

Normalization by Evaluation for System F

Andreas Abel

Department of Computer Science
Ludwig-Maximilians-University Munich

Department of Mathematics, Savoie University
Chambery, France
12 February 2010

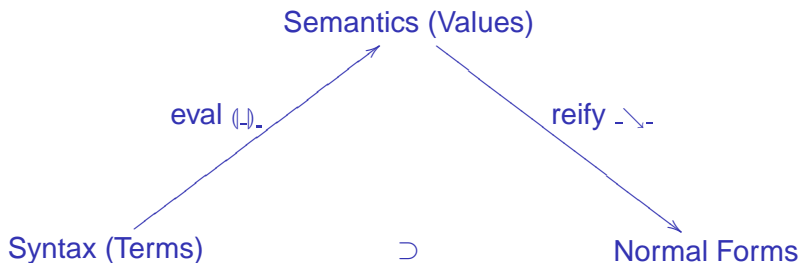
Introduction

- Normalizers appear in compilers (e.g., type-directed partial evaluation [Danvy, Filinski])
- and HOL theorem provers (Isabelle, Coq, Agda).

Normalization by evaluation is a framework to turn an evaluator for closed expressions (stop at lambda) into a normalizer for open expressions (go under lambda).

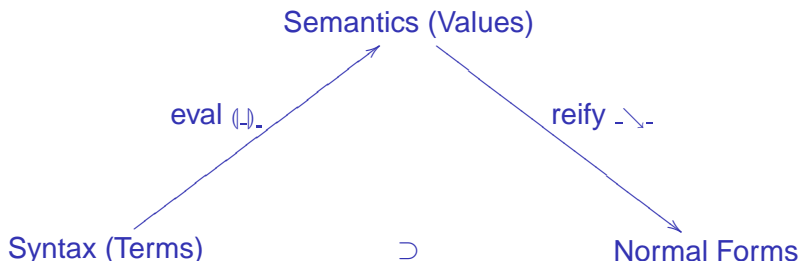
- Has clear semantic foundations.
- Is strong for extensional normalization (eta).
- My goal: NbE for Calculus of Constructions and Coq.

What is Normalization By Evaluation?



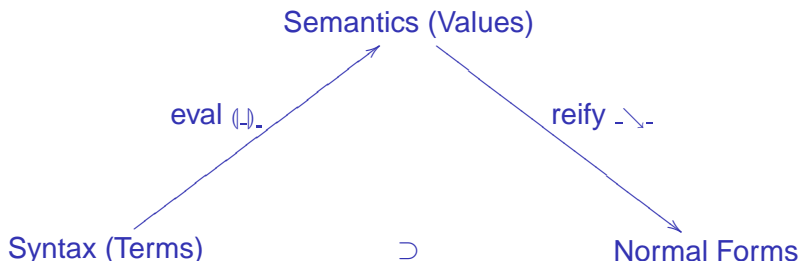
- You have: an interpreter $(|-)_$.
- You buy: my reifier $(_ \ \ \ _)$.
- You get for free: a *full normalizer!*

What is Normalization By Evaluation?



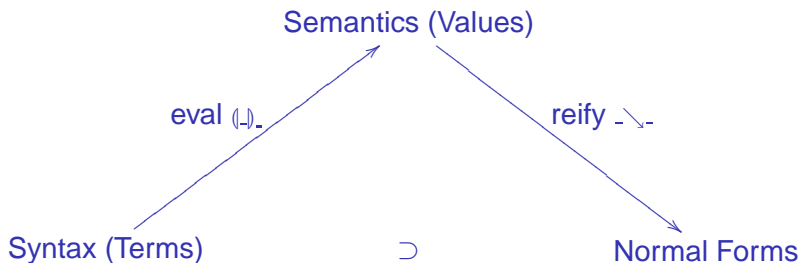
- You have: an interpreter (`(|_)_`).
- You buy: my reifier (`_ _ -`).
- You get for free: a *full normalizer!*

What is Normalization By Evaluation?



