Proofs as terms, positively

Jui-Hsuan Wu (Ray)

LIX, Ecole Polytechnique & Inria Saclay

Syntax Meets Semantics, IRIF, Paris, France

18 January 2024

Outline

Terms are important

We live in a world full of syntactic structures.

Terms (or expressions) are everywhere.
In programming languages, formal proofs, mathematical proofs, natural languages, etc.

Handling operations on terms can be tricky, especially with bindings.

- substitution
- equality checking
- evaluation
- sharing

Having a highly principled meta-theory is important for further studies.

Terms are important

We live in a world full of syntactic structures.

Terms (or expressions) are everywhere.
In programming languages, formal proofs, mathematical proofs, natural languages, etc.

Handling operations on terms can be tricky, especially with bindings.

- substitution
- equality checking
- evaluation
- sharing

Having a highly principled meta-theory is important for further studies.

Terms are important

We live in a world full of syntactic structures.

Terms (or expressions) are everywhere.
In programming languages, formal proofs, mathematical proofs, natural languages, etc.

Handling operations on terms can be tricky, especially with bindings.

- substitution
- equality checking
- evaluation
- sharing

Having a highly principled meta-theory is important for further studies.

Terms are important

We live in a world full of syntactic structures.

Terms (or expressions) are everywhere.
In programming languages, formal proofs, mathematical proofs, natural languages, etc.

Handling operations on terms can be tricky, especially with bindings.

- substitution
- equality checking
- evaluation
- sharing

Having a highly principled meta-theory is important for further studies.

Terms are important

We live in a world full of syntactic structures.

Terms (or expressions) are everywhere.
In programming languages, formal proofs, mathematical proofs, natural languages, etc.

Handling operations on terms can be tricky, especially with bindings.

- substitution
- equality checking
- evaluation
- sharing

Having a highly principled meta-theory is important for further studies.

Proof theory and term representation

Need a highly prinicipled and mathematically sound meta-theory
$\hookrightarrow$ (structural) proof theory might help

Starting from a given proof system, we can obtain a term representation by annotating proofs.

In addition to the structure of terms,
other operations can sometimes be mimicked by operations on proofs.
$\hookrightarrow$ Curry-Howard correspondence.

Sometimes, operations on terms are not provided by proof theory for free, but they can be inspired by some proof-theoretic observations.

Proof theory and term representation

Need a highly prinicipled and mathematically sound meta-theory
$\hookrightarrow$ (structural) proof theory might help

Starting from a given proof system, we can obtain a term representation by annotating proofs.

In addition to the structure of terms,
other operations can sometimes be mimicked by operations on proofs.

$\hookrightarrow$ Curry-Howard correspondence.

Sometimes, operations on terms are not provided by proof theory for free, but they can be inspired by some proof-theoretic observations.

Proof theory and term representation

Need a highly prinicipled and mathematically sound meta-theory
$\hookrightarrow$ (structural) proof theory might help

Starting from a given proof system, we can obtain a term representation by annotating proofs.

In addition to the structure of terms,
other operations can sometimes be mimicked by operations on proofs.

$\hookrightarrow$ Curry-Howard correspondence.

Sometimes, operations on terms are not provided by proof theory for free, but they can be inspired by some proof-theoretic observations.

Proof theory and term representation

Need a highly prinicipled and mathematically sound meta-theory
$\hookrightarrow$ (structural) proof theory might help

Starting from a given proof system, we can obtain a term representation by annotating proofs.

In addition to the structure of terms,
other operations can sometimes be mimicked by operations on proofs.
$\hookrightarrow$ Curry-Howard correspondence.

Sometimes, operations on terms are not provided by proof theory for free, but they can be inspired by some proof-theoretic observations.

Proof theory and term representation

Need a highly prinicipled and mathematically sound meta-theory
$\hookrightarrow$ (structural) proof theory might help

Starting from a given proof system, we can obtain a term representation by annotating proofs.

In addition to the structure of terms,
other operations can sometimes be mimicked by operations on proofs.
$\hookrightarrow$ Curry-Howard correspondence.

Sometimes, operations on terms are not provided by proof theory for free, but they can be inspired by some proof-theoretic observations.

Proof theory and term representation

Need a highly prinicipled and mathematically sound meta-theory
$\hookrightarrow$ (structural) proof theory might help

Starting from a given proof system, we can obtain a term representation by annotating proofs.

In addition to the structure of terms,
other operations can sometimes be mimicked by operations on proofs.

$\hookrightarrow$ Curry-Howard correspondence.

Sometimes, operations on terms are not provided by proof theory for free, but they can be inspired by some proof-theoretic observations.

Focusing

First introduced by Andreoli to reduce non-determinism in proof search for linear logic.

It comes from the following observation:

Rule	invertible	$\longleftrightarrow$	non-invertible
Information	non-essential	$\longleftrightarrow$	essential
Phase	negative $\Uparrow$	$\longleftrightarrow$	positive $\Downarrow$

Focusing gives more structure to proofs.
$\hookrightarrow$ a light canonical form of proofs

Various focused proof systems:

- LKQ and LKT by Danos, Joinet, and Schellinx.
- LKF and LJF by Liang and Miller.
- $\lambda\mu\tilde{\mu}$ by Curien and Herbelin.

Focusing

First introduced by Andreoli to reduce non-determinism in proof search for linear logic.

It comes from the following observation:

Rule	invertible	$\longleftrightarrow$	non-invertible
Information	non-essential	$\longleftrightarrow$	essential
Phase	negative $\Uparrow$	$\longleftrightarrow$	positive $\Downarrow$

Focusing gives more structure to proofs.
$\hookrightarrow$ a light canonical form of proofs

Various focused proof systems:

- LKQ and LKT by Danos, Joinet, and Schellinx.
- LKF and LJF by Liang and Miller.
- $\lambda\mu\tilde{\mu}$ by Curien and Herbelin.

Focusing

First introduced by Andreoli to reduce non-determinism in proof search for linear logic.

It comes from the following observation:

Rule	invertible	$\longleftrightarrow$	non-invertible
Information	non-essential	$\longleftrightarrow$	essential
Phase	negative $\Uparrow$	$\longleftrightarrow$	positive $\Downarrow$

Focusing gives more structure to proofs.
$\hookrightarrow$ a light canonical form of proofs

Various focused proof systems:

- LKQ and LKT by Danos, Joinet, and Schellinx.
- LKF and LJF by Liang and Miller.
- $\lambda\mu\tilde{\mu}$ by Curien and Herbelin.

Focusing

First introduced by Andreoli to reduce non-determinism in proof search for linear logic.

It comes from the following observation:

Rule	invertible	$\longleftrightarrow$	non-invertible
Information	non-essential	$\longleftrightarrow$	essential
Phase	negative $\Uparrow$	$\longleftrightarrow$	positive $\Downarrow$

Focusing gives more structure to proofs.
$\hookrightarrow$ a light canonical form of proofs

Various focused proof systems:

- LKQ and LKT by Danos, Joinet, and Schellinx.
- LKF and LJF by Liang and Miller.
- $\lambda\mu\tilde{\mu}$ by Curien and Herbelin.

Focusing

First introduced by Andreoli to reduce non-determinism in proof search for linear logic.

It comes from the following observation:

Rule	invertible	$\longleftrightarrow$	non-invertible
Information	non-essential	$\longleftrightarrow$	essential
Phase	negative $\Uparrow$	$\longleftrightarrow$	positive $\Downarrow$

Focusing gives more structure to proofs.
$\hookrightarrow$ a light canonical form of proofs

Various focused proof systems:

- LKQ and LKT by Danos, Joinet, and Schellinx.
- LKF and LJF by Liang and Miller.
- $\lambda\mu\tilde{\mu}$ by Curien and Herbelin.

The LJF system (with only implications)

Formulas are built using atomic formulas and implications.

Formulas are polarized:

- Implications are negative.
- Atomic formulas can be either positive or negative.

A polarized theory is a set of formulas together with an atomic bias assignment $\delta : \mathrm{ATOM} \to \{+, -\}$.

Example: $\{(A \supset B) \supset C, A \supset A \supset B, C \supset B\}$

For a given (unpolarized) LJ sequent, different polarizations do not affect provability, but give different forms of proofs.

The LJF system (with only implications)

Formulas are built using atomic formulas and implications.

Formulas are polarized:

- Implications are negative.
- Atomic formulas can be either positive or negative.

A polarized theory is a set of formulas together with an atomic bias assignment $\delta : \mathrm{ATOM} \to \{+, -\}$.

Example: $\{(A \supset B) \supset C, A \supset A \supset B, C \supset B\}$

For a given (unpolarized) LJ sequent, different polarizations do not affect provability, but give different forms of proofs.

The LJF system (with only implications)

Formulas are built using atomic formulas and implications.

Formulas are polarized:

- Implications are negative.
- Atomic formulas can be either positive or negative.

A polarized theory is a set of formulas together with an atomic bias assignment $\delta : \mathrm{ATOM} \to \{+, -\}$.

Example: $\{(A \supset B) \supset C, A \supset A \supset B, C \supset B\}$

For a given (unpolarized) LJ sequent, different polarizations do not affect provability, but give different forms of proofs.

The LJF system (with only implications)

Formulas are built using atomic formulas and implications.

Formulas are polarized:

- Implications are negative.
- Atomic formulas can be either positive or negative.

A polarized theory is a set of formulas together with an atomic bias assignment $\delta : \mathsf{ATOM} \to \{+, -\}$.

Example: $\{(A \supset B) \supset C, A \supset A \supset B, C \supset B\}$

For a given (unpolarized) LJ sequent, different polarizations do not affect provability, but give different forms of proofs.

The LJF system (with only implications)

Formulas are built using atomic formulas and implications.

Formulas are polarized:

- Implications are negative.
- Atomic formulas can be either positive or negative.

A polarized theory is a set of formulas together with an atomic bias assignment $\delta : \mathrm{ATOM} \to \{+, -\}$.

Example: $\{(A \supset B) \supset C, A \supset A \supset B, C \supset B\}$

For a given (unpolarized) LJ sequent, different polarizations do not affect provability, but give different forms of proofs.

The LJF system (with only implications)

Formulas are built using atomic formulas and implications.

Formulas are polarized:

- Implications are negative.
- Atomic formulas can be either positive or negative.

A polarized theory is a set of formulas together with an atomic bias assignment $\delta : \mathrm{ATOM} \to \{+, -\}$.

Example: $\{(A \supset B) \supset C, A \supset A \supset B, C \supset B\}$

For a given (unpolarized) LJ sequent, different polarizations do not affect provability, but give different forms of proofs.

The LJF system (with only implications)

Formulas are built using atomic formulas and implications.

Formulas are polarized:

- Implications are negative.
- Atomic formulas can be either positive or negative.

A polarized theory is a set of formulas together with an atomic bias assignment $\delta : \text{ATOM} \to \{+, -\}$.

Example: $\{(A \supset B) \supset C, A \supset A \supset B, C \supset B\}$

For a given (unpolarized) LJ sequent, different polarizations do not affect provability, but give different forms of proofs.

The LJF system (with only implications)

Formulas are built using atomic formulas and implications.

Formulas are polarized:

- Implications are negative.
- Atomic formulas can be either positive or negative.

A polarized theory is a set of formulas together with an atomic bias assignment $\delta : \mathrm{ATOM} \to \{+, -\}$.

Example: $\{(A \supset B) \supset C, A \supset A \supset B, C \supset B\}$

For a given (unpolarized) LJ sequent, different polarizations do not affect provability, but give different forms of proofs.

The LJF system (with only implications)

Formulas are built using atomic formulas and implications.

Formulas are polarized:

- Implications are negative.
- Atomic formulas can be either positive or negative.

A polarized theory is a set of formulas together with an atomic bias assignment $\delta : \mathsf{ATOM} \to \{+, -\}$.

Example: $\{(A \supset B) \supset C, A \supset A \supset B, C \supset B\}$

For a given (unpolarized) LJ sequent, different polarizations do not affect provability, but give different forms of proofs.

The LJF system

Structural rules:

$$\frac{N, \Gamma \Downarrow N \vdash A}{N, \Gamma \vdash A}\ D_l \qquad \frac{\Gamma \vdash P \Downarrow}{\Gamma \vdash P}\ D_R \qquad \frac{\Gamma \Uparrow P \vdash A}{\Gamma \Downarrow P \vdash A}\ R_l \qquad \frac{\Gamma \vdash N \Uparrow}{\Gamma \vdash N \Downarrow}\ R_r$$

$$\frac{\Gamma, B \Uparrow \Delta \vdash \Theta \Uparrow \Theta'}{\Gamma \Uparrow \Delta, B \vdash \Theta \Uparrow \Theta'}\ S_l \qquad \frac{\Gamma \Uparrow \Delta \vdash A}{\Gamma \Uparrow \Delta \vdash A \Uparrow}\ S_r$$

Initial rules:

$$\frac{A \text{ negative}}{\Gamma \Downarrow A \vdash A}\ I_l \qquad \frac{A \text{ positive}}{A, \Gamma \vdash A \Downarrow}\ I_R$$

Logical rules:

$$\frac{\Gamma \vdash B_1 \Downarrow \qquad \Gamma \Downarrow B_2 \vdash A}{\Gamma \Downarrow B_1 \supset B_2 \vdash A} \supset L \qquad \frac{\Gamma \Uparrow \Delta, B_1 \vdash B_2 \Uparrow}{\Gamma \Uparrow \Delta \vdash B_1 \supset B_2 \Uparrow} \supset R$$

Two-phase structure of focused proofs

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{I_R}
\UnaryInfC{$\Gamma, D \vdash B \Downarrow$}
\AxiomC{}
\RightLabel{I_l}
\UnaryInfC{$\Gamma, D \Downarrow C \vdash C$}
\RightLabel{$\supset L$}
\BinaryInfC{$\Gamma, D \Downarrow B \supset C \vdash C$}
\RightLabel{D_l}
\UnaryInfC{$\Gamma, D \vdash C$}
\RightLabel{S_r}
\UnaryInfC{$\Gamma, D \vdash C \Uparrow$}
\RightLabel{S_l}
\UnaryInfC{$\Gamma \Uparrow D \vdash C \Uparrow$}
\RightLabel{$\supset R$}
\UnaryInfC{$\Gamma \vdash D \supset C \Uparrow$}
\RightLabel{R_r}
\UnaryInfC{$\Gamma \vdash D \supset C \Downarrow$}
\AxiomC{}
\RightLabel{I_l}
\UnaryInfC{$\Gamma \Downarrow E \vdash E$}
\RightLabel{$\supset L$}
\BinaryInfC{$\Gamma \Downarrow (D \supset C) \supset E \vdash E$}
\RightLabel{D_l}
\UnaryInfC{$B, B \supset C, (D \supset C) \supset E \vdash E$}
\end{prooftree}
$$

Synthetic inference rules [Mar+22]

Focused proofs have a two-phase structure: $\Downarrow$-phase and $\Uparrow$-phase.
$\Downarrow$-phase + $\Uparrow$-phase = large-scale inference rule = synthetic inference rule

Definition

A *synthetic inference rule* for B is an inference rule of the form

$$\frac{B, \Gamma_1 \vdash A_1 \qquad \dots \qquad B, \Gamma_n \vdash A_n}{B, \Gamma \vdash A}\ B$$

justified by an LJF derivation of the form

$$\cfrac{\begin{array}{c} B, \Gamma_1 \vdash A_1 \qquad \dots \qquad B, \Gamma_n \vdash A_n \\ \vdots\ \Uparrow\text{-phase} \\ \vdots\ \Downarrow\text{-phase} \\ B, \Gamma \Downarrow B \vdash A \end{array}}{B, \Gamma \vdash A}\ D_l$$

Synthetic inference rules [Mar+22]

Focused proofs have a two-phase structure: $\Downarrow$-phase and $\Uparrow$-phase.

