All you ever wanted to know about Pure Type Systems

Vincent Siles

PPS - INRIA - Ecole Polytechnique

April 1th, 2010

1. What is a Pure Type System ?
2. An Alternative Presentation: PTS with Judgmental Equality
3. Equivalence

Definitions

- PTSs are a way to have general results over families of type systems (System **F**, Calculus of Constructions, Simply-Typed λ-Calculus,...).

Definitions

- PTSs are a way to have general results over families of type systems (System **F**, Calculus of Constructions, Simply-Typed λ-Calculus,...).
- Terms and Contexts:

$$\begin{array}{ccc} A, B, M, N & ::= & x \mid M\ N \mid \lambda x^A.M \\ \Gamma & ::= & [\,] \mid \Gamma, x : A \end{array}$$

Definitions

- PTSs are a way to have general results over families of type systems (System **F**, Calculus of Constructions, Simply-Typed λ-Calculus,...).
- Terms and Contexts:

$$\begin{array}{ccc} A, B, M, N & ::= & s \mid x \mid M\ N \mid \lambda x^A.M \mid \Pi x^A.B \ (\text{or } A \to B) \\ \Gamma & ::= & [\,] \mid \Gamma, x : A \end{array}$$

Definitions

- PTSs are a way to have general results over families of type systems (System **F**, Calculus of Constructions, Simply-Typed λ-Calculus,...).
- Terms and Contexts:
$$\begin{array}{rcl} A, B, M, N & ::= & s \mid x \mid M\ N \mid \lambda x^A.M \mid \Pi x^A.B \text{ (or } A \to B) \\ \Gamma & ::= & [\,] \mid \Gamma, x : A \end{array}$$
- The validity of typing judgments relies on two sets:
  - *Ax* is used to type sorts .
  - *Rel* is used to type functions (or Π-types).

Definitions

- PTSs are a way to have general results over families of type systems (System **F**, Calculus of Constructions, Simply-Typed λ-Calculus,...).
- Terms and Contexts:
$$\begin{array}{ccl} A, B, M, N & ::= & s \mid x \mid M\ N \mid \lambda x^A.M \mid \Pi x^A.B \text{ (or } A \rightarrow B) \\ \Gamma & ::= & [\,] \mid \Gamma, x : A \end{array}$$
- The validity of typing judgments relies on two sets:
  - *Ax* is used to type sorts .
  - *Rel* is used to type functions (or Π-types).
- Reduction :

$$(\lambda x^A.M)\ N \ \xrightarrow{\beta}\ M[N/x] + \text{congruences}$$

Typing Rules

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

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

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

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

Example: How to get the STλC

PTS instantiation for Simply-Typed λ-Calculus:

$S = \{$  ,  $\}$ $Ax = \{($  ,  $)\}$ $Rel = \{($  ,  ,  $)\}$

Example: How to get the STλC

PTS instantiation for Simply-Typed λ-Calculus:

$S = \{$  ,  $\}$ $Ax = \{($  ,  $)\}$ $Rel = \{($  ,  ,  $)\}$

What "types" are allowed in the empty context ?

Example: How to get the STλC

PTS instantiation for Simply-Typed λ-Calculus:

$S = \{\ ,\ \}$ $\quad Ax = \{(\ ,\ )\}$ $\quad Rel = \{(\ ,\ ,\ )\}$

What "types" are allowed in the empty context ?

- $[\,] \vdash\ :\ $

Example: How to get the STλC

PTS instantiation for Simply-Typed λ-Calculus:

$S = \{$$,$$\}$ $\quad Ax = \{($$,$$)\}$ $\quad Rel = \{($$,$$,$$)\}$

What "types" are allowed in the empty context ?

- $[\,] \vdash$  $:$  Ok

Example: How to get the STλC

PTS instantiation for Simply-Typed λ-Calculus:

$S = \{\ ,\ \}$ $\quad Ax = \{(\ ,\ )\}$ $\quad Rel = \{(\ ,\ ,\ )\}$

What “types” are allowed in the empty context ?

- $[\,] \vdash\ :\ $ Ok
- $$\frac{[\,] \vdash\ :\ \qquad x :\ \vdash\ :\ }{[\,] \vdash \Pi x.\ : !!!}$$ $\implies (\ \rightarrow\ )$ is not a valid.

