CoqMTU: a higher-order type theory with a predicative hierarchy of universes parameterized by a decidable first-order theory

Wang Qian

Joint work with Bruno Barras, Jean-Pierre Jouannaud and Pierre-Yves Strub

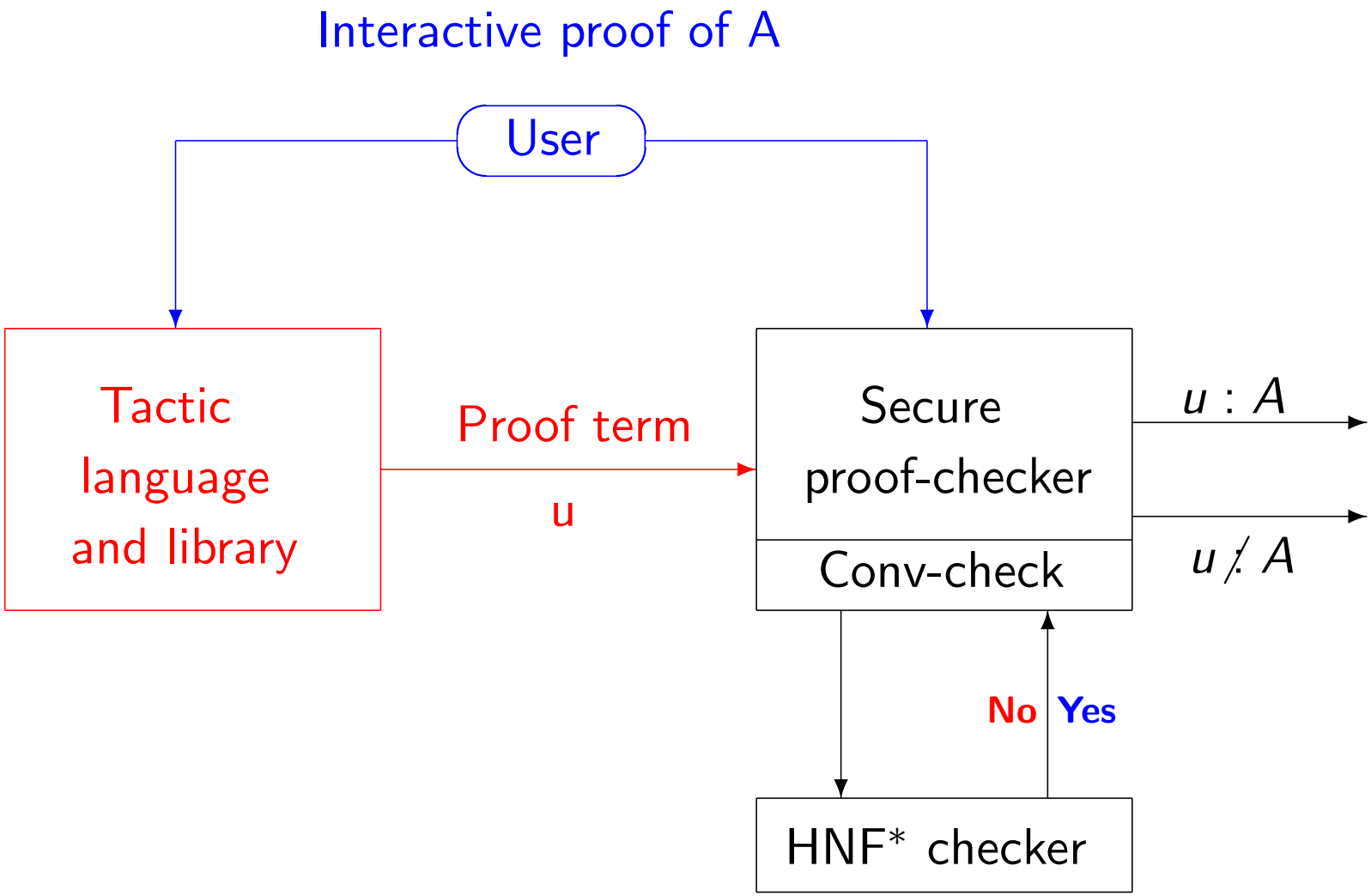
Twenty-Sixth Annual IEEE Symposium on

LOGIC IN COMPUTER SCIENCE

Content

Content

Architecture of Coq



Dependent Words in Coq : Natural Definition

- Define a dependent word inductively:

```
Inductive dword : nat -> Type :=
| empty : dword 0
| singleton : T -> dword 1
| append : forall n p, dword n -> dword p -> dword (n + p).
```