- You have: an interpreter $(|-)$.
- You buy: my reifier $(- \rightarrow -)$.
- You get for free: a *full normalizer!*

What is Normalization By Evaluation?



- You have: an interpreter $(|-)$.
- You buy: my reifier $(- \to -)$.
- You get for free: a *full normalizer*!

How to Reify a Function

- Functions are thought of as *black boxes*.
- How to print the code of a function?
- Apply it to a fresh variable!

$$\begin{aligned}\text{reify}(f) &= \lambda x. \text{reify}(f(x)) \\ \text{reify}(x \vec{d}) &= x \text{reify}(\vec{d})\end{aligned}$$

- Computation needs to be extended to handle variables (unknowns).

Choices of Semantics

- 1 β -normal forms (Agda 2, Ulf Norell)
- 2 Weak head normal forms (Constructive Engine, Randy Pollack)
- 3 Explicit substitutions (Twelf, Pfenning et.al.)
- 4 Closures (your favorite pure functional language, Epigram 2)
- 5 Virtual machine code (Coq: ZINC machine, Leroy/Gregoire)
- 6 Native machine code (Cayenne: i386, Dirk Kleeblatt)

These are all (partial) *applicative structures*.

Applicative Structures

An applicative structure consists of:

- A set D .
- Application operation $_ \cdot _ : D \times D \rightarrow D$.
- Interpretation $\langle t \rangle_\eta \in D$ for term t and environment η , satisfying:

$$\begin{aligned}\langle x \rangle_\eta &= \eta(x) \\ \langle r s \rangle_\eta &= \langle r \rangle_\eta \cdot \langle s \rangle_\eta \\ \langle \lambda x t \rangle_\eta \cdot d &= \langle t \rangle_{\eta[x \mapsto d]}\end{aligned}$$

Simple examples:

- 1 $D = (\text{Tm}/\equiv_\beta)$ terms modulo β -equality.
- 2 $D \cong [D \rightarrow D]$ reflexive (Scott) domain.

Applicative Structures with Variables

- Enrich \mathbf{D} with all neutral objects $x d_1 \dots d_n$, where x a variable and $d_1, \dots, d_n \in \mathbf{D}$.

- Application satisfies:

$$(x \vec{d}) \cdot d = x \vec{d} d$$

- Leroy/Gregoire call neutral objects *accumulators*.

β -NbE for Untyped Lambda-Calculus

Let $I = \lambda y. y$ identity.

$$\begin{aligned} & \downarrow \llbracket \lambda x. I x \rrbracket & = & \lambda x_1. \downarrow (x_1 \cdot \llbracket I \rrbracket) \\ = & \lambda x_1. \downarrow (\llbracket \lambda x. I x \rrbracket \cdot x_1) & = & \lambda x_1. x_1 (\downarrow \llbracket I \rrbracket) \\ = & \lambda x_1. \downarrow (\llbracket I x \rrbracket_{x \rightarrow x_1}) & = & \lambda x_1. x_1 (\lambda x_2. \downarrow (\llbracket I \rrbracket \cdot x_2)) \\ = & \lambda x_1. \downarrow (\llbracket I \rrbracket \cdot \llbracket x \rrbracket_{x \rightarrow x_1} \cdot \llbracket I \rrbracket) & = & \lambda x_1. x_1 (\lambda x_2. \downarrow \llbracket y \rrbracket_{y \rightarrow x_2}) \\ = & \lambda x_1. \downarrow (\llbracket y \rrbracket_{y \rightarrow \llbracket x \rrbracket_{x \rightarrow x_1}} \cdot \llbracket I \rrbracket) & = & \lambda x_1. x_1 (\lambda x_2. \downarrow x_2) \\ = & \lambda x_1. \downarrow (\llbracket x \rrbracket_{x \rightarrow x_1} \cdot \llbracket I \rrbracket) & = & \lambda x_1. x_1 (\lambda x_2. x_2) \end{aligned}$$

