Proofs as Terms, Terms as Graphs

Jui-Hsuan Wu (Ray)

LIX, Institut Polytechnique de Paris & Inria Saclay

APLAS 2023, Taipei, Taiwan

27 November 2023

Outline

Proof as terms

Proofs as terms

- Proof theory has been widely used in studying terms and programs, often via Curry-Howard correspondence.
- Which proof system to choose?
  Natural deduction: not sophisticated enough
  Sequent calculus: too little structure and too many redundancies
- Focusing: a light canonical form for (sequent) proofs with more structure
- Focused proof system *LJF* for Gentzen's LJ: Focusing and polarization
  - Connectives and atomic formulas are polarized
  - Different polarizaions do not affect provability, but they induce different forms of proofs
    $\hookrightarrow$ different styles of term structures [1]

[1] Dale Miller and Jui-Hsuan Wu. **A positive perspective on term representation**. CSL 2023.

Two encodings of untyped λ-terms

Using *LJF*, with the two axioms $D \supset D \supset D$ and $(D \supset D) \supset D$ where D is atomic, and by considering only sequents of the form:

$$D, \ldots, D \vdash D$$

we have the following rules:

D is given the negative polarity

$$D \in \Gamma \quad \frac{}{\Gamma \vdash D}\ nvar$$

$$\frac{\Gamma \vdash D \qquad \Gamma \vdash D}{\Gamma \vdash D}\ napp$$

$$\frac{\Gamma, D \vdash D}{\Gamma \vdash D}\ nabs$$

D is given the positive polarity

$$D \in \Gamma \quad \frac{}{\Gamma \vdash D}\ pvar$$

$$\{D, D\} \subseteq \Gamma \quad \frac{\Gamma, D \vdash D}{\Gamma \vdash D}\ papp$$

$$\frac{\Gamma, D \vdash D \qquad \Gamma, D \vdash D}{\Gamma \vdash D}\ pabs$$

Two encodings of untyped λ-terms

Using *LJF*, with the two axioms $D \supset D \supset D$ and $(D \supset D) \supset D$ where D is atomic, and by considering only sequents of the form:

$$D, \ldots, D \vdash D$$

we have the following rules:

D is given the negative polarity

$$\frac{x : D \in \Gamma}{\Gamma \vdash x : D}\ nvar$$

$$\frac{\Gamma \vdash s : D \qquad \Gamma \vdash t : D}{\Gamma \vdash st : D}\ napp$$

$$\frac{\Gamma, x : D \vdash t : D}{\Gamma \vdash \lambda x.t : D}\ nabs$$

D is given the positive polarity

$$D \in \Gamma \quad \frac{}{\Gamma \vdash D}\ pvar$$

$$\{D, D\} \subseteq \Gamma \quad \frac{\Gamma, D \vdash D}{\Gamma \vdash D}\ papp$$

$$\frac{\Gamma, D \vdash D \qquad \Gamma, D \vdash D}{\Gamma \vdash D}\ pabs$$

Two encodings of untyped λ-terms

Using *LJF*, with the two axioms $D \supset D \supset D$ and $(D \supset D) \supset D$ where D is atomic, and by considering only sequents of the form:

$$D, \ldots, D \vdash D$$

we have the following rules:

D is given the negative polarity

$$\frac{x : D \in \Gamma}{\Gamma \vdash x : D}\ nvar$$

$$\frac{\Gamma \vdash s : D \qquad \Gamma \vdash t : D}{\Gamma \vdash st : D}\ napp$$

$$\frac{\Gamma, x : D \vdash t : D}{\Gamma \vdash \lambda x.t : D}\ nabs$$

D is given the positive polarity

$$\frac{x : D \in \Gamma}{\Gamma \vdash x : D}\ pvar$$

$$\frac{\{y : D, z : D\} \subseteq \Gamma \qquad \Gamma, x : D \vdash t : D}{\Gamma \vdash t[x \leftarrow yz] : D}\ papp$$

$$\frac{\Gamma, y : D \vdash s : D \qquad \Gamma, x : D \vdash t : D}{\Gamma \vdash t[x \leftarrow \lambda y.s] : D}\ pabs$$

