Cumulative Types Systems and Levels

François Thiré

Cumulative Types Systems and Levels

François Thiré
INRIA Saclay
LSV, CNRS
ENS Paris-Saclay.
Cachan, France
francois.thire@inria.fr

Cumulative Typed Systems (CTS), extend Pure Type Systems with a subtyping relation on universes. We introduce *LCTS*, a CTS enriched with a notion of level. *LCTS* has subject reduction (reduction preserves types) but lacks a strong reduction property that levels are also preserved. We show that this strong subject reduction property implies two famous conjectures on CTS: Expansion postponement and the equivalence between explicit and implicit conversion. The former is an open conjecture in the general case for PTS/CTS. The latter has been proved by Siles [5] for PTS only and is still a conjecture for CTS. We rephrase this notion of level using a well-founded order on derivation trees. We show that the existence of such well-founded order implies a type system with the strong subject reduction property. Hence, these two conjectures is a direct consequence of the existence of such well-founded order. Yet, it is not known if such well-founded order exists in general.

1 Introduction

Cumulative Type Systems (CTS) are an extension of Pure Type Systems (PTS) [2] that were introduced by Bruno Barras [3]. This family of type systems extends PTS with an implicit subtyping relation on universes. Many systems that we find today can be seen as an extension of CTS: Agda, Coq, Lean or even systems of the HOL-family. Hence, a lot of properties on real systems can be derived from CTS.

While working on an explicit version of subtyping in CTS, extending Ali Assaf's work [1], we stumbled across a well-foundedness issue. Investigating this issue led to two famous conjectures already formulated on CTS. (1) Expansion postponement (EP) states that expansion β steps (right to left steps) can always be postponed at the end of type checking. (2) The equivalence between explicit and implicit conversion (EIE) states that the premise $t \equiv_\beta t'$ in the conversion rule can be transformed into a complete judgment with a derivation tree and is not purely a computation. While the latter implies the correction of our encoding, we discovered that assigning levels to derivation trees would be a generic solution to solve all these conjectures.

In Section 2, we introduce Layered-CTS (LCTS), an equivalent version of CTS where an ordered set is introduced to refine the typing relation with levels. Roughly speaking, levels measure the complexity of a derivation tree. While this system has the subject reduction property, we are interested in a stronger notion of subject reduction where levels cannot increase through reductions

Submitted to:
SOS 2007

$$\begin{array}{lccll} \text{Sorts} & s & \in \mathcal{S} & & \\ \text{Terms} & M,N,A,B & \in \mathcal{T} & ::= & x \mid s \mid M\,N \mid \lambda x{:}A.M \mid (x{:}A) \to B \\ \text{Contexts} & \Gamma & \in \mathcal{G} & ::= & \varnothing \mid \Gamma, x:A \end{array}$$

Figure 1: CTS syntax

$$\frac{A \equiv_\beta B}{A \preceq_{\mathscr{C}} B}\ \preceq_{\equiv_\beta} \qquad \frac{(s,s') \in \mathcal{C}_{\mathscr{C}}}{s \preceq_{\mathscr{C}} s'}\ \preceq_{\mathcal{C}} \qquad \frac{B \preceq_{\mathscr{C}} B'}{(x{:}A) \to B \preceq_{\mathscr{C}} (x{:}A) \to B'}\ \preceq_{\Pi}$$

$$\frac{A \preceq_{\mathscr{C}} B \qquad B \preceq_{\mathscr{C}} C}{A \preceq_{\mathscr{C}} C}\ \preceq_{tr}$$

Figure 2: LCTS subtyping relation

(SSR) . We show in Section 3 that this property implies EP and EIE. In Section 4 we give sufficient conditions to ensure SSR. We show that finding appropriate levels for LCTS to satisfy SSR is the same as finding a valid order $\succ$ on derivation trees which remains today an open conjecture in the general case.

