Deciding pure program equivalence with sums and the empty type

Gabriel Scherer

Northeastern University, Boston

January 26, 2017

Program equivalence

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 $\Lambda C(\alpha, \to)$: decidable.
Polymorphism $\Lambda C(\alpha, \to, \forall)$, dependent types $\Lambda C(\alpha, \to, \Pi)$: undecidable.

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

History

Decidability of equivalence:

- $\Lambda C(\alpha, \to)$: Tait, 1967 or earlier.
- $\Lambda C(\alpha, \to, \times)$: essentially the same proof.
- $\Lambda C(\alpha, \to, \times, 1)$: essentially the same proof.
- $\Lambda C(\alpha, \to, \times, 1, +)$: Ghani, 1995; Altenkirch, Dybjer, Hoffman, Scott: 2001; Balat, Di Cosmo, Fiore: 2004; Lindley, 2007; Ahmad, Licata, Harper, 2010 (draft).
- $\Lambda C(\alpha, \to, \times, 1, +, 0)$: ? (this work)

Why do we care?

A. It has remained an open problem for decades, despite recurrent work:

1. People still care $\implies$ it comes up in their problems.
   (Ryan Winesky, [types-list], 2013)
2. It requires a new perspective that will have other applications.

B. Deciding equivalence can teach us something deep about
the structure of programs.

C. We have a lot of practical tools to check **specifications** of program,
but not their **equivalence**.
Many applications: testing, refactoring verification, program specialization,
program synthesis...

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

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

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

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

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

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

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

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

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

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

$$\texttt{match}\ \sigma_i\ t\ \texttt{with}\ \left|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array}\right. \;\rhd_\beta\; u_i[t/x_i]$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \left|\begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array}\right. \;\rhd_\beta\; u_i[t/x_i]$$

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

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \rhd_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \left|\begin{array}{l} \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \sigma_2\ x_2 \to \sigma_2\ x_2 \end{array}\right.\end{array}}$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$\texttt{match}\ \sigma_i\ t\ \texttt{with}\ \left|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array}\right. \ \rhd_\beta\ u_i[t/x_i]$$

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

$$\frac{\Gamma \vdash t : A_1 + A_2}{t \rhd_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \left|\ \begin{array}{l} \sigma_1\ x_1 \to \sigma_1\ x_1 \\ \sigma_2\ x_2 \to \sigma_2\ x_2 \end{array}\right.\end{array}}$$

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

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

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

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

$$\texttt{match}\ \sigma_i\ t\ \texttt{with} \left|\ \begin{array}{l} \sigma_1\ x_1 \to u_1 \\ \sigma_2\ x_2 \to u_2 \end{array}\right. \ \rhd_\beta\ u_i[t/x_i]$$

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

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

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

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

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

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

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \left|\ \begin{array}{l} \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array}\right.\end{array}} \qquad \frac{\Gamma \vdash t : 0 \qquad \Gamma, y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

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

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

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

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

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{l}\texttt{match}\; t\;\texttt{with}\\ \left|\begin{array}{l}\sigma_1\; x_1 \to u[\sigma_1\; x_1/y]\\ \sigma_2\; x_2 \to u[\sigma_2\; x_2/y]\end{array}\right.\end{array}} \qquad \frac{\Gamma \vdash t : 0 \qquad \Gamma, y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

Derived rules :

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

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

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

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

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{c} \texttt{match}\; t\; \texttt{with} \\ \left|\; \begin{array}{l} \sigma_1\; x_1 \to u[\sigma_1\; x_1/y] \\ \sigma_2\; x_2 \to u[\sigma_2\; x_2/y] \end{array}\right. \end{array}} \qquad \frac{\Gamma \vdash t : 0 \qquad \Gamma, y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

Derived rules :

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

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

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

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

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

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \left|\ \begin{array}{l} \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array}\right.\end{array}} \qquad \frac{\Gamma \vdash t : 0 \qquad \Gamma, y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

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?

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

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

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

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \triangleright_\eta \begin{array}{l}\texttt{match}\ t\ \texttt{with} \\ \left|\ \begin{array}{l} \sigma_1\ x_1 \to u[\sigma_1\ x_1/y] \\ \sigma_2\ x_2 \to u[\sigma_2\ x_2/y] \end{array}\right.\end{array}} \qquad \frac{\Gamma \vdash t : 0 \qquad \Gamma, y : 0 \vdash u : C}{u[t/y] \triangleright_\eta \texttt{absurd}(t)}$$

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?

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

$$\texttt{match}\;\sigma_i\;t\;\texttt{with}\;\left|\begin{array}{l}\sigma_1\;x_1 \to u_1\\ \sigma_2\;x_2 \to u_2\end{array}\right. \;\rhd_\beta\; u_i[t/x_i]$$

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

$$\frac{\Gamma \vdash t : A_1 + A_2 \qquad \Gamma, y : A_1 + A_2 \vdash u : C}{u[t/y] \rhd_\eta \texttt{match}\;t\;\texttt{with}\;\left|\begin{array}{l}\sigma_1\;x_1 \to u[\sigma_1\;x_1/y]\\ \sigma_2\;x_2 \to u[\sigma_2\;x_2/y]\end{array}\right.} \qquad \frac{\Gamma \vdash t : 0 \qquad \Gamma, y : 0 \vdash u : C}{u[t/y] \rhd_\eta \texttt{absurd}(t)}$$

Derived rules :

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

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:

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

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:

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

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 ($\vdash_{\texttt{foc}}$).

