A Dependent Type Theory with Names and Binding

Ulrich Schöpp

Laboratory for Foundations of Computer Science
University of Edinburgh

September 3, 2004

Names

- Everywhere: abstract syntax, π-calculus, references, ...
- What are names?
- This talk: Add names to dependent type theory.
- What do we gain from adding names to type theory?
- Why not use natural numbers? (e.g. $\lambda x_1 . x_5\ x_1$)

Names

Example: Formal reasoning about the λ-calculus

$$\text{Lam} ::= \text{var of } \mathbf{N} \mid \text{app of Lam} \times \text{Lam} \mid \text{lam of } [\mathbf{N}]\text{Lam}$$

- Informal reasoning can be formalised with natural numbers for $\mathbf{N}$. [McKinna & Pollack], [Vestergaard & Brotherston]
- Common informal reasoning:

$$(\exists n \text{ fresh name. } P(n)) \implies (\forall n \text{ fresh names. } P(n))$$

- This holds for arbitrary P because there is little that can be done with names.
- With numbers we can do more, so that the implication needs proof for each P. [McKinna & Pollack]

$\Rightarrow$ We want to make known in the type theory that little can be done with names.

Fraenkel Mostowski Set Theory

FM set theory contains a nice notion of names [Gabbay & Pitts].

Name-swapping, Syntax-independent freshness,
Freshness quantifier И, Abstraction $[\mathbf{N}]X$, binding $n.x$,
Unique choice of fresh names $\mathrm{new}\ n.\, M$.

The Schanuel topos (category corresponding to FM set theory) is our principal and motivating model.

All you need to know/accept about the Schanuel topos.

- Set-like category.
- Object of names $\mathbf{N}$. Names can be compared but that's all.
- All elements $x : A$ somehow contain names.

Names in Dependent Type Theory

Add FM-style names to dependent type theory

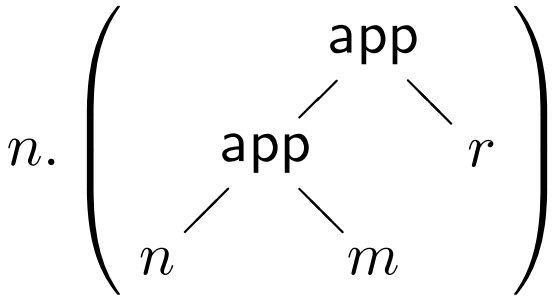
- Based on freshness rather than swapping

$$A * B = \{\langle a, b\rangle : A \times B \mid a \text{ and } b \text{ do not share names}\}$$

- Non-standard sums Σ^* and non-standard products Π^*.
- Names and FM constructions (И, $[\mathbf{N}]B$, $n.x$, $\text{new } n.\, M$) expressed in terms of Σ^* and Π^*.
- Schanuel topos models this structure.

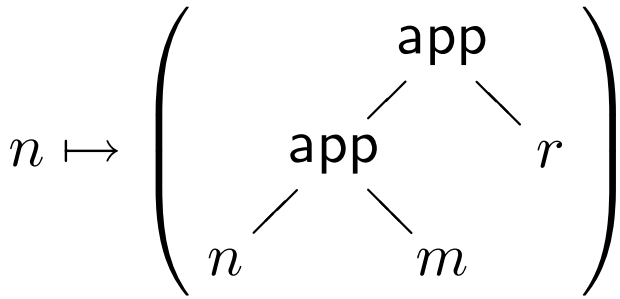
Representation of α-Equivalence Classes

1. Equivalence classes with respect to one name



n fresh for m, r $\quad [\mathbf{N}]\mathsf{Lam}$

2. Functions



$\mathbf{N} \rightharpoonup\!\!* \mathsf{Lam}$

Not all functions, e.g. not

$$n \mapsto \begin{cases} \mathsf{app}(n, n) & \text{if } n = m \\ \mathsf{lam}(n.n) & \text{otherwise} \end{cases}$$

$\Rightarrow$ Functions defined on fresh names.