2 Layered-CTS

LCTS are parameterized by a specification that we will denote $\mathscr{C}$. A LCTS *specification* is a 5-tuple $\mathscr{C} = (\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{C}, \mathcal{O}$ where: 1. $\mathcal{S}$ is a set of constants called *sorts*, 2. $\mathcal{A} \subseteq \mathcal{S} \times \mathcal{S}$ is a relation called *axioms*, 3. $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S} \times \mathcal{S}$ is a relation called *rules*, 4. $\mathcal{C} \subseteq \mathcal{S} \times \mathcal{S}$ is a relation called *cumulativity*, 5. $\mathcal{O}$ is a well-founded total ordered set. The syntax of LCTS is introduced in Fig. 1. It is shared for all specifications except for sorts. The typing system is introduced in Fig. 2 and Fig 3. They use the same rules as the usual CTS but judgments are indexed by a level $o \in \mathcal{O}$. When the level o is omitted, it means we are using the usual system and $\mathcal{O}$ is removed from the specification which is denoted by $\mathscr{C}^-$. We use $\mathscr{D}$ to refer to the type of derivation trees for LCTS. The LCTS system has been crafted so that if $\Gamma \vdash^{o_1}_{\mathscr{C}} t : A$, and A is not a sort, then we can derive $\Gamma \vdash^{o_2}_{\mathscr{C}} A : s$ with $o_2 <_{\mathcal{O}} o_1$. The example below will be instantiated for the case of the Calculus Of Constructions with $\star : \Box$ and the four usual products on them [2]. We will instantiate $\mathcal{O}_{\mathscr{C}}$ with $\mathbb{N}$ in our example. At the base layer $n = 0$, only types without variables can be derived: $\vdash^0_{\mathscr{C}} \star : \Box :$ or $\vdash^0_{\mathscr{C}} \star \to \star : \Box$. At level 1, types with free variables can be introduced such as $\vdash^1_{\mathscr{C}} (A:\star) \to A \to A : \star$. At level 2, one can derive the polymorphic identity function for example: $\vdash^2_{\mathscr{C}} \lambda A:\star.\lambda x{:}A.x : (A:\star) \to A \to A$. At level 3, one can derive types using terms derivable at level

$$\frac{o \in \mathcal{O}_{\mathscr{C}}}{\varnothing \vdash^{o}_{\mathscr{C}} \mathbf{wf}}\ \mathscr{C}^{\mathbf{wf}}_{\varnothing}$$

$$\frac{\Gamma \vdash^{o}_{\mathscr{C}} A : s \qquad x \notin \Gamma}{\Gamma, x : A \vdash^{o}_{\mathscr{C}} \mathbf{wf}}\ \mathscr{C}^{\mathbf{wf}}_{var}$$

$$\frac{\Gamma \vdash^{o_1}_{\mathscr{C}} \mathbf{wf} \qquad (x : A) \in \Gamma \qquad o_1 <_{\mathcal{O}} o_2}{\Gamma \vdash^{o_2}_{\mathscr{C}} x : A}\ \mathscr{C}_{var}$$

$$\frac{\Gamma \vdash^{o}_{\mathscr{C}} \mathbf{wf} \qquad (s_1, s_2) \in \mathcal{A}}{\Gamma \vdash^{o}_{\mathscr{C}} s_1 : s_2}\ \mathscr{C}_{sort}$$

$$\frac{\Gamma \vdash^{o_1}_{\mathscr{C}} A : s_1 \qquad \Gamma, x : A \vdash^{o_2}_{\mathscr{C}} B : s_2 \qquad (s_1, s_2, s_3) \in \mathcal{R} \qquad o_1 \leq_{\mathcal{O}} o_3 \wedge o_2 \leq_{\mathcal{O}} o_3}{\Gamma \vdash^{o_3}_{\mathscr{C}} (x{:}A) \to B : s_3}\ \mathscr{C}_{\Pi}$$

$$\frac{\Gamma, x : A \vdash^{o_1}_{\mathscr{C}} M : B \qquad \Gamma \vdash^{o_2}_{\mathscr{C}} (x{:}A) \to B : s \qquad o_1 \leq_{\mathcal{O}} o_3 \wedge o_2 <_{\mathcal{O}} o_3}{\Gamma \vdash^{o_3}_{\mathscr{C}} \lambda x{:}A.M : (x{:}A) \to B}\ \mathscr{C}_{\lambda}$$

$$\frac{\Gamma \vdash^{o_1}_{\mathscr{C}} M : (x{:}A) \to B \qquad \Gamma \vdash^{o_2}_{\mathscr{C}} N : A \qquad o_1 \leq_{\mathcal{O}} o_3 \wedge o_2 \leq_{\mathcal{O}} o_3}{\Gamma \vdash^{o_3}_{\mathscr{C}} M\, N : B\{x \leftarrow N\}}\ \mathscr{C}_{app}$$

