A Unified View of Pure Type Systems' Conversion

JAIST Spring School 2012

Vincent Siles

Göteborg University, Sweden

March 7, 2012

Simply Typed λ-Calculus

$$
\begin{array}{rcl}
M, N & ::= & x \mid \lambda x^A.M \mid M\ N \\
A, B & ::= & a \mid A \to B \\
\Gamma & ::= & \emptyset \mid \Gamma, x : A
\end{array}
$$

$$
\frac{\Gamma(x) = A}{\Gamma \vdash x : A} \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x^A.M : A \to B} \qquad \frac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\ N : B}
$$

$$
id\ A\ ==\ \lambda x^A.x\ :\ A \to A
$$

System-F

$$
\begin{array}{rcl}
M, N & ::= & x \mid \lambda x^A.M \mid M\ N \mid \Lambda a.M \mid M\ A \\
A, B & ::= & a \mid A \to B \mid \forall a, B \\
\Gamma & ::= & \emptyset \mid \Gamma, x : A
\end{array}
$$

$$
\frac{\Gamma(x) = A}{\Gamma \vdash x : A} \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x^A.M : A \to B} \qquad \frac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\ N : B}
$$

$$
\frac{\Gamma \vdash M : B}{\Gamma \vdash \Lambda a.M : \forall, a\ B} \qquad \frac{\Gamma \vdash M : \forall A, B}{\Gamma \vdash M\ A : B[A/a]}
$$

$$
id\ == \ \Lambda A.\lambda x^A.x\ :\ \forall A, A \to A
$$

Calculus of Constructions

$$\begin{array}{rcl} M, N, A, B & ::= & Prop \mid Type \mid x \mid \lambda x^A.M \mid M\ N \mid \Pi x : A.B \\ \Gamma & ::= & \emptyset \mid \Gamma, x : A \end{array}$$

$\ldots$

Pure Type Systems

Pure Type Systems have been built to *unify* all these different presentations in a single system:

Pure Type Systems

Pure Type Systems have been built to *unify* all these different presentations in a single system:

$$\begin{array}{rcl} M, N, A, B & ::= & x \mid \lambda x^A.M \mid M\ N \mid \Pi x^A.B^\dagger \mid s \\ \Gamma & ::= & \emptyset \mid \Gamma, x : A \end{array}$$

- PTSs are an abstraction of Barendregt's λ-cube, presented independently by Berardi and Terlouw.

†We write $A \to B$ when B does not depend on the input.

Pure Type Systems

Pure Type Systems have been built to *unify* all these different presentations in a single system:

$$\begin{array}{rcl} M, N, A, B & ::= & x \mid \lambda x^A.M \mid M\ N \mid \Pi x^A.B^\dagger \mid s \\ \Gamma & ::= & \emptyset \mid \Gamma, x : A \end{array}$$

- PTSs are an abstraction of Barendregt's λ-cube, presented independently by Berardi and Terlouw.
- To be able to deal with all the different type systems, PTSs have parameters that describe which type is valid: $Sorts, \mathcal{A}x$ and $\mathcal{R}el$.

† We write $A \to B$ when B does not depend on the input.

Typing Rules

$$\frac{}{\emptyset_{wf}} \qquad \frac{\Gamma \vdash A : s \quad x \notin Dom(\Gamma)}{(\Gamma, x : A)_{wf}} \qquad \frac{\Gamma_{wf} \quad (s,t) \in \mathcal{A}x}{\Gamma \vdash s : t} \qquad \frac{\Gamma_{wf} \quad \Gamma(x) = A}{\Gamma \vdash x : A}$$

$$\frac{\begin{array}{cc}\Gamma \vdash A : s & \Gamma, x : A \vdash B : t \\ (s,t,u) \in \mathcal{R}el & \Gamma, x : A \vdash M : B\end{array}}{\Gamma \vdash \lambda x^A.M : \Pi x^A.B}$$

$$\frac{\Gamma \vdash A : s \qquad \Gamma, x : A \vdash B : t \qquad (s,t,u) \in \mathcal{R}el}{\Gamma \vdash \Pi x^A.B : u}$$

$$\frac{\Gamma \vdash M : \Pi x^A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B[N/x]} \qquad \frac{\Gamma \vdash M : A \qquad A \overset{\beta}{\equiv} B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

Difference with STLC

STLC		PTS
$\dfrac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x^A.M : A \to B}$	$\Rightarrow$	$\dfrac{\Gamma \vdash \Pi x^A.B : s \qquad \Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x^A.M : \Pi x^A.B}$
$\dfrac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\ N : B}$	$\Rightarrow$	$\dfrac{\Gamma \vdash M : \Pi x^A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\ N : B[N/x]}$

Difference with STLC

STLC		PTS
$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x^A.M : A \rightarrow B}$$	$\Rightarrow$	$$\frac{\Gamma \vdash \Pi x^A.B : s \qquad \Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x^A.M : \Pi x^A.B}$$
$$\frac{\Gamma \vdash M : A \rightarrow B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\ N : B}$$	$\Rightarrow$	$$\frac{\Gamma \vdash M : \Pi x^A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\ N : B[N/x]}$$

$$\frac{\Gamma \vdash M : A \qquad A =_\beta B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}\ \text{CONV}$$

Why a conversion rule ?

- Let's consider the type of lists of $\mathbb{N}$ of length `n`: `list n`.

Why a conversion rule ?

- Let's consider the type of lists of $\mathbb{N}$ of length `n`: `list n`.
- What is the result and the type of `concat l l` ?
  $\Pi n^{nat}.\Pi m^{nat}.\ \ \texttt{list}\ n \rightarrow \texttt{list}\ m \rightarrow \texttt{list}\ (n+m)$

Why a conversion rule ?

