

Explicit Cumulativity in CC_ω

Vincent R.B. Blazy, Hugo Herbelin & Pierre Letouzey

5 avril 2024

1 Introduction : Cumulativity

2 The Link with Extraction

- ## 3 CC_{ω}^{sub}
- CC_{ω}^{sub} : The Syntax
 - CC_{ω}^{sub} : The Rules
 - CC_{ω}^{sub} : The Key Typing Rule & What Happens to The Example
 - CC_{ω}^{sub} : The Results

4 CC_{ω}^{sub} : The Algorithms

5 Work to be Done

Introduction : Cumulativity

A Cumulative Universe Hierarchy

In CIC , or already without inductives – in CC_ω – we have :

A hierarchy of Universes :

$$\text{Prop} : \text{Type}_1 : \dots : \text{Type}_i : \text{Type}_{i+1} : \dots$$

+ Implicit subtyping between them, the *cumulativity* of the hierarchy [The20, Let04] :

$$\text{Prop} \leq \text{Type}_1 \leq \dots \leq \text{Type}_i \leq \text{Type}_{i+1} \leq \dots$$

+

$$\frac{\Gamma \vdash t : A \quad A \leq B}{\Gamma \vdash t : B} \text{ sub}$$

Why Cumulativity :

- Formalizes the set-theoretic intuition of inclusion of smaller universes in bigger ones in a hierarchization of a unique global and *informal* "superuniverse", present for well-founded sets as the Von Neumann cumulative hierarchy in ZFC ;
- Allows for filling data structures with inhabitants not of a type but of a proposition : one may sometimes need lists of proof terms of a proposition, or to evaluate polymorphic functions on propositions, without having to redefine Prop-avatars (length of a list...) ;
- Allows for talking about equality of proof terms (in UIP, etc.) ;
- Avoids having to define multiple rules differing only by substitution of Prop for Type, typically :

Axiom funext : $\forall A B : \text{Type}, \forall f g : A \rightarrow B, (\forall x, f x = g x) \rightarrow f = g.$

Disadvantages of Cumulativity :

The subsomption rule `sub` makes it an *implicit* subtyping. Thus :

- No uniqueness of type for a term (even up to conversion and in a given context);
- Obstacle to proper extraction of programs from terms :
The extraction process eliminates any subterm $t : P : \text{Prop}$, as having purely *logical* content and therefore not participating in the computation (see [Let04]); even if it has *computational* content to integrate into the extracted program and only a subterm coming effectively from `Prop`, typically dominant due to its impredicativity.

The Link with Extraction

Example

```
Definition f (X: Type)(h: nat → X)(g: X → nat) := g (h 0).
```

```
Definition fprop := f True (fun _ => I) (fun _ => 1).
```

`fprop` is well-typed in Coq since `I: True`,

but `I : True : Prop`, so `(fun _ => I) : (nat → True) : Prop`, and the extraction replaces the entire subterm `(fun _ => I)` with an unimportant value.

To Remedy This :

Introduction of *coercions* $\text{coe}_{A,B}$ witnessing, at the level of their inhabitants, that $A \leq B$: if $t : A$ then $\text{coe}_{A,B}t : B$.

These explicit, ineradicable coercions then encode in the term itself the subtyping hidden in its typing derivation.

However, we will seek to keep only the relevant information, i.e. the possible passages from Prop to a Type_i ; or even some Type_i to some Type_j for $j \neq i$; their number and order, and their propagation through type formers with type arguments (and not necessarily, all the terms A and B themselves).

CC_{ω}^{sub}

CC_{ω}^{sub} : The Syntax

The option chosen to introduce this explicit subtyping is the key idea of *subtyping marks* m , in the following syntax grammar (metavariables will be freely, inductively primed) :

$$\begin{aligned}
 i & ::= 1 \mid i + 1 \\
 a, b, c & ::= x_i \\
 s, \sigma & ::= \text{Prop} \mid \text{Type}_i \\
 m & ::= \text{id} \mid \uparrow_s^{s'} \mid \uparrow_s^{s'} \mid \downarrow_s^{s'} \mid (m \rightarrow m') \mid m \circ m' \\
 t, u, A & ::= a \mid s \mid \Pi a : A : s.B \mid \lambda a : A : s.t \mid ut \mid \text{coe}_m t \\
 \Gamma & ::= \cdot \mid \Gamma, a : A : s
 \end{aligned}$$

preserving this information as an index of the symbol \leq of "the" subtyping relation, in order to then encode it in the terms via the introduction of a general symbol coe of coercion, indexed by the same mark.

