The Representational Adequacy of HYBRID

Roy L. Crole

University of Leicester

2 December 2011

Seminar Outline

A Brief Summary of Logical Frameworks

The HYBRID Logical Framework

Translating λ-Expressions into HYBRID

A Model of HYBRID

The Adequacy of de Bruijn Expressions for λ-Expressions

The Adequacy of HYBRID for λ-Expressions

Simple Representation Results

The Rationale for a Logical Framework

- At its simplest, a logical framework is a logic/type theory with
  - tools for representing syntax and semantics;
  - principles for reasoning about syntax and semantics.
- A logical framework is usually thought of as a meta-language into which object languages are translated.

- Logical frameworks enjoy a rich history (too long to summarize here):
-
  - "A Framework for Defining Logics" by Honsell, Harper and Plotkin (1993) proposed use of dependent type theory.
  - One may also use a λ-calculus with constants: Higher Order Abstract Syntax - HOAS; Martin Löf's Theory of Arities and Expressions.

The Rationale for a Logical Framework

- At its simplest, a logical framework is a logic/type theory with
  - tools for representing syntax and semantics;
  - principles for reasoning about syntax and semantics.
- A logical framework is usually thought of as a meta-language into which object languages are translated.

- An object language is represented in a logical framework by giving a translation $\ulcorner - \urcorner : \mathcal{OL} \to \mathcal{LF}$.
- The translation function should be representationally adequate, ie:
  - injective
  - a compositional homomorphism, ie commute with (capture avoiding) substitution

The Rationale for a Logical Framework

- At its simplest, a logical framework is a logic/type theory with
  - tools for representing syntax and semantics;
  - principles for reasoning about syntax and semantics.
- A logical framework is usually thought of as a meta-language into which object languages are translated.

- An object language is represented in a logical framework by giving a translation $\ulcorner - \urcorner : \mathcal{OL} \to \mathcal{LF}$.
- The translation function should be representationally adequate, ie:
  - injective
  - a compositional homomorphism, ie commute with (capture avoiding) substitution

Talk focuses on object languages with binding

How to Implement Object Syntax with Binding

How might we implement object level syntax such as

$$Q ::= V_i \mid Q \supset Q \mid \forall V_i.\, Q \qquad QPL$$

A Traditional Approach

- Define a recursive type specifying the raw syntax

$$exp ::= var \mid \textbf{Imp}\ exp\ exp \mid \textbf{All}\ var\ exp$$

- Define capture avoiding substitution (for given notions of free and bound variables), and
- hence define language expressions to be the quotient of exp by the $\sim_\alpha$ equivalence relation.

How to Implement Object Syntax with Binding

How might we implement object level syntax such as

$$Q ::= V_i \mid Q \supset Q \mid \forall V_i.\, Q \qquad \textit{OL}$$

Disadvantages: The definitions are required for each object logic.

A Traditional Approach

- Define a recursive type speci... raw syntax

$$exp ::= \ldots\ exp\ exp \mid \textbf{All}\ var\ exp$$

- Define ca... ...ding substitution (for given notions of free and ... ...ables), and
- ...efine language expressions to be the quotient of exp by ... $\sim_\alpha$ equivalence relation.

How to Implement Object Syntax with Binding

How might we implement object level syntax such as

$$Q ::= V_i \mid Q \supset Q \mid \forall V_i.\, Q \qquad QPL$$

Or – Higher Order Abstract Syntax

- Implement the λ-calculus *once and only once*

$$C ::= c \mid v_k \mid \textbf{LAM}\, v_k.\, C \mid C_1\ C_2$$

- Define substitution and $\alpha(\beta\eta)$-equivalence *once and only once*.

How to Implement Object Syntax with Binding

How might we implement object level syntax such as

$$Q ::= V_i \mid Q \supset Q \mid \forall V_i.\, Q \qquad QPL$$

Or – Higher Order Abstract Syntax

- We get a *logical framework* infrastructure. To encode QPL specify constants **Imp** $:: exp \Rightarrow exp \Rightarrow exp$ and **All** $:: (exp \Rightarrow exp) \Rightarrow exp$
- One can define an *encoding* function $\ulcorner - \urcorner$, where

$$\begin{aligned} \ulcorner Q_1 \supset Q_2 \urcorner &\stackrel{\text{def}}{=} \textbf{Imp}\ \ulcorner Q_1 \urcorner \ulcorner Q_2 \urcorner \\ \ulcorner \forall V_i.\, Q \urcorner &\stackrel{\text{def}}{=} \textbf{All}\ (\textbf{LAM}\, v_i.\, \ulcorner Q \urcorner) \end{aligned}$$

Human or Machine?

I prefer $\mathcal{LF}$ to have binders with names:

$$C ::= c \mid v_i \mid \textsf{LAM}\, v_i.\, C \mid C_1\ C_2$$

Human or Machine?

I prefer $\mathcal{LF}$ to have binders with names:

$$C ::= c \mid v_i \mid \textsf{LAM}\, v_i.\, C \mid C_1\ C_2$$

However, if I was a machine I'd prefer de Bruijn expressions

$$C ::= \textsf{CON}\ n \mid \textsf{VAR}\ i \mid \textsf{BND}\ j \mid \textsf{ABS}\ C \mid C_1\ \$\$\ C_2$$

We use locally nameless de Bruijn expressions: **BND** j is a bound variable with index j; but **VAR** i is a named variable, with name i.

Human or Machine?

I prefer $\mathcal{LF}$ to have binders with names:

$$C ::= c \mid v_i \mid \textbf{LAM}\, v_i.\, C \mid C_1\ C_2$$

However, if I was a machine I'd prefer de Bruijn expressions

$$C ::= \textbf{CON}\ n \mid \textbf{VAR}\ i \mid \textbf{BND}\ j \mid \textbf{ABS}\ C \mid C_1\ \$\$\ C_2$$

We use locally nameless de Bruijn expressions: **BND** j is a bound variable with index j; but **VAR** i is a named variable, with name i.

HYBRID gives us the best of both worlds ...

Introducing HYBRID

- HYBRID is a theory in Isabelle/HOL.
- There is a "λ-calculus datatype" which specifies a form of HOAS.
- HYBRID is a logical framework, in which both HOAS and (co)induction are consistent ... but that is another story.
- Object level variable binding is represented by Isabelle/HOL's internal meta-variable binding.

Introducing HYBRID

- HYBRID is a theory in Isabelle/HOL.
- There is a "λ-calculus datatype" which specifies a form of HOAS.
- HYBRID is a logical framework, in which both HOAS and (co)induction are consistent . . . but that is another story.
- Object level variable binding is represented by Isabelle/HOL's internal meta-variable binding.

HYBRID can convert λ-expressions into de Bruijn expressions

Introducing HYBRID

- HYBRID is a theory in Isabelle/HOL.
- There is a "λ-calculus datatype" which specifies a form of HOAS.
- HYBRID is a logical framework, in which both HOAS and (co)induction are consistent ... but that is another story.
- Object level variable binding is represented by Isabelle/HOL's internal meta-variable binding.

HYBRID can convert λ-expressions into de Bruijn expressions

user supplied

machine produced

The Heart of HYBRID

- HYBRID includes a datatype exp of de Bruijn expressions:

$$\begin{aligned} exp \;\; ::= \;\; & \textbf{CON}\ con \mid \textbf{VAR}\ var \mid \textbf{BND}\ bnd \\ & \mid \textbf{ABS}\ exp \mid exp \ \$\$\ exp \end{aligned}$$

- But a user can write expressions of the form

$$\begin{aligned} C \;\; ::= \;\; & \textbf{CON}\ v \mid \textbf{VAR}\ i \mid \textbf{BND}\ j \\ & \mid \textbf{ABS}\ C \mid C \ \$\$\ C \mid \textbf{LAM}\, v.\, C \end{aligned}$$

The Heart of Hybrid

- Hybrid includes a datatype *exp* of de Bruijn expressions:

$$exp \;::=\; \textbf{CON}\ con \mid \textbf{VAR}\ var \mid \textbf{BND}\ bnd \mid \textbf{ABS}\ exp \mid exp\ \$\$\ exp$$

- But a user can write expressions of the form

$$C \;::=\; \textbf{CON}\ v \mid \textbf{VAR}\ i \mid \textbf{BND}\ j \mid \textbf{ABS}\ C \mid C\ \$\$\ C \mid \textbf{LAM}\ v.\ C$$

Hybrid is a hybrid of λ-calculus and de Bruijn notation

- User *inputs* HYBRID expression

$$\mathbf{LAM}\, v_1.\, (\mathbf{LAM}\, v_0.\, (v_1 \,\$\$\, v_0)) \qquad \dagger$$

where $\mathbf{LAM}\, v_i.\, \xi$ is Isabelle/HOL *binder* syntax

- $\dagger$ can be automatically proved equal to a HYBRID expression

$$\mathbf{ABS}\ (\mathbf{ABS}\ (\mathbf{BND}\ 1 \,\$\$\, \mathbf{BND}\ 0)) :: exp$$

- This is implemented by a function *lbnd*

$$\mathbf{LAM}\, v_i.\, \xi \mapsto \mathbf{ABS}\ (lbnd\ 0\ \Lambda\, v_i.\, \xi)$$