- Let's consider the type of lists of $\mathbb{N}$ of length `n`: `list n`.
- What is the result and the type of `concat l l` ?
  $\Pi n^{nat}.\Pi m^{nat}.\ \ \texttt{list}\ n \rightarrow \texttt{list}\ m \rightarrow \texttt{list}\ (n+m)$
- $\texttt{l = }[1,3,5,7]\ \texttt{:}\ \ \texttt{list}\ 4$

Why a conversion rule ?

- Let's consider the type of lists of $\mathbb{N}$ of length `n`: `list n`.
- What is the result and the type of `concat l l` ?
  $\Pi n^{nat}.\Pi m^{nat}.\ \ \texttt{list}\ n \rightarrow \texttt{list}\ m \rightarrow \texttt{list}\ (n+m)$
- $\texttt{l} = [1,3,5,7]\ :\ \texttt{list}\ 4$
- $\texttt{concat 4 4 l l} =_\beta [1,3,5,7,1,3,5,7]\ :\ \texttt{list}\ (4{+}4)$

The conversion rule is here to *compute* at the level of types and change `list` (4+4) into `list` 8.

Facts about PTSs

For example:

Type Correctness

If $\Gamma \vdash M : T$ then there is $s \in \mathit{Sorts}$ such that $T \equiv s$ or $\Gamma \vdash T : s$.

Facts about PTSs

For example:

Type Correctness

If $\Gamma \vdash M : T$ then there is $s \in Sorts$ such that $T \equiv s$ or $\Gamma \vdash T : s$.

A more complex one:

Subject Reduction

If $\Gamma \vdash M : T$ and $M \to_\beta M'$ then $\Gamma \vdash M' : T$.

Π-injectivity

The proof of Subject-Reduction relies on the following property:

Injectivity of products

If $\Pi x^A.B =_\beta \Pi x^C.D$ then $A =_\beta C$ and $B =_\beta D$.

Proof.

By *Confluence* of β-reduction, there is M such that $\Pi x^A.B \twoheadrightarrow_\beta M$ and $\Pi x^C.D \twoheadrightarrow_\beta M$. By definition of $\twoheadrightarrow_\beta$, this implies that M is of the shape $\Pi x^K.L$ and that $A \twoheadrightarrow_\beta K$, $C \twoheadrightarrow_\beta K$, $B \twoheadrightarrow_\beta L$ and $D \twoheadrightarrow_\beta L$. $\square$

Untyped conversion considered harmful ?

What if the path between A and B is “ill-typed” ?

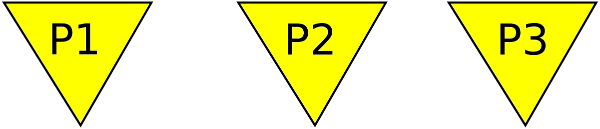
$$\frac{\Gamma \vdash M : A \qquad A =_\beta B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

$$\frac{}{(\lambda x^{A'}.P)\ Q =_\beta P[Q/x]}$$

Untyped conversion considered harmful ?

What if the path between *A* and *B* is “ill-typed” ?

Let’s consider *P* to be the following proof of $\Gamma \vdash M : B$.

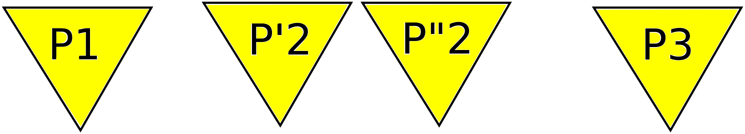


$$\frac{\Gamma \vdash M : A \qquad A =_\beta B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

Untyped conversion considered harmful ?

What if the path between *A* and *B* is “ill-typed” ?

By Confluence:



$$\frac{\Gamma \vdash M : A \quad A \twoheadrightarrow_\beta C \quad B \twoheadrightarrow_\beta C \quad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

Untyped conversion considered harmful ?

What if the path between *A* and *B* is “ill-typed” ?

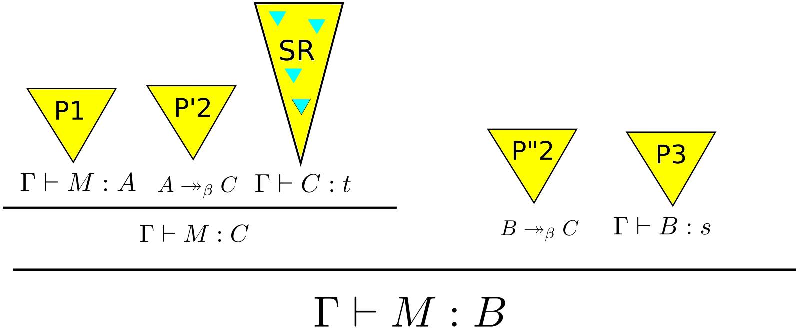
By Type Correctness and Subject Reduction:

$$\dfrac{\dfrac{\overset{\text{P1}}{\Gamma \vdash M : A} \quad \overset{\text{P'2}}{A \twoheadrightarrow_\beta C} \quad \overset{\text{SR}}{\Gamma \vdash C : t}}{\Gamma \vdash M : C} \qquad \overset{\text{P"2}}{B \twoheadrightarrow_\beta C} \quad \overset{\text{P3}}{\Gamma \vdash B : s}}{\Gamma \vdash M : B}$$

Untyped conversion considered harmful ?

What if the path between *A* and *B* is “ill-typed” ?