We will need to use sorted domains and declarations, and sorting-typing judgements as well, in order to keep the information of the universe that the typing happens in, as the only atomic term that denotes it, which by definition is an s – for *sort*.

They are currently constructed from four symbols of constants :

- **id** expressing the part of the subtyping that remains implicit... That is only its reflexive part, built in between every type – especially universe – and itself,
- $\uparrow_s^{s'}$ expressing the subtyping of any universe s into any bigger one s' ,
- $\uparrow_s^{s'}$ allowing to propagate the subtyping between types inhabiting some universe s and of its $\uparrow_s^{s'}$ -coerced image into any bigger s' and
- $\downarrow_s^{s'}$ expressing the reciprocal subtyping between the $\uparrow_s^{s'}$ -coerced type of one in s into any bigger s' , and this naked one itself, in other words such that $\text{coe}_{\downarrow_s^{s'}}^{\downarrow_s^{s'}}$ is a retraction of $\text{coe}_{\uparrow_s^{s'}}^{\uparrow_s^{s'}}$;

and two subtyping mark formers :

And two subtyping mark formers :

- \rightarrow between Π -types (parenthesized because it is infix and non-associative), allowing to propagate not only the subtyping between their codomains (in a covariant way), but also that between their domains, in a contravariant way, and
- \circ expressing the built-in transitivity of our subtyping (unparenthesized even if infix because – trivially – equiderivably associative), necessary in particular to handle coerced universes.

CC_{ω}^{sub} : The Rules

$$\frac{A \equiv B}{A \leq_{\text{id}} B} \text{ sub}_{\text{id}}$$

$$\frac{}{s \leq_{\uparrow_s^{s'}} s'} \text{ if } s \sqsubset s' ; \text{ sub}_{\uparrow_s^{s'}}$$

$$\frac{A \equiv B}{A \leq_{\uparrow_s^{s'}} \text{coe}_{\uparrow_s^{s'}} B} \text{ sub}_{s_{s'}}$$

$$\frac{A \equiv B}{\text{coe}_{\uparrow_s^{s'}} A \leq_{\downarrow_s^{s'}} B} \text{ sub}_{\downarrow_s^{s'}}$$

$$\frac{A' \leq_m A \quad B[\text{coe}_m a'/a] \leq_{m'} B'}{\Pi a : A : s.B \leq_{m \rightarrow m'} \Pi a' : A' : s.B'}$$

for a' being a or fresh for B ; sub_Π

$$\frac{A \leq_m B \quad B \leq_{m'} C}{A \leq_{m' \circ m} C} \text{sub}_\circ$$

$$\frac{A' \leq_m B}{A \leq_m B} \text{ if } A \triangleright A' ; \text{sub}_{\triangleright, L}$$

$$\frac{A \leq_m B'}{A \leq_m B} \text{ if } B \triangleright B' ; \text{sub}_{\triangleright, R}$$

The unusual Conversion Rules

$$\frac{t \equiv u}{\text{coe}_m t \equiv \text{coe}_m u} \quad \text{for } m \neq \text{id} \mid \uparrow_s^s \mid \uparrow_s^s \mid \downarrow_s^s ; \text{coe} \equiv$$

$$\frac{\text{coe}_{m'}(t \text{ coe}_m a) \equiv u a}{\text{coe}_{m \rightarrow m'} t \equiv u} \quad \omega_L^\ddagger \qquad \frac{u a \equiv \text{coe}_{m'}(t \text{ coe}_m a)}{u \equiv \text{coe}_{m \rightarrow m'} t} \quad \omega_R^\ddagger$$

\ddagger for t and u not containing a (even λ -abstracted).