$\Downarrow$-phase + $\Uparrow$-phase = large-scale inference rule = synthetic inference rule

Definition

A *synthetic inference rule* for B is an inference rule of the form

$$\frac{B, \Gamma_1 \vdash A_1 \qquad \ldots \qquad B, \Gamma_n \vdash A_n}{B, \Gamma \vdash A}\; B$$

justified by an LJF derivation of the form

$$\begin{array}{c}
B, \Gamma_1 \vdash A_1 \qquad \ldots \qquad B, \Gamma_n \vdash A_n \\
\vdots\; \Uparrow\text{-phase} \\
\vdots\; \Downarrow\text{-phase} \\
\dfrac{B, \Gamma \Downarrow B \vdash A}{B, \Gamma \vdash A}\; D_l
\end{array}$$

Synthetic inference rules [Mar+22]

Focused proofs have a two-phase structure: $\Downarrow$-phase and $\Uparrow$-phase.

$\Downarrow$-phase + $\Uparrow$-phase = large-scale inference rule = synthetic inference rule

Definition

A *synthetic inference rule* for B is an inference rule of the form

$$\frac{B, \Gamma_1 \vdash A_1 \qquad \dots \qquad B, \Gamma_n \vdash A_n}{B, \Gamma \vdash A}\ B$$

justified by an LJF derivation of the form

$$\dfrac{\begin{array}{c} B, \Gamma_1 \vdash A_1 \qquad \dots \qquad B, \Gamma_n \vdash A_n \\ \vdots\ \Uparrow\text{-phase} \\ \vdots\ \Downarrow\text{-phase} \\ B, \Gamma \Downarrow B \vdash A \end{array}}{B, \Gamma \vdash A}\ D_l$$

Synthetic inference rules [Mar+22]

Focused proofs have a two-phase structure: $\Downarrow$-phase and $\Uparrow$-phase.
$\Downarrow$-phase + $\Uparrow$-phase = large-scale inference rule = synthetic inference rule

Definition

A *synthetic inference rule* for B is an inference rule of the form

$$\frac{B, \Gamma_1 \vdash A_1 \quad \dots \quad B, \Gamma_n \vdash A_n}{B, \Gamma \vdash A}\ B$$

justified by an LJF derivation of the form

$$\begin{array}{c} B, \Gamma_1 \vdash A_1 \quad \dots \quad B, \Gamma_n \vdash A_n \\ \vdots\ \Uparrow\text{-phase} \\ \vdots\ \Downarrow\text{-phase} \\ \dfrac{B, \Gamma \Downarrow B \vdash A}{B, \Gamma \vdash A}\ D_l \end{array}$$

Synthetic inference rules [Mar+22]

Focused proofs have a two-phase structure: $\Downarrow$-phase and $\Uparrow$-phase.
$\Downarrow$-phase + $\Uparrow$-phase = large-scale inference rule = synthetic inference rule

Definition

A *synthetic inference rule* for B is an inference rule of the form

$$\frac{B, \Gamma_1 \vdash A_1 \qquad \dots \qquad B, \Gamma_n \vdash A_n}{B, \Gamma \vdash A}\ B$$

justified by an LJF derivation of the form

$$\dfrac{\dfrac{\dfrac{B, \Gamma_1 \vdash A_1 \qquad \dots \qquad B, \Gamma_n \vdash A_n}{\vdots\ \Uparrow\text{-phase}}}{\vdots\ \Downarrow\text{-phase}}}{\dfrac{B, \Gamma \Downarrow B \vdash A}{B, \Gamma \vdash A}\ D_l}$$

Synthetic inference rules

First remark:
for all i, $\Gamma \subseteq \Gamma_i$ and $\Gamma_i \setminus \Gamma$ depends only on B.

$$\begin{array}{c} B, \Gamma_1 \vdash A_1 \quad \ldots \quad B, \Gamma_n \vdash A_n \\ \vdots\ \Uparrow\text{-phase} \\ \vdots\ \Downarrow\text{-phase} \\ \dfrac{B, \Gamma \Downarrow B \vdash A}{B, \Gamma \vdash A}\ D_l \end{array}$$

A natural question arises: what kind of B can make $\Gamma_i \setminus \Gamma$ particularly simple?

Order of a formula:

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

If $ord(B) = k$, then $ord(C) \leq k - 2$ for all $C \in \Gamma_i \setminus \Gamma$.

In particular, $\Gamma_i \setminus \Gamma$ contains only atomic formulas if $ord(B) \leq 2$.

Synthetic inference rules

First remark:
for all i, $\Gamma \subseteq \Gamma_i$ and $\Gamma_i \setminus \Gamma$ depends only on B.

$$
\begin{array}{c}
B, \Gamma_1 \vdash A_1 \qquad \ldots \qquad B, \Gamma_n \vdash A_n \\
\vdots \ \Uparrow\text{-phase} \\
\vdots \ \Downarrow\text{-phase} \\
\dfrac{B, \Gamma \Downarrow B \vdash A}{B, \Gamma \vdash A} \ D_l
\end{array}
$$

A natural question arises: what kind of B can make $\Gamma_i \setminus \Gamma$ particularly simple?

Order of a formula:

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

If $ord(B) = k$, then $ord(C) \leq k - 2$ for all $C \in \Gamma_i \setminus \Gamma$.

In particular, $\Gamma_i \setminus \Gamma$ contains only atomic formulas if $ord(B) \leq 2$.

Synthetic inference rules

First remark:
for all i, $\Gamma \subseteq \Gamma_i$ and $\Gamma_i \setminus \Gamma$ depends only on B.

A natural question arises: what kind of B can make $\Gamma_i \setminus \Gamma$ particularly simple?

$$B, \Gamma_1 \vdash A_1 \quad \dots \quad B, \Gamma_n \vdash A_n$$

$\vdots$ $\Uparrow$-phase

$\vdots$ $\Downarrow$-phase

$$\frac{B, \Gamma \Downarrow B \vdash A}{B, \Gamma \vdash A}\ D_l$$

Order of a formula:

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

If $ord(B) = k$, then $ord(C) \leq k - 2$ for all $C \in \Gamma_i \setminus \Gamma$.

In particular, $\Gamma_i \setminus \Gamma$ contains only atomic formulas if $ord(B) \leq 2$.

Synthetic inference rules

First remark:
for all i, $\Gamma \subseteq \Gamma_i$ and $\Gamma_i \setminus \Gamma$ depends only on B.

A natural question arises: what kind of B can make $\Gamma_i \setminus \Gamma$ particularly simple?

$$\dfrac{\dfrac{B, \Gamma_1 \vdash A_1 \quad \ldots \quad B, \Gamma_n \vdash A_n}{\vdots \ \Uparrow\text{-phase}}}{\dfrac{\vdots \ \Downarrow\text{-phase}}{\dfrac{B, \Gamma \Downarrow B \vdash A}{B, \Gamma \vdash A}\ D_l}}$$

Order of a formula:

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

If $ord(B) = k$, then $ord(C) \leq k - 2$ for all $C \in \Gamma_i \setminus \Gamma$.

In particular, $\Gamma_i \setminus \Gamma$ contains only atomic formulas if $ord(B) \leq 2$.

Synthetic inference rules

First remark:
for all i, $\Gamma \subseteq \Gamma_i$ and $\Gamma_i \setminus \Gamma$ depends only on B.

$$
\begin{array}{c}
B, \Gamma_1 \vdash A_1 \qquad \dots \qquad B, \Gamma_n \vdash A_n \\
\vdots \ \Uparrow\text{-phase} \\
\vdots \ \Downarrow\text{-phase} \\
\dfrac{B, \Gamma \Downarrow B \vdash A}{B, \Gamma \vdash A} \ D_l
\end{array}
$$

A natural question arises: what kind of B can make $\Gamma_i \setminus \Gamma$ particularly simple?

Order of a formula:

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

If $ord(B) = k$, then $ord(C) \leq k - 2$ for all $C \in \Gamma_i \setminus \Gamma$.

In particular, $\Gamma_i \setminus \Gamma$ contains only atomic formulas if $ord(B) \leq 2$.

Axioms as rules

Definition (Extension $LJ\langle T\rangle$ of LJ)

Let T be a finite polarized theory of order at most 2. For every synthetic inference rule

$$\frac{B, \Gamma_1 \vdash A_1 \quad \dots \quad B, \Gamma_n \vdash A_n}{B, \Gamma \vdash A}\ B$$

with $B \in T$, $LJ\langle T\rangle$ includes the inference rule

$$\frac{\Gamma_1 \vdash A_1 \quad \dots \quad \Gamma_n \vdash A_n}{\Gamma \vdash A}\ B$$

$\hookrightarrow$ Make axioms implicit by adding rules.

Theorem

$T, \Gamma \vdash B$ provable in LJ $\Leftrightarrow$ $\Gamma \vdash B$ provable in $LJ\langle T\rangle$.

Axioms as rules

Definition (Extension $LJ\langle T\rangle$ of LJ)

Let T be a finite polarized theory of order at most 2. For every synthetic inference rule

$$\frac{B, \Gamma_1 \vdash A_1 \quad \dots \quad B, \Gamma_n \vdash A_n}{B, \Gamma \vdash A}\ B$$

with $B \in T$, $LJ\langle T\rangle$ includes the inference rule

$$\frac{\Gamma_1 \vdash A_1 \quad \dots \quad \Gamma_n \vdash A_n}{\Gamma \vdash A}\ B$$

$\hookrightarrow$ Make axioms implicit by adding rules.

Theorem

$T, \Gamma \vdash B$ provable in $LJ \Leftrightarrow \Gamma \vdash B$ provable in $LJ\langle T\rangle$.

Axioms as rules

Definition (Extension $LJ\langle T\rangle$ of LJ)

Let T be a finite polarized theory of order at most 2. For every synthetic inference rule

$$\frac{B, \Gamma_1 \vdash A_1 \quad \dots \quad B, \Gamma_n \vdash A_n}{B, \Gamma \vdash A}\ B$$

with $B \in T$, $LJ\langle T\rangle$ includes the inference rule

$$\frac{\Gamma_1 \vdash A_1 \quad \dots \quad \Gamma_n \vdash A_n}{\Gamma \vdash A}\ B$$

$\hookrightarrow$ Make axioms implicit by adding rules.

Theorem

$T, \Gamma \vdash B$ provable in LJ $\Leftrightarrow$ $\Gamma \vdash B$ provable in $LJ\langle T\rangle$.

Axioms as rules

Definition (Extension $LJ\langle T\rangle$ of LJ)

Let T be a finite polarized theory of order at most 2. For every synthetic inference rule

$$\frac{B, \Gamma_1 \vdash A_1 \quad \ldots \quad B, \Gamma_n \vdash A_n}{B, \Gamma \vdash A}\ B$$

with $B \in T$, $LJ\langle T\rangle$ includes the inference rule

$$\frac{\Gamma_1 \vdash A_1 \quad \ldots \quad \Gamma_n \vdash A_n}{\Gamma \vdash A}\ B$$

$\hookrightarrow$ Make axioms implicit by adding rules.

Theorem

$T, \Gamma \vdash B$ provable in $LJ \Leftrightarrow \Gamma \vdash B$ provable in $LJ\langle T\rangle$.

An example

Let T be the collection of formulas
$B_1 = A_0 \supset A_1, \ldots, B_n = A_0 \supset \cdots \supset A_n, \ldots$ where A_i are all atomic.

If A_i are all given the negative polarity, then $LJ\langle T \rangle$ includes

$$\frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n}\ B_n$$

If A_i are all given the positive polarity, then $LJ\langle T \rangle$ includes

$$\frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A}\ B_n$$

An example

Let T be the collection of formulas
$B_1 = A_0 \supset A_1, \ldots, B_n = A_0 \supset \cdots \supset A_n, \ldots$ where A_i are all atomic.

If A_i are all given the negative polarity, then $LJ\langle T \rangle$ includes

$$\frac{\Gamma \vdash A_0 \quad \cdots \quad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n}\ B_n$$

If A_i are all given the positive polarity, then $LJ\langle T \rangle$ includes

$$\frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A}\ B_n$$

An example

Let T be the collection of formulas
$B_1 = A_0 \supset A_1, \ldots, B_n = A_0 \supset \cdots \supset A_n, \ldots$ where A_i are all atomic.

If A_i are all given the negative polarity, then $LJ\langle T \rangle$ includes

$$\frac{\Gamma \vdash A_0 \quad \cdots \quad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n}\ B_n$$

If A_i are all given the positive polarity, then $LJ\langle T \rangle$ includes

$$\frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A}\ B_n$$

An example

Let T be the collection of formulas
$B_1 = A_0 \supset A_1, \ldots, B_n = A_0 \supset \cdots \supset A_n, \ldots$ where A_i are all atomic.

If A_i are all given the negative polarity, then $LJ\langle T\rangle$ includes

$$\frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n}\ B_n$$

If A_i are all given the positive polarity, then $LJ\langle T\rangle$ includes

$$\frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A}\ B_n$$

What do proofs look like?

What are the proofs of $A_0 \vdash A_n$?

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash A_0}{\Gamma \vdash A_1} \quad \frac{\Gamma \vdash A_0 \qquad \Gamma \vdash A_1}{\Gamma \vdash A_2} \quad \cdots \quad \frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n} \quad \cdots$$

There is a unique proof of exponential size.

When A_i are all given the positive polarity, we have:

$$\frac{\Gamma, A_0, A_1 \vdash A}{\Gamma, A_0 \vdash A} \quad \frac{\Gamma, A_0, A_1, A_2 \vdash A}{\Gamma, A_0, A_1 \vdash A} \quad \cdots \quad \frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A} \quad \cdots$$

There is a shortest proof of linear size.

What do proofs look like?

What are the proofs of $A_0 \vdash A_n$?

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash A_0}{\Gamma \vdash A_1} \quad \frac{\Gamma \vdash A_0 \qquad \Gamma \vdash A_1}{\Gamma \vdash A_2} \quad \cdots \quad \frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n} \quad \cdots$$

There is a unique proof of exponential size.

When A_i are all given the positive polarity, we have:

$$\frac{\Gamma, A_0, A_1 \vdash A}{\Gamma, A_0 \vdash A} \quad \frac{\Gamma, A_0, A_1, A_2 \vdash A}{\Gamma, A_0, A_1 \vdash A} \quad \cdots \quad \frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A} \quad \cdots$$

There is a shortest proof of linear size.

What do proofs look like?

What are the proofs of $A_0 \vdash A_n$?

