
Untyped Normalization-by-Evaluation

Klaus Aehlig Felix Joachimski

Mathematisches Institut

LMU München

{aehlig,joachski}@mathematik.uni-muenchen.de

Tallinn, 17.4.4

Klaus Aehlig and Felix Joachimski. [Operational aspects of untyped normalization by evaluation.](#)

Accepted by *Mathematical Structures in Computer Science*, 2003.

<http://www.mathematik.uni-muenchen.de/~aehlig>

Motivation

$[[\cdot]]_\xi$:	Terms	\longrightarrow	Semantics	(evaluation function)
\downarrow	:	Semantics	\longrightarrow	Normal forms	(inverse)
\uparrow	:	Variables	\longrightarrow	Semantics	(valuation)

▷ **Soundness:** $r \rightarrow s \Rightarrow [[r]]_\xi = [[s]]_\xi$

▷ **Reproduction:** r in normal form $\Rightarrow \downarrow[[r]]_\uparrow = r$

Motivation

$[[\cdot]]_\xi$:	Terms	\longrightarrow	Semantics	(evaluation function)
\downarrow	:	Semantics	\longrightarrow	Normal forms	(inverse)
\uparrow	:	Variables	\longrightarrow	Semantics	(valuation)

▷ **Soundness:** $r \rightarrow s \Rightarrow [[r]]_\xi = [[s]]_\xi$

▷ **Reproduction:** r in normal form $\Rightarrow \downarrow[[r]]_\uparrow = r$

$$r =_\beta s \in \text{NF}_\beta$$

Motivation

$[[\cdot]]_\xi$:	Terms	\longrightarrow	Semantics	(evaluation function)
\downarrow	:	Semantics	\longrightarrow	Normal forms	(inverse)
\uparrow	:	Variables	\longrightarrow	Semantics	(valuation)

▷ **Soundness:** $r \rightarrow s \Rightarrow [[r]]_\xi = [[s]]_\xi$

▷ **Reproduction:** r in normal form $\Rightarrow \downarrow[[r]]_\uparrow = r$

$$r =_\beta s \in \text{NF}_\beta \implies r \rightarrow^* s$$

Motivation

$[[\cdot]]_\xi$:	Terms	\longrightarrow	Semantics	(evaluation function)
\downarrow	:	Semantics	\longrightarrow	Normal forms	(inverse)
\uparrow	:	Variables	\longrightarrow	Semantics	(valuation)

▷ **Soundness:** $r \rightarrow s \Rightarrow [[r]]_\xi = [[s]]_\xi$

▷ **Reproduction:** r in normal form $\Rightarrow \downarrow[[r]]_\uparrow = r$

$$r =_\beta s \in \text{NF}_\beta \implies r \rightarrow^* s \implies \downarrow[[r]]_\uparrow$$

Motivation

$[[\cdot]]_\xi$:	Terms	\longrightarrow	Semantics	(evaluation function)
\downarrow	:	Semantics	\longrightarrow	Normal forms	(inverse)
\uparrow	:	Variables	\longrightarrow	Semantics	(valuation)

▷ **Soundness:** $r \rightarrow s \Rightarrow [[r]]_\xi = [[s]]_\xi$

▷ **Reproduction:** r in normal form $\Rightarrow \downarrow[[r]]_\uparrow = r$

$$r =_\beta s \in \text{NF}_\beta \implies r \rightarrow^* s \implies \downarrow[[r]]_\uparrow = \downarrow[[s]]_\uparrow$$

Motivation

$[[\cdot]]_\xi$:	Terms	\longrightarrow	Semantics	(evaluation function)
\downarrow	:	Semantics	\longrightarrow	Normal forms	(inverse)
\uparrow	:	Variables	\longrightarrow	Semantics	(valuation)

▷ **Soundness:** $r \rightarrow s \Rightarrow [[r]]_\xi = [[s]]_\xi$

▷ **Reproduction:** r in normal form $\Rightarrow \downarrow[[r]]_\uparrow = r$

$$r =_\beta s \in \text{NF}_\beta \implies r \rightarrow^* s \implies \downarrow[[r]]_\uparrow = \downarrow[[s]]_\uparrow = s$$

Example — The typed λ -calculus

▷ **Terms:** Λ with simple types

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota, \quad [[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota, \quad [[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x, \quad [[rs]]_\xi := [[r]]_\xi [[s]]_\xi, \quad [[\lambda x r]]_\xi := \lambda X [[r]]_{\xi, x:=X}$

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota$, $[[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \lambda X [[r]]_{\xi, x:=X}$
- ▷ $\uparrow_\rho : \{x\vec{r} \mid \vec{r} \text{ normal}\} \rightarrow [[\rho]]$ and $\downarrow_\rho : [[\rho]] \rightarrow \text{NF}_\rho$

$$\begin{array}{ll} \uparrow_\iota r & := r, & \uparrow_{\rho \rightarrow \sigma} r & := \lambda X \uparrow_\sigma (r \downarrow_\rho X), \\ \downarrow_\iota r & := r, & \downarrow_{\rho \rightarrow \sigma} R & := \lambda x \downarrow_\sigma (R \uparrow_\rho x), & x \text{ new} \end{array}$$

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota$, $[[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \mathbb{A}X [[r]]_{\xi, x:=X}$
- ▷ $\uparrow_\rho : \{x\vec{r} \mid \vec{r} \text{ normal}\} \rightarrow [[\rho]]$ and $\downarrow_\rho : [[\rho]] \rightarrow \text{NF}_\rho$

$$\begin{array}{ll} \uparrow_\iota r & := r, & \uparrow_{\rho \rightarrow \sigma} r & := \mathbb{A}X \uparrow_\sigma (r \downarrow_\rho X), \\ \downarrow_\iota r & := r, & \downarrow_{\rho \rightarrow \sigma} R & := \lambda x \downarrow_\sigma (R \uparrow_\rho x), & x \text{ new} \end{array}$$

- ▷ **Properties:** $\downarrow_{\rho \rightarrow \sigma} \mathbb{A}X R$

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota$, $[[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \mathbb{K}X [[r]]_{\xi, x := X}$
- ▷ $\uparrow_\rho : \{x\vec{r} \mid \vec{r} \text{ normal}\} \rightarrow [[\rho]]$ and $\downarrow_\rho : [[\rho]] \rightarrow \text{NF}_\rho$

$$\begin{array}{ll} \uparrow_\iota r & := r, & \uparrow_{\rho \rightarrow \sigma} r & := \mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X), \\ \downarrow_\iota r & := r, & \downarrow_{\rho \rightarrow \sigma} R & := \lambda x \downarrow_\sigma (R \uparrow_\rho x), & x \text{ new} \end{array}$$

