Normalization by Evaluation for Martin-Löf Type Theory with Typed Equality Judgements

Andreas Abel[1]
Thierry Coquand[2] Peter Dybjer[2]

[1]Ludwig-Maximilians-University Munich
[2]Chalmers University of Technology

Logic in Computer Science
Wrocław, Poland
10 July 2007

My Talk

- Dependent type theory basis for theorem provers (functional programming languages) Agda, Coq, Epigram, ...
- Intensional theory with predicative universes.
- Judgemental $\beta\eta$-equality.
- Deciding type equality with Normalization-By-Evaluation.
- Semantic proof of decidability of typing.

Dependent Types

- Dependent function space:

$$\frac{r : \Pi x\!:\!A.\, B[x] \qquad s : A}{r\, s : B[s]}$$

- Types contain terms, type equality non-trivial.
- Shape of types can depend on terms:

$$\mathsf{Vec}\, A\, n = \underbrace{A \times \cdots \times A}_{n \text{ factors}}$$

- Type conversion rule:

$$\frac{t : A}{t : B}\; A \cong B$$

- Deciding type checking requires injectivity of Π

$$\Pi x\!:\!A.B \cong \Pi x\!:\!A'.B' \text{ implies } A \cong A' \text{ and } B \cong B'$$

Untyped β-Equality

- One solution: $A \cong B$ iff A, B have common β-reduct.
- Confluence of β makes $\cong$ transitive.
- Injectivity of Π trivial.
- But we want also η! E.g.
  - Theorem prover should not distinguish between $P\,(\lambda x.\, f\, x)$ and $P\, f$,
  - or between two inhabitants of a one-element type.
- The stronger the type equality, the more (sound) programs are accepted by the type checker.

Untyped $\beta\eta$-Equality

- Try: $A \cong B$ iff A, B have common $\beta\eta$-reduct.
- $\beta\eta$-reduction (with surjective pairing) only confluent on strongly normalizing terms
- Proof of s.n. requires model construction
- ... which requires invariance of interpretation under reduction
- ... which requires subject reduction
- ... which requires strengthening
- ... hard to prove for pure type systems (van Benthem 1993)
- Even for untyped β, model construction difficult: Miquel Werner 2002: The not so simple proof-irrelevant model of CC

Typed $\beta\eta$-Equality

- Introduce equality judgement $\vdash A = B$.
- Relies on term equality $\vdash t = t' : C$.
- Simplifies model construction considerably.
- Now injectivity of Π is hard.
- Goguen 1994: Typed Operational Semantics for UTT.
  - "Syntactical" model.
  - Shows confluence, subject reduction, normalization in one go.
  - Impressive, technically demanding work.
- This work: simpler argument, in the same spirit.
- Slogan: semantics proves properties of syntax. (Altenkirch 1994).

Deciding judgemental equality

Normalization function $\mathsf{nf}^A(t)$.

- Completeness:
  $\vdash t = t' : A$ implies $\mathsf{nf}^A(t) = \mathsf{nf}^A(t')$ (syntactical equal).
- Soundness:
  $\vdash t : A$ implies $\vdash t = \mathsf{nf}^A(t) : A$.

Syntax of Terms and Types

- Lambda-calculus with constants

$$
\begin{array}{lcll}
r, s, t & ::= & c \mid x \mid \lambda x.t \mid r\,s & \\
c & ::= & \mathsf{N} & \text{type of natural numbers} \\
 & & \mathsf{z} & \text{zero} \\
 & & \mathsf{s} & \text{successor} \\
 & & \mathsf{rec} & \text{primitive recursion} \\
 & & \mathsf{Fun} & \text{function space constructor} \\
 & & \mathsf{U} & \text{universe of small types}
\end{array}
$$

- $\Pi x\!:\!A.B$ is written $\mathsf{Fun}\,A\,(\lambda x.B)$.

Judgements

- Essential judgements

$\Gamma \vdash A$	A is a well-formed type in Γ
$\Gamma \vdash t : A$	t has type A in Γ
$\Gamma \vdash A = A'$	A and A' are equal types in Γ
$\Gamma \vdash t = t' : A$	t and t' are equal terms of type A in Γ

- Typing of functions:

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

Rules for Judgmental Equality

- Equality axioms:

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

$$(\eta)\ \frac{\Gamma \vdash t : \mathsf{Fun}\, A\, (\lambda x.B)}{\Gamma \vdash (\lambda x.\, t\, x) = t : \mathsf{Fun}\, A\, (\lambda x.B)}\ x \notin \mathsf{FV}(t)$$

- Computation axioms for primitive recursion.
- Congruence rules.

