{m} \\
\mathbb{P}_{A\times B}(p) & \iff & \mathbb{P}_A(\pi_1(p)) \text{ and } \mathbb{P}_B(\pi_2(p))
\end{array}
$$

Logical Relations for System T

For each type τ in System T, let $[\![\tau]\!]$ be the set of closed terms of type τ, quotiented up to judgmental equality. Equivalently, this is $\mathrm{Hom}_{\mathsf{Syn}(\mathbf{SysT})}(1, \tau)$.

Then for each type τ in System T, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau]\!]$, as follows:

$$
\begin{array}{rcl}
\mathbb{P}_{\mathbf{1}}(u) & \iff & u \equiv \langle\rangle \\
\mathbb{P}_{\mathbb{N}}(n) & \iff & \exists m \text{ s.t. } n \equiv \overline{m} \\
\mathbb{P}_{\mathrm{A}\times\mathrm{B}}(p) & \iff & \mathbb{P}_{\mathrm{A}}(\pi_1(p)) \text{ and } \mathbb{P}_{\mathrm{B}}(\pi_2(p)) \\
\mathbb{P}_{\mathrm{A}\to\mathrm{B}}(f) & \iff & \forall a \in [\![\mathrm{A}]\!].\mathbb{P}_{\mathrm{A}}(a) \implies \mathbb{P}_{\mathrm{B}}(f(a))
\end{array}
$$

Logical Relations for System T

For each type τ in System T, let $[\![\tau]\!]$ be the set of closed terms of type τ, quotiented up to judgmental equality. Equivalently, this is $\mathrm{Hom}_{\mathsf{Syn}(\mathbf{SysT})}(1, \tau)$.

Then for each type τ in System T, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau]\!]$, as follows:

$$
\begin{array}{rcl}
\mathbb{P}_{\mathbf{1}}(u) & \iff & u \equiv \langle\rangle \\
\mathbb{P}_{\mathbb{N}}(n) & \iff & \exists m \text{ s.t. } n \equiv \overline{m} \\
\mathbb{P}_{A\times B}(p) & \iff & \mathbb{P}_A(\pi_1(p)) \text{ and } \mathbb{P}_B(\pi_2(p)) \\
\mathbb{P}_{A\to B}(f) & \iff & \forall a \in [\![A]\!].\mathbb{P}_A(a) \implies \mathbb{P}_B(f(a))
\end{array}
$$

Fundamental Theorem (FTLR): for every open term $\Gamma \vdash a : A$ in System T

$$\forall \gamma : \Gamma,\ \text{ if } \ \mathbb{P}_\Gamma(\gamma) \ \text{ then } \ \mathbb{P}_A(a[\gamma/\Gamma])$$

Logical Relations for System T

For each type τ in System T, let $[\![\tau]\!]$ be the set of closed terms of type τ, quotiented up to judgmental equality. Equivalently, this is $\mathrm{Hom}_{\mathsf{Syn}(\mathbf{SysT})}(1, \tau)$.

Then for each type τ in System T, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau]\!]$, as follows:

$$
\begin{aligned}
\mathbb{P}_{\mathbf{1}}(u) &\iff u \equiv \langle\rangle \\
\mathbb{P}_{\mathbb{N}}(n) &\iff \exists m \text{ s.t. } n \equiv \overline{m} \\
\mathbb{P}_{A\times B}(p) &\iff \mathbb{P}_A(\pi_1(p)) \text{ and } \mathbb{P}_B(\pi_2(p)) \\
\mathbb{P}_{A\to B}(f) &\iff \forall a \in [\![A]\!].\mathbb{P}_A(a) \implies \mathbb{P}_B(f(a))
\end{aligned}
$$

Fundamental Theorem (FTLR): for every open term $\Gamma \vdash a : A$ in System T

$$\forall \gamma : \Gamma,\ \text{ if }\ \mathbb{P}_\Gamma(\gamma)\ \text{ then }\ \mathbb{P}_A(a[\gamma/\Gamma])$$

Proof: induction on derivations.

Logical Relations for System T

For each type τ in System T, let $[\![\tau]\!]$ be the set of closed terms of type τ, quotiented up to judgmental equality. Equivalently, this is $\mathrm{Hom}_{\mathsf{Syn}(\mathbf{SysT})}(1, \tau)$.

Then for each type τ in System T, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau]\!]$, as follows:

$$
\begin{aligned}
\mathbb{P}_{\mathbf{1}}(u) &\iff u \equiv \langle\rangle \\
\mathbb{P}_{\mathbb{N}}(n) &\iff \exists m \text{ s.t. } n \equiv \overline{m} \\
\mathbb{P}_{\mathrm{A}\times\mathrm{B}}(p) &\iff \mathbb{P}_{\mathrm{A}}(\pi_1(p)) \text{ and } \mathbb{P}_{\mathrm{B}}(\pi_2(p)) \\
\mathbb{P}_{\mathrm{A}\to\mathrm{B}}(f) &\iff \forall a \in [\![\mathrm{A}]\!]. \mathbb{P}_{\mathrm{A}}(a) \implies \mathbb{P}_{\mathrm{B}}(f(a))
\end{aligned}
$$

Fundamental Theorem (FTLR): for every open term $\Gamma \vdash a : \mathrm{A}$ in System T

$$\forall \gamma : \Gamma, \text{ if } \mathbb{P}_\Gamma(\gamma) \text{ then } \mathbb{P}_{\mathrm{A}}(a[\gamma/\Gamma])$$

Proof: induction on derivations.

Canonicity then follows as a corollary of the fundamental theorem: for any term $n : \mathbb{N}$, we have $\mathbb{P}_{\mathbb{N}}(n)$, i.e. $n \equiv \overline{m}$ for some m.

Polymorphism

Consider the theory $\lambda[X]$ of a type X in STLC (i.e. the theory defined by a single atomic type X, with no constants or equations.) This theory has the following universal property in **STLC**:

Polymorphism

Consider the theory $\lambda[\mathrm{X}]$ of a type X in STLC (i.e. the theory defined by a single atomic type X, with no constants or equations.) This theory has the following universal property in **STLC**:

> *For any* λ*-theory* $\mathbb{T}$ *and any type* A *in* $\mathbb{T}$*, there is a unique translation*
>
> $$(-)[\mathrm{A}/\mathrm{X}] : \lambda[\mathrm{X}] \to \mathbb{T}$$
>
> *such that* $\mathrm{X}[\mathrm{A}/\mathrm{X}] = \mathrm{A}$*.*

Polymorphism

Consider the theory $\lambda[X]$ of a type X in STLC (i.e. the theory defined by a single atomic type X, with no constants or equations.) This theory has the following universal property in **STLC**:

> *For any* λ*-theory* $\mathbb{T}$ *and any type* A *in* $\mathbb{T}$*, there is a unique translation*
>
> $$(-)[A/X] : \lambda[X] \to \mathbb{T}$$
>
> *such that* $X[A/X] = A$*.*

We call a term $f : A[X] \to B[X]$ in $\lambda[X]$ a (unary) *polymorphic function*, since for any type C in any λ-theory $\mathbb{T}$, we can obtain a function

$$f[A/X] : A[C] \to B[C]$$

Polymorphism

Consider the theory $\lambda[X]$ of a type X in STLC (i.e. the theory defined by a single atomic type X, with no constants or equations.) This theory has the following universal property in **STLC**:

> *For any* λ*-theory* $\mathbb{T}$ *and any type* A *in* $\mathbb{T}$*, there is a unique translation*
>
> $$(-)[A/X] : \lambda[X] \to \mathbb{T}$$
>
> *such that* $X[A/X] = A$*.*

We call a term $f : A[X] \to B[X]$ in $\lambda[X]$ a (unary) *polymorphic function*, since for any type C in any λ-theory $\mathbb{T}$, we can obtain a function

$$f[A/X] : A[C] \to B[C]$$

Intuitively, Polymorphic functions can't inspect the types over which they are defined and so must behave uniformly for all types at which they are instantiated. But how to make this idea precise?

Polymorphism

Consider the theory $\lambda[X]$ of a type X in STLC (i.e. the theory defined by a single atomic type X, with no constants or equations.) This theory has the following universal property in **STLC**:

> *For any* λ*-theory* $\mathbb{T}$ *and any type* A *in* $\mathbb{T}$*, there is a unique translation*
>
> $$(-)[A/X] : \lambda[X] \to \mathbb{T}$$
>
> *such that* $X[A/X] = A$.

We call a term $f : A[X] \to B[X]$ in $\lambda[X]$ a (unary) *polymorphic function*, since for any type C in any λ-theory $\mathbb{T}$, we can obtain a function

$$f[A/X] : A[C] \to B[C]$$

Intuitively, Polymorphic functions can't inspect the types over which they are defined and so must behave uniformly for all types at which they are instantiated. But how to make this idea precise?

Reynolds (1983): Polymorphic functions should preserve all predicates/relations definable on types.

Parametricity

Let $\mathbb{T}$ be a λ-theory and C a type in $\mathbb{T}$. Let $P \subseteq [\![C]\!]$ be an arbitrary predicate on closed terms of type C in $\mathbb{T}$.

Parametricity

Let $\mathbb{T}$ be a λ-theory and C a type in $\mathbb{T}$. Let $P \subseteq [\![C]\!]$ be an arbitrary predicate on closed terms of type C in $\mathbb{T}$.

For each type $\tau[X]$ in $\lambda[X]$, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau[C]]\!]$ as follows:

Parametricity

Let $\mathbb{T}$ be a λ-theory and C a type in $\mathbb{T}$. Let $P \subseteq [\![C]\!]$ be an arbitrary predicate on closed terms of type C in $\mathbb{T}$.