- ▷ **Properties:** $\downarrow_{\rho \rightarrow \sigma} \mathbb{K}X R = \lambda x \downarrow_\sigma ((\mathbb{K}X R) \uparrow_\rho x)$

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota$, $[[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda xr]]_\xi := \mathbb{K}X [[r]]_{\xi, x:=X}$
- ▷ $\uparrow_\rho : \{x\vec{r} \mid \vec{r} \text{ normal}\} \rightarrow [[\rho]]$ and $\downarrow_\rho : [[\rho]] \rightarrow \text{NF}_\rho$

$$\begin{array}{ll} \uparrow_\iota r & := r, & \uparrow_{\rho \rightarrow \sigma} r & := \mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X), \\ \downarrow_\iota r & := r, & \downarrow_{\rho \rightarrow \sigma} R & := \lambda x \downarrow_\sigma (R \uparrow_\rho x), & x \text{ new} \end{array}$$

- ▷ **Properties:** $\downarrow_{\rho \rightarrow \sigma} \mathbb{K}X R = \lambda x \downarrow_\sigma ((\mathbb{K}X R) \uparrow_\rho x) = \lambda x \downarrow_\sigma (R[X := \uparrow_\rho x])$.

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota$, $[[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \mathbb{K}X [[r]]_{\xi, x := X}$
- ▷ $\uparrow_\rho : \{x\vec{r} \mid \vec{r} \text{ normal}\} \rightarrow [[\rho]]$ and $\downarrow_\rho : [[\rho]] \rightarrow \text{NF}_\rho$

$$\begin{array}{ll} \uparrow_\iota r & := r, & \uparrow_{\rho \rightarrow \sigma} r & := \mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X), \\ \downarrow_\iota r & := r, & \downarrow_{\rho \rightarrow \sigma} R & := \lambda x \downarrow_\sigma (R \uparrow_\rho x), & x \text{ new} \end{array}$$

- ▷ **Properties:** $\downarrow_{\rho \rightarrow \sigma} \mathbb{K}X R = \lambda x \downarrow_\sigma ((\mathbb{K}X R) \uparrow_\rho x) = \lambda x \downarrow_\sigma (R[X := \uparrow_\rho x]).$
 $(\uparrow_{\rho \rightarrow \sigma} r) S$

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota$, $[[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \mathbb{K}X [[r]]_{\xi, x := X}$
- ▷ $\uparrow_\rho : \{x\vec{r} \mid \vec{r} \text{ normal}\} \rightarrow [[\rho]]$ and $\downarrow_\rho : [[\rho]] \rightarrow \text{NF}_\rho$

$$\begin{array}{ll} \uparrow_\iota r & := r, & \uparrow_{\rho \rightarrow \sigma} r & := \mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X), \\ \downarrow_\iota r & := r, & \downarrow_{\rho \rightarrow \sigma} R & := \lambda x \downarrow_\sigma (R \uparrow_\rho x), & x \text{ new} \end{array}$$

- ▷ **Properties:** $\downarrow_{\rho \rightarrow \sigma} \mathbb{K}X R = \lambda x \downarrow_\sigma ((\mathbb{K}X R) \uparrow_\rho x) = \lambda x \downarrow_\sigma (R[X := \uparrow_\rho x]).$
 $(\uparrow_{\rho \rightarrow \sigma} r) S = (\mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X)) S$

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota$, $[[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \mathbb{K}X [[r]]_{\xi, x := X}$
- ▷ $\uparrow_\rho : \{x\vec{r} \mid \vec{r} \text{ normal}\} \rightarrow [[\rho]]$ and $\downarrow_\rho : [[\rho]] \rightarrow \text{NF}_\rho$

$$\begin{array}{ll} \uparrow_\iota r & := r, & \uparrow_{\rho \rightarrow \sigma} r & := \mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X), \\ \downarrow_\iota r & := r, & \downarrow_{\rho \rightarrow \sigma} R & := \lambda x \downarrow_\sigma (R \uparrow_\rho x), & x \text{ new} \end{array}$$

- ▷ **Properties:** $\downarrow_{\rho \rightarrow \sigma} \mathbb{K}X R = \lambda x \downarrow_\sigma ((\mathbb{K}X R) \uparrow_\rho x) = \lambda x \downarrow_\sigma (R[X := \uparrow_\rho x]).$
 $(\uparrow_{\rho \rightarrow \sigma} r) S = (\mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X)) S = \uparrow_\sigma (r \downarrow_\rho S).$

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota$, $[[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \mathbb{K}X [[r]]_{\xi, x := X}$
- ▷ $\uparrow_\rho : \{x\vec{r} \mid \vec{r} \text{ normal}\} \rightarrow [[\rho]]$ and $\downarrow_\rho : [[\rho]] \rightarrow \text{NF}_\rho$

$$\begin{array}{ll} \uparrow_\iota r & := r, & \uparrow_{\rho \rightarrow \sigma} r & := \mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X), \\ \downarrow_\iota r & := r, & \downarrow_{\rho \rightarrow \sigma} R & := \lambda x \downarrow_\sigma (R \uparrow_\rho x), & x \text{ new} \end{array}$$

- ▷ **Properties:**

$$\begin{array}{llll} \downarrow_{\rho \rightarrow \sigma} \mathbb{K}X R & = & \lambda x \downarrow_\sigma ((\mathbb{K}X R) \uparrow_\rho x) & = \lambda x \downarrow_\sigma (R[X := \uparrow_\rho x]). \\ (\uparrow_{\rho \rightarrow \sigma} r) S & = & (\mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X)) S & = \uparrow_\sigma (r \downarrow_\rho S). \\ \downarrow_\iota \uparrow_\iota r & & & \end{array}$$

Example — The typed λ -calculus

- ▷ **Terms:** Λ with simple types
- ▷ **Semantics:** $[[\iota]] := \text{NF}_\iota$, $[[\rho \rightarrow \sigma]] := [[\rho]] \rightarrow [[\sigma]]$
- ▷ **Evaluation:** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \mathbb{K}X [[r]]_{\xi, x := X}$
- ▷ $\uparrow_\rho : \{x\vec{r} \mid \vec{r} \text{ normal}\} \rightarrow [[\rho]]$ and $\downarrow_\rho : [[\rho]] \rightarrow \text{NF}_\rho$

$$\begin{array}{ll} \uparrow_\iota r & := r, & \uparrow_{\rho \rightarrow \sigma} r & := \mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X), \\ \downarrow_\iota r & := r, & \downarrow_{\rho \rightarrow \sigma} R & := \lambda x \downarrow_\sigma (R \uparrow_\rho x), & x \text{ new} \end{array}$$

