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

A Case Study on Higher-Order Abstract Syntax

Andreas Abel

Slide 1

LFM'04, IJCAR
Cork, Ireland
July 5, 2004

Work supported by: GKLI (DFG), TYPES & APPSEM-II (EU), CoVer (SSF)

Twelf

Slide 2

- Logical framework based on the Edinburgh LF (dependently-typed λ-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.

$$
\begin{array}{lcll}
\mathbb{K} & ::= & \mathsf{type} & \text{kind of types} \\
 & \mid & \{\mathbb{X}:\mathbb{A}\}\mathbb{K} & \text{dependent function kind} \\
\mathbb{A} & ::= & \mathbb{F}\ \mathbb{M}_1 \ldots \mathbb{M}_n & \text{base type (user-def.)} \\
 & \mid & \{\mathbb{X}:\mathbb{A}\}\mathbb{A} & \text{dependent function type} \\
 & \mid & \mathbb{A} \rightarrow \mathbb{A} & \text{non-dependent function type} \\
\mathbb{M} & ::= & \mathbb{C} & \text{term constant (user-def.)} \\
 & \mid & \mathbb{X} & \text{term variable} \\
 & \mid & [\mathbb{X}:\mathbb{A}]\mathbb{M} & \text{term abstraction} \\
 & \mid & \mathbb{M}\,\mathbb{M} & \text{term application}
\end{array}
$$

Slide 3

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

Representation of Syntactic Objects in Twelf

- Representation of simple types $A, B, C ::= * \mid A \rightarrow B$.

```
ty      : type.
*       : ty.
=>      : ty -> ty -> ty.
```

Slide 4

- Representation of λ-terms $r, s, t, u ::= x \mid \lambda x.t \mid r\, s$.

```
tm      : type.
lam     : (tm -> tm) -> tm.
app     : tm -> tm -> tm.
```

- HOAS = represent object variables by framework variables.

```
twice = lam [f:tm] lam [x:tm] app f (app f x).
```

Slide 5

Representation of Judgements without Hypotheses

- Weak head reduction $t \longrightarrow_{\mathsf{w}} t'$.

$$\frac{}{(\lambda x.t)\, s \longrightarrow_{\mathsf{w}} [s/x]t}\ \texttt{beta} \qquad \frac{r \longrightarrow_{\mathsf{w}} r'}{r\, s \longrightarrow_{\mathsf{w}} r'\, s}\ \texttt{appl}$$

Slide 6

- Representation in Twelf.

```
-->w  : tm -> tm -> type.

beta  : app (lam T) S -->w T S.
appl  : R -->w R' -> app R S -->w app R' S.
```

- Substitution in object theory is application of the framework.

Representation of Judgements with Hypotheses

Slide 7

- Type assignment, natural-deduction style.

$$\dfrac{\begin{matrix}\overline{x : A}\\ \vdots \\ t : B\end{matrix}}{\lambda x.t : A \to B}\ \texttt{of\_lam} \qquad\qquad \dfrac{r : A \to B \qquad s : A}{r\,s : B}\ \texttt{of\_app}$$

- Typing assumption is represented as hypothetical judgement.

```
of     : tm -> ty -> type.
of_lam : ({x:tm} of x A -> of (T x) B)
            -> of (lam [x:tm] T x) (A => B).
of_app : of R (A => B) -> of S A -> of (app R S) B.
```

Weak Head Reduction is Closed under Substitution

Slide 8

- Lemma: If $t \xrightarrow{\texttt{D}}_{\mathsf{w}} t'$ then $[u/y]t \xrightarrow{\texttt{D'}}_{\mathsf{w}} [u/y]t'$.
- Proof: By induction on the derivation $\texttt{D}$ of $t \longrightarrow_{\mathsf{w}} t'$.
  - Case $(\lambda x.t)\,s \xrightarrow{\texttt{beta}}_{\mathsf{w}} [s/x]t$. W.l.o.g. $x \neq y$ and x not free in u. Then,

$$\begin{array}{rcl} [u/y]((\lambda x.t)\,s) & = & (\lambda x.[u/y]t)\ [u/y]s \\ & \xrightarrow{\texttt{beta}}_{\mathsf{w}} & [[u/y]s/x][u/y]t \quad = \quad [u/y][s/x]t. \end{array}$$

  - Case $r\,s \xrightarrow{\texttt{appl D}}_{\mathsf{w}} r'\,s$ with $r \xrightarrow{\texttt{D}}_{\mathsf{w}} r'$. By ind. hyp., $[u/y]r \xrightarrow{\texttt{D'}}_{\mathsf{w}} [u/y]r'$. Hence,

$$\begin{array}{rcl} [u/y](r\,s) & = & ([u/y]r)\ ([u/y]s) \\ & \xrightarrow{\texttt{appl D'}}_{\mathsf{w}} & ([u/y]r')\ ([u/y]s) \quad = \quad [u/y](r'\,s) \end{array}$$