For each type $\tau[X]$ in $\lambda[X]$, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau[C]]\!]$ as follows:

$$\mathbb{P}_X(x) \iff P(x)$$

Parametricity

Let $\mathbb{T}$ be a λ-theory and C a type in $\mathbb{T}$. Let $\mathrm{P} \subseteq [\![\mathrm{C}]\!]$ be an arbitrary predicate on closed terms of type C in $\mathbb{T}$.

For each type $\tau[\mathrm{X}]$ in $\lambda[\mathrm{X}]$, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau[\mathrm{C}]]\!]$ as follows:

$$\begin{aligned} \mathbb{P}_{\mathrm{X}}(x) &\iff \mathrm{P}(x) \\ \mathbb{P}_{\mathbf{1}}(u) &\iff u \equiv \langle\rangle \end{aligned}$$

Parametricity

Let $\mathbb{T}$ be a λ-theory and C a type in $\mathbb{T}$. Let $\mathrm{P} \subseteq [\![\mathrm{C}]\!]$ be an arbitrary predicate on closed terms of type C in $\mathbb{T}$.

For each type $\tau[\mathrm{X}]$ in $\lambda[\mathrm{X}]$, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau[\mathrm{C}]]\!]$ as follows:

$$
\begin{aligned}
\mathbb{P}_{\mathrm{X}}(x) &\iff \mathrm{P}(x) \\
\mathbb{P}_{\mathbf{1}}(u) &\iff u \equiv \langle\rangle \\
\mathbb{P}_{\mathrm{A}[\mathrm{X}]\times\mathrm{B}[\mathrm{X}]}(p) &\iff \mathbb{P}_{\mathrm{A}[\mathrm{X}]}(\pi_1(p)) \text{ and } \mathbb{P}_{\mathrm{B}[\mathrm{X}]}(\pi_2(p))
\end{aligned}
$$

Parametricity

Let $\mathbb{T}$ be a λ-theory and C a type in $\mathbb{T}$. Let $\mathrm{P} \subseteq [\![\mathrm{C}]\!]$ be an arbitrary predicate on closed terms of type C in $\mathbb{T}$.

For each type $\tau[\mathrm{X}]$ in $\lambda[\mathrm{X}]$, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau[\mathrm{C}]]\!]$ as follows:

$$
\begin{aligned}
\mathbb{P}_{\mathrm{X}}(x) &\iff \mathrm{P}(x) \\
\mathbb{P}_{\mathbf{1}}(u) &\iff u \equiv \langle\rangle \\
\mathbb{P}_{\mathrm{A}[\mathrm{X}]\times\mathrm{B}[\mathrm{X}]}(p) &\iff \mathbb{P}_{\mathrm{A}[\mathrm{X}]}(\pi_1(p)) \text{ and } \mathbb{P}_{\mathrm{B}[\mathrm{X}]}(\pi_2(p)) \\
\mathbb{P}_{\mathrm{A}[\mathrm{X}]\to\mathrm{B}[\mathrm{X}]}(f) &\iff \forall a : \mathrm{A}[\mathrm{C}].\, \mathbb{P}_{\mathrm{A}[\mathrm{X}]}(a) \implies \mathbb{P}_{\mathrm{B}}[\mathrm{X}](f(a))
\end{aligned}
$$

Parametricity

Let $\mathbb{T}$ be a λ-theory and C a type in $\mathbb{T}$. Let $\mathrm{P} \subseteq [\![\mathrm{C}]\!]$ be an arbitrary predicate on closed terms of type C in $\mathbb{T}$.

For each type $\tau[\mathrm{X}]$ in $\lambda[\mathrm{X}]$, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau[\mathrm{C}]]\!]$ as follows:

$$
\begin{aligned}
\mathbb{P}_{\mathrm{X}}(x) &\iff \mathrm{P}(x) \\
\mathbb{P}_{\mathbf{1}}(u) &\iff u \equiv \langle\rangle \\
\mathbb{P}_{\mathrm{A}[\mathrm{X}]\times\mathrm{B}[\mathrm{X}]}(p) &\iff \mathbb{P}_{\mathrm{A}[\mathrm{X}]}(\pi_1(p)) \text{ and } \mathbb{P}_{\mathrm{B}[\mathrm{X}]}(\pi_2(p)) \\
\mathbb{P}_{\mathrm{A}[\mathrm{X}]\to\mathrm{B}[\mathrm{X}]}(f) &\iff \forall a : \mathrm{A}[\mathrm{C}].\ \mathbb{P}_{\mathrm{A}[\mathrm{X}]}(a) \implies \mathbb{P}_{\mathrm{B}}[\mathrm{X}](f(a))
\end{aligned}
$$

FTLR: for any open term $\Gamma[\mathrm{X}] \vdash a : \mathrm{A}[\mathrm{X}]$ in $\lambda[\mathrm{X}]$

$$\forall \gamma : \Gamma[\mathrm{C}],\ \text{ if } \ \mathbb{P}_{\Gamma[\mathrm{X}]}(\gamma) \ \text{ then } \ \mathbb{P}_{\mathrm{A}[\mathrm{X}]}(a[\mathrm{C}/\mathrm{X}][\gamma/\Gamma])$$

Parametricity

Let $\mathbb{T}$ be a λ-theory and C a type in $\mathbb{T}$. Let $P \subseteq [\![C]\!]$ be an arbitrary predicate on closed terms of type C in $\mathbb{T}$.

For each type $\tau[X]$ in $\lambda[X]$, define a predicate $\mathbb{P}_\tau \subseteq [\![\tau[C]]\!]$ as follows:

$$
\begin{aligned}
\mathbb{P}_X(x) &\iff P(x) \\
\mathbb{P}_{\mathbf{1}}(u) &\iff u \equiv \langle\rangle \\
\mathbb{P}_{A[X]\times B[X]}(p) &\iff \mathbb{P}_{A[X]}(\pi_1(p)) \text{ and } \mathbb{P}_{B[X]}(\pi_2(p)) \\
\mathbb{P}_{A[X]\to B[X]}(f) &\iff \forall a : A[C].\ \mathbb{P}_{A[X]}(a) \implies \mathbb{P}_B[X](f(a))
\end{aligned}
$$

FTLR: for any open term $\Gamma[X] \vdash a : A[X]$ in $\lambda[X]$

$$\forall \gamma : \Gamma[C],\ \text{ if }\ \mathbb{P}_{\Gamma[X]}(\gamma)\ \text{ then }\ \mathbb{P}_{A[X]}(a[C/X][\gamma/\Gamma])$$

Proof: induction on the derivation of $\Gamma[X] \vdash a : A[X]$

Consequences of Parametricity, part 1

The Old Chestnut: every polymorphic function $\alpha : X \to X$ in $\lambda[X]$ is extensionally equivalent to the identity function.

Consequences of Parametricity, part 1

The Old Chestnut: every polymorphic function $\alpha : X \rightarrow X$ in $\lambda[X]$ is extensionally equivalent to the identity function.

Proof:

Consequences of Parametricity, part 1

The Old Chestnut: every polymorphic function $\alpha : X \to X$ in $\lambda[X]$ is extensionally equivalent to the identity function.

Proof:

- By parametricity, we know that for any type C in any theory $\mathbb{T}$, and any predicate $P \subseteq [\![C]\!]$, we have

$$\forall x : C,\ \ P(x) \implies P(\alpha[C/X](x))$$

Consequences of Parametricity, part 1

The Old Chestnut: every polymorphic function $\alpha : X \to X$ in $\lambda[X]$ is extensionally equivalent to the identity function.

Proof:

- By parametricity, we know that for any type C in any theory $\mathbb{T}$, and any predicate $P \subseteq [\![C]\!]$, we have

$$\forall x : C, \;\; P(x) \implies P(\alpha[C/X](x))$$

- Hence for any $c : C$, let $P = \{[\![c]\!]\}$. By construction we have

$$x \in P \iff x \equiv c$$

Consequences of Parametricity, part 1

The Old Chestnut: every polymorphic function $\alpha : X \to X$ in $\lambda[X]$ is extensionally equivalent to the identity function.

Proof:

- By parametricity, we know that for any type C in any theory $\mathbb{T}$, and any predicate $P \subseteq [\![C]\!]$, we have

$$\forall x : C,\ \ P(x) \implies P(\alpha[C/X](x))$$

- Hence for any $c : C$, let $P = \{[\![c]\!]\}$. By construction we have

$$x \in P \iff x \equiv c$$

- It follows that

$$\alpha[C/X](c) \equiv c$$

  for all $c : C$.

