Representations of Binding: Local Representations

Randy Pollack

Version of October 26, 2011

Outline

Outline

Pure Lambda terms: Raw Syntax

- Countable set $\mathbb{X}$ of *atoms* used for *variables*: X, Y, Z.
  - Only relation needed on $\mathbb{X}$ is decidable equality.
- Datatype of *terms* ranged over by M, N, P, Q:

$$M ::= X \mid P{\cdot}Q \mid [X]M$$

- **Substitution;** naive definition as in "Software Foundations" and Greg's Coq file for STLC:

$$\begin{array}{lcl} [M/Y]X & := & \text{if } Y = X \text{ then } M \text{ else } X \\ [M/Y]N_1{\cdot}N_2 & := & ([M/Y]N_1){\cdot}[M/Y]N_2 \\ [M/Y]([X]N) & := & [X](\text{if } Y = X \text{ then } N \text{ else } [M/Y]N) \end{array}$$

- This is a definition by structural recursion.
- Well known that naive substitution is wrong: allows capture

$$[X/Y]([X]Y) = [X]X.$$

Substitution isn't the only problem: Typing

- Let $A, B, \ldots$ be *simple types* (implicational propositions).
- *Valid contexts* (Γ, Δ) are lists of uniquely labelled assumptions:

$$\Gamma ::= X_1, A_1, \ldots, X_n, A_n \qquad \text{where the } X_i \text{ are pairwise distinct}$$

- Consider the rules for Simply Typed Lambda Calculus (STLC):

$$\frac{\Gamma \text{ valid} \qquad X{:}A \in \Gamma}{\Gamma \vdash X : A} \qquad (\text{ELIM})\ \frac{\Gamma \vdash b : A{\to}B \qquad \Gamma \vdash a : A}{\Gamma \vdash b\,a : B}$$

$$(\text{INTRO})\ \frac{\Gamma, Y{:}A \vdash b : B}{\Gamma \vdash [Y]b : A{\to}B}$$

- Does this system accept the shadowing judgement

$$\vdash [Y]([Y]Y) : A{\to}A{\to}A\ ?$$

Substitution isn't the only problem: Typing (2)

Greg's Coq STLC does the rules differently to accept shadowing:

$$\frac{\text{LIFO lookup }(X,\Gamma) = A}{\Gamma \vdash X : A} \qquad (\text{ELIM})\ \frac{\Gamma \vdash b : A \to B \qquad \Gamma \vdash a : A}{\Gamma \vdash b\,a : B}$$

$$(\text{INTRO})\ \frac{\Gamma, Y{:}A \vdash b : B}{\Gamma \vdash [Y]b : A \to B}$$

- No validity requirement for contexts; LIFO lookup instead.
- Accepts the judgement $\vdash [Y]([Y]Y) : A \to A \to A$.
- But this system doesn't accept context permutation
  - which the previous system does.
- Worse, LIFO lookup cannot handle dependent types.

Shadowing in Dependent Types

- Consider a naive INTRO rule for dependent types

$$\textsc{Intro}\ \frac{\Gamma, X{:}A \vdash M : B}{\Gamma \vdash \lambda X{:}A.\ M : \Pi X{:}A.\ B}$$

- The following correct judgement with shadowing is not derivable:

$$A{:}\star, P{:}A{\to}\star \vdash \lambda X{:}A.\ (\lambda X{:}P{\cdot}X.\ X) : \Pi X{:}A.\ (\Pi Y{:}P{\cdot}X.\ P{\cdot}X)\ \ .$$

- The stacking of dependencies in terms and types can differ.
- LIFO lookup doesn't solve the shadowing problem for dependent types.

What is the problem?

- The raw syntax datatype doesn't respect our idea of binding:
  - $[X]X \neq [Y]Y$ as elements of the datatype.
  - Structural induction doesn't give the right induction hypothesis.
- Informal solution: quotient syntax by α-equivalence
  - but α-equivalence is usually defined in terms of substitution
    - Smallest congruence relation containing

$$\frac{Y \neq X \qquad Y \notin \mathrm{FV}(M)}{[X]M =_\alpha [Y]([Y/X]M)}$$

  - so first define substitution on raw syntax, then define α-equivalence and show substitution respects it.

Not very pretty: our game is to avoid defining α-equivalence at all.

Aside: α without substitution

- Let $(X\ Y)\bullet M$ be the operation that swaps (permutes) X and Y in term M.
  - Viewing M as raw syntax: no binding,
- α-equivalence is the smallest congruence containing

$$\frac{Z \sharp (X, Y, M, N) \qquad (X\ Z)\bullet M =_\alpha (Y\ Z)\bullet N}{[X]M =_\alpha [Y]N}$$

  - where $Z \sharp M$ means Z doesn't appear in the raw syntax M.
- The operations of swapping $\bullet$ and freshness $\sharp$ are definable on raw syntax by structural recursion.

