Explicit Convertibility Proofs in Pure Type Systems

Floris van Doorn[1] Herman Geuvers[2] Freek Wiedijk[2]

[1]Carnegie Mellon University

[2]Radboud University Nijmegen

LFMTP 2013

Pure Type Systems (1)

Pure Type Systems:

- represent a wide variety of type systems
- consist of sorts s, axioms (s_1, s_2) and relations (s_1, s_2, s_3)
- use dependent types

$$\frac{\Gamma \vdash A : s_1 \qquad \Gamma, x : A \vdash B : s_2}{\Gamma \vdash \Pi x{:}A.B : s_3} \; (s_1, s_2, s_3) \in \mathcal{R} \qquad \text{(prod)}$$

Pure Type Systems (2)

- Conversion rule

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash A' : s}{\Gamma \vdash a : A'} \, (A \simeq_\beta A') \qquad \text{(conv)}$$

- $\Gamma \vdash M : A \quad \Longrightarrow \quad M$ codes a proof for A
- Proof of $2 + 4 = 3 \cdot 2$ is the same as proof of $6 = 6$.

Motivation

In Pure Type Systems:

- computations are not part of the proof.
- the conclusion does not determine derivation
- the type of a term is only determined up to beta conversion

New version of PTS

PTS_f: Pure Type System with convertibility proofs

- Explicit proofs of computations
- Syntax directed
- The type of a term is determined up to alpha conversion

Terms

Terms:

$$\mathcal{T} = x \mid s \mid \Pi x{:}A.B \mid \lambda x{:}A.b \mid Fa \mid a^H$$

Terms

Terms:

$$\mathcal{T} = x \mid s \mid \Pi x{:}A.B \mid \lambda x{:}A.b \mid Fa \mid a^H$$

Convertibility Proofs:

$$\begin{aligned}\mathcal{H} = \beta(a) \mid \iota(a) & \quad \text{(beta rule and iota rule)} \\ \mid \{H, [x:A]H'\} & \\ \mid \langle H, [x:A]H' \rangle & \quad \text{(constructors)} \\ \mid HH' & \\ \mid \overline{A} \mid H^\dagger \mid H \cdot H' & \quad \text{(equivalence relation)}\end{aligned}$$

Judgements

Contexts:

$$\Gamma = \cdot \mid \Gamma, x : A.$$

Judgements:

$$\mathcal{J} = \Gamma \vdash_f \mid \Gamma \vdash_f a : A \mid \Gamma \vdash_f H : A = B.$$

PTS rules

$$\frac{\Gamma \vdash_f}{\Gamma \vdash_f s_1 : s_2} \ (s_1, s_2) \in \mathcal{A} \qquad \text{(sort)}$$

$$\frac{\Gamma \vdash_f}{\Gamma \vdash_f x : A} \ (x : A) \in \Gamma \qquad \text{(var)}$$

$$\frac{\Gamma \vdash_f A : s_1 \qquad \Gamma, x : A \vdash_f B : s_2}{\Gamma \vdash_f \Pi x{:}A.B : s_3} \ (s_1, s_2, s_3) \in \mathcal{R} \qquad \text{(prod)}$$

$$\frac{\Gamma, x : A \vdash_f b : B \qquad \Gamma \vdash_f \Pi x{:}A.B : s}{\Gamma \vdash_f \lambda x{:}A.b : \Pi x{:}A.B} \qquad \text{(abs)}$$

$$\frac{\Gamma \vdash_f F : \Pi x{:}A.B \qquad \Gamma \vdash_f a : A}{\Gamma \vdash_f Fa : B[x := a]} \qquad \text{(app)}$$

Conversion rule

Ordinary conversion rule:

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash A' : s}{\Gamma \vdash a : A'} \, (A \simeq_\beta A') \qquad \text{(conv)}$$

New conversion rule:

$$\frac{\Gamma \vdash_f a : A \qquad \Gamma \vdash_f A' : s \qquad \Gamma \vdash_f H : A = A'}{\Gamma \vdash_f a^H : A'} \qquad \text{(conv)}$$

Equivalence relation

Rules for the convertibility judgement:

$$\frac{\Gamma \vdash_f A : B}{\Gamma \vdash_f \overline{A} : A = A} \quad \text{(ref)}$$

$$\frac{\Gamma \vdash_f H : A = A'}{\Gamma \vdash_f H^{\dagger} : A' = A} \quad \text{(sym)}$$

$$\frac{\Gamma \vdash_f H : A = A' \qquad \Gamma \vdash_f H' : A' = A''}{\Gamma \vdash_f H \cdot H' : A = A''} \quad \text{(trans)}$$