Representation of α-Equivalence Classes

Fresh functions are uniform:

$$[\mathbf{N}]\mathsf{Lam} \quad \cong \quad \mathbf{N} \multimap_* \mathsf{Lam}$$

Plan

- Define $\Sigma^* x : A.\, B$ and $\Pi^* x : A.\, B$.
- Take an isomorphism $\Sigma^* x : \mathbf{N}.\, B \cong \Pi^* x : \mathbf{N}.\, B$ as the definition of names.

Simple Sums Σ and Products Π

Simple weakening from $\mathbb{B}/\Gamma$ to $\mathbb{B}/(\Gamma \times A)$ is given by

$$\pi_A^* : \begin{array}{c} B \\ f \downarrow \\ \Gamma \end{array} \longmapsto \begin{array}{c} B \times A \\ f \times A \downarrow \\ \Gamma \times A \end{array} \qquad \begin{array}{ccc} B \times A & \longrightarrow & B \\ {\scriptstyle f \times A}\downarrow & \lrcorner & \downarrow{\scriptstyle f} \\ \Gamma \times A & \xrightarrow[\pi_A]{} & \Gamma \end{array}$$

Simple products and sums are given by

$$\Sigma_A \dashv \pi_A^* \dashv \Pi_A$$

and are well-behaved under substitution

$$\Sigma_A(u \times A)^* \xrightarrow{\cong} u^*\Sigma_A\,, \qquad u^*\Pi_A \xrightarrow{\cong} \Pi_A(u \times A)^*\,.$$

Simple Sums Σ and Products Π

Simple sums always exist.

$$\Sigma_A : \quad \begin{array}{c} B \\ {\scriptstyle f}\downarrow \\ \Gamma \times A \end{array} \quad \longmapsto \quad \begin{array}{c} B \\ {\scriptstyle f}\downarrow \\ \Gamma \times A \\ {\scriptstyle \pi_A}\downarrow \\ \Gamma \end{array}$$

Simple products are equivalent to cartesian closure.

$$\Pi_A : \quad \begin{array}{c} B \\ {\scriptstyle f}\downarrow \\ \Gamma \times A \end{array} \quad \longmapsto \quad \begin{array}{ccc} \Pi_A B & \longrightarrow & A \Rightarrow B \\ {\scriptstyle \Pi_A f}\downarrow & & {\scriptstyle A \Rightarrow f}\downarrow \\ \Gamma & \xrightarrow[\eta]{} & A \Rightarrow (\Gamma \times A) \end{array}$$

Simple Monoidal Sums Σ^* and Products Π^*

Replace $(\times, \Rightarrow)$ by $(*, -\!\!*)$

Monoidal weakening from $\mathbb{B}/\Gamma$ to $\mathbb{B}/(\Gamma * A)$

$$W_A : \begin{array}{c} B \\ {\scriptstyle f}\downarrow \\ \Gamma \end{array} \longmapsto \begin{array}{c} B * A \\ {\scriptstyle f*A}\downarrow \\ \Gamma * A \end{array}$$

W_A interacts well with substitution if $- * A$ preserves pullbacks.
Simple monoidal products and sums

$$\Sigma_A^* \dashv W_A \dashv \Pi_A^*,$$

$$\Sigma_A^*(u * A)^* \xrightarrow{\cong} u^*\Sigma_A^*\,, \qquad u^*\Pi_A^* \xrightarrow{\cong} \Pi_A^*(u * A)^*\,.$$

Simple Monoidal Sums Σ^* and Products Π^*

Simple monoidal products are equivalent to monoidal closure.

$$\Pi_A^* : \quad \begin{array}{c} B \\ {\scriptstyle f}\downarrow \\ \Gamma * A \end{array} \quad \longmapsto \quad \begin{array}{ccc} \Pi_A^* B & \longrightarrow & A \multimap B \\ {\scriptstyle \Pi_A^* f}\downarrow & & \downarrow{\scriptstyle A \multimap f} \\ \Gamma & \xrightarrow[\eta]{} & A \multimap (\Gamma * A) \end{array}$$

Simple monoidal sums do *not* always exist.

