Provable Isomorphisms of Types

Carlos C. Martínez

Department of Mathematics and Computer Science,

Wesleyan University

cmartinez@wesleyan.edu

November, 2004

Overview

- Motivation.
- Prelude.
- Equational Theories.
- Exploring proof theory.

Motivation:

How to retrieve software components?

Language	Names	Types
SML	fold	$(A \times B \to B) \to LIST[A] \to B \to B$
Ocaml	List.fold_left	$(B \to A \to B) \to B \to LIST[A] \to B$
CAML	list_it	$(A \to B \to B) \to LIST[A] \to B \to B$
Haskell	foldl	$(B \to A \to B) \to B \to LIST[A] \to B$

by name? No, arbitrary
by Types? Better...

Rittri'89: types as search keys in program libraries.

Prelude

Types: τ

- Let ι be a *basic type*

$$\tau ::= \iota \mid (\tau \to \tau) \mid \ldots$$

Terms: Λ

- Let x be a *variable*

$$\Lambda ::= x \mid (\lambda x.\Lambda) \mid (\Lambda\Lambda) \mid \ldots$$

Prelude

Statements, Declaration, and Basis: $\mathrm{t} : A$, $\mathrm{x} : A$, Γ

- A *statement* is of the form $\mathrm{t} : A$, where $\mathrm{t} \in \Lambda$, $A \in \tau$.
- A *Declaration* is of the form $\mathrm{x} : A$, where $\mathrm{x} \in \mathrm{Var}$, $A \in \tau$.
- A *Basis* Γ is a finite set of declarations.

Prelude

Type system:

$$\Gamma, x : A \vdash x : A \qquad (Axiom)$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash (\lambda x.t) : (A \to B)} \qquad (\to \text{Elimination})$$

$$\frac{\Gamma \vdash t : A \to B \qquad \Gamma \vdash u : A}{\Gamma \vdash (tu) : B} \qquad (\to \text{Introduction})$$