Name permutation on raw syntax is the basic idea of nominal sets and nominal logic.

- This technique was already observed in [McKinna/Pollack 1993].

Back to the standard approach: Carefully define substitution on raw syntax

- Commonly used in modern presentations since Church; Curry and Feys.

$$
\begin{array}{lll}
[c/X]Y & := & \text{if } X = Y \text{ then } c \text{ else } Y \\
[c/X](b_1 \cdot b_2) & := & ([c/X]b_1) \cdot ([c/X]b_2) \\
[c/X]([Y]b) & := & [Z][c/X][Z/Y]b \qquad\qquad Z \text{ sufficiently fresh}
\end{array}
$$

- $[-/-]b$ is defined by recursion on length of b,
  - not on *structure* of b, since $[Z/Y]b$ is not a subterm of $[Y]b$.
- Arbitrary choice of fresh Z could be made canonical ...
  - e.g. "first name not occurring ..."
  - this definition is deterministic given a choice function over names.

Aside: Simultaneous Substitution is Structural

- A *simultaneous substitution*, ρ, is a finite partial function from $\mathbb{X}$ to terms.

$$
\begin{array}{lcll}
x\rho & := & \rho x & \\
(a \cdot b)\rho & := & (a\rho) \cdot (b\rho) & \\
([X]b)\rho & := & [Z](b(\rho, X{=}Z)) & \quad Z \text{ sufficiently fresh}
\end{array}
$$

  Primitive recursion with variable parameter.
- The choice of fresh Z can be canonical ...
  - then applying any substitution alpha-normalizes (Stoughton)
  - so testing alpha-equivalence becomes a test for identity.
- Note: this operation has funny properties:

$$[X/X]M \neq M$$

Back to our quest to avoid α-equivalence.

Outline

Distinct species of names

Why is it natural to identify bound names with free names?

- Example: $\to$-intro rule for simple types

$$(\text{INTRO})\ \frac{\Gamma, X{:}A \vdash b : B}{\Gamma \vdash [X]b : A \to B}$$

  - 'X' really occurs in the premise (free, global).
  - 'X' does not occur in the conclusion (locally bound).

This suggests:

- Syntactically seperate local (bound) variables from global (free) variables.
- Not a new idea: Frege, Gentzen and Prawitz all informally used different species of names.

Syntax of locally named pre-terms for pure λ

As in McKinna/Pollack [TLCA 1993, JAR 1999].

- Countable set $\mathbb{V}$ of atoms used for local *variables*: x, y, z .
- Countable set $\mathbb{X}$ of atoms, used for global *parameters*: X, Y, Z, p, q .
- Only relation needed on $\mathbb{V}$, $\mathbb{X}$ is decidable equality.

Symbolic Expressions ($\mathbb{S}$):

- Datatype of pre-terms ranged over by M, N, P, Q :

$$M ::= x \mid X \mid P{\cdot}Q \mid [x]M$$

  - No way to bind global names, X .
- In general, may be other classes of variables, parameters and expressions
  - e.g. types and terms in System F.

Occurrences of Names

- Occurrences of global names (parameters)
  - $X \sharp A$ means *"X does not occur syntactically in A"*.
    - Easily defined by structural recursion
  - Corresponds to nominal freshness (also written $\sharp$).

Substitution, Concretely

- Concretely defined by *structural* recursion:

$$
\begin{array}{lcl}
[M/X]x & = & x \\
[M/X]Y & = & \text{if } X = Y \text{ then } M \text{ else } Y \\
[M/X]N{\cdot}N & = & ([M/X]N){\cdot}[M/X]N \\
[M/X]([x]N) & = & [x][M/X]N
\end{array}
$$

- Deterministic: no choosing arbitrary names.
  - Thus has natural properties; e.g.

$$
\begin{array}{l}
[X/X]M = M. \\
X \sharp M \implies [P/X]M = M.
\end{array}
$$

- Does not prevent capture, e.g. $[x/X][x]X = [x]x$.
  - Will only be used in safe ways.
- Substitution is a B-algebra homomorophism; see Pollack and Sato (J. Symb. Comp.).

Not Substitution: a purely technical operation

- Used to fill a “hole” (free variable) created by going under a binder.
- Defined by structural recursion:

$$
\begin{array}{lcl}
[M/y]x & = & \text{if } y = x \text{ then } M \text{ else } x \\
[M/y]X & = & X \\
[M/y]N_1 \cdot N_2 & = & ([M/y]N_1) \cdot [M/y]N_2 \\
[M/y]([x]N) & = & [x](\text{if } y = x \text{ then } N \text{ else } [M/y]N)
\end{array}
$$

- Respects intended scope of binding.
- Does not prevent capture, e.g. $[x/y][x]y = [x]x$.

Overview: Symbolic expressions vs λ-terms

Sexprs do not faithfully represent λ-terms for two reasons.