Example: How to get the STλC

PTS instantiation for Simply-Typed λ-Calculus:

$S = \{$🍎$,$👶$\}$ $\quad Ax = \{($🍎$,$👶$)\}$ $\quad Rel = \{($🍎$,$🍎$,$🍎$)\}$

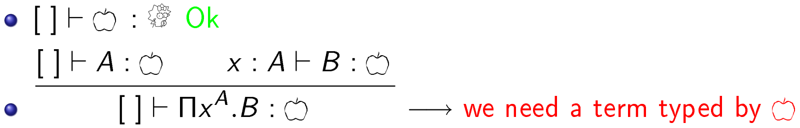
What "types" are allowed in the empty context ?

- $[\,] \vdash$ 🍎 $:$ 👶 Ok
- $$\frac{[\,] \vdash A : 🍎 \qquad x : A \vdash B : 🍎}{[\,] \vdash \Pi x^A . B : 🍎}$$ $\longrightarrow$ we need a term typed by 🍎

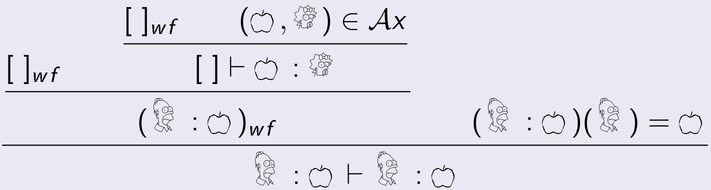
Example: How to get the STλC

PTS instantiation for Simply-Typed λ-Calculus:

What "types" are allowed in the empty context ?

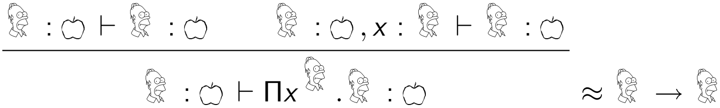


As usual, we introduce *base types* as a fresh variables:

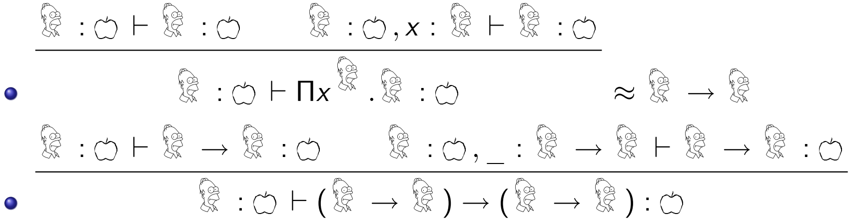


Example: How do get the STλC

Now, we can produce Π-types:

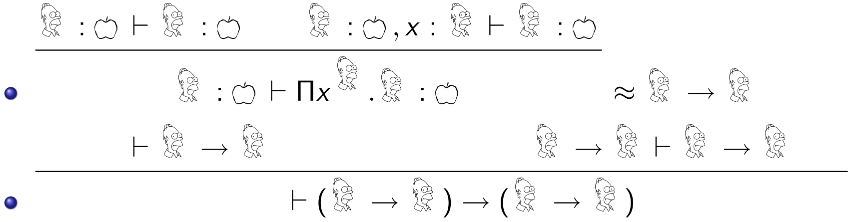
- 

Now, we can produce Π-types:

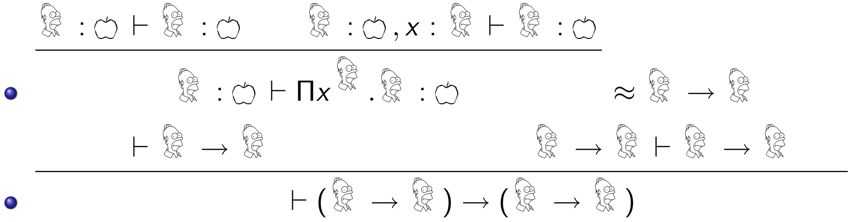


Example: How do get the STλC

Now, we can produce Π-types:

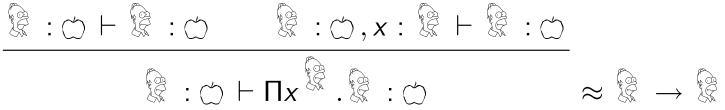
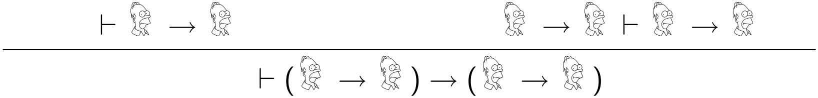


Now, we can produce Π-types:



- $\cdots$

Now, we can produce Π-types:

- 
- 
- $\ldots$

With those rules, we can only build *non-dependent* types.