A Bunched Dependent Type Theory with Σ^* and Π^*

Give a dependent type theory with $\Sigma^*_A \dashv W_A \dashv \Pi^*_A$.

Use *bunches* [O'Hearn, Pym] to integrate the monoidal structure

$$\frac{}{\diamond \text{ Bunch}} \qquad \frac{\Gamma \vdash A \text{ Type}}{\Gamma,\, x : A \text{ Bunch}} \qquad \frac{\Gamma \text{ Bunch} \qquad \Delta \text{ Bunch}}{\Gamma * \Delta \text{ Bunch}}$$

Examples:

$$(x : \mathsf{Lam},\, y : \mathsf{Lam}) * z : \mathsf{Lam}$$

$$((x : A * y : B),\, z : C) * u : D$$

Structural Rules

Projection

$$\frac{\Gamma \vdash A\ \mathsf{Type}}{\Gamma,\, x : A \vdash x : A}$$

Weakening

$$\frac{\Gamma(\Delta) \vdash \mathcal{J} \qquad \Delta \vdash A\ \mathsf{Type}}{\Gamma(\Delta,\, x : A) \vdash \mathcal{J}}$$

Substitution

$$\frac{\Delta \vdash M : A \qquad \Gamma(\Delta,\, x : A) \vdash \mathcal{J}}{\Gamma(\Delta)\,[M/x] \vdash \mathcal{J}\,[M/x]}$$

Unit, Swap, Assoc

$$\frac{\Gamma(\Delta) \vdash \mathcal{J}}{\Gamma(\Delta * \diamond) \vdash \mathcal{J}} \qquad \frac{\Gamma(\Delta * \Phi) \vdash \mathcal{J}}{\Gamma(\Phi * \Delta) \vdash \mathcal{J}} \qquad \frac{\Gamma((\Delta * \Phi) * \Psi) \vdash \mathcal{J}}{\Gamma(\Delta * (\Phi * \Psi)) \vdash \mathcal{J}}$$

Simple Monoidal Products

$(\Sigma^*_A \dashv W_A \dashv \Pi^*_A)$

$$\frac{\Gamma * x : A \vdash B \ \mathsf{Type}}{\Gamma \vdash \Pi^* x : A.\, B \ \mathsf{Type}}$$

$$\frac{\Gamma * x : A \vdash M : B}{\Gamma \vdash \lambda^* x : A.\, M : \Pi^* x : A.\, B} \qquad \frac{\Gamma \vdash M : \Pi^* x : A.\, B \qquad \Delta \vdash N : A}{\Gamma * \Delta \vdash M @ N : B\,[N/x]}$$

(plus β, η-equations)

Compare:

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A.\, M : \Pi x : A.\, B} \qquad \frac{\Gamma \vdash M : \Pi x : A.\, B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\ N : B\,[N/x]}$$

Freefrom Types

$(\Sigma_A^* \dashv W_A \dashv \Pi_A^*)$

The type $B^{*\,M:A}$ consists of all the elements of B that are *free from* the names in the term $M : A$.

- Can express more freshness assertions than bunches alone, e.g.

$$x : A,\ y : A,\ u : A^{*\,x:A},\ v : A^{*\,\langle x,y\rangle:A\times A}$$

- Gives (partial) syntax for W_A.

$$\Gamma \vdash B \text{ Type} \qquad\qquad \Gamma * x : A \vdash B^{*\,x:A} \text{ Type}$$

$$\begin{array}{c} B \\ {\scriptstyle f}\downarrow \\ \Gamma \end{array} \quad \longmapsto \quad \begin{array}{c} B * A \\ {\scriptstyle f*A}\downarrow \\ \Gamma * A \end{array}$$

Freefrom Types

- Current Restriction: Both A and B are closed in ${B^*}^{N:A}$.
- Use bunches to work with the freshness information in ${B^*}^{M:A}$.
- Can type

$$\text{remove} : \Pi n : \mathbf{N}.\ \left(\mathsf{L} \to {\mathsf{L}^*}^{n:\mathbf{N}}\right).$$