Reification (Simply-Typed)

- Given a type and a value of this type, produce a term.
- Context Γ records types of free variables.
- Inductively defined relation $\Gamma \vdash d \searrow v \uparrow A$.
- “In context Γ , value d reifies to term v at type A .”

$$\frac{\Gamma, x:A \vdash d \cdot x \searrow v \uparrow B}{\Gamma \vdash d \searrow \lambda xv \uparrow A \rightarrow B}$$

$$\frac{\Gamma \vdash d_i \searrow v_i \uparrow A_i \text{ for all } i}{\Gamma \vdash x \vec{d} \searrow x \vec{v} \uparrow *}$$
$$\Gamma(x) = \vec{A} \rightarrow *$$

- Inputs: Γ, d, A
- Output: v (β -normal η -long).

Reification (Step by Step)

- Reifying neutral values step by step:

$\Gamma \vdash e \searrow u \Downarrow A$ e reifies to u , inferring type A .

- Inputs: Γ , e (neutral value).
- Outputs: u (neutral β -normal η -long), A .
- Rules:

$$\frac{}{\Gamma \vdash x \searrow x \Downarrow \Gamma(x)}$$
$$\frac{\Gamma \vdash e \searrow u \Downarrow A \rightarrow B \quad \Gamma \vdash d \searrow v \Uparrow A}{\Gamma \vdash ed \searrow uv \Downarrow B}$$
$$\frac{\Gamma \vdash e \searrow u \Downarrow *}{\Gamma \vdash e \searrow u \Uparrow *}$$



Normalization by Evaluation

- Compose evaluation with reification:

$$\text{nbe}_A(t) = \text{the } v \text{ with } \vdash (t)_{\rho_{\text{id}}} \searrow v \uparrow A$$

- Completeness: NbE returns identical normal forms for all $\beta\eta$ -equal terms of the same type.

$$\text{If } \Gamma \vdash t = t' : A \text{ then } \Gamma \vdash (t)_{\rho_{\text{id}}} \searrow v \uparrow A \text{ and } \\ \Gamma \vdash (t')_{\rho_{\text{id}}} \searrow v \uparrow A.$$

- Soundness: NbE does not identify too many terms. The returned normal form is $\beta\eta$ -equal to the original term.

$$\text{If } \Gamma \vdash t : A \text{ then } \Gamma \vdash (t)_{\rho_{\text{id}}} \searrow v \uparrow A \text{ and } \Gamma \vdash t = v : A.$$

- Both proven by Kripke logical relations.

A Logical Relation for Soundness

- A Kripke logical relation $\mathcal{A} \in \mathbb{K}^A$ of type A is a map from contexts Γ to relations between values and terms of type A :

$$(\Gamma \in \text{Cxt}) \rightarrow \mathcal{P}(\mathcal{D} \times \text{Tm}_\Gamma^A)$$

- Monotonicity: extending Γ increases the relation.
- For each type A , define KLRs $\underline{A}, \overline{A}$ by

$$\begin{aligned}\overline{A}_\Gamma &= \{(d, t) \mid \Gamma \vdash d \searrow v \uparrow A \text{ and } \Gamma \vdash t = v : A \text{ for some } v\} \\ \underline{A}_\Gamma &= \{(e, t) \mid \Gamma \vdash e \searrow v \downarrow A \text{ and } \Gamma \vdash t = v : A \text{ for some } v\}\end{aligned}$$

- Soundness: If $\Gamma \vdash t : A$ then $((t)_{\rho_{\text{id}}}, t) \in \overline{A}_\Gamma$.
- Define KLR $\llbracket A \rrbracket \subseteq \overline{A}$ and show $((t)_{\rho_{\text{id}}}, t) \in \llbracket A \rrbracket_\Gamma$ (fundamental theorem).