Consequences of Parametricity, part 2

Further Example: every polymorphic function $X \to X \to X \times X$ is extensionally equivalent to one of the following four functions:

$$\lambda x.\lambda y.(x, y) \quad \lambda x.\lambda y.(y, x) \quad \lambda x.\lambda y.(x, x) \quad \lambda x.\lambda y.(y, y)$$

Consequences of Parametricity, part 2

Further Example: every polymorphic function $X \to X \to X \times X$ is extensionally equivalent to one of the following four functions:

$$\lambda x.\lambda y.(x, y) \quad \lambda x.\lambda y.(y, x) \quad \lambda x.\lambda y.(x, x) \quad \lambda x.\lambda y.(y, y)$$

Proof:

Consequences of Parametricity, part 2

Further Example: every polymorphic function $X \to X \to X \times X$ is extensionally equivalent to one of the following four functions:

$$\lambda x.\lambda y.(x, y) \quad \lambda x.\lambda y.(y, x) \quad \lambda x.\lambda y.(x, x) \quad \lambda x.\lambda y.(y, y)$$

Proof:

- As before, we can unfold the parametricity theorem for α to the following: for any type C in any theory $\mathbb{T}$ and any predicate $P \subseteq [\![C]\!]$

$$\begin{aligned} &\forall x_0, x_1 : C,\ \ P(x_0) \text{ and } P(x_1) \\ &\quad \Longrightarrow\ P(\pi_1(\alpha[C/X](x_0)(x_1))) \text{ and } P(\pi_2(\alpha[C/X](x_0)(x_1))) \end{aligned}$$

Consequences of Parametricity, part 2

Further Example: every polymorphic function $X \to X \to X \times X$ is extensionally equivalent to one of the following four functions:

$$\lambda x.\lambda y.(x, y) \quad \lambda x.\lambda y.(y, x) \quad \lambda x.\lambda y.(x, x) \quad \lambda x.\lambda y.(y, y)$$

Proof:

- As before, we can unfold the parametricity theorem for α to the following: for any type C in any theory $\mathbb{T}$ and any predicate $P \subseteq [\![C]\!]$

$$\begin{aligned} &\forall x_0, x_1 : C, \;\; P(x_0) \text{ and } P(x_1) \\ &\quad \implies P(\pi_1(\alpha[C/X](x_0)(x_1))) \text{ and } P(\pi_2(\alpha[C/X](x_0)(x_1))) \end{aligned}$$

- Hence for any $c_0, c_1 : C$, we can take $P = \{c_0, c_1\}$, by which it follows that

$$\pi_1(\alpha[C/X](c_0)(c_1)) \in \{c_0, c_1\} \quad \text{and} \quad \pi_2(\alpha[C/X](c_0)(c_1)) \in \{c_0, c_1\}$$

Monoidal Categories

A **monoidal category** is a (pseudo)monoid object in **Cat**, i.e. a category $\mathcal{M}$ equipped with

$$\mathrm{I} \in \mathcal{M} \quad \text{and} \quad \otimes : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$$

that are unital and associative up to coherent isomorphism.

Monoidal Categories

A **monoidal category** is a (pseudo)monoid object in **Cat**, i.e. a category $\mathcal{M}$ equipped with

$$I \in \mathcal{M} \quad \text{and} \quad \otimes : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$$

that are unital and associative up to coherent isomorphism.

A **symmetric monoidal category (SMC)** is a monoidal category $\mathcal{M}$ that additionally carries a coherent natural isomorphism

$$A \otimes B \cong B \otimes A$$

Monoidal Categories

A **monoidal category** is a (pseudo)monoid object in **Cat**, i.e. a category $\mathcal{M}$ equipped with

$$I \in \mathcal{M} \quad \text{and} \quad \otimes : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$$

that are unital and associative up to coherent isomorphism.

A **symmetric monoidal category (SMC)** is a monoidal category $\mathcal{M}$ that additionally carries a coherent natural isomorphism

$$A \otimes B \cong B \otimes A$$

A monoidal category $\mathcal{M}$ is **(bi)closed** if for all $A \in \mathcal{M}$ both $A \otimes -$ and $- \otimes A$ have right adjoints $A \multimap -$ and $- \multimapinv A$, respectively.

Monoidal Categories

A **monoidal category** is a (pseudo)monoid object in **Cat**, i.e. a category $\mathcal{M}$ equipped with

$$I \in \mathcal{M} \quad \text{and} \quad \otimes : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$$

that are unital and associative up to coherent isomorphism.

A **symmetric monoidal category (SMC)** is a monoidal category $\mathcal{M}$ that additionally carries a coherent natural isomorphism

$$A \otimes B \cong B \otimes A$$

A monoidal category $\mathcal{M}$ is **(bi)closed** if for all $A \in \mathcal{M}$ both $A \otimes -$ and $- \otimes A$ have right adjoints $A \multimap -$ and $- \multimapinv A$, respectively.

Examples:

Monoidal Categories

A **monoidal category** is a (pseudo)monoid object in **Cat**, i.e. a category $\mathcal{M}$ equipped with

$$I \in \mathcal{M} \quad \text{and} \quad \otimes : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$$

that are unital and associative up to coherent isomorphism.

A **symmetric monoidal category (SMC)** is a monoidal category $\mathcal{M}$ that additionally carries a coherent natural isomorphism

$$A \otimes B \cong B \otimes A$$

A monoidal category $\mathcal{M}$ is **(bi)closed** if for all $A \in \mathcal{M}$ both $A \otimes -$ and $- \otimes A$ have right adjoints $A \multimap -$ and $- \multimapinv A$, respectively.

Examples:

- A monoid can be regarded as monoidal category with trivial morphisms, and likewise for commutative monoids and SMCs.

Monoidal Categories

A **monoidal category** is a (pseudo)monoid object in **Cat**, i.e. a category $\mathcal{M}$ equipped with

$$I \in \mathcal{M} \quad \text{and} \quad \otimes : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$$

that are unital and associative up to coherent isomorphism.

A **symmetric monoidal category (SMC)** is a monoidal category $\mathcal{M}$ that additionally carries a coherent natural isomorphism

$$A \otimes B \cong B \otimes A$$

A monoidal category $\mathcal{M}$ is **(bi)closed** if for all $A \in \mathcal{M}$ both $A \otimes -$ and $- \otimes A$ have right adjoints $A \multimap -$ and $- \multimapinv A$, respectively.

Examples:

- A monoid can be regarded as monoidal category with trivial morphisms, and likewise for commutative monoids and SMCs.
- A Cartesian Closed Category canonically carries the structure of a closed SMC with the monoidal structure given by finite products.

Substructural λ-Calculus

We may obtain *substructural* variants of STLC by restricting the usage of variables in forming terms. Specifically:

Substructural λ-Calculus

We may obtain *substructural* variants of STLC by restricting the usage of variables in forming terms. Specifically:

- **Linear λ-calculus** requires each variable to be used exactly once.

Substructural λ-Calculus

We may obtain *substructural* variants of STLC by restricting the usage of variables in forming terms. Specifically:

- **Linear λ-calculus** requires each variable to be used exactly once.
- **Ordered λ-calculus** additionally requires variables to be used in the order in which they occur in the typing context.

Substructural λ-Calculus

We may obtain *substructural* variants of STLC by restricting the usage of variables in forming terms. Specifically:

- **Linear λ-calculus** requires each variable to be used exactly once.
- **Ordered λ-calculus** additionally requires variables to be used in the order in which they occur in the typing context.

Ordered λ-calculus may then be characterized as the internal language of *biclosed monoidal categories*, and linear λ-calculus as the internal language of *closed symmetric monoidal categories*, in that there are equivalences

$$\mathbf{STLC}^{\mathbf{Ord}} \underset{\mathcal{L}}{\overset{\mathsf{Syn}}{\underset{\simeq}{\rightleftarrows}}} \mathbf{MonCat}^{\mathbf{Biclosed}}$$

and

$$\mathbf{STLC}^{\mathbf{Lin}} \underset{\mathcal{L}}{\overset{\mathsf{Syn}}{\underset{\simeq}{\rightleftarrows}}} \mathbf{SMC}^{\mathbf{Closed}}$$

Substructural Parametricity?

Intuitively, linear/ordered λ-calculus is more restrictive than ordinary λ-calculus—hence we should be able to obtain *stronger* parametricity theorems about terms in the former.

Substructural Parametricity?

Intuitively, linear/ordered λ-calculus is more restrictive than ordinary λ-calculus—hence we should be able to obtain *stronger* parametricity theorems about terms in the former.

We could try to repeat the proof of parametricity for ordinary STLC by considering (e.g.) the theory $\lambda^{Ord}[X]$ of a type X in ordered STLC and defining logical relations for this theory as before. However, this gives us no additional information about terms in ordered λ-calculus beyond what could already be deduced from the parametricity theorem for ordinary λ-calculus.

Substructural Parametricity?

Intuitively, linear/ordered λ-calculus is more restrictive than ordinary λ-calculus—hence we should be able to obtain *stronger* parametricity theorems about terms in the former.

We could try to repeat the proof of parametricity for ordinary STLC by considering (e.g.) the theory $\lambda^{Ord}[\mathrm{X}]$ of a type X in ordered STLC and defining logical relations for this theory as before. However, this gives us no additional information about terms in ordered λ-calculus beyond what could already be deduced from the parametricity theorem for ordinary λ-calculus.

For instance, we can apply the same argument as before to show that any polymorphic function $\alpha : \mathrm{X} \to \mathrm{X} \to \mathrm{X} \otimes \mathrm{X}$ in ordered λ-calculus must be equivalent to one of the following four functions

$$\lambda x.\lambda y.(x, y) \quad \lambda x.\lambda y.(y, x) \quad \lambda x.\lambda y.(x, x) \quad \lambda x.\lambda y.(y, y)$$

but we cannot deduce the stronger (but correct) result that in fact α must be equivalent to only the first of these.

Substructural Logical Relations

The solution is to parameterize our logical relations by a *monoid* $(M, \epsilon, \cdot)$.

Substructural Logical Relations

The solution is to parameterize our logical relations by a *monoid* $(M, \epsilon, \cdot)$.

Fix a theory $\mathbb{T}$ in ordered STLC and a type C in $\mathbb{T}$ with a relation $R \subseteq M \times [\![C]\!]$. Then for each type τ in $\lambda^{Ord}[X]$ we define a relation

$$\Vdash_\tau \subseteq M \times [\![\tau]\!]$$

as follows:

Substructural Logical Relations

The solution is to parameterize our logical relations by a *monoid* $(\mathrm{M}, \epsilon, \cdot)$.