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

Complete: any term can be focused.

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

Proof search: Focusing

(existing work)

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

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

Complete: any term can be focused.

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

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

Proof search: Focusing

(existing work)

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

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

Complete: any term can be focused.

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

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

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

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:

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

Equivalence algorithm

Completeness of focusing:

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

Equivalence algorithm

Completeness of focusing:

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

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

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

Equivalence algorithm

Completeness of focusing:

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

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

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

Equivalence algorithm

Completeness of focusing:

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

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

$$\Gamma \vdash_{\texttt{foc}} v_1, v_2 : A \qquad \overset{\Leftrightarrow}{\rightsquigarrow} \qquad \begin{array}{c} \Gamma \vdash_{\texttt{sat:}\Phi} w_1, w_2 : A \\ w_i \approx_{\beta\eta} v_i \end{array}$$

Equivalence algorithm

Completeness of focusing:

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

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

$$\Gamma \vdash_{\texttt{foc}} v_1, v_2 : A \qquad \overset{\Leftrightarrow}{\rightsquigarrow} \qquad \begin{array}{c} \Gamma \vdash_{\texttt{sat}:\Phi} w_1, w_2 : A \\ w_i \approx_{\beta\eta} v_i \end{array}$$

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

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

Equivalence algorithm

Completeness of focusing:

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

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

$$\Gamma \vdash_{\texttt{foc}} v_1, v_2 : A \quad \overset{\Leftrightarrow}{\rightsquigarrow} \quad \begin{array}{c} \Gamma \vdash_{\texttt{sat:}\Phi} w_1, w_2 : A \\ w_i \approx_{\beta\eta} v_i \end{array}$$

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

$$w_1 \not\approx_{\texttt{stx}} w_2 \quad \Longrightarrow \quad w_1 \not\approx_{\texttt{ctx}} w_2$$

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

Equivalence algorithm

Completeness of focusing:

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

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

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

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

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

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

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:

?

$$n : (1 + \alpha) \to \alpha \vdash \qquad \not\approx_{\texttt{stx}} \qquad : \alpha$$

?

Canonicity: example

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

Saturated forms:

$$?$$

$$n : (1 + \alpha) \to \alpha \vdash \qquad \not\approx_{\mathtt{stx}} \qquad : \alpha$$

$$?$$

Canonicity: example

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

Saturated forms:

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

Canonicity: example

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

Saturated forms:

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

Canonicity: example

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

Saturated forms:

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

Canonicity: example

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

Saturated forms:

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

Canonicity: example

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

Saturated forms:

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

Shared context.

Canonicity: example

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

Saturated forms:

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

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:

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

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:

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

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:

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

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

Canonicity: example

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

Saturated forms:

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

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

Canonicity: example

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

Saturated forms:

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

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,

Canonicity: example

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

Saturated forms:

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

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 $\hat{o} \stackrel{\text{def}}{=} \sigma_2\,()$.

Canonicity: example

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

Saturated forms:

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

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, $\hat{\alpha} \overset{\text{def}}{=} 1 + 1$, $\hat{z} \overset{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \overset{\text{def}}{=} \sigma_2\,()$.
Separating context: $C\,[\Box] \overset{\text{def}}{=} (\lambda n.\,\Box)\ \hat{n}$

Canonicity: example

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

Saturated forms:

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

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

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

Canonicity: example

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

Saturated forms:

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

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

$$\hat{n} \stackrel{\text{def}}{=} \Bigg\{$$

Canonicity: example

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

Saturated forms:

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

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: $C\,[\square] \stackrel{\text{def}}{=} (\lambda n.\,\square)\;\hat{n}$

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

Canonicity: example

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

Saturated forms:

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

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: $C\,[\square] \stackrel{\text{def}}{=} (\lambda n.\,\square)\;\hat{n}$

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

Canonicity: example

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

Saturated forms:

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

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, $\hat{\alpha} \overset{\text{def}}{=} 1 + 1$, $\hat{z} \overset{\text{def}}{=} \sigma_1\,()$ and $\hat{o} \overset{\text{def}}{=} \sigma_2\,()$.
Separating context: $C\,[\square] \overset{\text{def}}{=} (\lambda n.\,\square)\ \hat{n}$

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

Canonicity: example

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

Saturated forms:

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

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

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

Canonicity: example

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

Saturated forms:

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

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: $C\,[\square] \stackrel{\text{def}}{=} (\lambda n.\,\square)\;\hat{n}$

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

Thanks

Take away:

- Proof theory, proof **search** have surprising PL applications.
- Focusing is cool, learn about it!
- 0 is okay.

https://arxiv.org/abs/1610.01213

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

$$P_\mathsf{a}, Q_\mathsf{a} ::= P, Q \mid \alpha^-, \beta^- \qquad N_\mathsf{a}, M_\mathsf{a} ::= N, M \mid \alpha^+, \beta^+$$

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

$$\langle N \rangle^+_\mathsf{a} \stackrel{\text{def}}{=} \langle N \rangle^+ \qquad \langle \alpha^+ \rangle^+_\mathsf{a} \stackrel{\text{def}}{=} \alpha^+$$

$$\langle P \rangle^-_\mathsf{a} \stackrel{\text{def}}{=} \langle P \rangle^- \qquad \langle \alpha^- \rangle^-_\mathsf{a} \stackrel{\text{def}}{=} \alpha^-$$

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

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

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

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

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

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

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

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

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

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

SINV-SAT

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

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

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