$$\frac{\Gamma \vdash^{o_1}_{\mathscr{C}} M : A \qquad \Gamma \vdash^{o_2}_{\mathscr{C}} B : s \qquad A \preceq_{\mathscr{C}} B \qquad o_1 \leq_{\mathcal{O}} o_3 \wedge o_2 <_{\mathcal{O}} o_3}{\Gamma \vdash^{o_3}_{\mathscr{C}} M : B}\ \mathscr{C}_{\preceq}$$

$$\frac{\Gamma \vdash^{o_1}_{\mathscr{C}} M : A \qquad A \preceq_{\mathscr{C}} s \qquad o_1 \leq_{\mathcal{O}} o_2}{\Gamma \vdash^{o_2}_{\mathscr{C}} M : s}\ \mathscr{C}^{s}_{\preceq}$$

Figure 3: Typing rules for LCTS

2 but also one can derive terms with types derivable at level 2, and so on... The main purpose of levels is to handle spurious type derivations as shown below.

Example 1 *While there is a trivial derivation of* $\vdash^0_{\mathscr{C}} \star : \Box$*, there is another derivation of this fact:*

$$\dfrac{\dfrac{\vdash^2_{\mathscr{C}} \star : \Box \qquad \vdash^1_{\mathscr{C}} (\lambda x{:}\Box.\Box)\star : \Box \qquad \Box \equiv_\beta (\lambda x{:}\Box.\Box)\star}{\vdash^2_{\mathscr{C}} \star : (\lambda x{:}\Box.\Box)\star} \qquad (\lambda x{:}\Box.\Box)\star \equiv_\beta \Box}{\vdash^2_{\mathscr{C}} \star : \Box}$$

One might check that $\vdash^1_{\mathscr{C}} (\lambda x{:}\Box.\Box)\star : \Box$ *is indeed derivable if* $\Box$ *has a sort. Levels make a distinction between the two derivations, and at level* 0*, we can guarantee that there is no such spurious derivation.*

Before going on to the equivalence theorem, we state the following lemma:

Lemma 2.1 (LCTS-sub-ht) *If* $\Gamma \vdash^{o_1}_{\mathscr{C}} t : A$*, then for all* o_2 *such that* $o_1 \leq_{\mathcal{O}} o_2$ *we have* $\Gamma \vdash^{o_2}_{\mathscr{C}} t : A$*.*

Proof *By induction on the derivation of* $\Gamma \vdash^{o_1}_{\mathscr{C}} t : A$*.*

Theorem 2.2 (CTS and LCTS are equivalent) $\forall \mathscr{C}^-, \Gamma, t, A, (\Gamma \vdash_{\mathscr{C}^-} t : A \Leftrightarrow \exists n, \Gamma \vdash^n_{\mathscr{C}} t : A)$ *where* $\mathscr{C} = (\mathscr{C}^-, \mathbb{N})$*.*

Proof *The right-to-left implication is trivial since levels are just a restriction. For the left-to-right implication. The proof is by induction on the derivation. We give here the proof for the* $\mathscr{C}_\lambda$ *case: By inversion we have* $\Gamma, x : A \vdash_{\mathscr{C}^-} M : B$ *and* $\Gamma \vdash_{\mathscr{C}^-} (x{:}A) \to B : s$*. By induction hypothesis, there exists* m_1 *and* m_2 *such that* $\Gamma, x : A \vdash^{m_1}_{\mathscr{C}} M : B$ *and* $\Gamma \vdash^{m_2}_{\mathscr{C}} (x{:}A) \to B : s$*. Using Lemma 2.1 (*LCTS*-sub-ht) we have* $\Gamma, x : A \vdash^{\max(m_1,m_2)+1}_{\mathscr{C}} M : B$ *and* $\Gamma \vdash^{\max(m_1,m_2)}_{\mathscr{C}} (x{:}A) \to B : s$*. Hence, one can conclude with* $n = \max(m_1, m_2)$ *using the rule* $\mathscr{C}_\lambda$*. All the other cases are similar.*

The proof above works also for any total and well-founded ordered which includes natural numbers:

Corollary 2.3 *The existence of a monotonic function from* $\mathbb{N}$ *to* $\mathcal{O}_{\mathscr{C}}$ *implies* $\forall \mathscr{C}^-, \Gamma, t, A, (\Gamma \vdash_{\mathscr{C}^-} t : A \Leftrightarrow \exists o, \Gamma \vdash^o_{\mathscr{C}} t : A)$*.*

