REDUCTION FREE NORMALIZATION FOR A PROOF-IRRELEVANT TYPE OF PROPOSITIONS

Logical Methods in Computer Science
Volume 19, Issue 3, 2023, pp. 5:1–5:11
https://lmcs.episciences.org/

Submitted Dec. 09, 2021
Published Jul. 13, 2023

REDUCTION FREE NORMALIZATION FOR A PROOF-IRRELEVANT TYPE OF PROPOSITIONS

THIERRY COQUAND

Computer Science Department, University of Gothenburg
e-mail address: Thierry.Coquand@cse.gu.se

Abstract. We show normalization for a type theory with a hierarchy of universes and a proof irrelevant type of propositions, close to the type system used in the proof assistant Lean. The proof uses the technique of Artin glueing between the term model and a suitable preseaf model. This can also be seen as a proof relevant version of Tait's computability argument.

Introduction

We show normalization and decidability of conversion for dependent type theory with a cumulative sequence of universes $\mathsf{U}_0, \mathsf{U}_1 \dots$ with η-conversion *and* where the type U_0 is an impredicative universe of proof-irrelevant propositions. One interest of such a system is that it is very close to the type system used by the proof assistant Lean [Car19].

Such a system with a hierarchy of universes, with the lowest level impredicative, was introduced in [Coq86]. It was conjectured there that this system is stronger than Zermelo set theory (without even introducing primitive data types). This conjecture was solved by A. Miquel in [Miq04], by encoding a non well-founded version of set theory where a set is interpreted as a pointed graph up to bissimulation. The notion of proof-irrelevant propositions goes back to de Bruijn [dB95].

Our proof is a direct adaptation of the normalization argument presented in [Coq19]. We recall three features of this approach

(1) we never need to consider a *reduction* relation,
(2) we only define a reducibility *predicate*, and this reducibility predicate is *proof-relevant*[1],
(3) the reducibility predicate is not defined by an inductive-recursive relation.

This approach has been much refined in [Ste22, Gra22]. One goal of this note is to illustrate further the flexibility of this "reduction free" approach, by combining it with an idea already used in [ACP09] for dealing with proof irrelevance. To each type A in a context Γ, we associate a set of syntactical expressions $\mathsf{Term}(\Gamma, A)$ and a set $\mathsf{Elem}(\Gamma, A)$ of expressions *modulo conversion*. We have a quotient map $\mathsf{Term}(\Gamma, A) \to \mathsf{Elem}(\Gamma, A)$ and the main result (Theorem 4.1) is to show that this map has a section.

Key words and phrases: Dependent Type Theory; Presheaf models; Normalization.

[1]A key point is to define reducibility as a *structure* and not only as a *property*. It is only for the lowest impredicative universe U_0 that reducibility is a property.

LOGICAL METHODS IN COMPUTER SCIENCE
DOI:10.46298/LMCS-19(3:5)2023

The metatheory used in the present note is the impredicative intuitionistic set theory IZFu_ω, introduced by P. Aczel [Acz98]. (Essentially the same argument works in a predicative version CZFu_ω for a predicative universe of proof-irrelevant propositions.)

As in the previous work [Coq19], the approach is *algebraic*. We first define a general operation which associates to any model M another *normalization model* M^* with a projection map $M^* \to M$. We apply then this general construction to the initial model to deduce various syntactical properties, such as normalization, decidability of conversion and type-checking.

1. What is a model of type theory

1.1. **Definition.** We present a formal system, which at the same time can be thought of describing the syntax of basic dependent type theory, with *explicit substitutions* and a *name-free* (de Bruijn index) presentation, and defining what is a model of type theory.

A model of type theory consists of one set Con of *contexts*. If Γ and Δ are in Con they determine a set $\Delta \to \Gamma$ of *substitutions*. If Γ is in Con, it determines a set $\mathsf{Type}(\Gamma)$ of *types* in the context Γ. Finally, if Γ is in Con and A is in $\mathsf{Type}(\Gamma)$ then this determines a set $\mathsf{Elem}(\Gamma, A)$ of elements of type A in the context Γ.