But Subject Reduction introduces new harmful conversions:



PTS with typed equality (PTSe)

To avoid this possibility, we can *type* every step of the conversion. They are called *semantical PTSs* or *PTSs with typed equality* [Geuvers93]. We have now two typing judgments:

PTS with typed equality (PTSe)

To avoid this possibility, we can *type* every step of the conversion. They are called *semantical PTSs* or *PTSs with typed equality* [Geuvers93]. We have now two typing judgments:

- One for terms: $\Gamma \vdash_e M : T$

PTS with typed equality (PTSe)

To avoid this possibility, we can *type* every step of the conversion. They are called *semantical PTSs* or *PTSs with typed equality* [Geuvers93]. We have now two typing judgments:

- One for terms: $\Gamma \vdash_e M : T$
- One for equalities: $\Gamma \vdash_e M =_\beta N : T$

PTS with typed equality (PTSe)

To avoid this possibility, we can *type* every step of the conversion. They are called *semantical PTSs* or *PTSs with typed equality* [Geuvers93]. We have now two typing judgments:

- One for terms: $\Gamma \vdash_e M : T$
- One for equalities: $\Gamma \vdash_e M =_\beta N : T$

$$\frac{\Gamma \vdash M : A \qquad A =_\beta B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

PTS with typed equality (PTSe)

To avoid this possibility, we can *type* every step of the conversion. They are called *semantical PTSs* or *PTSs with typed equality* [Geuvers93]. We have now two typing judgments:

- One for terms: $\Gamma \vdash_e M : T$
- One for equalities: $\Gamma \vdash_e M =_\beta N : T$

$$\frac{\Gamma \vdash_e M : A \qquad \Gamma \vdash_e A =_\beta B : s}{\Gamma \vdash_e M : B}$$

PTS with typed equality

Untyped β-equality is quite "small":

$$\frac{}{(\lambda x^A.M)\ N =_\beta M[N/x]}$$

$$\frac{A =_\beta A' \qquad M =_\beta M'}{\lambda x^A.M =_\beta \lambda x^{A'}.M'}$$

PTS with typed equality

Typed β-equality is notably "bigger":

$$\frac{\Gamma, x : A \vdash_e M : B \qquad \Gamma \vdash_e N : A \\ \Gamma \vdash_e A : s \qquad \Gamma, x : A \vdash_e B : t \qquad (s, t, u) \in \mathcal{R}el}{\Gamma \vdash_e (\lambda x^A.M)N =_\beta M[N/x] : B[N/x]}$$

$$\frac{\Gamma \vdash_e A =_\beta A' : s \qquad \Gamma, x : A \vdash_e M =_\beta M' : B \\ \Gamma, x : A \vdash_e B : t \qquad (s, t, u) \in \mathcal{R}el}{\Gamma \vdash_e \lambda x^A.M =_\beta \lambda x^{A'}.M' : \Pi x^A.B}$$

Facts about PTSe

Almost all the properties that we know about PTSs are easily proved valid for PTSe, and there are new ones:

Facts about PTSe

Almost all the properties that we know about PTSs are easily proved valid for PTSe, and there are new ones:

Left-hand/Right-hand reflexivity

If $\Gamma \vdash_e M =_\beta N : T$ then $\Gamma \vdash_e M : T$ and $\Gamma \vdash_e N : T$.

Facts about PTSe

Almost all the properties that we know about PTSs are easily proved valid for PTSe, and there are new ones:

Left-hand/Right-hand reflexivity

If $\Gamma \vdash_e M =_\beta N : T$ then $\Gamma \vdash_e M : T$ and $\Gamma \vdash_e N : T$.

However, Subject Reduction is really troublesome to prove:

Typed Subject Reduction:

If $\Gamma \vdash_e M : T$ and $M \to_\beta N$, then $\Gamma \vdash_e M =_\beta N : T$.

Back to the old proof

To prove Subject-Reduction for PTS, we relied on Confluence, and more precisely on Π-injectivity. Now that we have a typed equality, we need a typed version of this injectivity property.

Proving it for PTS_e is a difficult problem since we can't do it in the same way as PTS: the proof would rely on (typed) *Confluence*,

Back to the old proof

To prove Subject-Reduction for PTS, we relied on Confluence, and more precisely on Π-injectivity. Now that we have a typed equality, we need a typed version of this injectivity property.

Proving it for PTS_e is a difficult problem since we can't do it in the same way as PTS: the proof would rely on (typed) *Confluence*, which relies on *Subject Reduction*,

Back to the old proof

To prove Subject-Reduction for PTS, we relied on Confluence, and more precisely on Π-injectivity. Now that we have a typed equality, we need a typed version of this injectivity property.

Proving it for PTS_e is a difficult problem since we can't do it in the same way as PTS: the proof would rely on (typed) *Confluence*, which relies on *Subject Reduction*, which relies on Π-Injectivity,

Back to the old proof

To prove Subject-Reduction for PTS, we relied on Confluence, and more precisely on Π-injectivity. Now that we have a typed equality, we need a typed version of this injectivity property.

Proving it for PTS_e is a difficult problem since we can't do it in the same way as PTS: the proof would rely on (typed) *Confluence*, which relies on *Subject Reduction*, which relies on Π-Injectivity, which relies on ...

Back to the old proof

To prove Subject-Reduction for PTS, we relied on Confluence, and more precisely on Π-injectivity. Now that we have a typed equality, we need a typed version of this injectivity property.

Proving it for PTS_e is a difficult problem since we can't do it in the same way as PTS: the proof would rely on (typed) *Confluence*, which relies on *Subject Reduction*, which relies on Π-Injectivity, which relies on ...

Non-fact: Strong Π-injectivity

If $\Gamma \vdash_e \Pi x^A.B =_\beta \Pi x^C.D : u$, then there are s, t such that $(s, t, u) \in \mathcal{R}el$, $\Gamma \vdash_e A =_\beta C : s$ and $\Gamma, x : A \vdash_e B =_\beta D : t$.

Back to the old proof

To prove Subject-Reduction for PTS, we relied on Confluence, and more precisely on Π-injectivity. Now that we have a typed equality, we need a typed version of this injectivity property.