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash A_0}{\Gamma \vdash A_1} \quad \frac{\Gamma \vdash A_0 \qquad \Gamma \vdash A_1}{\Gamma \vdash A_2} \quad \cdots \quad \frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n} \quad \cdots$$

There is a unique proof of exponential size.

When A_i are all given the positive polarity, we have:

$$\frac{\Gamma, A_0, A_1 \vdash A}{\Gamma, A_0 \vdash A} \quad \frac{\Gamma, A_0, A_1, A_2 \vdash A}{\Gamma, A_0, A_1 \vdash A} \quad \cdots \quad \frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A} \quad \cdots$$

There is a shortest proof of linear size.

What do proofs look like?

What are the proofs of $A_0 \vdash A_n$?

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash A_0}{\Gamma \vdash A_1} \quad \frac{\Gamma \vdash A_0 \qquad \Gamma \vdash A_1}{\Gamma \vdash A_2} \quad \cdots \quad \frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n} \quad \cdots$$

There is a unique proof of exponential size.

When A_i are all given the positive polarity, we have:

$$\frac{\Gamma, A_0, A_1 \vdash A}{\Gamma, A_0 \vdash A} \quad \frac{\Gamma, A_0, A_1, A_2 \vdash A}{\Gamma, A_0, A_1 \vdash A} \quad \cdots \quad \frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A} \quad \cdots$$

There is a shortest proof of linear size.

What do proofs look like?

What are the proofs of $A_0 \vdash A_n$?

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash A_0}{\Gamma \vdash A_1} \quad \frac{\Gamma \vdash A_0 \qquad \Gamma \vdash A_1}{\Gamma \vdash A_2} \quad \cdots \quad \frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n} \quad \cdots$$

There is a unique proof of exponential size.

When A_i are all given the positive polarity, we have:

$$\frac{\Gamma, A_0, A_1 \vdash A}{\Gamma, A_0 \vdash A} \quad \frac{\Gamma, A_0, A_1, A_2 \vdash A}{\Gamma, A_0, A_1 \vdash A} \quad \cdots \quad \frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A} \quad \cdots$$

There is a shortest proof of linear size.

What do proofs look like?

What are the proofs of $A_0 \vdash A_n$?

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash A_0}{\Gamma \vdash A_1} \quad \frac{\Gamma \vdash A_0 \qquad \Gamma \vdash A_1}{\Gamma \vdash A_2} \quad \cdots \quad \frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n} \quad \cdots$$

There is a unique proof of exponential size.

When A_i are all given the positive polarity, we have:

$$\frac{\Gamma, A_0, A_1 \vdash A}{\Gamma, A_0 \vdash A} \quad \frac{\Gamma, A_0, A_1, A_2 \vdash A}{\Gamma, A_0, A_1 \vdash A} \quad \cdots \quad \frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A} \quad \cdots$$

There is a shortest proof of linear size.

Annotate rules and proofs

Now let us annotate the inference rules in the previous example.

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash A_0}{\Gamma \vdash A_1} \quad \frac{\Gamma \vdash A_0 \qquad \Gamma \vdash A_1}{\Gamma \vdash A_2} \quad \cdots$$

$$\frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n}$$

The unique proof of $A_0 \vdash A_4$ is annotated by the term:

$$\begin{array}{l}(B_4\ (B_3\ (B_2\ (B_1\ x_0)\ (B_1\ x_0))\\ \qquad\qquad (B_2\ (B_1\ x_0)\ (B_1\ x_0)))\\ \qquad (B_3\ (B_2\ (B_1\ x_0)\ (B_1\ x_0))\\ \qquad\qquad (B_2\ (B_1\ x_0)\ (B_1\ x_0))))\end{array}$$

Annotate rules and proofs

Now let us annotate the inference rules in the previous example.

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash A_0}{\Gamma \vdash A_1} \quad \frac{\Gamma \vdash A_0 \qquad \Gamma \vdash A_1}{\Gamma \vdash A_2} \quad \cdots$$

$$\frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n}$$

The unique proof of $A_0 \vdash A_4$ is annotated by the term:

$$\begin{array}{l}(B_4\ (B_3\ (B_2\ (B_1\ x_0)\ (B_1\ x_0)) \\ \qquad\qquad (B_2\ (B_1\ x_0)\ (B_1\ x_0))) \\ \qquad (B_3\ (B_2\ (B_1\ x_0)\ (B_1\ x_0)) \\ \qquad\qquad (B_2\ (B_1\ x_0)\ (B_1\ x_0))))\end{array}$$

Annotate rules and proofs

Now let us annotate the inference rules in the previous example.

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash A_0}{\Gamma \vdash A_1} \quad \frac{\Gamma \vdash A_0 \qquad \Gamma \vdash A_1}{\Gamma \vdash A_2} \quad \cdots$$

$$\frac{\Gamma \vdash A_0 \qquad \cdots \qquad \Gamma \vdash A_{n-1}}{\Gamma \vdash A_n}$$

The unique proof of $A_0 \vdash A_4$ is annotated by the term:

$$\begin{aligned}(B_4\ (B_3\ (B_2\ (B_1\ x_0)\ (B_1\ x_0))\\ (B_2\ (B_1\ x_0)\ (B_1\ x_0)))\\ (B_3\ (B_2\ (B_1\ x_0)\ (B_1\ x_0))\\ (B_2\ (B_1\ x_0)\ (B_1\ x_0))))\end{aligned}$$

Annotate rules and proofs

Now let us annotate the inference rules in the previous example.

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash t_0 : A_0}{\Gamma \vdash B_1\, t_0 : A_1} \quad \frac{\Gamma \vdash t_0 : A_0 \qquad \Gamma \vdash t_1 : A_1}{\Gamma \vdash B_2\, t_0\, t_1 : A_2} \quad \cdots$$

$$\frac{\Gamma \vdash t_0 : A_0 \qquad \cdots \qquad \Gamma \vdash t_{n-1} : A_{n-1}}{\Gamma \vdash B_n\, t_0 \cdots t_{n-1} : A_n}$$

The unique proof of $A_0 \vdash A_4$ is annotated by the term:

$$\begin{aligned}(B_4\ (B_3\ (B_2\ (B_1\ x_0)\ (B_1\ x_0))\\ (B_2\ (B_1\ x_0)\ (B_1\ x_0)))\\ (B_3\ (B_2\ (B_1\ x_0)\ (B_1\ x_0))\\ (B_2\ (B_1\ x_0)\ (B_1\ x_0))))\end{aligned}$$

Annotate rules and proofs

Now let us annotate the inference rules in the previous example.

When A_i are all given the negative polarity, we have:

$$\frac{\Gamma \vdash t_0 : A_0}{\Gamma \vdash B_1\, t_0 : A_1} \quad \frac{\Gamma \vdash t_0 : A_0 \qquad \Gamma \vdash t_1 : A_1}{\Gamma \vdash B_2\, t_0\, t_1 : A_2} \quad \cdots$$

$$\frac{\Gamma \vdash t_0 : A_0 \qquad \cdots \qquad \Gamma \vdash t_{n-1} : A_{n-1}}{\Gamma \vdash B_n\, t_0 \cdots t_{n-1} : A_n}$$

The unique proof of $A_0 \vdash A_4$ is annotated by the term:

$$\begin{aligned}(B_4\ (B_3\ &(B_2\ (B_1\ x_0)\ (B_1\ x_0))\\ &(B_2\ (B_1\ x_0)\ (B_1\ x_0)))\\ (B_3\ &(B_2\ (B_1\ x_0)\ (B_1\ x_0))\\ &(B_2\ (B_1\ x_0)\ (B_1\ x_0))))\end{aligned}$$

Annotate rules and proofs

Now let us annotate the inference rules in the previous example.

When A_i are all given the positive polarity, we have:

$$\frac{\Gamma, A_0, A_1 \vdash A}{\Gamma, A_0 \vdash A} \quad \frac{\Gamma, A_0, A_1, A_2 \vdash A}{\Gamma, A_0, A_1 \vdash A} \quad \ldots$$

$$\frac{\Gamma, A_0, \ldots, A_{n-1}, A_n \vdash A}{\Gamma, A_0, \ldots, A_{n-1} \vdash A}$$

The shortest proof of $A_0 \vdash A_4$ is annotated by the term:

$$\begin{array}{ll}
(B_1\ x_0 & (\lambda x_1. \\
(B_2\ x_0\ x_1 & (\lambda x_2. \\
(B_3\ x_0\ x_1\ x_2 & (\lambda x_3. \\
(B_4\ x_0\ x_1\ x_2\ x_3 & (\lambda x_4.\ x_4))))))))
\end{array}$$

Annotate rules and proofs

Now let us annotate the inference rules in the previous example.

When A_i are all given the positive polarity, we have:

$$\frac{\Gamma, x_0 : A_0, x_1 : A_1 \vdash t : A}{\Gamma, x_0 : A_0 \vdash B_1 x_0 (\lambda x_1 . t) : A} \quad \frac{\Gamma, x_0 : A_0, x_1 : A_1, x_2 : A_2 \vdash t : A}{\Gamma, x_0 : A_0, x_1 : A_1 \vdash B_2 x_0 x_1 (\lambda x_2 . t) : A} \quad \cdots$$

$$\frac{\Gamma, x_0 : A_0, \ldots, x_{n-1} : A_{n-1}, x_n : A_n \vdash t : A}{\Gamma, x_0 : A_0, \ldots, x_{n-1} : A_{n-1} \vdash B_n x_0 \cdots x_{n-1} (\lambda x_n . t) : A}$$

The shortest proof of $A_0 \vdash A_4$ is annotated by the term:

$$\begin{array}{ll}
(B_1\ x_0 & (\lambda x_1 . \\
(B_2\ x_0\ x_1 & (\lambda x_2 . \\
(B_3\ x_0\ x_1\ x_2 & (\lambda x_3 . \\
(B_4\ x_0\ x_1\ x_2\ x_3 & (\lambda x_4 .\ x_4)))))))
\end{array}$$

Annotate rules and proofs

Now let us annotate the inference rules in the previous example.

When A_i are all given the positive polarity, we have:

$$\frac{\Gamma, x_0 : A_0, x_1 : A_1 \vdash t : A}{\Gamma, x_0 : A_0 \vdash B_1 x_0 (\lambda x_1 . t) : A} \quad \frac{\Gamma, x_0 : A_0, x_1 : A_1, x_2 : A_2 \vdash t : A}{\Gamma, x_0 : A_0, x_1 : A_1 \vdash B_2 x_0 x_1 (\lambda x_2 . t) : A} \quad \cdots$$

$$\frac{\Gamma, x_0 : A_0, \ldots, x_{n-1} : A_{n-1}, x_n : A_n \vdash t : A}{\Gamma, x_0 : A_0, \ldots, x_{n-1} : A_{n-1} \vdash B_n x_0 \cdots x_{n-1} (\lambda x_n . t) : A}$$

The shortest proof of $A_0 \vdash A_4$ is annotated by the term:

$$\begin{array}{ll} (B_1\ x_0 & (\lambda x_1 . \\ (B_2\ x_0\ x_1 & (\lambda x_2 . \\ (B_3\ x_0\ x_1\ x_2 & (\lambda x_3 . \\ (B_4\ x_0\ x_1\ x_2\ x_3 & (\lambda x_4 .\ x_4)))))))) \end{array}$$

Two encodings of untyped λ-terms [MW23]

Let T be the set $\{D \supset D \supset D, (D \supset D) \supset D\}$ where D is atomic.
We consider $LJ\langle T\rangle$ and only sequents of the form $D, \ldots, D \vdash D$.

Logically, it does not seem interesting.
Once again, we do not care about provability but the structure of proofs.

If D is negative, then we have:

$$D \in \Gamma\ \frac{}{\Gamma \vdash D} \qquad \frac{\Gamma \vdash D \quad \Gamma \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D}{\Gamma \vdash D}$$

If D is positive, then we have:

$$D \in \Gamma\ \frac{}{\Gamma \vdash D} \qquad \{D, D\} \subseteq \Gamma\ \frac{\Gamma, D \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D \quad \Gamma, D \vdash D}{\Gamma \vdash D}$$

Two encodings of untyped λ-terms [MW23]

Let T be the set $\{D \supset D \supset D, (D \supset D) \supset D\}$ where D is atomic.
We consider $LJ\langle T\rangle$ and only sequents of the form $D, \ldots, D \vdash D$.

Logically, it does not seem interesting.
Once again, we do not care about provability but the structure of proofs.

If D is negative, then we have:

$$D \in \Gamma \; \frac{}{\Gamma \vdash D} \qquad \frac{\Gamma \vdash D \quad \Gamma \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D}{\Gamma \vdash D}$$

If D is positive, then we have:

$$D \in \Gamma \; \frac{}{\Gamma \vdash D} \qquad \{D, D\} \subseteq \Gamma \; \frac{\Gamma, D \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D \quad \Gamma, D \vdash D}{\Gamma \vdash D}$$

Two encodings of untyped λ-terms [MW23]

Let T be the set $\{D \supset D \supset D, (D \supset D) \supset D\}$ where D is atomic.
We consider $LJ\langle T\rangle$ and only sequents of the form $D, \ldots, D \vdash D$.

Logically, it does not seem interesting.
Once again, we do not care about provability but the structure of proofs.

If D is negative, then we have:

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

If D is positive, then we have:

$$D \in \Gamma\ \frac{}{\Gamma \vdash D} \qquad \{D, D\} \subseteq \Gamma\ \frac{\Gamma, D \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D \qquad \Gamma, D \vdash D}{\Gamma \vdash D}$$

Two encodings of untyped λ-terms [MW23]

Let T be the set $\{D \supset D \supset D, (D \supset D) \supset D\}$ where D is atomic.
We consider $LJ\langle T\rangle$ and only sequents of the form $D, \ldots, D \vdash D$.

Logically, it does not seem interesting.
Once again, we do not care about provability but the structure of proofs.

If D is negative, then we have:

$$D \in \Gamma \; \frac{}{\Gamma \vdash D} \qquad \frac{\Gamma \vdash D \quad \Gamma \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D}{\Gamma \vdash D}$$

If D is positive, then we have:

$$D \in \Gamma \; \frac{}{\Gamma \vdash D} \qquad \{D, D\} \subseteq \Gamma \; \frac{\Gamma, D \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D \quad \Gamma, D \vdash D}{\Gamma \vdash D}$$

Two encodings of untyped λ-terms [MW23]

Let T be the set $\{D \supset D \supset D, (D \supset D) \supset D\}$ where D is atomic.
We consider $LJ\langle T\rangle$ and only sequents of the form $D, \ldots, D \vdash D$.

Logically, it does not seem interesting.
Once again, we do not care about provability but the structure of proofs.

If D is negative, then we have:

$$D \in \Gamma\ \frac{}{\Gamma \vdash D} \qquad \frac{\Gamma \vdash D \quad \Gamma \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D}{\Gamma \vdash D}$$

If D is positive, then we have:

$$D \in \Gamma\ \frac{}{\Gamma \vdash D} \qquad \{D, D\} \subseteq \Gamma\ \frac{\Gamma, D \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D \quad \Gamma, D \vdash D}{\Gamma \vdash D}$$

Two encodings of untyped λ-terms [MW23]

Let T be the set $\{D \supset D \supset D, (D \supset D) \supset D\}$ where D is atomic.
We consider $LJ\langle T\rangle$ and only sequents of the form $D, \ldots, D \vdash D$.