- ▷ **Properties:**

$$\begin{aligned} \downarrow_{\rho \rightarrow \sigma} \mathbb{K}X R &= \lambda x \downarrow_\sigma ((\mathbb{K}X R) \uparrow_\rho x) = \lambda x \downarrow_\sigma (R[X := \uparrow_\rho x]). \\ (\uparrow_{\rho \rightarrow \sigma} r) S &= (\mathbb{K}X \uparrow_\sigma (r \downarrow_\rho X)) S = \uparrow_\sigma (r \downarrow_\rho S). \\ \downarrow_\iota \uparrow_\iota r &= r. \end{aligned}$$

From properties to axioms

▷ **Properties.**

$$\begin{aligned} \downarrow_{\rho \rightarrow \sigma} \lambda X R &= \lambda x \downarrow_{\sigma} ((\lambda X R) \uparrow_{\rho} x) &= \lambda x \downarrow_{\sigma} (R[X := \uparrow_{\rho} x]). \\ (\uparrow_{\rho \rightarrow \sigma} r) S &= (\lambda X \uparrow_{\sigma} (r \downarrow_{\rho} X)) S &= \uparrow_{\sigma} (r \downarrow_{\rho} S). \\ \downarrow_{\iota} \uparrow_{\iota} r &= r. \end{aligned}$$

From properties to axioms

▷ **Properties.**

$$\begin{aligned} \downarrow_{\rho \rightarrow \sigma} \lambda X R &= \lambda x \downarrow_{\sigma} ((\lambda X R) \uparrow_{\rho} x) &= \lambda x \downarrow_{\sigma} (R[X := \uparrow_{\rho} x]). \\ (\uparrow_{\rho \rightarrow \sigma} r) S &= (\lambda X \uparrow_{\sigma} (r \downarrow_{\rho} X)) S &= \uparrow_{\sigma} (r \downarrow_{\rho} S). \\ \downarrow_{\iota} \uparrow_{\iota} r &= r. \end{aligned}$$

Discard types — we only need to know whether function or not

From properties to axioms

▷ **Properties.**

$$\begin{aligned} \downarrow_{\rho \rightarrow \sigma} \lambda X R &= \lambda x \downarrow_{\sigma} ((\lambda X R) \uparrow_{\rho} x) &= \lambda x \downarrow_{\sigma} (R[X := \uparrow_{\rho} x]). \\ (\uparrow_{\rho \rightarrow \sigma} r) S &= (\lambda X \uparrow_{\sigma} (r \downarrow_{\rho} X)) S &= \uparrow_{\sigma} (r \downarrow_{\rho} S). \\ \downarrow_{\iota} \uparrow_{\iota} r &= r. \end{aligned}$$

Discard types — we only need to know whether function or not

▷ **Axioms.**

$$\begin{aligned} \downarrow \lambda X R &= \lambda x \downarrow (R[X := \uparrow x]). \\ (\uparrow r) S &= \uparrow (r \downarrow S). \\ \downarrow \uparrow r &= r. \end{aligned}$$

(Just another) 2-level-calculus

▷ **Terms.** $\Lambda \ni r, s, t ::= x \mid rs \mid \lambda x r,$

(Just another) 2-level-calculus

▷ **Terms.** $\Lambda \ni r, s, t ::= x \mid rs \mid \lambda x r,$
 $\Lambda_1 \ni r, s, t ::= x \mid rs \mid \lambda x r$

(Just another) 2-level-calculus

▷ **Terms.** $\Lambda \ni r, s, t ::= x \mid rs \mid \lambda x r,$
 $\Lambda_1 \ni r, s, t ::= x \mid rs \mid \lambda x r \mid \downarrow R$

(Just another) 2-level-calculus

▷ **Terms.**

Λ	\ni	r, s, t	$::=$	x		rs		$\lambda xr,$	
Λ_1	\ni	r, s, t	$::=$	x		rs		λxr	$\downarrow R$
Λ_2	\ni	R, S, T	$::=$	X		RS		λXR	

(Just another) 2-level-calculus

▷ **Terms.**

Λ	\ni	r, s, t	$::=$	x		rs		$\lambda xr,$	
Λ_1	\ni	r, s, t	$::=$	x		rs		λxr	$\downarrow R$
Λ_2	\ni	R, S, T	$::=$	X		RS		λXR	$\uparrow r$

(Just another) 2-level-calculus

- ▷ **Terms.**
- $$\begin{array}{l} \Lambda \quad \ni \quad r, s, t \quad ::= \quad x \mid rs \mid \lambda xr, \\ \Lambda_1 \quad \ni \quad r, s, t \quad ::= \quad x \mid rs \mid \lambda xr \mid \downarrow R \\ \Lambda_2 \quad \ni \quad R, S, T \quad ::= \quad X \mid RS \mid \lambda XR \mid \uparrow r \end{array}$$
- ▷ **Reductions.**
- $$\begin{array}{l} (\lambda XR)S \quad \mapsto_{\beta} \quad R[X := S] \\ (\uparrow r)S \quad \mapsto_a \quad \uparrow(r \downarrow S) \\ \downarrow \lambda XR \quad \mapsto_d \quad \lambda x \downarrow (R[X := \uparrow x]), \quad x \text{ new} \\ \downarrow \uparrow r \quad \mapsto_d \quad r \end{array}$$

(Just another) 2-level-calculus

- ▷ **Terms.** $\Lambda \ni r, s, t ::= x \mid rs \mid \lambda xr,$
 $\Lambda_1 \ni r, s, t ::= x \mid rs \mid \lambda xr \mid \downarrow R$
 $\Lambda_2 \ni R, S, T ::= X \mid RS \mid \lambda XR \mid \uparrow r$
- ▷ **Reductions.** $(\lambda XR)S \mapsto_{\beta} R[X := S]$
 $(\uparrow r)S \mapsto_a \uparrow(r \downarrow S)$
 $\downarrow \lambda XR \mapsto_d \lambda x \downarrow (R[X := \uparrow x]), \quad x \text{ new}$
 $\downarrow \uparrow r \mapsto_d r$
- ▷ This yields an orthogonal higher order rewrite system \Rightarrow **confluence**.