- Define a reverse function on dependent words:

```
Fixpoint rev n (xs : dword n) :=
match xs in dword n return dword n with
| empty => empty
| singleton x => singleton x
| append n1 n2 w1 w2 => append (rev w2) (rev w1)
end.
```

- does not type-check : conversion fails in the third branch:

$$dword\ (n1 + n2) \not\simeq_{\beta\iota} dword\ (n2 + n1)$$

Standard Solution in Coq

- Define a cast function:

```
Definition cast: forall n1 n2, n1=n2->dword n1->dword n2.
Proof. intros n1 n2 E xs;subst n2;exact xs. Defined.
```

- Use cast function to define reverse:

```
Fixpoint rev n (xs : dword n) :=
match xs in dword n return dword n with
| empty => empty
| singleton x => singleton x
| append n1 n2 w1 w2 =>
      cast (addC n2 n1) (append (rev w2) (rev w1))
end .
```

- This solution has several drawbacks:
  - ⋆ definitions get more complicated, and
  - ⋆ proofs involving reverse get complicated.

Example - A Better Solution : CoqMT

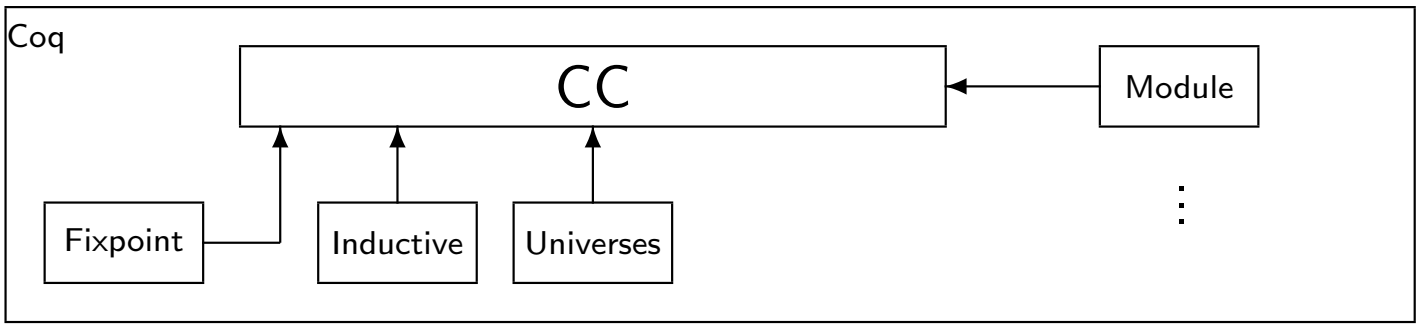
- Embed first-order theories into the conversion checker:

$$dword\ (n1 + n2) \simeq_{\beta\iota\mathcal{T}} dword\ (n2 + n1)$$

- Then,
  - the natural definition of reverse type-checks,
  - proofs involving reverse become natural.

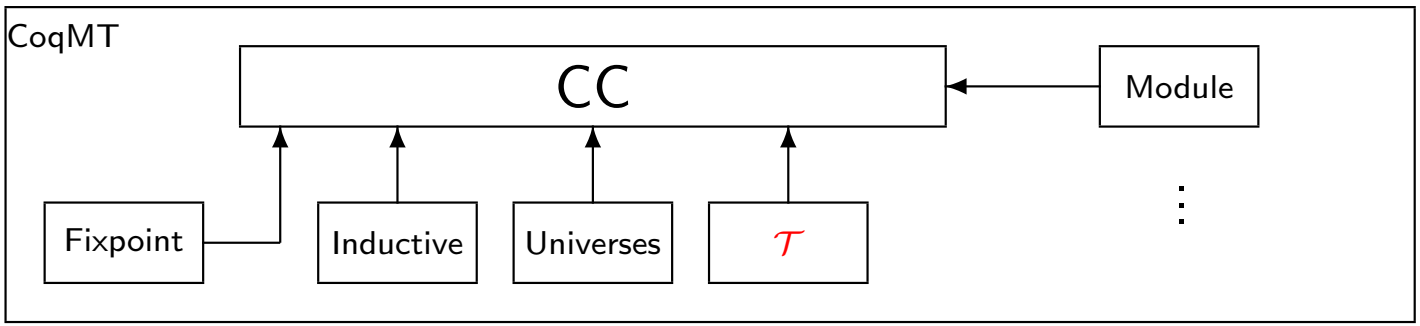
Coq : A Rich and Complex Type Theory

- Coq is based on the Calculus of Constructions (CC),
- but also incorporates:
  - ⋆ inductive types, co-inductive types,
  - ⋆ fixpoint definitions,
  - ⋆ a predicative hierarchy of universes,
  - ⋆ a module system,
  - ⋆ ...