This describes the *sort* of type theory. We describe now the *operations* and the equations they have to satisfy. For any context Γ we have an identity substitution $\mathsf{id} : \Gamma \to \Gamma$. We also have a composition operator $\sigma\delta : \Theta \to \Gamma$ if $\delta : \Theta \to \Delta$ and $\sigma : \Delta \to \Gamma$. The equations are

$$\sigma\ \mathsf{id} = \mathsf{id}\ \sigma = \sigma\ (\theta\sigma)\delta = \theta(\sigma\delta)$$

We have a terminal context 1 and for, any context Γ, a map $() : \Gamma \to 1$. Furthermore, $\sigma = ()$ if $\sigma : \Gamma \to 1$.

If A in $\mathsf{Type}(\Gamma)$ and $\sigma : \Delta \to \Gamma$ we should have $A\sigma$ in $\mathsf{Type}(\Delta)$. Furthermore, we have

$$A\ \mathsf{id} = A\ (A\sigma)\delta = A(\sigma\delta)$$

If a in $\mathsf{Elem}(\Gamma, A)$ and $\sigma : \Delta \to \Gamma$ we should have $a\sigma$ in $\mathsf{Elem}(\Delta, A\sigma)$. Furthermore

$$a\ \mathsf{id} = a\ (a\sigma)\delta = a(\sigma\delta)$$

We have a *context extension operation*: if A in $\mathsf{Type}(\Gamma)$ we have a new context $\Gamma.A$. There is a projection $\mathsf{p} : \Gamma.A \to \Gamma$ and a special element q in $\mathsf{Elem}(\Gamma.A, A\mathsf{p})$. If $\sigma : \Delta \to \Gamma$ and A in $\mathsf{Type}(\Gamma)$ and a in $\mathsf{Elem}(\Delta, A\sigma)$ we have an extension operation $(\sigma, a) : \Delta \to \Gamma.A$. We should have

$$\mathsf{p}(\sigma, a) = \sigma\ \mathsf{q}(\sigma, a) = a\ (\sigma, a)\delta = (\sigma\delta, a\delta)\ (\mathsf{p}, \mathsf{q}) = \mathsf{id}$$

If a in $\mathsf{Elem}(\Gamma, A)$ we write $[a] = (\mathsf{id}, a) : \Gamma \to \Gamma.A$. Thus if B in $\mathsf{Type}(\Gamma.A)$ and a in $\mathsf{Elem}(\Gamma, A)$ we have $B[a]$ in $\mathsf{Type}(\Gamma)$. If furthermore b in $\mathsf{Elem}(\Gamma.A, B)$ we have $b[a]$ in $\mathsf{Elem}(\Gamma, B[a])$.

If $\sigma : \Delta \to \Gamma$ and A in $\mathsf{Type}(\Gamma)$ we define $\sigma^+ : \Delta.A\sigma \to \Gamma.A$ to be $(\sigma\mathsf{p}, \mathsf{q})$.

The extension operation can then be defined as $(\sigma, u) = [u]\sigma^+$. Thus instead of the extension operation, we could have chosen the operations $[u]$ and σ^+ as primitive, like in [Ehr88]. Our argument is independent of this choice of primitive operations.

We suppose furthermore one operation $\Pi\ A\ B$ such that $\Pi\ A\ B$ in $\mathsf{Type}(\Gamma)$ if A in $\mathsf{Type}(\Gamma)$ and B in $\mathsf{Type}(\Gamma.A)$. We should have $(\Pi\ A\ B)\sigma = \Pi\ (A\sigma)\ (B\sigma^+)$.

We have an abstraction operation λb in $\mathsf{Elem}(\Gamma, \Pi\ A\ B)$ for b in $\mathsf{Elem}(\Gamma.A, B)$ and an application operation $c\ a$ in $\mathsf{Elem}(\Gamma, B[a])$ for c in $\mathsf{Elem}(\Gamma, \Pi\ A\ B)$ and a in $\mathsf{Elem}(\Gamma, A)$. These operations should satisfy the equations

$$(\lambda b)\ a = b[a],\ c = \lambda(c\mathsf{p}\ \mathsf{q}),\ (\lambda b)\sigma = \lambda(b\sigma^+),\ (c\ a)\sigma = c\sigma\ (a\sigma)$$

We assume each set $\mathsf{Type}(\Gamma)$ to be stratified in $\mathsf{Type}_0(\Gamma) \subseteq \mathsf{Type}_1(\Gamma) \subseteq \dots$.

