advanced questions

unsolvable terms
complexity of β-reduction
probabilistic reductions

untyped (pure) λ-calculus
simply-typed System F ?
advanced systems (MLTT, Iris...)

decidable checking?
consistency?
advanced questions

unsolvable terms
complexity of $\beta$-reduction
probabilistic reductions

binders
effects

untyped (pure) $\lambda$-calculus
simply-typed System F?
advanced systems (MLTT, Iris...)

decidable checking?
consistency?
advanced questions

unsolvable terms
complexity of $\beta$-reduction
probabilistic reductions

binders
effects
proof nets

untyped (pure) $\lambda$-calculus
simply-typed System F ?
linear logic

decidable checking?
consistency?

advanced systems (MLTT, Iris...)
advanced questions

unsolvable terms

complexity of β-reduction

probabilistic reductions

binders
effects
proof nets
equivalence
canonicity

decidable checking?
consistency?

untyped (pure) λ-calculus

simply-typed System F ?
linear logic

advanced systems (MLTT, Iris...)
Section 1

Focusing
Focusing

Focusing is a technique from proof theory [Andreoli, 1992].

It studies **invertibility** of connectives to structure the search space.

Type theory perspective: canonical representations.

\[ t \approx_{\beta \eta} u \quad \Rightarrow \quad ? \quad \Rightarrow \quad t \approx_{\alpha} u \]
Invertible vs. non-invertible rules. Positives vs. negatives.
Invertible vs. non-invertible rules. Positives vs. negatives.

\[
\begin{align*}
\Gamma \vdash A & \quad \Gamma, B \vdash C \\
\Gamma, A \to B \vdash C & \\
\end{align*}
\]

\[
\begin{align*}
\Gamma, A_i \vdash C & \\
\Gamma, A_1 \times A_2 \vdash C & \\
\end{align*}
\]

\[
\begin{align*}
\Gamma, A_1 \vdash C & \quad \Gamma, A_2 \vdash C \\
\Gamma, A_1 + A_2 \vdash C & \\
\end{align*}
\]

\[
\begin{align*}
\Gamma, 0 \vdash C & \\
\Gamma \vdash 1 & \\
\end{align*}
\]

\[
N, M ::= A \to B \mid A \times B \mid 1 \\
A, B ::= P \mid N \mid \alpha \\
P_a, Q_a ::= P \mid \alpha \\
Na, Ma ::= N \mid \alpha \\
P, Q ::= A + B \mid 0
\]
Invertible phase

\[
\frac{\?}{\alpha + \beta \vdash \alpha}
\]

\[
\frac{\alpha + \beta \vdash \alpha}{\alpha + \beta \vdash \beta + \alpha}
\]

If applied too early, non-invertible rules can ruin your proof.

**Focusing restriction 1: invertible phases**

Invertible rules must be applied as soon and as long as possible – and their order does not matter.
Invertible phase

\[
\begin{align*}
? \\
\alpha + \beta \vdash \alpha \\
\alpha + \beta \vdash \beta + \alpha
\end{align*}
\]

If applied too early, non-invertible rules can ruin your proof.

**Focusing restriction 1: invertible phases**

Invertible rules must be applied as soon and as long as possible – and their order does not matter.

Imposing this restriction gives a single proof of \((\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \beta)\) instead of two \((\lambda f. f\) and \(\lambda f. \lambda x. f x)\).

After all invertible rules, negative context \(\Gamma_{na}\), positive goal \(P_a\).
Non-invertible phases

After all invertible rules, negative context, positive goal.

Only step forward: select a formula, apply some non-invertible rule on it.
Non-invertible phases

After all invertible rules, negative context, positive goal.

Only step forward: select a formula, apply some non-invertible rule on it.

**Focusing restriction 2: non-invertible phase**

When a principal formula is selected for non-invertible rule, they should be applied as long as possible – until its polarity changes.

\[ \alpha_2, \beta_1 \vdash A \]

\[ \alpha_2 \times \alpha_3, \beta_1 \times \beta_2 \vdash A \]

\[ \alpha_1 \times \alpha_2 \times \alpha_3, \beta_1 \times \beta_2 \vdash A \]
Non-invertible phases

After all invertible rules, negative context, positive goal.