The Reduction Rules

- weak-head β -reduction, going under coercions too,
- $\text{coe}_m t \triangleright t$ for $m = \text{id} \mid \uparrow_s^s \mid \downarrow_s^s$;
- $\text{coe}_{m \circ m'} t \triangleright \text{coe}_m \text{coe}_{m'} t$;
- $\text{coe}_{\uparrow_{s'}^{s'}} \text{coe}_{\uparrow_s^{s'}} t \triangleright \text{coe}_{\uparrow_s^{s'}} t$, similarly for \uparrow 's, \downarrow 's and combinations of them;
- $(\text{coe}_{m \rightarrow m'} t u) \triangleright \text{coe}_{m'} (t (\text{coe}_m u))$;
- Rules such that whenever $s \sqsubseteq s'$ and $\sigma \sqsubset \sigma'$,

$$\begin{aligned} \text{coe}_{\uparrow_{\pi(s, \sigma)}^{\pi(s', \sigma')}} (\Pi a : A : s. B) &\equiv \Pi a : \text{coe}_{\uparrow_s^{s'}} A : s'. \text{coe}_{\uparrow_{\sigma'}^{\sigma}} B, \\ \text{coe}_{\uparrow_{\pi(s, \sigma)}^{\pi(s', \sigma')}} (\lambda a : A : s.t) &\equiv \lambda a : \text{coe}_{\uparrow_s^{s'}} A : s'. \text{coe}_{\uparrow_{\sigma'}^{\sigma}} t \left[\text{coe}_{\downarrow_{s'}^{s}} a/a \right] \\ &\equiv \text{coe}_{\downarrow_{s'}^{s} \rightarrow \uparrow_{\sigma'}^{\sigma}} (\lambda a : A : s.t) \\ \text{and} \\ \text{coe}_{\downarrow_{\pi(s, \sigma)}^{\pi(s', \sigma')}} (\lambda a : \text{coe}_{\uparrow_s^{s'}} A : s.t) &\equiv \lambda a : A : s'. \text{coe}_{\downarrow_{\sigma'}^{\sigma}} t \left[\text{coe}_{\uparrow_s^{s'}} a/a \right] \\ &\equiv \text{coe}_{\uparrow_{s'}^{s} \rightarrow \downarrow_{\sigma'}^{\sigma}} (\lambda a : A : s.t). \end{aligned}$$

It preserves subject reduction.

CC_{ω}^{sub} : The Key Typing Rule & What Happens to The Example

Explicit Coercive Version of the Subsumption Rule

The introduction of coercions is ruled as follows :

$$\frac{\Gamma \vdash t : A : s \quad A \leq_m B \quad \Gamma \vdash B : s' : N(s')}{\Gamma \vdash \text{coe}_m t : B : s'} \quad \text{sub}_{\text{coe}}$$

where N is the next-bigger-universe metafunction.

So that fprop is well-defined :

Definition $\text{fprop} := \text{f} (\text{coe}_{\text{Prop}}^{\text{Type1}} \text{True}) (\text{fun } _ \Rightarrow \text{coe}_{\text{Prop}}^{\text{Type1}} \text{I})$
 $(\text{fun } _ \Rightarrow 1)$.

And the extraction eliminates only the $\text{coe}_{\text{Prop}}^{\text{Type1}} \text{I}$, keeping notably the "fun".

CC_{ω}^{sub} : The Results

Usual or Commutation Lemmas

- Substitution by interconvertible terms at the same places does not affect the interconvertibility of types :

$$\frac{A \equiv B \quad v \equiv w}{A[v/b] \equiv B[w/b]} \text{ is admissible (for any } b).$$

- Substitutions are monotonic with respect to subtyping :

$$\frac{A \leq_m B \quad t \equiv u}{A[t/a] \leq_m B[u/a]} \text{ is admissible (for any } a).$$

- $$\frac{\Gamma, a : A : s : s, \Gamma' \text{ ok} \quad \Gamma, a : A : s, \Gamma' \vdash u : B : s' \quad \Gamma \vdash t : A : s}{\Gamma, \Gamma' [t/a] \vdash u [t/a] : B [t/a] : s'} \text{ subst}$$

is admissible.