Candidate Space

- Function space: given $\mathcal{A} \in \mathbb{K}^A$ and $\mathcal{B} \in \mathbb{K}^B$, define

$$(\mathcal{A} \Rightarrow \mathcal{B})_{\Gamma} = \{(f, r) \in D \times \text{Tm}_{\Gamma}^{A \rightarrow B} \mid (f \cdot d, r s) \in \mathcal{B}_{\Gamma'} \text{ if } \Gamma' \text{ extends } \Gamma \text{ and } (d, s) \in \mathcal{A}_{\Gamma'}\}$$

- $\underline{A}, \overline{A}$ form an *candidate space*, i. e.:

$$\begin{array}{ccc} * & \subseteq & * \\ \underline{A} \Rightarrow \overline{B} & \subseteq & \overline{A \rightarrow B} \\ \underline{A \rightarrow B} & \subseteq & \overline{A} \Rightarrow \underline{B} \end{array}$$

- We say $A \Vdash \mathcal{A}$ (A realizes \mathcal{A} , or \mathcal{A} is a candidate for A) if $\underline{A} \subseteq \mathcal{A} \subseteq \overline{A}$.

Justification of candidate space

- Law $\underline{*} \subseteq \overline{*}$

$$\frac{\Gamma \vdash e \searrow u \Downarrow *}{\Gamma \vdash e \searrow u \Uparrow *}$$

- Law $\underline{A} \Rightarrow \overline{B} \subseteq \overline{A \rightarrow B}$

$$\frac{\Gamma, x:A \vdash d \cdot x \searrow v \Uparrow B}{\Gamma \vdash d \searrow \lambda x v \Uparrow A \rightarrow B}$$

- Law $\underline{A \rightarrow B} \subseteq \overline{\underline{A} \Rightarrow \underline{B}}$

$$\frac{\Gamma \vdash e \searrow u \Downarrow A \rightarrow B \quad \Gamma \vdash d \searrow v \Uparrow A}{\Gamma \vdash ed \searrow uv \Downarrow B}$$

Justification of candidate space II

- Let \overline{A} the weakly normalizing terms of type A .
- Let \underline{A} the w.n. terms of shape $x s_1 \dots s_n$ of type A .
- Law $\underline{*} \subseteq \overline{*}$

$$\underline{A} \subseteq \overline{A}$$

- Law $\underline{A} \Rightarrow \overline{B} \subseteq \overline{A \rightarrow B}$

$$r x \in \overline{B} \text{ implies } r \in \overline{A \rightarrow B}$$

- Law $\underline{A \rightarrow B} \subseteq \overline{A} \Rightarrow \underline{B}$

$$r \in \underline{A \rightarrow B} \text{ and } s \in \overline{A} \text{ imply } r s \in \underline{B}$$

Type interpretation

- Define $\llbracket A \rrbracket$ by induction on A .

$$\begin{aligned}\llbracket * \rrbracket &= \bar{*} \\ \llbracket A \rightarrow B \rrbracket &= \llbracket A \rrbracket \Rightarrow \llbracket B \rrbracket\end{aligned}$$

- Theorem: $A \Vdash \llbracket A \rrbracket$.
- Now, the fundamental theorem implies soundness of NbE.
- Completeness by a similar logical relation.

What Have We Got?

- Abstractions in our proof:
 - 1 Applicative structures abstract over values and β .
 - 2 Fundamental theorem in a general form.
 - 3 Candidate spaces abstract over “good” semantical types. (*New!*)
- Other instances for \underline{A} , \overline{A} yield traditional weak $\beta(\eta)$ -normalization.
- Readily adapts to System F.