Logically, it does not seem interesting.
Once again, we do not care about provability but the structure of proofs.

If D is negative, then we have:

$$D \in \Gamma \ \frac{}{\Gamma \vdash D} \qquad \frac{\Gamma \vdash D \quad \Gamma \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D}{\Gamma \vdash D}$$

If D is positive, then we have:

$$D \in \Gamma \ \frac{}{\Gamma \vdash D} \qquad \{D, D\} \subseteq \Gamma \ \frac{\Gamma, D \vdash D}{\Gamma \vdash D} \qquad \frac{\Gamma, D \vdash D \quad \Gamma, D \vdash D}{\Gamma \vdash D}$$

Two encodings of untyped λ-terms

D is negative

$$\frac{}{\Gamma \vdash D}\; D \in \Gamma$$

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

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

negative bias syntax

D is positive

$$\frac{}{\Gamma \vdash D}\; D \in \Gamma$$

$$\frac{\Gamma, D \vdash D}{\Gamma \vdash D}\; \{D, D\} \subseteq \Gamma$$

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

positive bias syntax

Two encodings of untyped λ-terms

D is negative

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

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

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

negative bias syntax

D is positive

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

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

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

positive bias syntax

Two encodings of untyped λ-terms

D is negative

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

$$\frac{\Gamma \vdash t : D \qquad \Gamma \vdash u : D}{\Gamma \vdash tu : D}$$

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

negative bias syntax

D is positive

$$\frac{}{\Gamma \vdash D}\; D \in \Gamma$$

$$\frac{\Gamma, D \vdash D}{\Gamma \vdash D}\; \{D, D\} \subseteq \Gamma$$

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

positive bias syntax

Two encodings of untyped λ-terms

D is negative

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

$$\frac{\Gamma \vdash t : D \qquad \Gamma \vdash u : D}{\Gamma \vdash tu : D}$$

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

negative bias syntax

D is positive

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

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

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

positive bias syntax

Two encodings of untyped λ-terms

D is negative

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

$$\frac{\Gamma \vdash t : D \qquad \Gamma \vdash u : D}{\Gamma \vdash tu : D}$$

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

negative bias syntax

D is positive

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

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

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

positive bias syntax

Two encodings of untyped λ-terms

D is negative

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

$$\frac{\Gamma \vdash t : D \qquad \Gamma \vdash u : D}{\Gamma \vdash tu : D}$$

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

negative bias syntax

D is positive

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

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

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

positive bias syntax

Two encodings of untyped λ-terms

D is negative

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

$$\frac{\Gamma \vdash t : D \qquad \Gamma \vdash u : D}{\Gamma \vdash tu : D}$$

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

negative bias syntax

D is positive

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

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

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

positive bias syntax

Two encodings of untyped λ-terms

D is negative

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

$$\frac{\Gamma \vdash t : D \qquad \Gamma \vdash u : D}{\Gamma \vdash tu : D}$$

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

negative bias syntax

D is positive

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

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

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

positive bias syntax

Two encodings of untyped λ-terms

Negative bias syntax $t ::= x \mid tu \mid \lambda x.t$

$\hookrightarrow$ Usual syntax of untyped λ-terms, tree-structure, top-down

Positive bias syntax $t ::= x \mid t[x \leftarrow yz] \mid t[x \leftarrow \lambda y.u]$

$\hookrightarrow$ Allows sharing via explicit substitutions, DAG-structure, bottom-up

We only consider **cut-free** proofs. So what does **cut-elimination** tell us?

$\hookrightarrow$ In both cases, the cut-elimination of LJF provides us a natural notion of substitution.

- In the negative case, we get the usual meta-level substitution of untyped λ-calculus.
- In the positive case, we also get a straightforward notion of substitution.

Two encodings of untyped λ-terms

Negative bias syntax $t ::= x \mid tu \mid \lambda x.t$

$\hookrightarrow$ Usual syntax of untyped λ-terms, tree-structure, top-down

Positive bias syntax $t ::= x \mid t[x{\leftarrow}yz] \mid t[x{\leftarrow}\lambda y.u]$

$\hookrightarrow$ Allows sharing via explicit substitutions, DAG-structure, bottom-up

We only consider **cut-free** proofs. So what does **cut-elimination** tell us?

$\hookrightarrow$ In both cases, the cut-elimination of LJF provides us a natural notion of substitution.

- In the negative case, we get the usual meta-level substitution of untyped λ-calculus.
- In the positive case, we also get a straightforward notion of substitution.

Two encodings of untyped λ-terms

Negative bias syntax $t ::= x \mid tu \mid \lambda x.t$

$\hookrightarrow$ Usual syntax of untyped λ-terms, tree-structure, top-down

Positive bias syntax $t ::= x \mid t[x \leftarrow yz] \mid t[x \leftarrow \lambda y.u]$

$\hookrightarrow$ Allows sharing via explicit substitutions, DAG-structure, bottom-up

We only consider **cut-free** proofs. So what does **cut-elimination** tell us?

$\hookrightarrow$ In both cases, the cut-elimination of LJF provides us a natural notion of substitution.

- In the negative case, we get the usual meta-level substitution of untyped λ-calculus.
- In the positive case, we also get a straightforward notion of substitution.

Two encodings of untyped λ-terms

Negative bias syntax $t ::= x \mid tu \mid \lambda x.t$

$\hookrightarrow$ Usual syntax of untyped λ-terms, tree-structure, top-down

Positive bias syntax $t ::= x \mid t[x \leftarrow yz] \mid t[x \leftarrow \lambda y.u]$

$\hookrightarrow$ Allows sharing via explicit substitutions, DAG-structure, bottom-up

We only consider **cut-free** proofs. So what does **cut-elimination** tell us?

$\hookrightarrow$ In both cases, the cut-elimination of LJF provides us a natural notion of substitution.

- In the negative case, we get the usual meta-level substitution of untyped λ-calculus.
- In the positive case, we also get a straightforward notion of substitution.

Two encodings of untyped λ-terms

Negative bias syntax $t ::= x \mid tu \mid \lambda x.t$

$\hookrightarrow$ Usual syntax of untyped λ-terms, tree-structure, top-down

Positive bias syntax $t ::= x \mid t[x \leftarrow yz] \mid t[x \leftarrow \lambda y.u]$

$\hookrightarrow$ Allows sharing via explicit substitutions, DAG-structure, bottom-up

We only consider **cut-free** proofs. So what does **cut-elimination** tell us?

$\hookrightarrow$ In both cases, the cut-elimination of LJF provides us a natural notion of substitution.

- In the negative case, we get the usual meta-level substitution of untyped λ-calculus.
- In the positive case, we also get a straightforward notion of substitution.

Two encodings of untyped λ-terms

Negative bias syntax $t ::= x \mid tu \mid \lambda x.t$

$\hookrightarrow$ Usual syntax of untyped λ-terms, tree-structure, top-down

Positive bias syntax $t ::= x \mid t[x \leftarrow yz] \mid t[x \leftarrow \lambda y.u]$

$\hookrightarrow$ Allows sharing via explicit substitutions, DAG-structure, bottom-up

We only consider **cut-free** proofs. So what does **cut-elimination** tell us?

$\hookrightarrow$ In both cases, the cut-elimination of LJF provides us a natural notion of substitution.

- In the negative case, we get the usual meta-level substitution of untyped λ-calculus.
- In the positive case, we also get a straightforward notion of substitution.

Two encodings of untyped λ-terms

Negative bias syntax $t ::= x \mid tu \mid \lambda x.t$

$\hookrightarrow$ Usual syntax of untyped λ-terms, tree-structure, top-down

Positive bias syntax $t ::= x \mid t[x \leftarrow yz] \mid t[x \leftarrow \lambda y.u]$

$\hookrightarrow$ Allows sharing via explicit substitutions, DAG-structure, bottom-up

We only consider **cut-free** proofs. So what does **cut-elimination** tell us?

$\hookrightarrow$ In both cases, the cut-elimination of LJF provides us a natural notion of substitution.

- In the negative case, we get the usual meta-level substitution of untyped λ-calculus.
- In the positive case, we also get a straightforward notion of substitution.

Positive λ-calculus (λ_{pos})

Terms, contexts, and left contexts of λ_{pos} are defined as follows:

$$\begin{array}{rrl} \text{Terms} & t, u & ::= x \mid t[x{\leftarrow}yz] \mid t[x{\leftarrow}\lambda y.u] \\ \text{Contexts} & C & ::= \langle\cdot\rangle \mid C[x{\leftarrow}yz] \mid C[x{\leftarrow}\lambda y.t] \mid t[x{\leftarrow}\lambda y.C] \\ \text{Left Contexts} & L & ::= \langle\cdot\rangle \mid L[x{\leftarrow}yz] \mid L[x{\leftarrow}\lambda y.t] \end{array}$$

Every term can be written uniquely (up to α-equivalence) as $L\langle x\rangle$ for some L and x.

Structural equivalence:

$$t[x_1{\leftarrow}p_1][x_2{\leftarrow}p_2] \equiv t[x_2{\leftarrow}p_2][x_1{\leftarrow}p_1]$$

where $x_1 \notin fv(p_2)$ and $x_1 \notin fv(p_1)$

Positive λ-calculus (λ_{pos})

Terms, contexts, and left contexts of λ_{pos} are defined as follows:

$$\begin{array}{rrl} \text{Terms} & t, u & ::= x \mid t[x{\leftarrow}yz] \mid t[x{\leftarrow}\lambda y.u] \\ \text{Contexts} & C & ::= \langle\cdot\rangle \mid C[x{\leftarrow}yz] \mid C[x{\leftarrow}\lambda y.t] \mid t[x{\leftarrow}\lambda y.C] \\ \text{Left Contexts} & L & ::= \langle\cdot\rangle \mid L[x{\leftarrow}yz] \mid L[x{\leftarrow}\lambda y.t] \end{array}$$

Every term can be written uniquely (up to α-equivalence) as $L\langle x\rangle$ for some L and x.

Structural equivalence:

$$t[x_1{\leftarrow}p_1][x_2{\leftarrow}p_2] \equiv t[x_2{\leftarrow}p_2][x_1{\leftarrow}p_1]$$

where $x_1 \notin fv(p_2)$ and $x_1 \notin fv(p_1)$

Positive λ-calculus (λ_{pos})

Terms, contexts, and left contexts of λ_{pos} are defined as follows:

$$
\begin{array}{rrcl}
\text{Terms} & t, u & ::= & x \mid t[x{\leftarrow}yz] \mid t[x{\leftarrow}\lambda y.u] \\
\text{Contexts} & C & ::= & \langle\cdot\rangle \mid C[x{\leftarrow}yz] \mid C[x{\leftarrow}\lambda y.t] \mid t[x{\leftarrow}\lambda y.C] \\
\text{Left Contexts} & L & ::= & \langle\cdot\rangle \mid L[x{\leftarrow}yz] \mid L[x{\leftarrow}\lambda y.t]
\end{array}
$$

Every term can be written uniquely (up to α-equivalence) as $L\langle x\rangle$ for some L and x.

Structural equivalence:

$$t[x_1{\leftarrow}p_1][x_2{\leftarrow}p_2] \equiv t[x_2{\leftarrow}p_2][x_1{\leftarrow}p_1]$$

where $x_1 \notin fv(p_2)$ and $x_1 \notin fv(p_1)$

Positive λ-calculus (λ_{pos})

Terms, contexts, and left contexts of λ_{pos} are defined as follows:

$$
\begin{array}{rrcl}
\text{Terms} & t, u & ::= & x \mid t[x{\leftarrow}yz] \mid t[x{\leftarrow}\lambda y.u] \\
\text{Contexts} & C & ::= & \langle\cdot\rangle \mid C[x{\leftarrow}yz] \mid C[x{\leftarrow}\lambda y.t] \mid t[x{\leftarrow}\lambda y.C] \\
\text{Left Contexts} & L & ::= & \langle\cdot\rangle \mid L[x{\leftarrow}yz] \mid L[x{\leftarrow}\lambda y.t]
\end{array}
$$

Every term can be written uniquely (up to α-equivalence) as $L\langle x\rangle$ for some L and x.

Structural equivalence:

$$t[x_1{\leftarrow}p_1][x_2{\leftarrow}p_2] \equiv t[x_2{\leftarrow}p_2][x_1{\leftarrow}p_1]$$

where $x_1 \notin fv(p_2)$ and $x_1 \notin fv(p_1)$

λ_{pos}: Substitution

As mentioned before, cut-elimination of LJF provides a notion of subtitution, defined as follows:

$$t[x/y] = t\{x{\leftarrow}y\}$$
$$t[x/u[y{\leftarrow}zw]] = t[x/u][y{\leftarrow}zw]$$
$$t[x/r[y{\leftarrow}\lambda z.u]] = t[x/r][y{\leftarrow}\lambda z.u]$$

If we write u as $L\langle y\rangle$, then $t[x/u] = L\langle t\{x{\leftarrow}y\}\rangle$.

Ex: $t = x_0[x_0{\leftarrow}\lambda y.x][x_1{\leftarrow}fx]$ and $u = x_2[x_2{\leftarrow}\lambda z.x_3[x_3{\leftarrow}gz]]$

Then $t[x/u] = x_0[x_0{\leftarrow}\lambda y.x_2][x_1{\leftarrow}f\,x_2][x_2{\leftarrow}\lambda z.x_3[x_3{\leftarrow}gz]]$

λ_{pos}: Substitution

As mentioned before, cut-elimination of LJF provides a notion of subtitution, defined as follows:

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

If we write u as $L\langle y\rangle$, then $t[x/u] = L\langle t\{x{\leftarrow}y\}\rangle$.

Ex: $t = x_0[x_0{\leftarrow}\lambda y.x][x_1{\leftarrow}fx]$ and $u = x_2[x_2{\leftarrow}\lambda z.x_3[x_3{\leftarrow}gz]]$

Then $t[x/u] = x_0[x_0{\leftarrow}\lambda y.x_2][x_1{\leftarrow}f\,x_2][x_2{\leftarrow}\lambda z.x_3[x_3{\leftarrow}gz]]$

λ_{pos}: Substitution

As mentioned before, cut-elimination of LJF provides a notion of subtitution, defined as follows:

$$t[x/y] = t\{x \leftarrow y\}$$
$$t[x/u[y \leftarrow zw]] = t[x/u][y \leftarrow zw]$$
$$t[x/r[y \leftarrow \lambda z.u]] = t[x/r][y \leftarrow \lambda z.u]$$

If we write u as $L\langle y \rangle$, then $t[x/u] = L\langle t\{x \leftarrow y\} \rangle$.

Ex: $t = x_0[x_0 \leftarrow \lambda y.x][x_1 \leftarrow fx]$ and $u = x_2[x_2 \leftarrow \lambda z.x_3[x_3 \leftarrow gz]]$

Then $t[x/u] = x_0[x_0 \leftarrow \lambda y.x_2][x_1 \leftarrow f\,x_2][x_2 \leftarrow \lambda z.x_3[x_3 \leftarrow gz]]$

λ_{pos}: Substitution

As mentioned before, cut-elimination of LJF provides a notion of subtitution, defined as follows:

$$t[x/y] = t\{x{\leftarrow}y\}$$
$$t[x/u[y{\leftarrow}zw]] = t[x/u][y{\leftarrow}zw]$$
$$t[x/r[y{\leftarrow}\lambda z.u]] = t[x/r][y{\leftarrow}\lambda z.u]$$

If we write u as $L\langle y\rangle$, then $t[x/u] = L\langle t\{x{\leftarrow}y\}\rangle$.