Fix a theory $\mathbb{T}$ in ordered STLC and a type C in $\mathbb{T}$ with a relation $\mathrm{R} \subseteq \mathrm{M} \times [\![\mathrm{C}]\!]$. Then for each type τ in $\lambda^{Ord}[\mathrm{X}]$ we define a relation

$$\Vdash_\tau \subseteq \mathrm{M} \times [\![\tau]\!]$$

as follows:

$$m \Vdash_{\mathrm{X}} x \quad \Longleftrightarrow \quad \mathrm{R}(m, x)$$

Substructural Logical Relations

The solution is to parameterize our logical relations by a *monoid* $(M, \epsilon, \cdot)$.

Fix a theory $\mathbb{T}$ in ordered STLC and a type C in $\mathbb{T}$ with a relation $R \subseteq M \times [\![C]\!]$. Then for each type τ in $\lambda^{Ord}[X]$ we define a relation

$$\Vdash_\tau \subseteq M \times [\![\tau]\!]$$

as follows:

$$m \Vdash_X x \iff R(m, x)$$

$$m \Vdash_{\mathbf{1}} u \iff m = \epsilon \text{ and } u \equiv \langle\rangle$$

Substructural Logical Relations

The solution is to parameterize our logical relations by a *monoid* $(\mathrm{M}, \epsilon, \cdot)$.

Fix a theory $\mathbb{T}$ in ordered STLC and a type C in $\mathbb{T}$ with a relation $\mathrm{R} \subseteq \mathrm{M} \times [\![\mathrm{C}]\!]$. Then for each type τ in $\lambda^{Ord}[\mathrm{X}]$ we define a relation

$$\Vdash_\tau \subseteq \mathrm{M} \times [\![\tau]\!]$$

as follows:

$$\begin{aligned}
m \Vdash_{\mathrm{X}} x &\iff \mathrm{R}(m, x) \\
m \Vdash_{\mathbf{1}} u &\iff m = \epsilon \text{ and } u \equiv \langle\rangle \\
m \Vdash_{\mathrm{A[X]} \otimes \mathrm{B[X]}} \langle a, b\rangle &\iff \exists n, k \in \mathrm{M}, \text{ such that} \\
&\qquad m = n \cdot k,\ n \Vdash_{\mathrm{A[X]}} a, \text{ and } k \Vdash_{\mathrm{B[X]}} b
\end{aligned}$$

Substructural Logical Relations

The solution is to parameterize our logical relations by a *monoid* $(M, \epsilon, \cdot)$.

Fix a theory $\mathbb{T}$ in ordered STLC and a type C in $\mathbb{T}$ with a relation $R \subseteq M \times [\![C]\!]$. Then for each type τ in $\lambda^{Ord}[X]$ we define a relation

$$\Vdash_\tau \subseteq M \times [\![\tau]\!]$$

as follows:

$$\begin{aligned}
m \Vdash_X x &\iff R(m, x) \\
m \Vdash_{\mathbf{1}} u &\iff m = \epsilon \text{ and } u \equiv \langle\rangle \\
m \Vdash_{A[X] \otimes B[X]} \langle a, b\rangle &\iff \exists n, k \in M, \text{ such that} \\
&\qquad m = n \cdot k,\ n \Vdash_{A[X]} a, \text{ and } k \Vdash_{B[X]} b \\
m \Vdash_{A[X] \twoheadrightarrow B[X]} f &\iff \forall n \in M,\ a : A[C], \\
&\qquad n \Vdash_{A[X]} a \implies n \cdot m \Vdash_{B[X]} f(a)
\end{aligned}$$

Substructural Logical Relations

The solution is to parameterize our logical relations by a *monoid* $(M, \epsilon, \cdot)$.

Fix a theory $\mathbb{T}$ in ordered STLC and a type C in $\mathbb{T}$ with a relation $R \subseteq M \times [\![C]\!]$. Then for each type τ in $\lambda^{Ord}[X]$ we define a relation

$$\Vdash_\tau \subseteq M \times [\![\tau]\!]$$

as follows:

$$\begin{aligned}
m \Vdash_X x &\iff R(m, x) \\
m \Vdash_{\mathbf{1}} u &\iff m = \epsilon \text{ and } u \equiv \langle\rangle \\
m \Vdash_{A[X] \otimes B[X]} \langle a, b \rangle &\iff \exists n, k \in M, \text{ such that} \\
&\qquad m = n \cdot k,\ n \Vdash_{A[X]} a, \text{ and } k \Vdash_{B[X]} b \\
m \Vdash_{A[X] \twoheadrightarrow B[X]} f &\iff \forall n \in M,\ a : A[C], \\
&\qquad n \Vdash_{A[X]} a \implies n \cdot m \Vdash_{B[X]} f(a) \\
m \Vdash_{B[X] \twoheadleftarrow A[X]} f &\iff \forall n \in M,\ a : A[C], \\
&\qquad n \Vdash_{A[X]} a \implies m \cdot n \Vdash_{B[X]} f(a)
\end{aligned}$$

The Fundamental Theorem

We have the following:

The Fundamental Theorem

We have the following:

FTLR for Ordered STLC: For any type $\mathbb{C}$ in a theory $\mathbb{T}$ in Ordered STLC, if $\Gamma[X] \vdash a : A[X]$ is an open term in $\lambda^{Ord}[X]$, and $(M, \epsilon, \cdot)$ is *any* monoid, then

$$\forall \gamma : \Gamma[C],\ m \in M,\ \text{ if } m \Vdash_{\Gamma[X]} \gamma \text{ then } m \Vdash_{A[X]} a[C/X][\gamma/\Gamma]$$

The Fundamental Theorem

We have the following:

FTLR for Ordered STLC: For any type $\mathbb{C}$ in a theory $\mathbb{T}$ in Ordered STLC, if $\Gamma[X] \vdash a : A[X]$ is an open term in $\lambda^{Ord}[X]$, and $(M, \epsilon, \cdot)$ is *any* monoid, then

$$\forall \gamma : \Gamma[C],\ m \in M,\ \text{ if } m \Vdash_{\Gamma[X]} \gamma \text{ then } m \Vdash_{A[X]} a[C/X][\gamma/\Gamma]$$

and

The Fundamental Theorem

We have the following:

FTLR for Ordered STLC: For any type $\mathbb{C}$ in a theory $\mathbb{T}$ in Ordered STLC, if $\Gamma[X] \vdash a : A[X]$ is an open term in $\lambda^{Ord}[X]$, and $(M, \epsilon, \cdot)$ is *any* monoid, then

$$\forall \gamma : \Gamma[C],\ m \in M,\ \ \text{if } m \Vdash_{\Gamma[X]} \gamma \text{ then } m \Vdash_{A[X]} a[C/X][\gamma/\Gamma]$$

and

FTLR for Linear STLC: For any type $\mathbb{C}$ in a theory $\mathbb{T}$ in Linear STLC, if $\Gamma[X] \vdash a : A[X]$ is an open term in $\lambda^{Lin}[X]$, and $(M, \epsilon, \cdot)$ is any *commutative* monoid, then

$$\forall \gamma : \Gamma[C],\ m \in M,\ \ \text{if } m \Vdash_{\Gamma[X]} \gamma \text{ then } m \Vdash_{A[X]} a[C/X][\gamma/\Gamma]$$

The Fundamental Theorem

We have the following:

FTLR for Ordered STLC: For any type $\mathbb{C}$ in a theory $\mathbb{T}$ in Ordered STLC, if $\Gamma[X] \vdash a : A[X]$ is an open term in $\lambda^{Ord}[X]$, and $(M, \epsilon, \cdot)$ is *any* monoid, then

$$\forall \gamma : \Gamma[C],\ m \in M,\ \text{ if } m \Vdash_{\Gamma[X]} \gamma \text{ then } m \Vdash_{A[X]} a[C/X][\gamma/\Gamma]$$

and

FTLR for Linear STLC: For any type $\mathbb{C}$ in a theory $\mathbb{T}$ in Linear STLC, if $\Gamma[X] \vdash a : A[X]$ is an open term in $\lambda^{Lin}[X]$, and $(M, \epsilon, \cdot)$ is any *commutative* monoid, then

$$\forall \gamma : \Gamma[C],\ m \in M,\ \text{ if } m \Vdash_{\Gamma[X]} \gamma \text{ then } m \Vdash_{A[X]} a[C/X][\gamma/\Gamma]$$

Proof (both): induction on the derivation of $\Gamma[X] \vdash a : A[X]$

Applications of Substructural Parametricity

Theorem: every polymorphic function $F : \mathrm{X} \multimap \mathrm{X} \multimap \mathrm{X} \otimes \mathrm{X}$ in ordered STLC must be extensionally equivalent to

$$\lambda x.\lambda y.\langle x, y\rangle$$

Applications of Substructural Parametricity

Theorem: every polymorphic function $F : \mathrm{X} \multimap \mathrm{X} \multimap \mathrm{X} \otimes \mathrm{X}$ in ordered STLC must be extensionally equivalent to

$$\lambda x.\lambda y.\langle x, y\rangle$$

Proof: let M be the free monoid on two generators α, β, i.e. the set of strings $x_1 x_2 \ldots x_n$ with $x_i \in \{\alpha, \beta\}$, with $\cdot$ given by string concatenation.