Proving it for PTS_e is a difficult problem since we can't do it in the same way as PTS: the proof would rely on (typed) *Confluence*, which relies on *Subject Reduction*, which relies on Π-Injectivity, which relies on ...

We do not know any direct proof of this fact in a direct manner.

Another kind of equality

Some presentations of programming languages and type theories based on the work of Martin-Löf are using another form of equality:

Another kind of equality

Some presentations of programming languages and type theories based on the work of Martin-Löf are using another form of equality:

- The equality for terms is the same : $\Gamma \vdash_e M =_\beta N : T$.

Another kind of equality

Some presentations of programming languages and type theories based on the work of Martin-Löf are using another form of equality:

- The equality for terms is the same : $\Gamma \vdash_e M =_\beta N : T$.
- The equality for types is weaker : $\Gamma \vdash_e A =_\beta B$.

Another kind of equality

Some presentations of programming languages and type theories based on the work of Martin-Löf are using another form of equality:

- The equality for terms is the same : $\Gamma \vdash_e M =_\beta N : T$.
- The equality for types is weaker : $\Gamma \vdash_e A =_\beta B$.

If we assume that this equality enjoys Π-injectivity, then it is enough to prove *Subject Reduction* for PTSe. Sadly, there is also no known proof of this fact at the moment.

Is there another way to prove it ?

Another way to prove Subject Reduction for PTSe would be to use the Subject Reduction we have for PTSs. We need to prove some kind of equivalence between both systems.

Is there another way to prove it ?

Another way to prove Subject Reduction for PTSe would be to use the Subject Reduction we have for PTSs. We need to prove some kind of equivalence between both systems.

A more practical reason why we are looking for this equivalence is about *proof assistants*. Usually, the implementation is done with an untyped equality, whereas the consistency proof is done with a typed equality. Such an equivalence would bring closer the implementation from its theory.

Are PTSs and PTSe the same systems ?

[Geuvers93]

Easy part of the equivalence

We prove by mutual induction that

- If $\Gamma \vdash_e M : T$ then $\Gamma \vdash M : T$.
- If $\Gamma \vdash_e M =_\beta N : T$ then $\Gamma \vdash M : T$, $\Gamma \vdash N : T$ and $M =_\beta N$.
- If Γ_{wf_e} then Γ_{wf}.

Easy part of the equivalence

We prove by mutual induction that

- If $\Gamma \vdash_e M : T$ then $\Gamma \vdash M : T$.
- If $\Gamma \vdash_e M =_\beta N : T$ then $\Gamma \vdash M : T$, $\Gamma \vdash N : T$ and $M =_\beta N$.
- If Γ_{wf_e} then Γ_{wf}.

Here we just “lose” some information, nothing difficult.

Difficult part of the equivalence

The other way around needs a way to “type” a β-equivalence into a judgmental equality:

- If $\Gamma \vdash M : T$ then $\Gamma \vdash_e M : T$.
- If $\Gamma \vdash M : T$, $\Gamma \vdash N : T$ and $M =_\beta N$ then $\Gamma \vdash_e M =_\beta N : T$.
- If Γ_{wf} then Γ_{wf_e}.

Difficult part of the equivalence

The other way around needs a way to “type” a β-equivalence into a judgmental equality:

- If $\Gamma \vdash M : T$ then $\Gamma \vdash_e M : T$.
- If $\Gamma \vdash M : T$, $\Gamma \vdash N : T$ and $M =_\beta N$ then $\Gamma \vdash_e M =_\beta N : T$.
- If Γ_{wf} then Γ_{wf_e}.

Here, we need to find a way to type all the intermediate steps.

Difficult part of the equivalence

The other way around needs a way to “type” a β-equivalence into a judgmental equality:

- If $\Gamma \vdash M : T$ then $\Gamma \vdash_e M : T$.
- If $\Gamma \vdash M : T$, $\Gamma \vdash N : T$ and $M =_\beta N$ then $\Gamma \vdash_e M =_\beta N : T$.
- If Γ_{wf} then Γ_{wf_e}.

Here, we need to find a way to type all the intermediate steps.

But can we ?

The Big Answer

YES

[Siles 2010]

The Big Answer

YES

[Siles 2010]

but its quite complex and not really intuitive.

PTS $\rightarrow$ PTS$_e$: How do we do this ?

$\Gamma \vdash M : T \qquad M \qquad =_\beta \qquad N \qquad \Gamma \vdash N : T$

PTS $\rightarrow$ PTS$_e$: How do we do this ?

$$
\begin{tikzcd}
\Gamma \vdash M : T & M \arrow[dr, two heads] & =_\beta & N \arrow[dl, two heads] & \Gamma \vdash N : T \\
\Gamma \vdash_e M : T & & P & & \Gamma \vdash_e N : T
\end{tikzcd}
$$

PTS $\to$ PTS$_e$: How do we do this ?

$$
\begin{tikzcd}
\Gamma \vdash M : T & M \arrow[dr, two heads] & =_\beta & N \arrow[dl, two heads] & \Gamma \vdash N : T \\
\Gamma \vdash_e M : T & & P & & \Gamma \vdash_e N : T
\end{tikzcd}
$$

- P is well-typed in PTS by *Subject Reduction.*

PTS $\rightarrow$ PTS$_e$: How do we do this ?

$$\begin{tikzcd}
\Gamma \vdash M : T & M \arrow[dr, twoheadrightarrow] & =_\beta & N \arrow[dl, twoheadrightarrow] & \Gamma \vdash N : T \\
\Gamma \vdash_e M : T & & P & & \Gamma \vdash_e N : T
\end{tikzcd}$$

- P is well-typed in PTS by *Subject Reduction.*
- Is P well-typed in PTSe ?

