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?

$$
\begin{tikzcd}
 & \text{Semantics (Values)} \arrow[dr, "\text{reify } \_ \searrow \_"] & \\
\text{Syntax (Terms)} \arrow[ur, "\text{eval } (\!|\_|\!)_{\_}"] & \supset & \text{Normal Forms}
\end{tikzcd}
$$

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

What is Normalization By Evaluation?

$$
\begin{tikzcd}
 & \text{Semantics (Values)} \arrow[dr, "\text{reify } \_ \searrow \_"] & \\
\text{Syntax (Terms)} \arrow[ur, "\text{eval } (\!|\_|\!)_{\_}"] & \supset & \text{Normal Forms}
\end{tikzcd}
$$

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

What is Normalization By Evaluation?

$$\begin{tikzcd} & \text{Semantics (Values)} \arrow[dr, "\text{reify } \_ \searrow \_"] & \\ \text{Syntax (Terms)} \arrow[ur, "\text{eval } (\!|\_|\!)_{\_}"] & \supset & \text{Normal Forms} \end{tikzcd}$$

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

What is Normalization By Evaluation?

$$
\begin{tikzcd}
 & \text{Semantics (Values)} \arrow[dr, "\text{reify } \_ \searrow \_"] & \\
\text{Syntax (Terms)} \arrow[ur, "\text{eval } (\!|\_|\!)_{\_}"] & \supset & \text{Normal Forms}
\end{tikzcd}
$$

- You have: an interpreter ($(\!|\_|\!)_{\_}$).
- You buy: my reifyer ($\_ \searrow \_$).
- 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!

$$
\begin{array}{lcl}
\mathrm{reify}\,(f) & = & \lambda x.\,\mathrm{reify}(f\,(x)) \\
\mathrm{reify}\,(x\,\vec{d}) & = & x\,\mathrm{reify}(\vec{d})
\end{array}
$$

- 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 $\_ \cdot \_ : \mathsf{D} \times \mathsf{D} \to \mathsf{D}$.
- Interpretation $(\!|t|\!)_\eta \in \mathsf{D}$ for term t and environment η, satisfying:

$$
\begin{array}{rcl}
(\!|x|\!)_\eta & = & \eta(x) \\
(\!|r\ s|\!)_\eta & = & (\!|r|\!)_\eta \cdot (\!|s|\!)_\eta \\
(\!|\lambda x t|\!)_\eta \cdot d & = & (\!|t|\!)_{\eta[x \mapsto d]}
\end{array}
$$

Simple examples:

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

Applicative Structures with Variables

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

$$
\begin{array}{rl}
 & \downarrow [\![\lambda x.\, I\, x\, I]\!] \\
= & \lambda x_1.\, \downarrow ([\![\lambda x.\, I\, x\, I]\!] \cdot x_1) \\
= & \lambda x_1.\, \downarrow ([\![I\, x\, I]\!]_{x \mapsto x_1}) \\
= & \lambda x_1.\, \downarrow ([\![I]\!] \cdot [\![x]\!]_{x \mapsto x_1} \cdot [\![I]\!]) \\
= & \lambda x_1.\, \downarrow ([\![y]\!]_{y \mapsto [\![x]\!]_{x \mapsto x_1}} \cdot [\![I]\!]) \\
= & \lambda x_1.\, \downarrow ([\![x]\!]_{x \mapsto x_1} \cdot [\![I]\!])
\end{array}
$$

$$
\begin{array}{rl}
= & \lambda x_1.\, \downarrow (x_1 \cdot [\![I]\!]) \\
= & \lambda x_1.\, x_1\, (\downarrow [\![I]\!]) \\
= & \lambda x_1.\, x_1\, (\lambda x_2.\, \downarrow ([\![I]\!] \cdot x_2)) \\
= & \lambda x_1.\, x_1\, (\lambda x_2.\, \downarrow [\![y]\!]_{y \mapsto x_2}) \\
= & \lambda x_1.\, x_1\, (\lambda x_2.\, \downarrow x_2) \\
= & \lambda x_1.\, x_1\, (\lambda x_2.\, x_2)
\end{array}
$$

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."