1. Local variables may appear unbound in sexprs.
   - 'x' is an sexpr, not intended to represent any λ-term.
     - Remark: 'X' is an sexpr representing a λ-term with one (particular) global variable.
   - The fix: select the set of sexprs with no unbound local variables.
     - Call this subset *vclosed* for *variable closed*.
2. Different sexprs may represent the same λ-term.
   - '$[x]x$' and '$[y]y$'; not canonical.
   - Ignore this for the moment.
     - We show how to reason correctly with *vclosed* expressions.

Overview: Symbolic expressions vs λ-terms

Sexprs do not faithfully represent λ-terms for two reasons.

1. Local variables may appear unbound in sexprs.
   - 'x' is an sexpr, not intended to represent any λ-term.
     - Remark: 'X' is an sexpr representing a λ-term with one (particular) global variable.
   - The fix: select the set of sexprs with no unbound local variables.
     - Call this subset *vclosed* for *variable closed*.
2. Different sexprs may represent the same λ-term.
   - '$[x]x$' and '$[y]y$'; not canonical.
   - Ignore this for the moment.
     - We show how to reason correctly with *vclosed* expressions.

Overview: Symbolic expressions vs λ-terms

Sexprs do not faithfully represent λ-terms for two reasons.

1. Local variables may appear unbound in sexprs.
    - 'x' is an sexpr, not intended to represent any λ-term.
        - Remark: 'X' is an sexpr representing a λ-term with one (particular) global variable.
    - The fix: select the set of sexprs with no unbound local variables.
        - Call this subset *vclosed* for *variable closed*.
2. Different sexprs may represent the same λ-term.
    - '$[x]x$' and '$[y]y$'; not canonical.
    - Ignore this for the moment.
        - We show how to reason correctly with *vclosed* expressions.

Variable-Closed Sexprs

A predicate meaning “no free variables”.

$$\frac{}{\mathit{vclosed}\ X} \qquad \frac{\mathit{vclosed}\ M \quad \mathit{vclosed}\ N}{\mathit{vclosed}\ M{\cdot}N} \qquad \frac{\mathit{vclosed}\ [X/y]M}{\mathit{vclosed}\ [y]M}$$

- *vclosed* terms have no unbound local variables.
- An abstraction is *vclosed* when . . . .
- Every parameter is *vclosed* and no variable is *vclosed* .
- Use *vclosed* induction instead of sexpr structural induction . . .
- . . . no case for unbound variables.

Variable-Closed and Substitution

- Operations $[M/X]N$ and $[M/x]N$ are capture free on *vclosed* .
  - There are no free local names to get captured!
- *vclosed* is trivially closed under substitution:

$$\mathit{vclosed}\ M \ \wedge\ \mathit{vclosed}\ N \implies \mathit{vclosed}\ [M/X]N$$

- Think of *vclosed* as a “weak typing judgement”.
  - *vclosed* terms behave well for substitution, just as well-typed terms behave well for computation.

Aside: Alternative definitions of Variable-Closed

Only the rules for abstraction differ:

$$\frac{X \sharp M \quad \mathit{vclosed}\ [X/y]M}{\mathit{vclosed}\ [y]M} \qquad \frac{\forall X.\ X \sharp M \Longrightarrow \mathit{vclosed}\ [X/y]M}{\mathit{vclosed}\ [y]M}$$

$$\frac{\mathit{vclosed}\ [X/y]M}{\mathit{vclosed}\ [y]M} \qquad \frac{\forall X.\ \mathit{vclosed}\ [X/y]M}{\mathit{vclosed}\ [y]M}$$

$$\frac{\mathit{vclosed}\ M}{\mathit{vclosed}\ [x][x/X]M} \qquad \frac{\forall Y.\ (Y \sharp X \Longrightarrow Y \sharp M) \Longrightarrow \mathit{vclosed}\ [Y/X]M}{\mathit{vclosed}\ [x][x/X]M}$$

- These 6 relations are pairwise *extensionally* equivalent:
  - i.e. they derive the same judgments,
  - but their derivations are different,
  - they behave differently for induction.
- Each of the relations in the left hand column have infinitely many derivations of each derivable judgement.
- Each of the relations in the right hand column have at most one derivation of any judgement.

Simply Typed Lambda Calculus (STLC)

- Let *simple types* ($A, B, \ldots$) and *valid contexts* (Γ, Δ) be as above

$$\frac{\Gamma \text{ valid} \qquad p{:}A \in \Gamma}{\Gamma \vdash p : A} \qquad (\text{ELIM})\ \frac{\Gamma \vdash b : A {\to} B \qquad \Gamma \vdash a : A}{\Gamma \vdash b\, a : B}$$

$$(\text{INTRO})\ \frac{\Gamma, p{:}A \vdash [p/y]b : B \qquad p \sharp b}{\Gamma \vdash [y]b : A {\to} B}$$

- When going under a binder, substitute a suitably fresh parameter in the hole created.
  - The choice of p is arbitrary; we will have more to say.
- Why no mention of *vclosed* ?
  - **lemma:** $\Gamma \vdash b : B \implies \mathit{vclosed}\ b$.

