Deciding pure program equivalence with sums and the empty type

Gabriel Scherer

Northeastern University, Boston

March 1, 2017

Question

When are two program fragments t, u contextually equivalent?

$$\forall C, \qquad C\,[t] \approx C\,[u]$$

Specifics depend on the programming language: input/output, non-termination, just values?

Untyped λ-calculus: undecidable.
Simple type system $\Lambda\mathsf{C}(\alpha, \to)$: decidable.
Polymorphism $\Lambda\mathsf{C}(\alpha, \to, \forall)$, dependent types $\Lambda\mathsf{C}(\alpha, \to, \Pi)$: undecidable.

What's in the middle? Simple types, but richer datatypes?

History

Decidability of equivalence:

- $\Lambda\mathsf{C}(\alpha, \to)$: Tait, 1967 or earlier.
- $\Lambda\mathsf{C}(\alpha, \to, \times)$: essentially the same proof.
- $\Lambda\mathsf{C}(\alpha, \to, \times, 1)$: essentially the same proof.
- $\Lambda\mathsf{C}(\alpha, \to, \times, 1, +)$: Ghani, 1995; Altenkirch, Dybjer, Hoffman, Scott: 2001; Balat, Di Cosmo, Fiore: 2004.
- $\Lambda\mathsf{C}(\alpha, \to, \times, 1, +, 0)$: ? (this work)

Why are $(+, 0)$ so hard?

Section 1

Background

Simply-typed lambda-calculus

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.\, t : A \to B} \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma \vdash u : A}{\Gamma \vdash t\ u : B}$$

$$\frac{\Gamma \vdash t_1 : A_1 \quad \Gamma \vdash t_2 : A_2}{\Gamma \vdash (t_1, t_2) : A_1 \times A_2} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{\Gamma \vdash \pi_i\ t : A_i}$$

$$\frac{\Gamma \vdash t : A_i}{\Gamma \vdash \sigma_i\ t : A_1 + A_2} \qquad \frac{\Gamma \vdash t : A_1 + A_2 \qquad \begin{array}{l}\Gamma, x_1 : A_1 \vdash u_1 : C \\ \Gamma, x_2 : A_2 \vdash u_2 : C\end{array}}{\Gamma \vdash \texttt{match}\ t\ \texttt{with} \mid \begin{array}{l}\sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2\end{array} : C}$$

$$\frac{}{\Gamma \vdash () : 1} \qquad \frac{(x : A) \in \Gamma}{\Gamma \vdash x : A} \qquad \frac{\Gamma \vdash t : 0}{\Gamma \vdash \texttt{absurd}(t) : A}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad\qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad\qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad\qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad\qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \mid \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \triangleright_\eta \begin{array}{l} \texttt{match}\ t\ \texttt{with} \\ \mid\ \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \mid\ \sigma_2\ x_2 \to \sigma_2\ x_2 \end{array}}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \mid \sigma_1\ x_1 \to u_1 \mid \sigma_2\ x_2 \to u_2 \ \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \triangleright_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \mid\ \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \mid\ \sigma_2\ x_2 \to \sigma_2\ x_2\end{array}}$$

$$(t_1, t_2) \overset{?}{\approx}_\eta \begin{array}{l}\texttt{match}\ t_1\ \texttt{with} \\ \mid\ \sigma_1\ x_1 \to (\sigma_1\ x_1, t_2) \\ \mid\ \sigma_2\ x_2 \to (\sigma_2\ x_2, t_2)\end{array}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \mid \sigma_1\ x_1 \to u_1 \mid \sigma_2\ x_2 \to u_2 \ \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \triangleright_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \mid\ \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \mid\ \sigma_2\ x_2 \to \sigma_2\ x_2\end{array}}$$

$$(t_1, t_2) \overset{?}{\approx}_\eta \begin{array}{l}\texttt{match}\ t_1\ \texttt{with} \\ \mid\ \sigma_1\ x_1 \to (\sigma_1\ x_1, t_2) \\ \mid\ \sigma_2\ x_2 \to (\sigma_2\ x_2, t_2)\end{array}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \mid \sigma_1\ x_1 \to u_1 \mid \sigma_2\ x_2 \to u_2 \ \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \triangleright_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \mid \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \mid \sigma_2\ x_2 \to \sigma_2\ x_2\end{array}}$$

$$(t_1, t_2) \overset{?}{\approx}_\eta \begin{array}{l}\texttt{match}\ t_1\ \texttt{with} \\ \mid \sigma_1\ x_1 \to (\sigma_1\ x_1, t_2) \\ \mid \sigma_2\ x_2 \to (\sigma_2\ x_2, t_2)\end{array}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \mid \sigma_1\ x_1 \to u_1 \mid \sigma_2\ x_2 \to u_2 \ \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \triangleright_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \mid\ \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \mid\ \sigma_2\ x_2 \to \sigma_2\ x_2\end{array}}$$

$$(t_1, t_2) \overset{?}{\approx}_\eta \begin{array}{l}\texttt{match}\ t_1\ \texttt{with} \\ \mid\ \sigma_1\ x_1 \to (\sigma_1\ x_1, t_2) \\ \mid\ \sigma_2\ x_2 \to (\sigma_2\ x_2, t_2)\end{array}$$

$u[t_1/y]$ with $u \overset{\text{def}}{=} (y, t_2)$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \mid \sigma_1\ x_1 \to u_1 \mid \sigma_2\ x_2 \to u_2 \ \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \triangleright_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \mid\ \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \mid\ \sigma_2\ x_2 \to \sigma_2\ x_2\end{array}}$$

$$u[t_1/y] \overset{?}{\approx}_\eta \begin{array}{l}\texttt{match}\ t_1\ \texttt{with} \\ \mid\ \sigma_1\ x_1 \to (\sigma_1\ x_1, t_2) \\ \mid\ \sigma_2\ x_2 \to (\sigma_2\ x_2, t_2)\end{array}$$

$u[t_1/y]$ with $u \overset{\text{def}}{=} (y, t_2)$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \mid \sigma_1\ x_1 \to u_1 \mid \sigma_2\ x_2 \to u_2 \ \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \triangleright_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \mid \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \mid \sigma_2\ x_2 \to \sigma_2\ x_2\end{array}}$$

$$u[t_1/y] \overset{?}{\approx}_\eta \begin{array}{l}\texttt{match}\ t_1\ \texttt{with} \\ \mid \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \mid \sigma_2\ x_2 \to u[\sigma_2\ x_2/y]\end{array}$$