This equivalence property gives us all the good properties that we know on CTS, including subject reduction stated as follows:

Lemma 2.4 (LCTS-subject-reduction) *If there exists a monotonic function from* $\mathbb{N}$ *to* $\mathcal{O}_{\mathscr{C}}$*,* $\Gamma \vdash^{o_1}_{\mathscr{C}} t : A$ *and* $t \hookrightarrow_\beta t'$ *then there exists* o_2 *such that* $\Gamma \vdash^{o_2}_{\mathscr{C}} t' : A$*.*

However, this theorem does not connect o_1 with o_2. Instead, we are interested in a stronger property called *strong subject reduction*.

Definition 2.1 (Stability) *A judgment* $\Gamma \vdash^o_{\mathscr{C}} t : A$ *is* stable *if for all* t' *such that* $t \hookrightarrow^*_\beta t'$ *we have* $\Gamma \vdash^o_{\mathscr{C}} t' : A$*.*

Definition 2.2 (Strong Subject Reduction) *Given a derivation* Π *of the judgment* $\Gamma \vdash^{o}_{\mathscr{C}} t : A$*, we define* $SSR(\Pi)$ *as:* $\Gamma \vdash^{o}_{\mathscr{C}} t : A$ *is stable and all its subtrees are also stable. We define SSR as: For all derivations* Π *of the judgment* $\Gamma \vdash^{o}_{\mathscr{C}} t : A$*, there exists* Π' *a derivation of* $\Gamma \vdash^{o'}_{\mathscr{C}} t : A$ *such that* $SSR(\Pi')$.

It is not known if SSR is true for LCTS. One can check that levels assigned by Theorem 2.2 do not work because in the context $\Gamma = Nat : \star, Vec : Nat \to \star, l : (x{:}Nat) \to Vec\ x$ and assuming we have a proof that $\Gamma \vdash^{3}_{\mathscr{C}} 3 : Nat$, one can check that $\Gamma \vdash^{3}_{\mathscr{C}} (\lambda x{:}Nat.\, l\ x)\ 3 : (x{:}Nat) \to Vec\ x$ but it reduces to $\Gamma \vdash^{4}_{\mathscr{C}} l\ 3 : Vec\ 3$, and hence the derivation is not stable at level 3.

3 Strong subject reduction

In this section, we detail how the strong subject reduction property (SSR) can be used to derive the two following conjectures: Expansion postponement (EP) and the equivalence between explicit and implicit conversion (EIE).

3.1 Expansion postponement

EP is a strong property which states that it suffices to orient $\equiv_\beta$ during type checking, using first only $\hookrightarrow_\beta$ and then only $\hookleftarrow_\beta$.

Definition 3.1 (Expansion postponement) *Given a derivation* Π *of the judgment* $\Gamma \vdash^{o}_{\mathscr{C}} t : A$*, we define* $EP(\Pi)$ *as: there exists* A' *with* $A \hookrightarrow^{*}_{\beta} A'$ *such that* $\Gamma \vdash^{o}_{\mathscr{C}_r} t : A'$ *where we use the notation* $\Gamma \vdash^{o}_{\mathscr{C}_r} t : A$ *to denote the typing system where the rules* $\mathscr{C}_\preceq$ *and* $\mathscr{C}^{s}_\preceq$ *are restricted to* $\hookrightarrow^{*}_{\beta}$ *instead of* $\equiv_\beta$*. We define EP as: For all derivations* Π *of the judgment* $\Gamma \vdash^{o}_{\mathscr{C}} t : A$*, there exists* Π' *a derivation of* $\Gamma \vdash^{o'}_{\mathscr{C}} t : A$ *such that* $EP(\Pi')$.

Lemma 3.1 *For all* Π *then* $SSR(\Pi)$ *implies* $EP(\Pi)$.