Two encodings of untyped λ-terms

Using *LJF*, with the two axioms $D \supset D \supset D$ and $(D \supset D) \supset D$ where D is atomic, and by considering only sequents of the form:

$$D, \ldots, D \vdash D$$

we have the following rules:

D is given the negative polarity

$$\frac{}{\Gamma \vdash x : D}\; nvar \quad x : D \in \Gamma$$

$$\frac{\Gamma \vdash s : D \qquad \Gamma \vdash t : D}{\Gamma \vdash st : D}\; napp$$

$$\frac{\Gamma, x : D \vdash t : D}{\Gamma \vdash \lambda x.t : D}\; nabs$$

negative bias syntax

D is given the positive polarity

$$\frac{}{\Gamma \vdash x : D}\; pvar \quad x : D \in \Gamma$$

$$\frac{\Gamma, x : D \vdash t : D}{\Gamma \vdash t[x \leftarrow yz] : D}\; papp \quad \{y : D, z : D\} \subseteq \Gamma$$

$$\frac{\Gamma, y : D \vdash s : D \qquad \Gamma, x : D \vdash t : D}{\Gamma \vdash t[x \leftarrow \lambda y.s] : D}\; pabs$$

positive bias syntax

Two encodings of untyped λ-terms

- The **negative bias syntax** corresponds to the usual representation of untyped λ-terms: tree-structure, top-down
- The **positive bias syntax** gives a term structure where sharing is possible via named structures, or explicit substitutions: DAG, bottom-up
- What does cut-elimination do in these two cases?
  - Terms considered here correspond to cut-free proofs.
  - Cut-elimination $\neq$ Computation
  - If we introduce a cut between two cut-free proofs, cut-elimination provides a natural notion of **substitution**. As expected, in the negative case, the cut-elimination procedure of *LJF* yields the usual **meta-level substitution** for untyped λ-terms. What about the positive case?

positive λ-calculus

The positive λ-calculus

In the following, we are interested in the positive bias syntax.
Fix a set $\textsc{name} = \{x, y, z, \ldots\}$ of **names** (or **variables**). Terms, contexts and left contexts are defined as follows:

$$\begin{array}{rcl}
\textsc{Terms} & s, t & := x \mid t[x \leftarrow yz] \mid t[x \leftarrow \lambda y.s] \\
\textsc{Contexts} & C & := \Box \mid C[x \leftarrow yz] \mid C[x \leftarrow \lambda y.s] \mid t[x \leftarrow \lambda y.C] \\
\textsc{Left Contexts} & L & := \Box \mid L[x \leftarrow yz] \mid L[x \leftarrow \lambda y.s]
\end{array}$$

A term can be viewed as a list of **named structures** (or **explicit substitutions**) followed by a variable. Also note that every term can be written uniquely (up to α-equivalence) as $L\langle x\rangle$ for some left context L and variable x.

The positive λ-calculus: Structural equivalence

If two named structures are independent of each other, we should be able to permute them. By defining $fv(yz) = \{y, z\}$ and $fv(\lambda y.s) = fv(s) \setminus \{y\}$, this can be expressed using the equation:

$$t[x_1 \leftarrow p_1][x_2 \leftarrow p_2] \;\sim_{\mathtt{str}}\; t[x_2 \leftarrow p_2][x_1 \leftarrow p_1] \qquad \text{if } x_1 \notin fv(p_2) \text{ and } x_2 \notin fv(p_1)$$

Definition (Structural equivalence)

We define an equivalence relation $\equiv_{\mathtt{str}}$ on terms, called the **structural equivalence**, as the smallest congruence containing $\sim_{\mathtt{str}}$.

The positive λ-calculus: Substitution

Definition (Substitution on terms)

Let t, u be terms and x a name such that $x \notin fv(u)$. We define the result of substituting u for x in t, written $t[x/u]$, as follows:

$$
\begin{aligned}
t[x/y] &= t\{x/y\} \\
t[x/s[y \leftarrow zw]] &= t[x/s][y \leftarrow zw] \\
t[x/s[y \leftarrow \lambda z.u]] &= t[x/s][y \leftarrow \lambda z.u]
\end{aligned}
$$

Note that by expressing the term u uniquely as $L\langle y\rangle$, we have $t[x/u] = L\langle t\{x/y\}\rangle$ by a straightforward induction.