Coq : A Rich and Complex Type Theory

- Coq is based on the Calculus of Constructions (CC),
- but also incorporates:
  - ⋆ inductive types, co-inductive types,
  - ⋆ fixpoint definitions,
  - ⋆ a predicative hierarchy of universes,
  - ⋆ a module system,
  - ⋆ ...
- CoqMT also incorporates dynamic loading of first-order theories:



Soundness of Coq or CoqMT

- CIC : Decidability of Type Checking (DTC) paper proof

  B. Werner, "Une théorie des constructions inductives", Ph.D. dissertation, 1994.
- ECC : DTC paper proof of CC + Universes (plus sums, with weak-elim)

  Z. Luo, "ECC, an extended calculus of constructions", in LICS, 1989
- UTT : DTC paper proof of CIC+ONE predicative universe (with strong-elim)

  H. Goguen, "The metatheory of utt", in TYPES, 1994
- CIC+Universes : consistency paper proof

  B. Werner, "Sets in types, types in sets", in TACS, 1997
- CoqMT : DTC paper proof of CIC + $\mathcal{T}$

  P.-Y. Strub, "Coq Modulo Theory", in CSL, 2010

Soundness of Coq or CoqMT

- CIC : Decidability of Type Checking (DTC) paper proof

  B. Werner, "Une théorie des constructions inductives", Ph.D. dissertation, 1994.
- ECC : DTC paper proof of CC + Universes (plus sums, with weak-elim)

  Z. Luo, "ECC, an extended calculus of constructions", in LICS, 1989
- UTT : DTC paper proof of CIC+ONE predicative universe (with strong-elim)

  H. Goguen, "The metatheory of utt", in TYPES, 1994
- CIC+Universes : consistency paper proof

  B. Werner, "Sets in types, types in sets", in TACS, 1997
- CoqMT : DTC paper proof of CIC + $\mathcal{T}$

  P.-Y. Strub, "Coq Modulo Theory", in CSL, 2010

No paper proof of entire Coq or CoqMT yet ...

Reliability of Coq's kernel

- CIC : formal proof of DTC

  B. Barras, "Auto-validation d'un systéme de preuves avec familles inductives", 1999

- CIC modulo $\mathcal{T}$: formal proof of DTC meta theory (assuming SN)

  P.-Y. Strub, "Coq Modulo Theory", in CSL, 2010

Reliability of Coq's kernel

- CIC : formal proof of DTC

  B. Barras, "Auto-validation d'un systéme de preuves avec familles inductives", 1999
- CIC modulo $\mathcal{T}$: formal proof of DTC meta theory (assuming SN)

  P.-Y. Strub, "Coq Modulo Theory", in CSL, 2010

No formal proof of Coq or CoqMT yet ...

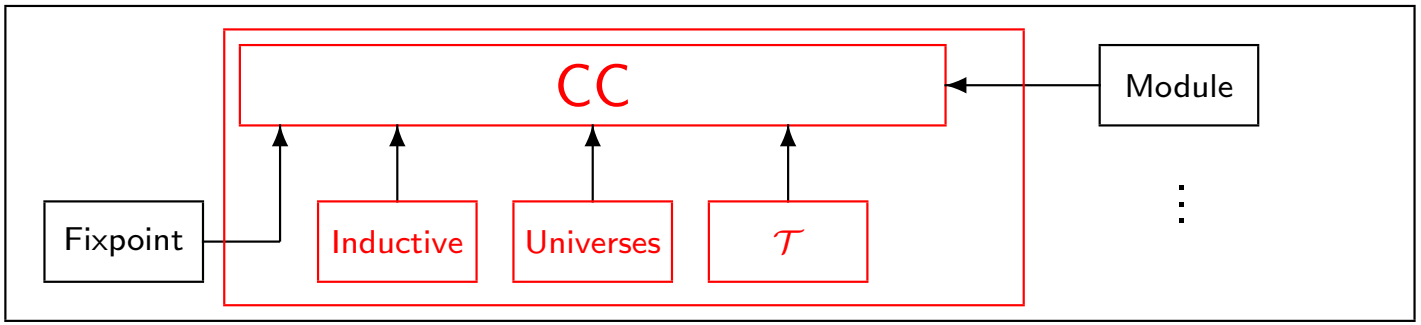
Program

- Give a complete paper proof of CoqMT.
- Give a complete formal proof of CoqMT.
- Develop a new kernel for Coq form the formal proof.