- User *inputs* HYBRID expression

$$\mathbf{LAM}\, v_1.\, (\mathbf{LAM}\, v_0.\, (v_1 \;\$\$\; v_0)) \qquad \dagger$$

where $\mathbf{LAM}\, v_i.\, \xi$ is Isabelle/HOL *binder* syntax

- $\dagger$ can be automatically proved equal to a HYBRID expression

$$\mathbf{ABS}\; (\mathbf{ABS}\; (\mathbf{BND}\; 1 \;\$\$\; \mathbf{BND}\; 0)) :: exp$$

- This is implemented by a function *lbnd*

$$\mathbf{LAM}\, v_i.\, \xi \mapsto \mathbf{ABS}\; (lbnd\; 0\; \Lambda\, v_i.\, \xi)$$

We need to define *lbnd*

HOL meta-binding $\Lambda.\, v_i$

Translating λ-Expressions

An example of $\ulcorner - \urcorner : \mathcal{LE} \to$ HYBRID:

$\boxed{\mathcal{LF}}$

$\boxed{\mathcal{OL}}$

Let $E_O = \lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3$. Then

$$E_H \stackrel{\text{def}}{=} \ulcorner E_O \urcorner \stackrel{\text{def}}{=} \mathbf{LAM}\, v_8.\, (\mathbf{LAM}\, v_2.\, (v_8\ \$\$\ \mathbf{VAR}\ 3))$$

In HYBRID E_H is provably equal to

$$\mathbf{ABS}\ (\mathbf{ABS}\ (\mathbf{BND}\ 1\ \$\$\ \mathbf{VAR}\ 3))$$

Key Translation Principles

- object level free variables v_i are expressed as Hybrid expressions of the form **VAR** i;
- object level bound variables v_j are expressed as Hybrid (bound) meta-variables v_j;
- object level abstractions $\lambda\, v_j.\, E$ are expressed as Hybrid expressions **LAM** $v_j.\, C$; and
- object level applications $E_1\ E_2$ are expressed as Hybrid expressions C_1 \$\$ C_2.

Main theorem: a proof that the translation function Θ , derived from these principles,

$$\Theta\ :\ \text{(object level) } \lambda\text{-expressions} \longrightarrow \textsc{Hybrid},$$

exists and is representationally adequate.

Defining **LAM** Via *lbnd*

Consider the (object level) expression $E_O \stackrel{\text{def}}{=} \lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_2$.

This expression is encoded in HYBRID as

$$E_H \stackrel{\text{def}}{=} \textsf{LAM}\, v_8.\, (\textsf{LAM}\, v_2.\, (v_8 \;\$\$\; v_2))$$

Thought 1: $\textsf{LAM}\, v_i.\, \xi$ denotes $\textsf{ABS}\ (\Lambda\, v_i.\, \xi)$. Then E_H would be

$$\textsf{ABS}\ (\Lambda\, v_8.\, (\textsf{ABS}\ (\Lambda\, v_2.\, (v_8 \;\$\$\; v_2))))$$

Defining **LAM** Via *lbnd*

Consider the (object level) expression $E_O \stackrel{\text{def}}{=} \lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_2$.

This expression is encoded in HYBRID as

$$E_H \stackrel{\text{def}}{=} \mathsf{LAM}\, v_8.\, (\mathsf{LAM}\, v_2.\, (v_8 \ \$\$\ v_2))$$

Thought 1: $\mathsf{LAM}\, v_i.\, \xi$ denotes $\mathsf{ABS}\ (\Lambda\, v_i.\, \xi)$. Then E_H would be

$$\mathsf{ABS}\ (\Lambda\, v_8.\, (\mathsf{ABS}\ (\Lambda\, v_2.\, (v_8 \ \$\$\ v_2))))$$

E_H should equal

$$\mathsf{ABS}\ (\Lambda\, v_8.\, (\mathsf{ABS}\ (\Lambda\, v_2.\, (\mathsf{BND}\ 1 \ \$\$\ \mathsf{BND}\ 0))))$$

but with the "meta binders and variables deleted".

Defining **LAM** Via *lbnd*

Consider the (object level) expression $E_O \stackrel{\textbf{def}}{=} \lambda\, v_8.\, \lambda\, v_2.\, v_8\; v_2$.

This expression is encoded in HYBRID as

$$E_H \stackrel{\textbf{def}}{=} \textsf{LAM}\, v_8.\, (\textsf{LAM}\, v_2.\, (v_8 \;\$\$\; v_2))$$

Thought 2: $\textsf{LAM}\, v_i.\, \xi$ denotes $\textsf{ABS}\ (lbnd_0(\Lambda\, v_i.\, \xi))$ and where (hopefully!)

$$\begin{aligned} &\textsf{LAM}\, v_8.\, (\textsf{LAM}\, v_2.\, (v_8 \;\$\$\; v_2)) \\ &\quad = \textsf{ABS}\ (lbnd_0(\Lambda\, v_8.\, \textsf{ABS}\ (lbnd_0(\Lambda\, v_2.\, v_8 \;\$\$\; v_2)))) \\ &\quad = \vdots \\ &\quad = \textsf{ABS}\ (\textsf{BND}\ 1 \;\$\$\; \textsf{BND}\ 0) \end{aligned}$$

We try to define $lbnd_n$

- recurse through the **ABS** nodes and use n to count them—in order to compute the bound de Bruijn indices;
- recurse over **$$** nodes;
- and in each case recursively move the meta-binders Λ towards the bound meta-variables.

$$
\begin{aligned}
& lbnd_0(\Lambda\, v_i.\,\textbf{ABS}\ (C[v_i, v_j])) \\
& \qquad = \qquad \textbf{ABS}\ (lbnd_1(\Lambda\, v_i.\, C[v_i, v_j])) \\
& \qquad \vdots \\
& \qquad = \qquad \textbf{ABS}\ (C[lbnd_{n_1}(\Lambda\, v_i.\, v_i), lbnd_{n_2}(\Lambda\, v_i.\, v_j)])
\end{aligned}
$$

We try to define $lbnd_n$

- recurse through the **ABS** nodes and use n to count them—in order to compute the bound de Bruijn indices;
- recurse over **\$\$** nodes;
- and in each case recursively move the meta-binders Λ towards the bound meta-variables.

$$\boxed{\mathbf{LAM}\, v_i.\, \xi \stackrel{\text{def}}{=} \mathbf{ABS}(lbnd_0(\Lambda\, v_i.\, \xi))}$$

$$\begin{aligned}
&\mathbf{ABS}(\\
&lbnd_0(\Lambda\, v_i.\, \mathbf{ABS}\ (C[v_i, v_j])))\\
&\qquad = \mathbf{ABS}(\mathbf{ABS}\ (lbnd_1(\Lambda\, v_i.\, C[v_i, v_j])))\\
&\qquad \vdots\\
&\qquad = \mathbf{ABS}(\mathbf{ABS}\ (C[lbnd_{n_1}(\Lambda\, v_i.\, v_i), lbnd_{n_2}(\Lambda\, v_i.\, v_j)]))
\end{aligned}$$

We try to define $lbnd_n$

- recurse through the **ABS** nodes and use n to count them—in order to compute the bound de Bruijn indices;
- recurse over **$$** nodes;
- and in each case recursively move the meta-binders Λ towards the bound meta-variables.

$$\boxed{\mathsf{LAM}\, v_i.\, \xi \stackrel{\text{def}}{=} \mathsf{ABS}(lbnd_0(\Lambda\, v_i.\, \xi))}$$

$$
\begin{aligned}
&\mathsf{ABS}(\\
&lbnd_0(\Lambda\, v_i.\, \mathsf{ABS}\ (C[v_i, v_j])))\\
&\qquad = \mathsf{ABS}(\mathsf{ABS}\ (lbnd_1(\Lambda\, v_i.\, C[v_i, v_j])))\\
&\qquad \vdots\\
&\qquad = \mathsf{ABS}(\mathsf{ABS}\ (C[lbnd_{n_1}(\Lambda\, v_i.\, v_i), lbnd_{n_2}(\Lambda\, v_i.\, v_j)]))\\
&\qquad = \mathsf{ABS}(\mathsf{ABS}\ (C[\mathsf{BND}\ n_1, lbnd_{n_2}(\Lambda\, v_i.\, v_j)]))
\end{aligned}
$$

We try to define $lbnd_n$

- recurse through the **ABS** nodes and use n to count them—in order to compute the bound de Bruijn indices;
- recurse over **\$\$** nodes;
- and in each case recursively move the meta-binders Λ towards the bound meta-variables.

$$\boxed{\mathsf{LAM}\, v_i.\, \xi \stackrel{\text{def}}{=} \mathsf{ABS}(lbnd_0(\Lambda\, v_i.\, \xi))}$$

$$\begin{aligned}
&\mathsf{ABS}(\\
&lbnd_0(\Lambda\, v_i.\, \mathsf{ABS}\ (C[v_i, v_j])))\\
&\qquad = \mathsf{ABS}(\mathsf{ABS}\ (lbnd_1(\Lambda\, v_i.\, C[v_i, v_j])))\\
&\qquad\ \vdots\\
&\qquad = \mathsf{ABS}(\mathsf{ABS}\ (C[lbnd_{n_1}(\Lambda\, v_i.\, v_i), lbnd_{n_2}(\Lambda\, v_i.\, v_j)]))\\
&\qquad = \mathsf{ABS}(\mathsf{ABS}\ (C[\mathsf{BND}\ n_1, lbnd_{n_2}(\Lambda\, v_i.\, v_j)]))\\
&\qquad = \mathsf{ABS}(\mathsf{ABS}\ (C[\mathsf{BND}\ n_1, v_j]))
\end{aligned}$$