Evaluation

▷ **Definition.** $[[x]]_{\xi} := \xi x$, $[[rs]]_{\xi} := [[r]]_{\xi} [[s]]_{\xi}$, $[[\lambda x r]]_{\xi} := \lambda X. [[r]]_{\xi, x:=X}$.

Evaluation

- ▷ **Definition.** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \lambda X. [[r]]_{\xi, x:=X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies [[r]]_\xi \rightarrow_\beta [[s]]_\xi$.

Evaluation

- ▷ **Definition.** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \lambda X. [[r]]_{\xi, x:=X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies [[r]]_\xi \rightarrow_\beta [[s]]_\xi$.

$$[[(\lambda x r)s]]_\xi = (\lambda X [[r]]_{\xi, x:=X}) [[s]]_\xi$$

Evaluation

- ▷ **Definition.** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \lambda X. [[r]]_{\xi, x:=X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies [[r]]_\xi \rightarrow_\beta [[s]]_\xi$.

$$[[(\lambda x r) s]]_\xi = (\lambda X [[r]]_{\xi, x:=X}) [[s]]_\xi \rightarrow_\beta [[r]]_{\xi, x:=X} [X := [[s]]_\xi]$$

Evaluation

- ▷ **Definition.** $\llbracket x \rrbracket_\xi := \xi x$, $\llbracket rs \rrbracket_\xi := \llbracket r \rrbracket_\xi \llbracket s \rrbracket_\xi$, $\llbracket \lambda x r \rrbracket_\xi := \lambda X. \llbracket r \rrbracket_{\xi, x := X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies \llbracket r \rrbracket_\xi \rightarrow_\beta \llbracket s \rrbracket_\xi$.

$$\begin{aligned} \llbracket (\lambda x r) s \rrbracket_\xi &= (\lambda X \llbracket r \rrbracket_{\xi, x := X}) \llbracket s \rrbracket_\xi \rightarrow_\beta \llbracket r \rrbracket_{\xi, x := X} [X := \llbracket s \rrbracket_\xi] \\ &= \llbracket r \rrbracket_{\xi, x := \llbracket s \rrbracket_\xi} \end{aligned}$$

Evaluation

- ▷ **Definition.** $\llbracket x \rrbracket_\xi := \xi x$, $\llbracket rs \rrbracket_\xi := \llbracket r \rrbracket_\xi \llbracket s \rrbracket_\xi$, $\llbracket \lambda x r \rrbracket_\xi := \lambda X. \llbracket r \rrbracket_{\xi, x := X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies \llbracket r \rrbracket_\xi \rightarrow_\beta \llbracket s \rrbracket_\xi$.

$$\begin{aligned} \llbracket (\lambda x r) s \rrbracket_\xi &= (\lambda X \llbracket r \rrbracket_{\xi, x := X}) \llbracket s \rrbracket_\xi \rightarrow_\beta \llbracket r \rrbracket_{\xi, x := X} [X := \llbracket s \rrbracket_\xi] \\ &= \llbracket r \rrbracket_{\xi, x := \llbracket s \rrbracket_\xi} = \llbracket r[x := s] \rrbracket_\xi \end{aligned}$$

Evaluation

- ▷ **Definition.** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda xr]]_\xi := \lambda X. [[r]]_{\xi, x:=X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies [[r]]_\xi \rightarrow_\beta [[s]]_\xi$.

$$\begin{aligned} [[(\lambda xr)s]]_\xi &= (\lambda X [[r]]_{\xi, x:=X}) [[s]]_\xi \rightarrow_\beta [[r]]_{\xi, x:=X} [X := [[s]]_\xi] \\ &= [[r]]_{\xi, x:=[[s]]_\xi} = [[r[x := s]]]_\xi \end{aligned}$$

- ▷ **Reproduction.** $\downarrow [[r]]_\uparrow = r$ for normal forms r .

$$\downarrow [[x\vec{r}]]_\uparrow$$

Evaluation

- ▷ **Definition.** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \lambda X. [[r]]_{\xi, x:=X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies [[r]]_\xi \rightarrow_\beta [[s]]_\xi$.

$$\begin{aligned} [[(\lambda x r)s]]_\xi &= (\lambda X [[r]]_{\xi, x:=X}) [[s]]_\xi \rightarrow_\beta [[r]]_{\xi, x:=X} [X := [[s]]_\xi] \\ &= [[r]]_{\xi, x:=[[s]]_\xi} = [[r[x := s]]]_\xi \end{aligned}$$

- ▷ **Reproduction.** $\downarrow [[r]]_\uparrow = r$ for normal forms r .

$$\downarrow [[x\vec{r}]]_\uparrow = \downarrow ((\uparrow x) [[\vec{r}]]_\uparrow)$$

Evaluation

- ▷ **Definition.** $\llbracket x \rrbracket_\xi := \xi x$, $\llbracket rs \rrbracket_\xi := \llbracket r \rrbracket_\xi \llbracket s \rrbracket_\xi$, $\llbracket \lambda x r \rrbracket_\xi := \lambda X. \llbracket r \rrbracket_{\xi, x := X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies \llbracket r \rrbracket_\xi \rightarrow_\beta \llbracket s \rrbracket_\xi$.

$$\begin{aligned} \llbracket (\lambda x r) s \rrbracket_\xi &= (\lambda X \llbracket r \rrbracket_{\xi, x := X}) \llbracket s \rrbracket_\xi \rightarrow_\beta \llbracket r \rrbracket_{\xi, x := X} [X := \llbracket s \rrbracket_\xi] \\ &= \llbracket r \rrbracket_{\xi, x := \llbracket s \rrbracket_\xi} = \llbracket r[x := s] \rrbracket_\xi \end{aligned}$$

- ▷ **Reproduction.** $\downarrow \llbracket r \rrbracket_\uparrow = r$ for normal forms r .

$$\downarrow \llbracket x \vec{r} \rrbracket_\uparrow = \downarrow ((\uparrow x) \llbracket \vec{r} \rrbracket_\uparrow) \rightarrow_a^* \downarrow \uparrow (x \downarrow \llbracket \vec{r} \rrbracket_\uparrow)$$

Evaluation

- ▷ **Definition.** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \lambda X. [[r]]_{\xi, x := X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies [[r]]_\xi \rightarrow_\beta [[s]]_\xi$.