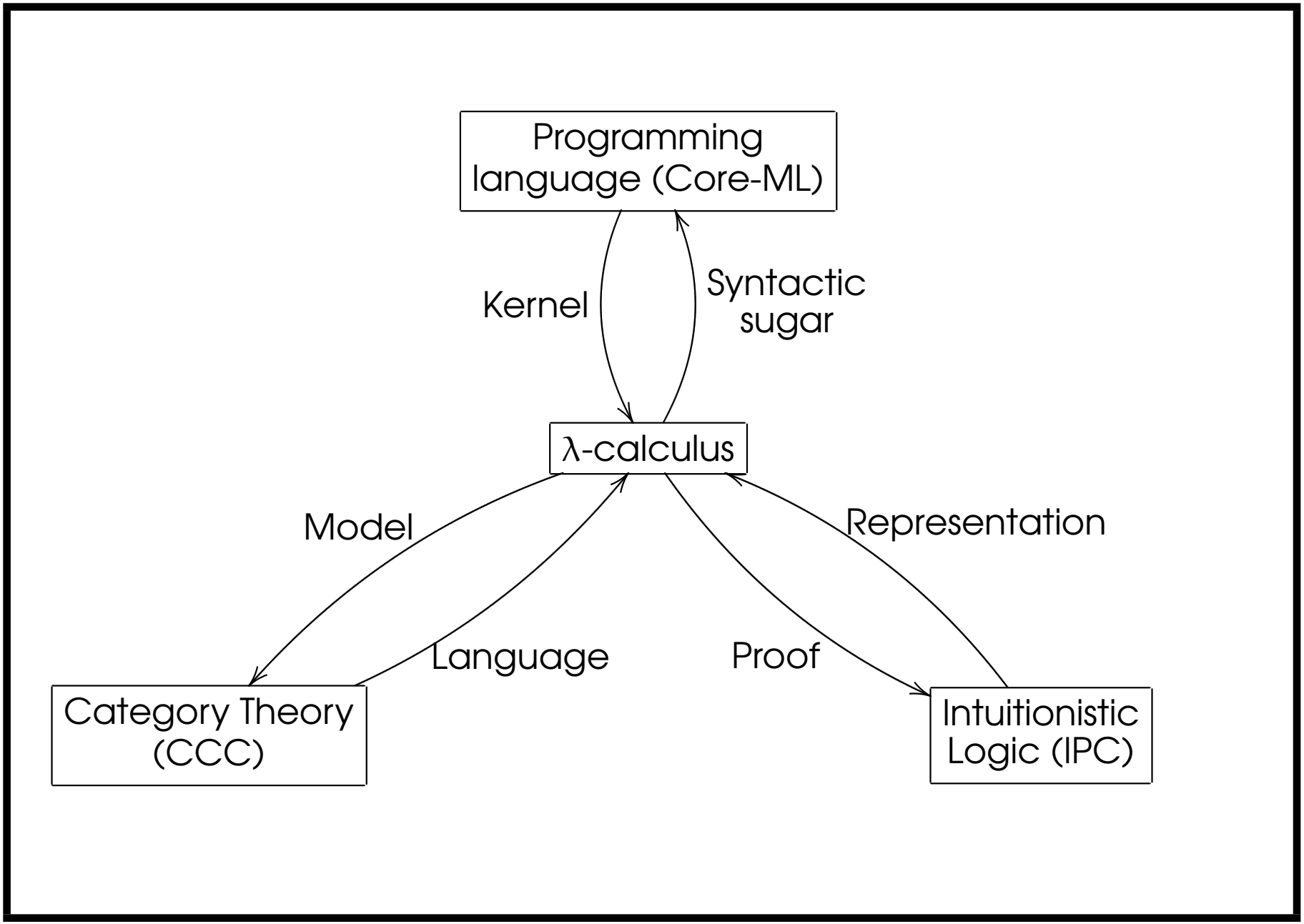
Typable Terms: $\Lambda^{\to}$

$$\Lambda^{\to} := \{t \in \Lambda : \exists \Gamma, A \vdash t : A\}$$

Prelude

Computations: $\triangleright$

- $(\lambda x.t)u \triangleright_\beta t[x := u]$ *(beta-reduction)*.
- $(\lambda x.tx) \triangleright_\eta t$, if x is not free in t *(eta-reduction)*.
- ...

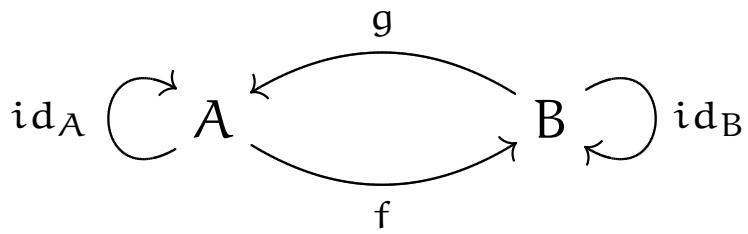

Programming
language (Core-ML)
Kernel
Syntactic
sugar
λ-calculus
Model
Representation
Language
Proof
Category Theory
(CCC)
Intuitionistic
Logic (IPC)

Isomorphism of types, object and formulae.

λ-calculus	Category	Logical System	Theory	References
$\lambda\beta\eta$		$\mathrm{IPC}(\Rightarrow)$	$\mathcal{S}wap$	Martin'72
$\lambda\beta\eta*$		$\mathrm{IPC}(\Rightarrow,\mathbf{T})$	$\mathcal{T}\mathrm{h}_{\mathbf{T}}$	DiCosmo'95
$\lambda\beta\eta\pi$	CC	$\mathrm{IPC}(\Rightarrow,\wedge)$	$\mathcal{T}\mathrm{h}_{\times}$	DiCosmo'95
$\lambda\beta\eta\pi*$	CCC	$\mathrm{IPC}(\Rightarrow,\wedge,\mathbf{T})$	$\mathcal{T}\mathrm{h}_{\times,\mathbf{T}}$	Sol83, BDCL'92
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$

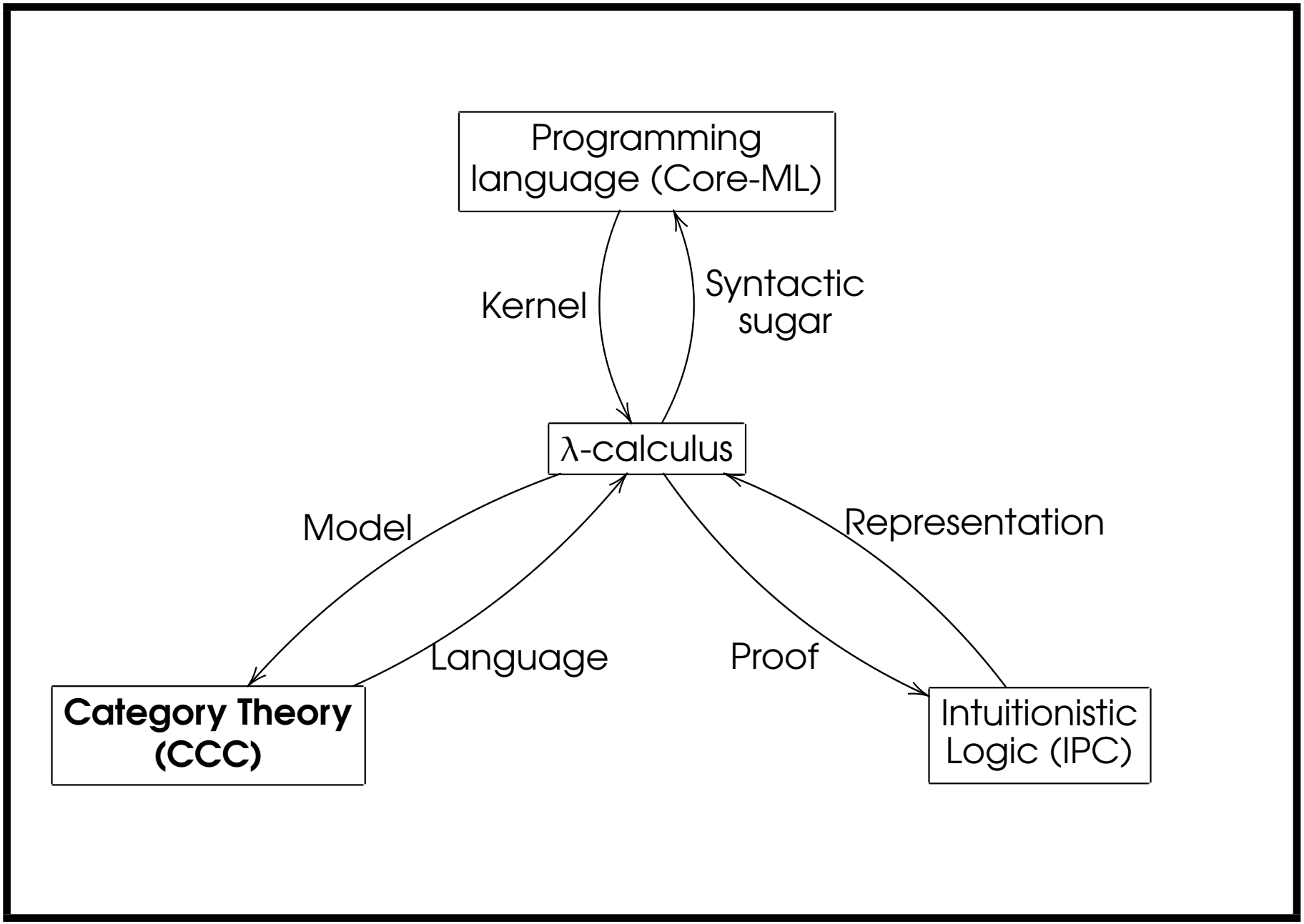
Isomorphism.