Proof *The proof of this theorem can be done in* LCTS *using SSR as hypothesis and is done by induction on the level. The case where o is a minimal element is trivial since there is no variable in play. Assuming expansion postponement at level* o'*,* $o' <_{\mathcal{O}} o$*, one may prove expansion postponement at level o. This is done by induction on the derivation. We detail the proof for* $\mathscr{C}_\lambda$*. By induction hypothesis, we have* $\Gamma, x : A \vdash^{o_1}_{\mathscr{C}_r} M : B'$ *with* $o_1 \leq_{\mathcal{O}} o$ *and* $B \hookrightarrow^{*}_{\beta} B'$*. By induction hypothesis, we also have* $\Gamma \vdash^{o_2}_{\mathscr{C}_r} (x{:}A) \to B : s$ *with* $o_2 <_{\mathcal{O}} o$*. By equivalence (at level* o_2*), we have* $\Gamma \vdash^{o_2}_{\mathscr{C}} (x{:}A) \to B : s$*. By SSR we have* $\Gamma \vdash^{o_2}_{\mathscr{C}} (x{:}A) \to B' : s$*. Using expansion postponement at level* o_2 *we get* $\Gamma \vdash^{o_2}_{\mathscr{C}_r} (x{:}A) \to B' : s$ *which allows us to conclude that* $\Gamma \vdash^{o}_{\mathscr{C}} \lambda x{:}A.t : (x{:}A) \to B'$ *with* $\mathscr{C}_\lambda$ *since* $(x{:}A) \to B \hookrightarrow^{*}_{\beta} (x{:}A) \to B'$*. The other cases are trivial.*

Theorem 3.2 *SSR implies EP*

Proof *By definition of SSR, applying Lemma 3.1.*

3.2 Explicit conversion

In the classical setting, β conversion is purely computational. But one can wonder if this system is equivalent to a typed version which uses a judgment $\Gamma \vdash^o_{\mathscr{C}_e} A \equiv_\beta B : s$ meaning that A and B are convertible, both of type s. Rules defining $\Gamma \vdash^o_{\mathscr{C}_e} A \equiv_\beta B : s$ and $\Gamma \vdash^o_{\mathscr{C}_e} t : A$ are not written here. We refer to Siles PhD Thesis [5] for a detailed definition for PTS. His rules extend easily for CTS.

Definition 3.2 (Equivalence between explicit and implicit version) *Given a derivation Π of the judgment $\Gamma \vdash^o_{\mathscr{C}} t : A$, we define $EIE(\Pi)$ as: $\Gamma \vdash^o_{\mathscr{C}} t : A$ if and only if $\Gamma \vdash^o_{\mathscr{C}_e} t : A$. The equivalence between the explicit and implicit version (EIE) is defined as: For all derivations Π of the judgment $\Gamma \vdash^o_{\mathscr{C}} t : A$, there exists Π' a derivation of $\Gamma \vdash^{o'}_{\mathscr{C}} t : A$ such that $EIE(\Pi')$.*

At first, it seems trivial via subject reduction that this system is equivalent to the classical setting. But the devil is in the details... As mentioned by Siles in [5], the difficult case to prove is the conversion rule $\mathscr{C}_\preceq$ restricted here with only β conversion for simplicity:

$$\frac{\Gamma \vdash^{o_1}_{\mathscr{C}} M : A \qquad \Gamma \vdash^{o_2}_{\mathscr{C}} B : s \qquad A \equiv_\beta B \qquad o_1 \leq_{\mathcal{O}} o_3 \wedge_2 <_{\mathcal{O}} o_3}{\Gamma \vdash^{o_3}_{\mathscr{C}} M : B}\ \mathscr{C}_\preceq$$

To conclude the equivalence, we need subject reduction for the explicit version. However, proving subject reduction in this version is difficult, especially because the injectivity of products is not free anymore since now β is typed. To overcome this issue, we can process as we did for EP and prove the equivalence by induction on the level. In the following, let $\mathcal{L}(\Pi)$ denotes the level of the derivation Π and I_o the proposition: $\forall \Pi', \mathcal{L}(\Pi') < o \Rightarrow SSR(\Pi') \Rightarrow EIE(\Pi')$.