$$\begin{aligned} [[(\lambda x r)s]]_\xi &= (\lambda X [[r]]_{\xi, x := X}) [[s]]_\xi \rightarrow_\beta [[r]]_{\xi, x := X} [X := [[s]]_\xi] \\ &= [[r]]_{\xi, x := [[s]]_\xi} = [[r[x := s]]]_\xi \end{aligned}$$

- ▷ **Reproduction.** $\downarrow [[r]]_\uparrow = r$ for normal forms r .

$$\downarrow [[x\vec{r}]]_\uparrow = \downarrow ((\uparrow x) [[\vec{r}]]_\uparrow) \rightarrow_a^* \downarrow \uparrow (x \downarrow [[\vec{r}]]_\uparrow) \rightarrow_d x \downarrow [[\vec{r}]]_\uparrow$$

Evaluation

- ▷ **Definition.** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \lambda X. [[r]]_{\xi, x := X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies [[r]]_\xi \rightarrow_\beta [[s]]_\xi$.

$$\begin{aligned} [[(\lambda x r)s]]_\xi &= (\lambda X [[r]]_{\xi, x := X}) [[s]]_\xi \rightarrow_\beta [[r]]_{\xi, x := X} [X := [[s]]_\xi] \\ &= [[r]]_{\xi, x := [[s]]_\xi} = [[r[x := s]]]_\xi \end{aligned}$$

- ▷ **Reproduction.** $\downarrow [[r]]_\uparrow = r$ for normal forms r .

$$\downarrow [[x\vec{r}]]_\uparrow = \downarrow ((\uparrow x) [[\vec{r}]]_\uparrow) \rightarrow_a^* \downarrow \uparrow (x \downarrow [[\vec{r}]]_\uparrow) \rightarrow_d x \downarrow [[\vec{r}]]_\uparrow \rightarrow_s^* x\vec{r}.$$

Evaluation

- ▷ **Definition.** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \lambda X. [[r]]_{\xi, x:=X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies [[r]]_\xi \rightarrow_\beta [[s]]_\xi$.

$$\begin{aligned} [[(\lambda x r) s]]_\xi &= (\lambda X [[r]]_{\xi, x:=X}) [[s]]_\xi \rightarrow_\beta [[r]]_{\xi, x:=X} [X := [[s]]_\xi] \\ &= [[r]]_{\xi, x:=[[s]]_\xi} = [[r[x := s]]]_\xi \end{aligned}$$

- ▷ **Reproduction.** $\downarrow [[r]]_\uparrow = r$ for normal forms r .

$$\downarrow [[x\vec{r}]]_\uparrow = \downarrow ((\uparrow x) [[\vec{r}]]_\uparrow) \rightarrow_a^* \downarrow \uparrow (x \downarrow [[\vec{r}]]_\uparrow) \rightarrow_d x \downarrow [[\vec{r}]]_\uparrow \rightarrow_s^* x\vec{r}.$$

$$\downarrow [[\lambda x r]]_\uparrow$$

Evaluation

- ▷ **Definition.** $\llbracket x \rrbracket_\xi := \xi x$, $\llbracket rs \rrbracket_\xi := \llbracket r \rrbracket_\xi \llbracket s \rrbracket_\xi$, $\llbracket \lambda x r \rrbracket_\xi := \lambda X. \llbracket r \rrbracket_{\xi, x := X}$.
- ▷ **Simulation.** $r \rightarrow_\beta s \implies \llbracket r \rrbracket_\xi \rightarrow_\beta \llbracket s \rrbracket_\xi$.

$$\begin{aligned} \llbracket (\lambda x r) s \rrbracket_\xi &= (\lambda X \llbracket r \rrbracket_{\xi, x := X}) \llbracket s \rrbracket_\xi \rightarrow_\beta \llbracket r \rrbracket_{\xi, x := X} [X := \llbracket s \rrbracket_\xi] \\ &= \llbracket r \rrbracket_{\xi, x := \llbracket s \rrbracket_\xi} = \llbracket r[x := s] \rrbracket_\xi \end{aligned}$$

- ▷ **Reproduction.** $\downarrow \llbracket r \rrbracket_\uparrow = r$ for normal forms r .

$$\downarrow \llbracket x \vec{r} \rrbracket_\uparrow = \downarrow ((\uparrow x) \llbracket \vec{r} \rrbracket_\uparrow) \rightarrow_a^* \downarrow \uparrow (x \downarrow \llbracket \vec{r} \rrbracket_\uparrow) \rightarrow_d x \downarrow \llbracket \vec{r} \rrbracket_\uparrow \rightarrow_s^* x \vec{r}.$$

$$\downarrow \llbracket \lambda x r \rrbracket_\uparrow = \downarrow \lambda X. \llbracket r \rrbracket_{\uparrow, x := X}$$

Evaluation

▷ **Definition.** $\llbracket x \rrbracket_\xi := \xi x$, $\llbracket rs \rrbracket_\xi := \llbracket r \rrbracket_\xi \llbracket s \rrbracket_\xi$, $\llbracket \lambda x r \rrbracket_\xi := \lambda X. \llbracket r \rrbracket_{\xi, x := X}$.

▷ **Simulation.** $r \rightarrow_\beta s \implies \llbracket r \rrbracket_\xi \rightarrow_\beta \llbracket s \rrbracket_\xi$.

$$\begin{aligned} \llbracket (\lambda x r) s \rrbracket_\xi &= (\lambda X \llbracket r \rrbracket_{\xi, x := X}) \llbracket s \rrbracket_\xi \rightarrow_\beta \llbracket r \rrbracket_{\xi, x := X} [X := \llbracket s \rrbracket_\xi] \\ &= \llbracket r \rrbracket_{\xi, x := \llbracket s \rrbracket_\xi} = \llbracket r[x := s] \rrbracket_\xi \end{aligned}$$

▷ **Reproduction.** $\downarrow \llbracket r \rrbracket_\uparrow = r$ for normal forms r .

$$\downarrow \llbracket x \vec{r} \rrbracket_\uparrow = \downarrow ((\uparrow x) \llbracket \vec{r} \rrbracket_\uparrow) \rightarrow_a^* \downarrow \uparrow (x \downarrow \llbracket \vec{r} \rrbracket_\uparrow) \rightarrow_d x \downarrow \llbracket \vec{r} \rrbracket_\uparrow \rightarrow_s^* x \vec{r}.$$

$$\downarrow \llbracket \lambda x r \rrbracket_\uparrow = \downarrow \lambda X. \llbracket r \rrbracket_{\uparrow, x := X} \rightarrow_d \lambda x \downarrow \llbracket r \rrbracket_{\uparrow, x := X} [X := \uparrow x]$$

Evaluation

▷ **Definition.** $[[x]]_\xi := \xi x$, $[[rs]]_\xi := [[r]]_\xi [[s]]_\xi$, $[[\lambda x r]]_\xi := \lambda X. [[r]]_{\xi, x := X}$.