Scaling to System F

- Extending the notion of candidate space:

$$\begin{aligned}\overline{A[X/Y]} &\subseteq \overline{\forall YA} && \text{for a new } X \\ \overline{\forall YA} &\subseteq \overline{A[B/Y]} && \text{for any } B\end{aligned}$$

- Extending type interpretation:

$$\begin{aligned}[[X]]_{\rho} &= \rho(X) \\ [[A \rightarrow B]]_{\rho} &= [[A]]_{\rho} \rightarrow [[B]]_{\rho} \\ [[\forall XA]]_{\rho} &= \bigcap_{B \Vdash B} [[A]]_{\rho[X \mapsto B]}\end{aligned}$$

- Extending applicative structures, reification... (unproblematic).

Church-Style System F

- Terms and Typing

$$\overline{\Gamma \vdash x : \Gamma(x)}$$

$$\frac{\Gamma, x:A \vdash t : B}{\Gamma \vdash \lambda x:A. t : A \rightarrow B}$$

$$\frac{\Gamma \vdash r : A \rightarrow B \quad \Gamma \vdash s : A}{\Gamma \vdash r s : B}$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \Lambda X t : \forall X A} \quad X \notin \text{FV}(\Gamma)$$

$$\frac{\Gamma \vdash t : \forall X A}{\Gamma \vdash t B : A[B/X]}$$

Judgemental Equality for System F

- The typed equational theory of System F is induced by

$$\frac{\Gamma, x:A \vdash t : B \quad \Gamma \vdash s : A}{\Gamma \vdash (\lambda x:A. t) s = t[s/x] : B}$$

$$\frac{\Gamma \vdash t : A \rightarrow B}{\Gamma \vdash \lambda x:A. tx = t : A \rightarrow B} \quad x \notin \text{FV}(t)$$

$$\frac{\Gamma \vdash t : A \quad X \notin \text{FV}(\Gamma)}{\Gamma \vdash (\Lambda X t) B = t[B/X] : A[B/X]}$$

$$\frac{\Gamma \vdash t : \forall X A}{\Gamma \vdash \Lambda X. t X = t : \forall X A} \quad X \notin \text{FV}(t)$$

Evaluation

- We assume an evaluation function $\langle - \rangle_\eta \in \mathbf{Tm} \rightarrow \mathbf{D}$, satisfying

$$\begin{aligned}
 \langle x \rangle_\eta &= \eta(x) \\
 \langle r \ s \rangle_\eta &= \langle r \rangle_\eta \cdot \langle s \rangle_\eta \\
 \langle r \ A \rangle_\eta &= \langle r \rangle_\eta \cdot A\eta \\
 \langle \lambda x : A. t \rangle_\eta \cdot d &= \langle t \rangle_{\eta[x \mapsto d]} \\
 \langle \Lambda X t \rangle_\eta \cdot A &= \langle t \rangle_{\eta[X \mapsto A]} \\
 \langle t[s/x] \rangle_\eta &= \langle t \rangle_{\eta[x \mapsto \langle s \rangle_\eta]} \\
 \langle t[A/x] \rangle_\eta &= \langle t \rangle_{\eta[x \mapsto A\eta]} \\
 \langle t \rangle_\eta &= \langle t \rangle_{\eta'} \quad \text{if } \eta(x) = \eta'(x) \text{ for all } x \in \text{FV}(t)
 \end{aligned}$$

Contextual reification

- We can read back values as terms; this is called reification.

$$\begin{array}{l} \Gamma \vdash d \searrow t \uparrow A \\ \Gamma \vdash d \searrow t \downarrow A \end{array} \quad \begin{array}{l} d \text{ reifies to } t \text{ at type } A, \\ d \text{ reifies to } t, \text{ inferring type } A. \end{array}$$