Only step forward: select a formula, apply some non-invertible rule on it.

Focusing restriction 2: non-invertible phase

When a principal formula is selected for non-invertible rule, they should be applied as long as possible – until its polarity changes.

Completeness: this restriction preserves provability. **Non-trivial**!

Example of removed redundancy:

\[
\begin{align*}
\alpha_2, & \quad \beta_1 \vdash A \\
\alpha_2 \times \alpha_3, & \quad \beta_1 \vdash A \\
\alpha_2 \times \alpha_3, & \quad \beta_1 \times \beta_2 \vdash A \\
\alpha_1 \times \alpha_2 \times \alpha_3, & \quad \beta_1 \times \beta_2 \vdash A
\end{align*}
\]
This was focusing:
- invertible as long as a rule matches, until $\Gamma_{na} \vdash P_a$
- then pick a formula
- then non-invertible as long as a rule matches, until polarity change

Completeness:

\[
\Gamma \vdash A \quad \implies \quad \Gamma \vdash_{\text{foc}} A
\]
a focused natural deduction

\[ N, M ::= A \to B \mid A \times B \mid 1 \]
\[ P, Q ::= A + B \mid 0 \]
\[ A, B ::= P \mid N \mid \alpha \]
\[ P_a, Q_a ::= P \mid \alpha \]
\[ N_a, M_a ::= N \mid \alpha \]
\[ \Gamma_{na} ::= \emptyset \mid \Gamma_{na}, N_a \]

\( \Gamma_{na}; \Delta \vdash_{\text{inv}} A \) invertible phase (decomposes \( \Delta, A \))

\( \Gamma_{na} \vdash_{\text{foc}} P_a \) choice of focus

\( \Gamma_{na}; N \Downarrow M_a \) non-invertible negative rules

\( \Gamma_{na} \Uparrow P \) non-invertible positive rules

(inspired by Brock-Nannestad and Schürmann [2010])
(some simplifications, see Scherer [2016] for full details)
Section 2

Focused $\lambda$-calculus
**β-normal forms (negative)**

**β-short normal forms:**

\[
\pi_1 (t, u) = t
\]

\[
v, w ::= \lambda x. v | (v, w) | n
\]

\[
n, m ::= \pi_i n | n v | x
\]
\textbf{\(\beta\)-normal forms (negative)}

\textit{\(\beta\)-short normal forms:}

\[ \pi_1 (t, u) = t \]

\[ v, w ::= \lambda x. v \mid (v, w) \mid n \]

\[ n, m ::= \pi_i n \mid n v \mid x \]

\textit{\(\beta\)-short \(\eta\)-long:}

\[ (y : \alpha \rightarrow \beta) = \lambda x : \alpha. (y x : \beta) \]
**β-normal forms (negative)**

**β-short normal forms:**

\[ \pi_1 (t, u) = t \]

\[ v, w ::= \lambda x. v \mid (v, w) \mid n \]

\[ n, m ::= \pi_i n \mid n v \mid x \]

**β-short η-long:**

\[ (y : \alpha \rightarrow \beta) = \lambda x : \alpha. (y x : \beta) \]

\[ v, w ::= \lambda x. v \mid (v, w) \mid (n : \alpha) \]

\[ n, m ::= \pi_i n \mid n v \mid x \]
What about sums?

\[ v, w ::= \lambda x. v | (v, w) | \sigma_i v | (n : \alpha) \]

\[ n, m ::= \pi_i n | n v | \left( \text{match } n \text{ with } \begin{array}{l}
\sigma_1 y_1 \to v_1 \\
\sigma_2 y_2 \to v_2
\end{array} \right) \bigg| x \]