Each subset $\mathsf{Type}_n(\Gamma)$ is closed by dependent product, and we have U_n in $\mathsf{Type}_{n+1}(\Gamma)$ such that $\mathsf{Elem}(\Gamma, \mathsf{U}_n) = \mathsf{Type}_n(\Gamma)$.

Finally we assume U_0 to be impredicative and types in U_0 to be proof-irrelevant. *Impredicativity* means that $\Pi\ A\ B$ is in $\mathsf{Type}_0(\Gamma)$ if B is in $\mathsf{Type}_0(\Gamma.A)$ where A can be *any* type, and *proof-irrelevance* means that $a_0 = a_1 : \mathsf{Elem}(\Gamma, A)$ whenever A is in $\mathsf{Type}_0(\Gamma)$ and a_0 and a_1 are in $\mathsf{Elem}(\Gamma, A)$.

We think of types in $\mathsf{Type}_0(\Gamma)$ as proof-irrelevant propositions.

Note that, in an arbitrary model we may have some equality of the form[2] $\Pi\ A\ B = \mathsf{U}_0$ and the operations, like product operations, don't need to be injective.

1.2. **Examples of Models.** Like for equational theories, there is always the *terminal* model where all sorts are interpreted by a singleton.

P. Aczel in [Acz98] provides a model in in a impredicative intuitionistic set theory IZFu_ω, with intuitionistic versions of Grothendieck universes $\mathcal{V}_0, \mathcal{V}_1, \dots, \mathcal{V}_\omega$.

A context is interpreted as a set in $\mathcal{V}_\omega$, and $\mathsf{Type}(\Gamma)$ is interpreted by $\Gamma \to \mathcal{V}_\omega$. The lowest universe U_0 is interpreted by the set of *truth values* $\mathcal{V}_0$: the set of subsets of $1 = \{0\}$. In order to interpret the fact that U_0 is closed by arbitrary products, P. Aczel introduces a non-standard encoding of dependent products, see [Acz98], which we use in building our normalization model (see Appendix). This encoding of dependent products $\Pi_{x\in A}B(x)$ is such that $\Pi_{x\in A}B(x) \subseteq 1$ if we have $B(x) \subseteq 1$ for all x in A.

M. Hofmann [Hof97] shows how to refine a presheaf model over an arbitrary small category to a model of type theory. It models universes, and if we use Aczel's encoding of dependent products, we also get a model where the lowest universe $\mathcal{U}_0$ is interpreted by the presheaf of sieves. Using Aczel's non-standard encoding [Acz98] of dependent products mentioned above, we see that $\mathcal{U}_0$ is closed by dependent products of families valued in $\mathcal{U}_0$. We write $\mathcal{U}_0,\ \mathcal{U}_1,\ \dots$ for the universes corresponding to $\mathcal{V}_0,\ \mathcal{V}_1,\ \dots$

We will work in the last section with the *initial* or *term* model M_0 (see Appendix). This is the model where elements are syntactical expressions *modulo* equations/conversion rules. One important result which follows from the "normalization model" we present in the next section, is that equality is *decidable* for the initial model, and that *constructors are injective*; this means in particular that we cannot have an equality of the form $\mathsf{U}_0 = \Pi\ A\ B$ and that $\Pi\ A_0\ B_0 = \Pi\ A_1\ B_1$ in $\mathsf{Type}(\Gamma)$ implies $A_0 = A_1$ in $\mathsf{Type}(\Gamma)$ and $B_0 = B_1$ in $\mathsf{Type}(\Gamma.A_0)$. This injectivity property may not hold in general for an arbitrary model; for instance in the set model, we have $\emptyset^A = \emptyset$ for any non empty set A.

[2]This can even be the case *a priori* in the term model, though it follows from our proof that this is *not* the case.

2. Normalization Model

We present a variation of the model used in [Coq19]. As in [Coq19], we work in a suitable *presheaf* topos, but with a slight variation for the choice of the base category. We start from an arbitrary model M.