Simply Typed Lambda Calculus (2)

$$\frac{\Gamma \text{ valid} \qquad p{:}A \in \Gamma}{\Gamma \vdash p : A} \qquad (\textsc{Elim})\ \frac{\Gamma \vdash b : A \to B \qquad \Gamma \vdash a : A}{\Gamma \vdash b\,a : B}$$

$$(\textsc{Intro})\ \frac{\Gamma, p{:}A \vdash [p/y]b : B \qquad p \sharp b}{\Gamma \vdash [y]b : A \to B}$$

- The side condition is needed in rule INTRO to prevent too many judgements being derivable:
  - if $p \in b$ then $p \in \lambda y.b$ in the conclusion, where p is not bound in the context Γ.
- Validity side conditions are *not* required in rules INTRO and ELIM because they follow from the premises.
- This definition of $\vdash$ is easily formalised in Coq, Isabelle/HOL, ...

Beta Reduction

$$(\beta)\ \frac{\mathit{vclosed}\ [x]b \qquad \mathit{vclosed}\ s}{([x]b)\ s > [s/x]b}$$

$$(\xi)\ \frac{[p/x]s > [p/y]t \qquad p \sharp (s,t)}{[x]s > [y]t}$$

$$\frac{s_1 > t \qquad \mathit{vclosed}\ s_2}{s_1\ s_2 > t\ s_2} \qquad \frac{s_2 > t \qquad \mathit{vclosed}\ s_1}{s_1\ s_2 > s_1\ t}$$

- In rule (ξ) we must let x and y possibly be different ...
  - or else some correct reduction judgements won't be derivable.
- In β, we must restrict to $(\mathit{vclosed}\ s)$ for safety.
  - Otherwise free variables in s might be captured in $[s/x]b$.
- The other *vclosed* restrictions are for hygiene:
  - **lemma:** $a > b \implies \mathit{vclosed}\ a \wedge \mathit{vclosed}\ b$.

Dependent Types

- Now we see how to handle the problem of dependent types

$$\text{(INTRO)}\ \frac{\Gamma, p{:}A \vdash [p/x]M : [p/y]B \qquad p \sharp (M, B)}{\Gamma \vdash \lambda x{:}A.\ M : \Pi y{:}A.\ B}$$

- Allowing different names to be bound in different parts of the judgement accounts for different dependencies in terms vs. types.

What can we do with these relations?

- Definitions and statements of lemmas are natural using names.
- All the expected judgements are derivable:
  - The set of derivable judgments is closed under α-conversion and renaming.
- The standard metatheory can be developed:
  - Weakening, substitution lemma, subject reduction ...
- We almost never need to define or reason about α-conversion.
  - Church-Rosser for β-reduction does not hold on-the-nose for these *vclosed* terms: not canonical.
  - Church-Rosser does hold for Tait/Martin-Löf parallel reduction.
- We can use the Locally Nameless representation or the Sato representation to get canonicity.

Weakening for STLC

- Define subcontext:

$$\Gamma \sqsubseteq \Delta \qquad \text{iff} \qquad \forall p, A \,.\, p{:}A \in \Gamma \implies p{:}A \in \Delta$$

Δ contains every assumption occurring in Γ.

Lemma (Weakening)

$$\Gamma \vdash a : A \implies (\forall \Delta \,.\, \Gamma \sqsubseteq \Delta \wedge \Delta \text{ valid} \implies \Delta \vdash a : A).$$

Remark

- *We trivially have the lemma:* $\Gamma \vdash a : A \implies \Gamma \text{ valid}$.
- *. . . so the premise '* Δ valid *' is needed in the statement of weakening.*

Prove weakening

$$\Gamma \vdash a : A \implies (\forall \Delta \,.\, \Gamma \sqsubseteq \Delta \ \wedge\ \Delta \text{ valid} \implies \Delta \vdash a : A).$$

Proof: Attempt proof by induction on the derivation of $\Gamma \vdash a : A$

- Consider case for rule INTRO: $\dfrac{\Gamma, p{:}A \vdash [p/y]b : B \quad p \sharp b}{\Gamma \vdash [y]b : A \to B}$
- The goal is $\Delta \vdash [y]b : A \to B$
- The IH is:

$$\forall \Phi \,.\, (\Gamma, p_0{:}A \sqsubseteq \Phi \wedge \Phi \text{ valid}) \implies \Phi \vdash [p_0/y]b : B$$

  for some particular $p_0 \sharp b$.
- By rule INTRO we need to show

$$\Delta, q{:}A \vdash [q/y]b : B \quad \text{for some } q \sharp b.$$

- It seems we want to instantiate Φ in IH with $\Delta, p_0{:}A$ ...
- ... but $\Delta, p_0{:}A$ may not be valid, as p_0 may occur in Δ.