PTS $\rightarrow$ PTS$_e$: How do we do this ?

$$
\begin{tikzcd}
\Gamma \vdash M : T & M \arrow[dr, two heads] & =_\beta & N \arrow[dl, two heads] & \Gamma \vdash N : T \\
\Gamma \vdash_e M : T & & P & & \Gamma \vdash_e N : T
\end{tikzcd}
$$

- P is well-typed in PTS by *Subject Reduction.*
- Is P well-typed in PTSe ?
- How do we type $M =_\beta P$ and $N =_\beta P$ in PTSe ?

Some partial solutions

- Early attempts to prove such an equivalence did not aim at the whole generality of PTSs, and were based on the construction of a model [Geuvers93,Goguen94].

Some partial solutions

- Early attempts to prove such an equivalence did not aim at the whole generality of PTSs, and were based on the construction of a model [Geuvers93,Goguen94].
- A first *syntactical* criterion was shown for a subclass of PTSs [Adams06] called *functional* PTSs, by adding annotations inside the syntax of terms.

Some partial solutions

- Early attempts to prove such an equivalence did not aim at the whole generality of PTSs, and were based on the construction of a model [Geuvers93,Goguen94].
- A first *syntactical* criterion was shown for a subclass of PTSs [Adams06] called *functional* PTSs, by adding annotations inside the syntax of terms.
- By using the same intermediate system, Herbelin and I extended this result to other subclasses of PTSs called *semi-full* and *full*.

Extension of the TPOSR solution

Adams introduced an additional annotation inside the applications:

$$M, N, A, B \quad ::= \quad x \mid \lambda x^A.M \mid M_{(x)B}\ N \mid \Pi x^A.B \mid s$$

Extension of the TPOSR solution

Adams introduced an additional annotation inside the applications:

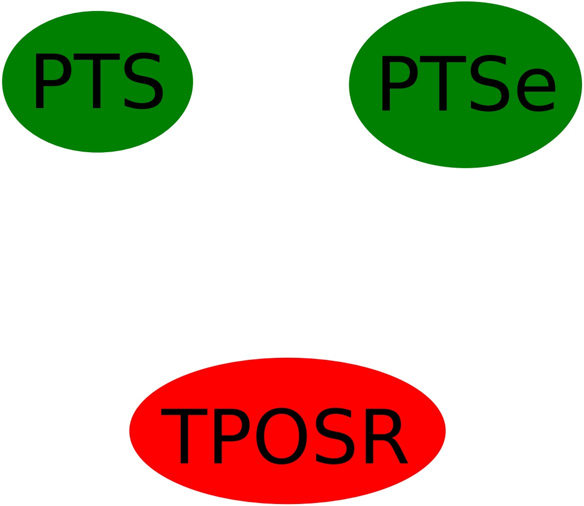
$$M, N, A, B \quad ::= \quad x \mid \lambda x^A.M \mid M_{(x)B}\ N \mid \Pi x^A.B \mid s$$

Also, its system called *Typed Parallel One Step Reduction* is no longer based on equality but on *reduction*:

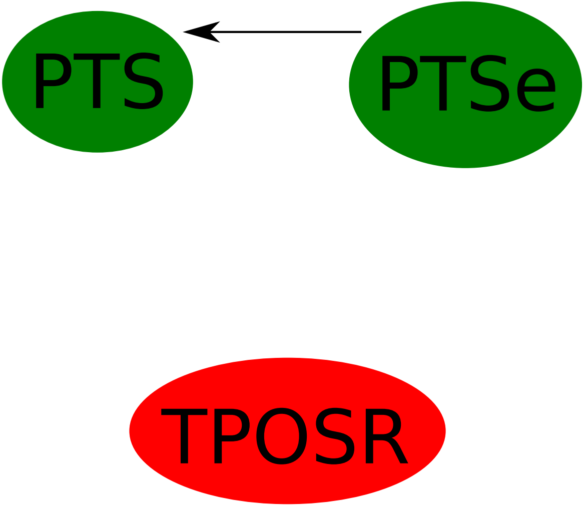
$$\frac{\Gamma \vdash M \rhd N : A \qquad \Gamma \vdash A \cong B : s}{\Gamma \vdash M \rhd N : B}$$

$$\frac{\begin{array}{c}\Gamma \vdash A \rhd A' : s \qquad \Gamma, x : A \vdash B \rhd B' : t \\ \Gamma, x : A \vdash M \rhd M' : B \qquad \Gamma \vdash N \rhd N' : A \qquad (s, t, u) \in \mathcal{R}el\end{array}}{\Gamma \vdash (\lambda x^A.M)_{(x)B} N \rhd M'[N'/x] : B[N/x]}$$