An example:
Let t be the term $y[y \leftarrow \lambda z.w[w \leftarrow za]][x \leftarrow aa]$ and u the term $a_2[a_2 \leftarrow a_1a_1][a_1 \leftarrow a_0a_0]$. Then

$$t[a/u] = y[y \leftarrow \lambda z.w[w \leftarrow za_2]][x \leftarrow a_2a_2][a_2 \leftarrow a_1a_1][a_1 \leftarrow a_0a_0]$$

The positive λ-calculus: Unfolding

How to compare a term of the positive λ-calculus with a usual λ-term?

We can unfold all the named structures.

Definition (Unfolding)

The **unfolding** $\underline{t}$ of a term t is the untyped λ-term defined as follows:

$$\underline{x} = x \qquad \underline{t[x \leftarrow yz]} = \underline{t}\{x/yz\} \qquad \underline{t[x \leftarrow \lambda y.s]} = \underline{t}\{x/\lambda y.\underline{s}\}$$

where $\{\cdot/\cdot\}$ is the meta-level substitution of untyped λ-terms.

Note that, this definition can also be justified by manipulating *LJF* proofs via cut-elimination.

The positive λ-calculus: Reduction

How should we evaluate a term t in the positive λ-calculus?

A possible way is to compute its unfolding $\underline{t}$ and evaluate it in the untyped λ-calculus. In this case, we can refer to the β-normal form of $\underline{t}$ (if it exists) as the *meaning* of t.

However, this can be costly as the unfolding of a term might have exponential size with respect to the original term.

As a result, we look for a reduction system for the positive λ-calculus that is compatible with the β-reduction in the untyped λ-calculus.

The positive λ-calculus: Reduction

We propose the following beta-rule and gc-rule.

$$C\langle t[z \leftarrow xw]\rangle[x \leftarrow \lambda y.L\langle y'\rangle] \mapsto_{\mathtt{beta}} C\langle L\langle t\{z/y'\}\rangle\{y/w\}\rangle[x \leftarrow \lambda y.L\langle y'\rangle]$$
$$t[x \leftarrow \lambda y.s] \mapsto_{\mathtt{gc}} t \qquad\qquad \text{if } x \notin fv(t)$$

How we define the beta-rule:

1. for a given term t, consider its corresponding (cut-free) proof Π
2. identify a certain pattern (that actually corresponds to a beta-redex) in Π and transform the proof into a proof with cut Π'
3. apply cut-elimination to Π'

The positive λ-calculus: Reduction

$$
\begin{array}{rcll}
C\langle t[z \leftarrow xw]\rangle[x \leftarrow \lambda y.L\langle y'\rangle] & \mapsto_{\mathtt{beta}} & C\langle L\langle t\{z/y'\}\rangle\{y/w\}\rangle[x \leftarrow \lambda y.L\langle y'\rangle] & \\
t[x \leftarrow \lambda y.s] & \mapsto_{\mathtt{gc}} & t & \text{if } x \notin fv(t)
\end{array}
$$

An example:

$$
\begin{array}{rl}
 & x_2[x_2 \leftarrow gx_1][x_1 \leftarrow fx_0][f \leftarrow \lambda x.z[z \leftarrow yy][y \leftarrow xx]] \\
\rightarrow_{\mathtt{beta}} & x_2[x_2 \leftarrow gz_1][z_1 \leftarrow y_1y_1][y_1 \leftarrow x_0x_0][f \leftarrow \lambda x.z[z \leftarrow yy][y \leftarrow xx]] \\
\rightarrow_{\mathtt{gc}} & x_2[x_2 \leftarrow gz_1][z_1 \leftarrow y_1y_1][y_1 \leftarrow x_0x_0]
\end{array}
$$

We define $\rightarrow_{\mathtt{pos}}$ as $\rightarrow_{\mathtt{beta}} \cup \rightarrow_{\mathtt{gc}}$.

The positive λ-calculus: Reduction

$$
\begin{array}{rcll}
C\langle t[z \leftarrow xw]\rangle[x \leftarrow \lambda y.L\langle y'\rangle] & \mapsto_{\mathtt{beta}} & C\langle L\langle t\{z/y'\}\rangle\{y/w\}\rangle[x \leftarrow \lambda y.L\langle y'\rangle] & \\
t[x \leftarrow \lambda y.s] & \mapsto_{\mathtt{gc}} & t & \text{if } x \notin fv(t)
\end{array}
$$

Proposition

Let s and t be terms such that $s \rightarrow_{\mathtt{pos}} t$. Then $\underline{s} \rightarrow_{\beta}^{*} \underline{t}$.