$u[t_1/y]$ with $u \overset{\text{def}}{=} (y, t_2)$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \mid \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \triangleright_\eta \begin{array}{l} \texttt{match}\ t\ \texttt{with} \\ \mid \begin{array}{l} \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \mid\ \sigma_2\ x_2 \to \sigma_2\ x_2 \end{array} \end{array}}$$

$$u[t_1/y] \approx_\eta \begin{array}{l} \texttt{match}\ t_1\ \texttt{with} \\ \mid \begin{array}{l} \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \mid\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array} \end{array}$$

$$u[t_1/y] \quad \text{with} \quad u \stackrel{\text{def}}{=} (y, t_2)$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad\qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \ \begin{array}{l} \texttt{match}\ t\ \texttt{with} \\ \Big|\ \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \Big|\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array}}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad\qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{l} \texttt{match}\ t\ \texttt{with} \\ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array} \end{array}}$$

$$\frac{\Gamma \vdash t : 0 \qquad \Gamma,\ y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad\qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \ \begin{array}{l} \texttt{match}\ t\ \texttt{with} \\ \Big|\ \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \Big|\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array}}$$

$$\frac{\Gamma \vdash t : 0 \qquad \Gamma,\ y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

Derived rules :

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad\qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{l} \texttt{match}\ t\ \texttt{with} \\ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array} \end{array}}$$

$$\frac{\Gamma \vdash t : 0 \qquad \Gamma,\ y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

Derived rules : $$\frac{}{\Gamma \vdash t_1 \approx_\eta t_2 : 1}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{l} \texttt{match}\ t\ \texttt{with} \\ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array} \end{array}}$$

$$\frac{\Gamma \vdash t : 0 \qquad \Gamma, y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

Derived rules :

$$\frac{}{\Gamma \vdash t_1 \approx_\eta t_2 : 1} \qquad \frac{\Gamma \vdash t : 0 \qquad \Gamma \vdash u_1, u_2 : A}{\Gamma \vdash u_1 \approx_\eta u_2 : A}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{l} \texttt{match}\ t\ \texttt{with} \\ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array} \end{array}}$$

$$\frac{\Gamma \vdash t : 0 \qquad \Gamma, y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

Derived rules :

$$\frac{}{\Gamma \vdash t_1 \approx_\eta t_2 : 1} \qquad \frac{\Gamma \vdash t : 0 \qquad \Gamma \vdash u_1, u_2 : A}{\Gamma \vdash u_1 \approx_\eta u_2 : A}$$

Simply-typed $\beta\eta$-equivalence; Why is it difficult?

$$(\lambda x.\, t)\ u \ \triangleright_\beta\ t[u/x] \qquad \pi_i\ (t_1, t_2) \ \triangleright_\beta\ t_i$$

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array} \quad \triangleright_\beta\ u_i[t/x_i]$$

$$\frac{\Gamma \vdash t : A \to B}{t \triangleright_\eta \lambda x.\, (t\ x)} \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{t \triangleright_\eta (\pi_1\ t, \pi_2\ t)} \qquad \frac{\Gamma \vdash t : 1}{t \triangleright_\eta ()}$$

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{l} \texttt{match}\ t\ \texttt{with} \\ \Big|\ \begin{array}{l} \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array} \end{array}}$$

$$\frac{\Gamma \vdash t : 0 \qquad \Gamma, y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

Derived rules :

$$\frac{}{\Gamma \vdash t_1 \approx_\eta t_2 : 1} \qquad \frac{\Gamma \vdash t : 0 \qquad \Gamma \vdash u_1, u_2 : A}{\Gamma \vdash u_1 \approx_\eta u_2 : A}$$

$\beta\eta$ vs. contextual (approx)

$\beta\eta$-equivalence is a **syntactic** equivalence: inference judgments that rewrite term fragments.
contextual equivalence is more **semantic**: based on interaction with the outside.

They coincide in simple type systems (no polymorphism)

$\beta\eta$ vs. contextual (approx)

$\beta\eta$-equivalence is a **syntactic** equivalence: inference judgments that rewrite term fragments.
contextual equivalence is more **semantic**: based on interaction with the outside.

They coincide in simple type systems (no polymorphism)

Note: two terms t, u at judgment $\Gamma \vdash A$ with some type variables $\alpha, \beta \ldots$ are semantically equal if they are contextually equal for **any** instantiation of the $\alpha, \beta \ldots$ by closed types. Consider:

$$x : \alpha, y : \alpha \vdash x \overset{?}{\approx}_{\mathtt{ctx}} y : \alpha$$

Existing approaches (1): rewriting

Neil Ghani; Sam Lindley

Move case-splits
$$\begin{array}{l}\texttt{match}\ t\ \texttt{with}\\ \mid\ \sigma_1\ x \to \ldots\\ \mid\ \sigma_2\ x \to \ldots\end{array}$$
up in the term.

$$t \left(\begin{array}{l}\texttt{match}\ u\ \texttt{with}\\ \mid\ \sigma_1\ x \to r_1\\ \mid\ \sigma_2\ x \to r_2\end{array}\right) \quad \triangleright \quad \left(\begin{array}{l}\texttt{match}\ u\ \texttt{with}\\ \mid\ \sigma_1\ x \to t\ r_1\\ \mid\ \sigma_2\ x \to t\ r_2\end{array}\right)$$

(unlock redexes)
Cases are blocked by scoping: $\lambda x.\texttt{match}\ x\ \texttt{with}\ \ldots$
Then permute/merge/delete cases locally.