Applications of Substructural Parametricity

Theorem: every polymorphic function $F : \mathrm{X} \multimap \mathrm{X} \multimap \mathrm{X} \otimes \mathrm{X}$ in ordered STLC must be extensionally equivalent to

$$\lambda x.\lambda y.\langle x, y\rangle$$

Proof: let M be the free monoid on two generators α, β, i.e. the set of strings $x_1 x_2 \ldots x_n$ with $x_i \in \{\alpha, \beta\}$, with $\cdot$ given by string concatenation.

Then for any type C with $c_0, c_1 : \mathrm{C}$, define $\mathrm{R} \subseteq \mathrm{M} \times [\![\mathrm{C}]\!]$ by

$$\mathrm{R}(m, x) \iff \text{either } m = \alpha \text{ and } x = c_0 \text{ or } m = \beta \text{ and } x = c_1$$

Applications of Substructural Parametricity

Theorem: every polymorphic function $F : X \multimap X \multimap X \otimes X$ in ordered STLC must be extensionally equivalent to

$$\lambda x.\lambda y.\langle x, y\rangle$$

Proof: let M be the free monoid on two generators α, β, i.e. the set of strings $x_1 x_2 \ldots x_n$ with $x_i \in \{\alpha, \beta\}$, with $\cdot$ given by string concatenation.

Then for any type C with $c_0, c_1 : C$, define $R \subseteq M \times [\![C]\!]$ by

$$R(m, x) \iff \text{either } m = \alpha \text{ and } x = c_0 \text{ or } m = \beta \text{ and } x = c_1$$

By parametricity for F, we have the following:

Applications of Substructural Parametricity

Theorem: every polymorphic function $F : \mathrm{X} \twoheadrightarrow \mathrm{X} \twoheadrightarrow \mathrm{X} \otimes \mathrm{X}$ in ordered STLC must be extensionally equivalent to

$$\lambda x.\lambda y.\langle x, y\rangle$$

Proof: let M be the free monoid on two generators α, β, i.e. the set of strings $x_1 x_2 \ldots x_n$ with $x_i \in \{\alpha, \beta\}$, with $\cdot$ given by string concatenation.

Then for any type C with $c_0, c_1 : \mathrm{C}$, define $\mathrm{R} \subseteq \mathrm{M} \times [\![\mathrm{C}]\!]$ by

$$\mathrm{R}(m, x) \iff \text{either } m = \alpha \text{ and } x = c_0 \text{ or } m = \beta \text{ and } x = c_1$$

By parametricity for F, we have the following:

- For all $m, n \in \mathrm{M}$, if $\mathrm{R}(m, c_0)$ and $\mathrm{R}(n, c_1)$, we have $F[\mathrm{C}/\mathrm{X}]\ c_0\ c_1 \equiv \langle c_0', c_1'\rangle$ and there exist $m', n' \in \mathrm{M}$ and $c_0', c_1' : \mathrm{A}$ such that $mn = m'n'$ and $\mathrm{R}(m', c_0')$ and $\mathrm{R}(n', c_1')$.

Applications of Substructural Parametricity

Theorem: every polymorphic function $F : X \multimap X \multimap X \otimes X$ in ordered STLC must be extensionally equivalent to

$$\lambda x.\lambda y.\langle x, y\rangle$$

Proof: let M be the free monoid on two generators α, β, i.e. the set of strings $x_1 x_2 \ldots x_n$ with $x_i \in \{\alpha, \beta\}$, with $\cdot$ given by string concatenation.

Then for any type C with $c_0, c_1 : C$, define $R \subseteq M \times [\![C]\!]$ by

$$R(m, x) \iff \text{either } m = \alpha \text{ and } x = c_0 \text{ or } m = \beta \text{ and } x = c_1$$

By parametricity for F, we have the following:

- For all $m, n \in M$, if $R(m, c_0)$ and $R(n, c_1)$, we have $F[C/X]\ c_0\ c_1 \equiv \langle c_0', c_1'\rangle$ and there exist $m', n' \in M$ and $c_0', c_1' : A$ such that $mn = m'n'$ and $R(m', c_0')$ and $R(n', c_1')$.
- Substituting α for m and β for n, this implies that we have $m', n' \in \{\alpha, \beta\}$ such that $\alpha\beta = m'n'$, which implies that $m' = \alpha$ and $n' = \beta$, and therefore $c_0' = c_0$ and $c_1' = c_1$, i.e.

$$F[C/X] c_0\ c_1 \equiv \langle c_0, c_1\rangle$$

How did I do that?

From Logical to Categorical Relations

Question: Can we derive parametricity from properties of $\mathsf{Syn}_{\lambda^{Ord}[X]}$, rather than going via laborious inductions on syntax?

From Logical to Categorical Relations

Question: Can we derive parametricity from properties of $\mathsf{Syn}_{\lambda^{Ord}[X]}$, rather than going via laborious inductions on syntax?

Fact: $\mathsf{Syn}_{\lambda^{Ord}[X]}$ is the initial object in the category of pointed biclosed monoidal categories and strict functors between them.

From Logical to Categorical Relations

Question: Can we derive parametricity from properties of $\mathsf{Syn}_{\lambda^{Ord}[X]}$, rather than going via laborious inductions on syntax?

Fact: $\mathsf{Syn}_{\lambda^{Ord}[X]}$ is the initial object in the category of pointed biclosed monoidal categories and strict functors between them.

Idea: given

- a monoid $(M, \epsilon, \cdot)$,
- a type C in a theory $\mathbb{T}$ of ordered STLC,
- a relation $R \subseteq M \times [\![C]\!]$

construct a pointed biclosed monoidal “Category of Relations” $\mathsf{Rel}_{M,C,R}$, and derive parametricity from the existence of a (strict) biclosed monoidal functor

$$\mathsf{Syn}_{\lambda^{Ord}[X]} \to \mathsf{Rel}_{M,C,R}$$

From Logical to Categorical Relations

Question: Can we derive parametricity from properties of $\mathsf{Syn}_{\lambda^{Ord}[X]}$, rather than going via laborious inductions on syntax?

Fact: $\mathsf{Syn}_{\lambda^{Ord}[X]}$ is the initial object in the category of pointed biclosed monoidal categories and strict functors between them.

Idea: given

- a monoid $(M, \epsilon, \cdot)$,
- a type C in a theory $\mathbb{T}$ of ordered STLC,
- a relation $R \subseteq M \times [\![C]\!]$

construct a pointed biclosed monoidal "Category of Relations" $\mathsf{Rel}_{M,C,R}$, and derive parametricity from the existence of a (strict) biclosed monoidal functor

$$\mathsf{Syn}_{\lambda^{Ord}[X]} \to \mathsf{Rel}_{M,C,R}$$

But how to construct $\mathsf{Rel}_{M,C,R}$?

Indexed Posets & The Grothendieck Construction

For a category C a C-**indexed poset** is a functor $C \to \mathbf{Pos}$.

Indexed Posets & The Grothendieck Construction

For a category $\mathcal{C}$ a $\mathcal{C}$**-indexed poset** is a functor $\mathcal{C} \to \mathbf{Pos}$.

Given a $\mathcal{C}$-indexed poset P, its **Grothendieck Construction** is a category *fibred over* $\mathcal{C}$, i.e.

$$\begin{array}{c} \int^{\mathcal{C}} \mathrm{P} \\ {\scriptstyle \pi}\big\downarrow \\ \mathcal{C} \end{array}$$

defined as follows:

Indexed Posets & The Grothendieck Construction

For a category $\mathcal{C}$ a $\mathcal{C}$-**indexed poset** is a functor $\mathcal{C} \to \mathbf{Pos}$.

Given a $\mathcal{C}$-indexed poset P, its **Grothendieck Construction** is a category *fibred over* $\mathcal{C}$, i.e.

$$\begin{array}{c} \int^{\mathcal{C}} \mathrm{P} \\ {\scriptstyle \pi}\big\downarrow \\ \mathcal{C} \end{array}$$

defined as follows:

- The objects of $\int^{\mathcal{C}} \mathrm{P}$ are pairs $(\mathrm{A}, p_{\mathrm{A}})$, where $\mathrm{A} \in \mathcal{C}$ and $p_{\mathrm{A}} \in \mathrm{P}(\mathrm{A})$.

Indexed Posets & The Grothendieck Construction

For a category $\mathcal{C}$ a $\mathcal{C}$**-indexed poset** is a functor $\mathcal{C} \to \mathbf{Pos}$.

Given a $\mathcal{C}$-indexed poset P, its **Grothendieck Construction** is a category *fibred over* $\mathcal{C}$, i.e.

$$\begin{array}{c} \int^{\mathcal{C}} \mathrm{P} \\ \downarrow \pi \\ \mathcal{C} \end{array}$$

defined as follows:

- The objects of $\int^{\mathcal{C}} \mathrm{P}$ are pairs $(\mathrm{A}, p_{\mathrm{A}})$, where $\mathrm{A} \in \mathcal{C}$ and $p_{\mathrm{A}} \in \mathrm{P}(\mathrm{A})$.
- A morphism $(\mathrm{A}, p_{\mathrm{A}}) \to (\mathrm{B}, p_{\mathrm{B}}) \in \int^{\mathcal{C}} \mathrm{P}$ is a morphism $f : \mathrm{A} \to \mathrm{B} \in \mathcal{C}$ such that

$$\mathrm{P}(f)(p_{\mathrm{A}}) \leq p_{\mathrm{B}}$$

Indexed Posets & The Grothendieck Construction

For a category $\mathcal{C}$ a $\mathcal{C}$-**indexed poset** is a functor $\mathcal{C} \to \mathbf{Pos}$.