Example Calculation of **LAM** Via *lbnd*

$\mathsf{LAM}\, v_i.\, \xi \stackrel{\mathrm{def}}{=} \mathsf{ABS}\ (lbnd_0(\Lambda\, v_i.\, \xi))$ and hence

$$\mathsf{LAM}\, v_8.\, (\mathsf{LAM}\, v_2.\, (v_8\ \$\$\ v_2))$$

$$= \mathsf{ABS}\ (lbnd_0(\Lambda\, v_8.\, \mathsf{ABS}\ (lbnd_0(\Lambda\, v_2.\, v_8\ \$\$\ v_2))))$$

Example Calculation of **LAM** Via *lbnd*

$\mathsf{LAM}\, v_i.\, \xi \stackrel{\mathbf{def}}{=} \mathsf{ABS}\ (\mathit{lbnd}_0(\Lambda\, v_i.\, \xi))$ and hence

$$\mathsf{LAM}\, v_8.\, (\mathsf{LAM}\, v_2.\, (v_8\ \$\$\ v_2))$$

$$= \mathsf{ABS}\ (\mathit{lbnd}_0(\Lambda\, v_8.\, \mathsf{ABS}\ (\mathit{lbnd}_0(\Lambda\, v_2.\, v_8\ \$\$\ v_2))))$$

$$= \mathsf{ABS}\ (\mathit{lbnd}_0(\Lambda\, v_8.\, \mathsf{ABS}\ (\mathit{lbnd}_0(\Lambda\, v_2.\, v_8)\ \$\$\ \mathit{lbnd}_0(\Lambda\, v_2.\, v_2))))$$

Example Calculation of **LAM** Via *lbnd*

$\mathsf{LAM}\, v_i.\,\xi \stackrel{\mathbf{def}}{=} \mathsf{ABS}\,(lbnd_0(\Lambda\, v_i.\,\xi))$ and hence

$\mathsf{LAM}\, v_8.\,(\mathsf{LAM}\, v_2.\,(v_8 \;\$\$\; v_2))$

$= \mathsf{ABS}\,(lbnd_0(\Lambda\, v_8.\,\mathsf{ABS}\,(lbnd_0(\Lambda\, v_2.\, v_8 \;\$\$\; v_2))))$

$= \mathsf{ABS}\,(lbnd_0(\Lambda\, v_8.\,\mathsf{ABS}\,(lbnd_0(\Lambda\, v_2.\, v_8) \;\$\$\; lbnd_0(\Lambda\, v_2.\, v_2))))$

$= \mathsf{ABS}\,(lbnd_0(\Lambda\, v_8.\,\mathsf{ABS}\,(v_8 \;\$\$\; \mathsf{BND}\; 0)))$

Example Calculation of **LAM** Via *lbnd*

$\mathsf{LAM}\, v_i .\, \xi \stackrel{\text{def}}{=} \mathsf{ABS}\ (lbnd_0(\Lambda\, v_i .\, \xi))$ and hence

$$\mathsf{LAM}\, v_8 .\, (\mathsf{LAM}\, v_2 .\, (v_8\ \$\$\ v_2))$$

$$= \mathsf{ABS}\ (lbnd_0(\Lambda\, v_8 .\, \mathsf{ABS}\ (lbnd_0(\Lambda\, v_2 .\, v_8\ \$\$\ v_2))))$$

$$= \mathsf{ABS}\ (lbnd_0(\Lambda\, v_8 .\, \mathsf{ABS}\ (lbnd_0(\Lambda\, v_2 .\, v_8)\ \$\$\ lbnd_0(\Lambda\, v_2 .\, v_2))))$$

$$= \mathsf{ABS}\ (lbnd_0(\Lambda\, v_8 .\, \mathsf{ABS}\ (v_8\ \$\$\ \mathsf{BND}\ 0)))$$

$$= \mathsf{ABS}\ (\mathsf{ABS}\ (lbnd_1(\Lambda\, v_8 .\, v_8)\ \$\$\ lbnd_1(\Lambda\, v_8 .\, (\mathsf{BND}\ 0))))$$

Example Calculation of **LAM** Via *lbnd*

$\mathsf{LAM}\, v_i.\, \xi \stackrel{\text{def}}{=} \mathsf{ABS}\ (lbnd_0(\Lambda\, v_i.\, \xi))$ and hence

$$\mathsf{LAM}\, v_8.\, (\mathsf{LAM}\, v_2.\, (v_8 \ \$\$\ v_2))$$

$$= \mathsf{ABS}\ (lbnd_0(\Lambda\, v_8.\, \mathsf{ABS}\ (lbnd_0(\Lambda\, v_2.\, v_8 \ \$\$\ v_2))))$$

$$= \mathsf{ABS}\ (lbnd_0(\Lambda\, v_8.\, \mathsf{ABS}\ (lbnd_0(\Lambda\, v_2.\, v_8) \ \$\$\ lbnd_0(\Lambda\, v_2.\, v_2))))$$

$$= \mathsf{ABS}\ (lbnd_0(\Lambda\, v_8.\, \mathsf{ABS}\ (v_8 \ \$\$\ \mathsf{BND}\ 0)))$$

$$= \mathsf{ABS}\ (\mathsf{ABS}\ (lbnd_1(\Lambda\, v_8.\, v_8) \ \$\$\ lbnd_1(\Lambda\, v_8.\, (\mathsf{BND}\ 0))))$$

$$= \mathsf{ABS}\ (\mathsf{ABS}\ (\mathsf{BND}\ 1 \ \$\$\ \mathsf{BND}\ 0))$$

A Mathematical Model of Hybrid

$$\boldsymbol{\Theta} \; : \; \text{(object level) } \lambda\text{-expressions} \longrightarrow \textsc{Hybrid}$$

Task: formally define $\boldsymbol{\Theta}$ and prove it representationally adequate.

A Mathematical Model of Hybrid

$$\Theta : \mathcal{LE}/\!\sim_\alpha \longrightarrow \textsc{Hybrid}$$

Task: formally define Θ and prove it representationally adequate.

- We take the object expressions to be $\mathcal{LE}/\!\sim_\alpha$.

A Mathematical Model of HYBRID

$$\Theta : \mathcal{LE}/\!\sim_{\alpha} \longrightarrow \text{HYBRID}$$

Task: formally define Θ and prove it representationally adequate.

- We take the object expressions to be $\mathcal{LE}/\!\sim_{\alpha}$.
- We take HYBRID to be a model of a subset of the system implemented in Isabelle/HOL.

A Mathematical Model of HYBRID

$$\Theta : \mathcal{LE}/\sim_\alpha \longrightarrow \textsc{Hybrid}$$

Task: formally define Θ and prove it representationally adequate.

- We take the object expressions to be $\mathcal{LE}/\sim_\alpha$.
- We take HYBRID to be a model of a subset of the system implemented in Isabelle/HOL.
- **!! Our model is a theory in a logical framework !!**:

A Mathematical Model of Hybrid

$$\Theta : \mathcal{LE}/{\sim_\alpha} \longrightarrow \textsc{Hybrid}$$

Task: formally define Θ and prove it representationally adequate.

- We take the object expressions to be $\mathcal{LE}/{\sim_\alpha}$.
- We take Hybrid to be a model of a subset of the system implemented in Isabelle/HOL.
- **!! Our model is a theory in a logical framework !!**:
- The meta-variables of the logical framework play the rôle of Isabelle/HOL meta-variables of implemented Hybrid; and

A Mathematical Model of HYBRID

$$\Theta : \mathcal{LE}/{\sim_\alpha} \longrightarrow \text{HYBRID}$$

Task: formally define Θ and prove it representationally adequate.

- We take the object expressions to be $\mathcal{LE}/{\sim_\alpha}$.
- We take HYBRID to be a model of a subset of the system implemented in Isabelle/HOL.
- **!! Our model is a theory in a logical framework !!**:
- The meta-variables of the logical framework play the rôle of Isabelle/HOL meta-variables of implemented HYBRID; and
- logical framework binding and application play the rôle of Isabelle/HOL meta-binding and meta-application respectively.

HYBRID Types and Canonical Expressions

- The types are

$$\sigma ::= exp \mid con \mid var \mid bnd \mid \sigma \Rightarrow \sigma$$

- The constants are

$$\begin{array}{rcl@{\quad}rcl}
N & :: & con & \textbf{CON} & :: & con \Rightarrow exp \\
i & :: & var & \textbf{VAR} & :: & var \Rightarrow exp \\
j & :: & bnd & \textbf{BND} & :: & bnd \Rightarrow exp \\
\$\$ & :: & exp \Rightarrow exp \Rightarrow exp & \textbf{ABS} & :: & exp \Rightarrow exp
\end{array}$$

- The inductive definition of canonical forms C is standard ...