▷ **Simulation.** $r \rightarrow_\beta s \implies [[r]]_\xi \rightarrow_\beta [[s]]_\xi$.

$$\begin{aligned} [[(\lambda x r)s]]_\xi &= (\lambda X [[r]]_{\xi, x := X}) [[s]]_\xi \rightarrow_\beta [[r]]_{\xi, x := X} [X := [[s]]_\xi] \\ &= [[r]]_{\xi, x := [[s]]_\xi} = [[r[x := s]]]_\xi \end{aligned}$$

▷ **Reproduction.** $\downarrow [[r]]_\uparrow = r$ for normal forms r .

$$\downarrow [[x\vec{r}]]_\uparrow = \downarrow ((\uparrow x) [[\vec{r}]]_\uparrow) \rightarrow_a^* \downarrow \uparrow (x \downarrow [[\vec{r}]]_\uparrow) \rightarrow_d x \downarrow [[\vec{r}]]_\uparrow \rightarrow_s^* x\vec{r}.$$

$$\downarrow [[\lambda x r]]_\uparrow = \downarrow \lambda X. [[r]]_\uparrow, x := X \rightarrow_d \lambda x \downarrow [[r]]_\uparrow, x := X [X := \uparrow x] = \lambda x \downarrow [[r]]_\uparrow$$

Evaluation

▷ **Definition.** $\llbracket x \rrbracket_\xi := \xi x$, $\llbracket rs \rrbracket_\xi := \llbracket r \rrbracket_\xi \llbracket s \rrbracket_\xi$, $\llbracket \lambda x r \rrbracket_\xi := \lambda X. \llbracket r \rrbracket_{\xi, x := X}$.

▷ **Simulation.** $r \rightarrow_\beta s \implies \llbracket r \rrbracket_\xi \rightarrow_\beta \llbracket s \rrbracket_\xi$.

$$\begin{aligned} \llbracket (\lambda x r) s \rrbracket_\xi &= (\lambda X \llbracket r \rrbracket_{\xi, x := X}) \llbracket s \rrbracket_\xi \rightarrow_\beta \llbracket r \rrbracket_{\xi, x := X} [X := \llbracket s \rrbracket_\xi] \\ &= \llbracket r \rrbracket_{\xi, x := \llbracket s \rrbracket_\xi} = \llbracket r[x := s] \rrbracket_\xi \end{aligned}$$

▷ **Reproduction.** $\downarrow \llbracket r \rrbracket_\uparrow = r$ for normal forms r .

$$\downarrow \llbracket x \vec{r} \rrbracket_\uparrow = \downarrow ((\uparrow x) \llbracket \vec{r} \rrbracket_\uparrow) \rightarrow_a^* \downarrow \uparrow (x \downarrow \llbracket \vec{r} \rrbracket_\uparrow) \rightarrow_d x \downarrow \llbracket \vec{r} \rrbracket_\uparrow \rightarrow_s^* x \vec{r}.$$

$$\downarrow \llbracket \lambda x r \rrbracket_\uparrow = \downarrow \lambda X. \llbracket r \rrbracket_{\uparrow, x := X} \rightarrow_d \lambda x \downarrow \llbracket r \rrbracket_{\uparrow, x := X} [X := \uparrow x] = \lambda x \downarrow \llbracket r \rrbracket_\uparrow \rightarrow_s^* \lambda x r.$$

Evaluation

▷ **Definition.** $\llbracket x \rrbracket_\xi := \xi x$, $\llbracket rs \rrbracket_\xi := \llbracket r \rrbracket_\xi \llbracket s \rrbracket_\xi$, $\llbracket \lambda x r \rrbracket_\xi := \lambda X. \llbracket r \rrbracket_{\xi, x := X}$.

▷ **Simulation.** $r \rightarrow_\beta s \implies \llbracket r \rrbracket_\xi \rightarrow_\beta \llbracket s \rrbracket_\xi$.

$$\begin{aligned} \llbracket (\lambda x r) s \rrbracket_\xi &= (\lambda X \llbracket r \rrbracket_{\xi, x := X}) \llbracket s \rrbracket_\xi \rightarrow_\beta \llbracket r \rrbracket_{\xi, x := X} [X := \llbracket s \rrbracket_\xi] \\ &= \llbracket r \rrbracket_{\xi, x := \llbracket s \rrbracket_\xi} = \llbracket r[x := s] \rrbracket_\xi \end{aligned}$$

▷ **Reproduction.** $\downarrow \llbracket r \rrbracket_\uparrow = r$ for normal forms r .

$$\downarrow \llbracket x \vec{r} \rrbracket_\uparrow = \downarrow ((\uparrow x) \llbracket \vec{r} \rrbracket_\uparrow) \rightarrow_a^* \downarrow \uparrow (x \downarrow \llbracket \vec{r} \rrbracket_\uparrow) \rightarrow_d x \downarrow \llbracket \vec{r} \rrbracket_\uparrow \rightarrow_s^* x \vec{r}.$$

$$\downarrow \llbracket \lambda x r \rrbracket_\uparrow = \downarrow \lambda X. \llbracket r \rrbracket_{\uparrow, x := X} \rightarrow_d \lambda x \downarrow \llbracket r \rrbracket_{\uparrow, x := X} [X := \uparrow x] = \lambda x \downarrow \llbracket r \rrbracket_\uparrow \rightarrow_s^* \lambda x r.$$

▷ **Corollary.** $r =_\beta s \in \mathbf{NF}_\beta \implies \downarrow \llbracket r \rrbracket_\uparrow \rightarrow^* s$.