Existing approaches (2): delimited control

Vincent Balat, Roberti Di Cosmo, Marcelo Fiore

Idea: when we encounter a sub-term of positive type, we **go back in time**.

$$\lambda x. \ldots (\lambda\ y. \ldots (\lambda z. \ldots \texttt{match}\ x\ y\ \texttt{with}\ \ldots))$$

Existing approaches (2): delimited control

Vincent Balat, Roberti Di Cosmo, Marcelo Fiore

Idea: when we encounter a sub-term of positive type, we **go back in time**.

$$\lambda x. \ldots (\lambda y. \ldots (\lambda z. \ldots \texttt{match}\ x\ y\ \texttt{with}\ \ldots))$$

Existing approaches (2): delimited control

Vincent Balat, Roberti Di Cosmo, Marcelo Fiore

Idea: when we encounter a sub-term of positive type, we **go back in time**.

$$\lambda x.\ldots.(\lambda\ y.\ldots.(\lambda z.\ldots.\texttt{match}\ x\ y\ \texttt{with}\ \ldots))$$

as if we had be given

$$\lambda x.\ldots.(\lambda y.\texttt{match}\ x\ y\ \texttt{with}\ \left|\begin{array}{l}\sigma_1\ x' \to \ldots(\lambda z.\ldots.\texttt{match}\ \sigma_1\ x'\ \texttt{with}\ \ldots)\\ \sigma_2\ x' \to \ldots(\lambda z.\ldots.\texttt{match}\ \sigma_2\ x'\ \texttt{with}\ \ldots)\end{array}\right.)$$

Existing approaches (3): categorical magic

Thorsten Altenkirch, Peter Dybjer, Martin Hofmann, Philip Scott

Normalization by Evaluation... in a very carefully defined semantic domains (sheaves).

(What does it actually do on terms?)

(NbE on terms have a nice explanation for just booleans in Thorsten Altenkirch, Tarmo Utsalu. See then Arbob Ahmad, Daniel Licata, Robert Harper.)

Section 2

High-level view

Question

What is a **canonical form** for equivalence of simply-typed terms?

Redundancy: two (syntactically) distinct terms that are equivalent.

Canonical representation: a syntax of programs with no redundancy:

$$(\not\approx_{\mathtt{stx}}) \implies (\not\approx_{\mathtt{ctx}})$$

Question

What is a **canonical form** for equivalence of simply-typed terms?

Redundancy: two (syntactically) distinct terms that are equivalent.

Canonical representation: a syntax of programs with no redundancy:

$$(\not\approx_{\mathtt{stx}}) \implies (\not\approx_{\mathtt{ctx}})$$

With only functions and pairs, there is a reasonable notion of β-short η-long normal form.

Question

What is a **canonical form** for equivalence of simply-typed terms?

Redundancy: two (syntactically) distinct terms that are equivalent.

Canonical representation: a syntax of programs with no redundancy:

$$(\not\approx_{\mathtt{stx}}) \implies (\not\approx_{\mathtt{ctx}})$$

With only functions and pairs, there is a reasonable notion of β-short η-long normal form. It does not scale to sums.

Question

What is a **canonical form** for equivalence of simply-typed terms?

Redundancy: two (syntactically) distinct terms that are equivalent.

Canonical representation: a syntax of programs with no redundancy:

$$(\not\approx_{\mathtt{stx}}) \implies (\not\approx_{\mathtt{ctx}})$$

With only functions and pairs, there is a reasonable notion of β-short η-long normal form. It does not scale to sums.

Normal form (for reduction) $\neq$ Canonical form (for equivalence)

(see also Watkins, Cervesato, Pfenning, Walker, 2002)

Idea

Curry-Howard, again: programs as proofs.

The structure of

canonical forms

corresponds to the structure of

proof **search**

Restricting the search space restricts expression redundancy.

Proof search: Focusing

(existing work)

Gives a term representation ($\vdash_{\mathtt{foc}}$).

Canonical for **effectful** programs.
(Noam Zeilberger's thesis, 2009)
Not canonical for pure programs (stronger equivalences).

Complete: any term can be focused.

$$\Gamma \vdash A \qquad \Longrightarrow \qquad \Gamma \vdash_{\mathtt{foc}} A$$

Proof search: Focusing

(existing work)

Gives a term representation ($\vdash_{\mathtt{foc}}$).

Canonical for **effectful** programs.
(Noam Zeilberger's thesis, 2009)
Not canonical for pure programs (stronger equivalences).

Complete: any term can be focused.

$$\Gamma \vdash A \qquad \Longrightarrow \qquad \Gamma \vdash_{\mathtt{foc}} A$$

$$\Gamma \vdash t : A \qquad \Longrightarrow \qquad \exists v \approx_{\beta\eta} t, \quad \Gamma \vdash_{\mathtt{foc}} v : A$$

Proof search: Focusing

(existing work)

Gives a term representation ($\vdash_{\mathtt{foc}}$).

Canonical for **effectful** programs.
(Noam Zeilberger's thesis, 2009)
Not canonical for pure programs (stronger equivalences).

Complete: any term can be focused.

$$\Gamma \vdash A \qquad \Longrightarrow \qquad \Gamma \vdash_{\mathtt{foc}} A$$

$$\Gamma \vdash t : A \qquad \Longrightarrow \qquad \exists v \approx_{\beta\eta} t, \quad \Gamma \vdash_{\mathtt{foc}} v : A$$

Remark: $(\approx_{\beta\eta}) \subseteq (\approx_{\mathtt{ctx}})$

Proof search: Saturation

(my contribution, 2015).

Family of representations ($\vdash_{\mathtt{sat}:\Phi}$).

Canonical for **pure** programs (this work).

Locally complete: for any finite set of terms,
there is a Φ such that ($\vdash_{\mathtt{sat}:\Phi}$) is complete.

Extends to the empty type (this work).

Equivalence algorithm

$$\Gamma \vdash t_1, t_2 : A$$

Equivalence algorithm

Completeness of focusing:

$$\Gamma \vdash t_1, t_2 : A \qquad \rightsquigarrow$$

Equivalence algorithm

Completeness of focusing:

$$\Gamma \vdash t_1, t_2 : A \qquad \rightsquigarrow \qquad \begin{array}{c} \Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \\ v_i \approx_{\beta\eta} t_i \end{array}$$

Equivalence algorithm

Completeness of focusing:

$$\Gamma \vdash t_1, t_2 : A \quad \overset{\Leftrightarrow}{\rightsquigarrow} \quad \Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \quad v_i \approx_{\beta\eta} t_i$$

Equivalence algorithm

Completeness of focusing:

$$\Gamma \vdash t_1, t_2 : A \quad \leftrightsquigarrow \quad \Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \quad v_i \approx_{\beta\eta} t_i$$

Local completeness of saturation: pick Φ complete for $\{v_1, v_2\}$,

$$\Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A$$

Equivalence algorithm

Completeness of focusing:

$$\Gamma \vdash t_1, t_2 : A \quad \begin{matrix} \Leftrightarrow \\ \rightsquigarrow \end{matrix} \quad \begin{matrix} \Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \\ v_i \approx_{\beta\eta} t_i \end{matrix}$$