Proposition

If s is a normal term with respect to $\rightarrow_{\mathtt{pos}}$, then $\underline{s}$ is β-normal.

The positive λ-calculus and the VSC

The value substitution calculus (VSC) is a call-by-value λ-calculus with explicit substitutions proposed by Accattoli and Paolini.

The syntax and the reduction rules of the VSC are shown below:

$$
\begin{array}{rrl}
\textsc{Terms} & t, u & := v \mid tu \mid t[x \leftarrow u] \\
\textsc{Values} & v & := x \mid \lambda x.t \\
\textsc{Contexts} & C & := \square \mid tC \mid Ct \mid \lambda x.C \mid C[x \leftarrow t] \mid t[x \leftarrow C] \\
\textsc{Left Contexts} & L & := \square \mid L[x \leftarrow t]
\end{array}
$$

$$
\begin{array}{rcl}
L\langle \lambda x.t \rangle u & \mapsto_{\mathtt{m}} & L\langle t[x \leftarrow u] \rangle \\
t[x \leftarrow L\langle v \rangle] & \mapsto_{\mathtt{e}} & L\langle t\{x/v\} \rangle
\end{array}
$$

It is easy to see that all terms and contexts of the positive λ-calculus are included in the VSC.

Usefulness

For example, consider the term

$$t = w[w \leftarrow fx][f \leftarrow \lambda z_0.z_3[z_3 \leftarrow G(z_2)][z_2 \leftarrow G(z_1)][z_1 \leftarrow G(z_0)]][x \leftarrow \lambda y.s].$$

where $G(t) = \lambda w_0.w_3[w_3 \leftarrow w_1 w_2][w_2 \leftarrow gt][w_1 \leftarrow gt]$ with g a fixed name and s a normal term in positive λ-calculus. After one `beta`-step and one `gc`-step, we obtain a normal term

$$z_3'[z_3' \leftarrow G(z_2')][z_2' \leftarrow G(z_1')][z_1' \leftarrow G(x)][x \leftarrow \lambda y.s].$$

in the positive λ-calculus. However, in the VSC, we have

$$\begin{array}{ll} z_3'[z_3' \leftarrow G(z_2')][z_2' \leftarrow G(z_1')][z_1' \leftarrow G(x)][x \leftarrow \lambda y.s] & \rightarrow_{\mathsf{e}} \\ z_3'[z_3' \leftarrow G(z_2')][z_2' \leftarrow G(z_1')][z_1' \leftarrow G(\lambda y.s)] & \rightarrow_{\mathsf{e}} \\ z_3'[z_3' \leftarrow G(z_2')][z_2' \leftarrow G(G(\lambda y.s))] & \rightarrow_{\mathsf{e}} \\ z_3'[z_3' \leftarrow G(G(G(\lambda y.s)))] & \rightarrow_{\mathsf{e}} \\ G(G(G(\lambda y.s))) & \end{array}$$

The positive λ-calculus and the VSC

We can actually consider a variant of the VSC, called **micro-step** as substitutions are treated one by one instead of using meta-level substitution.

$$
\begin{array}{rcll}
L\langle \lambda x.t\rangle u & \mapsto_{\mathtt{m}} & L\langle t[x \leftarrow u]\rangle & \\
C\langle x\rangle[x \leftarrow L\langle v\rangle] & \mapsto_{\mathsf{e}'} & L\langle C\langle v\rangle[x \leftarrow v]\rangle & \\
t[x \leftarrow L\langle v\rangle] & \mapsto_{\mathsf{gc}'} & t & \text{if } x \notin fv(t)
\end{array}
$$