Implementation: Object calculus

Implementation: Object calculus

Choose de Bruijn terms for simplicity.

```
data Term = Var Integer | App Term Term | Abs Term deriving Show
```

Implementation: Object calculus

Choose de Bruijn terms for simplicity.

```
data Term = Var Integer | App Term Term | Abs Term deriving Show
```

An interface to make lifting easy:

```
type TERM = Integer -> Term

inspect :: TERM -> Term
inspect f = f 0

freevar k = \ n -> Var (n + k)
boundvar k = \ n -> Var (n - k - 1)
apply r s = \ n -> App (r n) (s n)
bind f     = \ n -> Abs (f (boundvar n) (n + 1))
```

Implementation — Metalevel calculus

Terms.

$\uparrow r$ | $\lambda X R$ | RS | X

Implementation — Metalevel calculus

Terms.

$\uparrow r$ | $\lambda X R$ | RS | X

data LAM = Up TERM | ABS (LAM -> LAM)

Implementation — Metalevel calculus

Terms.

 $\uparrow r \mid \lambda X R \mid RS \mid X$

```
data LAM = Up TERM | ABS (LAM -> LAM)
```

Reductions for \downarrow :

$$\begin{array}{l} \downarrow \uparrow r \quad \mapsto_d \quad r \\ \downarrow \lambda X R \quad \mapsto_d \quad \lambda x \downarrow (R[X := \uparrow x]) \end{array}$$

Implementation — Metalevel calculus

Terms.

 $\uparrow r \mid \lambda X R \mid RS \mid X$

```
data LAM = Up TERM | ABS (LAM -> LAM)
```

Reductions for \downarrow :

$$\begin{array}{l} \downarrow \uparrow r \mapsto_d r \\ \downarrow \lambda X R \mapsto_d \lambda x \downarrow (R[X := \uparrow x]) \end{array}$$

```
down :: LAM -> TERM          -- the d-rules
down (ABS f) = bind (\ x -> (down (f (Up x))))
down (Up r)  = r
```

Implementation — Metalevel calculus

Terms.

$\uparrow r \mid \lambda X R \mid RS \mid X$

```
data LAM = Up TERM | ABS (LAM -> LAM)
```

Reductions for \downarrow :

$$\begin{aligned} \downarrow \uparrow r &\mapsto_d r \\ \downarrow \lambda X R &\mapsto_d \lambda x \downarrow (R[X := \uparrow x]) \end{aligned}$$

```
down :: LAM -> TERM          -- the d-rules
down (ABS f) = bind (\ x -> (down (f (Up x))))
down (Up r)  = r
```

Reductions for application:

$$\begin{aligned} (\lambda X R) S &\mapsto_\beta R[X := S] \\ (\uparrow r) S &\mapsto_a \uparrow (r \downarrow S) \end{aligned}$$

Implementation — Metalevel calculus

Terms.

 $\uparrow r \mid \lambda X R \mid RS \mid X$

```
data LAM = Up TERM | ABS (LAM -> LAM)
```

Reductions for \downarrow :

$$\begin{aligned} \downarrow \uparrow r &\mapsto_d r \\ \downarrow \lambda X R &\mapsto_d \lambda x \downarrow (R[X := \uparrow x]) \end{aligned}$$

```
down :: LAM -> TERM          -- the d-rules
down (ABS f) = bind (\ x -> (down (f (Up x))))
down (Up r)  = r
```

Reductions for application:

$$\begin{aligned} (\lambda X R) S &\mapsto_\beta R[X := S] \\ (\uparrow r) S &\mapsto_a \uparrow (r \downarrow S) \end{aligned}$$

```
app :: LAM -> LAM -> LAM
app (ABS f) r = f r          -- the beta-rule
app (Up r) s = Up (apply r (down s)) -- the a-rule
```

Implementation — Evaluation

```
type Valuation = Integer -> LAM
comma :: Valuation -> LAM -> Valuation
comma xi r n = if n == 0 then r else xi (n - 1)            $\xi, x := R$ 

eval :: Term -> Valuation -> LAM
eval (Var n)    xi = xi n                                   $\xi x$ 
eval (Abs r)    xi = ABS (\ a -> eval r (xi 'comma' a))   $\lambda X. [[r]]_{\xi, x := X}$ 
eval (App r s) xi = (eval r xi) 'app' (eval s xi)        $[[r]]_{\xi} [[s]]_{\xi}$ 

nbe r = inspect (down (eval r (Up . freevar)))
          =      (down (eval r (Up . freevar))) 0         $\downarrow [[r]]_{\uparrow}$ 
```

Implementation — Examples

`k = Abs (Abs (Var 1))`

$k = \lambda\lambda 1$

`s = Abs (Abs (Abs (Var 2 'App' (Var 0) 'App' (Var 1 'App' (Var 0)))))`

$s = \lambda\lambda\lambda.20(10)$

`i = s 'App' k 'App' k`

$\omega = \lambda.00$

`omega = Abs (Var 0 'App' (Var 0))`

$\Omega = \omega\omega$

`oomega = omega 'App' omega`

Implementation — Examples

`k = Abs (Abs (Var 1))`

$k = \lambda\lambda 1$

`s = Abs (Abs (Abs (Var 2 'App' (Var 0) 'App' (Var 1 'App' (Var 0)))))`

$s = \lambda\lambda\lambda.20(10)$

`i = s 'App' k 'App' k`

$\omega = \lambda.00$

`omega = Abs (Var 0 'App' (Var 0))`

$\Omega = \omega\omega$

`oomega = omega 'App' omega`

`Main> nbe k`

`Abs (Abs (Var 1))`

Implementation — Examples

`k = Abs (Abs (Var 1))`

$k = \lambda\lambda 1$

`s = Abs (Abs (Abs (Var 2 'App' (Var 0) 'App' (Var 1 'App' (Var 0)))))`

$s = \lambda\lambda\lambda.20(10)$

`i = s 'App' k 'App' k`

$\omega = \lambda.00$

`omega = Abs (Var 0 'App' (Var 0))`

$\Omega = \omega\omega$

`oomega = omega 'App' omega`

`Main> nbe k`

`Abs (Abs (Var 1))`

`Main> nbe i`

`Abs (Var 0)`

Implementation — Examples

`k = Abs (Abs (Var 1))`

$k = \lambda\lambda 1$

`s = Abs (Abs (Abs (Var 2 'App' (Var 0) 'App' (Var 1 'App' (Var 0)))))`

$s = \lambda\lambda\lambda.20(10)$

`i = s 'App' k 'App' k`

$\omega = \lambda.00$

`omega = Abs (Var 0 'App' (Var 0))`

$\Omega = \omega\omega$

`oomega = omega 'App' omega`

`Main> nbe k`

`Abs (Abs (Var 1))`

`Main> nbe i`

`Abs (Var 0)`

`Main> nbe (k 'App' i 'App' oomega)`

`Abs (Var 0)`

Standardization

- ▷ **Standard reduction.** Head redexes are executed first or never.