2.1. **Category of telescopes.** As in [Coq19], we define first the collection of *telescopes* $X, Y, Z, \ldots$. These are finite list $X = A_0, A_1, \ldots, A_{n-1}$ with A_0 in $\mathsf{Type}()$, A_1 in $\mathsf{Type}(().A_0)$ and so on. Any telescope X has an interpretation $\langle X\rangle$ which is a context of the model M, by taking $\langle X\rangle = ().A_0.A_1.\ldots.A_{n-1}$. We can have $\langle X\rangle = \langle Y\rangle$ in M without having $X = Y$. We write $()$ the empty telescope. If X is a telescope and A in $\mathsf{Type}\langle X\rangle$, we may write $X.A$ for X, A.

We can now define the base category of the presheaf model. A map $\alpha : Y \to_S X$ is a syntactical object defined inductively. We have $() : Y \to_S ()$ and if we have already define $\alpha : Y \to_S X$ then we can either add a type to Y getting $\alpha\mathsf{p} : Y, B \to_S X$, or we can add a type to X, getting $\alpha^+ : Y, A\langle\alpha\rangle \to_S X, A$. We define at the same time $\langle()\rangle$ by the clauses:

$$\langle()\rangle = () \ \langle\alpha\mathsf{p}\rangle = \langle\alpha\rangle\mathsf{p} \ \langle\alpha^+\rangle = \langle\alpha\rangle^+$$

A map $\alpha : Y \to_S X$ can be seen as a proof relevant witness that Y extends X (which was the relation used in [CG90]). It is direct to define a syntactical identity map $\mathsf{id}_S : X \to_S X$ by induction on X so that $\langle\mathsf{id}_S\rangle = \mathsf{id}$ and to define a composition operation. We get in this way a category $\mathcal{C}$ of telescopes[3].

We can also define a syntactic projection map $\mathsf{p}_S : X.A \to X$ such that $\langle\mathsf{p}_S\rangle = \mathsf{p}$ by induction on X. This category of syntactic extensions will be the base category $\mathcal{C}$ for the presheaf topos $\hat{\mathcal{C}}$ in which we define the normalization model[4].

2.2. **Syntactic expressions.** We introduce, for A in $\mathsf{Type}\langle X\rangle$, the set $\mathsf{Term}(X, A)$. This is a set of *syntactical expressions*. Contrary to the set $\mathsf{Elem}(\langle X\rangle, A)$, these expressions are *not* quotiented up to conversion. Also the syntactical expressions don't use explicit substitutions and can be thought of as annotated λ-expressions.

The *syntactical* expressions are described by the following grammar

$$K, L, k \ ::= \ v_n \mid \mathsf{U}_n \mid \mathsf{app}\ K\ L\ k\ k \mid \lambda\ K\ K\ k \mid \Pi\ K\ L \mid 0$$

where v_n are de Bruijn index. This forms a set with a *decidable* equality. We define then inductively for A in $\mathsf{Type}\langle X\rangle$ a *subset* $\mathsf{Term}(X, A)$ of this set of syntactical expressions. Each such set $\mathsf{Term}(X, A)$ is then also a set with a decidable equality. If k is in $\mathsf{Term}(X, A)$ we define by induction on k an element $\langle k\rangle$ in $\mathsf{Elem}(\langle X\rangle, A)$. This can be thought of as the *interpretation* of the syntactical expression k. We can also see the map

$$k \mapsto \langle k\rangle \ \mathsf{Term}(X, A) \to \mathsf{Elem}(\langle X\rangle, A)$$

as a *quotient* map.

We have U_n in $\mathsf{Term}(X, \mathsf{U}_l)$ if $n < l$ and $\langle\mathsf{U}_n\rangle = \mathsf{U}_n$.

We have v_0 in $\mathsf{Term}(X.A, A\mathsf{p})$ and v_{n+1} in $\mathsf{Term}(X.A, B\mathsf{p})$ if v_n is in $\mathsf{Term}(X, B)$.

[3]It would also have been possible to use *renaming* as maps, as in [Coq19, Ste22, Gra22].

[4]The use of context as world for a normalization argument goes back to [CG90]. It was introduced there as a solution of the problem of having empty types, problem which was solved in [Gir71] by the introduction of a constant in all types with special reduction rules.