HYBRID Types and Canonical Expressions

$$\frac{\Gamma(v_k) = \sigma_1 \Rightarrow \sigma_2 \Rightarrow \ldots \sigma_n \Rightarrow \gamma \qquad \Gamma \vdash_{can} C_i :: \sigma_i \;\; {\scriptstyle (0 \leq i \leq n)}}{\Gamma \vdash_{can} v_k\, \vec{C} :: \gamma}$$

$$\frac{\kappa :: \sigma_1 \Rightarrow \sigma_2 \Rightarrow \ldots \sigma_n \Rightarrow \gamma \qquad \Gamma \vdash_{can} C_i :: \sigma_i \;\; {\scriptstyle (0 \leq i \leq n)}}{\Gamma \vdash_{can} \kappa\, \vec{C} :: \gamma}$$

$$\frac{\Gamma, v_k :: \sigma \vdash_{can} C :: \sigma'}{\Gamma \vdash_{can} \Lambda\, v_k.\, C :: \sigma \Rightarrow \sigma'}$$

Examples:

$$\mathsf{BND}\ 0 \qquad \Lambda\, v_k.\, \mathsf{BND}\ 0 \qquad \mathsf{ABS}\ C$$

HYBRID Types and Canonical Expressions

$$\frac{\Gamma(v_k) = \sigma_1 \Rightarrow \sigma_2 \Rightarrow \ldots \sigma_n \Rightarrow \gamma \qquad \Gamma \vdash_{can} C_i :: \sigma_i \ {}_{(0 \leq i \leq n)}}{\Gamma \vdash_{can} v_k\ \vec{C} :: \gamma}$$

$$\frac{\kappa :: \sigma_1 \Rightarrow \sigma_2 \Rightarrow \ldots \sigma_n \Rightarrow \gamma \qquad \Gamma \vdash_{can} C_i :: \sigma_i \ {}_{(0 \leq i \leq n)}}{\Gamma \vdash_{can} \kappa\ \vec{C} :: \gamma}$$

$$\frac{\Gamma, v_k :: \sigma \vdash_{can} C :: \sigma'}{\Gamma \vdash_{can} \Lambda\, v_k. C :: \sigma \Rightarrow \sigma'}$$

Examples:

$$C_1 \ \$\$\ C_2 \qquad \Lambda\, v_k. v_k \ \$\$\ \textbf{VAR}\ 3 \qquad \textbf{ABS}\ (\textbf{BND}\ 0\ \$\$\ v_4)$$

and **LAM** v_4. **ABS** $(\textbf{BND}\ 0\ \$\$\ v_4)$ is equal to a canonical expression.

HYBRID Types and Canonical Expressions

$$\frac{\Gamma(v_k) = \sigma_1 \Rightarrow \sigma_2 \Rightarrow \ldots \sigma_n \Rightarrow \gamma \qquad \Gamma \vdash_{can} C_i :: \sigma_i \;\; {\scriptstyle (0 \leq i \leq n)}}{\Gamma \vdash_{can} v_k\, \vec{C} :: \gamma}$$

$$\frac{\kappa :: \sigma_1 \Rightarrow \sigma_2 \Rightarrow \ldots \sigma_n \Rightarrow \gamma \qquad \Gamma \vdash_{can} C_i :: \sigma_i \;\; {\scriptstyle (0 \leq i \leq n)}}{\Gamma \vdash_{can} \kappa\, \vec{C} :: \gamma}$$

$$\frac{\Gamma, v_k :: \sigma \vdash_{can} C :: \sigma'}{\Gamma \vdash_{can} \Lambda\, v_k.\, C :: \sigma \Rightarrow \sigma'}$$

We shall define:

$$\mathcal{CLF}_\sigma(\Gamma) \stackrel{\mathbf{def}}{=} \{C \mid \underbrace{v_{i_1} :: \sigma_1, \ldots, v_{i_m} :: \sigma_m}_{\Gamma} \vdash_{can} C :: \sigma\}$$

Formally Defining *lbnd*

If $\Gamma^L_{exp} = v_{k_1} :: exp, \ldots, v_{k_m} :: exp$ there is a unique function

$$lbnd\ n : \mathcal{CLF}_{exp \Rightarrow exp}(\Gamma^L_{exp}) \to \mathcal{CLF}_{exp}(\Gamma^L_{exp})$$

which satisfies the following recursion equations

- $lbnd\ n\ (\Lambda\, v_k.\, v_k) = \textsf{BND}\ n$
- $lbnd\ n\ (\Lambda\, v_k.\, v_{k'}) = v_{k'}$ where $k \neq k'$
- $lbnd\ n\ (\Lambda\, v_k.\, \textsf{VAR}\ i) = \textsf{VAR}\ i$
- $lbnd\ n\ (\Lambda\, v_k.\, \textsf{BND}\ j) = \textsf{BND}\ j$
- $lbnd\ n\ (\Lambda\, v_k.\, C_1\ \$\$\ C_2) = (lbnd\ n\ (\Lambda\, v_k.\, C_1))\ \$\$\ (lbnd\ n\ (\Lambda\, v_k.\, C_2))$
- $lbnd\ n\ (\Lambda\, v_k.\, \textsf{ABS}\ C) = \textsf{ABS}\ (lbnd\ (n+1)\ (\Lambda\, v_k.\, C))$

de Bruijn Representational Adequacy

Let $\mathcal{DB}(l)$ be de Bruijn expressions of level l.

Let $L = v_{k_1}, \ldots, v_{k_m}$. Then there is a function

$$\theta_L : \mathcal{LE}/{\sim_\alpha} \to \mathcal{DB}(|L|)$$

where, for example,

$\theta_\epsilon[\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha$

de Bruijn Representational Adequacy

Let $\mathcal{DB}(l)$ be de Bruijn expressions of level l.

Let $L = v_{k_1}, \ldots, v_{k_m}$. Then there is a function

$$\theta_L : \mathcal{LE}/{\sim_\alpha} \to \mathcal{DB}(|L|)$$

where, for example,

$$\theta_\epsilon[\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha$$

$$= \textbf{ABS}\ (\textbf{ABS}\ \theta_{[v_2, v_8]}([v_8]_\alpha\ [v_3]_\alpha))$$

de Bruijn Representational Adequacy

Let $\mathcal{DB}(l)$ be de Bruijn expressions of level l.

Let $L = v_{k_1}, \ldots, v_{k_m}$. Then there is a function

$$\theta_L : \mathcal{LE}/\!\sim_\alpha \rightarrow \mathcal{DB}(|L|)$$

where, for example,

$$\begin{aligned}
&\theta_\epsilon[\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha \\
&\qquad = \textbf{ABS}\ (\textbf{ABS}\ \theta_{[v_2, v_8]}([v_8]_\alpha\ [v_3]_\alpha)) \\
&\qquad = \textbf{ABS}\ (\textbf{ABS}\ (\textbf{BND}\ (pos\, v_8\, [v_2, v_8])\ \$\$\ \textbf{VAR}\ 3))
\end{aligned}$$

de Bruijn Representational Adequacy

Let $\mathcal{DB}(l)$ be de Bruijn expressions of level l.

Let $L = v_{k_1}, \ldots, v_{k_m}$. Then there is a function

$$\theta_L : \mathcal{LE}/\!\sim_\alpha \to \mathcal{DB}(|L|)$$

where, for example,

$$\begin{aligned}
&\theta_\epsilon[\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha \\
&\quad = \textsf{ABS}\ (\textsf{ABS}\ \theta_{[v_2,v_8]}([v_8]_\alpha\ [v_3]_\alpha)) \\
&\quad = \textsf{ABS}\ (\textsf{ABS}\ (\textsf{BND}\ (pos\, v_8\, [v_2, v_8])\ \$\$\ \textsf{VAR}\ 3)) \\
&\quad = \textsf{ABS}\ (\textsf{ABS}\ (\textsf{BND}\ 1\ \$\$\ \textsf{VAR}\ 3))
\end{aligned}$$

de Bruijn Representational Adequacy

Theorem

There is a function

$$\theta : \mathcal{LE}/{\sim_\alpha} \to \mathcal{DB}(0) \subseteq \mathcal{DB}$$

which is representationally adequate, that is to say θ *is a compositional isomorphism* $\mathcal{LE}/{\sim_\alpha} \cong \mathcal{DB}(0)$.

de Bruijn Representational Adequacy

Theorem

There is a function

$$\theta : \mathcal{LE}/\!\sim_\alpha \to \mathcal{DB}(0) \subseteq \mathcal{DB}$$

which is representationally adequate, that is to say θ is a compositional isomorphism $\mathcal{LE}/\!\sim_\alpha \cong \mathcal{DB}(0)$.

Equivalently

- B θ is bijective; and
- CH θ is a compositional homomorphism

$$\theta([E]_\alpha[[E']_\alpha/v_k]) = \theta([E]_\alpha)[\theta([E']_\alpha)/\mathsf{VAR}\ k]$$

de Bruijn Representational Adequacy

Theorem

There is a function

$$\theta : \mathcal{LE}/{\sim_\alpha} \to \mathcal{DB}(0) \subseteq \mathcal{DB}$$

which is representationally adequate, that is to say θ is a compositional isomorphism $\mathcal{LE}/{\sim_\alpha} \cong \mathcal{DB}(0)$.

- Shankar (1988) gave a mechanical proof for pure de Bruijn.
- Crole (MSCS, 2011) is the first detailed proof for locally nameless de Bruijn.