Definition *A and B are isomorphic if and only if there exist f and g such that:*



Where $\mathtt{A}, \mathtt{B}$ *and* $\mathtt{f}, \mathtt{g}$ *may be:*

λ-Calculus:	Types	invertible terms.
Category Theory:	Objects	invertible morphisms.
Logic:	Formulas	strong proofs.


Programming
language (Core-ML)
Kernel
Syntactic
sugar
λ-calculus
Model
Representation
Language
Proof
Category Theory
(CCC)
Intuitionistic
Logic (IPC)

Type Isomorphism:

Equational Theories.

(A0) $A \to (B \to C) = B \to (A \to C)$ (Swap)

(A1) $A \times B = B \times A$

(A2) $A \times (B \times C) = (A \times B) \times C$

(A3) $(A \times B) \to C = A \to (B \to C)$

(A4) $C \to (A \times B) = (C \to A) \times (C \to B)$

(A5) $A \times T = A$

(A6) $A \to T = T$

(A7) $T \to A = A$

Type Systems:

Type System	Axioms
Swap	A0
Product Types	A1, A2 & (*)
Linear Types	A1, A2, A3 & (*)
First Order Types	A1, A2, A3, A4 & (*)

Where (*) are A5 - A7.

Decidability

Via an appropriate Rewriting System; **confluent**, **strong normalizing** up to commutative product + sorting of primitive types.