- Any **sorted** type of a term in a well-formed context is itself indeed well-sorted there **by inhabiting this sort**, as the notation suggests : if $\Gamma \text{ ok}$ and $\Gamma \vdash t : A : s$ for some t , then $\Gamma \vdash A : s : N(s)$.
- For any well-typable A, B , and A', B' , if $A \leq_m B$ and $A' \leq_m B'$ for some m , then $A \equiv A'$ if and only if $B \equiv B'$.

Unicity of Typing

In $\text{CC}_\omega^{\text{sub}}$, thanks to the explicitation of the cumulativity of the full universe hierarchy, any term may only inhabit interconvertible types in a given well-formed context, and this in a same sort.

That is,

$$\frac{\Gamma \text{ ok} \quad \Gamma \vdash t : A : s \quad \Gamma \vdash t : A' : s'}{A \equiv A'} \text{ unicity}$$

is admissible by $\text{CC}_\omega^{\text{sub}}$, and in addition from those premises follow that we have $s = s'$.

This is also what makes propositions, not types and types, not propositions, for sure.

- Reduction lifting : if $|t| \triangleright u$ – i.e. $|t| \triangleright_{\beta} u$ – then there exists a decoration u' of u such that $t \triangleright u'$, actually even $t \triangleright_{\beta} u'$ too.
- Weak conversion lifting : any two terms that differ only by (potentially) several head coercions, and inhabit a same type in a same (sort and) well-formed context in CC_{ω}^{sub} , are mutually inter-convertible there.
- Subtyping lifting : if there is a $\Gamma \text{ ok}_{CC_{\omega}^{\text{sub}}}$ such that $\Gamma \vdash_{CC_{\omega}^{\text{sub}}} A : s : N(s)$ for some s and $\Gamma \vdash_{CC_{\omega}^{\text{sub}}} B : s' : N(s')$ for some s' , and if $|A| \leq |B|$ in CC_{ω} for some m , then there exists m such that $A \leq_m B$ in CC_{ω}^{sub} .
- Typing lifting : ? We need to decorate termes and types with well-placed coercions.

CC_{ω}^{sub} : The Algorithms

An Explicitation Algorithm

We define, by structural induction on terms t in $\mathbb{C}\mathbb{C}_\omega$, an inference function ϕ taking, a well-formed context Γ in $\mathbb{C}\mathbb{C}_\omega^{\text{sub}}$ and producing, if it exists, a decoration t' of t , an A and an s in $\mathbb{C}\mathbb{C}_\omega^{\text{sub}}$ for which we hope that $\Gamma \vdash_{\mathbb{C}\mathbb{C}_\omega^{\text{sub}}} t' : A : s$, in order to get a section of the erasure of coercions that preserves typing.

Its nontrivial cases :

$$\phi_\Gamma(\Pi a : A.B) \triangleq (\Pi a : A' : s.B', \pi(s, s'), N(\pi(s, s')))$$

where :

$(A', S, N(s)) \triangleq \phi_\Gamma(A)$	for the s such that $S \triangleright^* s$, or
$(A'', S, \sigma) \triangleq \phi_\Gamma(A)$	such that $S \triangleright^* \text{coe}_{\uparrow N(s)}^\sigma s$ and
$A' \triangleq \text{coe}_{\downarrow N(s)}^\sigma A''$, and in both cases
$(B', S', N(s')) \triangleq \phi_{\Gamma, a:A':s}(B)$	for the s' such that $S' \triangleright^* s'$, or
$(B'', S', \sigma') \triangleq \phi_{\Gamma, a:A':s}(B)$	such that
$S' \triangleright^* \text{coe}_{\uparrow N(s')}^{\sigma'}$	s'' and
$B' \triangleq \text{coe}_{\downarrow N(s')}^{\sigma'} B'';$	

An Explicitation Algorithm

$$\phi_{\Gamma}(\lambda a : A.t) \triangleq (\lambda a : A' : s.t', \Pi a : A' : s.B', \pi(s, s'))$$

where :