Local completeness of saturation: pick Φ complete for $\{v_1, v_2\}$,

$$\Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \quad \rightsquigarrow \quad \begin{matrix} \Gamma \vdash_{\mathtt{sat}:\Phi} w_1, w_2 : A \\ w_i \approx_{\beta\eta} v_i \end{matrix}$$

Equivalence algorithm

Completeness of focusing:

$$\Gamma \vdash t_1, t_2 : A \quad \Leftrightsquigarrow \quad \Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \quad v_i \approx_{\beta\eta} t_i$$

Local completeness of saturation: pick Φ complete for $\{v_1, v_2\}$,

$$\Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \quad \Leftrightsquigarrow \quad \Gamma \vdash_{\mathtt{sat}:\Phi} w_1, w_2 : A \quad w_i \approx_{\beta\eta} v_i$$

Equivalence algorithm

Completeness of focusing:

$$\Gamma \vdash t_1, t_2 : A \qquad \leftrightsquigarrow \qquad \begin{array}{l} \Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \\ v_i \approx_{\beta\eta} t_i \end{array}$$

Local completeness of saturation: pick Φ complete for $\{v_1, v_2\}$,

$$\Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \qquad \leftrightsquigarrow \qquad \begin{array}{l} \Gamma \vdash_{\mathtt{sat}:\Phi} w_1, w_2 : A \\ w_i \approx_{\beta\eta} v_i \end{array}$$

Canonicity of saturated focused forms:
Check syntactic equality of w_1, w_2:

$$w_1 \not\approx_{\mathtt{stx}} w_2 \qquad \Longrightarrow \qquad w_1 \not\approx_{\mathtt{ctx}} w_2$$

Equivalence algorithm

Completeness of focusing:

$$\Gamma \vdash t_1, t_2 : A \qquad \leftrightsquigarrow \qquad \begin{array}{l} \Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \\ v_i \approx_{\beta\eta} t_i \end{array}$$

Local completeness of saturation: pick Φ complete for $\{v_1, v_2\}$,

$$\Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \qquad \leftrightsquigarrow \qquad \begin{array}{l} \Gamma \vdash_{\mathtt{sat}:\Phi} w_1, w_2 : A \\ w_i \approx_{\beta\eta} v_i \end{array}$$

Canonicity of saturated focused forms:
Check syntactic equality of w_1, w_2:

$$w_1 \not\approx_{\mathtt{stx}} w_2 \qquad \Longrightarrow \qquad w_1 \not\approx_{\mathtt{ctx}} w_2$$

Corollary: $(\not\approx_{\beta\eta}) \subseteq (\not\approx_{\mathtt{ctx}})$,

Equivalence algorithm

Completeness of focusing:

$$\Gamma \vdash t_1, t_2 : A \qquad \leftrightsquigarrow \qquad \Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \quad v_i \approx_{\beta\eta} t_i$$

Local completeness of saturation: pick Φ complete for $\{v_1, v_2\}$,

$$\Gamma \vdash_{\mathtt{foc}} v_1, v_2 : A \qquad \leftrightsquigarrow \qquad \Gamma \vdash_{\mathtt{sat}:\Phi} w_1, w_2 : A \quad w_i \approx_{\beta\eta} v_i$$

Canonicity of saturated focused forms:
Check syntactic equality of w_1, w_2:

$$w_1 \not\approx_{\mathtt{stx}} w_2 \qquad \Longrightarrow \qquad w_1 \not\approx_{\mathtt{ctx}} w_2$$

Corollary: $(\not\approx_{\beta\eta}) \subseteq (\not\approx_{\mathtt{ctx}})$, so $(\approx_{\beta\eta})$ and $(\approx_{\mathtt{ctx}})$ coincide.

Section 3

Focusing

Unique inhabitant? Enumerate canonical representations

We turn to logic: Logicians use canonicity to reduce the search space

$$\frac{\Gamma \vdash \underline{A} \qquad \Gamma, \underline{B} \vdash C}{\Gamma, \underline{A \to B} \vdash C}\ -$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$$

$$\frac{\Gamma, \underline{A_i} \vdash C}{\Gamma, \underline{A_1 \times A_2} \vdash C}\ -$$

$$\frac{\Gamma \vdash A_1 \qquad \Gamma \vdash A_2}{\Gamma \vdash A_1 \times A_2}$$

$$\frac{\Gamma, A_1 \vdash C \qquad \Gamma, A_2 \vdash C}{\Gamma, A_1 + A_2 \vdash C}$$

$$\frac{\Gamma \vdash \underline{A_i}}{\Gamma \vdash \underline{A_1 + A_2}}\ +$$

$$\frac{}{\Gamma, 0 \vdash C}\ +$$

$$\frac{}{\Gamma \vdash 1}\ -$$

Invertible vs. non-invertible rules. Positives vs. negatives.

$$\frac{\Gamma \vdash \underline{A} \qquad \Gamma, \underline{B} \vdash C}{\Gamma, \underline{A \to B} \vdash C}\ -$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$$

$$\frac{\Gamma, \underline{A_i} \vdash C}{\Gamma, \underline{A_1 \times A_2} \vdash C}\ -$$

$$\frac{\Gamma \vdash A_1 \qquad \Gamma \vdash A_2}{\Gamma \vdash A_1 \times A_2}$$

$$\frac{\Gamma, A_1 \vdash C \qquad \Gamma, A_2 \vdash C}{\Gamma, A_1 + A_2 \vdash C}$$

$$\frac{\Gamma \vdash \underline{A_i}}{\Gamma \vdash \underline{A_1 + A_2}}\ +$$

$$\frac{}{\Gamma, 0 \vdash C}\ +$$

$$\frac{}{\Gamma \vdash 1}\ -$$

Invertible vs. non-invertible rules. Positives vs. negatives.

$$N, M ::= A \to B \mid A \times B \mid 1 \qquad P, Q ::= A + B \mid 0$$

$$A, B ::= P \mid N \mid \alpha \qquad P_{\mathsf{a}}, Q_{\mathsf{a}} ::= P \mid \alpha \qquad N_{\mathsf{a}}, M_{\mathsf{a}} ::= N \mid \alpha$$

Invertible phase

$$\dfrac{\dfrac{?}{\alpha + \beta \vdash \alpha}}{\alpha + \beta \vdash \beta + \alpha}$$

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

$$\dfrac{\dfrac{?}{\alpha + \beta \vdash \alpha}}{\alpha + \beta \vdash \beta + \alpha}$$

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.

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

After all invertible rules, negative context Γ_{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.