Ex: $t = x_0[x_0{\leftarrow}\lambda y.x][x_1{\leftarrow}fx]$ and $u = x_2[x_2{\leftarrow}\lambda z.x_3[x_3{\leftarrow}gz]]$

Then $t[x/u] = x_0[x_0{\leftarrow}\lambda y.x_2][x_1{\leftarrow}f\,x_2][x_2{\leftarrow}\lambda z.x_3[x_3{\leftarrow}gz]]$

λ_{pos}: Unfolding and Equality

How to compare a λ_{pos}-term with a usual λ-term?

We can unfold all the explicit substitutions.

Definition (Unfolding)

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

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

How to compare two λ_{pos}-terms? Compare their unfoldings.

Not a good idea because of size explosion.
$\hookrightarrow$ Bisimulation on λ-graphs [CAC19]

λ_{pos}: Unfolding and Equality

How to compare a λ_{pos}-term with a usual λ-term?

We can unfold all the explicit substitutions.

Definition (Unfolding)

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

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

How to compare two λ_{pos}-terms? Compare their unfoldings.

Not a good idea because of size explosion.
$\hookrightarrow$ Bisimulation on λ-graphs [CAC19]

λ_{pos}: Unfolding and Equality

How to compare a λ_{pos}-term with a usual λ-term?

We can unfold all the explicit substitutions.

Definition (Unfolding)

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

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

How to compare two λ_{pos}-terms? Compare their unfoldings.

Not a good idea because of size explosion.
$\hookrightarrow$ Bisimulation on λ-graphs [CAC19]

λ_{pos}: Unfolding and Equality

How to compare a λ_{pos}-term with a usual λ-term?

We can unfold all the explicit substitutions.

Definition (Unfolding)

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

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

How to compare two λ_{pos}-terms? Compare their unfoldings.

Not a good idea because of size explosion.
$\hookrightarrow$ Bisimulation on λ-graphs [CAC19]

λ_{pos}: Unfolding and Equality

How to compare a λ_{pos}-term with a usual λ-term?

We can unfold all the explicit substitutions.

Definition (Unfolding)

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

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

How to compare two λ_{pos}-terms? Compare their unfoldings.

Not a good idea because of size explosion.
$\hookrightarrow$ Bisimulation on λ-graphs [CAC19]

λ_{pos}: Unfolding and Equality

How to compare a λ_{pos}-term with a usual λ-term?

We can unfold all the explicit substitutions.

Definition (Unfolding)

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

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

How to compare two λ_{pos}-terms? Compare their unfoldings.

Not a good idea because of size explosion.
$\hookrightarrow$ Bisimulation on λ-graphs [CAC19]

λ_{pos}: Unfolding and Equality

How to compare a λ_{pos}-term with a usual λ-term?

We can unfold all the explicit substitutions.

Definition (Unfolding)

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

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

How to compare two λ_{pos}-terms? Compare their unfoldings.

Not a good idea because of size explosion.
$\hookrightarrow$ Bisimulation on λ-graphs [CAC19]

λ_{pos}: Unfolding and Equality

How to compare a λ_{pos}-term with a usual λ-term?

We can unfold all the explicit substitutions.

Definition (Unfolding)

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

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

How to compare two λ_{pos}-terms? Compare their unfoldings.

Not a good idea because of size explosion.
$\hookrightarrow$ Bisimulation on λ-graphs [CAC19]

λ_{pos}: Reduction

How should we evaluate a λ_{pos}-term t?

First, λ_{pos}-terms are cut-free LJF proofs.

A possible way is to compute its unfolding $\underline{t}$ and then apply β-reduction in the untyped λ-calculus. If so, we can refer to the β-normal form of $\underline{t}$ as the meaning of t.

However, this can be costly as $\underline{t}$ might have exponential size with respect to that of t.

As a result, we look for a reduction for λ_{pos}-terms that is compatible with β-reduction on their unfoldings.

λ_{pos}: Reduction

How should we evaluate a λ_{pos}-term t?

First, λ_{pos}-terms are cut-free LJF proofs.

A possible way is to compute its unfolding $\underline{t}$ and then apply β-reduction in the untyped λ-calculus. If so, we can refer to the β-normal form of $\underline{t}$ as the meaning of t.

However, this can be costly as $\underline{t}$ might have exponential size with respect to that of t.

As a result, we look for a reduction for λ_{pos}-terms that is compatible with β-reduction on their unfoldings.

λ_{pos}: Reduction

How should we evaluate a λ_{pos}-term t?

First, λ_{pos}-terms are cut-free LJF proofs.

A possible way is to compute its unfolding $\underline{t}$ and then apply β-reduction in the untyped λ-calculus. If so, we can refer to the β-normal form of $\underline{t}$ as the meaning of t.

However, this can be costly as $\underline{t}$ might have exponential size with respect to that of t.

As a result, we look for a reduction for λ_{pos}-terms that is compatible with β-reduction on their unfoldings.

λ_{pos}: Reduction

How should we evaluate a λ_{pos}-term t?

First, λ_{pos}-terms are cut-free LJF proofs.

A possible way is to compute its unfolding $\underline{t}$ and then apply β-reduction in the untyped λ-calculus. If so, we can refer to the β-normal form of $\underline{t}$ as the meaning of t.

However, this can be costly as $\underline{t}$ might have exponential size with respect to that of t.

As a result, we look for a reduction for λ_{pos}-terms that is compatible with β-reduction on their unfoldings.

λ_{pos}: Reduction

How should we evaluate a λ_{pos}-term t?

First, λ_{pos}-terms are cut-free LJF proofs.

A possible way is to compute its unfolding $\underline{t}$ and then apply β-reduction in the untyped λ-calculus. If so, we can refer to the β-normal form of $\underline{t}$ as the meaning of t.

However, this can be costly as $\underline{t}$ might have exponential size with respect to that of t.

As a result, we look for a reduction for λ_{pos}-terms that is compatible with β-reduction on their unfoldings.

λ_{pos}: Reduction

How should we evaluate a λ_{pos}-term t?

First, λ_{pos}-terms are cut-free LJF proofs.

A possible way is to compute its unfolding $\underline{t}$ and then apply β-reduction in the untyped λ-calculus. If so, we can refer to the β-normal form of $\underline{t}$ as the meaning of t.

However, this can be costly as $\underline{t}$ might have exponential size with respect to that of t.

As a result, we look for a reduction for λ_{pos}-terms that is compatible with β-reduction on their unfoldings.

λ_{pos}: Reduction

We work with cut-free proofs. So we cannot simply apply cut-elimination. How should we define reduction then?

Here is 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'

We also consider a gc rule for garbage collection.

$$\begin{array}{rcl} C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ t[x\leftarrow\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x\notin fv(t)) \end{array}$$

λ_{pos}: Reduction

We work with cut-free proofs. So we cannot simply apply cut-elimination. How should we define reduction then?

Here is 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'

We also consider a gc rule for garbage collection.

$$\begin{array}{rcl} C\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] & \mapsto_{\mathsf{beta}} & C\langle L\langle t\{x{\leftarrow}w'\}\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\ t[x{\leftarrow}\lambda y.u] & \mapsto_{\mathsf{gc}} & t \qquad (x \notin fv(t)) \end{array}$$

λ_{pos}: Reduction

We work with cut-free proofs. So we cannot simply apply cut-elimination. How should we define reduction then?

Here is 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'

We also consider a gc rule for garbage collection.

$$\begin{array}{rcl} C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\mathrm{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ t[x\leftarrow\lambda y.u] & \mapsto_{\mathrm{gc}} & t \qquad (x\notin fv(t)) \end{array}$$

λ_{pos}: Reduction

We work with cut-free proofs. So we cannot simply apply cut-elimination. How should we define reduction then?

Here is 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'

We also consider a gc rule for garbage collection.

$$\begin{array}{rcl} C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ t[x\leftarrow\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x\notin fv(t)) \end{array}$$

λ_{pos}: Reduction

We work with cut-free proofs. So we cannot simply apply cut-elimination. How should we define reduction then?

Here is 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'

We also consider a gc rule for garbage collection.

$$\begin{array}{rcl} C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ t[x\leftarrow\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x\notin fv(t)) \end{array}$$

λ_{pos}: Reduction

We work with cut-free proofs. So we cannot simply apply cut-elimination. How should we define reduction then?

Here is 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'

We also consider a gc rule for garbage collection.

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

λ_{pos}: Reduction

We work with cut-free proofs. So we cannot simply apply cut-elimination. How should we define reduction then?

Here is 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'

We also consider a gc rule for garbage collection.

$$
\begin{array}{rcl}
C\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] & \mapsto_{\mathsf{beta}} & C\langle L\langle t\{x{\leftarrow}w'\}\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\
t[x{\leftarrow}\lambda y.u] & \mapsto_{\mathsf{gc}} & t \qquad (x \notin \mathit{fv}(t))
\end{array}
$$

λ_{pos}: Reduction

$$
\begin{array}{rcl}
C\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x{\leftarrow}w'\}\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\
t[x{\leftarrow}\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x \notin \mathit{fv}(t))
\end{array}
$$

An example:

$$
\begin{array}{rl}
 & x_2[x_2{\leftarrow}gx_1][x_1{\leftarrow}f\,x_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\
\to_{\text{beta}} & x_2[x_2{\leftarrow}g\,z_1][z_1{\leftarrow}hy_1][y_1{\leftarrow}g\,x_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\
\to_{\text{beta}} & z_1[z_1{\leftarrow}hy_1][y_1{\leftarrow}g\,x_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\
\to_{\text{beta}} & z_1[z_1{\leftarrow}hx_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\
\to_{\text{gc}} & z_1[z_1{\leftarrow}hx_0][g{\leftarrow}\lambda w.w] \\
\to_{\text{gc}} & z_1[z_1{\leftarrow}hx_0]
\end{array}
$$

We define $\to_{\text{pos}} = \to_{\text{beta}} \cup \to_{\text{gc}}$.

λ_{pos}: Reduction

$$
\begin{array}{rcl}
C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\mathrm{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\
t[x\leftarrow\lambda y.u] & \mapsto_{\mathrm{gc}} & t \qquad (x\notin fv(t))
\end{array}
$$

An example:

$$
\begin{array}{rl}
& x_2[x_2\leftarrow gx_1][x_1\leftarrow f\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{beta}} & x_2[x_2\leftarrow g\,z_1][z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{beta}} & z_1[z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{beta}} & z_1[z_1\leftarrow hx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{gc}} & z_1[z_1\leftarrow hx_0][g\leftarrow\lambda w.w] \\
\to_{\mathrm{gc}} & z_1[z_1\leftarrow hx_0]
\end{array}
$$

We define $\to_{\mathrm{pos}} = \to_{\mathrm{beta}} \cup \to_{\mathrm{gc}}$.

λ_{pos}: Reduction

$$
\begin{array}{rcl}
C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\mathsf{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\
t[x\leftarrow\lambda y.u] & \mapsto_{\mathsf{gc}} & t \qquad (x\notin \mathit{fv}(t))
\end{array}
$$

An example:

$$
\begin{array}{rl}
& x_2[x_2\leftarrow gx_1][x_1\leftarrow \underline{f\,x_0}][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathsf{beta}} & x_2[x_2\leftarrow gz_1][z_1\leftarrow hy_1][y_1\leftarrow gx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathsf{beta}} & z_1[z_1\leftarrow hy_1][y_1\leftarrow gx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathsf{beta}} & z_1[z_1\leftarrow hx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathsf{gc}} & z_1[z_1\leftarrow hx_0][g\leftarrow\lambda w.w] \\
\to_{\mathsf{gc}} & z_1[z_1\leftarrow hx_0]
\end{array}
$$

We define $\to_{\mathsf{pos}} = \to_{\mathsf{beta}} \cup \to_{\mathsf{gc}}$.

λ_{pos}: Reduction

$$\begin{array}{rcl} C\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] & \mapsto_{\mathrm{beta}} & C\langle L\langle t\{x{\leftarrow}w'\}\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\ t[x{\leftarrow}\lambda y.u] & \mapsto_{\mathrm{gc}} & t \qquad (x \notin fv(t)) \end{array}$$

An example:

$$\begin{array}{rl} & x_2[x_2{\leftarrow}gx_1][x_1{\leftarrow}\underline{f\,x_0}][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\ \to_{\mathrm{beta}} & x_2[x_2{\leftarrow}g\,z_1][z_1{\leftarrow}hy_1][y_1{\leftarrow}g\,x_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\ \to_{\mathrm{beta}} & z_1[z_1{\leftarrow}hy_1][y_1{\leftarrow}gx_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\ \to_{\mathrm{beta}} & z_1[z_1{\leftarrow}hx_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\ \to_{\mathrm{gc}} & z_1[z_1{\leftarrow}hx_0][g{\leftarrow}\lambda w.w] \\ \to_{\mathrm{gc}} & z_1[z_1{\leftarrow}hx_0] \end{array}$$

We define $\to_{\mathrm{pos}} = \to_{\mathrm{beta}} \cup \to_{\mathrm{gc}}$.

λ_{pos}: Reduction

$$
\begin{array}{rcl}
C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\mathrm{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\
t[x\leftarrow\lambda y.u] & \mapsto_{\mathrm{gc}} & t \qquad (x\notin fv(t))
\end{array}
$$

An example:

$$
\begin{array}{rl}
& x_2[x_2\leftarrow gx_1][x_1\leftarrow f\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{beta}} & x_2[x_2\leftarrow \underline{g\,z_1}][z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{beta}} & z_1[z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{beta}} & z_1[z_1\leftarrow hx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{gc}} & z_1[z_1\leftarrow hx_0][g\leftarrow\lambda w.w] \\
\to_{\mathrm{gc}} & z_1[z_1\leftarrow hx_0]
\end{array}
$$

We define $\to_{\mathrm{pos}} = \to_{\mathrm{beta}} \cup \to_{\mathrm{gc}}$.

λ_{pos}: Reduction

$$\begin{array}{rcl} C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ t[x\leftarrow\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x\notin fv(t)) \end{array}$$

An example:

$$\begin{array}{rl} & x_2[x_2\leftarrow gx_1][x_1\leftarrow f\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \to_{\text{beta}} & x_2[x_2\leftarrow \underline{g\,z_1}][z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \to_{\text{beta}} & z_1[z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \to_{\text{beta}} & z_1[z_1\leftarrow hx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \to_{\text{gc}} & z_1[z_1\leftarrow hx_0][g\leftarrow\lambda w.w] \\ \to_{\text{gc}} & z_1[z_1\leftarrow hx_0] \end{array}$$

We define $\to_{\text{pos}} = \to_{\text{beta}} \cup \to_{\text{gc}}$.