Does not work:

\[
\left( \text{match } n \text{ with } \begin{array}{l}
\sigma_1 y_1 \to \lambda z. v_1 \\
\sigma_2 y_2 \to \lambda z. v_2
\end{array} \right) v
\]

\[
\text{match } n \text{ with } \begin{array}{l}
\sigma_1 x \to \sigma_2 x \\
\sigma_2 x \to \sigma_1 x
\end{array}
\]
Focusing to the rescue

\[ \begin{align*}
v, w & ::= \lambda x. v \mid (v, w) \mid (n : \alpha) \\
n, m & ::= \pi_i n \mid n v \mid x
\end{align*} \]

\[ \downarrow \]

\[ \begin{align*}
v, w & ::= \lambda x. v \mid (v, w) \mid () \\
& \mid \text{absurd}(x) \mid \left( \text{match } x \text{ with } \begin{array}{c}
\sigma_1 y_1 \to v_1 \\
\sigma_2 y_2 \to v_2
\end{array} \right) \\
& \mid (\Gamma_{na} \vdash f : P_a)
\end{align*} \]

\[ \begin{align*}
n, m & ::= \pi_i n \mid n p \mid x \\
p, q & ::= \sigma_i p \mid (v : N_a)
\end{align*} \]

\[ f ::= (n : \alpha) \mid (p : P) \mid \text{let } x = (n : P) \text{ in } v \]

(See also Munch-Maccagnoni [2013])
Completeness of focusing

Logic:

\[ \Gamma \vdash A \quad \implies \quad \Gamma \vdash_{foc} A \]
Completeness of focusing

Logic:

\[ \Gamma \vdash A \implies \Gamma \vdash \text{foc} \ A \]

Programming:

\[ \Gamma \vdash t : A \implies \exists v, \Gamma \vdash \text{foc} \ v : A \quad v \approx_{\beta\eta} t \]
Canonicity

Focused normal forms are canonical for the impure $\lambda$-calculus.

Proof in Zeilberger [2009], using ideas from Girard’s ludics.
Canonicity

Focused normal forms are canonical for the impure $\lambda$-calculus.

Proof in Zeilberger [2009], using ideas from Girard’s ludics.

Not canonical for the pure calculus.

\[
\text{let } x = n \text{ in } C \left[ \text{let } x' = n' \text{ in } v \right]
\]

\[
\text{let } x' = n' \text{ in } C \left[ \text{let } x = n \text{ in } v \right]
\]
Canonicity

Focused normal forms are canonical for the impure $\lambda$-calculus.

Proof in Zeilberger [2009], using ideas from Girard’s ludics.

Not canonical for the **pure** calculus.

\[
\begin{align*}
\text{let } x = n \text{ in } C & \left[ \text{let } x' = n' \text{ in } v \right] \\
\text{let } x' = n' \text{ in } C & \left[ \text{let } x = n \text{ in } v \right]
\end{align*}
\]

Solution: “saturation” [Scherer, 2017]

\[
f ::= \quad \text{let } \bar{x} = \bar{n} \text{ in } v \mid (n : \alpha) \mid (p : P)
\]

inspired by multi-focusing [Chaudhuri, Miller, and Saurin, 2008].
Recap

\[ \Gamma_{na}; \Delta \vdash_{\text{inv}} \nu : A \]

\[ \nu, w ::= \lambda x. \nu \mid (\nu, w) \mid () \]

\[ \mid \text{absurd}(x) \mid \text{match } x \text{ with} \]

\[ \mid \sigma_1 y_1 \rightarrow \nu_1 \mid \sigma_2 y_2 \rightarrow \nu_2 \]

\[ \mid (\Gamma_{na} \vdash f : P_a) \]

\[ \Gamma_{na} \vdash n \downarrow N_a \]

\[ n, m ::= \nu \mid n \mid p \mid x \]

\[ \Gamma_{na} \vdash p \uparrow P_a \]

\[ p, q ::= \sigma_i p \mid (\nu : N_a) \]

\[ \Gamma_{na} \vdash_{\text{foc}} f : A \]

\[ f ::= \text{let } \bar{x} = (n : P) \text{ in } \nu \]

\[ \mid (n : \alpha) \mid (p : P) \]

(plus saturation conditions)

(decision diagrams!
Altenkirch and Uustalu [2004], Ahmad, Licata, and Harper [2010])
Applications

A clean way to extend our understanding to positives (+, 0).

- evaluation order in presence of effects
- which types have a unique inhabitant?
- decidability of equivalence
- Böhm separation results: contextual and \((\beta\eta)\) coincide
- \(\lambda\)-definability?
- (your result here!)
Section 3

Questions
Saturation for System F?