We let $\langle v_n\rangle$ to be qp^n and $\langle \mathsf{U}_l\rangle = \mathsf{U}_l$.

If K is in $\mathsf{Term}(X, \mathsf{U}_n)$ and L in $\mathsf{Term}(X.\langle K\rangle, \mathsf{U}_n)$ then $\Pi\ K\ L$ is in $\mathsf{Term}(X, \mathsf{U}_n)$ and $\langle \Pi\ K\ L\rangle = \Pi\ \langle K\rangle\ \langle L\rangle$. If furthermore k' is in $\mathsf{Term}(X, \langle \Pi\ K\ L\rangle)$ and k in $\mathsf{Term}(X, \langle K\rangle)$ then $\mathsf{app}\ K\ L\ k'\ k$ is in $\mathsf{Term}(X, \langle L\rangle[\langle k\rangle])$ and then $\langle \mathsf{app}\ K\ L\ k'\ k\rangle = \langle k'\rangle\ \langle k\rangle$.

If K is in $\mathsf{Term}(X, \mathsf{U}_n)$ and L in $\mathsf{Term}(X.\langle K\rangle, \mathsf{U}_n)$ and t is in $\mathsf{Term}(X.\langle K\rangle, \langle L\rangle)$ then $\lambda\ K\ L\ t$ is in $\mathsf{Term}(X, \langle \Pi\ K\ L\rangle)$ and $\langle \lambda\ K\ L\ t\rangle = \lambda\ \langle t\rangle$.

If K is in $\mathsf{Term}(X, \mathsf{U}_l)$ and $l \leq n$ then K is in $\mathsf{Term}(X, \mathsf{U}_n)$.

One *key* addition to this notion of syntactical expressions, introduced in order to deal with proof-irrelevant propositions, is the special constant 0. We have 0 in $\mathsf{Term}(X, A)$ whenever A is in $\mathsf{Type}_0(\langle X\rangle)$ and $\mathsf{Elem}(\langle X\rangle, A)$ is *inhabited*.

Since $\mathsf{Elem}(\langle X\rangle, A)$ is a *subsingleton* we can define $\langle 0\rangle$ to be any element u of $\mathsf{Elem}(\langle X\rangle, A)$. (We don't need any choice since $u = v$ if u and v are in $\mathsf{Elem}(\langle X\rangle, A)$.)

If u is in $\mathsf{Elem}(\langle X\rangle, A)$ we write $\mathsf{Term}(X, A)|u$ the subset of syntactical expressions k in $\mathsf{Term}(X, A)$ such that $\langle k\rangle = u$.

Like in [Coq19], we need to define two subsets of $\mathsf{Term}(X, A)$, the subsets $\mathsf{Norm}(X, A)$ of *normal* terms and $\mathsf{Neut}(X, A)$ of *neutral* terms. These are defined inductively by the following clauses.

We have v_0 in $\mathsf{Neut}(X.A, A\mathsf{p})$ and v_{n+1} in $\mathsf{Neut}(X.A, B\mathsf{p})$ if v_n is in $\mathsf{Neut}(X, B)$.

We have $\mathsf{app}\ K\ L\ k\ t$ in $\mathsf{Neut}(X, \langle L\rangle[\langle t\rangle])$ if K in $\mathsf{Norm}(X, \mathsf{U}_n)$ and L in $\mathsf{Norm}(X.\langle K\rangle, \mathsf{U}_n)$ and k in $\mathsf{Neut}(X, \langle \Pi\ K\ L\rangle)$ and t in $\mathsf{Norm}(X, \langle K\rangle)$.

We have $\lambda\ K\ L\ t$ in $\mathsf{Norm}(X, \langle \Pi\ K\ L\rangle[\langle t\rangle])$ if K in $\mathsf{Norm}(X, \mathsf{U}_n)$ and L in $\mathsf{Norm}(X.\langle K\rangle, \mathsf{U}_n)$ and k in $\mathsf{Neut}(X, \langle \Pi\ K\ L\rangle)$ and t in $\mathsf{Norm}(X, \langle K\rangle)$.