- FreshML-style rules such as

$$\frac{\begin{array}{l}\Gamma \vdash n : \mathbf{N} \\ \Gamma,\, x : A \vdash M : B \\ \Gamma \vdash x_i : {A_i^*}^{n:\mathbf{N}} \text{ all } x_i \in \mathrm{FV}(M) \setminus \{x\}\end{array}}{\Gamma \vdash \text{“}\lambda x : A.\, M\text{”} \ :\ {(A \to B)^*}^{n:\mathbf{N}}}$$

  are admissible.

Simple Monoidal Sums

$(\Sigma^*_A \dashv W_A \dashv \Pi^*_A)$

$$\frac{\Gamma * x : A \vdash B \ \mathsf{Type}}{\Gamma \vdash \Sigma^* x : A.\, B \ \mathsf{Type}}$$

$$\frac{x : A \vdash B \ \mathsf{Type} \qquad \Gamma \vdash M : A \qquad \Gamma \vdash N : B\,[M/x]}{\Gamma \vdash \mathrm{bind}(M, N) : (\Sigma^* x : A.\, B)^{*\,M:A}}$$

The freefrom type comes from the unit of the adjunction $\Sigma^*_A \dashv W_A$.

$$\eta_B : B \longrightarrow W_A \Sigma^*_A B$$

$$(\Delta * x : A),\ y : B \vdash \mathrm{bind}(x, y) : (\Sigma^* x : A.\, B)^{*\,x:A}$$

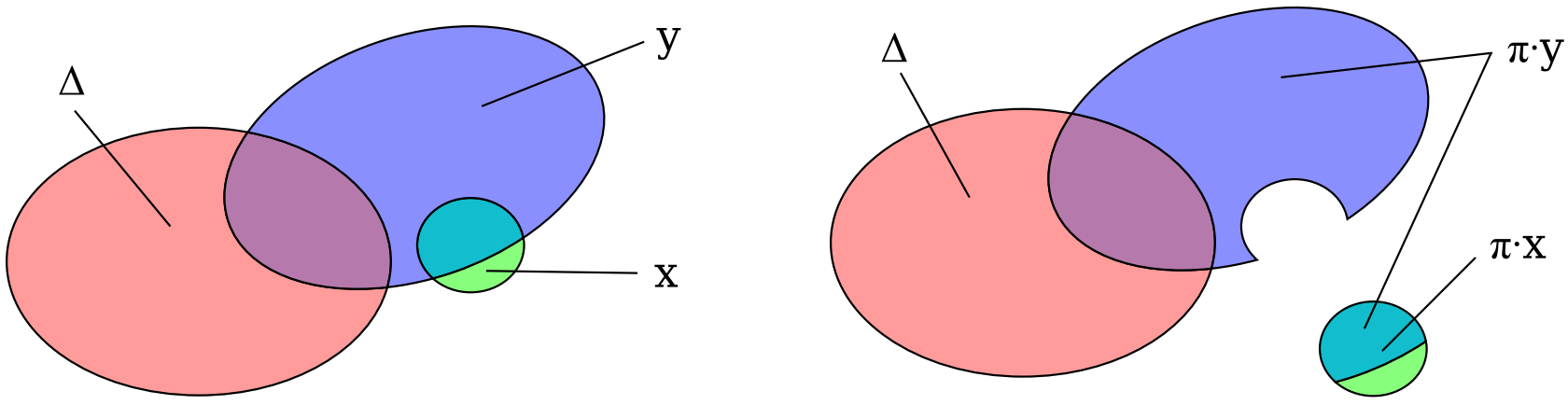
Abbreviation:

$$M.N \stackrel{\mathrm{def}}{=} (\mathrm{let}\ \mathrm{bind}(M, N)\ \mathrm{be}\ y^{*x}\ \mathrm{in}\ y)\ :\ \Sigma^* x : A.\, B$$