(R2) $A(BC) > (AB)C$

(R3) $(C^B)^A > C^{AB}$

(R4) $(AB)^C > (A^C)(B^C)$

(R5) $A1 > A$

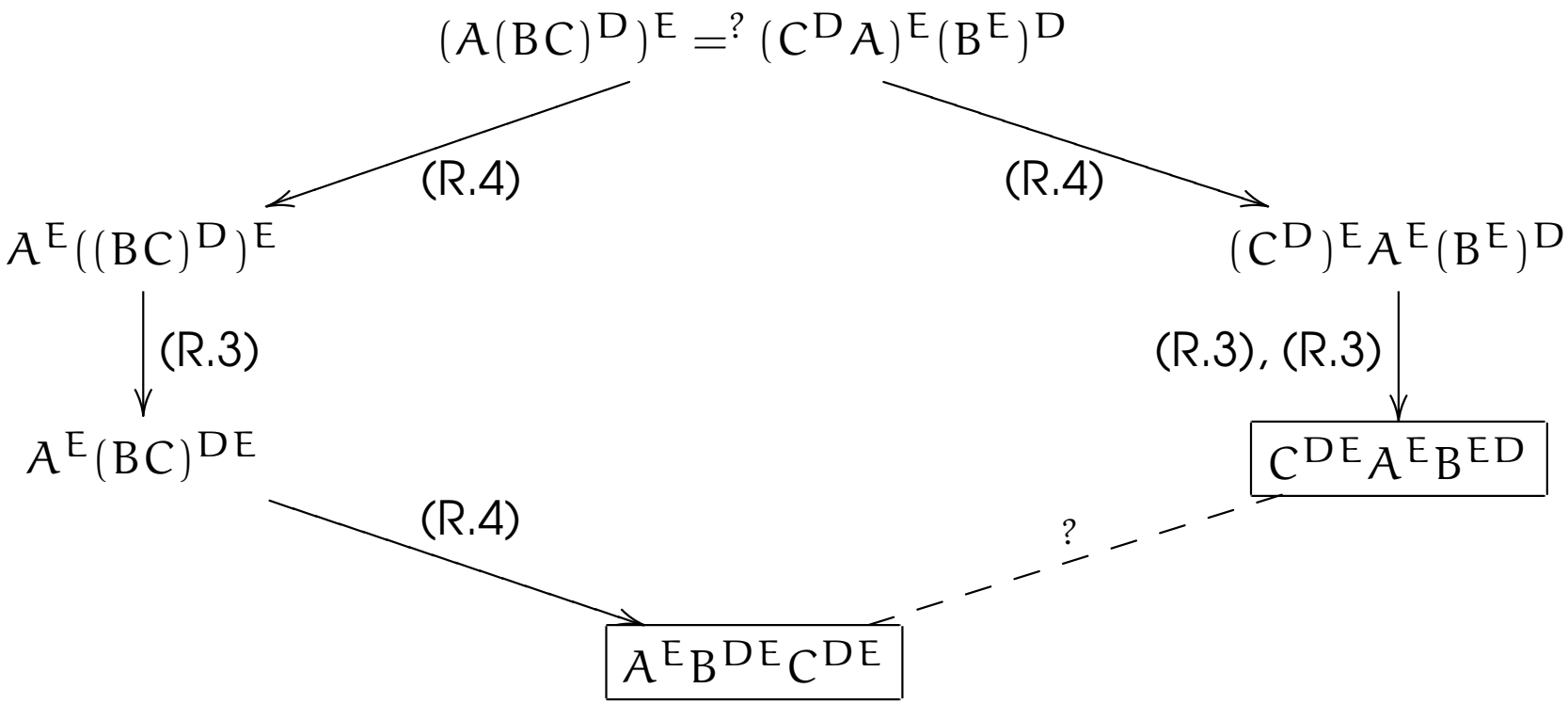
(R5)' $1A > A$

(R6) $A^1 > A$

(R7) $1^A > 1$

Observe: (R5) - (R7) eliminates unit types.

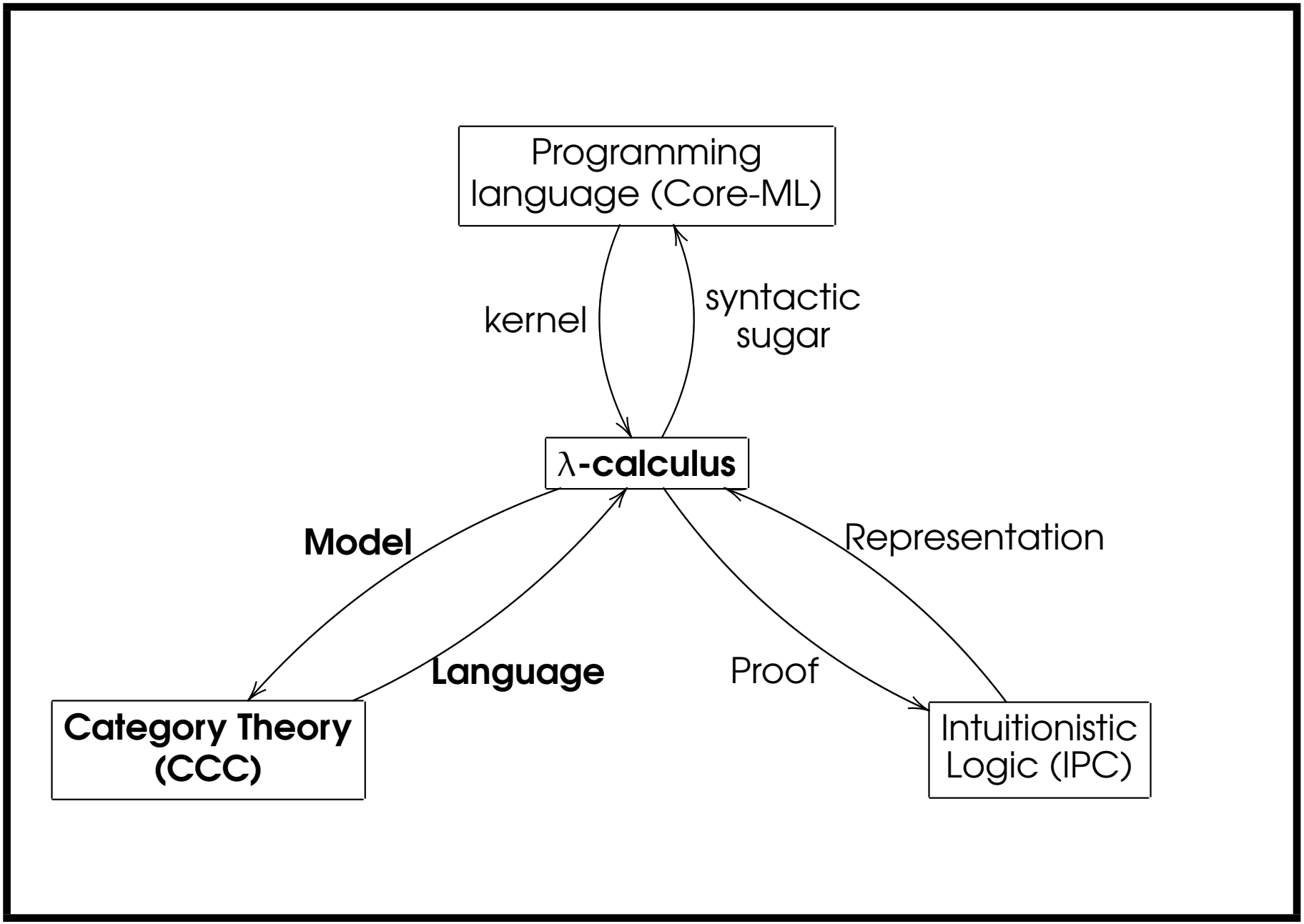
Example:



Sorting products: $C^{DE}A^E B^{ED} \rightarrow A^E B^{DE} C^{DE}$

Complexity:

Theorem *(Zibin, Gil, Considine'03)* *Type isomorphism can be decided in* $O(n \log^2(n))$ *time and* $O(n)$ *space, where* n *is the size of the input.*


Programming
language (Core-ML)
kernel
syntactic
sugar
λ-calculus
Model
Representation
Language
Proof
Category Theory
(CCC)
Intuitionistic
Logic (IPC)

Definable isomorphisms: λ-Calculus

Definition *Two type* A *and* B *are definably isomorphic* $(A \cong_d B)$ *iff there exists* λ*-terms* $M : A \to B$ *and* $N : B \to A$ *such that* $M \circ N = \lambda x^B.x$ *and* $N \circ M = \lambda x^A.x$*, the identities of type* A *and* B.

Note: $f \circ g = \lambda x.\lambda f.\lambda g.(f(gx))$

Semantic isomorphisms: Models

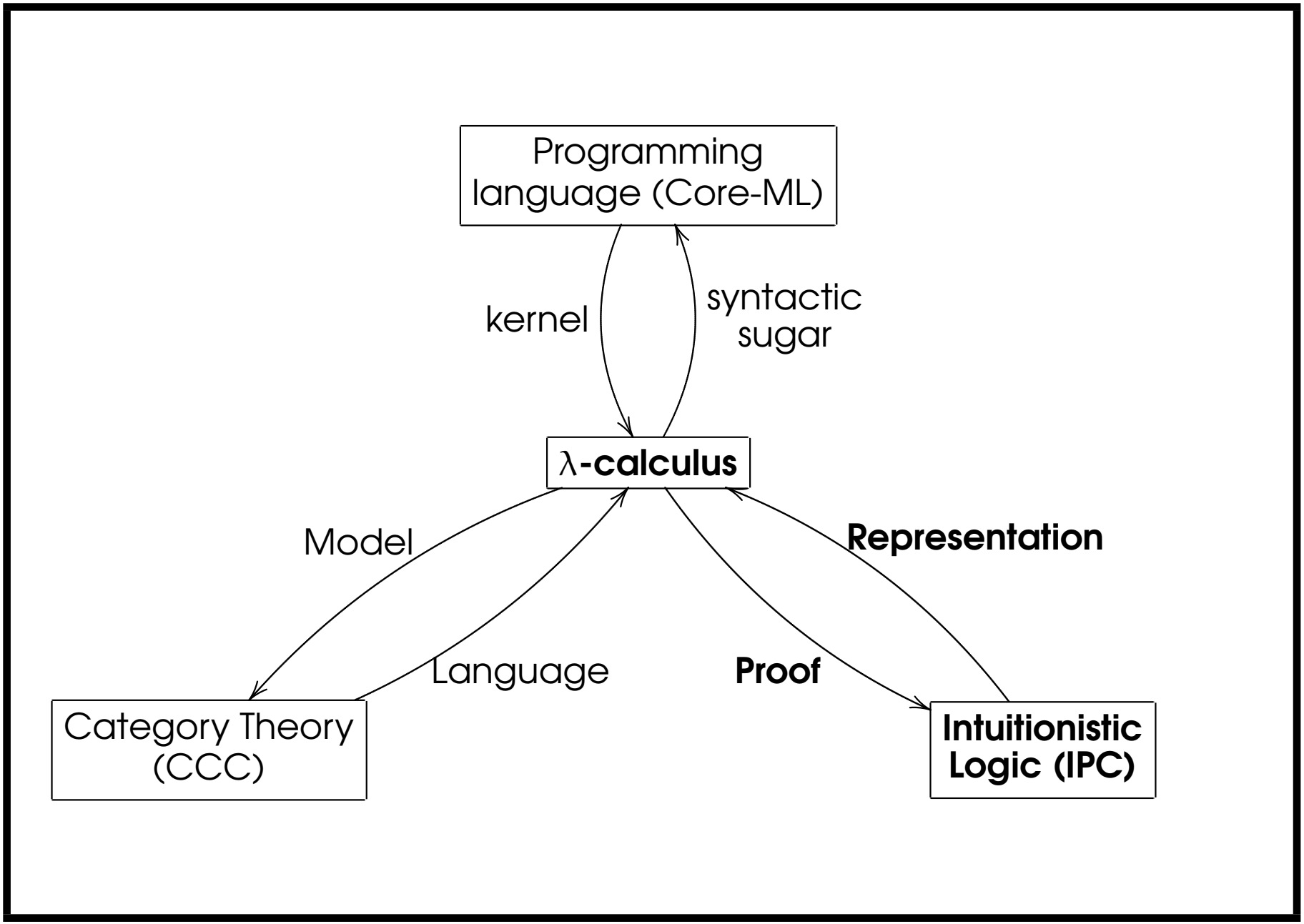
Definition *Two type* A *and* B *are isomorphic in a model* $\mathcal{M}$ *if their interpretations are isomorphic en* $\mathcal{M}$ *in a usual sense (i.e. there are in the model invertible functions* f *and* g *between them), we write* $\mathcal{M} \models \mathrm{A} \cong \mathrm{B}$. *Two types are semantically isomorphic* $\mathrm{A} \cong \mathrm{B}$, *if* $\mathcal{M} \models \mathrm{A} \cong \mathrm{B}$ *holds for every model* $\mathcal{M}$ *of the calculus.*