Approaching Hybrid Adequacy

There is a well-defined function

$$\Theta_\epsilon : \mathcal{LE}/\sim_\alpha \to \Theta_\epsilon\,(\mathcal{LE}/\sim_\alpha) \subseteq \mathcal{CLF}_{exp}(\epsilon)$$

arising from the family of unique well-defined functions

$$\Theta_L : \mathcal{LE}/\sim_\alpha \to \mathcal{CLF}_{exp}(\Gamma^L_{exp})$$

satisfying the recursion equations

$$\Theta_L\,([E_1\ E_2]_\alpha) \stackrel{\mathbf{def}}{=} (\Theta_L\,[E_1]_\alpha)\ \$\$\ (\Theta_L\,[E_2]_\alpha)$$

Approaching HYBRID Adequacy

There is a well-defined function

$$\Theta_\epsilon \; : \; \mathcal{LE}/\!\sim_\alpha \; \rightarrow \Theta_\epsilon \, (\mathcal{LE}/\!\sim_\alpha) \subseteq \mathcal{CLF}_{exp}(\epsilon)$$

arising from the family of unique well-defined functions

$$\Theta_L \; : \; \mathcal{LE}/\!\sim_\alpha \; \rightarrow \mathcal{CLF}_{exp}(\Gamma^L_{exp})$$

satisfying the recursion equations

$$\Theta_L \, ([\lambda \, v_i . \, E]_\alpha) \stackrel{\text{def}}{=} \textsf{LAM} \, v_i . \, \Theta_{v_i, L} \, ([E]_\alpha)$$

where we write $\textsf{LAM} \, v_i . \, \xi$ as an abbreviation for $\textsf{ABS} \; (lbnd \; \mathbf{0} \; (\Lambda \, v_i . \, \xi))$.

Approaching HYBRID Adequacy

There is a well-defined function

$$\Theta_\epsilon : \mathcal{LE}/{\sim_\alpha} \rightarrow \Theta_\epsilon \left(\mathcal{LE}/{\sim_\alpha}\right) \subseteq \mathcal{CLF}_{exp}(\epsilon)$$

arising from the family of unique well-defined functions

$$\Theta_L : \mathcal{LE}/{\sim_\alpha} \rightarrow \mathcal{CLF}_{exp}(\Gamma^L_{exp})$$

satisfying the recursion equations

$$\Theta_L\left([v_i]_\alpha\right) \stackrel{\mathbf{def}}{=} \begin{cases} v_i \text{ if } v_i \in L \\ \textbf{VAR}\ i \text{ if } v_i \notin L \end{cases}$$

Hybrid Adequacy

Theorem

The function

$$\Theta_\epsilon : \mathcal{LE}/{\sim_\alpha} \to \Theta_\epsilon\,(\mathcal{LE}/{\sim_\alpha}) \subseteq \mathcal{CLF}_{exp}(\epsilon)$$

is representationally adequate, that is

- B *it is bijective (onto its image); and*
- CH *it is a compositional homomorphism which means that*

$$\Theta_\epsilon\,([E]_\alpha[[E']_\alpha/v_k]) = \Theta_\epsilon\,([E]_\alpha)[\Theta_\epsilon\,([E']_\alpha)/\textbf{VAR}\ k]$$

Example Calculation of Θ

$$\begin{aligned}
&\Theta_\epsilon\ [\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha \\
&\quad = \ \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, \Theta_{[v_2, v_8]}([v_8]_\alpha\ [v_3]_\alpha)
\end{aligned}$$

Example Calculation of Θ

$$\Theta_{\epsilon}\,[\lambda\, v_8.\,\lambda\, v_2.\, v_8\ v_3]_{\alpha}$$

$$= \ \mathsf{LAM}\, v_8.\,\mathsf{LAM}\, v_2.\,\Theta_{[v_2,v_8]}([v_8]_{\alpha}\ [v_3]_{\alpha})$$

$$= \ \mathsf{LAM}\, v_8.\,\mathsf{LAM}\, v_2.\,(\Theta_{[v_2,v_8]}[v_8]_{\alpha}\ \$\$\ \Theta_{[v_2,v_8]}[v_3]_{\alpha})$$

Example Calculation of Θ

$$
\begin{aligned}
&\Theta_{\epsilon}\,[\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_{\alpha} \\
&\quad = \ \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, \Theta_{[v_2,v_8]}([v_8]_{\alpha}\ [v_3]_{\alpha}) \\
&\quad = \ \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, (\Theta_{[v_2,v_8]}[v_8]_{\alpha}\ \$\$\ \Theta_{[v_2,v_8]}[v_3]_{\alpha}) \\
&\quad = \ \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, (v_8\ \$\$\ \mathsf{VAR}\ 3)
\end{aligned}
$$

Example Calculation of Θ

$$
\begin{aligned}
&\Theta_{\epsilon}\,[\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_{\alpha} \\
&\quad = \ \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, \Theta_{[v_2, v_8]}([v_8]_{\alpha}\ [v_3]_{\alpha}) \\
&\quad = \ \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, (\Theta_{[v_2, v_8]}[v_8]_{\alpha}\ \$\$\ \Theta_{[v_2, v_8]}[v_3]_{\alpha}) \\
&\quad = \ \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, (v_8\ \$\$\ \mathsf{VAR}\ 3) \\
&\quad = \ \mathsf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_2.\, (v_8\ \$\$\ \mathsf{VAR}\ 3))))
\end{aligned}
$$

Example Calculation of Θ

$$
\begin{array}{ll}
\multicolumn{2}{l}{\Theta_{\epsilon}\,[\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_{\alpha}} \\
= & \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, \Theta_{[v_2, v_8]}([v_8]_{\alpha}\ [v_3]_{\alpha}) \\
= & \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, (\Theta_{[v_2, v_8]}[v_8]_{\alpha}\ \$\$\ \Theta_{[v_2, v_8]}[v_3]_{\alpha}) \\
= & \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_2.\, (v_8\ \$\$\ \mathsf{VAR}\ 3) \\
= & \mathsf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_2.\, (v_8\ \$\$\ \mathsf{VAR}\ 3)))) \\
= & \vdots \\
= & \mathsf{ABS}\ (\mathsf{ABS}\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))
\end{array}
$$

Outline Proof of Hybrid Adequacy

$$
\begin{tikzcd}
\mathcal{LE}/{\sim_\alpha} \arrow[rrr, "\Theta_L"] \arrow[d, equal] & & & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[d, equal] & \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] \arrow[d, equal] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] \arrow[d, equal] & \mathcal{DB} \arrow[r, "inst\ \mathbf{0}\ L"] \arrow[d, equal] & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[r, "hdb\ \mathbf{0}\ L"] & \mathcal{DB} \arrow[d, equal] \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] & \mathcal{DB} \arrow[rr, "id"] & & \mathcal{DB}
\end{tikzcd}
$$

We show Θ_L exists by

- LEMMA proving existence of *inst*
- PROPOSITION showing $\Theta_L = (inst\ \mathbf{0}\ L) \circ \iota \circ \theta_L$

Outline Proof of Hybrid Adequacy

$$
\begin{tikzcd}
\mathcal{LE}/{\sim_\alpha} \arrow[rrr, "\Theta_L"] \arrow[d, equal] & & & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[d, equal] & \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] \arrow[d, equal] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] \arrow[d, equal] & \mathcal{DB} \arrow[r, "inst\ \mathbf{0}\ L"] \arrow[d, equal] & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[r, "hdb\ \mathbf{0}\ L"] & \mathcal{DB} \arrow[d, equal] \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] & \mathcal{DB} \arrow[rr, "id"] & & \mathcal{DB}
\end{tikzcd}
$$

A Proof of Adequacy:

- $\theta = \theta_\epsilon$ is representationally adequate (de Bruijn adequacy).
- Since $\Theta = \Theta_\epsilon = (inst\ \mathbf{0}\ \epsilon) \circ \iota \circ \theta_\epsilon$, then Θ_ϵ is representationally adequate if $inst\ \mathbf{0}\ \epsilon$ is.

Outline Proof of Hybrid Adequacy

$$
\begin{tikzcd}
\mathcal{LE}/{\sim_\alpha} \arrow[rrr, "\Theta_L"] \arrow[d, equal] & & & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[d, equal] & \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] \arrow[d, equal] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] \arrow[d, equal] & \mathcal{DB} \arrow[r, "inst\ 0\ L"] \arrow[d, equal] & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[r, "hdb\ 0\ L"] & \mathcal{DB} \arrow[d, equal] \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] & \mathcal{DB} \arrow[rr, "id"] & & \mathcal{DB}
\end{tikzcd}
$$

PROPOSITION We show *inst* is injective by

- LEMMA proving existence of *hdb*, and
- LEMMA showing *hdb* is a left inverse for *inst*.

Outline Proof of Hybrid Adequacy

$$
\begin{tikzcd}
\mathcal{LE}/{\sim_\alpha} \arrow[rrr, "\boldsymbol{\Theta}_L"] \arrow[d, equal] & & & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[d, equal] & \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] \arrow[d, equal] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] \arrow[d, equal] & \mathcal{DB} \arrow[r, "inst\ \mathbf{0}\ L"] \arrow[d, equal] & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[r, "hdb\ \mathbf{0}\ L"] & \mathcal{DB} \arrow[d, equal] \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] & \mathcal{DB} \arrow[rr, "id"] & & \mathcal{DB}
\end{tikzcd}
$$

PROPOSITION We show that *inst* is a compositional homomorphism by direct proof (omitted from this talk—see the paper).