Given a $\mathcal{C}$-indexed poset P, its **Grothendieck Construction** is a category *fibred over* $\mathcal{C}$, i.e.

$$\begin{array}{c} \int^{\mathcal{C}} \mathrm{P} \\ \pi \big\downarrow \\ \mathcal{C} \end{array}$$

defined as follows:

- The objects of $\int^{\mathcal{C}} \mathrm{P}$ are pairs $(\mathrm{A}, p_{\mathrm{A}})$, where $\mathrm{A} \in \mathcal{C}$ and $p_{\mathrm{A}} \in \mathrm{P}(\mathrm{A})$.
- A morphism $(\mathrm{A}, p_{\mathrm{A}}) \to (\mathrm{B}, p_{\mathrm{B}}) \in \int^{\mathcal{C}} \mathrm{P}$ is a morphism $f : \mathrm{A} \to \mathrm{B} \in \mathcal{C}$ such that

$$\mathrm{P}(f)(p_{\mathrm{A}}) \leq p_{\mathrm{B}}$$

- $\pi(\mathrm{A}, p_{\mathrm{A}}) = \mathrm{A}$ and $\pi(f) = f$.

Lax Monoidal Copresheaves

A **lax monoidal copresheaf (LMC)** on a monoidal category $\mathcal{M}$ is a functor $\mathcal{M} \to \mathbf{Set}$ equipped with coherent natural transformations:

$$\epsilon : 1 \to \Gamma(I) \quad \text{and} \quad (\cdot) : \Gamma(A) \times \Gamma(B) \to \Gamma(A \otimes B)$$

Lax Monoidal Copresheaves

A **lax monoidal copresheaf (LMC)** on a monoidal category $\mathcal{M}$ is a functor $\mathcal{M} \to \mathbf{Set}$ equipped with coherent natural transformations:

$$\epsilon : 1 \to \Gamma(I) \quad \text{and} \quad (\cdot) : \Gamma(A) \times \Gamma(B) \to \Gamma(A \otimes B)$$

Examples/closure properties of LMCs:

Lax Monoidal Copresheaves

A **lax monoidal copresheaf (LMC)** on a monoidal category $\mathcal{M}$ is a functor $\mathcal{M} \to \mathbf{Set}$ equipped with coherent natural transformations:

$$\epsilon : 1 \to \Gamma(I) \quad \text{and} \quad (\cdot) : \Gamma(A) \times \Gamma(B) \to \Gamma(A \otimes B)$$

Examples/closure properties of LMCs:

- For any monoidal category $(\mathcal{M}, I, \otimes)$, the **global sections** functor

  $$\mathrm{H}om(I, -) : \mathcal{M} \to \mathbf{Set}$$

  carries the structure of an LMC.

Lax Monoidal Copresheaves

A **lax monoidal copresheaf (LMC)** on a monoidal category $\mathcal{M}$ is a functor $\mathcal{M} \to \mathbf{Set}$ equipped with coherent natural transformations:

$$\epsilon : 1 \to \Gamma(I) \quad \text{and} \quad (\cdot) : \Gamma(A) \times \Gamma(B) \to \Gamma(A \otimes B)$$

Examples/closure properties of LMCs:

- For any monoidal category $(\mathcal{M}, I, \otimes)$, the **global sections** functor

  $$\mathrm{H}om(I, -) : \mathcal{M} \to \mathbf{Set}$$

  carries the structure of an LMC.
- A monoid M may be regarded as a LMC on the terminal category $\{*\}$.

Lax Monoidal Copresheaves

A **lax monoidal copresheaf (LMC)** on a monoidal category $\mathcal{M}$ is a functor $\mathcal{M} \longrightarrow \mathbf{Set}$ equipped with coherent natural transformations:

$$\epsilon : 1 \longrightarrow \Gamma(I) \quad \text{and} \quad (\cdot) : \Gamma(A) \times \Gamma(B) \longrightarrow \Gamma(A \otimes B)$$

Examples/closure properties of LMCs:

- For any monoidal category $(\mathcal{M}, I, \otimes)$, the **global sections** functor

  $$\mathrm{H}om(I, -) : \mathcal{M} \longrightarrow \mathbf{Set}$$

  carries the structure of an LMC.
- A monoid M may be regarded as a LMC on the terminal category $\{*\}$.
- Given LMCs $\Gamma : \mathcal{M} \longrightarrow \mathbf{Set}$ and $\Delta : \mathcal{N} \longrightarrow \mathbf{Set}$, the functor

  $$(\Gamma \times \Delta)(A, B) = \Gamma(A) \times \Delta(B)$$

  carries the structure of an LMC on $\mathcal{M} \times \mathcal{N}$.

Lax Monoidal Copresheaves

A **lax monoidal copresheaf (LMC)** on a monoidal category $\mathcal{M}$ is a functor $\mathcal{M} \to \mathbf{Set}$ equipped with coherent natural transformations:

$$\epsilon : 1 \to \Gamma(I) \quad \text{and} \quad (\cdot) : \Gamma(A) \times \Gamma(B) \to \Gamma(A \otimes B)$$

Examples/closure properties of LMCs:

- For any monoidal category $(\mathcal{M}, I, \otimes)$, the **global sections** functor

  $$Hom(I, -) : \mathcal{M} \to \mathbf{Set}$$

  carries the structure of an LMC.
- A monoid M may be regarded as a LMC on the terminal category $\{*\}$.
- Given LMCs $\Gamma : \mathcal{M} \to \mathbf{Set}$ and $\Delta : \mathcal{N} \to \mathbf{Set}$, the functor

  $$(\Gamma \times \Delta)(A, B) = \Gamma(A) \times \Delta(B)$$

  carries the structure of an LMC on $\mathcal{M} \times \mathcal{N}$.
- If $\Gamma : \mathcal{N} \to \mathbf{Set}$ is an LMC on $\mathcal{N}$, and $F : \mathcal{M} \to \mathcal{N}$ is a monoidal functor, then the precomposition of Γ with F carries the structure of an MDO on $\mathcal{M}$:

  $$\mathcal{M} \xrightarrow{F} \mathcal{N} \xrightarrow{\Gamma} \mathbf{Set}$$

Categories of Relations

Let $\mathcal{P}$: **Set** $\to$ **Pos** be the functor that takes a set S to its poset of subsets $\mathcal{P}(S)$, ordered by inclusion.

Categories of Relations

Let $\mathcal{P} : \mathbf{Set} \to \mathbf{Pos}$ be the functor that takes a set S to its poset of subsets $\mathcal{P}(S)$, ordered by inclusion.

Note that for any copresheaf $\Gamma : \mathcal{C} \to \mathbf{Set}$, the composite

$$\mathcal{C} \xrightarrow{\Gamma} \mathbf{Set} \xrightarrow{\mathcal{P}} \mathbf{Pos}$$

is a $\mathcal{C}$-indexed poset.

Let $\mathcal{P} : \mathbf{Set} \to \mathbf{Pos}$ be the functor that takes a set S to its poset of subsets $\mathcal{P}(S)$, ordered by inclusion.

Note that for any copresheaf $\Gamma : \mathcal{C} \to \mathbf{Set}$, the composite

$$\mathcal{C} \xrightarrow{\Gamma} \mathbf{Set} \xrightarrow{\mathcal{P}} \mathbf{Pos}$$

is a $\mathcal{C}$-indexed poset. Taking the Grothendieck construction of $\mathcal{P} \circ \Gamma$ then gives a category fibred over $\mathcal{C}$

$$\pi : \int^{\mathcal{C}} \mathcal{P} \circ \Gamma \to \mathcal{C}$$

Categories of Relations

Let $\mathcal{P} : \mathbf{Set} \to \mathbf{Pos}$ be the functor that takes a set S to its poset of subsets $\mathcal{P}(S)$, ordered by inclusion.

Note that for any copresheaf $\Gamma : \mathcal{C} \to \mathbf{Set}$, the composite

$$\mathcal{C} \xrightarrow{\Gamma} \mathbf{Set} \xrightarrow{\mathcal{P}} \mathbf{Pos}$$

is a $\mathcal{C}$-indexed poset. Taking the Grothendieck construction of $\mathcal{P} \circ \Gamma$ then gives a category fibred over $\mathcal{C}$

$$\pi : \int^{\mathcal{C}} \mathcal{P} \circ \Gamma \to \mathcal{C}$$

This is equivalently (the projection onto $\mathcal{C}$ of) the full subcategory of the comma category $\mathbf{Set} \downarrow \Gamma$ (aka the Freyd Cover or Sieprinski Cone) spanned by monic maps

$$P_A \rightarrowtail \Gamma(A)$$

i.e. (unary) *relations* on $\Gamma(A)$ for $A \in \mathcal{C}$.

Categories of Relations

Let $\mathcal{P} : \mathbf{Set} \to \mathbf{Pos}$ be the functor that takes a set S to its poset of subsets $\mathcal{P}(S)$, ordered by inclusion.

Note that for any copresheaf $\Gamma : \mathcal{C} \to \mathbf{Set}$, the composite

$$\mathcal{C} \xrightarrow{\Gamma} \mathbf{Set} \xrightarrow{\mathcal{P}} \mathbf{Pos}$$

is a $\mathcal{C}$-indexed poset. Taking the Grothendieck construction of $\mathcal{P} \circ \Gamma$ then gives a category fibred over $\mathcal{C}$

$$\pi : \int^{\mathcal{C}} \mathcal{P} \circ \Gamma \to \mathcal{C}$$

This is equivalently (the projection onto $\mathcal{C}$ of) the full subcategory of the comma category $\mathbf{Set} \downarrow \Gamma$ (aka the Freyd Cover or Sieprinski Cone) spanned by monic maps

$$P_A \rightarrowtail \Gamma(A)$$

i.e. (unary) *relations* on $\Gamma(A)$ for $A \in \mathcal{C}$.