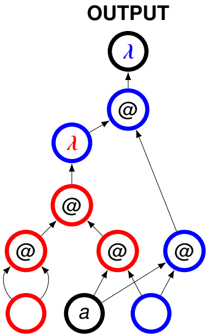
The `beta`-rule can actually be simulated by the VSC as follows:

$$
\begin{array}{r}
C\langle t[z \leftarrow xw]\rangle[x \leftarrow \lambda y.L\langle y'\rangle] \to_{\mathsf{e}'} \\
C\langle t[z \leftarrow (\lambda y.L\langle y'\rangle)w]\rangle[x \leftarrow \lambda y.L\langle y'\rangle] \to_{\mathtt{m}} \\
C\langle t[z \leftarrow L\langle y'\rangle[y \leftarrow w]]\rangle[x \leftarrow \lambda y.L\langle y'\rangle] \to_{\mathsf{e}'}^{*}\to_{\mathsf{gc}'} \\
C\langle t[z \leftarrow L\langle y'\rangle\{y/w\}]\rangle[x \leftarrow \lambda y.L\langle y'\rangle] \to_{\mathsf{e}'}^{*}\to_{\mathsf{gc}'} \\
C\langle L\langle t\{z/y'\}\rangle\{y/w\}\rangle[x \leftarrow \lambda y.L\langle y'\rangle]
\end{array}
$$

λ-graphs with bodies

λ-graphs with bodies

We also propose a graphical representation for the positive λ-calculus.



$$n_1[n_1 \leftarrow (\lambda b.b_3[b_3 \leftarrow b_2 b_1][b_2 \leftarrow (\lambda r.r_3[r_3 \leftarrow r_1 r_2][r_2 \leftarrow ab][r_1 \leftarrow rr])][b_1 \leftarrow ab])]$$

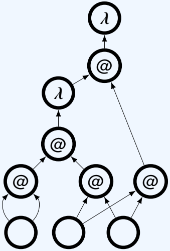
$$n_1[n_1 \leftarrow (\lambda b.b_3[b_3 \leftarrow b_2 b_1][b_1 \leftarrow ab][b_2 \leftarrow (\lambda r.r_3[r_3 \leftarrow r_1 r_2][r_1 \leftarrow rr][r_2 \leftarrow ab])])]$$

λ-graphs with bodies: Definition

Definition

A **pre-graph** is a DAG built with the following three kinds of nodes:

- **Application**: an application node is labeled with `@` and has two incoming edges (left and right).
- **Abstraction**: an abstraction node is labeled with λ and has one incoming edge.
- **Variable**: a variable node has no incoming edge.

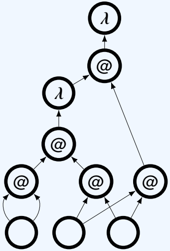


λ-graphs with bodies: Definition

Definition

An **unlabeled λ-graph with bodies** is a pre-graph $\mathcal{G}$ together with two functions $bv : \Lambda_{\mathcal{G}} \to \mathcal{V}_{\mathcal{G}}$ and $body : \Lambda_{\mathcal{G}} \to 2^{\mathcal{N}_{\mathcal{G}} \setminus \mathcal{V}_{\mathcal{G}}}$ ($\Lambda_{\mathcal{G}}$: abstraction nodes of $\mathcal{G}$, $\mathcal{V}_{\mathcal{G}}$: variable nodes of $\mathcal{G}$) such that:

1. $body(l) \cap body(l') = \emptyset$ for $l \neq l'$.
2. $\mathcal{B}_{\mathcal{G}} = (\Lambda_{\mathcal{G}}, \{(l, l') \mid l, l' \in \Lambda_{\mathcal{G}}, l \in body(l')\})$, called the **scope graph** of $\mathcal{G}$, is a DAG.
3. If $n = bv(l)$ or $n \in body(l)$ and $(n, m) \in \mathcal{E}_{\mathcal{G}}$, then we have
    - $m = l$, or
    - $m \in body(l')$ s.t. there is a path from l' to l in $\mathcal{B}_{\mathcal{G}}$.