Definable and Semantic Isomorphisms

Theorem *Let* A *and* B *be types, then* $\mathrm{A} \cong \mathrm{B} \Leftrightarrow \mathrm{A} \cong_{\mathrm{d}} \mathrm{B}$.

Proof *(*$\Rightarrow$*) Straightforward. (*$\Leftarrow$*) Take the term model.*

Remark if we have $\mathcal{M} \models \mathrm{A} \cong \mathrm{B}$ for some particular model, it might be the case that $\mathrm{A} \ncong \mathrm{B}$


Programming
language (Core-ML)
kernel
syntactic
sugar
λ-calculus
Model
Representation
Language
Proof
Category Theory
(CCC)
Intuitionistic
Logic (IPC)

Isomorphisms in Logic

Definition *Two propositions* A *and* B *are strongly equivalent if*

- $A \Leftrightarrow B$*, this is there exists proofs* $f : A \Rightarrow B$ *and* $g : B \Rightarrow A$
- *The compositions* $f \circ g$ *and* $g \circ f$ *by* CUT *reduce, after normalization to* $\dfrac{}{A \vdash A}\,(Ax)$.

$$\dfrac{\dfrac{f}{(A \Rightarrow B)} \qquad \dfrac{g}{(B \Rightarrow A)}}{A \vdash A}\,(CUT)$$

Isomorphisms and Invertibility

Definition ***(Finite hereditary permutation, f.h.p.)*** *An untyped λ-term M is a f.h.p if and only if*

- $M = \lambda z.z$,
- $M = \lambda z.\lambda x_1 \ldots \lambda x_n.z(Q_1 x_{\pi(1)}) \ldots (Q_n x_{\pi(n)})$ *where* $\pi : n \to n$ *is a permutation and* Q_i *is a f.h.p. for all* $1 \leq i \leq n$

Theorem ***(Dezani-Ciancaglini 1976)*** *Let M be an untyped term possessing $\beta\eta$-normal form then M is invertible if and only if M is a finite hereditary permutation.*

Soundness and Completeness

Definition *We say that an equational theory Th is a sound theory of isomorphism for a calculus if*

$$\forall \mathrm{A}, \mathrm{B} \ \epsilon \ \mathcal{T} : \mathit{Th} \vdash \mathrm{A} = \mathrm{B} \Rightarrow \mathrm{A} \cong \mathrm{B}$$

Respectively, an equational theory Th is a complete theory of isomorphism for a calculus

$$\forall \mathrm{A}, \mathrm{B} \ \epsilon \ \mathcal{T} : \mathrm{A} \cong \mathrm{B} \Rightarrow \mathit{Th} \vdash \mathrm{A} = \mathrm{B}$$

Where $\mathcal{T}$ is an arbitrary type system.

Example: Swap

Theorem ***(Soundness)***

$$\forall A, B \in \mathcal{T} : \mathit{Swap} \vdash A = B \Rightarrow A \cong B$$

Proof Recall $A \cong B \Leftrightarrow A \cong_d B$, then it is enough to show $\mathit{Swap} \vdash A = B \Rightarrow A \cong_d B$, observe $\lambda x^{A \to (B \to C)}.\lambda y^{B \to C}.\lambda z^{A}.xzy$ is invertible and witness the equation $A \to (B \to C) = B \to (A \to C)$.