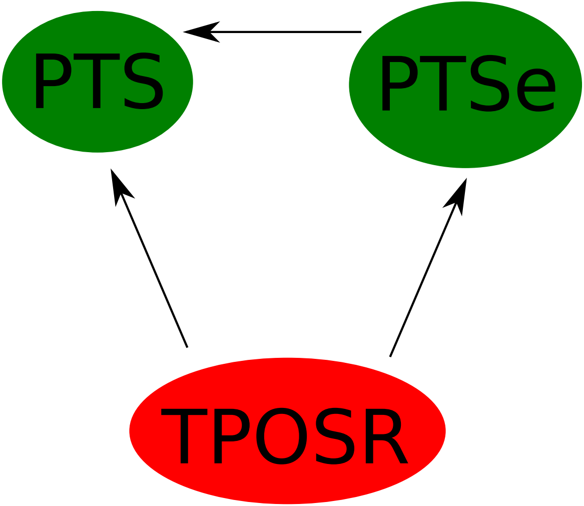
Summary of the proof



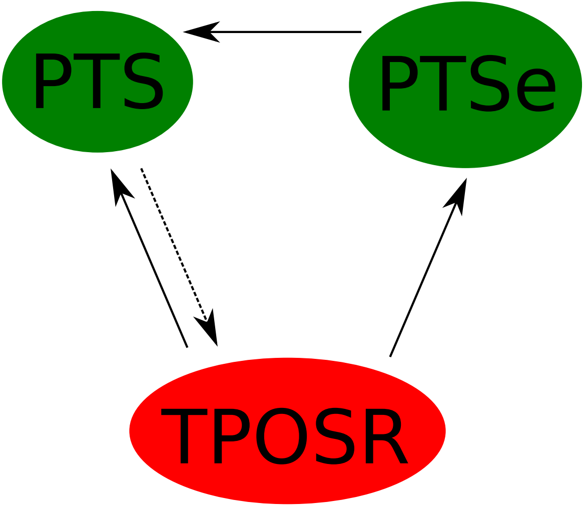
Summary of the proof



Summary of the proof



Summary of the proof



Road map to a proof of equivalence

The idea is to prove that:

- TPOSR's equality is *Confluent*.
- TPOSR's equality has *Injectivity of* Π*-types*.
- TPOSR has *Subject-Reduction*.
- TPOSR is equivalent to PTS and PTSe.

Road map to a proof of equivalence

The idea is to prove that:

- TPOSR's equality is *Confluent*.
- TPOSR's equality has *Injectivity of* Π*-types*.
- TPOSR has *Subject-Reduction*.
- TPOSR is equivalent to PTS and PTSe.

$\Uparrow$

tricky part

PTS with Annotated Typed Reduction

To achieve the equivalence for *all* PTSs, we need to improve TPOSR. The idea came from [Streicher91], but was used for completeness results.

PTS with Annotated Typed Reduction

To achieve the equivalence for *all* PTSs, we need to improve TPOSR. The idea came from [Streicher91], but was used for completeness results.

- Our idea is to extend the annotation on application: $M_{\Pi x^A.B}\ N$.

PTS with Annotated Typed Reduction

To achieve the equivalence for *all* PTSs, we need to improve TPOSR. The idea came from [Streicher91], but was used for completeness results.

- Our idea is to extend the annotation on application: $M_{\Pi x^A.B}\ N$.
- And we have to change the typing rule to deal with this new annotation:

PTS with Annotated Typed Reduction

To achieve the equivalence for *all* PTSs, we need to improve TPOSR. The idea came from [Streicher91], but was used for completeness results.

- Our idea is to extend the annotation on application: $M_{\Pi x^A . B}\ N$.
- And we have to change the typing rule to deal with this new annotation:

$$\frac{\dots \qquad \Gamma \vdash A \cong A' : s \qquad \Gamma, x : A \vdash M \rhd M' : B \qquad \Gamma \vdash N \rhd N' : A}{\Gamma \vdash (\lambda x^A . M)_{\Pi x^{A'} . B} N \rhd M'[N'/x] : B[N/x]}$$

PTS with Annotated Typed Reduction

To achieve the equivalence for *all* PTSs, we need to improve TPOSR. The idea came from [Streicher91], but was used for completeness results.

- Our idea is to extend the annotation on application: $M_{\Pi x^A.B}\ N$.
- And we have to change the typing rule to deal with this new annotation:

$$\frac{\begin{array}{ccc} \ldots & \Gamma \vdash A_0 \rhd^+ A : s & \Gamma \vdash A_0 \rhd^+ A' : s \\ & \Gamma, x : A \vdash M \rhd M' : B & \Gamma \vdash N \rhd N' : A \end{array}}{\Gamma \vdash (\lambda x^A.M)_{\Pi x^{A'}.B} N \rhd M'[N'/x] : B[N/x]}$$

Typed Confluence and Injectivity

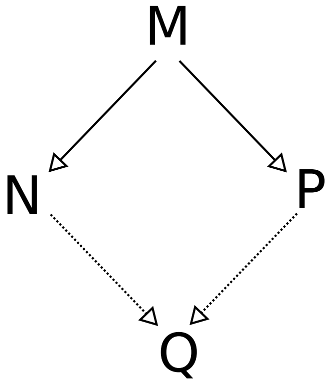
Diamond property for PTS_{atr}

If $\Gamma \vdash M \rhd N : A$ and $\Gamma \vdash M \rhd P : B$ then there is Q such that $\Gamma \vdash N \rhd Q : A, B$ and $\Gamma \vdash P \rhd Q : A, B$.

Typed Confluence and Injectivity

Diamond property for PTS_{atr}

If $\Gamma \vdash M \rhd N : A$ and $\Gamma \vdash M \rhd P : B$ then there is Q such that $\Gamma \vdash N \rhd Q : A, B$ and $\Gamma \vdash P \rhd Q : A, B$.



Typed Confluence and Injectivity

Diamond property for PTS_{atr}

If $\Gamma \vdash M \rhd N : A$ and $\Gamma \vdash M \rhd P : B$ then there is Q such that $\Gamma \vdash N \rhd Q : A, B$ and $\Gamma \vdash P \rhd Q : A, B$.

$$\Gamma \vdash MN : \Pi x^A.B \Rightarrow \exists C, \exists D,$$
$$\Gamma \vdash M : \Pi x^C.D \wedge \Pi x^A.B =_\beta \Pi x^C.D$$

Typed Confluence and Injectivity

Diamond property for PTS_{atr}

If $\Gamma \vdash M \rhd N : A$ and $\Gamma \vdash M \rhd P : B$ then there is Q such that $\Gamma \vdash N \rhd Q : A, B$ and $\Gamma \vdash P \rhd Q : A, B$.