Representation of Theorems and Proofs

- A theorem is represented as a functional relation.

```
subst_red : {U:tm} ({y:tm} T y -->w T' y)
                        -> T U -->w T' U -> type.
%mode subst_red +U +D -D'.
```

Slide 9

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

```
subst_red_beta: subst_red U ([y] beta) beta.
subst_red_appl: subst_red U ([y] appl (D y)) (appl D')
                <- subst_red U D D'.
%terminates D (subst_red _ D _).
```

- Function must be total to represent a valid proof.
- This requires *termination* and *coverage* of all possible inputs.

Slide 10

A Formalized Proof of Weak Normalization for the STL

Slide 11

- 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*.
- Forerunners: Goguen (1995), van Raamsdonk and Severi (1995).

Inductive Characterization of Weakly Normalizing Terms

Slide 12

- "De-vectorized" version of Joachimski and Matthes (2004)
- $\Gamma \vdash t \Downarrow A$: *$t$ is weakly normalizing of type A*.
- $\Gamma \vdash t \downarrow^x A$: *$t$ is wn and neutral of type A*.
- Rules:

$$\frac{(x\!:\!A) \in \Gamma}{\Gamma \vdash x \downarrow^x A} \qquad \frac{\Gamma \vdash r \downarrow^x A \to B \qquad \Gamma \vdash s \Downarrow A}{\Gamma \vdash r\, s \downarrow^x B}\ \texttt{wne\_app}$$

$$\frac{\Gamma \vdash r \downarrow^x A}{\Gamma \vdash r \Downarrow A}\ \texttt{wn\_ne}$$

$$\frac{\Gamma, x\!:\!A \vdash t \Downarrow B}{\Gamma \vdash \lambda x.t \Downarrow A \to B}\ \texttt{wn\_lam} \qquad \frac{r \longrightarrow_{\mathsf{w}} r' \qquad \Gamma \vdash r' \Downarrow A}{\Gamma \vdash r \Downarrow A}\ \texttt{wn\_exp}$$

Difficult: Closure under Application

Slide 13

- Lemma: Let $\mathcal{D} :: \Gamma \vdash s \Downarrow A$.
  1. If $\mathcal{E} :: \Gamma \vdash r \Downarrow A \to C$ then $\Gamma \vdash r\, s \Downarrow C$.
  2. If $\mathcal{E} :: \Gamma, x\!:\!A \vdash t \Downarrow C$, then $\Gamma \vdash [s/x]t \Downarrow C$.
  3. If $\mathcal{E} :: \Gamma, x\!:\!A \vdash t \downarrow^x C$, then $\Gamma \vdash [s/x]t \Downarrow C$ and C is a subexpression of A.
  4. If $\mathcal{E} :: \Gamma, x\!:\!A \vdash t \downarrow^y C$ 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 $\mathcal{E}$.
- 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

Slide 14

- Representation of lemma as 4 type families.
- "C is a subexpression of A" expressed by `%reduces C <= A`.
- Mutual lexicographic termination order.

Soundness of Inductive Characterization

Slide 15

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

Tait-Style Proofs in Twelf?

Slide 16

- Heart of Tait's proof is the rule:

$$\frac{\forall s.\ \ s \Downarrow A \ \Rightarrow\ r\,s \Downarrow B}{r \Downarrow A \to B}$$

- Literal encoding in Twelf...

```
({S:tm} wn S A -> wn (app R S) B) -> wn R (A => B).
```

- ... means something else:

  if for a fresh term `S` for which we assume `wn S A` it holds that `wn (app R S) B`, then `wn R (A => B)`.

- Problem: Tait's infinitary premise is not expressible.

Strong Normalization in Twelf?

Slide 17

- Classical definition of *strongly normalizing*: no infinite reduction sequences.
- No good in a constructive setting.
- Inductive definition of *strongly normalizing*: wellfounded part of reduction relation.

$$\frac{\forall t'.\ t \longrightarrow t' \Rightarrow \mathsf{sn}\, t'}{\mathsf{sn}\, t},$$

- Suffers likewise from an infinitary premise.

Conclusion

Slide 18

- 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

Slide 19

- Altenkirch (1993): SN for System F in LEGO.
- Filinski in 1990s: Feasibility of logical relations in Twelf. Not published.
- Berghofer and Nipkow: Joachimski and Matthes' proof in Isabelle. Submitted.
- Watkins and Crary: Normalization for Concurrent LF in Twelf.