We have $\Pi\ K\ L$ in $\mathsf{Norm}(X, \mathsf{U}_n)$ if K in $\mathsf{Norm}(X, \mathsf{U}_n)$ and L in $\mathsf{Norm}(X.\langle K\rangle, \mathsf{U}_n)$.

We have K in $\mathsf{Norm}(X, \mathsf{U}_n)$ if K is in $\mathsf{Neut}(X, \mathsf{U}_l)$ and $l \leq n$.

We have U_l in $\mathsf{Norm}(X, \mathsf{U}_n)$ if $l < n$.

We have 0 in $\mathsf{Norm}(X, K)$ if K is in $\mathsf{Neut}(X, \mathsf{U}_0)$ and $\mathsf{Elem}(\langle X\rangle, \langle K\rangle)$ is inhabited

We have k in $\mathsf{Norm}(X, K)$ if K is in $\mathsf{Neut}(X, \mathsf{U}_n)$ with $n > 0$ and k is in $\mathsf{Neut}(X, K)$.

As in [Hof97, Coq19], we freely use the notations of type theory for operations in the presheaf topos $\hat{\mathcal{C}}$. In this presheaf models we have a cumulative sequence of universe $\mathcal{U}_n$, for $n = 0, 1, \ldots, \omega$. Furthermore, as noticed above, $\mathcal{U}_0$ inherits from $\mathcal{V}_0$ the fact that it is closed by arbitrary products.

In this model, we have a family of types Type_n (in the universe $\mathcal{U}_1$) with families of types $\mathsf{Elem}(T)$ and $\mathsf{Term}(T)$ for $T : \mathsf{Type}_n$. We have two subtypes $\mathsf{Norm}(T)$ and $\mathsf{Neut}(T)$ of $\mathsf{Term}(T)$. We also have an interpretation function $\mathsf{Term}(T) \to \mathsf{Elem}(T)$. Because of our choice of morphisms for the category of telescopes, each $\mathsf{Term}(T)$ has (internally) a decidable equality.

2.3. **Artin Glueing.** We define now a pseudomorphism [KHS19] between the model M and the presheaf model $\hat{\mathcal{C}}$. The normalisation model M^* will be a refinement of the glued model [KHS19] along this pseudomorphism.

To each context Γ in M, we associate a presheaf $|\Gamma|$ of $\hat{\mathcal{C}}$ by taking $|\Gamma|(X)$ to be the set $\langle X\rangle \to \Gamma$, with restriction maps $\rho \mapsto \rho\alpha = \rho\langle\alpha\rangle$ for $\alpha : Y \to_S X$.

Each element A in $\mathsf{Type}_n(\Gamma)$ in the model M defines then a presheaf map $|A| : |\Gamma| \to \mathsf{Type}_n$, by $\rho \mapsto A\rho$. Similarly, each element a in $\mathsf{Elem}(\Gamma, A)$ in the model M defines a global element $|a| : \Pi_{\rho:|\Gamma|}\mathsf{Elem}(|A|\rho)$.

For any A in $\mathsf{Type}_n(\Gamma)$ in M, we have a constant in the presheaf model $\hat{\mathcal{C}}$

$$\mathsf{mk} : \Pi_{\rho:|\Gamma|}|A|\rho \to |\Gamma.A|$$

and projections $\mathsf{fst} : |\Gamma.A| \to |\Gamma|$ and $\mathsf{snd} : \Pi_{\nu:|\Gamma.A|}|A|(\mathsf{fst}\ \nu)$ satisfying the equations

$$\mathsf{fst}\ (\mathsf{mk}\ \rho\ u) = \rho\ \mathsf{snd}\ (\mathsf{mk}\ \rho\ u) = u\ \nu = \mathsf{mk}\ (\mathsf{fst}\ \nu)\ (\mathsf{snd}\ \nu)$$

This defines a pseudomorphism between the model M and the model $\hat{\mathcal{C}}$.

Given B in $\mathsf{Type}_n(\Gamma.A)$, let us write $C = \Pi\ A\ B$ in $\mathsf{Type}_n(\Gamma)$. If $\rho : |\Gamma|$ and w in $\mathsf{Elem}(|C|\rho)$ and u in $\mathsf{Elem}(|A|\rho)$ we can define $w\ u$ in $\mathsf{Elem}(|B|(\mathsf{mk}\ \rho\ u))$, which is levelwise the application.