Content

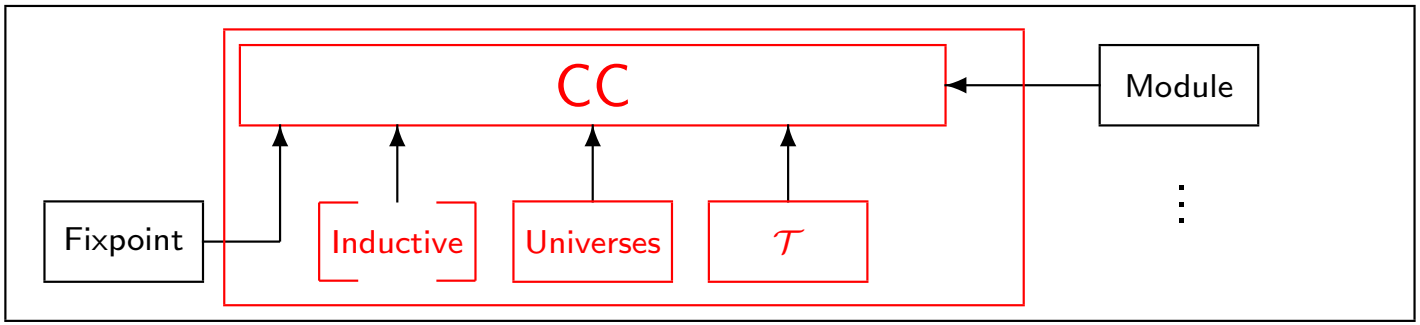
Contribution:The first step of our program

- Notion of axiomatic first-order inductive type,
- Definition of CoqMTU,



Contribution:The first step of our program

- Notion of axiomatic first-order inductive type,
- Definition of CoqMTU,
- Decidability proof of type-checking in presence of weak elimination.



Content

Axiomatic First-Order Equality Theory $\mathcal{T}$

Specified by :

- its constructed symbols $\mathcal{C}$, defined symbols $\mathcal{D}$, variables $\mathcal{X}$,
- its (decidable) abstract equivalence relation $\leftrightarrow_{\mathcal{T}}$,
- axioms constraining the relation $\leftrightarrow_{\mathcal{T}}$:

Non-triviality.	$\mathcal{T}(\mathcal{C})$ contains at least two different terms.
Freeness.	For all constructor term s, t, $s \leftrightarrow_{\mathcal{T}} t$, then $s = t$.
Completeness.	For any $t \in \mathcal{T}(\mathcal{C}, \mathcal{D})$, there exists $u \in \mathcal{T}(\mathcal{C})$ s.t. $t \leftrightarrow_{\mathcal{T}} u$.

Axiomatic First-Order Equality Theory $\mathcal{T}$

Specified by :

- its constructed symbols $\mathcal{C}$, defined symbols $\mathcal{D}$, variables $\mathcal{X}$,
- its (decidable) abstract equivalence relation $\leftrightarrow_{\mathcal{T}}$,
- axioms constraining the relation $\leftrightarrow_{\mathcal{T}}$:

Non-triviality.	$\mathcal{T}(\mathcal{C})$ contains at least two different terms.
Freeness.	For all constructor term s, t, $s \leftrightarrow_{\mathcal{T}} t$, then $s = t$.
Completeness.	For any $t \in \mathcal{T}(\mathcal{C}, \mathcal{D})$, there exists $u \in \mathcal{T}(\mathcal{C})$ s.t. $t \leftrightarrow_{\mathcal{T}} u$.

Example - Natural numbers

$\mathcal{C} = \{\mathbf{0}, \mathbf{S}\}$, $\mathcal{D} = \{+\}$

Axiomatic First-Order Equality Theory $\mathcal{T}$

Specified by :

- its constructed symbols $\mathcal{C}$, defined symbols $\mathcal{D}$, variables $\mathcal{X}$,
- its (decidable) abstract equivalence relation $\leftrightarrow_{\mathcal{T}}$,
- axioms constraining the relation $\leftrightarrow_{\mathcal{T}}$:

Non-triviality.	$\mathcal{T}(\mathcal{C})$ contains at least two different terms.
Freeness.	For all constructor term s, t, $s \leftrightarrow_{\mathcal{T}} t$, then $s = t$.
Completeness.	For any $t \in \mathcal{T}(\mathcal{C}, \mathcal{D})$, there exists $u \in \mathcal{T}(\mathcal{C})$ s.t. $t \leftrightarrow_{\mathcal{T}} u$.