Simple Monoidal Sums: Binding

$$(\Delta * x : A),\ y : B \vdash \mathrm{bind}(x, y) : (\Sigma^* x : A.\ B)^{*\, x:A}$$

"$x.y$ is a pair in which the names in x are hidden"



$$x.y \qquad = \qquad (\pi \cdot x).(\pi \cdot y)$$

$$\begin{array}{ll} \Sigma^* x : \mathbf{N}.\ \mathbf{N} & n.n = m.m \neq n.r = m.r \\ \Sigma^* x : \mathbb{N}.\ \mathbf{N} & 0.n \neq 1.n \neq 2.n \cdots \neq 0.m \ldots \end{array}$$

Simple Monoidal Sums: Elimination

$$\frac{\Gamma \vdash M : \Sigma^* x : A.\, B \qquad (\Gamma * x : A),\, y : B \vdash N : C^{* x:A}}{\Gamma \vdash \text{let } M \text{ be } x.y \text{ in } N : C}$$

$$\frac{B \longrightarrow W_A C \text{ in } \mathbb{B}/(\Gamma * A)}{\Sigma_A^* B \longrightarrow C \text{ in } \mathbb{B}/\Gamma}$$

A term

$$(\Gamma * x : A),\, y : B \vdash N : C^{* x:A}$$

amounts to

$(\Gamma * x : A),\, y : B \vdash N' : C,$ $\qquad N'$ does not depend on names in x.

$\Rightarrow$ Reinforces the view of $\Sigma^* x : A.\, B$ as a type of pairs with name-hiding.

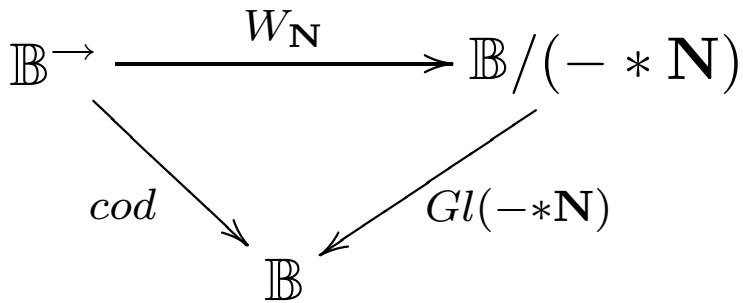
Categories with Bindable Names

A *category with bindable names* $(\mathbb{B}, *, \mathbf{N})$ is given by

$\mathbb{B}$	–	Category (with pullbacks, coproducts,. . .)
$*$	–	Strict affine symmetric monoidal structure
$\mathbf{N}$	–	Object of $\mathbb{B}$

such that

- $\mathbf{N}$ has decidable equality and $(n \neq m) \iff \langle n, m \rangle \in \mathbf{N} * \mathbf{N}$.
- The functor $W_{\mathbf{N}}$ is a fibred equivalence.



The fibred equivalence $W_{\mathbf{N}}$

- Equivalence of $\mathbb{B}/\Gamma$ and $\mathbb{B}/(\Gamma * \mathbf{N})$.

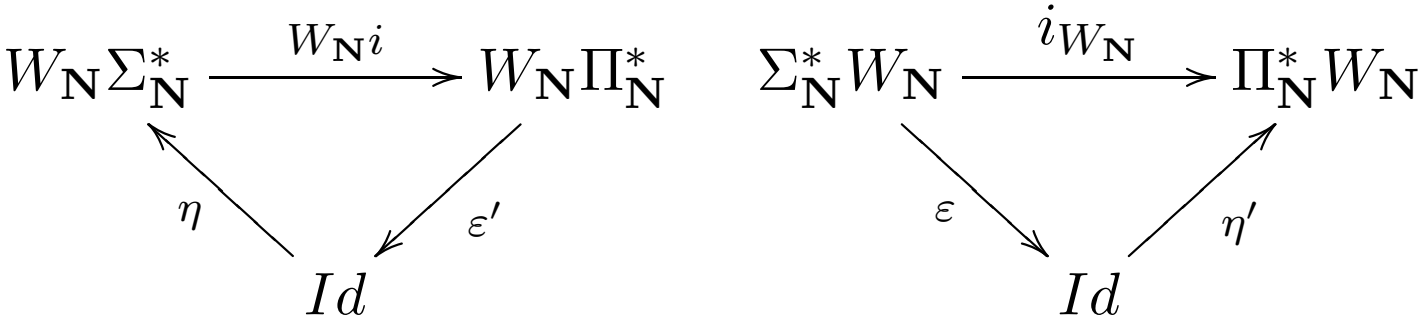
  "It doesn't matter if we work with an α-equivalence class or with a concretely named instance."

