Weak Normalization for the Simply-Typed Lambda-Calculus in Twelf

A Case Study on Higher-Order Abstract Syntax

Andreas Abel

Twelf

- Logical framework based on the Edinburgh LF (dependently-typed $\lambda$-calculus)
- Propositions-as-types, derivations-as-objects
- Higher-order abstract syntax (HOAS)
- No internal recursion or induction
- Higher-order logic programming
- Applications:
  - Prototyping of logics and programming languages
  - Verification of syntactic properties (e.g., Church-Rosser, subject reduction, cut elimination)
  - Type-checking dependent types (Appel, Foundational PCC; Stump, SVC)
Twelf Syntax

- Kinds, types and terms.
  \[ K ::= \text{type} \quad \text{kind of types} \]
  \[ | \{X:A\}K \quad \text{dependent function kind} \]
  \[ A ::= F M_1 \ldots M_n \quad \text{base type (user-def.)} \]
  \[ | \{X:A\}A \quad \text{dependent function type} \]
  \[ | A \rightarrow A \quad \text{non-dependent function type} \]

- Terms considered upto \(\beta\eta\)-equality
- No user-def. reduction rules: all functions parametrics

Representation of Syntactic Objects in Twelf

- Representation of simple types \(A, B, C ::= * | A \rightarrow B\).

\[
\begin{array}{ll}
ty & : \text{type}.
\star & : \text{ty}.
=> & : \text{ty} \rightarrow \text{ty} \rightarrow \text{ty}.
\end{array}
\]

- Representation of \(\lambda\)-terms \(r, s, t, u ::= x | \lambda x.t | r.s\).

\[
\begin{array}{ll}
tm & : \text{type}.
lam & : (\text{tm} \rightarrow \text{tm}) \rightarrow \text{tm}.
app & : \text{tm} \rightarrow \text{tm} \rightarrow \text{tm}.
\end{array}
\]
• HOAS = represent object variables by framework variables.

\[ \text{twice} = \text{lam} \ [f:tm] \ \text{lam} \ [x:tm] \ \text{app} \ f \ (\text{app} \ f \ x). \]

Slide 5

Representation of Judgements without Hypotheses

• Weak head reduction \( t \rightarrow_w t' \).

\[
\begin{align*}
(\lambda x. t) \ s & \rightarrow_w [s/x] t \\
\text{beta} & \\
\frac{r \rightarrow_w r'}{r \ s \rightarrow_w r' \ s} \text{ appl}
\end{align*}
\]

Slide 6

• Representation in Twelf.

\[
\begin{align*}
\rightarrow_w : \ & \text{tm} \rightarrow \text{tm} \rightarrow \text{type}. \\
\text{beta} : \ & \text{app} \ (\lambda x. T) \ S \rightarrow_w T \ S. \\
\text{appl} : \ & R \rightarrow_w R' \rightarrow \text{app} \ R \ S \rightarrow_w \text{app} \ R' \ S.
\end{align*}
\]

• Substitution in object theory is application of the framework.
Representation of Judgements with Hypotheses

- Type assignment, natural-deduction style.

\[
\begin{align*}
\overline{x: A} & \quad \overline{r: A \rightarrow B} & \quad \overline{s: A} \\
\vdots & \quad \vdots & \quad \vdots \\
t: B & \quad rs: B \\
\lambda x.t: A \rightarrow B & \quad \text{of}_\text{lam}
\end{align*}
\]

Slide 7

- Typing assumption is represented as hypothetical judgement.

\[
\begin{align*}
of & : \text{tm} \rightarrow \text{ty} \rightarrow \text{type.} \\
of_\text{lam} : ((x: \text{tm}) \ of \ A \rightarrow \ of \ (T \ x) \ B) \\
& \rightarrow \ of \ (\text{lam} \ [x: \text{tm}] \ T \ x) \ (A \Rightarrow B). \\
of_\text{app} : \ of \ R \ (A \Rightarrow B) \rightarrow \ of \ S \ A \rightarrow \ of \ (\text{app} \ R \ S) \ B.
\end{align*}
\]

Weak Head Reduction is Closed under Substitution

- Lemma: If \( t \xrightarrow{\text{D}}_w t' \) then \([u/y] t \xrightarrow{\text{D}'}_w [u/y] t'\).
- Proof: By induction on the derivation \( \text{D} \) of \( t \xrightarrow{\cdot}_w t' \).

Slide 8

- Case \( (\lambda x.t) s \xrightarrow{\text{beta}}_w [s/x] t \). W.l.o.g. \( x \neq y \) and \( x \) not free in \( u \).

Then,

\[
\begin{align*}
[u/y]((\lambda x.t) s) &= (\lambda x.[u/y] t) [u/y] s \\
&\xrightarrow{\text{beta}}_w [[u/y]s/x][u/y] t = [u/y][s/x] t.
\end{align*}
\]

- Case \( r s \xrightarrow{\text{appl} \text{D}}_w r' s \) with \( r \xrightarrow{\text{D}}_w r' \). By ind. hyp.,

\[
[u/y]r \xrightarrow{\text{D}'}_w [u/y] r'. \text{ Hence,}
\]

\[
[u/y](r s) = ([u/y]r) ([u/y] s) \\
\xrightarrow{\text{appl} \text{D}'}_w ([u/y] r') ([u/y] s) = [u/y](r' s)
\]
Representation of Theorems and Proofs

- A theorem is represented as a functional relation.

  \[
  \text{subst_red} : \{U:\text{tm}\} \ (\{y:\text{tm}\} \ T \ y \rightarrow^w T' \ y) \\
  \rightarrow T \ U \rightarrow^w T' \ U \rightarrow \text{type.}
  \]

  \[\%\text{mode subst_red } +U \ +D \ -D'.\]