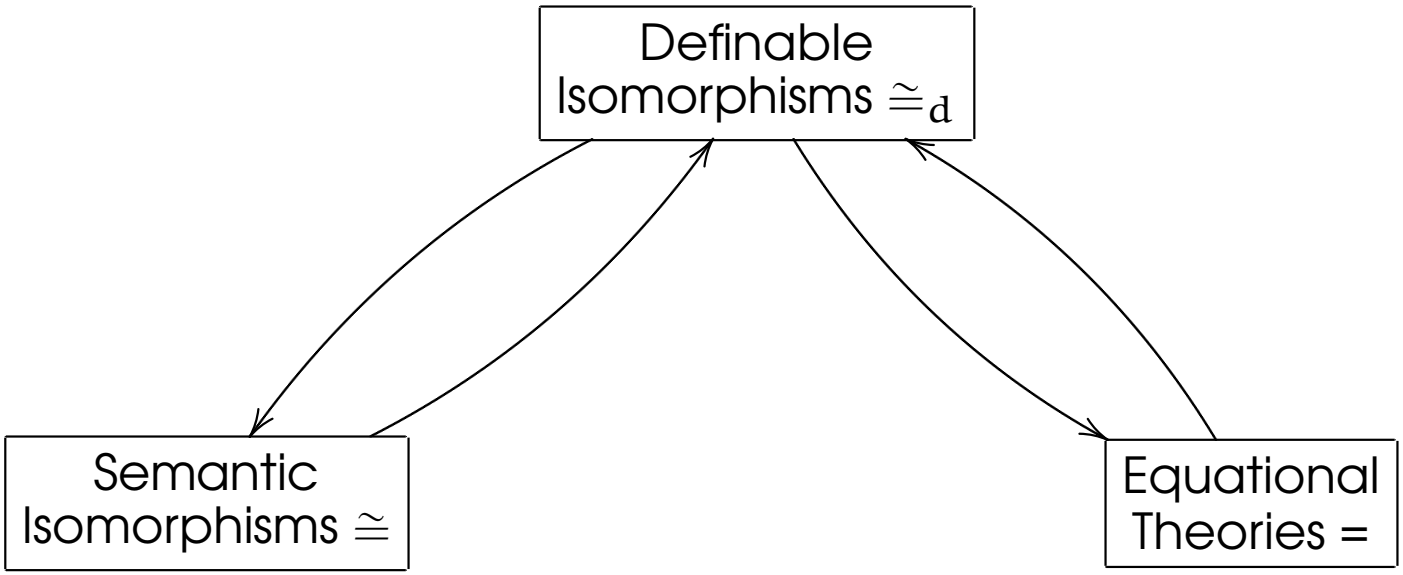
Example: Swap

Theorem ***(Completeness)***

$$\forall A, B \in \mathcal{T} : A \cong B \Rightarrow \mathit{Swap} \vdash A = B$$

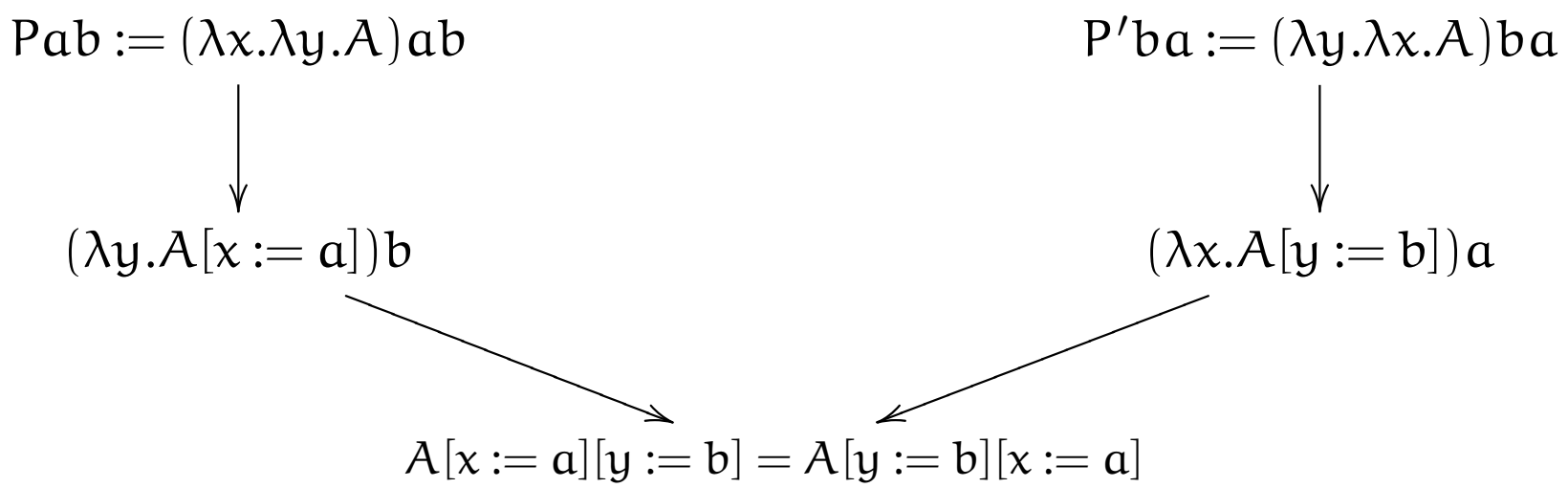
Proof It is enough to show $A \cong_d B \Rightarrow \mathit{Swap} \vdash A = B$, suppose that $M : A \cong_d B : N$ then proceed by structural induction on M, where M is a f.h.p.

We have...for Swap



Motivation:

Let P, P' be programs such



Motivation:

Let Q, Q' be programs such

$$Q(a,b)c := (\lambda z.\lambda x.\pi_1 zx)(a,b)c \qquad Q'abc := (\lambda u.\lambda v.\lambda x.ux)abc$$

$$\Downarrow \qquad\qquad\qquad \Downarrow$$

$$(\lambda x.\pi_1 zx[z := (a,b)])c \qquad (\lambda v.\lambda x.ux[u := a])bc$$

$$\Downarrow \qquad\qquad\qquad \Downarrow$$

$$(\pi_1(a,b)x[x := c]) \qquad (\lambda x.ux[v := b])c$$

$$\Downarrow \qquad\qquad\qquad \Downarrow$$

$$(ac) \qquad\qquad\qquad (ax[x := c])$$

$$\searrow \qquad\qquad \swarrow$$

$$ac = ac$$

Motivation:

What can we say about P, P' and Q, Q'?