- Agrees with informal practice (cf. Barendregt Variable Convention).
- Not just on terms but on the whole category.

Bindable Names in terms of Σ^* and Π^*

The fibred equivalence $W_{\mathbf{N}}$ is equivalent to:

- A fibred adjunction $\Sigma^*_{\mathbf{N}} \dashv W_{\mathbf{N}} \dashv \Pi^*_{\mathbf{N}}$,
- A vertical natural isomorphism $i : \Sigma^*_{\mathbf{N}} \longrightarrow \Pi^*_{\mathbf{N}}$ with



Use *hidden-name types* to add the isomorphism to the type theory.

$$\Sigma^* n : \mathbf{N}.\, B \quad \cong \quad \mathrm{H} n.\, B \quad \cong \quad \Pi^* n : \mathbf{N}.\, B$$

Rules and equations for H are those from both Σ^* and Π^* plus equations from the above triangles.

Hidden-Name Types: Unique Choice of Fresh Names

Unique choice of a fresh name

$$\frac{\Gamma * n : \mathbf{N} \vdash M : B^{*n:\mathbf{N}}}{\Gamma \vdash \mathrm{new}\ n.\, M : B}$$

is derivable by

$$\frac{\dfrac{\Gamma * n : \mathbf{N} \vdash M : B^{*n:\mathbf{N}}}{\Gamma \vdash \lambda^* n.\, M : \mathrm{H}n.\, B^{*n:\mathbf{N}}} \qquad \dfrac{\vdots}{(\Gamma * n : \mathbf{N}),\, x : B^{*n:\mathbf{N}} \vdash x : B^{*n:\mathbf{N}}}}{\Gamma \vdash \mathrm{let}\ (\lambda^* n.\, M)\ \mathrm{be}\ n.x\ \mathrm{in}\ x : B}$$

Example equations:

$$\frac{\Gamma \vdash M : B}{\Gamma \vdash (\mathrm{new}\ n.\, M^{*n}) = M : B}$$

$$\frac{\Gamma * n : \mathbf{N} \vdash M : B \qquad \cdots}{\Gamma \vdash (\mathrm{new}\ n.\, \mathrm{bind}(n, M)) = \lambda^* n.\, M : \mathrm{H}n.\, B}$$

Hidden-Name Types: Substitution

$$\mathsf{Lam} ::= \mathsf{var\ of}\ \mathbf{N} \mid \mathsf{app\ of}\ \mathsf{Lam} \times \mathsf{Lam} \mid \mathsf{lam\ of}\ \mathrm{H}n.\,\mathsf{Lam}$$

Recursive definition of substitution in wHOAS-style (Π^*).

Substitution of $R : \mathsf{Lam}$ for $m : \mathbf{N}$.

$$\begin{aligned}
\mathsf{subst} &: \mathsf{Lam} \to \mathsf{Lam} \\
\mathsf{subst}(\mathsf{var}(n)) &= \text{if } (n = m) \text{ then } R \text{ else } \mathsf{var}(n) \\
\mathsf{subst}(\mathsf{app}(M, N)) &= \mathsf{app}(\mathsf{subst}(M), \mathsf{subst}(N)) \\
\mathsf{subst}(\mathsf{lam}(M)) &= \mathsf{lam}(\lambda^* n.\,\mathsf{subst}(M@n))
\end{aligned}$$

Same function in FM-style (Σ^*).