Deriving *inst*

$$\boxed{\lambda \underbrace{v_6}_{2} . \lambda \underbrace{v_8}_{1} . \lambda \underbrace{v_2}_{0} . v_6\ v_3}$$

seek $\Theta_\epsilon = (inst\ 0\ \epsilon) \circ \iota \circ \theta_\epsilon$

$$\begin{tikzcd}
\lambda v_2 . v_6\ v_3 \arrow[rr, "\Theta_L"] \arrow[d, equal] & & \mathsf{ABS}\ (v_6\ \$\$\ \mathsf{VAR}\ 3) \arrow[d, equal] \\
\lambda v_2 . v_6\ v_3 \arrow[r, "\theta_L"] & \mathsf{ABS}\ (\mathsf{BND}\ 2\ \$\$\ \mathsf{VAR}\ 3) \arrow[r, "inst\ 0\ L"] & \mathsf{ABS}\ (v_6\ \$\$\ \mathsf{VAR}\ 3)
\end{tikzcd}$$

where $L \stackrel{\text{def}}{=} [v_8, v_6]$

Deriving *inst*

$$\boxed{\lambda \underbrace{v_6}_{2} . \lambda \underbrace{v_8}_{1} . \lambda \underbrace{v_2}_{0} . v_6\ v_3}$$

$$\mathit{inst}\ \mathbf{0}\ [v_8, v_6]\ (\textsf{ABS}\ (\textsf{BND}\ 2\ \$\$\ \textsf{VAR}\ 3))$$

$$= \textsf{ABS}\ (\mathit{inst} \underbrace{\mathbf{1}}_{n} \underbrace{[v_8, v_6]}_{L} (\textsf{BND} \underbrace{\mathbf{2}}_{j}\ \$\$\ \textsf{VAR}\ 3))$$

- **BND 2** matches λ-binding variable 2.
- λ-binding v_2 has been counted by $\mathbf{1}$.
- Therefore **BND 2** matches λ-binding $(\mathbf{2} - \mathbf{1})$ in $[v_8, v_6] = v_6$.
- **BND** j matches λ-binder $(j - n)$ in L.

Deriving *inst*

$$\boxed{\lambda \underbrace{v_6}_{2} . \lambda \underbrace{v_8}_{1} . \lambda \underbrace{v_2}_{0} . v_6\ v_3}$$

$$inst\ \mathbf{0}\ [v_8, v_6]\ (\mathsf{ABS}\ (\mathsf{BND}\ 2\ \$\$\ \mathsf{VAR}\ 3))$$

$$= \mathsf{ABS}\ (inst\ \underbrace{1}_{n}\ \underbrace{[v_8, v_6]}_{L}\ (\mathsf{BND}\ \underbrace{2}_{j}\ \$\$\ \mathsf{VAR}\ 3))$$

- **BND 2** matches λ-binding variable 2.
- λ-binding v_2 has been counted by **1**.
- Therefore **BND 2** matches λ-binding $(2 - 1)$ in $[v_8, v_6] = v_6$.
- **BND** j matches λ-binder $(j - n)$ in L.

Deriving *inst*

$$\boxed{\lambda \underbrace{v_6}_{2} . \lambda \underbrace{v_8}_{1} . \lambda \underbrace{v_2}_{0} . v_6\ v_3}$$

$$inst\ \mathbf{0}\ [v_8, v_6]\ (\textsf{ABS}\ (\textsf{BND}\ 2\ \$\$\ \textsf{VAR}\ 3))$$

$$= \textsf{ABS}\ (inst\ \underbrace{\mathbf{1}}_{n}\ \underbrace{[v_8, v_6]}_{L}\ (\textsf{BND}\ \underbrace{\mathbf{2}}_{j}\ \$\$\ \textsf{VAR}\ 3))$$

- **BND 2** matches λ-binding variable 2.
- λ-binding v_2 has been counted by **1**.
- Therefore **BND 2** matches λ-binding $(2 - 1)$ in $[v_8, v_6] = v_6$.
- **BND** j matches λ-binder $(j - n)$ in L.

Deriving *inst*

$$\boxed{\lambda \underbrace{v_6}_{2} . \lambda \underbrace{v_8}_{1} . \lambda \underbrace{v_2}_{0} . v_6\ v_3}$$

$$inst\ \mathbf{0}\ [v_8, v_6]\ (\textsf{ABS}\ (\textsf{BND}\ \mathbf{2}\ \$\$\ \textsf{VAR}\ \mathbf{3}))$$

$$= \textsf{ABS}\ (inst\ \underbrace{\mathbf{1}}_{n}\ \underbrace{[v_8, v_6]}_{L}\ (\textsf{BND}\ \underbrace{\mathbf{2}}_{j}\ \$\$\ \textsf{VAR}\ \mathbf{3}))$$

- **BND 2** matches λ-binding variable 2.
- λ-binding v_2 has been counted by $\mathbf{1}$.
- Therefore **BND 2** matches λ-binding $(2 - 1)$ in $[v_8, v_6] = v_6$.
- **BND** j matches λ-binder $(j - n)$ in L.

Deriving *inst*

$$\boxed{\lambda \underbrace{v_6}_{2}.\lambda \underbrace{v_8}_{1}.\lambda \underbrace{v_2}_{0}.v_6\ v_3}$$

$$\begin{aligned} &\mathit{inst}\ 0\ [v_8, v_6]\ (\mathsf{ABS}\ (\mathsf{BND}\ 2\ \$\$\ \mathsf{VAR}\ 3)) \\ &\quad = \mathsf{ABS}\ (\mathit{inst}\ \underbrace{1}_{n}\ \underbrace{[v_8, v_6]}_{L}\ (\mathsf{BND}\ \underbrace{2}_{j}\ \$\$\ \mathsf{VAR}\ 3)) \\ &\quad = \mathsf{ABS}\ (v_6\ \$\$\ \mathsf{VAR}\ 3) \end{aligned}$$

$$\mathit{inst}\ n\ L\ (\mathsf{BND}\ j) \stackrel{\text{def}}{=} \mathit{elt}\ (j - n)\ L$$

Lemma

There is a unique function

$$inst\ n\ L\ :\ \mathcal{DB} \to \mathcal{CLF}_{exp}(\Gamma^{L}_{exp})$$

which satisfies the recursion equations

- $inst\ n\ L\ (\mathsf{VAR}\ i) = \mathsf{VAR}\ i$
-

$$inst\ n\ L\ (\mathsf{BND}\ j) \;=\; \begin{cases} elt\ (j-n)\ L & \textit{if}\ 0 \le j-n < |L| \\ \mathsf{BND}\ j & \textit{otherwise} \end{cases}$$

- $inst\ n\ L\ (D_1\ \$\$\ D_2) = (inst\ n\ L\ D_1)\ \$\$\ (inst\ n\ L\ D_2)$
- $inst\ n\ L\ (\mathsf{ABS}\ D) = \mathsf{ABS}\ (inst\ (n+1)\ L\ D)$

Example Calculation of *inst*

level counter

meta-variables

de Bruijn expression

$$\textit{inst}\quad n\ [v_{i_1},\ldots,v_{i_m}]\ D$$

$$\textit{inst}\ \mathbf{0}\ [v_2, v_8]\ (\mathsf{ABS}\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{BND}\ 0\ \$\$\ \mathsf{VAR}\ 3))$$

$$= \mathsf{ABS}\ (\textit{inst}\ \mathbf{1}\ [v_2, v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{BND}\ 0\ \$\$\ \mathsf{VAR}\ 3))$$

$$= \mathsf{ABS}\ (\textit{inst}\ 1\ [v_2, v_8]\ (\mathsf{BND}\ 1)\ \$\$\ \ldots$$

$$\ldots \textit{inst}\ 1\ [v_2, v_8]\ (\mathsf{BND}\ 0)\ \$\$\ \textit{inst}\ 1\ [v_2, v_8]\ (\mathsf{VAR}\ 3))$$

$$= v_2\ \$\$\ \mathsf{BND}\ 0\ \$\$\ \mathsf{VAR}\ 3$$

Example Calculation of *inst*

level counter

meta-variables

de Bruijn expression

$$inst \quad n \; [v_{i_1}, \ldots, v_{i_m}] \; D$$

$$inst\ 0\ [v_2, v_8]\ (\mathbf{ABS}\ (\mathbf{BND}\ 1\ \$\$\ \mathbf{BND}\ 0\ \$\$\ \mathbf{VAR}\ 3))$$

$$= \mathbf{ABS}\ (inst\ 1\ [v_2, v_8]\ (\mathbf{BND}\ 1\ \$\$\ \mathbf{BND}\ 0\ \$\$\ \mathbf{VAR}\ 3))$$

$$= \mathbf{ABS}\ (inst\ 1\ [v_2, v_8]\ (\mathbf{BND}\ 1)\ \$\$\ \ldots$$

$$\ldots inst\ 1\ [v_2, v_8]\ (\mathbf{BND}\ 0)\ \$\$\ inst\ 1\ [v_2, v_8]\ (\mathbf{VAR}\ 3))$$