Facts about β-conversion

Some basic properties of β-reduction:

- Church-Rosser property:

if $M \xrightarrow{\beta}\!\!\!\!\rightarrow N$ and $M \xrightarrow{\beta}\!\!\!\!\rightarrow P$ then there is M' such that $N \xrightarrow{\beta}\!\!\!\!\rightarrow M'$ and $P \xrightarrow{\beta}\!\!\!\!\rightarrow M'$.

- Confluence:

if $M \stackrel{\beta}{\equiv} N$ then there is P such that $M \xrightarrow{\beta}\!\!\!\!\rightarrow P$ and $N \xrightarrow{\beta}\!\!\!\!\rightarrow P$.

- Injectivity of Products:

If $\Pi x^A.B \stackrel{\beta}{\equiv} \Pi x^C.D$ then $A \stackrel{\beta}{\equiv} C$ and $B \stackrel{\beta}{\equiv} D$.

Facts about PTS

- Inversion lemmas :

e.g. if $\Gamma \vdash \lambda x^A.M : T$ then there are s, t, u and B such that

- $(s, t, u) \in \mathcal{R}el$, $T \stackrel{\beta}{\equiv} \Pi x^A.B$
- $\Gamma \vdash A : s$ and $\Gamma, x : A \vdash B : t$ and $\Gamma, x : A \vdash M : B$.

Facts about PTS

- Inversion lemmas :

e.g. if $\Gamma \vdash \lambda x^A.M : T$ then there are s, t, u and B such that

- $(s,t,u) \in \mathcal{R}el$, $T \overset{\beta}{\equiv} \Pi x^A.B$
- $\Gamma \vdash A : s$ and $\Gamma, x : A \vdash B : t$ and $\Gamma, x : A \vdash M : B$.

- Correctness of types :

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

Facts about PTS

- Inversion lemmas :

e.g. if $\Gamma \vdash \lambda x^A.M : T$ then there are s, t, u and B such that

- $(s, t, u) \in \mathcal{R}el,\ T \overset{\beta}{\equiv} \Pi x^A.B$
- $\Gamma \vdash A : s$ and $\Gamma, x : A \vdash B : t$ and $\Gamma, x : A \vdash M : B$.

- Correctness of types :

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

- Subject Reduction:

If $\Gamma \vdash M : T$ and $M \xrightarrow{\beta} M'$ then $\Gamma \vdash M' : T$.

Facts about PTS

- Shape of Types (Jutting [93]):

If $\Gamma \vdash M : A$ and $\Gamma \vdash M : B$, then

- either $A \overset{\beta}{\equiv} B$
- or $A \overset{\beta}{\twoheadrightarrow} \Pi x^{A_1} \ldots x^{A_n}.s$ and $B \overset{\beta}{\twoheadrightarrow} \Pi x^{A_1} \ldots x^{A_n}.t$

Facts about PTS

- Shape of Types (Jutting [93]):

If $\Gamma \vdash M : A$ and $\Gamma \vdash M : B$, then

- either $A \overset{\beta}{\equiv} B$
- or $A \overset{\beta}{\twoheadrightarrow} \Pi x^{A_1} \dots x^{A_n}.s$ and $B \overset{\beta}{\twoheadrightarrow} \Pi x^{A_1} \dots x^{A_n}.t$

- Not-Fact Normalization: there are some non-terminating PTS (🍏:🍏).

Facts about PTS

- Shape of Types (Jutting [93]):

If $\Gamma \vdash M : A$ and $\Gamma \vdash M : B$, then

- either $A \overset{\beta}{\equiv} B$
- or $A \overset{\beta}{\twoheadrightarrow} \Pi x^{A_1} \ldots x^{A_n}.s$ and $B \overset{\beta}{\twoheadrightarrow} \Pi x^{A_1} \ldots x^{A_n}.t$

- Not-Fact Normalization: there are some non-terminating PTS (🍏:🍏).
- Not-Fact Type Checking / Inference : type checking dependent types is undecidable.