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:

$$\dfrac{\dfrac{\dfrac{\alpha_2, \quad \beta_1 \vdash A}{\alpha_2 \times \alpha_3, \quad \beta_1 \vdash A}}{\alpha_2 \times \alpha_3, \quad \beta_1 \times \beta_2 \vdash A}}{\alpha_1 \times \alpha_2 \times \alpha_3, \beta_1 \times \beta_2 \vdash A}$$

This was focusing:

- invertible as long as a rule matches, until $\Gamma_{\mathsf{na}} \vdash P_{\mathsf{a}}$
- then pick a formula
- then non-invertible as long as a rule matches, until polarity change

Completeness:

$$\Gamma \vdash A \qquad \Longrightarrow \qquad \Gamma \vdash_{\mathtt{foc}} A$$

Section 4

Focused λ-calculus

β-normal forms (negative)

β-short normal forms:

$$\pi_1\ (t, u) = t$$

$$\begin{array}{rcl} v, w & ::= & \lambda x.\, v \mid (v, w) \mid n \\ n, m & ::= & \pi_i\ n \mid n\ v \mid x \end{array}$$

β-normal forms (negative)

β-short normal forms:

$$\pi_1\ (t, u) = t$$

$$\begin{array}{rcl} v, w & ::= & \lambda x.\, v \mid (v, w) \mid n \\ n, m & ::= & \pi_i\ n \mid n\ v \mid x \end{array}$$

β-short η-long:

$$(y : \alpha \to \beta) = \lambda x : \alpha.\, (y\ x : \beta)$$

β-normal forms (negative)

β-short normal forms:

$$\pi_1\ (t, u) = t$$

$$\begin{array}{rcl} v, w & ::= & \lambda x.\, v \mid (v, w) \mid n \\ n, m & ::= & \pi_i\ n \mid n\ v \mid x \end{array}$$

β-short η-long:

$$(y : \alpha \to \beta) = \lambda x : \alpha.\, (y\ x : \beta)$$

$$\begin{array}{rcl} v, w & ::= & \lambda x.\, v \mid (v, w) \mid (n : \alpha) \\ n, m & ::= & \pi_i\ n \mid n\ v \mid x \end{array}$$

What about sums?

$$
\begin{array}{rcl}
v, w & ::= & \lambda x.\, v \mid (v, w) \mid \sigma_i\ v \mid (n : \alpha) \\
n, m & ::= & \pi_i\ n \mid n\ v \mid \left(\texttt{match}\ n\ \texttt{with} \;\middle|\; \begin{array}{l} \sigma_1\ y_1 \to v_1 \\ \sigma_2\ y_2 \to v_2 \end{array} \right) \mid x
\end{array}
$$

Does not work:

$$
\left(\begin{array}{l} \texttt{match}\ n\ \texttt{with} \\ \mid\ \sigma_1\ y_1 \to \lambda z.\, v_1 \\ \mid\ \sigma_2\ y_2 \to \lambda z.\, v_2 \end{array} \right) v
$$

$$
\begin{array}{l} \texttt{match}\ n\ \texttt{with} \\ \mid\ \sigma_1\ x \to \sigma_2\ x \\ \mid\ \sigma_2\ x \to \sigma_1\ x \end{array}
$$

Focusing to the rescue

$$
\begin{array}{rcl}
v, w & ::= & \lambda x.\, v \mid (v, w) \mid (n : \alpha) \\
n, m & ::= & \pi_i\ n \mid n\ v \mid x
\end{array}
$$

$$
\begin{array}{rcl}
v, w & ::= & \lambda x.\, v \mid (v, w) \mid () \\
 & & \mid \mathtt{absurd}(x) \mid \left(\mathtt{match}\ x\ \mathtt{with} \;\middle|\; \begin{array}{l} \sigma_1\ y_1 \to v_1 \\ \sigma_2\ y_2 \to v_2 \end{array} \right) \\
 & & \mid (\Gamma_{\mathsf{na}} \vdash f : P_{\mathsf{a}}) \\
n, m & ::= & \pi_i\ n \mid n\ p \mid x \\
p, q & ::= & \sigma_i\ p \mid (v : N_{\mathsf{a}}) \\
f & ::= & (n : \alpha) \mid (p : P) \mid \mathtt{let}\ x = (n : P)\ \mathtt{in}\ v
\end{array}
$$

Focusing to the rescue

$$
\begin{array}{rcl}
v, w & ::= & \lambda x.\, v \mid (v, w) \mid (n : \alpha) \\
n, m & ::= & \pi_i\ n \mid n\ v \mid x
\end{array}
$$

$$
\begin{array}{rcl}
v, w & ::= & \lambda x.\, v \mid (v, w) \mid () \\
 & & \mid \mathtt{absurd}(x) \mid \left(\mathtt{match}\ x\ \mathtt{with} \;\middle|\; \begin{array}{l} \sigma_1\ y_1 \to v_1 \\ \sigma_2\ y_2 \to v_2 \end{array} \right) \\
 & & \mid (\Gamma_{\mathsf{na}} \vdash f : P_{\mathsf{a}}) \\
n, m & ::= & \pi_i\ n \mid n\ p \mid x \\
p, q & ::= & \sigma_i\ p \mid (v : N_{\mathsf{a}}) \\
f & ::= & (n : \alpha) \mid (p : P) \mid \mathtt{let}\ x = (n : P)\ \mathtt{in}\ v
\end{array}
$$

Remark: "broken neutrals" are gone

$$
\pi_1 \left(\mathtt{match}\ x\ \mathtt{with} \;\middle|\; \begin{array}{l} \sigma_1\ y \to n_1 \\ \sigma_2\ y \to n_2 \end{array} \right)
$$

Completeness of focusing

Logic:

$$\Gamma \vdash A \qquad \Longrightarrow \qquad \Gamma \vdash_{\mathtt{foc}} A$$

Completeness of focusing

Logic:

$$\Gamma \vdash A \qquad \Longrightarrow \qquad \Gamma \vdash_{\mathtt{foc}} A$$

Programming:

$$\Gamma \vdash t : A \qquad \Longrightarrow \qquad \exists v, \begin{array}{c} \Gamma \vdash_{\mathtt{foc}} v : A \\ v \approx_{\beta\eta} t \end{array}$$

Canonicity

Focused normal forms are canonical for the impure λ-calculus.

Proof in Noam Zeilberger's thesis (2009), using ideas from ludics.

Section 5

Saturation

Nope !