- Rules:

$$\frac{}{\Gamma \vdash x \searrow x \downarrow \Gamma(x)} \quad \frac{\Gamma \vdash e \searrow r \downarrow A \rightarrow B \quad \Gamma \vdash d \searrow s \uparrow A}{\Gamma \vdash ed \searrow rs \downarrow B}$$

$$\frac{\Gamma \vdash e \searrow r \downarrow \forall X A}{\Gamma \vdash eB \searrow rB \downarrow A[B/X]} \quad \frac{\Gamma \vdash e \searrow r \downarrow X}{\Gamma \vdash e \searrow r \uparrow X}$$

$$\frac{\Gamma, x:A \vdash f \cdot x \searrow t \uparrow B}{\Gamma \vdash f \searrow \lambda x:A. t \uparrow A \rightarrow B} \quad \frac{\Gamma \vdash F \cdot X \searrow t \uparrow A}{\Gamma \vdash F \searrow \Lambda X t \uparrow \forall X A}$$

Candidate space

- For each type A , define KLRs $\underline{A}, \overline{A}$ by

$$\begin{aligned}\overline{A}_\Gamma &= \{(d, t) \mid \Gamma \vdash d \searrow v \uparrow A \text{ and } \Gamma \vdash t = v : A \text{ for some } v\} \\ \underline{A}_\Gamma &= \{(e, t) \mid \Gamma \vdash e \searrow v \downarrow A \text{ and } \Gamma \vdash t = v : A \text{ for some } v\}\end{aligned}$$

- $\underline{A}, \overline{A}$ form an *candidate space* fulfilling the conditions

$$\begin{aligned}\underline{A \rightarrow B} &\subseteq \overline{A} \rightarrow \underline{B} \\ \underline{A} \rightarrow \overline{B} &\subseteq \overline{A \rightarrow B} \\ \underline{\forall Y A} &\subseteq \underline{A[B/Y]} \quad \text{for any } B \\ \overline{A[X/Y]} &\subseteq \overline{\forall Y A} \quad \text{for a new } X\end{aligned}$$

Type interpretation

- We interpret quantification by an intersection which is indexed only by the *realizable* semantic types.

$$\begin{aligned} \llbracket X \rrbracket_\rho &= \rho(X) \\ \llbracket A \rightarrow B \rrbracket_\rho &= \llbracket A \rrbracket_\rho \rightarrow \llbracket B \rrbracket_\rho \\ \llbracket \forall X A \rrbracket_\rho &= \bigcap_{B \Vdash B} \llbracket A \rrbracket_{\rho[X \mapsto B]} \end{aligned}$$

- Types realize their interpretation: If $\sigma(X) \Vdash \rho(X)$ for all X , then $A\sigma \Vdash \llbracket A \rrbracket_\rho$.
- Proof: Induction on A , using the closure conditions of the candidate space.

Soundness of NbE for System F

- Now, prove the fundamental theorem for System F.
- Let $\sigma(X) \Vdash \eta(X)$ for all X .
 If $\Gamma \vdash t : A$ and $(\eta(x), \sigma(x)) \in \llbracket \Gamma(x) \rrbracket_\eta$ for all x then
 $((t)_\eta, t\sigma) \in \llbracket A \rrbracket_\eta$.
- As before, this entails soundness.

Related Work

- Altenkirch, Hofmann, and Streicher (1997) describe another version of NbE for System F.
- Each type is interpreted by a syntactical type A , a semantical type \mathcal{A} , and a normalization function nf^A for terms of type A .
- Construction carried out in category theory.
- Other work on NbE: Martin-Löf, Schwichtenberg, Berger, Danvy, Filinski, Dybjer, Scott, Aehlig, Joachimski, Coquand, and many more.

Conclusions

- This work: NbE for System F with conventional means.
- Follows the structure of a weak normalization proof.
- Variation of Girard's scheme.
- Future work: scale to the Calculus of Constructions.

Acknowledgments: This work was carried out during a visit to Frédéric Blanqui and Cody Roux at LORIA, Nancy, France, financed by the Bayerisch-Französisches Hochschulzentrum.

Thanks to Christophe and Pierre for the invitation.