Lemma 3.3 *If Π is a derivation of $\Gamma \vdash^o_{\mathscr{C}} t : A$ such that $SSR(\Pi)$ and $I_{\mathcal{L}(\Pi)}$, then for all t' such that $t \hookrightarrow_\beta t'$ we have $\Gamma \vdash^o_{\mathscr{C}_e} t \equiv_\beta t' : A$.*

Proof *By induction on $t \hookrightarrow_\beta t'$. The proof is almost the same as a proof of subject reduction for* CTS *or* LCTS *except for injectivity of products. Proving subject reduction at level o needs injectivity of products for types at level $o' <_{\mathcal{O}} o$ for the explicit system. We can use our equivalence hypothesis to get back to the implicit system. Then we use SSR to derive injectivity of products at level o'. We reuse our equivalence to derive injectivity of product in the explicit system.*

Lemma 3.4 *For all Π then $SSR(\Pi)$ and $I_{\mathcal{L}(\Pi)}$ implies $EIE(\Pi)$.*

Proof *The right to left implication is easy to prove since we simply remove all the convertibility trees. The left to right implication is proved by induction on the derivation Π. The interesting case is $\mathscr{C}_\preceq$. We would like to derive the judgment $\Gamma \vdash^{\max(o_1,o_2)}_{\mathscr{C}_e} A \equiv_\beta B : s$ from $A \equiv_\beta B$ where $\Gamma \vdash^{o_1}_{\mathscr{C}} A : s$ and $\Gamma \vdash^{o_2}_{\mathscr{C}} B : s$. We use Lemma 3.3 and conclude thanks to the transitivity of $\equiv$. The case $\mathscr{C}^s_\preceq$ is similar and all the other cases are proved using the induction hypothesis.*

Lemma 3.5 *For all Π then $SSR(\Pi)$ implies $EIE(\Pi)$.*

Proof *This proof is done by induction on the $\mathcal{L}(\Pi)$. The base case is trivial, the induction case is handled by Lemma 3.4.*

Theorem 3.6 *SSR implies EIE*

Proof *By definition of SSR, applying Lemma 3.5.*

4 Sufficient conditions to ensure Strong subject reduction

As we saw in Section 2, we were unable to prove the strong subject reduction property (SSR). The purpose of this section is to settle the problem differently by introducing an order $\succ$ on derivation trees. As shown in example 1, this order cannot work directly on judgments, and instead has to be formulated on trees. This implies to have a constructive proof of subject reduction which maps derivation trees to derivation trees. The same applies for the well-sorted property which states that every type is well-sorted.

Definition 4.1 *Let be* $\mathbf{SR} : \mathscr{D} \to \mathcal{T} \to \mathscr{D}$ *a function such that:*

$$\mathbf{SR}\left(\frac{\begin{array}{c}\Pi\\ \hline \Gamma \vdash_{\mathscr{C}^-} t : A\end{array}}{},t'\right) = \frac{\begin{array}{c}\Pi'\\ \hline \Gamma \vdash_{\mathscr{C}^-} t' : A\end{array}}{} \text{ if } t \hookrightarrow_\beta t'$$

We denote by $\Pi_{\mathbf{SR}}$ *the tree computed by* **SR**.

Definition 4.2 *Let be* $\mathbf{WS} : \mathscr{D} \to \mathscr{D}$ *a partial function such that*

$$\mathbf{WS}\left(\frac{\begin{array}{c}\Pi\\ \hline \Gamma \vdash_{\mathscr{C}^-} t : A\end{array}}{}\right) = \frac{\begin{array}{c}\Pi'\\ \hline \Gamma \vdash_{\mathscr{C}^-} A : s\end{array}}{}$$

We denote by $\Pi_{\mathbf{WS}}$ *the tree computed by* **WS**. *Notice that this function is total if all sorts have a type.*

Notation 1 $\Pi \triangleright \Pi'$ *denotes the fact that* Π' *is a subtree of* Π.

With these definitions, we can now give a proper definition for validity conditions of $\succ$.

Definition 4.3 *A well-founded order* $\succ$ *on derivation trees and two functions* **SR** *and* **WS** *are* valid *if the following properties are satisfied:*

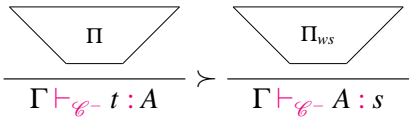
1. *For all* Π *and* Π' *such that* $\Pi \triangleright \Pi'$ *then*

$$\frac{\begin{array}{c}\Pi\\ \hline \Gamma \vdash_{\mathscr{C}^-} t : A\end{array}}{} \succeq \frac{\begin{array}{c}\Pi'\\ \hline \Gamma' \vdash_{\mathscr{C}^-} u : B\end{array}}{}$$

2. *For all* Π *and if* $t \hookrightarrow_\beta t'$ *then*

$$\frac{\begin{array}{c}\Pi\\ \hline \Gamma \vdash_{\mathscr{C}^-} t : A\end{array}}{} \succeq \frac{\begin{array}{c}\Pi_{\mathbf{SR}}\\ \hline \Gamma \vdash_{\mathscr{C}^-} t' : A\end{array}}{}$$

3. *For all* Π *then either* $A = s$ *or*



In practice, the usual subject reduction lemma, and the well-sorted lemma give us two functions that can be used for $\succ$, but of course, other functions that could fit in this picture.