Lemma 2.1. *In the presheaf topos $\hat{\mathcal{C}}$, we have the following operations, for $\rho : |\Gamma|$ and $K : \mathsf{Norm}(\mathsf{U}_n)$ such that $\langle K\rangle = |A|\rho$ and $G : \Pi_{k:\mathsf{Neut}(A\rho)}\mathsf{Norm}(\mathsf{U}_n)$ such that $\langle Gk\rangle = |B|(\mathsf{mk}\ \rho\ \langle k\rangle)$:*

(1) $\Pi_S\ K\ G : \mathsf{Norm}(\mathsf{U}_n)$ *such that* $\langle \Pi_S\ K\ G\rangle = |C|\rho$,
(2) $\lambda_S\ g : \mathsf{Norm}(|C|\rho)|w$ *for* w *in* $\mathsf{Elem}(|C|\rho)$ *and* $g : \Pi_{k:\mathsf{Neut}(A\rho)}\mathsf{Norm}(|B|(\mathsf{mk}\ \rho\ \langle k\rangle))|(w\ \langle k\rangle)$,
(3) $\mathsf{app}_S\ K\ G\ k'\ k : \mathsf{Neut}(|B|(\mathsf{mk}\ \rho\ u))|(w\ u)$ *for* w *in* $\mathsf{Elem}(|C|\rho)$ *and* u *in* $\mathsf{Elem}(|A|\rho)$ *and* $k' : \mathsf{Neut}(|C|\rho)|w$ *and* $k : \mathsf{Norm}(|A|\rho)|u$.

Proof. We prove the first point, the arguments for the two other points being similar.

We have to define $\Pi_S\ K\ G$ in $\mathsf{Term}(X, \mathsf{U}_n)$ such that $\langle \Pi_S\ K\ G\rangle = C\rho$. Here ρ is in $\langle X\rangle \to \Gamma$ and K is in $\mathsf{Norm}(X, \mathsf{U}_n)$ and such that $\langle K\rangle = A\rho$. Furthermore, G is an operation such that $G\alpha\ k$ is an element of $\mathsf{Term}(Y, \mathsf{U}_n)$ satisfying $\langle G\alpha\ k\rangle = B(\rho\alpha, \langle k\rangle)$ for $\alpha : Y \to X$ and k in $\mathsf{Term}(Y, A\rho\alpha)$ and satisfying $(G\alpha\ k)\alpha_1 = G(\alpha\alpha_1)\ (k\alpha_1)$, for $\alpha_1 : Z \to_S Y$.

We then take $\Pi_S\ K\ G$ to be $\Pi\ K\ (G\mathsf{p}_S\ v_0)$.

We have $\langle \Pi_S\ K\ G\rangle = \Pi\ \langle K\rangle\ \langle G\mathsf{p}_S\ v_0\rangle$ and $\langle K\rangle = A\rho$ and $\langle G\mathsf{p}_S\ v_0\rangle = B(\rho\mathsf{p}, \mathsf{q})$.

If $\alpha : Y \to_S X$ we have $(\Pi_S\ K\ G)\alpha = (\Pi\ K\ (G\mathsf{p}_S\ v_0))\alpha = \Pi\ K\alpha\ (G\mathsf{p}_S\ v_0)(\alpha\mathsf{p}_S, v_0)$ and $(G\mathsf{p}_S\ v_0)(\alpha\mathsf{p}_S, v_0) = G\alpha\mathsf{p}_S\ v_0$, so the operation Π_S is functorial. □

3. Normalization model

3.1. **Internal definitions.** The first definitions are purely internal to the model $\hat{\mathcal{C}}$.

For T in Type_n, we define $\mathsf{Type}'_n(T)$ to be the set of 4-tuples $(T', K, \mathsf{q}_T, \mathsf{r}_T)$ where[5]

(1) T' is in $\mathsf{Elem}(T) \to \mathcal{U}_n$,
(2) K is in $\mathsf{Norm}(\mathsf{U}_n)|T$,
(3) q_T, a "quote" function, is in $\Pi_{u:\mathsf{Elem}(T)}T'u \to \mathsf{Norm}(T)|u$,

[5]This definition goes back to the unpublished paper [TAS97] for system F; one contribution of [Coq19] is to explain how to treat universes and general dependent products, and the contribution of the present paper is to extend this to an impredicative universe of proof irrelevant propositions.