Proof of weakening (contd)

- Since p_0 may not be fresh enough, we want to exchange it for a fresh parameter.
- Let $(p\,q)\bullet b$ mean *permute all occurrences of* p *and* q *in* b.
- As a lemma (equivariance), show

$$\Gamma \vdash a : A \implies \forall p\, q\, .\ (p\,q)\bullet\Gamma \vdash (p\,q)\bullet a : A\,. \qquad (1)$$

  - This is easy to prove, but even better . . .
  - nominal Isabelle defines polymorphic permutations and proves equivariance automatically.
- Now, pick $q \sharp (\Delta, b, \Gamma)$. It suffices to show

$$\Delta, q{:}A \vdash [q/y]b : B$$

which, by (1) and IH is difficult but possible.

We have a proof; what's the problem?

- We must use name swapping explicitly (as in the *weakening* proof above) to handle each example where eigenvariable problems appear.

Better: we can package this swapping reasoning for each relation (typing, reduction, ...) once and for all.

- This technique from McKinna/Pollack (1993).

A more uniform solution to eigenvariable problems

The following judgements are equivalent:

- Arbitrary choice of p in INTRO: judgements may have infinitely many derivations:

$$\frac{\Gamma\ \text{valid} \qquad p{:}A \in \Gamma}{\Gamma \vdash p : A} \qquad (\text{ELIM})\ \frac{\Gamma \vdash b : A \to B \qquad \Gamma \vdash a : A}{\Gamma \vdash b\,a : B}$$

$$(\text{INTRO})\ \frac{\Gamma, p{:}A \vdash [p/y]b : B \qquad p \sharp b}{\Gamma \vdash [y]b : A \to B}$$

- No arbitrary choices: judgements have at most one derivation:

$$\frac{\Gamma\ \text{valid} \qquad p{:}A \in \Gamma}{\Gamma \Vdash p : A} \qquad (\text{ELIM})\ \frac{\Gamma \Vdash b : A \to B \qquad \Gamma \Vdash a : A}{\Gamma \Vdash b\,a : B}$$

$$(\text{INTRO})\ \frac{\forall p \,.\, p \sharp \Gamma \implies \Gamma, p{:}A \Vdash [p/y]b : B}{\Gamma \Vdash [y]b : A \to B}$$

A more uniform solution to eigenvariable problems

- $\vdash$ is the "official" relation, $\Vdash$ is an auxiliary notion.
- $\vdash$ is ordinary syntax: formalizable in Primitive Recursive Arithmetic.
  - $\vdash$ derivations are well-founded, finitely branching trees.
- $\Vdash$, defined by generalized inductive definition, is not formalizable in PRA.
  - $\Vdash$ derivations are well-founded but infinitely branching trees.
- By induction on the derivation of $\Gamma \Vdash a : A$, it is trivial that

$$\Gamma \Vdash a : A \implies \Gamma \vdash a : A \ .$$

The other direction takes some work.

Why are we interested in this equivalence?

- With this equivalence we can prove weakening by rule induction without name swapping.
- The equivalence of $\vdash$ and $\Vdash$ "packages" the eigenvariable reasoning that we need for all examples.
  - Introduction of $\vdash$ is easy: only need property for one fresh variable.
  - Elimination of $\Vdash$ is powerful: get the IH for all sufficiently fresh variables.
  - We use $\Vdash$ as a "derived induction principle" for $\vdash$
- Instead of using swapping arguments for every rule induction on every relation (typing, reduction, ...)
- ...we use it once for each relation, to prove strengthened induction principle.
- There is much more to say about this: cofinite quantification, nominal freshness contexts, ...

Applying the equivalence to prove weakening

- It is easy to prove weakening:

$$\Gamma \Vdash a : A \implies (\Gamma \sqsubseteq \Delta \;\wedge\; \Delta \text{ valid} \implies \Delta \vdash a : A).$$

  hence, equivalently, weakening for $\vdash$.
- **Proof:** by induction on the derivation of $\Gamma \Vdash a : A$.
- Consider case for INTRO: $\dfrac{\forall p \,.\, p \sharp \Gamma \implies \Gamma, p{:}A \Vdash [p/y]b : B}{\Gamma \Vdash [y]b : A{\to}B}$
- By rule INTRO (for $\vdash$) we need to show

$$\Delta, p{:}A \vdash [p/y]b : B \quad \text{for some } p \sharp b\,.$$

  using IH:

$$\forall \Phi\; \forall p \,.\, (p \sharp \Gamma \wedge \Gamma, p{:}A \sqsubseteq \Phi \wedge \Phi \text{ valid}) \implies \Phi \vdash [p/y]b : B\;.$$

- Select $p_0 \sharp (b, \Gamma, \Phi)$ and instantiate Φ in IH with $\Delta, p_0{:}A$. □

Proof of the equivalence of $\vdash$ and $\Vdash$

Lemma $\Gamma \vdash a : A \implies \Gamma \Vdash a : A$.