- Its proof is represented as a logic program which implements the relation.

  \[
  \text{subst_red_beta}: \text{subst_red} \ U \ ([y] \beta) \beta.
  \]

  \[
  \text{subst_red_appl}: \text{subst_red} \ U \ ([y] \text{appl} \ (D \ y)) \ (\text{appl} \ D').
  \]

  \[\%\text{terminates D } (\text{subst_red } _D \ _).\]

- Function must be total to represent a valid proof.
- This requires \textit{termination} and \textit{coverage} of all possible inputs.
A Formalized Proof of Weak Normalization for the STL

- Structure of a normalization proof:
  1. Define a relation \( t \Downarrow A \) which is closed under application.
  2. Show: If \( t : A \) then \( t \Downarrow A \).
  3. Show: If \( t \Downarrow A \) then \( t \) is normalizing.

- Tait and crowd: \( t \Downarrow A \) is a *logical relation* (semantical).
- Joachimski and Matthes (2004): \( t \Downarrow A \) is a *finitary inductive definition*.

Inductive Characterization of Weakly Normalizing Terms

- \( \Gamma \vdash t \Downarrow A \): *\( t \) is weakly normalizing of type \( A \).*
- \( \Gamma \vdash t \Downarrow^x A \): *\( t \) is wn and neutral of type \( A \).*
- Rules:

\[
\frac{(x:A) \in \Gamma}{\Gamma \vdash x \downarrow^x A} \quad \frac{\Gamma \vdash r \downarrow^x A \rightarrow B \quad \Gamma \vdash s \downarrow A}{\Gamma \vdash r \; s \downarrow B} \quad \text{wne_app}
\]

\[
\frac{\Gamma \vdash r \downarrow^x A}{\Gamma \vdash r \downarrow A} \quad \text{wn_ne}
\]

\[
\frac{\Gamma, x:A \vdash t \Downarrow B}{\Gamma \vdash \lambda x.t \Downarrow A \rightarrow B} \quad \text{wn_lam} \quad \frac{r \rightarrow^\omega r' \quad \Gamma \vdash r' \Downarrow A}{\Gamma \vdash r \Downarrow A} \quad \text{wn_exp}
\]
Difficult: Closure under Application

- Lemma: Let \( D : \Gamma \vdash s \downarrow A \).
  1. If \( E : \Gamma \vdash r \downarrow A \rightarrow C \) then \( \Gamma \vdash r \, s \downarrow C \).
  2. If \( E : \Gamma, x : A \vdash t \downarrow C \), then \( \Gamma \vdash [s/x]t \downarrow C \).
  3. If \( E : \Gamma, x : A \vdash t \downarrow^x C \), then \( \Gamma \vdash [s/x]t \downarrow C \)
      and \( C \) is a subexpression of \( A \).
  4. If \( E : \Gamma, x : A \vdash t \downarrow^y C \) with \( x \neq y \), then \( \Gamma \vdash [s/x]t \downarrow^y C \).

- Proof: Simultaneously by main induction on type \( A \) (for part 3) and side induction on the derivation \( E \).

- Similar to Girard, Lafont and Taylor (1989): Lexicographic induction on highest degree (=type) of a redex and the number of redexes of highest degree.

Closure under Application and Substitution in Twelf

- Representation of lemma as 4 type families.
- "\( C \) is a subexpression of \( A \)" expressed by \( \%\text{reduces} \ C \% A \).
- Mutual lexicographic termination order.
Soundness of Inductive Characterization

- Simple induction: \( t \Downarrow A \) for every typed term \( t : A \).
- Lemma (Soundness): If \( t \Downarrow A \) then \( t \rightarrow^* v \) for some \( v \).
- Requires characterization of valued and properties of reduction.
- Technical, but well understood.

Slide 15

Tait-Style Proofs in Twelf?

- Heart of Tait’s proof is the rule:

\[
\forall s. \ s \Downarrow A \Rightarrow r \ s \Downarrow B \quad \Rightarrow \quad r \Downarrow A \rightarrow B
\]

- Literal encoding in Twelf...

\[
\{S:tm\} \ \text{wn} \ S \ A \rightarrow \ \text{wn} \ (\text{app} \ R \ S) \ B \rightarrow \ \text{wn} \ R \ (A \Rightarrow B).
\]

- ...means something else:

if for a fresh term \( S \) for which we assume \( \text{wn} \ S \ A \) it holds that \( \text{wn} \ (\text{app} \ R \ S) \ B \), then \( \text{wn} \ R \ (A \Rightarrow B) \).

- Problem: Tait’s infinitary premise is not expressible.
Strong Normalization in Twelf?

- Classical definition of \textit{strongly normalizing}: no infinite reduction sequences.
- No good in a constructive setting.
- Inductive definition of \textit{strongly normalizing}: wellfounded part of reduction relation.
\[
\forall t'. t \rightarrow t' \Rightarrow \text{sn} \ t' \Rightarrow \text{sn} \ t.
\]
- Suffers likewise from an infinitary premise.

Conclusion

- Normalization for a proof-theoretically weak object theory directly implementable in Twelf.
- Limits for normalization proofs: expressiveness of Twelf, termination checker.
- Conjecture 1: Infinitary premises not expressible in Twelf.
- Conjecture 2: Strong normalization not expressible in Twelf.
- Conjecture 3: Proof-theoretical strength of Twelf bounded by arithmetic.
Related Work

• Altenkirch (1993): SN for System F in LEGO.
• Berghofer and Nipkow: Joachimski and Matthes’ proof in Isabelle. Submitted.
• Watkins and Crary: Normalization for Concurrent LF in Twelf.