Focusing is still not canonical – for pure languages.

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

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

Nope !

Focusing is still not canonical – for pure languages.

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

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

We want the $\texttt{let}\ x = n$ to be "as early as possible" – maximal multi-focusing. "Split neutrals early".

Nope !

Focusing is still not canonical – for pure languages.

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

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

We want the $\texttt{let}\ x = n$ to be "as early as possible" – maximal multi-focusing. "Split neutrals early".

Is $v \approx_{\beta\eta} w$? Pull the let-bindings to the roots and compare. Works for sums.

PhD topic strikes back

I wanted to **enumerate** the canonical inhabitants at a given type.

No existing term to start with.

Saturation: split on **all** possible neutrals.

Saturated focused λ-terms

$$
\begin{array}{rcl}
v, w & ::= & \lambda x.\, v \mid (v, w) \mid () \\
& & \mid () \mid \mathtt{absurd}(x) \mid \left(\mathtt{match}\ x\ \mathtt{with} \;\middle|\; \begin{array}{l} \sigma_1\ y_1 \to v_1 \\ \sigma_2\ y_2 \to v_2 \end{array} \right) \\
& & \mid (\Gamma_{\mathsf{na}} \vdash f : P_{\mathsf{a}}) \\
n, m & ::= & \pi_i\ n \mid n\ p \mid x \\
p, q & ::= & \sigma_i\ p \mid (v : N_{\mathsf{a}}) \\
f & ::= & \mathtt{let}\ \bar{x} = \bar{n}\ \mathtt{in}\ v \mid (n : \alpha) \mid (p : P)
\end{array}
$$

Plus side-condition on the $\mathtt{let}\ \bar{x} = \bar{n}$:

- they are a set (no duplicates)
- **freshness**: must use a variable of the preceding invertible phase v
- **saturation**: $n \mid p$ can only be chosen if no fresh variable

(See the paper for their (simple) formal expression.)

Selection

Which $\bar{n}$ to split in a given context Γ?

"All of them" $\implies$ infinite set $(x : \mathbb{N} \to P \vdash \dots)$

Parameter: a **selection function** $\Phi(\Gamma)$ returning the (finite) $\bar{n}$.

(For unicity: "at most two at each type")

Local completeness:

Selection

Which $\bar{n}$ to split in a given context Γ?

"All of them" $\implies$ infinite set $(x : \mathbb{N} \to P \vdash \dots)$

Parameter: a **selection function** $\Phi(\Gamma)$ returning the (finite) $\bar{n}$.

(For unicity: "at most two at each type")

Local completeness:

$$(\Gamma \vdash_{\mathtt{foc}} v : A) \qquad \Longrightarrow \qquad \exists \Phi, v', \quad \begin{array}{c} \Gamma \vdash_{\mathtt{sat}:\Phi} v' : A \\ v \approx_{\beta\eta} v' \end{array}$$

Idea: $\Phi(\Gamma) \supseteq \{\Gamma \vdash_{\mathtt{foc}} n : P \mid n \in v\}$ suffices.
$\Phi \cup \Phi' \implies$ complete for finite sets of terms.

Empty type?

$$f : 1 \to \beta, g : \beta \to 0, x : \alpha, y : \alpha \vdash ? : \alpha$$

x, y would be bad saturated terms.

Empty type?

$$f : 1 \to \beta, g : \beta \to 0, x : \alpha, y : \alpha \vdash ? : \alpha$$

x, y would be bad saturated terms.

Additional condition on Φ:

$$(\exists n,\ \Gamma \vdash_{\mathtt{foc}} n : P) \qquad \Longrightarrow \qquad (\exists n \in \Phi(\Gamma),\ \Gamma \vdash_{\mathtt{foc}} n : P)$$

Idea: set of those P is finite – subformula property.
Idea: complete for provability.

Canonicity

$$\Gamma \vdash_{\mathtt{sat}:\Phi} v, w : A \quad v \not\approx_{\alpha} w \quad \Longrightarrow \quad v \not\approx_{\mathtt{ctx}} w$$

(The hard part.)

Canonicity

$$\Gamma \vdash_{\mathtt{sat}:\Phi} v, w : A \quad v \not\approx_{\alpha} w \qquad \Longrightarrow \qquad v \not\approx_{\mathtt{ctx}} w$$

(The hard part.)

Corollary: $(\approx_{\beta\eta}) = (\approx_{\mathtt{ctx}})$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$\begin{array}{lll} & ? & \\ n : (1 + \alpha) \to \alpha \vdash & \not\approx_{\mathtt{stx}} & : \alpha \\ & ? & \end{array}$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$\begin{array}{lll} & ? & \\ n : (1 + \alpha) \to \alpha \vdash & \not\approx_{\mathtt{stx}} & : \alpha \\ & ? & \end{array}$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l}\texttt{let } z = n\,(\sigma_1\,())\ \texttt{in}\\ \quad ?\end{array} \not\approx_{\texttt{stx}} \begin{array}{l}\texttt{let } z = n\,(\sigma_1\,())\ \texttt{in}\\ \quad ?\end{array} : \alpha$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l}\texttt{let } z = n\,(\sigma_1\,())\ \texttt{in}\\ \quad ?\end{array} \not\approx_{\texttt{stx}} \begin{array}{l}\texttt{let } z = n\,(\sigma_1\,())\ \texttt{in}\\ \quad ?\end{array} : \alpha$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l}\texttt{let } z = n\,(\sigma_1\,())\ \texttt{in}\\ \quad ?\end{array} \not\approx_{\texttt{stx}} \begin{array}{l}\texttt{let } z = n\,(\sigma_1\,())\ \texttt{in}\\ \quad ?\end{array} : \alpha$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ ? \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ ? \end{array} : \alpha$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ ? \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ ? \end{array} : \alpha$$

Shared context.