- Proof by induction on the derivation of $\Gamma \vdash a : A$.
- Consider the case of rule INTRO.
- Any derivation of $\vdash$ will use a particular parameter, say p_0.
- The IH for this case is

$$\Gamma, p_0{:}A \Vdash [p_0/y]b : B \quad (p_0 \sharp b) \quad (\text{also } p_0 \sharp \Gamma)$$

but to use the INTRO rule for $\Vdash$ we need the premise

$$\forall p\,.\, p \sharp \Gamma \implies \Gamma, p{:}A \Vdash [p/y]b : B$$

- How to reason from a particular parameter to all parameters?

Proof continued: $\Gamma \vdash a : A \implies \Gamma \Vdash a : A$

We solve the problem using swapping, as in the *weakening* proof.

- As a lemma, have equivariance of $\Vdash$:

$$\Gamma \Vdash a : A \implies \forall p\, q \,.\, (p\, q)\bullet\Gamma \Vdash (p\, q)\bullet a : A. \tag{2}$$

  Nominal Isabelle can prove this automatically, and provides the lemmas about $\sharp$ and $(-\,-)\bullet -$ that we need.

- We are trying to prove

$$\forall p \,.\, p \sharp \Gamma \implies \Gamma, p{:}A \Vdash [p/y]b : B \,.$$

  So pick $p \sharp \Gamma$. (Hence $p \sharp b$.)

- From IH and (2) have

$$(p\, p_0)\bullet(\Gamma, p_0{:}A) \Vdash (p\, p_0)\bullet([p_0/y]b) : B$$

  i.e. $\Gamma, p{:}A \Vdash [p/y]b : B$ as required.

Aside: Stronger inversion principles

- Rule INTRO $\dfrac{\Gamma, p{:}A \vdash [p/y]b : B \quad p \sharp b}{\Gamma \vdash [y]b : A \to B}$

  gives rise (by induction) to an *inversion principle*:

  $$\Gamma \vdash [y]b : T \implies$$
  $$\exists A, B, p \,.\, \Gamma, p{:}A \vdash [p/y]b : B \wedge p \sharp b \wedge T = A \to B\,.$$

- Rule INTRO $\dfrac{\forall p \,.\, p \sharp \Gamma \implies \Gamma, p{:}A \Vdash [p/y]b : B}{\Gamma \Vdash [y]b : A \to B}$

  gives a stronger inversion principle (using $\vdash \Leftrightarrow \Vdash$):

  $$\Gamma \vdash [y]b : T \implies$$
  $$\exists A, B \,.\, \forall p \sharp \Gamma \,.\, \Gamma, p{:}A \vdash [p/y]b : B \wedge p \sharp b \wedge T = A \to B\,.$$

- Some proofs need stronger inversion.

Outline

Limitations of *vclosed* representation

- Feasible way to work with *representatives* of α classes ...
  - Never needed to define α-conversion in [McKinna/Pollack].
  - Can prove parallel-β-conversion is confluent.
- ... but Church–Rosser for β-reduction fails on-the-nose for *vclosed*.
  - If we wanted to reason about β-reduction we would need to reason about α-conversion.
- *vclosed* representation not canonical: $[x]x \neq [y]y$.
- We can get canonical representations:
  - using de Bruijn indexes (locally nameless)
  - or by choosing bound variable names canonically (Sato representation).
- In both these approaches, use strengthened induction principles as above.

Locally Nameless Representation

- This is described in detail [POPL 2008] and in a JAR paper by Arthur Charguéraud.
- Charguéraud's paper comes with a Coq library and many examples.
  - Arthur's scripts work in Coq 8.3pl2 (26 Oct 2011), but there are some assumtions waitng to be filled in with proofs.
- Use de Bruijn indexes (numbers, n) for local (bound) variables.
- Use $X, Y, \ldots$ for global parameters, as above.
- Datatype of raw *terms*: $M ::= X \mid n \mid P{\cdot}Q \mid []M$

Locally Nameless

- Substitution, $[M/p]N$ is defined as before.
  - No need to lift the free indexes in M because *vclosed* M have no free indexes.
- "Hole filling", $[M/n]N$, is defined by structural recursion on N

$$
\begin{array}{lcl}
[M/k]i & = & \text{if } k = i \text{ then } M \text{ else } i \\
[M/k]X & = & X \\
[M/k]N_1 \cdot N_2 & = & ([M/k]N_1) \cdot [M/k]N_2 \\
[M/k]([]N) & = & []([M/k+1]N)
\end{array}
$$

  - This index arithmetic does make reasoning more difficult.

$$
\frac{}{\textit{vclosed}\ X} \qquad \frac{\textit{vclosed}\ M \quad \textit{vclosed}\ N}{\textit{vclosed}\ M \cdot N} \qquad \frac{\textit{vclosed}\ [X/0]M}{\textit{vclosed}\ []M}
$$

- I leave you to look up the details.