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

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

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

Reification (Step by Step)

- Reifying neutral values step by step:

$$\Gamma \vdash e \searrow u \Downarrow A \qquad e \text{ reifies to } u \text{, inferring type } A.$$

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

$$\frac{}{\Gamma \vdash x \searrow x \Downarrow \Gamma(x)} \qquad \frac{\Gamma \vdash e \searrow u \Downarrow A \to B \qquad \Gamma \vdash d \searrow v \Uparrow A}{\Gamma \vdash e\, d \searrow u\, v \Downarrow B}$$

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

-

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.

  If $\Gamma \vdash t = t' : A$ *then* $\Gamma \vdash (\!|t|\!)_{\rho_{\text{id}}} \searrow v \Uparrow A$ *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.

  If $\Gamma \vdash t : A$ *then* $\Gamma \vdash (\!|t|\!)_{\rho_{\text{id}}} \searrow v \Uparrow A$ *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:

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

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

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

- Soundness: If $\Gamma \vdash t : A$ then $((\!|t|\!)_{\rho_{\mathsf{id}}}, t) \in \overline{A}_\Gamma$.
- Define KLR $[\![A]\!] \subseteq \overline{A}$ and show $((\!|t|\!)_{\rho_{\mathsf{id}}}, t) \in [\![A]\!]_\Gamma$ (fundamental theorem).

Candidate Space

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

$$\begin{array}{rcl}(\mathcal{A} \Rightarrow \mathcal{B})_\Gamma & = & \{(f, r) \in \mathsf{D} \times \mathsf{Tm}_\Gamma^{A \to B} \mid (f \cdot d, r\, s) \in \mathcal{B}_{\Gamma'} \\ & & \quad \text{if } \Gamma' \text{ extends } \Gamma \text{ and } (d, s) \in \mathcal{A}_{\Gamma'}\}\end{array}$$

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

$$\begin{array}{rcl}\underline{*} & \subseteq & \overline{*} \\ \underline{A} \Rightarrow \overline{B} & \subseteq & \overline{A \to B} \\ \underline{A \to B} & \subseteq & \overline{A} \Rightarrow \underline{B}\end{array}$$

- 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 $\underline{A} \Rightarrow \overline{B} \subseteq \overline{A \to B}$

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

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

$$\frac{\Gamma \vdash e \searrow u \Downarrow A \to B \qquad \Gamma \vdash d \searrow v \Uparrow A}{\Gamma \vdash e\, d \searrow u\, v \Downarrow B}$$

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 $\underline{A} \Rightarrow \overline{B} \subseteq \overline{A \to B}$

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

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

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

Type interpretation

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

$$\begin{array}{lcl} \llbracket * \rrbracket & = & \overline{*} \\ \llbracket A \to B \rrbracket & = & \llbracket A \rrbracket \Rightarrow \llbracket B \rrbracket \end{array}$$

- 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:

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

- Extending type interpretation:

$$\begin{array}{rcl} [\![X]\!]_\rho & = & \rho(X) \\ [\![A \to B]\!]_\rho & = & [\![A]\!]_\rho \to [\![B]\!]_\rho \\ [\![\forall X A]\!]_\rho & = & \bigcap_{B \Vdash \mathcal{B}} [\![A]\!]_{\rho[X \mapsto \mathcal{B}]} \end{array}$$

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

Church-Style System F

- Terms and Typing

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

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

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \Lambda X t : \forall X A}\ X \notin \mathrm{FV}(\Gamma) \qquad \frac{\Gamma \vdash t : \forall X A}{\Gamma \vdash t\, B : A[B/X]}$$

Judgemental Equality for System F

- The typed equational theory of System F is induced by

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

$$\frac{\Gamma \vdash t : A \to B}{\Gamma \vdash \lambda x\!:\!A.\,t\,x = t : A \to B}\; x \notin \mathrm{FV}(t)$$