Facts about PTS

- Shape of Types (Jutting [93]):

If $\Gamma \vdash M : A$ and $\Gamma \vdash M : B$, then

- either $A \overset{\beta}{\equiv} B$
- or $A \overset{\beta}{\twoheadrightarrow} \Pi x^{A_1} \ldots x^{A_n}.s$ and $B \overset{\beta}{\twoheadrightarrow} \Pi x^{A_1} \ldots x^{A_n}.t$

- Not-Fact Normalization: there are some non-terminating PTS ($\circlearrowleft : \circlearrowleft$).
- Not-Fact Type Checking / Inference : type checking dependent types is undecidable.
- Not-Fact Expansion Postponement : replace conversion with reduction only:

If $\Gamma \vdash M : T$ then $\Gamma \vdash' M : T'$ and $T \overset{\beta}{\twoheadrightarrow} T'$.

Why do we want a typed equality ?

- In the conversion rules the intermediate steps are not checked.

$$\frac{\Gamma \vdash M : A \qquad A \overset{\beta}{\equiv} B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

Why do we want a typed equality ?

- In the conversion rules the intermediate steps are not checked.

$$\frac{\Gamma \vdash M : A \qquad A \overset{\beta}{\equiv} B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

- β-equality is all about *program computation*, where types are useless.

Why do we want a typed equality ?

- In the conversion rules the intermediate steps are not checked.

$$\frac{\Gamma \vdash M : A \qquad A \overset{\beta}{\equiv} B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

- β-equality is all about *program computation*, where types are useless.
- Other kind of equalities may depend on types (η-expansion, external axioms).

Why do we want a typed equality ?

- In the conversion rules the intermediate steps are not checked.

$$\frac{\Gamma \vdash M : A \qquad A \overset{\beta}{\equiv} B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

- β-equality is all about *program computation*, where types are useless.
- Other kind of equalities may depend on types (η-expansion, external axioms).
- So, what if we check each conversion step during conversion ?

Why do we want a typed equality ?

- In the conversion rules the intermediate steps are not checked.

$$\frac{\Gamma \vdash M : A \qquad A \overset{\beta}{\equiv} B \qquad \Gamma \vdash B : s}{\Gamma \vdash M : B}$$

- β-equality is all about *program computation*, where types are useless.
- Other kind of equalities may depend on types (η-expansion, external axioms).
- So, what if we check each conversion step during conversion ?

$\hookrightarrow$ all this lead to the definition of PTS *with Judgmental Equality*.

PTSe typing rules (1)

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

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

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

$$\frac{\Gamma \vdash_e M : \Pi x^A.B \qquad \Gamma \vdash_e N : A}{\Gamma \vdash_e MN : B[x/N]} \qquad \frac{\Gamma \vdash_e M : A \qquad \Gamma \vdash_e A = B : s}{\Gamma \vdash_e M : B}$$

PTSe typing rules (1)

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

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

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

$$\frac{\Gamma \vdash_e M : \Pi x^A.B \qquad \Gamma \vdash_e N : A}{\Gamma \vdash_e MN : B[x/N]} \qquad \frac{\Gamma \vdash_e M : A \qquad \boxed{\Gamma \vdash_e A = B : s}}{\Gamma \vdash_e M : B}$$

PTSe typing rules (2)

$$\frac{\Gamma_{wf_e} \qquad (s,t) \in \mathcal{A}x}{\Gamma \vdash_e s = s : t} \qquad \frac{\Gamma_{wf_e} \qquad \Gamma(x) = A}{\Gamma \vdash_e x = x : A}$$

$$\frac{\Gamma \vdash_e M = M' : \Pi x^A.B \qquad \Gamma \vdash_e N = N' : A}{\Gamma \vdash_e MN = M'N' : B[x/N]}$$

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

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

PTSe typing rules (3)

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

$$\frac{\Gamma \vdash_e M : A}{\Gamma \vdash_e M = M : A} \qquad \frac{\Gamma \vdash_e M = N : A}{\Gamma \vdash_e N = M : A} \qquad \frac{\Gamma \vdash_e M = N : A \quad \Gamma \vdash_e N = P : A}{\Gamma \vdash_e M = P : A}$$

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

Are both systems the same ?