Small and Large Types

- Small types (sets):

$$\frac{}{\Gamma \vdash \mathsf{N} : \mathsf{U}} \qquad \frac{\Gamma \vdash A : \mathsf{U} \qquad \Gamma, x : A \vdash B : \mathsf{U}}{\Gamma \vdash \mathsf{Fun}\, A\, (\lambda x. B) : \mathsf{U}}$$

- U includes types defined by recursion like $\mathsf{Vec}\, A\, n$.
- (Large) types:

$$\frac{\Gamma \vdash A : \mathsf{U}}{\Gamma \vdash A} \qquad \frac{}{\Gamma \vdash \mathsf{U}} \qquad \frac{\Gamma \vdash A \qquad \Gamma, x : A \vdash B}{\Gamma \vdash \mathsf{Fun}\, A\, (\lambda x. B)}$$

λ-Model

- Consider a (total) combinatorial algebra D
- with constructors $\mathsf{N}, \mathsf{z}, \mathsf{s}, \mathsf{Fun}, \mathsf{U}$.
- Evaluation $[\![t]\!]_\rho$: Standard.

$$
\begin{array}{rcll}
[\![c]\!]_\rho & = & c & (c \text{ constant}) \\
[\![x]\!]_\rho & = & \rho(x) & \\
[\![r\,s]\!]_\rho & = & [\![r]\!]_\rho\ [\![s]\!]_\rho & \\
[\![\lambda x.t]\!]_\rho\ d & = & [\![t]\!]_{\rho[x \mapsto d]} &
\end{array}
$$

- Example: $[\![\mathsf{Fun}\,A\,(\lambda x.B)]\!] = \mathsf{Fun}\,X\,F$ where $X = [\![A]\!]$ and $F\,d = [\![B]\!]_{[x \mapsto d]}$.
- We enrich D with term variables:
- $\mathsf{Up}\,u \in \mathsf{D}$ for each neutral term $u ::= x\,\vec{v}$ (generalized variable).

Reification (Printing)

- Reification $\downarrow^{X} d$ produces a η-long β-normal term.

$$
\begin{array}{lcl}
\downarrow^{\mathsf{N}} \mathsf{z} & = & \mathsf{z} \\
\downarrow^{\mathsf{N}} (\mathsf{s}\, d) & = & \mathsf{s}\,(\downarrow^{\mathsf{N}} d) \\
\downarrow^{\mathsf{N}} (\mathsf{Up}\, u) & = & u \\
\downarrow^{\mathsf{Up}\, u'} (\mathsf{Up}\, u) & = & u \\
\downarrow^{\mathsf{Fun}\, X\, F} f & = & \lambda x. \downarrow^{F\,(\uparrow^{X} x)} (f\,(\uparrow^{X} x)), \quad x \text{ fresh}
\end{array}
$$

- Reflection $\uparrow^{X} u$ embeds a neutral term u into D, η-expanded.

$$
\begin{array}{lcl}
(\uparrow^{\mathsf{Fun}\, X\, F} u)\, d & = & \uparrow^{F\, d} (u \downarrow^{X} d) \\
\uparrow^{X} u & = & \mathsf{Up}\, u
\end{array}
$$

- Normalization of closed terms $\vdash t : A$

$$
\mathsf{nf}^{A}(t) = \downarrow^{[\![A]\!]} [\![t]\!].
$$

PER Model

- A PER is a symmetric and transitive relation on D.
- Small types: define a PER $\mathcal{U}$ and a PER $[X]$ for $X \in \mathcal{U}$.

$$\frac{}{\mathsf{N} = \mathsf{N} \in \mathcal{U}} \qquad \frac{}{\mathsf{z} = \mathsf{z} \in [\mathsf{N}]} \qquad \frac{d = d' \in [\mathsf{N}]}{\mathsf{s}\, d = \mathsf{s}\, d' \in [\mathsf{N}]} \qquad \frac{u \text{ neutral}}{\mathsf{Up}\, u = \mathsf{Up}\, u \in [\mathsf{N}]}$$

$$\frac{u \text{ neutral}}{\mathsf{Up}\, u = \mathsf{Up}\, u \in \mathcal{U}} \qquad \frac{u, u' \text{ neutral}}{\mathsf{Up}\, u' = \mathsf{Up}\, u' \in [\mathsf{Up}\, u]}$$

$$\frac{X = X' \in \mathcal{U} \qquad F\, d = F'\, d' \in \mathcal{U} \text{ for all } d = d' \in [X]}{\mathsf{Fun}\, X\, F = \mathsf{Fun}\, X'\, F' \in \mathcal{U}}$$

$$\frac{f\, d = f'\, d' \in [F\, d] \text{ for all } d = d' \in [X]}{f = f' \in [\mathsf{Fun}\, X\, F]}$$