Theorem 4.1 *Given a specification* $\mathscr{C}^-$ *and a valid order* $\succ$*, there exists a well-founded total ordered set* $\mathcal{O}$ *such that* $\mathscr{C} = (\mathscr{C}^-, \mathcal{O})$ *which satisfies SSR.*

Proof *The idea is to use* $\succ$ *to give concretes levels to* LCTS. *The proof is by induction on the tree. We sketch the* $\mathscr{C}_\lambda$ *case here. By induction hypothesis, there exists* o_1, o_2 *such that* $\Gamma, x : A \vdash^{o_1}_{\mathscr{C}} t : B$ *and* $\Gamma \vdash^{o_2}_{\mathscr{C}} (x{:}A) \to B : s$*. The level we use for* $\lambda x{:}A.t$ *can be* o_1 *using the first condition. And hence, we have* $\Gamma \vdash^{o_1}_{\mathscr{C}} \lambda x{:}A.t : (x{:}A) \to B$*. By validity of* $\succ$ *using the third condition, we have* $o_1 > o_2$*. SSR is a direct consequence of the second property.*

This proof tells us that finding appropriate level for LCTS which ensure SSR can be reduced to find a valid order $\succ$ which we recall at that time is still an open conjecture for the general case.

However, we can wonder if such valid order exists at all. So far, we were able to apply these results for some specifications without dependent types. This is the case for example for the system $\mathscr{F}_\omega$ with cumulativity ($(\star, \Box) \in \mathcal{C}_{\mathscr{F}_\omega}$) for which we can prove SSR. The reason why it works for $\mathscr{F}_\omega$ is because we can assign a level to a type which cannot grow through reductions since types are not dependent. Also, we have checked that in the Girard's system $\mathscr{U}^-$, the non-terminating provided by Hurkens paradox does not terminate, its level (with $\mathcal{O}_{\mathscr{U}^-} = \mathbb{N}$) is stable at level 3.

5 Conclusion

In this paper, we have introduced a notion of level which aims to classify terms. We have first introduced a new type system LCTS which is equivalent to CTS. We defined the strong subject reduction property SSR and proved how this property implies EP and EIE for LCTS and hence for CTS too. Then, we have introduced sufficient conditions an order $\succ$ should satisfy to ensure SSR.

It is not clear at all if such valid order could exist for all CTS specifications. But we have found some CTS specifications such as $\mathscr{F}_\omega$ with cumulativity for which such order exists. Further investigations on loop combinators or fixed-point combinators [4] could be done on LCTS to see whether the level of these terms increases infinitely often. Finally, one can also wonder about the converse implications which today are still conjectures:

- Does EP implies SSR for LCTS?
- Does EIE implies SSR for LCTS?

References

[1] Ali Assaf (2015): *A framework for defining computational higher-order logics. (Un cadre de définition de logiques calculatoires d'ordre supérieur)*. Ph.D. thesis, École Polytechnique, Palaiseau, France. Available at https://tel.archives-ouvertes.fr/tel-01235303.

[2] H. P. Barendregt (1992): *Handbook of Logic in Computer Science (Vol. 2)*. Oxford University Press, Inc., New York, NY, USA. Available at http://dl.acm.org/citation.cfm?id=162552.162561.

[3] B. Barras (1999): *Auto-validation d'un système de preuves avec familles inductives*. Thèse de doctorat, Université Paris 7.

[4] Thierry Coquand & Hugo Herbelin (1994): *A - Translation and Looping Combinators in Pure Type Systems*. *J. Funct. Program.* 4(1), pp. 77–88, doi:10.1017/S0956796800000952. Available at https://doi.org/10.1017/S0956796800000952.

[5] Vincent Siles (2010): *Investigation on the typing of equality in type systems. (Etude sur le typage de l'égalité dans les systèmes de types)*. Ph.D. thesis, École Polytechnique, Palaiseau, France. Available at https://tel.archives-ouvertes.fr/pastel-00556578.