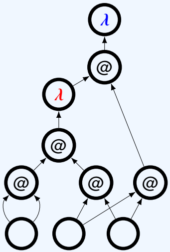


λ-graphs with bodies: Definition

Definition

An **unlabeled λ-graph with bodies** is a pre-graph $\mathcal{G}$ together with two functions $bv : \Lambda_{\mathcal{G}} \to \mathcal{V}_{\mathcal{G}}$ and $body : \Lambda_{\mathcal{G}} \to 2^{\mathcal{N}_{\mathcal{G}} \setminus \mathcal{V}_{\mathcal{G}}}$ ($\Lambda_{\mathcal{G}}$: abstraction nodes of $\mathcal{G}$, $\mathcal{V}_{\mathcal{G}}$: variable nodes of $\mathcal{G}$) such that:

1. $body(l) \cap body(l') = \emptyset$ for $l \neq l'$.
2. $\mathcal{B}_{\mathcal{G}} = (\Lambda_{\mathcal{G}}, \{(l, l') \mid l, l' \in \Lambda_{\mathcal{G}}, l \in body(l')\})$, called the **scope graph** of $\mathcal{G}$, is a DAG.
3. If $n = bv(l)$ or $n \in body(l)$ and $(n, m) \in \mathcal{E}_{\mathcal{G}}$, then we have
   - $m = l$, or
   - $m \in body(l')$ s.t. there is a path from l' to l in $\mathcal{B}_{\mathcal{G}}$.

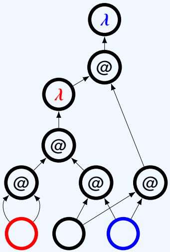


λ-graphs with bodies: Definition

Definition

An **unlabeled λ-graph with bodies** is a pre-graph $\mathcal{G}$ together with two functions $bv : \Lambda_{\mathcal{G}} \to \mathcal{V}_{\mathcal{G}}$ and $body : \Lambda_{\mathcal{G}} \to 2^{\mathcal{N}_{\mathcal{G}} \setminus \mathcal{V}_{\mathcal{G}}}$ ($\Lambda_{\mathcal{G}}$: abstraction nodes of $\mathcal{G}$, $\mathcal{V}_{\mathcal{G}}$: variable nodes of $\mathcal{G}$) such that:

1. $body(l) \cap body(l') = \emptyset$ for $l \neq l'$.
2. $\mathcal{B}_{\mathcal{G}} = (\Lambda_{\mathcal{G}}, \{(l, l') \mid l, l' \in \Lambda_{\mathcal{G}}, l \in body(l')\})$, called the **scope graph** of $\mathcal{G}$, is a DAG.
3. If $n = bv(l)$ or $n \in body(l)$ and $(n, m) \in \mathcal{E}_{\mathcal{G}}$, then we have
   - $m = l$, or
   - $m \in body(l')$ s.t. there is a path from l' to l in $\mathcal{B}_{\mathcal{G}}$.

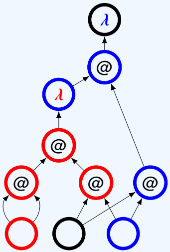


λ-graphs with bodies: Definition

Definition

An **unlabeled λ-graph with bodies** is a pre-graph $\mathcal{G}$ together with two functions $bv : \Lambda_{\mathcal{G}} \to \mathcal{V}_{\mathcal{G}}$ and $body : \Lambda_{\mathcal{G}} \to 2^{\mathcal{N}_{\mathcal{G}} \setminus \mathcal{V}_{\mathcal{G}}}$ ($\Lambda_{\mathcal{G}}$: abstraction nodes of $\mathcal{G}$, $\mathcal{V}_{\mathcal{G}}$: variable nodes of $\mathcal{G}$) such that:

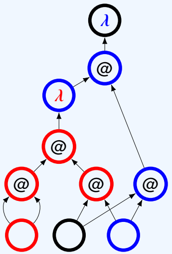
1. $body(l) \cap body(l') = \emptyset$ for $l \neq l'$.
2. $\mathcal{B}_{\mathcal{G}} = (\Lambda_{\mathcal{G}}, \{(l, l') \mid l, l' \in \Lambda_{\mathcal{G}}, l \in body(l')\})$, called the **scope graph** of $\mathcal{G}$, is a DAG.
3. If $n = bv(l)$ or $n \in body(l)$ and $(n, m) \in \mathcal{E}_{\mathcal{G}}$, then we have
   - $m = l$, or
   - $m \in body(l')$ s.t. there is a path from l' to l in $\mathcal{B}_{\mathcal{G}}$.