$$\mathsf{subst}(\mathsf{lam}(M)) =$$
$$\text{let } M \text{ be } n.x \text{ in } (\text{let } \mathrm{bind}(n, \mathsf{subst}(x)) \text{ be } y^{*n} \text{ in } \mathsf{lam}(y)^{*n})$$

Hidden-Name Types: Free Variables

$$\begin{array}{rcl} \mathsf{singleton} & : & \mathbf{N} \to \mathsf{L} \\ \mathsf{concat} & : & \mathsf{L} \to \mathsf{L} \to \mathsf{L} \\ \mathsf{remove} & : & \Pi n : \mathbf{N}.\, \mathsf{L} \to \mathsf{L}^{*\, n:\mathbf{N}} \end{array}$$

Free variables of a term in FM-style (Σ^*).

$$\begin{array}{rcl} \mathsf{fv} & : & \mathsf{Lam} \to \mathsf{L} \\ \mathsf{fv}(\mathsf{var}(n)) & = & \mathsf{singleton}(n) \\ \mathsf{fv}(\mathsf{app}(M, N)) & = & \mathsf{concat}(\mathsf{fv}(M), \mathsf{fv}(N)) \\ \mathsf{fv}(\mathsf{lam}(M)) & = & \text{let } M \text{ be } n.x \text{ in } \mathsf{remove}(n, \mathsf{fv}(x)) \end{array}$$

Closest to wHOAS style (uses Σ^* in new):

$$\mathsf{fv}(\mathsf{lam}(M)) = \text{new } n.\, \mathsf{remove}(n, \mathsf{fv}(M@n))$$

Hidden-Name Types and the И-Quantifier

Σ^*_A and Π^*_A restrict to quantifiers:

$$\frac{\Gamma \mid \Theta, \exists^* x : A.\, \varphi \vdash \psi}{\Gamma * x : A \mid \Theta, \varphi \vdash \psi} \qquad \frac{\Gamma \mid \Theta \vdash \forall^* x : A.\, \varphi}{\Gamma * x : A \mid \Theta \vdash \varphi}$$

Then $\Sigma^*_{\mathbf{N}} \cong \Pi^*_{\mathbf{N}}$ implies $\exists^*_{\mathbf{N}} = \forall^*_{\mathbf{N}}$, i.e. freshness quantifier И.

Where are freefrom propositions?

$$\begin{array}{ccc} B * A & \longrightarrow & B \\ {\scriptstyle \varphi * A}\downarrow & \lrcorner & \downarrow{\scriptstyle \varphi} \\ \Gamma * A & \xrightarrow[\pi_1]{} & \Gamma \end{array} \qquad \Longrightarrow \qquad \frac{\Gamma \vdash \varphi \text{ Prop}}{\Gamma * x : A \vdash \varphi \rightleftarrows \varphi^{*x:A}}$$

Conclusion and Further Work

Conclusion

Σ^* and Π^* are good for names!

With Σ^* and Π^* we can

- define names;
- work with names;
- add names to dependent type theory.

Further Work

- Better freefrom types.
- Decidability questions.
- Practicable?
- ...

Related Work

- Nominal Logic [Pitts], FM-HOL [Gabbay], Sequent Calculus for Nominal Logic [Gabbay & Cheney], α-Prolog [Cheney & Urban], Nominal Unification [Urban, Pitts, Gabbay]
- FreshML 2000 [Pitts & Gabbay], Fresh OCamL [Shinwell]
- Spatial Logics [Cardelli & Caires], Manipulating Trees with Hidden Labels [Cardelli & Gardner & Ghelli]
- Theory of Contexts [Honsell, Miculan, Scagnetto], Modal λ-calculi [Pfenning, Nanevski, ...], Meta-Reasoning [McDowell, Tiu, Miller], [Pfenning, Schürmann], [Ambler, Crole, Momigliano], ...
- Bunches [O'Hearn], [Pym]
- Presheaf semantics [Fiore, Plotkin, Turi], [Hofmann]
- Categorical logic of $И$ [Menni]
- Categorical formulation of substitution [Tanaka, Power]