Easy part of the equivalence

We proove by mutual induction that

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

Easy part of the equivalence

We proove by mutual induction that

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

Here we just "loose" some information, nothing complicated.

Hard Part

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 $M \stackrel{\beta}{\equiv} N$ then $\Gamma \vdash_e M = N : T$.
- If Γ_{wf} then Γ_{wf_e}.

Hard Part

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 $M \stackrel{\beta}{\equiv} N$ then $\Gamma \vdash_e M = N : T$.
- If Γ_{wf} then Γ_{wf_e}.

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

Hard Part

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 $M \stackrel{\beta}{\equiv} N$ then $\Gamma \vdash_e M = N : T$.
- If Γ_{wf} then Γ_{wf_e}.

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

But can we ?

How do we do this ?

$$\Gamma \vdash M : T \qquad M \qquad \stackrel{\beta}{\equiv} \qquad N \qquad \Gamma \vdash N : T$$

How do we do this ?

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

How do we do this ?

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

- P is welltyped in PTS by *Subject Reduction*.

How do we do this ?

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

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

How do we do this ?

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

- P is welltyped in PTS by *Subject Reduction.*
- Is P welltyped in PTSe ?
- How do we type $M = P$ and $N = P$ in PTSe ?

The need of Subject Reduction

To do so, we need to proove that PTSe have the *Subject Reduction* property:

Subject Reduction:

If $\Gamma \vdash_e M : T$ and $M \overset{\beta}{\twoheadrightarrow} N$, then $\Gamma \vdash_e M = N : T$.

The need of Subject Reduction

To do so, we need to proove that PTSe have the *Subject Reduction* property:

Subject Reduction:

If $\Gamma \vdash_e M : T$ and $M \stackrel{\beta}{\twoheadrightarrow} N$, then $\Gamma \vdash_e M = N : T$.

But to proove this, we need Π-injectivity, which is still an open question for PTSe since it relies on *Confluency*,

The need of Subject Reduction

To do so, we need to proove that PTSe have the *Subject Reduction* property:

Subject Reduction:

If $\Gamma \vdash_e M : T$ and $M \overset{\beta}{\twoheadrightarrow} N$, then $\Gamma \vdash_e M = N : T$.

But to proove this, we need Π-injectivity, which is still an open question for PTSe since it relies on *Confluency*, which relies on *Subject Reduction*,

The need of Subject Reduction

To do so, we need to proove that PTSe have the *Subject Reduction* property:

Subject Reduction:

If $\Gamma \vdash_e M : T$ and $M \stackrel{\beta}{\twoheadrightarrow} N$, then $\Gamma \vdash_e M = N : T$.

But to proove this, we need Π-injectivity, which is still an open question for PTSe since it relies on *Confluency*, which relies on *Subject Reduction*,which relies on Π-injectivity,

The need of Subject Reduction

To do so, we need to proove that PTSe have the *Subject Reduction* property:

Subject Reduction:

If $\Gamma \vdash_e M : T$ and $M \overset{\beta}{\twoheadrightarrow} N$, then $\Gamma \vdash_e M = N : T$.

But to proove this, we need Π-injectivity, which is still an open question for PTSe since it relies on *Confluency*, which relies on *Subject Reduction*,which relies on Π-injectivity,which relies on ...

Current status of the equivalence

We only some partials results:

- for functional PTS : R. Adams [06] “Pure Type Systems with Judgmental Equality”.

Current status of the equivalence

We only some partials results:

- for functional PTS : R. Adams [06] “Pure Type Systems with Judgmental Equality”.
- for semi-full and full PTS : V. Siles and H. Herbelin [10] “Equality is typable in Semi-Full Pure Type Systems”.

Current status of the equivalence

We only some partials results:

- for functional PTS : R. Adams [06] “Pure Type Systems with Judgmental Equality”.
- for semi-full and full PTS : V. Siles and H. Herbelin [10] “Equality is typable in Semi-Full Pure Type Systems”.
- But the question is still open for general PTS !

That's all folks !

Thank you for your time.