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ ? \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ ? \end{array} : \alpha$$

Shared context. Source of inequality:

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \\ \qquad \not\approx_{\texttt{stx}} \\ \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.
Type variables:

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.
Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.
Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Here,

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.
Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.
Here, $\hat{\alpha} \stackrel{\text{def}}{=} 1 + 1$, $\hat{z} \stackrel{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Here, $\hat{\alpha} \stackrel{\text{def}}{=} 1 + 1$, $\hat{z} \stackrel{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Separating context: $C\,[\Box] \stackrel{\text{def}}{=} (\lambda n.\,\Box)\,\hat{n}$

Canonicity: example

$$n : (1+\alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1+\alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Here, $\hat{\alpha} \stackrel{\text{def}}{=} 1 + 1$, $\hat{z} \stackrel{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Separating context: $C\,[\Box] \stackrel{\text{def}}{=} (\lambda n.\,\Box)\ \hat{n}$

$$\hat{n} \stackrel{\text{def}}{=} \left\{ \right.$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \\ \qquad \not\approx_{\texttt{stx}} \\ \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Here, $\hat{\alpha} \stackrel{\text{def}}{=} 1 + 1$, $\hat{z} \stackrel{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Separating context: $C\,[\Box] \stackrel{\text{def}}{=} (\lambda n.\,\Box)\ \hat{n}$

$$\hat{n} \stackrel{\text{def}}{=} \left\{ \right.$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \\ \qquad \not\approx_{\texttt{stx}} \\ \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Here, $\hat{\alpha} \stackrel{\text{def}}{=} 1 + 1$, $\hat{z} \stackrel{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Separating context: $C\,[\Box] \stackrel{\text{def}}{=} (\lambda n.\,\Box)\ \hat{n}$

$$\hat{n} \stackrel{\text{def}}{=} \left\{ \begin{array}{l} \sigma_1\,() \quad \mapsto \\ \end{array} \right.$$

Canonicity: example

$$n : (1+\alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1+\alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \\ \qquad \not\approx_{\texttt{stx}} \\ \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Here, $\hat{\alpha} \stackrel{\text{def}}{=} 1 + 1$, $\hat{z} \stackrel{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Separating context: $C\,[\Box] \stackrel{\text{def}}{=} (\lambda n.\,\Box)\,\hat{n}$

$$\hat{n} \stackrel{\text{def}}{=} \left\{ \begin{array}{l} \sigma_1\,() \quad \mapsto \\ \end{array} \right.$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Here, $\hat{\alpha} \stackrel{\text{def}}{=} 1 + 1$, $\hat{z} \stackrel{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Separating context: $C\,[\Box] \stackrel{\text{def}}{=} (\lambda n.\,\Box)\ \hat{n}$

$$\hat{n} \stackrel{\text{def}}{=} \left\{ \begin{array}{lcl} \sigma_1\,() & \mapsto & \hat{z} \\ \end{array} \right.$$

Canonicity: example

$$n : (1 + \alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1 + \alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \\ \qquad \not\approx_{\texttt{stx}} \\ \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Here, $\hat{\alpha} \stackrel{\text{def}}{=} 1 + 1$, $\hat{z} \stackrel{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Separating context: $C\,[\Box] \stackrel{\text{def}}{=} (\lambda n.\,\Box)\ \hat{n}$

$$\hat{n} \stackrel{\text{def}}{=} \left\{ \begin{array}{lcl} \sigma_1\,() & \mapsto & \hat{z} \\ \end{array} \right.$$

Canonicity: example

$$n : (1+\alpha) \to \alpha \vdash n\,(\sigma_1\,()),\ n\,(\sigma_2\ \ (n\,\sigma_1\,())\ ) : \alpha$$

Saturated forms:

$$n : (1+\alpha) \to \alpha \vdash \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ z \end{array} \not\approx_{\texttt{stx}} \begin{array}{l} \texttt{let}\ z = n\,(\sigma_1\,())\ \texttt{in} \\ \quad \texttt{let}\ o = n\,(\sigma_2\ z)\ \texttt{in}\ o \end{array} : \alpha$$

Shared context. Source of inequality: $z \not\approx_{\texttt{stx}} o$.

Type variables: pick a finite type of codes of the form $1 + (1 + \ldots)$.

Here, $\hat{\alpha} \stackrel{\text{def}}{=} 1 + 1$, $\hat{z} \stackrel{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Separating context: $C\,[\Box] \stackrel{\text{def}}{=} (\lambda n.\,\Box)\,\hat{n}$

$$\hat{n} \stackrel{\text{def}}{=} \begin{cases} \sigma_1\,() & \mapsto \quad \hat{z} \\ \sigma_2\,\hat{z} & \mapsto \quad \hat{o} \end{cases}$$

Thanks

Questions?

negative types $N, M ::= \alpha^-, \beta^-, \gamma^- \mid P \to N \mid N_1 \times N_2 \mid 1 \mid \langle P \rangle^-$

positive types $P, Q ::= \alpha^+, \beta^+, \gamma^+ \mid P_1 + P_2 \mid 0 \mid \langle N \rangle^+$

$$P_a, Q_a ::= P, Q \mid \alpha^-, \beta^- \quad N_a, M_a ::= N, M \mid \alpha^+, \beta^+$$

invertible terms $t, u, r ::= \lambda x.\, t \mid () \mid (t_1, t_2) \mid (f : P)$
$\mid \mathtt{absurd}(x) \mid \mathtt{match}\ x\ \mathtt{with}\ (\sigma_i\ x \to t_i)^i$

focusing terms $f, g ::= \mathtt{let}\ (x : P) = n\ \mathtt{in}\ t \mid (n : \alpha^-) \mid p$

negative neutrals $n, m ::= (x : N) \mid n\ p \mid \pi_i\ n$

positive neutrals $p, q ::= \sigma_i\ p \mid (x : \alpha^+)$

shift-or-atom notations

$$\langle N \rangle_a^+ \stackrel{\text{def}}{=} \langle N \rangle^+ \quad \langle \alpha^+ \rangle_a^+ \stackrel{\text{def}}{=} \alpha^+$$
$$\langle P \rangle_a^- \stackrel{\text{def}}{=} \langle P \rangle^- \quad \langle \alpha^- \rangle_a^- \stackrel{\text{def}}{=} \alpha^-$$

$$\frac{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}}, x : P \vdash_{\mathrm{inv}} t : N \mid \emptyset}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}} \vdash_{\mathrm{inv}} \lambda x.\, t : P \to N \mid \emptyset} \qquad \frac{(\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}} \vdash_{\mathrm{inv}} t_i : N_i \mid \emptyset)^i}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}} \vdash_{\mathrm{inv}} (t_1, t_2) : N_1 \times N_2 \mid \emptyset}$$

$$\frac{(\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}}, x : Q_i \vdash_{\mathrm{inv}} t_i : N \mid P_{\mathrm{a}})^i}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}}, x : Q_1 + Q_2 \vdash_{\mathrm{inv}} \mathtt{match}\ x\ \mathtt{with}\ (\sigma_i\ x \to t_i)^i : N \mid P_{\mathrm{a}}}$$