Call this the **category of relations** Rel_Γ of Γ.

Day Convolution

Theorem (Day Convolution): given an LMC $\Gamma : \mathcal{M} \to \mathbf{Set}$ on a biclosed monoidal category $\mathcal{M}$, Rel_Γ carries the structure of a (strict) biclosed monoidal category over $\mathcal{M}$.

Day Convolution

Theorem (Day Convolution): given an LMC $\Gamma : \mathcal{M} \to \mathbf{Set}$ on a biclosed monoidal category $\mathcal{M}$, Rel_Γ carries the structure of a (strict) biclosed monoidal category over $\mathcal{M}$.

Proof sketch: Define a biclosed monoidal structure on Rel_Γ as follows:

Day Convolution

Theorem (Day Convolution): given an LMC $\Gamma : \mathcal{M} \to \mathbf{Set}$ on a biclosed monoidal category $\mathcal{M}$, Rel_Γ carries the structure of a (strict) biclosed monoidal category over $\mathcal{M}$.

Proof sketch: Define a biclosed monoidal structure on Rel_Γ as follows:

- The monoidal unit is the pair $(I, \{\epsilon\})$

Day Convolution

Theorem (Day Convolution): given an LMC $\Gamma : \mathcal{M} \to \mathbf{Set}$ on a biclosed monoidal category $\mathcal{M}$, Rel_Γ carries the structure of a (strict) biclosed monoidal category over $\mathcal{M}$.

Proof sketch: Define a biclosed monoidal structure on Rel_Γ as follows:

- The monoidal unit is the pair $(\mathrm{I}, \{\epsilon\})$
- For $\mathrm{A}, \mathrm{B} \in \mathcal{M}$ with $\mathrm{P_A} \subseteq \Gamma(\mathrm{A})$ and $\mathrm{P_B} \subseteq \Gamma(\mathrm{B})$, the monoidal product $(\mathrm{A}, \mathrm{P_A}) \otimes (\mathrm{B}, \mathrm{P_B})$ is

$$\left(\mathrm{A} \otimes \mathrm{B},\ \left\{x \in \Gamma(\mathrm{A} \otimes \mathrm{B}) \mid \ \exists a \in \mathrm{P_A},\ b \in \mathrm{P_B},\ \ x = a \cdot b\ \right\}\right)$$

Day Convolution

Theorem (Day Convolution): given an LMC $\Gamma : \mathcal{M} \to \mathbf{Set}$ on a biclosed monoidal category $\mathcal{M}$, Rel_Γ carries the structure of a (strict) biclosed monoidal category over $\mathcal{M}$.

Proof sketch: Define a biclosed monoidal structure on Rel_Γ as follows:

- The monoidal unit is the pair $(\mathrm{I}, \{\epsilon\})$
- For $\mathrm{A}, \mathrm{B} \in \mathcal{M}$ with $\mathrm{P_A} \subseteq \Gamma(\mathrm{A})$ and $\mathrm{P_B} \subseteq \Gamma(\mathrm{B})$, the monoidal product $(\mathrm{A}, \mathrm{P_A}) \otimes (\mathrm{B}, \mathrm{P_B})$ is

$$\left(\mathrm{A} \otimes \mathrm{B},\ \left\{x \in \Gamma(\mathrm{A} \otimes \mathrm{B}) \mid \ \exists a \in \mathrm{P_A},\ b \in \mathrm{P_B},\ \ x = a \cdot b\ \right\}\right)$$

- For A, B and $\mathrm{P_A}, \mathrm{P_B}$ as above, the left closure $(\mathrm{A}, \mathrm{P_A}) \multimap (\mathrm{B}, \mathrm{P_B})$ is

$$\left(\mathrm{A} \multimap \mathrm{B},\ \left\{f \in \Gamma(\mathrm{A} \multimap \mathrm{B}) \mid \ \forall a \in \mathrm{P_A},\ \ \Gamma(\alpha)(a \cdot f) \in \mathrm{P_B}\ \right\}\right)$$

  where $\alpha : \mathrm{A} \otimes (\mathrm{A} \multimap \mathrm{B}) \longrightarrow \mathrm{B}$ is the counit of the adjunction

$$\mathrm{A} \otimes - \dashv \mathrm{A} \multimap -$$

Day Convolution

Theorem (Day Convolution): given an LMC $\Gamma : \mathcal{M} \to \mathbf{Set}$ on a biclosed monoidal category $\mathcal{M}$, Rel_Γ carries the structure of a (strict) biclosed monoidal category over $\mathcal{M}$.

Proof sketch: Define a biclosed monoidal structure on Rel_Γ as follows:

- The monoidal unit is the pair $(\mathrm{I}, \{\epsilon\})$
- For $\mathrm{A}, \mathrm{B} \in \mathcal{M}$ with $\mathrm{P_A} \subseteq \Gamma(\mathrm{A})$ and $\mathrm{P_B} \subseteq \Gamma(\mathrm{B})$, the monoidal product $(\mathrm{A}, \mathrm{P_A}) \otimes (\mathrm{B}, \mathrm{P_B})$ is

$$\left(\mathrm{A} \otimes \mathrm{B},\ \left\{x \in \Gamma(\mathrm{A} \otimes \mathrm{B}) \mid\ \exists a \in \mathrm{P_A},\ b \in \mathrm{P_B},\ \ x = a \cdot b\ \right\}\right)$$

- For A, B and $\mathrm{P_A}, \mathrm{P_B}$ as above, the left closure $(\mathrm{A}, \mathrm{P_A}) \multimap (\mathrm{B}, \mathrm{P_B})$ is

$$\left(\mathrm{A} \multimap \mathrm{B},\ \left\{f \in \Gamma(\mathrm{A} \multimap \mathrm{B}) \mid\ \forall a \in \mathrm{P_A},\ \ \Gamma(\alpha)(a \cdot f) \in \mathrm{P_B}\ \right\}\right)$$

  where $\alpha : \mathrm{A} \otimes (\mathrm{A} \multimap \mathrm{B}) \longrightarrow \mathrm{B}$ is the counit of the adjunction

$$\mathrm{A} \otimes - \dashv \mathrm{A} \multimap -$$

- And similarly for the right closure.

Putting it all together

Given a theory $\mathbb{T}$ together with a type C in $\mathbb{T}$, and a monoid M together with with a relation $R \subseteq M \times [\![C]\!]$, define Γ to be the following composite

$$\Gamma \quad := \quad \mathsf{Syn}_{\lambda^{Ord}[X]} \xrightarrow{[C/X]} \mathsf{Syn}_{\mathbb{T}} \xrightarrow{Hom_{\mathsf{Syn}_{\mathbb{T}}}(I,-)\times M} \mathbf{Set}$$

Putting it all together

Given a theory $\mathbb{T}$ together with a type C in $\mathbb{T}$, and a monoid M together with with a relation $R \subseteq M \times [\![C]\!]$, define Γ to be the following composite

$$\Gamma \quad := \quad \mathsf{Syn}_{\lambda^{Ord}[X]} \xrightarrow{[C/X]} \mathsf{Syn}_{\mathbb{T}} \xrightarrow{Hom_{\mathsf{Syn}_{\mathbb{T}}}(I,-)\times M} \mathbf{Set}$$

By construction, this carries the structure of an LMC on $\mathsf{Syn}_{\lambda^{Ord}[X]}$.

Putting it all together

Given a theory $\mathbb{T}$ together with a type C in $\mathbb{T}$, and a monoid M together with with a relation $R \subseteq M \times [\![C]\!]$, define Γ to be the following composite

$$\Gamma \quad := \quad \mathsf{Syn}_{\lambda^{Ord}[X]} \xrightarrow{[C/X]} \mathsf{Syn}_{\mathbb{T}} \xrightarrow{Hom_{\mathsf{Syn}_{\mathbb{T}}}(I,-)\times M} \mathbf{Set}$$

By construction, this carries the structure of an LMC on $\mathsf{Syn}_{\lambda^{Ord}[X]}$.

Applying Day Convolution yields a biclosed monoidal structure on

$$\pi : \mathsf{Rel}_{\Gamma} \longrightarrow \mathsf{Syn}_{\lambda^{Ord}[X]}$$

Putting it all together

Given a theory $\mathbb{T}$ together with a type C in $\mathbb{T}$, and a monoid M together with with a relation $R \subseteq M \times [\![C]\!]$, define Γ to be the following composite

$$\Gamma \quad := \quad \mathsf{Syn}_{\lambda^{Ord}[X]} \xrightarrow{[C/X]} \mathsf{Syn}_{\mathbb{T}} \xrightarrow{Hom_{\mathsf{Syn}_{\mathbb{T}}}(I,-)\times M} \mathbf{Set}$$

By construction, this carries the structure of an LMC on $\mathsf{Syn}_{\lambda^{Ord}[X]}$.