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

$$\frac{\Gamma \vdash t : \forall X A}{\Gamma \vdash \Lambda X.\,t\,X = t : \forall X A}\; X \notin \mathrm{FV}(t)$$

Evaluation

- We assume an evaluation function $(\!|-|\!)_\eta \in \mathsf{Tm} \to \mathsf{D}$, satisfying

$$\begin{array}{rcll}
(\!|x|\!)_\eta & = & \eta(x) & \\
(\!|r\,s|\!)_\eta & = & (\!|r|\!)_\eta \cdot (\!|s|\!)_\eta & \\
(\!|r\,A|\!)_\eta & = & (\!|r|\!)_\eta \cdot A\eta & \\
(\!|\lambda x\!:\!A.\,t|\!)_\eta \cdot d & = & (\!|t|\!)_{\eta[x \mapsto d]} & \\
(\!|\Lambda X t|\!)_\eta \cdot A & = & (\!|t|\!)_{\eta[X \mapsto A]} & \\
(\!|t[s/x]|\!)_\eta & = & (\!|t|\!)_{\eta[x \mapsto (\!|s|\!)_\eta]} & \\
(\!|t[A/x]|\!)_\eta & = & (\!|t|\!)_{\eta[x \mapsto A\eta]} & \\
(\!|t|\!)_\eta & = & (\!|t|\!)_{\eta'} & \text{if } \eta(x) = \eta'(x) \text{ for all } x \in \mathsf{FV}(t)
\end{array}$$

Contextual reification

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

$\Gamma \vdash d \searrow t \Uparrow A$ — d reifies to t at type A,

$\Gamma \vdash d \searrow t \Downarrow A$ — d reifies to t, inferring type A.

- Rules:

$$\frac{}{\Gamma \vdash x \searrow x \Downarrow \Gamma(x)} \qquad \frac{\Gamma \vdash e \searrow r \Downarrow A \to B \qquad \Gamma \vdash d \searrow s \Uparrow A}{\Gamma \vdash e\, d \searrow r\, s \Downarrow B}$$

$$\frac{\Gamma \vdash e \searrow r \Downarrow \forall X A}{\Gamma \vdash e\, B \searrow r\, B \Downarrow A[B/X]} \qquad \frac{\Gamma \vdash e \searrow r \Downarrow X}{\Gamma \vdash e \searrow r \Uparrow X}$$

$$\frac{\Gamma, x{:}A \vdash f \cdot x \searrow t \Uparrow B}{\Gamma \vdash f \searrow \lambda x{:}A.\, t \Uparrow A \to B} \qquad \frac{\Gamma \vdash F \cdot X \searrow t \Uparrow A}{\Gamma \vdash F \searrow \Lambda X t \Uparrow \forall X A}$$

Candidate space

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

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

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

$$
\begin{array}{rcll}
\underline{A \to B} & \subseteq & \overline{A} \to \underline{B} & \\
\underline{A} \to \overline{B} & \subseteq & \overline{A \to B} & \\
\underline{\forall Y A} & \subseteq & \underline{A[B/Y]} & \text{for any } B \\
\overline{A[X/Y]} & \subseteq & \overline{\forall Y A} & \text{for a new } X
\end{array}
$$

Type interpretation

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

$$\begin{array}{rcl} [\![X]\!]_\rho & = & \rho(X) \\ [\![A \to B]\!]_\rho & = & [\![A]\!]_\rho \to [\![B]\!]_\rho \\ [\![\forall X A]\!]_\rho & = & \bigcap_{B \Vdash \mathcal{B}} [\![A]\!]_{\rho[X \mapsto \mathcal{B}]} \end{array}$$

- Types realize their interpretation: If $\sigma(X) \Vdash \rho(X)$ for all X, then $A\sigma \Vdash [\![A]\!]_\rho$.
- 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* $(\eta(x), \sigma(x)) \in [\![\Gamma(x)]\!]_\eta$ *for all* x *then* $((\!|t|\!)_\eta, t\sigma) \in [\![A]\!]_\eta$.

- 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.