Strong Reduction for the Pure $\lambda$-calculus
by Benjamin Grégoire and Xavier Leroy

Suzanne van den Bosch

December 5, 2014
- Weak $\beta$-reduction
- Strong $\beta$-reduction
The Pure $\lambda$-calculus

Terms:

\[ a ::= x | \lambda x. a | a_1 a_2 \]

Rules:

\[ (\lambda x. a) a' \Rightarrow a\{x \leftarrow a'\} \]  
\[ \Gamma(a) \Rightarrow \Gamma(a') \text{ if } a \Rightarrow a' \]  
\[ (\text{context}) \]

with $\Gamma ::= \lambda x.[] | [] a | a []$. We assume all $\lambda$-terms $a$ are strongly normalizing.
Two computational problems

- To compute the normal form $\mathcal{N}(a)$ of a closed, strongly normalizing term $a$.
- To decide whether two closed, strongly normalizing term $a_1$ and $a_2$ are $\beta$-equivalent, written as $a_1 \approx a_2$. 
Two computational problems

- To compute the normal form $N(a)$ of a closed, strongly normalizing term $a$.
- To decide whether two closed, strongly normalizing term $a_1$ and $a_2$ are $\beta$-equivalent, written as $a_1 \approx a_2$. 
Strong reduction by iterated symbolic weak reduction and readback

\[ N(a) = N(\lambda x. a') = \lambda x. N(a') \]

Problem: \( a' \) is not necessarily closed.
The extended version

Terms:

\[ b ::= x \mid \lambda x. b \mid b_1 b_2 \mid [\tilde{x}v_1\ldots v_n] \]

Values:

\[ v ::= \lambda x. b \mid [\tilde{x}v_1\ldots v_n] \]

Rules:

\[ (\lambda x. b)v \rightarrow_v b[x \leftarrow v] \quad (\beta_v) \]

\[ [\tilde{x}v_1\ldots v_n]v \rightarrow_v [\tilde{x}v_1\ldots v_nv] \quad (\beta_s) \]

\[ \Gamma_v(a) \rightarrow_v \Gamma_v(a') \quad \text{if } a \rightarrow_v a' \quad (\text{context}_v) \]

with \( \Gamma_v ::= [\ ] v \mid b[\ ] \).
Strong normalization procedure

1. Normalize weakly
2. Read back
\[ N(b) = R(\mathcal{N}(b)) \]  
\[ R(\lambda x. b) = \lambda y. N((\lambda x. b)[\tilde{y}]) \quad (y \text{ fresh}) \]  
\[ R([\tilde{x}v_1...v_n]) = xR(v_1)...R(v_n) \]

\( R \) transforms values \( v \) into normalized source terms \( a \).
Consider the following source term

\[ a = (\lambda x.x)(\lambda y.(\lambda z.z)y(\lambda t.t)). \]

Weak symbolic evaluation reduces \( a \) to

\[ v = \lambda y.(\lambda z.z)y(\lambda t.t). \]

The readback will restart weak symbolic evaluation on

\[ b = (\lambda y.(\lambda z.z)y(\lambda t.t))[\tilde{u}]. \]

After the weak symbolic evaluation, the value is

\[ v' = [\tilde{u}(\lambda t.t)]. \]

Eventually, we will get

\[ \mathcal{N}(a) = \mathcal{R}(v) = \lambda u.u(\lambda w.w). \]