λ-graphs with bodies: Definition

Definition

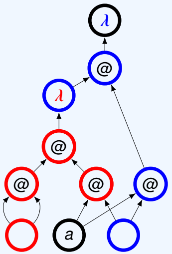
- A **λ-graph with bodies** is an unlabeled λ-graph with bodies with a unique label assigned to each free variable node, and with a global node chosen, called the **output** of the λ-graph with bodies.
- A **Σ-λ-graph with bodies** is a λ-graph with bodies with a free variable node labeled by each element of a signature Σ.



λ-graphs with bodies: Definition

Definition

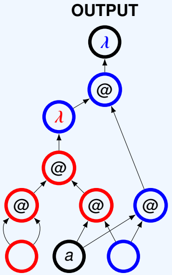
- A **λ-graph with bodies** is an unlabeled λ-graph with bodies with a unique label assigned to each free variable node, and with a global node chosen, called the **output** of the λ-graph with bodies.
- A **Σ-λ-graph with bodies** is a λ-graph with bodies with a free variable node labeled by each element of a signature Σ.



λ-graphs with bodies: Definition

Definition

- A **λ-graph with bodies** is an unlabeled λ-graph with bodies with a unique label assigned to each free variable node, and with a global node chosen, called the **output** of the λ-graph with bodies.
- A **Σ-λ-graph with bodies** is a λ-graph with bodies with a free variable node labeled by each element of a signature Σ.



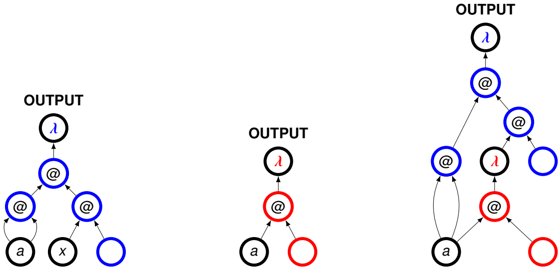
λ-graphs with bodies and terms

λ-graphs with bodies capture the structural equivalence on terms.

Theorem

We have a one-to-one correspondence between Σ-λ-graphs with bodies and Σ-terms up to $\equiv_{\mathtt{str}}$.

Substitution on λ-graphs with bodies can be defined in a straightforward way:



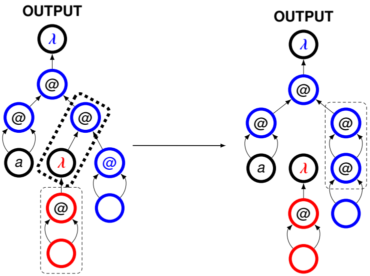
λ-graphs with bodies: Reduction

Definition

Let $\mathcal{G}$ be a λ-graph with bodies and l an abstraction node. We define the **box** of l as the union of bodies together with their bound variable nodes *below* l:

$$box(l) = \bigcup_{l' \rightsquigarrow l \text{ in } \mathcal{B}_{\mathcal{G}}} (body(l') \cup \{bv(l')\})$$

Reduction can then be defined by duplicating boxes and by applying substitutions.



Generalization and Conclusion

Generalization

Here, we use two specific axioms $D \supset D \supset D$ and $(D \supset D) \supset D$ to provide encodings for untyped λ-terms.

In fact, thanks to *LJF*, similar term structures can be defined using any set of formulas of order at most 2 where the order $ord(B)$ of a formula B is defined as follows:

$$ord(A) = 0 \qquad\qquad ord(B_1 \supset B_2) = max(ord(B_1) + 1, B_2)$$

Note that $ord(D \supset D \supset D) = 1$ and $ord((D \supset D) \supset D) = 2$.

Any formula F of order at most 2 can be written as $B_1 \supset \cdots \supset B_n \supset A$ with A atomic and $ord(B_i) \leq 1$. If $ord(B_i) = 1$ for some i, then the node corresponding to F comes with a notion of body.

Conclusion

- We define the positive λ-calculus, whose reduction does not correspond to cut-elimination but is also inspired by some proof-theoretic consideration.
- The positive λ-calculus is closely related to the VSC but does *useful* substitutions of abstractions.
- λ-graphs with bodies captures the structural equivalence on terms and operations can be implemented on them in a straightforward way.
- Some future directions:
  - Explore more connections between the positive λ-calculus and the VSC using usefulness
  - Extend to the settings where mixed polarities for atoms are considered