Example - Natural numbers

$\mathcal{C} = \{\mathbf{0}, \mathbf{S}\}$, $\mathcal{D} = \{+\}$

'Any' first-order algebra is an inductive type in CoqMTU.

Pseudo-Terms

$$
\begin{array}{lll}
t, u, T, U ::= & \mathbf{Prop} \mid \mathbf{Type}_{j>0} & \text{(Universes)} \\
& \mid \mathcal{V}ar \mid t\ u \mid \lambda[x : U].\, t \mid \forall(x : U).\, T & \text{(CC)} \\
& \mid o \mid \mathcal{C} \mid \mathcal{D} \mid \textsc{Elim}_o(T, \overrightarrow{u}, t) & (\mathcal{T})
\end{array}
$$

Reductions

- β-reduction is defined as usual:

$$(\lambda[x : T].\, u)t \to_\beta u\{x \mapsto t\}$$

Reductions

- β-reduction is defined as usual:

$$(\lambda[x : T].\, u)t \to_\beta u\{x \mapsto t\}$$

- ι-reduction is the same as in CIC.

Reductions

- β-reduction is defined as usual:

$$(\lambda[x : T].\, u)t \to_\beta u\{x \mapsto t\}$$

- ι-reduction is the same as in CIC.

- $\iota_{\mathcal{T}}$-reduction generalizes pure ι-reduction.

  For our example of natural number:

$$\textsc{Elim}_{nat}(Q, f_{\mathbf{0}}, f_{\mathbf{S}}, t) \to_{\iota_{\mathcal{T}}} \begin{cases} f_{\mathbf{0}} & (1) \\ f_{\mathbf{S}}\, u\, \textsc{Elim}_{nat}(Q, f_{\mathbf{0}}, f_{\mathbf{S}}, u) & (2) \end{cases}$$

  provided

  - $t \leftrightarrow_{\mathcal{T}} \mathbf{0}$ for case (1), and
  - exists u, $t \leftrightarrow_{\mathcal{T}} \mathbf{S}\ u$ and $\mathbf{S}\, u$ simplifies t for case (2).

Conversion

- $\rightarrow$ is one of $\rightarrow_\beta$, $\rightarrow_\iota$, $\rightarrow_{\iota_\mathcal{T}}$.
- The conversion relation $\simeq$ is the reflexive, symmetric and transitive closure of $\rightarrow \cup \leftrightarrow_\mathcal{T}$.

Typing: Rules for CC

$$\frac{\Gamma \vdash T : \mathbf{Type}_j}{\Gamma, x : T \vdash x : T} \qquad \frac{\Gamma \vdash t : T, \quad \Gamma \vdash V : \mathbf{Type}_j}{\Gamma, x : V \vdash t : T}$$

$$\frac{\Gamma, (x : U) \vdash t : V \quad \Gamma \vdash \forall (x : U).\, V : \mathbf{Type}_j}{\Gamma \vdash \lambda [x : U].\, t : \forall (x : U).\, V}$$

$$\frac{\Gamma \vdash u : \forall (x : U).\, V \quad \Gamma \vdash v : U}{\Gamma \vdash u\, v : V[x \mapsto v]}$$

$$\frac{\Gamma \vdash t : U \quad \Gamma \vdash U' : \mathbf{Type}_j \quad U \simeq U'}{\Gamma \vdash t : U'}$$

Typing: Rules for CC

$$\frac{\Gamma \vdash T : \mathbf{Type}_j}{\Gamma, x : T \vdash x : T} \qquad \frac{\Gamma \vdash t : T, \quad \Gamma \vdash V : \mathbf{Type}_j}{\Gamma, x : V \vdash t : T}$$

$$\frac{\Gamma, (x : U) \vdash t : V \quad \Gamma \vdash \forall (x : U).\, V : \mathbf{Type}_j}{\Gamma \vdash \lambda [x : U].\, t : \forall (x : U).\, V}$$

$$\frac{\Gamma \vdash u : \forall (x : U).\, V \quad \Gamma \vdash v : U}{\Gamma \vdash u\, v : V[x \mapsto v]}$$

$$\frac{\Gamma \vdash t : U \quad \Gamma \vdash U' : \mathbf{Type}_j \quad U \simeq U'}{\Gamma \vdash t : U'}$$