Standardization

- ▷ **Standard reduction.** Head redexes are executed first or never.
- ▷ **Theorem.** Reduction sequences can be standardized.

Standardization

- ▷ **Standard reduction.** Head redexes are executed first or never.
- ▷ **Theorem.** Reduction sequences can be standardized.
- ▷ **Corollary.** Normalization (= reduction to normal form)
 - need not reduce under meta-abstractions,
 - may follow a call-by-name strategy.

Standardization

- ▷ **Standard reduction.** Head redexes are executed first or never.
- ▷ **Theorem.** Reduction sequences can be standardized.
- ▷ **Corollary.** Normalization (= reduction to normal form)
 - need not reduce under meta-abstractions,
 - may follow a call-by-name strategy.
- ▷ **Corollary.** We can compute (the defined parts of) Böhm trees — normalization for diverging terms.

Böhm trees — Examples

```
pair = Abs (Abs (Abs (Var 0 'App' Var 2 'App' Var 1)))  
swap = Abs (Abs (Abs (Var 0 'App' Var 1 'App' Var 2)))
```

Böhm trees — Examples

```
pair = Abs (Abs (Abs (Var 0 'App' Var 2 'App' Var 1)))  
swap = Abs (Abs (Abs (Var 0 'App' Var 1 'App' Var 2)))
```

```
Main> nbe (pair 'App' i 'App' oomega)  
Abs (App (App (Var 0) (Abs (Var 0))) {Interrupted!})
```

Böhm trees — Examples

```
pair = Abs (Abs (Abs (Var 0 'App' Var 2 'App' Var 1)))
swap = Abs (Abs (Abs (Var 0 'App' Var 1 'App' Var 2)))
```

```
Main> nbe (pair 'App' i 'App' oomega)
Abs (App (App (Var 0) (Abs (Var 0))) {Interrupted!})
```

```
Main> cut (nbe (pair 'App' i 'App' oomega)) ["01"]
Abs (App (App (Var 0) (Abs (Var 0))) (Var (-1)))
```

Böhm trees — Examples

```
pair = Abs (Abs (Abs (Var 0 'App' Var 2 'App' Var 1)))  
swap = Abs (Abs (Abs (Var 0 'App' Var 1 'App' Var 2)))
```

```
Main> nbe (pair 'App' i 'App' oomega)  
Abs (App (App (Var 0) (Abs (Var 0))) {Interrupted!})
```

```
Main> cut (nbe (pair 'App' i 'App' oomega)) ["01"]  
Abs (App (App (Var 0) (Abs (Var 0))) (Var (-1)))
```

```
Main> nbe (pair 'App' i 'App' oomega 'App' swap)  
Abs (App (App (Var 0) {Interrupted!})
```

Böhm trees — Examples

```
pair = Abs (Abs (Abs (Var 0 'App' Var 2 'App' Var 1)))
swap = Abs (Abs (Abs (Var 0 'App' Var 1 'App' Var 2)))
```

```
Main> nbe (pair 'App' i 'App' oomega)
Abs (App (App (Var 0) (Abs (Var 0))) {Interrupted!})
```

```
Main> cut (nbe (pair 'App' i 'App' oomega)) ["01"]
Abs (App (App (Var 0) (Abs (Var 0))) (Var (-1)))
```

```
Main> nbe (pair 'App' i 'App' oomega 'App' swap)
Abs (App (App (Var 0) {Interrupted!})
```

```
Main> cut (nbe (pair 'App' i 'App' oomega 'App' swap)) ["001"]
Abs (App (App (Var 0) (Var (-1))) (Abs (Var 0)))
```

Types — Computing η -long normal forms

Types — Computing η -long normal forms

▷ **Head expansion.** Instead of $\downarrow\uparrow r \mapsto_d r$ define

$$\downarrow\uparrow r^\rho \mapsto_d \eta_\rho r \quad \text{with} \quad \eta_\iota r := r \quad \text{and} \quad \eta_{\rho \rightarrow \sigma} r := \lambda x. \eta_\sigma (r \eta_\rho x).$$

Types — Computing η -long normal forms

▷ **Head expansion.** Instead of $\downarrow\uparrow r \mapsto_d r$ define

$$\downarrow\uparrow r^\rho \mapsto_d \eta_\rho r \quad \text{with} \quad \eta_\iota r := r \quad \text{and} \quad \eta_{\rho \rightarrow \sigma} r := \lambda x. \eta_\sigma (r \eta_\rho x).$$

▷ **Theorem.** On typed terms \rightarrow is SN and computes η -long normal forms.

▷ **Corollary.** Typed NBE (as defined in the beginning) is a model.

Conclusions

- ▷ Normalization by Evaluation needs no types — the information of whether a function or not suffices.
- ▷ Non-termination: (partial) normal forms are found whenever they exist.
- ▷ Use Haskell for normalization (= doing the substitutions).
- ▷ Type-directed NBE as an example.

Technical details

Theorem. $r =_{\beta} \lambda \vec{x}(x \vec{r}) \ \& \ s \rightarrow^* \downarrow[[r]]$
 $\implies \exists \vec{t}, \vec{s}. s \twoheadrightarrow \lambda \vec{x}(x \vec{s}) \ \& \ \vec{r} =_{\beta} \vec{t} \quad \& \ \vec{s} \rightarrow^* \downarrow[[\vec{t}]].$

Proof.

$r \rightarrow_{\beta}^* \lambda \vec{x}(x \vec{t}) \xrightarrow{\beta}^* \lambda \vec{x}(x \vec{r})$ by the Church-Rosser prop. of \rightarrow_{β} ,
 $s \rightarrow^* \downarrow[[r]] \rightarrow^* \lambda \vec{x}(x \downarrow[[\vec{t}]])$
 $s \rightsquigarrow \lambda \vec{x}(x \downarrow[[\vec{t}]])$ by standardization,
 $s \twoheadrightarrow \lambda \vec{x}(x \vec{s}) \rightarrow^* \lambda \vec{x}(x \downarrow[[\vec{t}]])$ property of \twoheadrightarrow .

Theorem. $\downarrow[[t]] \rightarrow^* \lambda \vec{x}(x \vec{R}) \implies t \rightarrow^* \lambda \vec{x}(x \vec{t}).$