λ_{pos}: Reduction

$$\begin{array}{rcl} C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ t[x\leftarrow\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x\notin fv(t)) \end{array}$$

An example:

$$\begin{array}{rl} & x_2[x_2\leftarrow gx_1][x_1\leftarrow f\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \to_{\text{beta}} & x_2[x_2\leftarrow g\,z_1][z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \to_{\text{beta}} & z_1[z_1\leftarrow hy_1][y_1\leftarrow\underline{g\,x_0}][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \to_{\text{beta}} & z_1[z_1\leftarrow hx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \to_{\text{gc}} & z_1[z_1\leftarrow hx_0][g\leftarrow\lambda w.w] \\ \to_{\text{gc}} & z_1[z_1\leftarrow hx_0] \end{array}$$

We define $\to_{\text{pos}} = \to_{\text{beta}} \cup \to_{\text{gc}}$.

λ_{pos}: Reduction

$$C\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \mapsto_{\mathrm{beta}} C\langle L\langle t\{x{\leftarrow}w'\}\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle]$$
$$t[x{\leftarrow}\lambda y.u] \mapsto_{\mathrm{gc}} t \qquad (x \notin fv(t))$$

An example:

$$\begin{array}{ll}
 & x_2[x_2{\leftarrow}gx_1][x_1{\leftarrow}f\,x_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\
\to_{\mathrm{beta}} & x_2[x_2{\leftarrow}g\,z_1][z_1{\leftarrow}hy_1][y_1{\leftarrow}g\,x_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\
\to_{\mathrm{beta}} & z_1[z_1{\leftarrow}hy_1][y_1{\leftarrow}\underline{g\,x_0}][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\
\to_{\mathrm{beta}} & z_1[z_1{\leftarrow}hx_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\
\to_{\mathrm{gc}} & z_1[z_1{\leftarrow}hx_0][g{\leftarrow}\lambda w.w] \\
\to_{\mathrm{gc}} & z_1[z_1{\leftarrow}hx_0]
\end{array}$$

We define $\to_{\mathrm{pos}} = \to_{\mathrm{beta}} \cup \to_{\mathrm{gc}}$.

λ_{pos}: Reduction

$$\begin{array}{rcl} C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ t[x\leftarrow\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x \notin fv(t)) \end{array}$$

An example:

$$\begin{array}{ll} & x_2[x_2\leftarrow gx_1][x_1\leftarrow f\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \rightarrow_{\text{beta}} & x_2[x_2\leftarrow g\,z_1][z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \rightarrow_{\text{beta}} & z_1[z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \rightarrow_{\text{beta}} & z_1[z_1\leftarrow hx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\ \rightarrow_{\text{gc}} & z_1[z_1\leftarrow hx_0][g\leftarrow\lambda w.w] \\ \rightarrow_{\text{gc}} & z_1[z_1\leftarrow hx_0] \end{array}$$

We define $\rightarrow_{\text{pos}} = \rightarrow_{\text{beta}} \cup \rightarrow_{\text{gc}}$.

λ_{pos}: Reduction

$$
\begin{array}{rcl}
C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\mathrm{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\
t[x\leftarrow\lambda y.u] & \mapsto_{\mathrm{gc}} & t \qquad (x\notin fv(t))
\end{array}
$$

An example:

$$
\begin{array}{rl}
 & x_2[x_2\leftarrow gx_1][x_1\leftarrow f\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{beta}} & x_2[x_2\leftarrow g\,z_1][z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{beta}} & z_1[z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{beta}} & z_1[z_1\leftarrow hx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\mathrm{gc}} & z_1[z_1\leftarrow hx_0][g\leftarrow\lambda w.w] \\
\to_{\mathrm{gc}} & z_1[z_1\leftarrow hx_0]
\end{array}
$$

We define $\to_{\mathrm{pos}} = \to_{\mathrm{beta}} \cup \to_{\mathrm{gc}}$.

λ_{pos}: Reduction

$$
\begin{array}{rll}
C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\
t[x\leftarrow\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x\notin fv(t))
\end{array}
$$

An example:

$$
\begin{array}{rl}
 & x_2[x_2\leftarrow gx_1][x_1\leftarrow f\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\text{beta}} & x_2[x_2\leftarrow g\,z_1][z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\text{beta}} & z_1[z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\text{beta}} & z_1[z_1\leftarrow hx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\text{gc}} & z_1[z_1\leftarrow hx_0][g\leftarrow\lambda w.w] \\
\to_{\text{gc}} & z_1[z_1\leftarrow hx_0]
\end{array}
$$

We define $\to_{\text{pos}} = \to_{\text{beta}} \cup \to_{\text{gc}}$.

λ_{pos}: Reduction

$$
\begin{array}{rcl}
C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\
t[x\leftarrow\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x\notin fv(t))
\end{array}
$$

An example:

$$
\begin{array}{rl}
& x_2[x_2\leftarrow gx_1][x_1\leftarrow f\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\text{beta}} & x_2[x_2\leftarrow g\,z_1][z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\text{beta}} & z_1[z_1\leftarrow hy_1][y_1\leftarrow g\,x_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\text{beta}} & z_1[z_1\leftarrow hx_0][f\leftarrow\lambda x.z[z\leftarrow hy][y\leftarrow gx]][g\leftarrow\lambda w.w] \\
\to_{\text{gc}} & z_1[z_1\leftarrow hx_0][g\leftarrow\lambda w.w] \\
\to_{\text{gc}} & z_1[z_1\leftarrow hx_0]
\end{array}
$$

We define $\to_{\text{pos}} = \to_{\text{beta}} \cup \to_{\text{gc}}$.

λ_{pos}: Reduction

$$\begin{array}{rcl} C\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] & \mapsto_{\text{beta}} & C\langle L\langle t\{x{\leftarrow}w'\}\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\ t[x{\leftarrow}\lambda y.u] & \mapsto_{\text{gc}} & t \qquad (x \notin fv(t)) \end{array}$$

An example:

$$\begin{array}{rl} & x_2[x_2{\leftarrow}gx_1][x_1{\leftarrow}f\,x_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\ \to_{\text{beta}} & x_2[x_2{\leftarrow}g\,z_1][z_1{\leftarrow}hy_1][y_1{\leftarrow}g\,x_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\ \to_{\text{beta}} & z_1[z_1{\leftarrow}hy_1][y_1{\leftarrow}g\,x_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\ \to_{\text{beta}} & z_1[z_1{\leftarrow}hx_0][f{\leftarrow}\lambda x.z[z{\leftarrow}hy][y{\leftarrow}gx]][g{\leftarrow}\lambda w.w] \\ \to_{\text{gc}} & z_1[z_1{\leftarrow}hx_0][g{\leftarrow}\lambda w.w] \\ \to_{\text{gc}} & z_1[z_1{\leftarrow}hx_0] \end{array}$$

We define $\to_{\text{pos}} = \to_{\text{beta}} \cup \to_{\text{gc}}$.

λ_{pos}: Reduction

Like Plotkin's CBV λ-calculus, $\rightarrow_{\mathsf{pos}}$ is not terminating as shown by the term $x[x \leftarrow yy][y \leftarrow \lambda z.w[w \leftarrow zz]]$.

$\rightarrow_{\mathsf{pos}}$ is confluent.

The structural equivalence $\equiv$ is a strong bisimulation with respect to $\rightarrow_{\mathsf{pos}}$

- If $t \equiv u$ and $t \rightarrow_{\mathsf{pos}} t'$, then there exists u' such that $u \rightarrow_{\mathsf{pos}} u'$ and $t' \equiv u'$.

λ_{pos} is compatible with the untyped λ-calculus.

- If $t \rightarrow_{\mathsf{pos}} u$, then $\underline{t} \rightarrow^*_{\beta} \underline{u}$.
- If t is $\rightarrow_{\mathsf{pos}}$-normal, then $\underline{t}$ is β-normal.

λ_{pos}: Reduction

Like Plotkin's CBV λ-calculus, $\rightarrow_{\text{pos}}$ is not terminating as shown by the term $x[x{\leftarrow}yy][y{\leftarrow}\lambda z.w[w{\leftarrow}zz]]$.

$\rightarrow_{\text{pos}}$ is confluent.

The structural equivalence $\equiv$ is a strong bisimulation with respect to $\rightarrow_{\text{pos}}$

- If $t \equiv u$ and $t \rightarrow_{\text{pos}} t'$, then there exists u' such that $u \rightarrow_{\text{pos}} u'$ and $t' \equiv u'$.

λ_{pos} is compatible with the untyped λ-calculus.

- If $t \rightarrow_{\text{pos}} u$, then $\underline{t} \rightarrow_{\beta}^{*} \underline{u}$.
- If t is $\rightarrow_{\text{pos}}$-normal, then $\underline{t}$ is β-normal.

λ_{pos}: Reduction

Like Plotkin's CBV λ-calculus, $\rightarrow_{\mathsf{pos}}$ is not terminating as shown by the term $x[x{\leftarrow}yy][y{\leftarrow}\lambda z.w[w{\leftarrow}zz]]$.

$\rightarrow_{\mathsf{pos}}$ is confluent.

The structural equivalence $\equiv$ is a strong bisimulation with respect to $\rightarrow_{\mathsf{pos}}$

- If $t \equiv u$ and $t \rightarrow_{\mathsf{pos}} t'$, then there exists u' such that $u \rightarrow_{\mathsf{pos}} u'$ and $t' \equiv u'$.

λ_{pos} is compatible with the untyped λ-calculus.

- If $t \rightarrow_{\mathsf{pos}} u$, then $\underline{t} \rightarrow^*_{\beta} \underline{u}$.
- If t is $\rightarrow_{\mathsf{pos}}$-normal, then $\underline{t}$ is β-normal.

λ_{pos}: Reduction

Like Plotkin's CBV λ-calculus, $\to_{\text{pos}}$ is not terminating as shown by the term $x[x\leftarrow yy][y\leftarrow\lambda z.w[w\leftarrow zz]]$.

$\to_{\text{pos}}$ is confluent.

The structural equivalence $\equiv$ is a strong bisimulation with respect to $\to_{\text{pos}}$

- If $t \equiv u$ and $t \to_{\text{pos}} t'$, then there exists u' such that $u \to_{\text{pos}} u'$ and $t' \equiv u'$.

λ_{pos} is compatible with the untyped λ-calculus.

- If $t \to_{\text{pos}} u$, then $\underline{t} \to_{\beta}^{*} \underline{u}$.
- If t is $\to_{\text{pos}}$-normal, then $\underline{t}$ is β-normal.

λ_{pos}: Reduction

Like Plotkin's CBV λ-calculus, $\to_{\mathsf{pos}}$ is not terminating as shown by the term $x[x \leftarrow yy][y \leftarrow \lambda z.w[w \leftarrow zz]]$.

$\to_{\mathsf{pos}}$ is confluent.

The structural equivalence $\equiv$ is a strong bisimulation with respect to $\to_{\mathsf{pos}}$

- If $t \equiv u$ and $t \to_{\mathsf{pos}} t'$, then there exists u' such that $u \to_{\mathsf{pos}} u'$ and $t' \equiv u'$.

λ_{pos} is compatible with the untyped λ-calculus.

- If $t \to_{\mathsf{pos}} u$, then $\underline{t} \to_{\beta}^{*} \underline{u}$.
- If t is $\to_{\mathsf{pos}}$-normal, then $\underline{t}$ is β-normal.

λ_{pos} and VSC

λ_{pos} is closely related to Accattoli and Paolini's value substitution calculus (VSC), a call-by-value λ-calculus with explicit substitutions.

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

$$\begin{array}{rl}
\text{Multiplicative root rule} & L\langle\lambda x.t\rangle u \mapsto_{\mathsf{m}} L\langle t[x{\leftarrow}u]\rangle \\
\text{Exponential root rule} & t[x{\leftarrow}L\langle v\rangle] \mapsto_{\mathsf{e}} L\langle t\{x{\leftarrow}v\}\rangle
\end{array}$$

Note that all λ_{pos}-terms can be seen as VSC-terms.

λ_{pos} and VSC

λ_{pos} is closely related to Accattoli and Paolini's value substitution calculus (VSC), a call-by-value λ-calculus with explicit substitutions.

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

$$\begin{array}{rl} \text{Multiplicative root rule} & L\langle\lambda x.t\rangle u \mapsto_{\mathsf{m}} L\langle t[x{\leftarrow}u]\rangle \\ \text{Exponential root rule} & t[x{\leftarrow}L\langle v\rangle] \mapsto_{\mathsf{e}} L\langle t\{x{\leftarrow}v\}\rangle \end{array}$$

Note that all λ_{pos}-terms can be seen as VSC-terms.

λ_{pos} and VSC

λ_{pos} is closely related to Accattoli and Paolini's value substitution calculus (VSC), a call-by-value λ-calculus with explicit substitutions.

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

$$
\begin{array}{rl}
\text{Multiplicative root rule} & L\langle\lambda x.t\rangle u \mapsto_{\mathsf{m}} L\langle t[x{\leftarrow}u]\rangle \\
\text{Exponential root rule} & t[x{\leftarrow}L\langle v\rangle] \mapsto_{\mathsf{e}} L\langle t\{x{\leftarrow}v\}\rangle
\end{array}
$$

Note that all λ_{pos}-terms can be seen as VSC-terms.

λ_{pos} and VSC

λ_{pos} is closely related to Accattoli and Paolini's value substitution calculus (VSC), a call-by-value λ-calculus with explicit substitutions.

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

$$
\begin{array}{rrcl}
\text{Multiplicative root rule} & L\langle\lambda x.t\rangle u & \mapsto_{\mathsf{m}} & L\langle t[x{\leftarrow}u]\rangle \\
\text{Exponential root rule} & t[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathsf{e}} & L\langle t\{x{\leftarrow}v\}\rangle
\end{array}
$$

Note that all λ_{pos}-terms can be seen as VSC-terms.

Usefulness

For example, consider the term

$$z[z \leftarrow fx][f \leftarrow \lambda x_0.x_3[x_3 \leftarrow G(x_2)][x_2 \leftarrow G(x_1)][x_1 \leftarrow G(x_0)]][x \leftarrow \lambda y.t]$$

where $G(u) = \lambda y_0.y_3[y_3 \leftarrow y_1 y_2][y_2 \leftarrow gu][y_1 \leftarrow gu]$ with g a fixed variable and t a normal term in λ_{pos}.
After one beta-step and one gc-step, we obtain a λ_{pos}-normal term

$$x_3'[x_3' \leftarrow G(x_2')][x_2' \leftarrow G(x_1')][x_1' \leftarrow G(x)][x \leftarrow \lambda y.t]$$

However, in the VSC, we have

$$\begin{aligned}
& x_3'[x_3' \leftarrow G(x_2')][x_2' \leftarrow G(x_1')][x_1' \leftarrow G(x)][x \leftarrow \lambda y.t] \\
\rightarrow_{\mathrm{e}}\ & x_3'[x_3' \leftarrow G(x_2')][x_2' \leftarrow G(x_1')][x_1' \leftarrow G(\lambda y.t)] \\
\rightarrow_{\mathrm{e}}\ & x_3'[x_3' \leftarrow G(x_2')][x_2' \leftarrow G(G(\lambda y.t))] \\
\rightarrow_{\mathrm{e}}\ & x_3'[x_3' \leftarrow G(G(G(\lambda y.t)))] \\
\rightarrow_{\mathrm{e}}\ & G(G(G(\lambda y.t)))
\end{aligned}$$

Usefulness

For example, consider the term

$$z[z \leftarrow fx][f \leftarrow \lambda x_0.x_3[x_3 \leftarrow G(x_2)][x_2 \leftarrow G(x_1)][x_1 \leftarrow G(x_0)]][x \leftarrow \lambda y.t]$$

where $G(u) = \lambda y_0.y_3[y_3 \leftarrow y_1 y_2][y_2 \leftarrow gu][y_1 \leftarrow gu]$ with g a fixed variable and t a normal term in λ_{pos}.