Applying Day Convolution yields a biclosed monoidal structure on

$$\pi : \mathsf{Rel}_{\Gamma} \longrightarrow \mathsf{Syn}_{\lambda^{Ord}[X]}$$

Moreover, we have $(X, R) \in \mathsf{Rel}_{\Gamma}$. Hence there is a pointed biclosed monoidal functor

$$\Vdash_{(-)} : \mathsf{Syn}_{\lambda^{Ord}[X]} \longrightarrow \mathsf{Rel}_{\Gamma}$$

Putting it all together

Hence the composite $\pi \circ \Vdash_{(-)} : \mathsf{Syn}_{\lambda^{Ord}[X]} \to \mathsf{Syn}_{\lambda^{Ord}[X]}$ is also a pointed biclosed monoidal functor. But since $\mathsf{Syn}_{\lambda^{Ord}[X]}$ is the initial pointed biclosed monoidal category it follows that this must be the identity on $\mathsf{Syn}_{\lambda^{Ord}[X]}$, i.e.

$$\begin{array}{ccc} \mathsf{Syn}_{\lambda^{Ord}[X]} & \xrightarrow{\Vdash_{(-)}} & \mathsf{Rel}_{\Gamma} \\ & \searrow\!\!\!= & \downarrow \pi \\ & & \mathsf{Syn}_{\lambda^{Ord}[X]} \end{array}$$

Putting it all together

Hence the composite $\pi \circ \Vdash_{(-)} : \mathsf{Syn}_{\lambda^{Ord}}[X] \longrightarrow \mathsf{Syn}_{\lambda^{Ord}}[X]$ is also a pointed biclosed monoidal functor. But since $\mathsf{Syn}_{\lambda^{Ord}}[X]$ is the initial pointed biclosed monoidal category it follows that this must be the identity on $\mathsf{Syn}_{\lambda^{Ord}}[X]$, i.e.

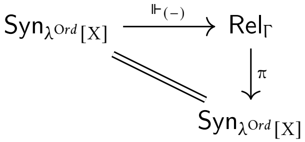
$$\begin{array}{ccc} \mathsf{Syn}_{\lambda^{Ord}}[X] & \xrightarrow{\Vdash_{(-)}} & \mathsf{Rel}_{\Gamma} \\ & \diagdown\!\!\diagdown & \big\downarrow \pi \\ & & \mathsf{Syn}_{\lambda^{Ord}}[X] \end{array}$$

Unpacking this, by functoriality of $\Vdash_{(-)}$, for all $f : A[X] \longrightarrow B[X] \in \mathsf{Syn}_{\lambda^{Ord}}[X]$ we have that

$$\mathcal{P}(\Gamma(f))(\Vdash_{A[X]}) \subseteq \Vdash_{B[X]}$$

Putting it all together

Hence the composite $\pi \circ \Vdash_{(-)} : \mathsf{Syn}_{\lambda^{Ord}[X]} \to \mathsf{Syn}_{\lambda^{Ord}[X]}$ is also a pointed biclosed monoidal functor. But since $\mathsf{Syn}_{\lambda^{Ord}[X]}$ is the initial pointed biclosed monoidal category it follows that this must be the identity on $\mathsf{Syn}_{\lambda^{Ord}[X]}$, i.e.



Unpacking this, by functoriality of $\Vdash_{(-)}$, for all $f : A[X] \to B[X] \in \mathsf{Syn}_{\lambda^{Ord}[X]}$ we have that

$$\mathcal{P}(\Gamma(f))(\Vdash_{A[X]}) \subseteq \Vdash_{B[X]}$$

i.e.

$$\forall m \in M,\ a \in Hom(1, A[C]),\ \text{ if } m \Vdash_{A[X]} a \text{ then } m \Vdash_{B[X]} f \circ a$$

Putting it all together

Hence the composite $\pi \circ \Vdash_{(-)} : \mathsf{Syn}_{\lambda^{Ord}}[X] \longrightarrow \mathsf{Syn}_{\lambda^{Ord}}[X]$ is also a pointed biclosed monoidal functor. But since $\mathsf{Syn}_{\lambda^{Ord}}[X]$ is the initial pointed biclosed monoidal category it follows that this must be the identity on $\mathsf{Syn}_{\lambda^{Ord}}[X]$, i.e.

$$\begin{array}{ccc} \mathsf{Syn}_{\lambda^{Ord}}[X] & \xrightarrow{\ \Vdash_{(-)}\ } & \mathsf{Rel}_{\Gamma} \\ & \searrow{=} & \downarrow{\pi} \\ & & \mathsf{Syn}_{\lambda^{Ord}}[X] \end{array}$$

Unpacking this, by functoriality of $\Vdash_{(-)}$, for all $f : A[X] \longrightarrow B[X] \in \mathsf{Syn}_{\lambda^{Ord}}[X]$ we have that

$$\mathcal{P}(\Gamma(f))(\Vdash_{A[X]}) \subseteq \Vdash_{B[X]}$$

i.e.

$$\forall m \in M,\ a \in Hom(1, A[C]),\ \ \text{if } m \Vdash_{A[X]} a \text{ then } m \Vdash_{B[X]} f \circ a$$

which precisely FLTR.

A Recipe for Logical Relations

To define logical relations and prove FTLR for a given type theory $\mathbb{T}$

A Recipe for Logical Relations

To define logical relations and prove FTLR for a given type theory $\mathbb{T}$

1. Show that $\mathsf{Syn}_{\mathbb{T}}$ is the initial object in some category of structured categories $\mathcal{C}$.

A Recipe for Logical Relations

To define logical relations and prove FTLR for a given type theory $\mathbb{T}$

1. Show that $\mathsf{Syn}_{\mathbb{T}}$ is the initial object in some category of structured categories $\mathcal{C}$.
2. For a suitably-chosen functor $\Gamma : \mathsf{Syn}_{\mathbb{T}} \to \mathbf{Set}$ and *category of relations* $\mathsf{Rel}_{\Gamma} \hookrightarrow \mathbf{Set} \downarrow \Gamma$, show that Rel_{Γ} is also an object of $\mathcal{C}$ over $\mathsf{Syn}_{\mathbb{T}}$ via the projection $\mathsf{Rel}_{\mathbb{T}} \to \mathsf{Syn}_{\mathbb{T}}$.

A Recipe for Logical Relations

To define logical relations and prove FTLR for a given type theory $\mathbb{T}$

1. Show that $\mathsf{Syn}_{\mathbb{T}}$ is the initial object in some category of structured categories $\mathcal{C}$.
2. For a suitably-chosen functor $\Gamma : \mathsf{Syn}_{\mathbb{T}} \to \mathbf{Set}$ and *category of relations* $\mathsf{Rel}_{\Gamma} \hookrightarrow \mathbf{Set} \downarrow \Gamma$, show that Rel_{Γ} is also an object of $\mathcal{C}$ over $\mathsf{Syn}_{\mathbb{T}}$ via the projection $\mathsf{Rel}_{\mathbb{T}} \to \mathsf{Syn}_{\mathbb{T}}$.
3. Derive FTLR from the existence of a section of this projection, which follows from initiality of $\mathsf{Syn}_{\mathbb{T}}$ as below:

$$\begin{array}{ccc} \mathsf{Syn}_{\mathbb{T}} & \dashrightarrow & \mathsf{Rel}_{\Gamma} \\ & \searrow\!\!\!= & \downarrow \\ & & \mathsf{Syn}_{\mathbb{T}} \end{array}$$

Conclusion

Category theory provides a general and powerfully simplifying language for logical relations. In this talk, we have seen how this method can be used to handle logical relations for substructural λ-calculus, but the applications of this method are seemingly without limit.

Conclusion

Category theory provides a general and powerfully simplifying language for logical relations. In this talk, we have seen how this method can be used to handle logical relations for substructural λ-calculus, but the applications of this method are seemingly without limit.

E.g., the categorical semantics of *dependent type theory* can be used to extend logical relations to systems with dependent types, and so on.

Conclusion

Category theory provides a general and powerfully simplifying language for logical relations. In this talk, we have seen how this method can be used to handle logical relations for substructural λ-calculus, but the applications of this method are seemingly without limit.

E.g., the categorical semantics of *dependent type theory* can be used to extend logical relations to systems with dependent types, and so on.

Several questions remain, however:

Conclusion

Category theory provides a general and powerfully simplifying language for logical relations. In this talk, we have seen how this method can be used to handle logical relations for substructural λ-calculus, but the applications of this method are seemingly without limit.

E.g., the categorical semantics of *dependent type theory* can be used to extend logical relations to systems with dependent types, and so on.

Several questions remain, however:

1. What makes the Day Convolution structure on a category of relations the "right" one for logical relations/parametricity on biclosed monoidal categories? Can we characterize this with a universal property?

Conclusion

Category theory provides a general and powerfully simplifying language for logical relations. In this talk, we have seen how this method can be used to handle logical relations for substructural λ-calculus, but the applications of this method are seemingly without limit.

E.g., the categorical semantics of *dependent type theory* can be used to extend logical relations to systems with dependent types, and so on.

Several questions remain, however:

1. What makes the Day Convolution structure on a category of relations the "right" one for logical relations/parametricity on biclosed monoidal categories? Can we characterize this with a universal property?
    - If you're interested in this, ask me about it—I have thoughts!

Conclusion

Category theory provides a general and powerfully simplifying language for logical relations. In this talk, we have seen how this method can be used to handle logical relations for substructural λ-calculus, but the applications of this method are seemingly without limit.

E.g., the categorical semantics of *dependent type theory* can be used to extend logical relations to systems with dependent types, and so on.

Several questions remain, however:

1. What makes the Day Convolution structure on a category of relations the "right" one for logical relations/parametricity on biclosed monoidal categories? Can we characterize this with a universal property?
   - If you're interested in this, ask me about it—I have thoughts!
2. How do the forms of *external* parametricity considered in this talk relate to type theories with *internal* parametricity and their categorical semantics? (Next time!)

Thank you!