Typing: Rules for Universes

$$\frac{}{\vdash \mathbf{Type}_j : \mathbf{Type}_{j+1}} \qquad \frac{\Gamma \vdash T : \mathbf{Type}_j}{\Gamma \vdash T : \mathbf{Type}_{j+1}}$$

$$\frac{\Gamma \vdash U : \mathbf{Type}_j \quad \Gamma, x : U \vdash V : \mathbf{Type}_0}{\Gamma \vdash \forall (x : U).\, V : \mathbf{Type}_0}$$

$$\frac{\Gamma \vdash U : \mathbf{Type}_i \quad \Gamma, x : U \vdash V : \mathbf{Type}_{j \neq 0}}{\Gamma \vdash \forall (x : U).\, V : \mathbf{Type}_{max(i,j)}}$$

Typing: Rules for Theory $\mathcal{T}$

$$\frac{}{\vdash \mathbf{nat} : \mathbf{Prop}}$$

$$\frac{}{\vdash \mathbf{0} : \mathbf{nat}} \qquad \frac{}{\vdash \mathbf{S} : \mathbf{nat} \to \mathbf{nat}} \qquad \frac{}{\vdash + : \mathbf{nat} \to \mathbf{nat} \to \mathbf{nat}}$$

$$\frac{\Gamma \vdash t : \mathbf{nat} \quad \Gamma \vdash P : \forall (x : \mathbf{nat}).\, \mathbf{Prop} \quad \Gamma \vdash f_{\mathbf{0}} : P\ \mathbf{0} \quad \Gamma \vdash f_{\mathbf{S}} : \forall (x : \mathbf{nat}).\, (P\ x \to P\ (\mathbf{S}\ x))}{\Gamma \vdash \textsc{wElim}_{nat}(P, f_{\mathbf{0}}, f_{\mathbf{S}}, t) : P\ t}$$

Typing: Rules for Theory $\mathcal{T}$

$$\frac{}{\vdash \mathbf{nat} : \mathbf{Prop}}$$

$$\frac{}{\vdash \mathbf{0} : \mathbf{nat}} \qquad \frac{}{\vdash \mathbf{S} : \mathbf{nat} \to \mathbf{nat}} \qquad \frac{}{\vdash + : \mathbf{nat} \to \mathbf{nat} \to \mathbf{nat}}$$

$$\frac{\Gamma \vdash t : \mathbf{nat} \quad \Gamma \vdash P : \forall (x : \mathbf{nat}).\, \mathbf{Prop} \quad \Gamma \vdash f_{\mathbf{0}} : P\ \mathbf{0} \quad \Gamma \vdash f_{\mathbf{S}} : \forall (x : \mathbf{nat}).\, (P\ x \to P\ (\mathbf{S}\ x))}{\Gamma \vdash \textsc{wElim}_{nat}(P, f_{\mathbf{0}}, f_{\mathbf{S}}, t) : P\ t}$$

Typing: Rules for Theory $\mathcal{T}$

$$\frac{}{\vdash \mathbf{nat} : \mathbf{Prop}}$$

$$\frac{}{\vdash \mathbf{0} : \mathbf{nat}} \qquad \frac{}{\vdash \mathbf{S} : \mathbf{nat} \to \mathbf{nat}} \qquad \frac{}{\vdash + : \mathbf{nat} \to \mathbf{nat} \to \mathbf{nat}}$$

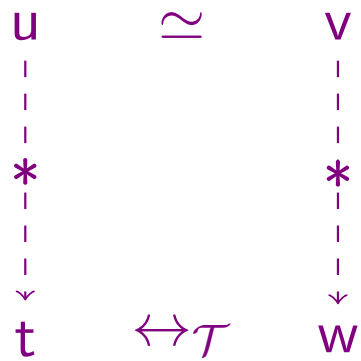
$$\frac{\Gamma \vdash t : \mathbf{nat} \quad \Gamma \vdash P : \forall (x : \mathbf{nat}).\, \mathbf{Prop} \quad \Gamma \vdash f_{\mathbf{0}} : P\ \mathbf{0} \quad \Gamma \vdash f_{\mathbf{S}} : \forall (x : \mathbf{nat}).\, (P\ x \to P\ (\mathbf{S}\ x))}{\Gamma \vdash \textsc{wElim}_{nat}(P, f_{\mathbf{0}}, f_{\mathbf{S}}, t) : P\ t}$$

Content

Main properties

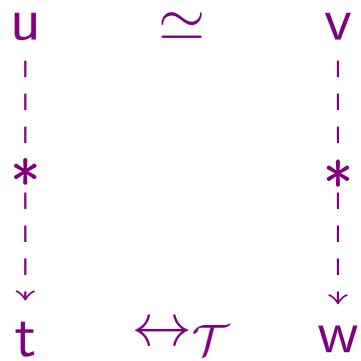
- Church - Rosser
- Subject Reduction
- Strong Normalization
- Consistency
- Decidability of Type Checking