After one beta-step and one gc-step, we obtain a λ_{pos}-normal term

$$x_3'[x_3' \leftarrow G(x_2')][x_2' \leftarrow G(x_1')][x_1' \leftarrow G(x)][x \leftarrow \lambda y.t]$$

However, in the VSC, we have

$$\begin{aligned}
& x_3'[x_3' \leftarrow G(x_2')][x_2' \leftarrow G(x_1')][x_1' \leftarrow G(x)][x \leftarrow \lambda y.t] \\
\rightarrow_{\mathsf{e}}\ & x_3'[x_3' \leftarrow G(x_2')][x_2' \leftarrow G(x_1')][x_1' \leftarrow G(\lambda y.t)] \\
\rightarrow_{\mathsf{e}}\ & x_3'[x_3' \leftarrow G(x_2')][x_2' \leftarrow G(G(\lambda y.t))] \\
\rightarrow_{\mathsf{e}}\ & x_3'[x_3' \leftarrow G(G(G(\lambda y.t)))] \\
\rightarrow_{\mathsf{e}}\ & G(G(G(\lambda y.t)))
\end{aligned}$$

Usefulness

For example, consider the term

$$z[z{\leftarrow}fx][f{\leftarrow}\lambda x_0.x_3[x_3{\leftarrow}G(x_2)][x_2{\leftarrow}G(x_1)][x_1{\leftarrow}G(x_0)]][x{\leftarrow}\lambda y.t]$$

where $G(u) = \lambda y_0.y_3[y_3{\leftarrow}y_1y_2][y_2{\leftarrow}gu][y_1{\leftarrow}gu]$ with g a fixed variable and t a normal term in λ_{pos}.
After one beta-step and one gc-step, we obtain a λ_{pos}-normal term

$$x_3'[x_3'{\leftarrow}G(x_2')][x_2'{\leftarrow}G(x_1')][x_1'{\leftarrow}G(x)][x{\leftarrow}\lambda y.t]$$

However, in the VSC, we have

$$\begin{aligned} & x_3'[x_3'{\leftarrow}G(x_2')][x_2'{\leftarrow}G(x_1')][x_1'{\leftarrow}G(x)][x{\leftarrow}\lambda y.t] \\ \to_{\mathsf{e}}\ & x_3'[x_3'{\leftarrow}G(x_2')][x_2'{\leftarrow}G(x_1')][x_1'{\leftarrow}G(\lambda y.t)] \\ \to_{\mathsf{e}}\ & x_3'[x_3'{\leftarrow}G(x_2')][x_2'{\leftarrow}G(G(\lambda y.t))] \\ \to_{\mathsf{e}}\ & x_3'[x_3'{\leftarrow}G(G(G(\lambda y.t)))] \\ \to_{\mathsf{e}}\ & G(G(G(\lambda y.t))) \end{aligned}$$

Usefulness

For example, consider the term

$$z[z\leftarrow fx][f\leftarrow\lambda x_0.x_3[x_3\leftarrow G(x_2)][x_2\leftarrow G(x_1)][x_1\leftarrow G(x_0)]][x\leftarrow\lambda y.t]$$

where $G(u) = \lambda y_0.y_3[y_3\leftarrow y_1 y_2][y_2\leftarrow gu][y_1\leftarrow gu]$ with g a fixed variable and t a normal term in λ_{pos}.
After one beta-step and one gc-step, we obtain a λ_{pos}-normal term

$$x_3'[x_3'\leftarrow G(x_2')][x_2'\leftarrow G(x_1')][x_1'\leftarrow G(x)][x\leftarrow\lambda y.t]$$

However, in the VSC, we have

$$\begin{aligned}
& x_3'[x_3'\leftarrow G(x_2')][x_2'\leftarrow G(x_1')][x_1'\leftarrow G(x)][x\leftarrow\lambda y.t] \\
\to_{\mathsf{e}}\ & x_3'[x_3'\leftarrow G(x_2')][x_2'\leftarrow G(x_1')][x_1'\leftarrow G(\lambda y.t)] \\
\to_{\mathsf{e}}\ & x_3'[x_3'\leftarrow G(x_2')][x_2'\leftarrow G(G(\lambda y.t))] \\
\to_{\mathsf{e}}\ & x_3'[x_3'\leftarrow G(G(G(\lambda y.t)))] \\
\to_{\mathsf{e}}\ & G(G(G(\lambda y.t)))
\end{aligned}$$

Usefulness in the VSC

The key is to substitute a variable for an abstraction only when the variable is applicative, i.e., applied to some term.

This only makes sense when we treat substitutions one by one, instead of using meta-level substitution.
$\hookrightarrow$ micro-step variant of the VSC.

The exponential rule

$$t[x{\leftarrow}L\langle v\rangle] \mapsto_{\mathrm{e}} L\langle t\{x{\leftarrow}v\}\rangle$$

is replaced by the micro-step exponential and garbage collection rules.

$$\begin{array}{rcll} C\langle x\rangle[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathrm{e^{ms}}} & L\langle C\langle v\rangle[x{\leftarrow}v]\rangle & \\ t[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathrm{gc}} & L\langle t\rangle & (x \notin fv(t)) \end{array}$$

Usefulness in the VSC

The key is to substitute a variable for an abstraction only when the variable is applicative, i.e., applied to some term.

This only makes sense when we treat substitutions one by one, instead of using meta-level substitution.
$\hookrightarrow$ micro-step variant of the VSC.

The exponential rule

$$t[x{\leftarrow}L\langle v\rangle] \mapsto_{\mathsf{e}} L\langle t\{x{\leftarrow}v\}\rangle$$

is replaced by the micro-step exponential and garbage collection rules.

$$\begin{array}{rcll} C\langle x\rangle[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathsf{ems}} & L\langle C\langle v\rangle[x{\leftarrow}v]\rangle & \\ t[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathsf{gc}} & L\langle t\rangle & (x \notin fv(t)) \end{array}$$

Usefulness in the VSC

The key is to substitute a variable for an abstraction only when the variable is applicative, i.e., applied to some term.

This only makes sense when we treat substitutions one by one, instead of using meta-level substitution.
$\hookrightarrow$ micro-step variant of the VSC.

The exponential rule

$$t[x{\leftarrow}L\langle v\rangle] \mapsto_{\mathsf{e}} L\langle t\{x{\leftarrow}v\}\rangle$$

is replaced by the micro-step exponential and garbage collection rules.

$$\begin{array}{rcll} C\langle x\rangle[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathsf{ems}} & L\langle C\langle v\rangle[x{\leftarrow}v]\rangle & \\ t[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathsf{gc}} & L\langle t\rangle & (x \notin fv(t)) \end{array}$$

Usefulness in the VSC

The key is to substitute a variable for an abstraction only when the variable is applicative, i.e., applied to some term.

This only makes sense when we treat substitutions one by one, instead of using meta-level substitution.
$\hookrightarrow$ micro-step variant of the VSC.

The exponential rule

$$t[x{\leftarrow}L\langle v\rangle] \mapsto_{\mathsf{e}} L\langle t\{x{\leftarrow}v\}\rangle$$

is replaced by the micro-step exponential and garbage collection rules.

$$\begin{array}{rcll} C\langle x\rangle[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathsf{e^{ms}}} & L\langle C\langle v\rangle[x{\leftarrow}v]\rangle & \\ t[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathsf{gc}} & L\langle t\rangle & (x \notin fv(t)) \end{array}$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\to_{\text{e}} \subseteq \to^*_{\text{e}^{\text{ms}}} \to_{\text{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$\begin{array}{rcll} C\langle x\rangle[x\leftarrow L\langle v\rangle] & \mapsto_{\text{e}^{\text{ms}}} & L\langle C\langle v\rangle[x\leftarrow v]\rangle & \\ t[x\leftarrow L\langle v\rangle] & \mapsto_{\text{gc}} & L\langle t\rangle & (x \notin fv(t)) \end{array}$$

$$C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \mapsto_{\text{beta}} C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle]$$

If $x \in fv(t)$, then we have

$$\begin{array}{rl} & C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}} & C\langle t[x\leftarrow (\lambda w.L\langle w'\rangle)z]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\text{m}} & C\langle t[x\leftarrow L\langle w'\rangle[w\leftarrow z]]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to^*_{\text{e}^{\text{ms}}}\to_{\text{gc}} & C\langle t[x\leftarrow L\langle w'\rangle\{w\leftarrow z\}]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}} & C\langle L\langle t[x\leftarrow w']\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to^*_{\text{e}^{\text{ms}}}\to_{\text{gc}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \end{array}$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\to_{\mathsf{e}} \subseteq \to^*_{\mathsf{e}^{\mathsf{ms}}} \to_{\mathsf{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$\begin{array}{rcl} C\langle x\rangle[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathsf{e}^{\mathsf{ms}}} & L\langle C\langle v\rangle[x{\leftarrow}v]\rangle \\ t[x{\leftarrow}L\langle v\rangle] & \mapsto_{\mathsf{gc}} & L\langle t\rangle \qquad (x \notin \mathit{fv}(t)) \end{array}$$

$$C\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \mapsto_{\mathsf{beta}} C\langle L\langle t\{x{\leftarrow}w'\}\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle]$$

If $x \in \mathit{fv}(t)$, then we have

$$\begin{array}{rl} & C\langle t[x{\leftarrow}yz]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\ \to_{\mathsf{e}^{\mathsf{ms}}} & C\langle t[x{\leftarrow}(\lambda w.L\langle w'\rangle)z]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\ \to_{\mathsf{m}} & C\langle t[x{\leftarrow}L\langle w'\rangle[w{\leftarrow}z]]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\ \to^*_{\mathsf{e}^{\mathsf{ms}}}\to_{\mathsf{gc}} & C\langle t[x{\leftarrow}L\langle w'\rangle\{w{\leftarrow}z\}]\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\ \to_{\mathsf{e}^{\mathsf{ms}}} & C\langle L\langle t[x{\leftarrow}w']\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \\ \to^*_{\mathsf{e}^{\mathsf{ms}}}\to_{\mathsf{gc}} & C\langle L\langle t\{x{\leftarrow}w'\}\rangle\{w{\leftarrow}z\}\rangle[y{\leftarrow}\lambda w.L\langle w'\rangle] \end{array}$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\rightarrow_{\mathsf{e}} \subseteq \rightarrow^{*}_{\mathsf{e}^{\mathsf{ms}}} \rightarrow_{\mathsf{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$
\begin{array}{rcll}
C\langle x\rangle[x \leftarrow L\langle v\rangle] & \mapsto_{\mathsf{e}^{\mathsf{ms}}} & L\langle C\langle v\rangle[x \leftarrow v]\rangle & \\
t[x \leftarrow L\langle v\rangle] & \mapsto_{\mathsf{gc}} & L\langle t\rangle & (x \notin fv(t))
\end{array}
$$

$$
C\langle t[x \leftarrow yz]\rangle[y \leftarrow \lambda w.L\langle w'\rangle] \mapsto_{\mathsf{beta}} C\langle L\langle t\{x \leftarrow w'\}\rangle\{w \leftarrow z\}\rangle[y \leftarrow \lambda w.L\langle w'\rangle]
$$

If $x \in fv(t)$, then we have

$$
\begin{array}{rl}
& C\langle t[x \leftarrow yz]\rangle[y \leftarrow \lambda w.L\langle w'\rangle] \\
\rightarrow_{\mathsf{e}^{\mathsf{ms}}} & C\langle t[x \leftarrow (\lambda w.L\langle w'\rangle)z]\rangle[y \leftarrow \lambda w.L\langle w'\rangle] \\
\rightarrow_{\mathsf{m}} & C\langle t[x \leftarrow L\langle w'\rangle[w \leftarrow z]]\rangle[y \leftarrow \lambda w.L\langle w'\rangle] \\
\rightarrow^{*}_{\mathsf{e}^{\mathsf{ms}}} \rightarrow_{\mathsf{gc}} & C\langle t[x \leftarrow L\langle w'\rangle\{w \leftarrow z\}]\rangle[y \leftarrow \lambda w.L\langle w'\rangle] \\
\rightarrow_{\mathsf{e}^{\mathsf{ms}}} & C\langle L\langle t[x \leftarrow w']\rangle\{w \leftarrow z\}\rangle[y \leftarrow \lambda w.L\langle w'\rangle] \\
\rightarrow^{*}_{\mathsf{e}^{\mathsf{ms}}} \rightarrow_{\mathsf{gc}} & C\langle L\langle t\{x \leftarrow w'\}\rangle\{w \leftarrow z\}\rangle[y \leftarrow \lambda w.L\langle w'\rangle]
\end{array}
$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\rightarrow_{\text{e}} \subseteq \rightarrow^*_{\text{e}^{\text{ms}}} \rightarrow_{\text{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$\begin{array}{rcll} C\langle x\rangle[x\leftarrow L\langle v\rangle] & \mapsto_{\text{e}^{\text{ms}}} & L\langle C\langle v\rangle[x\leftarrow v]\rangle & \\ t[x\leftarrow L\langle v\rangle] & \mapsto_{\text{gc}} & L\langle t\rangle & (x \notin fv(t)) \end{array}$$

$$C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \mapsto_{\text{beta}} C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle]$$

If $x \in fv(t)$, then we have

$$\begin{array}{rl} & C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow_{\text{e}^{\text{ms}}} & C\langle t[x\leftarrow (\lambda w.L\langle w'\rangle)z]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow_{\text{m}} & C\langle t[x\leftarrow L\langle w'\rangle[w\leftarrow z]]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow^*_{\text{e}^{\text{ms}}}\rightarrow_{\text{gc}} & C\langle t[x\leftarrow L\langle w'\rangle\{w\leftarrow z\}]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow_{\text{e}^{\text{ms}}} & C\langle L\langle t[x\leftarrow w']\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow^*_{\text{e}^{\text{ms}}}\rightarrow_{\text{gc}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \end{array}$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\to_{\mathsf{e}} \subseteq \to^*_{\mathsf{e}^{\mathsf{ms}}} \to_{\mathsf{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$\begin{array}{rcll} C\langle x\rangle[x\leftarrow L\langle v\rangle] & \mapsto_{\mathsf{e}^{\mathsf{ms}}} & L\langle C\langle v\rangle[x\leftarrow v]\rangle & \\ t[x\leftarrow L\langle v\rangle] & \mapsto_{\mathsf{gc}} & L\langle t\rangle & (x \notin fv(t)) \end{array}$$

$$C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \mapsto_{\mathsf{beta}} C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle]$$