A Canonical Locally Named Representation

- Consider again the *vclosed* rules:

$$\frac{}{\mathit{vclosed}\ X} \qquad \frac{\mathit{vclosed}\ M \quad \mathit{vclosed}\ N}{\mathit{vclosed}\ M \cdot N} \qquad \frac{\mathit{vclosed}\ M}{\mathit{vclosed}\ [x][x/X]M}$$

  Local variable 'x' not determined in the rule for abstraction.
- To define a canonical subset $\mathbb{L}_\mathrm{F}$, choose 'x' deterministically:

$$\frac{}{X : \mathbb{L}_\mathrm{F}} \qquad \frac{M : \mathbb{L}_\mathrm{F} \quad N : \mathbb{L}_\mathrm{F}}{M \cdot N : \mathbb{L}_\mathrm{F}} \qquad \frac{M : \mathbb{L}_\mathrm{F} \quad x = \mathrm{F}_X(M)}{[x][x/X]M : \mathbb{L}_\mathrm{F}}$$

  parameterized by a height function $\mathrm{F} : \mathbb{X} \times \mathbb{S} \to \mathbb{V}$.
- Clearly $M : \mathbb{L}_\mathrm{F} \Longrightarrow \mathit{vclosed}\ M$, so substitution is capture free.
- Not obvious that $\mathbb{L}_\mathrm{F}$ is closed under substitution.
- Still to do: specify F such that $\mathbb{L}_\mathrm{F}$ well behaved.

Improve Notation

- Everything is parameterised by a height function F; drop explicit subscript.
- Considering rule:

$$\frac{M : \mathbb{L} \qquad x = \mathrm{F}_X(M)}{[x][x/X]M : \mathbb{L}}$$

define "abstraction":

$$\mathrm{abs}_X M \triangleq [\mathrm{F}_X(M)][\mathrm{F}_X(M)/X]M.$$

- Abstraction rule can now be written more abstractly.

$$\frac{}{X : \mathbb{L}} \qquad \frac{M : \mathbb{L} \qquad N : \mathbb{L}}{M \bullet N : \mathbb{L}} \qquad \frac{M : \mathbb{L}}{\mathrm{abs}_X M : \mathbb{L}}$$

- $\mathrm{abs}_X M$ will behave like informal '$\lambda X.M$'.
  - X does not occur in $\mathrm{abs}_X M$; $\mathrm{abs}_X X \equiv \mathrm{abs}_Y Y$.

A Datatype of Lambda Terms?

Assume good properties for F (to be discussed below).

- $\mathbb{L}$ can be formalized as a type
  - $\mathbb{L}$ is a decidable predicate on $\mathbb{S}$;
    - can be a proof irrelevant Σ-type in Type Theory.
  - $\mathbb{L}$ is a non-empty predicate on $\mathbb{S}$;
    - can be a defined type in HOL.
- But no amount of clever indexing ..., inductive-recursive ... can make $\mathbb{L}$ into a datatype ...
- ... because 'abs' isn't injective.

Another topic for another talk.

Three Good Properties of $\mathrm{F} : \mathbb{X} \times \mathbb{S} \to \mathbb{V}$

(HE) F is equivariant: Let π be a permutation over $\mathbb{X}$, then

$$M : \mathbb{L} \implies \mathrm{F}_X(M) = \mathrm{F}_{\pi \cdot X}(\pi \cdot M).$$

(HP) F is preserved by substitution:

$$M : \mathbb{L} \ \wedge \ Q : \mathbb{L} \ \wedge \ X \neq Y \ \wedge \ X \sharp Q \implies \mathrm{F}_X(M) = \mathrm{F}_X([Q/Y]M).$$

(HF) $\mathrm{F}_X(M)$ does not occur in binding position on any path from the root of M to any occurrence of X in M.

$$M : \mathbb{L} \implies \mathrm{F}_X(M) \notin \mathrm{E}_X(M)$$

where $\mathrm{E}_X(M) : \mathbb{X} \times \mathbb{S} \to (\mathbb{V}\ \mathrm{set})$ is defined:

$$\begin{array}{rcll}
\mathrm{E}_X(\alpha) & \triangleq & \{\} & \text{if } \alpha \text{ is atomic} \\
\mathrm{E}_X(M \cdot N) & \triangleq & \mathrm{E}_X(M) \cup \mathrm{E}_X(N) & \\
\mathrm{E}_X([x]M) & \triangleq & \begin{cases} \{\} & \text{if } X \sharp M \\ \{x\} \cup \mathrm{E}_X(M) & \text{otherwise} \end{cases} &
\end{array}$$