$(A', S, N(s)) \triangleq \phi_{\Gamma}(A)$	for the s such that $S \triangleright^* s$, or
$(A'', S, \sigma) \triangleq \phi_{\Gamma}(A)$	such that
$S \triangleright^* \text{coe}_{\uparrow_{N(s)}^{\sigma}} s$	and
$A' \triangleq \text{coe}_{\downarrow_{N(s)}^{\sigma}} A''$, and in both cases
$(t', B', s') \triangleq \phi_{\Gamma, a:A':s}(t);$	

$$\phi_{\Gamma}(u t) \triangleq (u' t', B[t'/a], s')$$

where :

$(u', C, \pi(s, s')) \triangleq \phi_{\Gamma}(u)$	for the s such that
$C \triangleright^* \Pi a : A : s.B$	for some a, A and B and the s'^1
such that $\Gamma, a : A : s \vdash_{\text{CC}_{\omega}^{\text{sub}}} B : s' : N(s')$, and
$t' \triangleq \psi_{\Gamma, A}(t).$	

1. Existing by induction (see the Theorem) and unique by Proposition ??.

A Mark Synthesis Algorithm

The square brackets below are options :

$$\text{mark}_{\text{whd}} \left(\left[\text{coe}_{\uparrow_{\text{Prop}}^s} \right] \Pi a : A : \sigma.A', \left[\text{coe}_{\uparrow_{\text{Prop}}^{s'}} \right] \Pi b : B : \sigma'.B' \right) \triangleq$$
$$\left[\uparrow_{\text{Prop}}^{s'} \circ \right] \left(\text{mark}(B, A) \rightarrow \text{mark}(A' \left[\text{coe}_{\text{mark}(B,A)} b/a \right], B') \right) \left[\circ \downarrow_{\text{Prop}}^s \right]$$

when b is a or fresh for A' , $s \sqsubseteq \sigma$ and $s' \sqsubseteq \sigma'$

$$\text{mark}_{\text{whd}}(A, B) \triangleq \text{id}$$

when A and B are convertible and not both Π -types, even coerced

A Mark Synthesis Algorithm

$$\begin{aligned}
 \text{mark}(A, B) &\triangleq \text{mark}_{\text{whd}}(\text{whd}(A), \text{whd}(B)) \\
 \text{mark}_{\text{whd}}(A, \text{coe}_{\uparrow_s^{s'}} B) &\triangleq \uparrow_s^{s'} && \text{when } A \equiv B \\
 \text{mark}_{\text{whd}}(\text{coe}_{\uparrow_s^{s'}} A, B) &\triangleq \downarrow_s^{s'} && \text{when } A \equiv B \\
 \text{mark}_{\text{whd}}(\text{coe}_{\uparrow_s^{s'}} A, \text{coe}_{\uparrow_s^{s''}} B) &\triangleq \uparrow_s^{s''} \circ \downarrow_s^{s'} && \text{when } A \equiv B \text{ and } s' \neq s'' \\
 \text{mark}_{\text{whd}}(s, s') &\triangleq \uparrow_s^{s'} && \text{when } s \sqsubset s' \\
 \text{mark}_{\text{whd}}(\text{coe}_{\uparrow_{\mathbf{N}(s)}^{s'}} A, s'') &\triangleq \uparrow_s^{s''} \circ \downarrow_{\mathbf{N}(s)}^{s'} && \text{when } A \triangleright^* s \sqsubset s'' \\
 \text{mark}_{\text{whd}}(s'', \text{coe}_{\uparrow_{\mathbf{N}(s)}^{s'}} B) &\triangleq \uparrow_{\mathbf{N}(s)}^{s'} \circ \uparrow_{s''}^{s''} && \text{when } s'' \sqsubset s^* \triangleleft B \\
 \text{mark}_{\text{whd}}(\text{coe}_{\uparrow_{\mathbf{N}(s)}^{s'}} A, \text{coe}_{\uparrow_{\mathbf{N}(\sigma)}^{\sigma'}} B) &\triangleq (\uparrow_{\mathbf{N}(\sigma)}^{\sigma'} \circ \uparrow_s^{\sigma}) \circ \downarrow_{\mathbf{N}(s)}^{s'} && \text{when } A \triangleright^* s \sqsubset \sigma^* \triangleleft B
 \end{aligned}$$