Church-Rosser



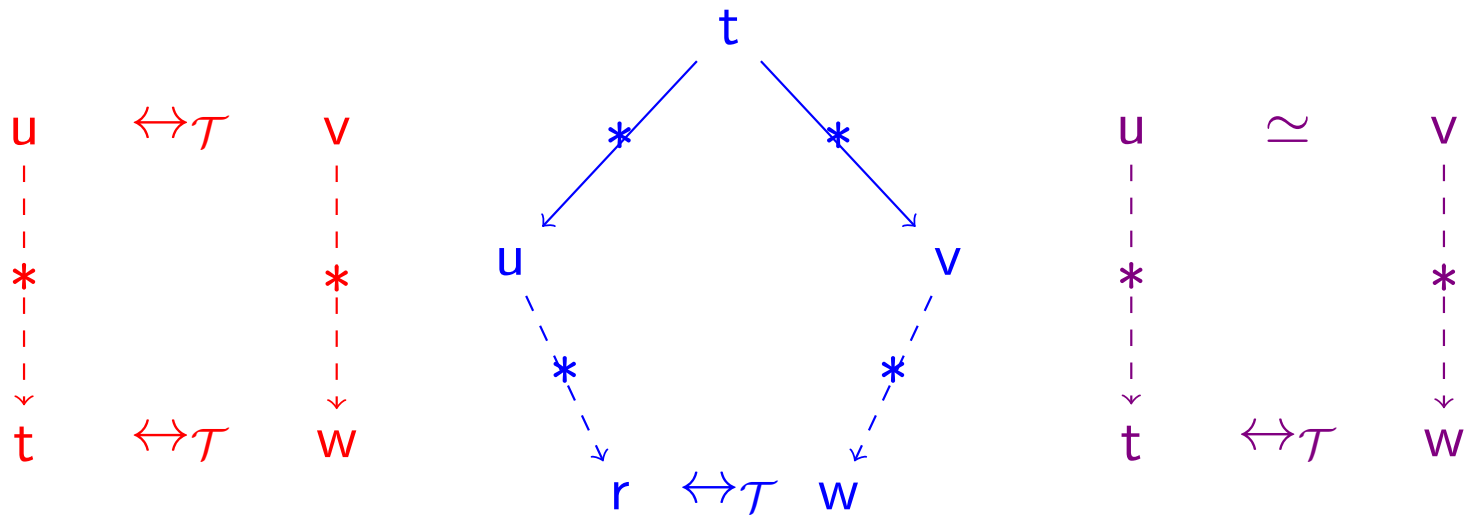
Church-Rosser

- Step 1 : define a parallel reduction $\Rightarrow$ as usual, and then prove two properties relating $\rightarrow$ and $\Rightarrow$:
  - ⋆ For all u, v such that $u \rightarrow v$, $u \Rightarrow v$.
  - ⋆ For all u, v such that $u \Rightarrow v$, $u \rightarrow^* v$.