$$\Gamma \vdash_e MN : \Pi x^A.B \Rightarrow \exists C, \exists D,$$
$$\Gamma \vdash_e M : \Pi x^C.D \wedge \Gamma \vdash_e \Pi x^A.B =_\beta \Pi x^C.D$$

Typed Confluence and Injectivity

Diamond property for PTS$_{atr}$

If $\Gamma \vdash M \rhd N : A$ and $\Gamma \vdash M \rhd P : B$ then there is Q such that $\Gamma \vdash N \rhd Q : A, B$ and $\Gamma \vdash P \rhd Q : A, B$.

$$\Gamma \vdash M_{\Pi x^A.B} N \rhd M'_{\Pi x^{A'}.B'} N' : B[N/x] \Rightarrow$$
$$\Gamma \vdash M \rhd M' : \Pi x^A.B$$

Typed Confluence and Injectivity

Diamond property for PTS_{atr}

If $\Gamma \vdash M \rhd N : A$ and $\Gamma \vdash M \rhd P : B$ then there is Q such that $\Gamma \vdash N \rhd Q : A, B$ and $\Gamma \vdash P \rhd Q : A, B$.

As a direct consequence:

Π-Injectivity for PTS_{atr}

If $\Gamma \vdash \Pi x^A.B \cong \Pi x^C.D$ then $\Gamma \vdash A \cong C$ and $\Gamma, x : A \vdash B \cong D$.

Typed Subject Reduction and annotations

As we said before, the key point of the equivalence is the *Subject Reduction* of the typed system:

Typed Subject Reduction and annotations

As we said before, the key point of the equivalence is the *Subject Reduction* of the typed system:

Subject Reduction for PTS_{atr}

If $\Gamma \vdash M \rhd P : T$ and $M \to_\beta N$ then $\Gamma \vdash M \rhd^+ N : T$.

Typed Subject Reduction and annotations

As we said before, the key point of the equivalence is the *Subject Reduction* of the typed system:

Subject Reduction for PTS$_{atr}$

If $\Gamma \vdash M \rhd P : T$ and $M \to_\beta N$ then $\Gamma \vdash M \rhd^+ N : T$.

The proof is almost the same as the usual one for PTSs. Some additional work is required for the β case: we need to provide the A_0 that links both annotations.

$$\frac{\dots \qquad \Gamma \vdash A_0 \rhd^+ A : s \qquad \Gamma \vdash A_0 \rhd^+ A' : s}{\Gamma \vdash (\lambda x^A.M)_{\Pi x^{A'}.B} N \rhd M'[N'/x] : B[N/x]}$$

Equivalence between PTSs and PTS$_{atr}$

- It is easy to translate a PTS$_{atr}$ judgment into PTSs (same kind of erasure than PTSe).

Equivalence between PTSs and PTS$_{atr}$

- It is easy to translate a PTS$_{atr}$ judgment into PTSs (same kind of erasure than PTSe).
- However, from usual PTSs, we need to compute the additional annotations needed by PTS$_{atr}$:

Equivalence between PTSs and PTS$_{atr}$

- It is easy to translate a PTS$_{atr}$ judgment into PTSs (same kind of erasure than PTSe).
- However, from usual PTSs, we need to compute the additional annotations needed by PTS$_{atr}$:

From PTS to PTS$_{atr}$

If $\Gamma \vdash M : T$, then there is Γ^*, M^* and T^* such that $\Gamma^* \vdash M^* \rhd M^* : T^*$, where $|\Gamma^*| \equiv \Gamma$, $|M^*| \equiv M$ and $|T^*| \equiv T$.

Equivalence between PTSs and PTS$_{atr}$

- It is easy to translate a PTS$_{atr}$ judgment into PTSs (same kind of erasure than PTSe).
- However, from usual PTSs, we need to compute the additional annotations needed by PTS$_{atr}$:

From PTS to PTS$_{atr}$

If $\Gamma \vdash M : T$, then there is Γ^*, M^* and T^* such that $\Gamma^* \vdash M^* \rhd M^* : T^*$, where $|\Gamma^*| \equiv \Gamma$, $|M^*| \equiv M$ and $|T^*| \equiv T$.

- How do we compute the valid Γ^*, M^* and T^* ?.

Annotations and Typing

The proof is done by induction:

$$\frac{\Gamma \vdash A : s \qquad \Gamma, x : A \vdash B : t \qquad (s, t, u) \in \mathcal{R}el}{\Gamma \vdash \Pi x^A.B : u}$$

Annotations and Typing

The proof is done by induction:

$$\frac{\Gamma \vdash A : s \qquad \Gamma, x : A \vdash B : t \qquad (s, t, u) \in \mathcal{R}el}{\Gamma \vdash \Pi x^A . B : u}$$

By induction, we have:

- Γ_1, A_1 such that $\Gamma_1 \vdash A_1 \rhd A_1 : s$, $|\Gamma_1| \equiv \Gamma$ and $|A_1| \equiv A$.

Annotations and Typing

The proof is done by induction:

$$\frac{\Gamma \vdash A : s \qquad \Gamma, x : A \vdash B : t \qquad (s, t, u) \in \mathcal{R}el}{\Gamma \vdash \Pi x^A.B : u}$$

By induction, we have:

- Γ_1, A_1 such that $\Gamma_1 \vdash A_1 \rhd A_1 : s$, $|\Gamma_1| \equiv \Gamma$ and $|A_1| \equiv A$.
- Γ_2, A_2 and B_2 such that $\Gamma_2, x : A_2 \vdash B_2 \rhd B_2 : t$, $|\Gamma_2| \equiv \Gamma$, $|A_2| \equiv A$ and $|B_2| \equiv B$.