$$= v_2\ \$\$\ \mathbf{BND}\ 0\ \$\$\ \mathbf{VAR}\ 3$$

Example Calculation of *inst*

level counter

meta-variables

de Bruijn expression

$$\mathit{inst}\quad n\ [v_{i_1}, \ldots, v_{i_m}]\ D$$

$$\mathit{inst}\ 0\ [v_2, v_8]\ (\mathsf{ABS}\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{BND}\ 0\ \$\$\ \mathsf{VAR}\ 3))$$

$$= \mathsf{ABS}\ (\mathit{inst}\ 1\ [v_2, v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{BND}\ 0\ \$\$\ \mathsf{VAR}\ 3))$$

$$= \mathsf{ABS}\ (\mathit{inst}\ 1\ [v_2, v_8]\ (\mathsf{BND}\ 1)\ \$\$ \ldots$$

$$\ldots \mathit{inst}\ 1\ [v_2, v_8]\ (\mathsf{BND}\ 0)\ \$\$\ \mathit{inst}\ 1\ [v_2, v_8]\ (\mathsf{VAR}\ 3))$$

$$= v_2\ \$\$\ \mathsf{BND}\ 0\ \$\$\ \mathsf{VAR}\ 3$$

Example Calculation of *inst*

level counter

meta-variables

de Bruijn expression

$$inst \quad n \ [v_{i_1}, \ldots, v_{i_m}] \ D$$

$$inst\ 0\ [v_2, v_8]\ (\mathbf{ABS}\ (\mathbf{BND}\ 1\ \$\$\ \mathbf{BND}\ 0\ \$\$\ \mathbf{VAR}\ 3))$$

$$= \mathbf{ABS}\ (inst\ 1\ [v_2, v_8]\ (\mathbf{BND}\ 1\ \$\$\ \mathbf{BND}\ 0\ \$\$\ \mathbf{VAR}\ 3))$$

$$= \mathbf{ABS}\ (inst\ 1\ [v_2, v_8]\ (\mathbf{BND}\ 1)\ \$\$\ \ldots$$

$$\ldots inst\ 1\ [v_2, v_8]\ (\mathbf{BND}\ 0)\ \$\$\ inst\ 1\ [v_2, v_8]\ (\mathbf{VAR}\ 3))$$

$$= v_2\ \$\$\ \mathbf{BND}\ 0\ \$\$\ \mathbf{VAR}\ 3$$

Proposition

$$\Theta_\epsilon = (inst\ 0\ \epsilon) \circ \iota \circ \theta_\epsilon$$

$$\begin{tikzcd}
\mathcal{LE}/{\sim_\alpha} \arrow[rrr, "\Theta_L"] \arrow[d, equal] & & & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[d, equal] & \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] \arrow[d, equal] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] \arrow[d, equal] & \mathcal{DB} \arrow[r, "inst\ 0\ L"] \arrow[d, equal] & \mathcal{CLF}_{exp}(\Gamma^L_{exp}) \arrow[r, dashed, "hdb\ 0\ L"] & \mathcal{DB} \arrow[d, equal] \\
\mathcal{LE}/{\sim_\alpha} \arrow[r, "\theta_L"] & \mathcal{DB}(|L|) \arrow[r, hook, "\iota"] & \mathcal{DB} \arrow[rr, "id"] & & \mathcal{DB}
\end{tikzcd}$$

Let's illustrate the proof . . . uses a key lemma . . .

Lemma

$$lbnd\ n\ (\Lambda\, v_k.\, inst\ n\ (v_k, L)\ D) = inst\ (n+1)\ L\ D$$

$$
\begin{aligned}
&\Theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha) \\
&= \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_3.\, \Theta_{[v_2, v_8]}([v_8]_\alpha\ [v_3]_\alpha) \\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_2.\, (v_8\ \$\$\ \mathsf{VAR}\ 3)))) \\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (\ldots \\
&\qquad\qquad \ldots lbnd\ 0\ \Lambda\, v_2.\, inst\ 0\ [v_2, v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3)))) \\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (inst\ 1\ [v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3)))) \\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (inst\ 0\ [v_8]\ \mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3)))) \\
&= \mathsf{ABS}\ (inst\ 1\ \epsilon\ (\mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3)))) \\
&= inst\ 0\ \epsilon\ (\mathsf{ABS}\ (\mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))) \\
&= inst\ 0\ \epsilon\ \theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha)
\end{aligned}
$$

Lemma

$$\mathit{lbnd}\ n\ (\Lambda\, v_k.\, \mathit{inst}\ n\ (v_k, L)\ D) = \mathit{inst}\ (n+1)\ L\ D$$

$$
\begin{aligned}
& \Theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha) \\
&= \textsf{LAM}\, v_8.\, \textsf{LAM}\, v_3.\, \Theta_{[v_2,v_8]}([v_8]_\alpha\ [v_3]_\alpha) \\
&= \textsf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\textsf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_2.\, (v_8\ \$\$\ \textsf{VAR}\ 3)))) \\
&= \textsf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\textsf{ABS}\ (\ldots \\
&\qquad \ldots \mathit{lbnd}\ 0\ \Lambda\, v_2.\, \mathit{inst}\ 0\ [v_2, v_8]\ (\textsf{BND}\ 1\ \$\$\ \textsf{VAR}\ 3)))) \\
&= \textsf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\textsf{ABS}\ (\mathit{inst}\ 1\ [v_8]\ (\textsf{BND}\ 1\ \$\$\ \textsf{VAR}\ 3)))) \\
&= \textsf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\mathit{inst}\ 0\ [v_8]\ \textsf{ABS}\ ((\textsf{BND}\ 1\ \$\$\ \textsf{VAR}\ 3)))) \\
&= \textsf{ABS}\ (\mathit{inst}\ 1\ \epsilon\ (\textsf{ABS}\ ((\textsf{BND}\ 1\ \$\$\ \textsf{VAR}\ 3)))) \\
&= \mathit{inst}\ 0\ \epsilon\ (\textsf{ABS}\ (\textsf{ABS}\ ((\textsf{BND}\ 1\ \$\$\ \textsf{VAR}\ 3))) \\
&= \mathit{inst}\ 0\ \epsilon\ \theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha)
\end{aligned}
$$

Lemma

$$\mathit{lbnd}\ n\ (\Lambda\, v_k.\, \mathit{inst}\ n\ (v_k, L)\ D) = \mathit{inst}\ (n+1)\ L\ D$$

$$\begin{aligned}
&\Theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha) \\
&= \mathbf{LAM}\, v_8.\, \mathbf{LAM}\, v_3.\, \Theta_{[v_2, v_8]}([v_8]_\alpha\ [v_3]_\alpha) \\
&= \mathbf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\mathbf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_2.\, (v_8\ \$\$\ \mathbf{VAR}\ 3)))) \\
&= \mathbf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\mathbf{ABS}\ (\ldots \\
&\qquad\qquad \ldots \mathit{lbnd}\ 0\ \Lambda\, v_2.\, \mathit{inst}\ 0\ [v_2, v_8]\ (\mathbf{BND}\ 1\ \$\$\ \mathbf{VAR}\ 3)))) \\
&= \mathbf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\mathbf{ABS}\ (\mathit{inst}\ 1\ [v_8]\ (\mathbf{BND}\ 1\ \$\$\ \mathbf{VAR}\ 3)))) \\
&= \mathbf{ABS}\ (\mathit{lbnd}\ 0\ \Lambda\, v_8.\, (\mathit{inst}\ 0\ [v_8]\ \mathbf{ABS}\ ((\mathbf{BND}\ 1\ \$\$\ \mathbf{VAR}\ 3)))) \\
&= \mathbf{ABS}\ (\mathit{inst}\ 1\ \epsilon\ (\mathbf{ABS}\ ((\mathbf{BND}\ 1\ \$\$\ \mathbf{VAR}\ 3)))) \\
&= \mathit{inst}\ 0\ \epsilon\ (\mathbf{ABS}\ (\mathbf{ABS}\ ((\mathbf{BND}\ 1\ \$\$\ \mathbf{VAR}\ 3))) \\
&= \mathit{inst}\ 0\ \epsilon\ \theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha)
\end{aligned}$$

Lemma

$$lbnd\ n\ (\Lambda\, v_k.\, inst\ n\ (v_k, L)\ D) = inst\ (n+1)\ L\ D$$

$$
\begin{aligned}
&\Theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha)\\
&= \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_3.\, \Theta_{[v_2,v_8]}([v_8]_\alpha\ [v_3]_\alpha)\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_2.\, (v_8\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (\ldots\\
&\qquad\qquad \ldots lbnd\ 0\ \Lambda\, v_2.\, inst\ 0\ [v_2, v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (inst\ 1\ [v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (inst\ 0\ [v_8]\ \mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (inst\ 1\ \epsilon\ (\mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= inst\ 0\ \epsilon\ (\mathsf{ABS}\ (\mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3)))\\
&= inst\ 0\ \epsilon\ \theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha)
\end{aligned}
$$

Lemma

$$lbnd\ n\ (\Lambda\, v_k.\, inst\ n\ (v_k, L)\ D) = inst\ (n+1)\ L\ D$$

$$
\begin{aligned}
&\Theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha)\\
&= \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_3.\, \Theta_{[v_2,v_8]}([v_8]_\alpha\ [v_3]_\alpha)\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_2.\, (v_8\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (\ldots\\
&\qquad\qquad \ldots lbnd\ 0\ \Lambda\, v_2.\, inst\ 0\ [v_2, v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (inst\ 1\ [v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (inst\ 0\ [v_8]\ \mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (inst\ 1\ \epsilon\ (\mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= inst\ 0\ \epsilon\ (\mathsf{ABS}\ (\mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3)))\\
&= inst\ 0\ \epsilon\ \theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha)
\end{aligned}
$$