Theorem (Typing lifting)

Theorem (Th1.1)

If $\Gamma \text{ ok}_{\text{CC}_{\omega}^{\text{sub}}}$ and $|\Gamma| \vdash_{\text{CC}_{\omega}} A : s$ then $\psi_{\Gamma,s}$ is defined at A and $\Gamma \vdash_{\text{CC}_{\omega}^{\text{sub}}} \psi_{\Gamma,s}(A) : s : \mathbb{N}(s)$. For Γ and s given, $\psi_{\Gamma,s}(A)$ is the unique decoration of A up to coe-conversion such that the latter holds.

Theorem (Th1.2)

If $\Gamma \text{ ok}_{\text{CC}_{\omega}^{\text{sub}}}$, $\Gamma \vdash_{\text{CC}_{\omega}^{\text{sub}}} A : s : \mathbb{N}(s)$ for some s and $|\Gamma| \vdash_{\text{CC}_{\omega}} t : |A|$, then are defined at t both ϕ_{Γ} with value (t', B, s') such that $\Gamma \vdash_{\text{CC}_{\omega}^{\text{sub}}} t' : B : s'$, and even $\psi_{\Gamma,A}$ with $\Gamma \vdash_{\text{CC}_{\omega}^{\text{sub}}} \psi_{\Gamma,A}(t) : A : s$. For Γ and A given, $\psi_{\Gamma,A}(t)$ is the unique decoration of t up to coe-conversion such that the latter holds.

Theorem (Th1.3)

If $\Gamma \text{ ok}_{\text{CC}_{\omega}}$ then there exists a decoration Γ' of Γ such that $\Gamma' \text{ ok}_{\text{CC}_{\omega}^{\text{sub}}}$. It is unique given the sorts of its type declarations, up to coe-conversion of the types appearing.




Work to be Done

In the Medium Term

- Refine the definitions to precisely and completely determine the best possible CIC^{expl} system, the one that will have the best expected properties, notably through the treatment of inductive types, above (the best) CC_{ω}^{sub} ;
- Confront the implementation of CIC^{expl} in Coq, and the subsequent modifications of the extraction process.
- Explore more properties of \triangleright , the possible categorical semantics of CC_{ω}^{sub} and CIC^{expl} ...
- Exploration of the similarities of CC_{ω}^{sub} with the system presented in [GCST19], designed to hook up the universe Prop of propositions with inter-convertible proof terms of Coq to the cumulative hierarchy of universes : it is an inductive type "box" \square corresponding to $coe_{\uparrow Prop}^{Type_1}$, whose constructor *box* would correspond to $coe_{\uparrow Prop}^{Type_1}$ and the destructor to $coe_{\downarrow Prop}^{Type_1}$;

Long-Term Perspectives

- Another path, in which this work constitutes the first steps is the hope, by means of having explicated the cumulativity of the full CIC universe hierarchy, to arrive at a better understanding of its fine structure, and thus – via set theory – to obtain new results regarding the ordinal strength of this type theory and those that are related to it.
- Another one is to study the feasibility of an interpretation of Prop as a truncation *à la* HoTT which, if ever possible, seems to necessarily involve an analysis of the properties that an explicit encoding of impredicativity in terms of universe subtyping must verify, also aiming in the long term at a deeper understanding of impredicativity by reducing it to more elementary components. Further on, this issue could teach us a lot about the qualitative leap that distinguishes, within the cone of second-order arithmetic in terms of theories ordered by their expressive power, its strict subsystems - predicative – from its supersystems – impredicative.

-  G. Gilbert, J. Cockx, M. Sozeau & N. Tabareau – « Definitional proof-irrelevance without k », *Proceedings of the ACM on Programming Languages*, Vol. 3, Issue POPL (2019), p. Art. n°3.
-  P. Letouzey – « Programmation fonctionnelle certifiée », Thèse, Université de Paris XI Orsay, 2004.
-  The Coq Development Team – « The coq reference manual (version 8.12.0) », juil. 2020.