Annotations and Typing

The proof is done by induction:

$$\frac{\Gamma \vdash A : s \qquad \Gamma, x : A \vdash B : t \qquad (s,t,u) \in \mathcal{R}el}{\Gamma \vdash \Pi x^A.B : u}$$

By induction, we have:

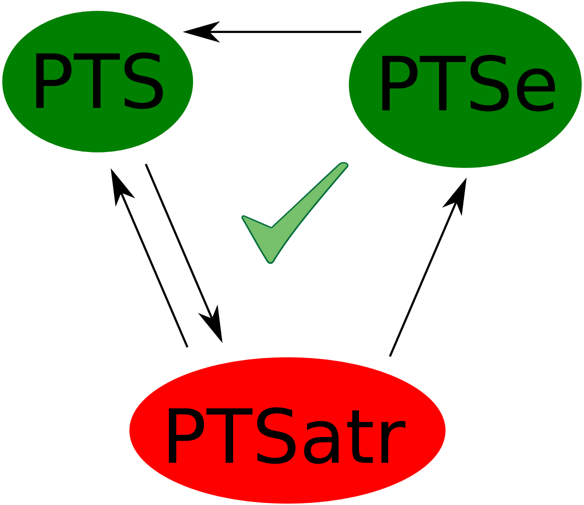
- Γ_1, A_1 such that $\Gamma_1 \vdash A_1 \rhd A_1 : s$, $|\Gamma_1| \equiv \Gamma$ and $|A_1| \equiv A$.
- Γ_2, A_2 and B_2 such that $\Gamma_2, x : A_2 \vdash B_2 \rhd B_2 : t$, $|\Gamma_2| \equiv \Gamma$, $|A_2| \equiv A$ and $|B_2| \equiv B$.
- We need a way to glue things together:

Erased Conversion

If $|A| \equiv |B|$, and if A and B are well-formed types in PTS_{atr}, then $\Gamma \vdash A \cong B$.

The proof of this lemma is very technical, and the most difficult proof of my thesis.

Complete Equivalence



Consequences of the equivalence

Complete Equivalence:

$$\begin{cases} \Gamma \vdash_e M : T & \textit{iff} \quad \Gamma \vdash M : T \\ \Gamma \vdash_e M =_\beta N : T & \textit{iff} \quad \Gamma \vdash M : T, \Gamma \vdash N : T \text{ and } M =_\beta N \\ \Gamma_{wf} & \textit{iff} \quad \Gamma_{wf_e} \end{cases}$$

Consequences of the equivalence

Complete Equivalence:

$$\begin{cases} \Gamma \vdash_e M : T & \textit{iff} \quad \Gamma \vdash M : T \\ \Gamma \vdash_e M =_\beta N : T & \textit{iff} \quad \Gamma \vdash M : T, \Gamma \vdash N : T \text{ and } M =_\beta N \\ \Gamma_{wf} & \textit{iff} \quad \Gamma_{wf_e} \end{cases}$$

Proving Subject Reduction for PTSe is now trivial:

- If $\Gamma \vdash_e M : T$ and $M \to_\beta N$, then $\Gamma \vdash M : T$.
- By Subject Reduction in PTS, $\Gamma \vdash N : T$, so $\Gamma \vdash_e N : T$.
- Once again, by equivalence, $\Gamma \vdash_e M =_\beta N : T$.

Consequences of the equivalence

Complete Equivalence:

$$\begin{cases} \Gamma \vdash_e M : T & \textit{iff} \quad \Gamma \vdash M : T \\ \Gamma \vdash_e M =_\beta N : T & \textit{iff} \quad \Gamma \vdash M : T, \Gamma \vdash N : T \text{ and } M =_\beta N \\ \Gamma_{wf} & \textit{iff} \quad \Gamma_{wf_e} \end{cases}$$

Proving Subject Reduction for PTSe is now trivial:

- If $\Gamma \vdash_e M : T$ and $M \rightarrow_\beta N$, then $\Gamma \vdash M : T$.
- By Subject Reduction in PTS, $\Gamma \vdash N : T$, so $\Gamma \vdash_e N : T$.
- Once again, by equivalence, $\Gamma \vdash_e M =_\beta N : T$.

Corollary: Weak Π-Injectivity

If $\Gamma \vdash_e \Pi x^A.B =_\beta \Pi x^C.D$ then $\Gamma \vdash_e A =_\beta C$ and $\Gamma, x : A \vdash_e B =_\beta D$.

By the way

Formalizing such theorem in a proof assistant has proved to be quite helpfull (if not mandatory):

- You do the non-interesting things once (handling binders, weakening, substitution,...)
- So you can only focus on the interesting (design a new system, adapt proofs,...)

By the way

Formalizing such theorem in a proof assistant has proved to be quite helpfull (if not mandatory):

- You do the non-interesting things once (handling binders, weakening, substitution,...)
- So you can only focus on the interesting (design a new system, adapt proofs,...)

Proved in Coq

Conclusion

What to bring home:

+ PTS and PTS_e are equivalent: you can choose what's best for you.
- The proof of equivalence is completely syntactic and (I think) too complex.
- Trying to extend the system (e.g. with subtyping of sorts, η-conversion, ...) will certainly break the proof: it doesn't scale nicely.
+ Doing proofs within a proof assistant can save you a lot of time when trying to define new systems whose syntax is not clear yet.

Conclusion

What to bring home:

- \+ PTS and PTS_e are equivalent: you can choose what's best for you.
- \- The proof of equivalence is completely syntactic and (I think) too complex.
- \- Trying to extend the system (e.g. with subtyping of sorts, η-conversion, $\ldots$) will certainly break the proof: it doesn't scale nicely.
- \+ Doing proofs within a proof assistant can save you a lot of time when trying to define new systems whose syntax is not clear yet.

That's all folks, thank you for your time !