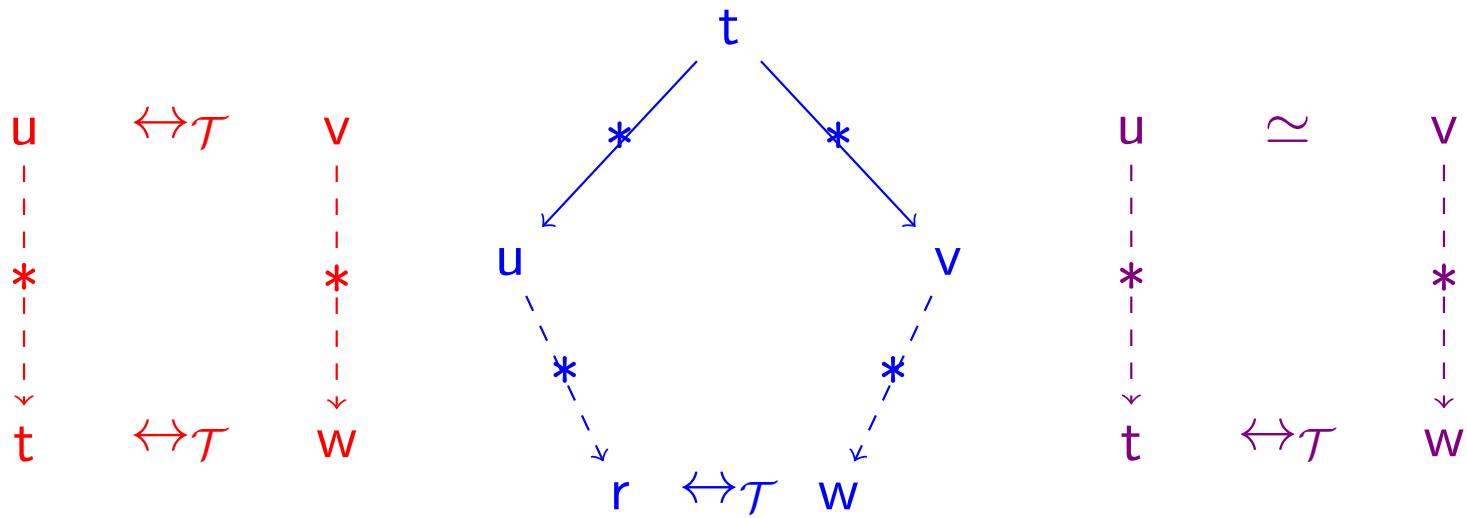
Church-Rosser

- Step 1 : define a parallel reduction $\Rightarrow$ as usual, and then prove two properties relating $\rightarrow$ and $\Rightarrow$:
  - ⋆ For all u, v such that $u \rightarrow v$, $u \Rightarrow v$.
  - ⋆ For all u, v such that $u \Rightarrow v$, $u \rightarrow^* v$.
- Step 2 : prove coherence and confluence of $\rightarrow$ by proving coherence and confluence of $\Rightarrow$ and Step 1:



Church-Rosser

- Step 1 : define a parallel reduction $\Rightarrow$ as usual, and then prove two properties relating $\rightarrow$ and $\Rightarrow$:
  - ⋆ For all u, v such that $u \rightarrow v$, $u \Rightarrow v$.
  - ⋆ For all u, v such that $u \Rightarrow v$, $u \rightarrow^* v$.
- Step 2 : prove coherence and confluence of $\rightarrow$ by proving coherence and confluence of $\Rightarrow$ and Step 1:
- Step 3 : prove Church-Rosser by induction on the the length of $\simeq$.



Subject Reduction

Subject Reduction

Let $\Gamma \vdash t : T$ and $t \to t'$. Then $\Gamma \vdash t' : T$.

- To prove subject reduction, properties of environment, such as weakening and strengthening, and inversion are needed.
- New : prove $\mathcal{T}$-Irrelevance.

  Let $\Gamma \vdash t : T$ and t' simplifies t. Then $\Gamma \vdash t' : T$.
- Subject reduction follows by simultaneous induction of:
  - $\Gamma \to \Gamma'$, then $\Gamma' \vdash t : T$; and
  - $t \to t'$, then $\Gamma \vdash t' : T$

Strong Normalization

Strong Normalization

Let $\Gamma \vdash t : T$. Then t is strongly normalizable.

- Strong normalization is proved using Tait and Girard's method:
  - It is sufficient to prove: $\vdash t : T$, then t is strongly normalizable.
  - Prove $t = t\theta \in \mathit{Eval}(T) \in \mathit{Val}(T\theta)$.
- A complexity measure Δ (inspired from Luo) is needed to define *Val* (values of terms).

Consistency and DTC

- Consistency: $\not\vdash M : \forall(P : \mathbf{Prop}).\, P$.
  - By contradiction, and using strong normalization.
  - Difficulty comes from algebraic inductive type.

Consistency and DTC

- Consistency: $\not\vdash M : \forall(P : \mathbf{Prop}).\, P$.
  - By contradiction, and using strong normalization.
  - Difficulty comes from algebraic inductive type.
- Decidability of Type Checking: By
  - decidability and correctness of type inference, and
  - decidability of $\simeq$ and $\preceq$, which are the result of strong normalization, subject reduction and Church-Rosser.

Content

Work Finished

- Defined an abstract calculus, which contains:

  CC, Universes and an embedded first-order theory $\mathcal{T}$.

- Prove :
  - confluence and subject reduction,
  - strong normalization, consistency and DTC in presence of weak elimination.
- For more details:
  - Full version : A draft at `http://formes.asia/people/Wang.Qian`
  - A partial formal proof in Coq : `http://strub.nu/research/coqmt`
  - Implementation : `http://git.strub.nu/git/coqmt`

Work to Do

- Strong normalization in presence of strong elimination.
  - Important ongoing recent advance of Bruno Barras.
- Consider other important features (of Coq), such as:
  - fixpoint definitions,
  - implicit arguments,
  - module system,
  - type classes,
  - ...

Questions?