$$\frac{}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}}, x : 0 \vdash_{\mathrm{inv}} \mathtt{absurd}(x) : N \mid P_{\mathrm{a}}} \qquad \frac{}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}} \vdash_{\mathrm{inv}} () : 1 \mid \emptyset}$$

$$\frac{\Gamma_{\mathrm{na}}, \Gamma'_{\mathrm{na}} \vdash_{\mathrm{foc}} f : (P_{\mathrm{a}} \mid Q_{\mathrm{a}})}{\Gamma_{\mathrm{na}}; \langle \Gamma'_{\mathrm{na}} \rangle^+_{\mathrm{a}} \vdash_{\mathrm{inv}} f : \langle P_{\mathrm{a}} \rangle^-_{\mathrm{a}} \mid Q_{\mathrm{a}}} \qquad \frac{\Gamma_{\mathrm{na}} \vdash p \Uparrow P}{\Gamma_{\mathrm{na}} \vdash_{\mathrm{foc}} p : P}$$

$$\frac{\Gamma_{\mathrm{na}} \vdash n \Downarrow \alpha^-}{\Gamma_{\mathrm{na}} \vdash_{\mathrm{foc}} n : \alpha^-} \qquad \frac{\Gamma_{\mathrm{na}} \vdash n \Downarrow \langle P \rangle^- \qquad \Gamma_{\mathrm{na}}; x : P \vdash_{\mathrm{inv}} t : \emptyset \mid Q_{\mathrm{a}}}{\Gamma_{\mathrm{na}} \vdash_{\mathrm{foc}} \mathtt{let}\ x = n\ \mathtt{in}\ t : Q_{\mathrm{a}}} \qquad \frac{}{\Gamma_{\mathrm{na}}, x : N \vdash x \Downarrow N}$$

$$\frac{}{\Gamma_{\mathrm{na}}, x : \alpha^+ \vdash x \Uparrow \alpha^+} \qquad \frac{\Gamma_{\mathrm{na}} \vdash n \Downarrow N_1 \times N_2}{\Gamma_{\mathrm{na}} \vdash \pi_i\ n \Downarrow N_i} \qquad \frac{\Gamma_{\mathrm{na}}; \emptyset \vdash_{\mathrm{inv}} t : N \mid \emptyset}{\Gamma_{\mathrm{na}} \vdash t \Uparrow \langle N \rangle^+}$$

$$\frac{\Gamma_{\mathrm{na}} \vdash n \Downarrow P \to N \qquad \Gamma_{\mathrm{na}} \vdash p \Uparrow P}{\Gamma_{\mathrm{na}} \vdash n\ p \Downarrow N} \qquad \frac{\Gamma_{\mathrm{na}} \vdash p \Uparrow P_i}{\Gamma_{\mathrm{na}} \vdash \sigma_i\ p \Uparrow P_1 + P_2}$$

$$\frac{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}}, x : P \vdash_{\mathrm{sinv}} t : N \mid \emptyset}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}} \vdash_{\mathrm{sinv}} \lambda x.\, t : P \to N \mid \emptyset} \qquad \frac{(\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}} \vdash_{\mathrm{sinv}} t_i : N_i \mid \emptyset)^i}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}} \vdash_{\mathrm{sinv}} (t_1, t_2) : N_1 \times N_2 \mid \emptyset}$$

$$\frac{}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}} \vdash_{\mathrm{sinv}} () : 1 \mid \emptyset} \qquad \frac{}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}}, x : 0 \vdash_{\mathrm{sinv}} \mathtt{absurd}(x) : N \mid Q_{\mathrm{a}}}$$

$$\frac{(\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}}, x : P_i \vdash_{\mathrm{sinv}} t_i : N \mid Q_{\mathrm{a}})^i}{\Gamma_{\mathrm{na}}; \Sigma_{\mathrm{p}}, x : P_1 + P_2 \vdash_{\mathrm{sinv}} \mathtt{match}\ x\ \mathtt{with}\ (\sigma_i\ x \to t_i)^i : N \mid Q_{\mathrm{a}}}$$

$$\frac{\Gamma_{\mathrm{na}}; \Gamma'_{\mathrm{na}} \vdash_{\mathrm{sat}} f : (P_{\mathrm{a}} \mid Q_{\mathrm{a}})}{\Gamma_{\mathrm{na}}; \langle \Gamma'_{\mathrm{na}} \rangle_{\mathrm{a}}^{+} \vdash_{\mathrm{sinv}} f : \langle P_{\mathrm{a}} \rangle_{\mathrm{a}}^{-} \mid Q_{\mathrm{a}}} \qquad \frac{\Gamma_{\mathrm{na}} \vdash_{\mathrm{s}} p \Uparrow P}{\Gamma_{\mathrm{na}}; \emptyset \vdash_{\mathrm{sat}} p : P} \qquad \frac{\Gamma_{\mathrm{na}} \vdash_{\mathrm{s}} n \Downarrow \alpha^{-}}{\Gamma_{\mathrm{na}}; \emptyset \vdash_{\mathrm{sat}} n : \alpha^{-}}$$

$$\frac{\begin{array}{c}(\bar{n}, \bar{P}) \stackrel{\mathrm{def}}{=} \Phi(\Gamma_{\mathrm{na}}, \Gamma'_{\mathrm{na}}) \left\{ (n, P) \mid \begin{array}{c}(\Gamma_{\mathrm{na}}, \Gamma'_{\mathrm{na}} \vdash_{\mathrm{s}} n \Downarrow \langle P \rangle^{-}) \\ \wedge\ \exists x \in \Gamma'_{\mathrm{na}}, x \in n\end{array} \right\} \\ \Gamma_{\mathrm{na}}, \Gamma'_{\mathrm{na}}; \bar{x} : \bar{P} \vdash_{\mathrm{sinv}} t : \emptyset \mid Q_{\mathrm{a}}\end{array}}{\Gamma_{\mathrm{na}}; \Gamma'_{\mathrm{na}} \vdash_{\mathrm{sat}} \mathtt{let}\ \bar{x} = \bar{n}\ \mathtt{in}\ t : Q_{\mathrm{a}}}$$

$(\Gamma_{\mathrm{na}} \vdash_{\mathrm{s}} n \Downarrow N), (\Gamma_{\mathrm{na}} \vdash_{\mathrm{s}} p \Uparrow P)$, as before