Proposition

$$\Theta_\epsilon = (inst\ 0\ \epsilon) \circ \iota \circ \theta_\epsilon$$

$$
\begin{aligned}
&\Theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha)\\
&= \mathsf{LAM}\, v_8.\, \mathsf{LAM}\, v_3.\, \Theta_{[v_2,v_8]}([v_8]_\alpha\ [v_3]_\alpha)\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_2.\, (v_8\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (\ldots\\
&\qquad\qquad \ldots lbnd\ 0\ \Lambda\, v_2.\, inst\ 0\ [v_2, v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (\mathsf{ABS}\ (inst\ 1\ [v_8]\ (\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (lbnd\ 0\ \Lambda\, v_8.\, (inst\ 0\ [v_8]\ \mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= \mathsf{ABS}\ (inst\ 1\ \epsilon\ (\mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3))))\\
&= inst\ 0\ \epsilon\ (\mathsf{ABS}\ (\mathsf{ABS}\ ((\mathsf{BND}\ 1\ \$\$\ \mathsf{VAR}\ 3)))\\
&= inst\ 0\ \epsilon\ \theta_\epsilon([\lambda\, v_8.\, \lambda\, v_2.\, v_8\ v_3]_\alpha)
\end{aligned}
$$

Defining *hdb*

Lemma

There is a unique function (an inverse to $inst$*)*

$$hdb\ n\ L : \mathcal{CLF}_{exp}(\Gamma^{L}_{exp}) \to \mathcal{DB}$$

satisfying the recursion equations

- $hdb\ n\ L\ v_k = \mathsf{BND}\ ((pos\ v_k\ L) + n)$
- $hdb\ n\ L\ (\mathsf{VAR}\ i) = \mathsf{VAR}\ i$
- $hdb\ n\ L\ (\mathsf{BND}\ j) = \mathsf{BND}\ j$
- $hdb\ n\ L\ (C_1\ \$\$\ C_2) = (hdb\ n\ L\ (C_1))\ \$\$\ (hdb\ n\ L\ (C_2))$
- $hdb\ n\ L\ (\mathsf{ABS}\ C) = \mathsf{ABS}\ (hdb\ (n+1)\ L\ C)$

Proposition

inst is injective.

$\mathbf{0} \leq j - n < |L|$

$$
\begin{aligned}
& hdb\ n\ L\ (inst\ n\ L\ (\textsf{BND}\ j)) \\
& \qquad = \textsf{BND}\ (pos\ ((elt\ (j-n)\ L)\ L)) + n \\
& \qquad = \textsf{BND}\ ((j-n)+n) = \textsf{BND}\ j
\end{aligned}
$$

Proposition

inst is injective.

$$\mathbf{0} \leq j - n < |L|$$

$$\begin{aligned} & hdb\ n\ L\ (inst\ n\ L\ (\mathsf{BND}\ j)) \\ & \quad = \mathsf{BND}\ (pos\ ((elt\ (j-n)\ L)\ L)) + n \\ & \quad = \mathsf{BND}\ ((j-n)+n) = \mathsf{BND}\ j \end{aligned}$$

Proposition

inst is injective.

$0 \leq j - n < |L|$

$$\begin{aligned} & hdb\ n\ L\ (inst\ n\ L\ (\mathsf{BND}\ j)) \\ & \quad = \mathsf{BND}\ (pos\ ((elt\ (j-n)\ L)\ L)) + n \\ & \quad = \mathsf{BND}\ ((j-n)+n) = \mathsf{BND}\ j \end{aligned}$$

$n > j$ or $j - n \geq |L|$

$$hdb\ n\ L\ (inst\ n\ L\ (\mathsf{BND}\ j)) = hdb\ n\ L\ (\mathsf{BND}\ j) = \mathsf{BND}\ j$$

Proposition

inst is a compositional homomorphism: see journal paper

HYBRID Representation Results

- Suppose **ABS** C is proper eg
  ABS $($**ABS** $($**BND** 0 **$$** **BND** $1))$.

HYBRID Representation Results

- Suppose **ABS** C is proper eg
  ABS (**ABS** (**BND** $\mathbf{0}$ $\$\$$ **BND** $\mathbf{1}$)).
- Then C is of level 1, and some bound indices may dangle; for example **ABS** (**BND** $\mathbf{0}$ $\$\$$ **BND** $\mathbf{1}$).

HYBRID Representation Results

- Suppose $\textbf{ABS}\ C$ is proper eg $\textbf{ABS}\ (\textbf{ABS}\ (\textbf{BND}\ 0\ \$\$\ \textbf{BND}\ 1))$.
- Then C is of level 1, and some bound indices may dangle; for example $\textbf{ABS}\ (\textbf{BND}\ 0\ \$\$\ \textbf{BND}\ 1)$.
- An abstraction is produced by replacing each occurrence of a dangling index with a meta-variable and then abstracting: $\Lambda\, v.\ \textbf{ABS}\ (\textbf{BND}\ 0\ \$\$\ v)$.

HYBRID Representation Results

- Suppose $\textbf{ABS}\ C$ is proper eg $\textbf{ABS}\ (\textbf{ABS}\ (\textbf{BND}\ 0\ \$\$\ \textbf{BND}\ 1))$.
- Then C is of level 1, and some bound indices may dangle; for example $\textbf{ABS}\ (\textbf{BND}\ 0\ \$\$\ \textbf{BND}\ 1)$.
- An abstraction is produced by replacing each occurrence of a dangling index with a meta-variable and then abstracting: $\Lambda\, v.\ \textbf{ABS}\ (\textbf{BND}\ 0\ \$\$\ v)$.

Abstractions feature in the induction rule

$$\frac{\begin{array}{l}\forall i.\ \Phi(\textbf{VAR}\ i) \\ \forall C\ , C'.\ \textbf{proper}\ C \wedge \Phi(C) \wedge \textbf{proper}\ C' \wedge \Phi(C') \Longrightarrow \Phi(C\ \$\$\ C') \\ \forall C.\ \textbf{abst}\ C \wedge \\ (\forall C'.\ \textbf{proper}\ C' \Longrightarrow \Phi(C') \Longrightarrow \Phi(C\ C')) \Longrightarrow \Phi(\textbf{LAM}\ v_i.\ C\ v_i)\end{array}}{\Phi(C)}$$

HYBRID Representation Results

Proposition

There is a unique function

$$abst\ n : \mathcal{CLF}_{exp \Rightarrow exp}(\Gamma^{L}_{exp}) \rightarrow \mathbb{B}$$

which satisfies the following recursion equations

- $abst\ n\ (\Lambda\, v_k.\, v_k) = T$
- $abst\ n\ (\Lambda\, v_k.\, v_{k'}) = F$ $\boxed{\text{no free meta-variables}}$
- $abst\ n\ (\Lambda\, v_k.\, \mathbf{VAR}\ i) = T$
- $abst\ n\ (\Lambda\, v_k.\, \mathbf{BND}\ j) = n < j$ $\boxed{j \text{ does not dangle}}$
- $abst\ n\ (\Lambda\, v_k.\, C_1\ \$\$\ C_2) =$
  $(abst\ n\ (\Lambda\, v_k.\, C_1)) \wedge (abst\ n\ (\Lambda\, v_k.\, C_2))$
- $abst\ n\ (\Lambda\, v_k.\, \mathbf{ABS}\ C) = abst\ (n+1)\ (\Lambda\, v_k.\, C)$

HYBRID Representation Results

We say that an element $C \in \mathcal{CLF}_{exp \Rightarrow exp}(\Gamma^L_{exp})$ is an abstraction if $abst\ \mathbf{0}\ C$ is equal to T.

Theorem

Suppose that $C \in \mathcal{CLF}_{exp \Rightarrow exp}(\epsilon)$ and that C is an abstraction. Then there exists $[\lambda\, v_i.\, E]_\alpha \in \mathcal{LE}/{\sim_\alpha}$ such that

$$\Theta_\epsilon\, [\lambda\, v_i.\, E]_\alpha = \mathsf{LAM}\, v_i.\, C\, v_i$$

Related Work

- Nameless binders introduced by de Bruijn (1972). Free variables may be named, the locally nameless approach, or specified as indices, the pure approach.
- Shankar (1988) investigated bijections between pure de Bruijn and λ-expressions:
  - The closest work to ours detailing a bijection.
  - No need to identify proper expressions, but complicated substitution.
- Our work extends Gordon's (1994) on locally nameless de Bruijn expressions and conversions from λ-expressions; he does not formalise α-equivalence classes.
- Norrish and Vestergaard (2007) provide a very thorough survey of such bijections. They work with another variation of de Bruijn expressions ...

Conclusions

- We have shown the HYBRID system representationally adequate for the λ-calculus.
- We have representation results that link HYBRID predicates and λ-expressions.
- Further work could involve the investigation of the notion of n-ary abstraction and associated higher order induction principles ...
- We might consider a dependently typed version of HYBRID ...
- Categorical and nominal models and techniques ... especially presheaf models.