Termination of saturation: **subformula property**. Not in F!

\[ \Gamma, A[B/\alpha] \vdash C \]

\[ \Gamma \ni \forall \alpha. A \vdash C \]

Equivalence is undecidable in F: no decidable canonical forms.

Could we have a partial algorithm that works sometimes?
Eliminating polymorphism

Idea: probe the structure of $\forall\alpha. A$ through proof search.

$$
\Gamma \vdash A \quad \Gamma \vdash B
$$

$\Gamma \overset{\text{def}}{=} A \rightarrow B \rightarrow \alpha \vdash \alpha$

$\vdash \forall\alpha. (A \rightarrow B \rightarrow \alpha) \rightarrow \alpha$

(Ongoing discussions with Li-Yao Xia and Jean-Philippe Bernardy)
Eliminating polymorphism

Idea: probe the structure of $\forall \alpha. A$ through proof search.

$$
\frac{
\Gamma \vdash A \quad \Gamma \vdash B
}{
\Gamma \overset{\text{def}}{=} A \rightarrow B \rightarrow \alpha \vdash \alpha
}
\frac{
\Gamma \vdash A \quad \oplus \quad \Gamma \vdash B
}{
\Gamma \overset{\text{def}}{=} A \rightarrow \alpha, B \rightarrow \alpha \vdash \alpha
}
\frac{
\Gamma \vdash A \rightarrow \alpha, B \rightarrow \alpha \vdash \alpha
}{
\vdash \forall \alpha. (A \rightarrow \alpha) \rightarrow (B \rightarrow \alpha) \rightarrow \alpha
}
$$
Eliminating polymorphism

Idea: probe the structure of $\forall \alpha. A$ through proof search.

\[
\Gamma \vdash A \quad \Gamma \vdash B
\]

\[
\Gamma \overset{\text{def}}{=} A \to B \to \alpha \vdash \alpha
\]

\[
\vdash \forall \alpha. (A \to B \to \alpha) \to \alpha
\]

\[
\Gamma \vdash A \quad \Gamma \vdash B
\]

\[
\Gamma \overset{\text{def}}{=} A \to \alpha, B \to \alpha \vdash \alpha
\]

\[
\vdash \forall \alpha. (A \to \alpha) \to (B \to \alpha) \to \alpha
\]

\[
\Gamma \vdash \alpha \quad \Gamma \vdash \alpha
\]

\[
\Gamma \overset{\text{def}}{=} \alpha \to \alpha, \alpha \vdash \alpha
\]

\[
\vdash \forall \alpha. (\alpha \to \alpha) \to \alpha \to \alpha
\]

(Ongoing discussions with Li-Yao Xia and Jean-Philippe Bernardy)
The place of bi-directional systems?

Bidirectional systems: natural fit for normal forms.

\[
\Delta \in (x : T) \quad \frac{\Delta \vdash n_1 = n_2 \in (T \to U)}{\Delta \vdash T \ni v_1 = v_2}
\]

\[
\Delta, x : T \vdash U \ni v_1 x = v_2 x \quad \frac{\Delta \vdash n_1 = n_2 \in \alpha}{\Delta \vdash \alpha \ni n_1 = n_2}
\]

General programs? Program equivalence? Type inference?
Saturation in practice?

Is it possible to be efficient?

(in presence of software libraries?)

relations to program synthesis
Positives in richer systems?

$\eta$ for sums:

$$C[\square : A + B] = \text{match} \square \text{ with } \begin{align*}
\sigma_1 \times_1 & \rightarrow C[\sigma_1 \times_1] \\
\sigma_2 \times_2 & \rightarrow C[\sigma_2 \times_2]
\end{align*}$$

$\eta$ for natural numbers sounds very difficult!
Positives in richer systems?

η for sums:

\[ C[\square : A + B] = \text{match} \square \text{ with} \begin{cases} \sigma_1 x_1 \rightarrow C[\sigma_1 x_1] \\ \sigma_2 x_2 \rightarrow C[\sigma_2 x_2] \end{cases} \]

η for natural numbers sounds very difficult!

\[ C[\square : \mathbb{N}] = \text{rec}(\square, t_0, D_1) \]

\[ C \circ S = D_1 \circ H \]
\[ C \circ 0 = t_0 \]