Consistency and independence of goodness

- Consistency: A good height function exists.
  - Interpret $\mathbb{V}$ as natural numbers, $\mathbb{N}$.
  - $\mathrm{H} : \mathbb{X} \times \mathbb{S} \to \mathbb{N}$ defined by structural recursion:

$$
\begin{array}{rcl}
\mathrm{H}_X(Y) & \triangleq & \begin{cases} 1 & \text{if } X = Y \\ 0 & \text{if } X \neq Y \end{cases} \\
\mathrm{H}_X(x) & \triangleq & 0 \\
\mathrm{H}_X(M{\cdot}N) & \triangleq & \max(\mathrm{H}_X(M), \mathrm{H}_X(N)) \\
\mathrm{H}_X([x]M) & \triangleq & \begin{cases} \mathrm{H}_X(M) & \text{if } \mathrm{H}_X(M) = 0 \text{ or } \mathrm{H}_X(M) > x \\ x + 1 & \text{otherwise} \end{cases}
\end{array}
$$

- Independence: No two of (HE), (HP) and (HF) imply the third.

Proof by examples; see JAR paper.

$\mathsf{abs}_X M$ behaves like abstraction should

We develop a theory of good F. For example:

- From (HE)

$$\text{“}\lambda X.M \overset{\alpha}{\sim} \lambda Y.N\text{”} \implies \mathsf{abs}_X M = \mathsf{abs}_Y N$$

- From (HF)

$$\mathsf{abs}_X M = \mathsf{abs}_Y N \implies \text{“}\lambda X.M \overset{\alpha}{\sim} \lambda Y.N\text{”}$$

- From (HP), substitution “under a binder”

$$X \neq Y \ \wedge \ X \sharp Q \implies [Q/Y]\mathsf{abs}_X M = \mathsf{abs}_X [Q/Y]M$$

Together, (HE), (HF) and (HP) show ‘ abs ’ behaves correctly for α -conversion and substitution.

Is the representation "adequate"?

Formal vs informal Relationship between a formal thing and an informal thing is not formalizable.

Formal vs formal Adequacy of representation of one formal thing by another formal thing depends on which properties we intend to preserve.

We show formally that $\mathbb{L}$ is an adequate representation of pure λ-terms in Nominal Isabelle.

- Let A, B, C range over nominal terms

$$A ::= X \mid B{\cdot}C \mid [X]A$$

- One more definition: 'instantiation' (nominal and Sato terms).

$$([x]M)\triangledown N \triangleq [N/x]M \qquad ([X]A)\blacktriangledown B \triangleq A[X::=B]$$

Isomorphism with Nominal Lambda Terms

- Define a *representation function* by "structural recursion":

$$
\begin{aligned}
!X &\triangleq X \\
!(A \bullet B) &\triangleq !A \bullet !B \\
![X]A &\triangleq \mathrm{abs}_X !A
\end{aligned}
$$

  - Need (HE) (F equivariant), to show $!$ is a function.
- Assuming F is good, $!$ is a bijection that preserves substitution and instantiation:

$$
\begin{array}{rclr}
M : \mathbb{L} & \Longrightarrow & \exists A.\ !A = M & \text{! is surjective,} \\
!A = !B & \Longrightarrow & A = B & \text{! is injective,} \\
!(A[X ::= B]) & = & [!B/X]!A & \text{! respects substitution,} \\
!(([X]A) \blacktriangledown Y) & = & (\mathrm{abs}_X !A) \triangledown Y & \text{! respects instantiation} \\
& & & \text{by parameters.}
\end{array}
$$

A Converse

- Assume ! is a bijection that preserves instantiation. Then (HE), (HP) and (HF) hold.
- (Thus preservation of substitution is not independent of the other properties. There is still something to figure out here.)

Example: β-reduction

$$\frac{P : \mathbb{L} \qquad N : \mathbb{L}}{(\mathrm{abs}_{X} P) \cdot N \to (\mathrm{abs}_{X} P) \triangledown N} \ (\beta)$$

$$\frac{M_1 \to M_2 \qquad N : \mathbb{L}}{M_1 \cdot N \to M_2 \cdot N} \qquad \frac{M : \mathbb{L} \qquad N_1 \to N_2}{M \cdot N_1 \to M \cdot N_2}$$

$$\frac{M \to N}{\mathrm{abs}_{X} M \to \mathrm{abs}_{X} N} \ (\xi)$$

- $\to$ is well behaved, e.g.
  - $\to$ is equivariant.
  - Reduction and well-formedness: $M \to N \implies M : \mathbb{L} \wedge N : \mathbb{L}$.
  - Reduction respects representation: $A \to B \Leftrightarrow\ !A \to !B$