If $x \in fv(t)$, then we have

$$\begin{array}{rl} & C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to_{\mathsf{e}^{\mathsf{ms}}} & C\langle t[x\leftarrow(\lambda w.L\langle w'\rangle)z]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to_{\mathsf{m}} & C\langle t[x\leftarrow L\langle w'\rangle[w\leftarrow z]]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to^*_{\mathsf{e}^{\mathsf{ms}}}\to_{\mathsf{gc}} & C\langle t[x\leftarrow L\langle w'\rangle\{w\leftarrow z\}]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to_{\mathsf{e}^{\mathsf{ms}}} & C\langle L\langle t[x\leftarrow w']\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to^*_{\mathsf{e}^{\mathsf{ms}}}\to_{\mathsf{gc}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \end{array}$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\rightarrow_{\mathsf{e}} \subseteq \rightarrow^{*}_{\mathsf{e}^{\mathsf{ms}}} \rightarrow_{\mathsf{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$\begin{array}{rcll} C\langle x\rangle[x\leftarrow L\langle v\rangle] & \mapsto_{\mathsf{e}^{\mathsf{ms}}} & L\langle C\langle v\rangle[x\leftarrow v]\rangle & \\ t[x\leftarrow L\langle v\rangle] & \mapsto_{\mathsf{gc}} & L\langle t\rangle & \quad (x \notin \mathit{fv}(t)) \end{array}$$

$$C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \mapsto_{\mathsf{beta}} C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle]$$

If $x \in \mathit{fv}(t)$, then we have

$$\begin{array}{rl} & C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow_{\mathsf{e}^{\mathsf{ms}}} & C\langle t[x\leftarrow (\lambda w.L\langle w'\rangle)z]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow_{\mathsf{m}} & C\langle t[x\leftarrow L\langle w'\rangle[w\leftarrow z]]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow^{*}_{\mathsf{e}^{\mathsf{ms}}}\rightarrow_{\mathsf{gc}} & C\langle t[x\leftarrow L\langle w'\rangle\{w\leftarrow z\}]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow_{\mathsf{e}^{\mathsf{ms}}} & C\langle L\langle t[x\leftarrow w']\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow^{*}_{\mathsf{e}^{\mathsf{ms}}}\rightarrow_{\mathsf{gc}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \end{array}$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\to_{\text{e}} \subseteq \to_{\text{e}^{\text{ms}}}^{*} \to_{\text{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$\begin{array}{rcll} C\langle x\rangle[x\leftarrow L\langle v\rangle] & \mapsto_{\text{e}^{\text{ms}}} & L\langle C\langle v\rangle[x\leftarrow v]\rangle & \\ t[x\leftarrow L\langle v\rangle] & \mapsto_{\text{gc}} & L\langle t\rangle & (x\notin \mathit{fv}(t)) \end{array}$$

$$C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \mapsto_{\text{beta}} C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle]$$

If $x\in \mathit{fv}(t)$, then we have

$$\begin{array}{rl} & C\langle t[x\leftarrow yz]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}} & C\langle t[x\leftarrow(\lambda w.L\langle w'\rangle)z]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to_{\text{m}} & C\langle t[x\leftarrow L\langle w'\rangle[w\leftarrow z]]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}}^{*}\to_{\text{gc}} & C\langle t[x\leftarrow L\langle w'\rangle\{w\leftarrow z\}]\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}} & C\langle L\langle t[x\leftarrow w']\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}}^{*}\to_{\text{gc}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow\lambda w.L\langle w'\rangle] \end{array}$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\to_{\text{e}} \subseteq \to_{\text{e}^{\text{ms}}}^{*} \to_{\text{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$\begin{array}{rcll} C\langle x\rangle[x\leftarrow L\langle v\rangle] & \mapsto_{\text{e}^{\text{ms}}} & L\langle C\langle v\rangle[x\leftarrow v]\rangle & \\ t[x\leftarrow L\langle v\rangle] & \mapsto_{\text{gc}} & L\langle t\rangle & (x \notin fv(t)) \end{array}$$

$$C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \mapsto_{\text{beta}} C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle]$$

If $x \in fv(t)$, then we have

$$\begin{array}{rl} & C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}} & C\langle t[x\leftarrow (\lambda w.L\langle w'\rangle)z]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\text{m}} & C\langle t[x\leftarrow L\langle w'\rangle[w\leftarrow z]]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}}^{*}\to_{\text{gc}} & C\langle t[x\leftarrow L\langle w'\rangle\{w\leftarrow z\}]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}} & C\langle L\langle t[x\leftarrow w']\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\text{e}^{\text{ms}}}^{*}\to_{\text{gc}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \end{array}$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\to_{\mathsf{e}} \subseteq \to_{\mathsf{e}^{\mathsf{ms}}}^{*} \to_{\mathsf{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$\begin{array}{rll} C\langle x\rangle[x\leftarrow L\langle v\rangle] & \mapsto_{\mathsf{e}^{\mathsf{ms}}} & L\langle C\langle v\rangle[x\leftarrow v]\rangle \\ t[x\leftarrow L\langle v\rangle] & \mapsto_{\mathsf{gc}} & L\langle t\rangle \qquad (x \notin fv(t)) \end{array}$$

$$C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \mapsto_{\mathsf{beta}} C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle]$$

If $x \in fv(t)$, then we have

$$\begin{array}{rl} & C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\mathsf{e}^{\mathsf{ms}}} & C\langle t[x\leftarrow (\lambda w.L\langle w'\rangle)z]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\mathsf{m}} & C\langle t[x\leftarrow L\langle w'\rangle[w\leftarrow z]]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\mathsf{e}^{\mathsf{ms}}}^{*}\to_{\mathsf{gc}} & C\langle t[x\leftarrow L\langle w'\rangle\{w\leftarrow z\}]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\mathsf{e}^{\mathsf{ms}}} & C\langle L\langle t[x\leftarrow w']\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \to_{\mathsf{e}^{\mathsf{ms}}}^{*}\to_{\mathsf{gc}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \end{array}$$

λ_{pos} and the micro-step VSC

The e-rule can be simulated using the e^{ms} and gc rules ($\rightarrow_{\mathrm{e}} \subseteq \rightarrow^{*}_{\mathrm{e}^{\mathrm{ms}}} \rightarrow_{\mathrm{gc}}$).

λ_{pos} can be simulated in the micro-step VSC.

$$\begin{array}{rcll} C\langle x\rangle[x\leftarrow L\langle v\rangle] & \mapsto_{\mathrm{e}^{\mathrm{ms}}} & L\langle C\langle v\rangle[x\leftarrow v]\rangle & \\ t[x\leftarrow L\langle v\rangle] & \mapsto_{\mathrm{gc}} & L\langle t\rangle & (x \notin fv(t)) \end{array}$$

$$C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \mapsto_{\mathrm{beta}} C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle]$$

If $x \in fv(t)$, then we have

$$\begin{array}{rl} & C\langle t[x\leftarrow yz]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow_{\mathrm{e}^{\mathrm{ms}}} & C\langle t[x\leftarrow (\lambda w.L\langle w'\rangle)z]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow_{\mathrm{m}} & C\langle t[x\leftarrow L\langle w'\rangle[w\leftarrow z]]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow^{*}_{\mathrm{e}^{\mathrm{ms}}}\rightarrow_{\mathrm{gc}} & C\langle t[x\leftarrow L\langle w'\rangle\{w\leftarrow z\}]\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow_{\mathrm{e}^{\mathrm{ms}}} & C\langle L\langle t[x\leftarrow w']\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \\ \rightarrow^{*}_{\mathrm{e}^{\mathrm{ms}}}\rightarrow_{\mathrm{gc}} & C\langle L\langle t\{x\leftarrow w'\}\rangle\{w\leftarrow z\}\rangle[y\leftarrow \lambda w.L\langle w'\rangle] \end{array}$$

Some comments on usefulness

Usefulness is not a new subject. Some similar considerations can be found in the literature, mostly at the level of abstract machines. See [ACC21] for example.

The novelty here is that we did not choose to consider only useful substitutions.

It is somehow enforced by the structure of λ_{pos}-terms.

In some way, λ_{pos} can be seem as the useful part of the VSC (ongoing work with Accattoli).

Some comments on usefulness

Usefulness is not a new subject. Some similar considerations can be found in the literature, mostly at the level of abstract machines. See [ACC21] for example.

The novelty here is that we did not choose to consider only useful substitutions.

It is somehow enforced by the structure of λ_{pos}-terms.

In some way, λ_{pos} can be seem as the useful part of the VSC (ongoing work with Accattoli).

Some comments on usefulness

Usefulness is not a new subject. Some similar considerations can be found in the literature, mostly at the level of abstract machines. See [ACC21] for example.

The novelty here is that we did not choose to consider only useful substitutions.

It is somehow enforced by the structure of λ_{pos}-terms.

In some way, λ_{pos} can be seem as the useful part of the VSC (ongoing work with Accattoli).

Some comments on usefulness

Usefulness is not a new subject. Some similar considerations can be found in the literature, mostly at the level of abstract machines. See [ACC21] for example.

The novelty here is that we did not choose to consider only useful substitutions.

It is somehow enforced by the structure of λ_{pos}-terms.

In some way, λ_{pos} can be seem as the useful part of the VSC (ongoing work with Accattoli).

Some comments on usefulness

Usefulness is not a new subject. Some similar considerations can be found in the literature, mostly at the level of abstract machines. See [ACC21] for example.

The novelty here is that we did not choose to consider only useful substitutions.

It is somehow enforced by the structure of λ_{pos}-terms.

In some way, λ_{pos} can be seem as the useful part of the VSC (ongoing work with Accattoli).

Graphical representation for λ_{pos}

We also propose a graphical representation for λ_{pos}, called λ-graphs with bodies.

Some reasons why it is reasonable to consider a graphical representation:

- Focusing only induces a light canonical form for proofs. Permutations of phases (or synthetic inference rules) are still possible. Some have considered multi-focused proofs and maximal multi-focusing. However, it is not the case here.
- Reduction at a distance is more natural and easier to express on graphs.
  $\hookrightarrow$ graphs capture the structural equivalence on terms.

See [Wu23] for more details.

Graphical representation for λ_{pos}

We also propose a graphical representation for λ_{pos}, called λ-graphs with bodies.

Some reasons why it is reasonable to consider a graphical representation:

- Focusing only induces a light canonical form for proofs. Permutations of phases (or synthetic inference rules) are still possible. Some have considered multi-focused proofs and maximal multi-focusing. However, it is not the case here.
- Reduction at a distance is more natural and easier to express on graphs.
  $\hookrightarrow$ graphs capture the structural equivalence on terms.

See [Wu23] for more details.

Graphical representation for λ_{pos}

We also propose a graphical representation for λ_{pos}, called λ-graphs with bodies.

Some reasons why it is reasonable to consider a graphical representation:

- Focusing only induces a light canonical form for proofs. Permutations of phases (or synthetic inference rules) are still possible. Some have considered multi-focused proofs and maximal multi-focusing. However, it is not the case here.
- Reduction at a distance is more natural and easier to express on graphs.
  $\hookrightarrow$ graphs capture the structural equivalence on terms.

See [Wu23] for more details.

Graphical representation for λ_{pos}

We also propose a graphical representation for λ_{pos}, called λ-graphs with bodies.

Some reasons why it is reasonable to consider a graphical representation:

- Focusing only induces a light canonical form for proofs. Permutations of phases (or synthetic inference rules) are still possible. Some have considered multi-focused proofs and maximal multi-focusing. However, it is not the case here.
- Reduction at a distance is more natural and easier to express on graphs.
  $\hookrightarrow$ graphs capture the structural equivalence on terms.

See [Wu23] for more details.

Graphical representation for λ_{pos}

We also propose a graphical representation for λ_{pos}, called λ-graphs with bodies.

Some reasons why it is reasonable to consider a graphical representation:

- Focusing only induces a light canonical form for proofs. Permutations of phases (or synthetic inference rules) are still possible. Some have considered multi-focused proofs and maximal multi-focusing. However, it is not the case here.
- Reduction at a distance is more natural and easier to express on graphs.
  $\hookrightarrow$ graphs capture the structural equivalence on terms.

See [Wu23] for more details.

Graphical representation for λ_{pos}

We also propose a graphical representation for λ_{pos}, called λ-graphs with bodies.

Some reasons why it is reasonable to consider a graphical representation:

- Focusing only induces a light canonical form for proofs. Permutations of phases (or synthetic inference rules) are still possible. Some have considered multi-focused proofs and maximal multi-focusing. However, it is not the case here.
- Reduction at a distance is more natural and easier to express on graphs.
  $\hookrightarrow$ graphs capture the structural equivalence on terms.

See [Wu23] for more details.

Conclusion

We use the focused proof system LJF to design the structure of terms, but not to give (directly) a notion of computation
$\hookrightarrow$ proofs-as-terms instead of proofs-as-programs

We show that how different polarizations in LJF induce different styles of term structures.

We use the positive encoding of untyped λ-terms to show that the structure of terms (or proofs) guides us towards a reasonable notion of computation, in line with some existing calculi (notably the VSC) in the literature.

Conclusion

We use the focused proof system LJF to design the structure of terms, but not to give (directly) a notion of computation
$\hookrightarrow$ proofs-as-terms instead of proofs-as-programs

We show that how different polarizations in LJF induce different styles of term structures.

We use the positive encoding of untyped λ-terms to show that the structure of terms (or proofs) guides us towards a reasonable notion of computation, in line with some existing calculi (notably the VSC) in the literature.

Conclusion

We use the focused proof system LJF to design the structure of terms, but not to give (directly) a notion of computation
$\hookrightarrow$ proofs-as-terms instead of proofs-as-programs

We show that how different polarizations in LJF induce different styles of term structures.

We use the positive encoding of untyped λ-terms to show that the structure of terms (or proofs) guides us towards a reasonable notion of computation, in line with some existing calculi (notably the VSC) in the literature.

Conclusion

We use the focused proof system LJF to design the structure of terms, but not to give (directly) a notion of computation
$\hookrightarrow$ proofs-as-terms instead of proofs-as-programs

We show that how different polarizations in LJF induce different styles of term structures.

We use the positive encoding of untyped λ-terms to show that the structure of terms (or proofs) guides us towards a reasonable notion of computation, in line with some existing calculi (notably the VSC) in the literature.

References

[ACC21] Beniamino Accattoli, Andrea Condoluci, and Claudio Sacerdoti Coen. "Strong call-by-value is reasonable, implosively". In: *2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE. 2021, pp. 1–14.

[CAC19] Andrea Condoluci, Beniamino Accattoli, and Claudio Sacerdoti Coen. "Sharing equality is linear". In: *Proceedings of the 21st International Symposium on Principles and Practice of Declarative Programming*. 2019, pp. 1–14.

[Mar+22] Sonia Marin et al. "From axioms to synthetic inference rules via focusing". In: *Annals of Pure and Applied Logic* 173.5 (2022), pp. 1–32.

[MW23] Dale Miller and Jui-Hsuan Wu. "A positive perspective on term representations". In: *31st EACSL Annual Conference on Computer Science Logic (CSL 2023)*. 2023, 3:1–3:21.

[Wu23] Jui-Hsuan Wu. "Proofs as Terms, Terms as Graphs". In: *Asian Symposium on Programming Languages and Systems*. Springer. 2023, pp. 91–111.

Thank you for your attention!