Congruence

$$\frac{\Gamma \vdash_f H : A = A' \qquad \Gamma, x : A \vdash_f H' : B = B'[x' := x^H]}{\Gamma \vdash_f \{H, [x : A]H'\} : \Pi x{:}A.B = \Pi x'{:}A'.B'} \quad \text{(prod-eq)}$$

$$\frac{\Gamma \vdash_f H : A = A' \qquad \Gamma, x : A \vdash_f H' : b = b'[x' := x^H]}{\Gamma \vdash_f \langle H, [x : A]H'\rangle : \lambda x{:}A.b = \lambda x'{:}A'.b'} \quad \text{(abs-eq)}$$

$$\frac{\Gamma \vdash_f H : F = F' \qquad \Gamma \vdash_f H' : a = a'}{\Gamma \vdash_f HH' : Fa = F'a'} \quad \text{(app-eq)}$$

Beta and Iota

$$\frac{\Gamma \vdash_f (\lambda x{:}A.b)a : B}{\Gamma \vdash_f \beta((\lambda x{:}A.b)a) : (\lambda x{:}A.b)a = b[x := a]} \quad \text{(beta)}$$

$$\frac{\Gamma \vdash_f a^H : A}{\Gamma \vdash_f \iota(a^H) : a = a^H} \quad \text{(iota)}$$

Erasure map

- Define the erasure map map $|\cdot|$ from PTS_f terms to PTS terms by erasing all convertibility proofs.

$$|s| \equiv s \qquad |\Pi x{:}A.B| \equiv \Pi x{:}|A|.|B| \qquad |Fa| \equiv |F||a|$$
$$|x| \equiv x \qquad |\lambda x{:}A.b| \equiv \lambda x{:}|A|.|b| \qquad |a^H| \equiv |a|$$

- Extend to contexts

$$|x_1 : A_1, \ldots, x_n : A_n| \equiv x_1 : |A_1|, \ldots, x_n : |A_n|.$$

- A' is called a lift of A if $|A'| \equiv A$.

Equivalence between PTS and PTS_f

Theorem

$\Gamma \vdash a : A$ *iff there are lifts* Γ', a', A' *such that* $\Gamma' \vdash_f a' : A'$.

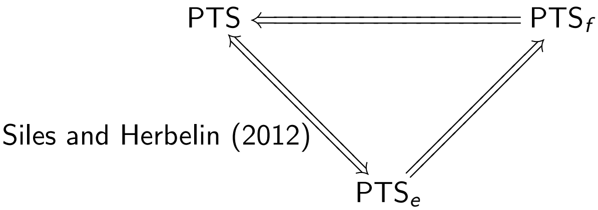
Equivalence between PTS and PTS_f

Theorem

- *If* $\Gamma \vdash$ *then there exists a lift* Γ' *such that* $\Gamma' \vdash_f$;
- *If* $\Gamma \vdash a : A$, *then for every legal lift* Γ' *there are lifts* a', A' *such that* $\Gamma' \vdash_f a' : A'$;
- *If* $A \simeq_\beta B$ *and* A *and* B *both have a type under* Γ, *then for every legal lift* Γ' *there are lifts* A', B' *such that* $\Gamma' \vdash_f H : A' = B'$ *for some* H.

Another PTS: PTS_e

- In the proof we used another version of PTS, PTS_e: Pure Type System with typed judgemental equality.



Key Lemma

Lemma

Suppose the following judgements hold:

- $\Gamma \vdash_f a_1 = a_2$
- $\Gamma, x : T \vdash_f M : N$
- $\Gamma \vdash_f a_1 : T$
- $\Gamma \vdash_f a_2 : T$

Then $\Gamma \vdash_f M[x := a_1] = M[x := a_2]$.

Formalisation

Proof is fully formalised in Coq!

`Theorem` PTSlequivPTSF : $(\forall \Gamma\ M\ N, (\Gamma \vdash' M : N)\%UT \leftrightarrow \exists \Gamma'\ M'\ N', \epsilon c\ \Gamma' = \Gamma \wedge \epsilon\ M' = M \wedge \epsilon\ N' = N \wedge \Gamma' \vdash M' : N') \wedge$
$(\forall \Gamma\ M\ N, (\exists A\ B, (\Gamma \vdash' M : A)\%UT \wedge (\Gamma \vdash' N : B)\%UT \wedge M \equiv N) \text{<->} \exists \Gamma'\ M'\ N', \epsilon c\ \Gamma' = \Gamma \wedge \epsilon\ M' = M \wedge \epsilon\ N' = N \wedge \Gamma' \vdash M' = N') \wedge$
$(\forall \Gamma, (\Gamma \vdash')\%UT \leftrightarrow \exists \Gamma', \epsilon c\ \Gamma' = \Gamma \wedge \Gamma' \vdash)$.

Used libraries of Siles and Herbelin

Conclusion

- PTS_f is equivalent to PTS
- Unique derivation: useful for meta-theory
- Unique typing: Enables to model a specific PTS in a LF framework
- Type checking is very easy
- Terms correspond more closely to proofs

Thank you

Thank you for your attention!