Modelling Large Types

- Large types: Define PER $\mathcal{T}\mathit{ype}$ and extend $[\_]$ to $\mathcal{T}\mathit{ype}$.

$$\mathcal{U} \subseteq \mathcal{T}\mathit{ype}$$

$$\frac{X = X' \in \mathcal{T}\mathit{ype} \qquad F\,d = F'\,d' \in \mathcal{T}\mathit{ype} \text{ for all } d = d' \in [X]}{\mathsf{Fun}\,X\,F = \mathsf{Fun}\,X'\,F' \in \mathcal{T}\mathit{ype}}$$

$$\frac{}{\mathsf{U} = \mathsf{U} \in \mathcal{T}\mathit{ype}} \qquad [\mathsf{U}] = \mathcal{U}$$

- PERs contain only total elements of D.
- These can be printed (converted to terms).

Checking Semantic Equality

Lemma

Let $X = X' \in \mathcal{T}ype$.

1. $\uparrow^{X} u = \uparrow^{X'} u \in [X]$.
2. *If* $d = d' \in [X]$ *then* $\downarrow^{X} d =_{\alpha} \downarrow^{X'} d'$.

Proof.

Simultaneously by induction on $X = X' \in \mathcal{T}ype$. $\square$

Completeness of NbE

Theorem (Validity of judgements in PER model)

Let $\rho(x) = \rho'(x) \in [\![\Gamma(x)]\!]_\rho$ *for all* x.

- *If* $\Gamma \vdash t : A$ *then* $[\![t]\!]_\rho = [\![t]\!]_{\rho'} \in [[\![A]\!]_\rho]$.
- *If* $\Gamma \vdash t = t' : A$ *then* $[\![t]\!]_\rho = [\![t']\!]_{\rho'} \in [[\![A]\!]_\rho]$.

Corollary (Completeness of nf)

If $\vdash t = t' : A$ *then* $\mathsf{nf}^A(t) =_\alpha \mathsf{nf}^A(t')$.

Soundness remains: If $\vdash t : A$ then $\vdash t = \mathsf{nf}^A(t) : A$.

Kripke Logical Relation

Relate well-typed terms modulo equality to inhabitants of PERs.

Lemma (Into and out of the logical relation)

1. *If* $\Gamma \vdash r = u : C$ *then* $\Gamma \vdash r : C \circledR \uparrow^X u \in [X]$.
2. *If* $\Gamma \vdash r : C \circledR d \in [X]$ *then* $\Gamma \vdash r = \downarrow^X d : C$.

Definition

$$\Gamma \vdash r : C \circledR d \in [X] :\Longleftrightarrow \Gamma \vdash r = \downarrow^X d : C \qquad \text{for } X \text{ base type,}$$

$$\begin{array}{l} \Gamma \vdash r : C \circledR f \in [\mathsf{Fun}\, X\, F] :\Longleftrightarrow \\ \qquad \Gamma \vdash C = \mathsf{Fun}\, A\, (\lambda x.B) \text{ for some } A, B \text{ and} \\ \qquad \text{for all } \Delta \geq \Gamma \text{ and } \Delta \vdash s : A \circledR d \in [X], \\ \qquad\qquad \Delta \vdash r\, s : B[s/x] \circledR f\, d \in [F\, d]. \end{array}$$

Soundness of NbE

- Prove the fundamental theorem.
- Corollary: $\vdash t : A$ implies $\vdash t : A \circledR [\![t]\!] \in [[\![A]\!]]$.
- Escaping the log.rel.: $\vdash t = \downarrow^{[\![A]\!]}[\![t]\!] : A$.
- Hence, nf is also sound.
- Decidability of judgemental equality entails injectivity of Π.

Conclusion

- Semantic metatheory of Martin-Löf Type Theory.
- Inference rules directly justified by PER model.
- No need to prove strengthening, subject reduction, confluence, normalization.
- Future work:
  - Extend to Σ-types, singleton-types, proof-irrelevance.
  - Adopt to syntax of categories-with-families (de Bruijn indices and explicit substitutions).

Related Work

- Martin-Löf 1975: NbE for Type Theory (weak conversion)
- Martin-Löf 2004: Talk on NbE (philosophical justification)
- Danvy et al: Type-directed partial evaluation
- Altenkirch Hofmann Streicher 1996: NbE for λ-free System F
- Berger Eberl Schwichtenberg 2003: Term rewriting for NbE
- Aehlig Joachimski 2004: Untyped NbE, operationally
- Filinski Rohde 2004: Untyped NbE, denotationally
- Danielsson 2006: strongly typed NbE for LF
- Altenkirch Chapman 2007: Tait in one big step

Special thanks to Klaus Aehlig.