- Consider $\mathit{Swap} : \mathrm{A} \to (\mathrm{B} \to \mathrm{C}) \to \mathrm{B} \to (\mathrm{A} \to \mathrm{C})$ as the following term

$$\mathit{Swap} := \lambda z.\lambda y.\lambda x.zxy$$

  it not hard to see that $\mathit{Swap}(\mathrm{P}) \equiv_\alpha \mathrm{P}'$

- Consider $\mathrm{Curry} : \mathrm{A} \to (\mathrm{B} \to \mathrm{C}) \to (\mathrm{A} \times \mathrm{B} \to \mathrm{C})$ as the following term

$$\mathrm{Curry} := \lambda z.\lambda x.\lambda y.z\langle x, y\rangle$$

  similarly, $\mathrm{Curry}(\mathrm{Q}) \equiv_\alpha \mathrm{Q}'$

Motivation:

Now, let's consider the following terms

$$Swap^* := \lambda y.\lambda x.\lambda z.zxy \qquad Curry^* := \lambda x.\lambda y.\lambda z.z\langle x, y\rangle$$

we will see that $\begin{cases} Swap^* baP \equiv_\alpha P'ba, \\ Curry^* abcQ \equiv_\alpha Q'abc. \end{cases}$

References:

Bruce K., Di Cosmo R., Longo G.
Provable Isomorphism of Types, Mathematical Structures in Computer Science, **2**, 231 - 247, 1991.

Dezani-Ciancaglini M.
Characterization of Normal Forms Possessing Inverse in the $\lambda_{\beta\eta}$-Calculus, Theoretical Computer Science, **2**, 323 - 337, 1976.

Di Cosmo R.
Isomorphism of Types: from λ-calculus to information retrieval and language design, Birkhäuser, 1995.

References:

Rittri M.
Retrieving Library Functions by Unifying Types Modulo Linear Type Isomorphism, Theoretical Informatics and Applications, **27**, 71 -89, 1993.

Soloviev S.
The Category of Finite Sets and Cartesian Closed Categories, Journal of Soviet Mathematics, **22(3)**, 1387 - 1400, 1983.

Zibin Y., Gil J., Considine J.
Efficient Algorithm for Isomorphism of Simple Types, Proceedings POPL'03 in ACM SIGPLAN, **38**, 160 - 171, 2003.

Tarski's high school algebra problem.

Let $\mathcal{E}$ be the equational theory given by:

$$AB = BA, \qquad (AB)C = A(BC),$$
$$C^{AB} = (C^B)^A, \qquad (AB)^C = A^C B^C$$
$$1A = A,\ A^1 = A \qquad 1^A = 1$$

$$A + B = B + A, \qquad (A + B) + C = A + (B + C),$$
$$A(B + C) = AB + AC, \qquad C^{A+B} = C^A C^B$$
$$0 + A = A, \qquad A0 = 0$$

Doner-Tarski conjecture: $\vdash_{\mathcal{E}} = \models_{\langle \mathbb{N}, 0, 1, +, *, \uparrow \rangle}$?

Tarski's high school algebra problem..

- **Martin'72**: $\mathsf{TI}_{\rightarrow}$ is equational complete for $\langle \mathbb{N}, \uparrow \rangle$, and decidable.

$$\boxed{\vdash_{\mathsf{TI}_{\rightarrow}} = \models_{\langle \mathbb{N}, \uparrow \rangle}}$$

- **Wilkie'81**: was the first to establish Tarski's conjecture is false.

$$\boxed{[\nvdash_{\mathcal{E}}, \models_{\langle \mathbb{N}, 0, 1, +, *, \uparrow \rangle}](A^x + B^x)^y (C^y + D^y)^x = (A^y + B^y)^x (C^x + D^x)^y}$$

  where $A = 1 + x$, $B = 1 + x + x^2$, $C = 1 + x^3$ and $D = 1 + x^2 + x^4$

- **Soloviev'83**: TI is equational complete for $\langle \mathbb{N}, 1, \times, \uparrow \rangle$, and decidable.

$$\boxed{\vdash_{\mathsf{TI}} = \models_{\langle \mathbb{N}, 1, \times, \uparrow \rangle}}$$

Tarski's high school algebra problem...

- **Gurevič'85**: found a 59-element model in which Tarski's identities are valid and Wilkie's identity is false
- **Gurevič'90:** $\models_{\langle \mathbb{N}, 1, +, *, \uparrow \rangle}$ is not finitely axiomatizable, introduce a infinite family of equations such that for every sound finite set of axioms one of the equation is not derivable from it.
- **Fiore, DiCosmo, Valat'02**: In the presence of arrow, empty, and sum types, type isomorphism and arithmetical equality no longer coincide.