(4) r_T, a "reflect" function, is in $\Pi_{k:\mathsf{Neut}(T)} T'\langle k\rangle$.

We define $\mathsf{q}_{\mathsf{U}_n}\ A\ (A', K, \mathsf{q}_A, \mathsf{r}_A) = K$.

For $n > 0$ and K in $\mathsf{Neut}(\mathsf{U}_n)$ we define $\mathsf{r}_{\mathsf{U}_n}\ K$ to be $(K', K, \mathsf{q}_K, \mathsf{r}_K)$ where $K'u$ is $\mathsf{Norm}(K)|u$ and $\mathsf{q}_K\ u\ u' = u'$ and $\mathsf{r}_K\ k = k$.

For $n = 0$, and K in $\mathsf{Neut}(\mathsf{U}_n)$, we define $\mathsf{r}_{\mathsf{U}_n}\ K$ to be $(K', K, \mathsf{q}_K, \mathsf{r}_K)$ where $K'u$ is $\{0\}$ and[6] $\mathsf{q}_K\ u\ u' = 0$ and $\mathsf{r}_K\ k = 0$.

3.2. **The glued model for normalization.** We can now define the normalization model M^*, where a context is a pair Γ, Γ' where Γ is a context of M and Γ' is a dependent family over $|\Gamma|$ in the model $\hat{\mathcal{C}}$.

We define $()'$ to be the constant family of constant presheaf $\{0\}$.

The set $\mathsf{Type}^*_n(\Gamma, \Gamma')$ is defined to be the set of pairs $A, \overline{A}$ where A is in $\mathsf{Type}^M_n(\Gamma)$ and $\overline{A}$ is a global element of

$$\Pi_{\rho:|\Gamma|}\Gamma'(\rho) \to \mathsf{Type}'_n(|A|\rho)$$

An element of this type $A, \overline{A}$ is a pair $a, \overline{a}$ where a is in $\mathsf{Elem}^M(\Gamma, A)$ and $\overline{a}$ is a global element of

$$\Pi_{\rho:|\Gamma|}\Pi_{\rho':\Gamma'(\rho)}\overline{A}\rho\rho'.1(|a|\rho)$$

We define $\overline{\mathsf{U}_n} = \mathsf{U}_n, \mathsf{Type}'_n, \mathsf{q}_{\mathsf{U}_n}, \mathsf{r}_{\mathsf{U}_n}$ and U^*_n is the pair $\mathsf{U}_n, \overline{\mathsf{U}_n}$.

The extension operation is defined by $(\Gamma, \Gamma').(A, \overline{A}) = \Gamma.A, (\Gamma.A)'$ where $(\Gamma.A)'(\rho, u)$ is the set of pairs ρ', u' with $\rho' \in \Gamma'(\rho)$ and u' in $\overline{A}\rho\rho'.1(u)$.

As in [Coq19], we define a new operation $\Pi^*\ (A, \overline{A})\ (B, \overline{B}) = C, \overline{C}$ where $C = \Pi\ A\ B$.

We write $(T', K, \mathsf{q}_T, \mathsf{r}_T) = \overline{A}\rho\rho'$ in $\mathsf{Type}'_n(|A|\rho)$ and for each u in $\mathsf{Elem}(|A|\rho)$ and u' in $T'(u)$ we write

$$(F'uu', F_0uu', \mathsf{q}_Fuu', \mathsf{r}_Fuu') = \overline{B}(\mathsf{mk}\ \rho\ u)(\rho', u')$$

in $\mathsf{Type}'_n(|B|(\mathsf{mk}\ \rho\ u))$. We then define $\overline{C}\rho\rho'$ in $\mathsf{Type}'_n(|C|\rho)$ to be the tuple

- $R'(w) = \Pi_{u:\mathsf{Elem}(|A|\rho)}\Pi_{u':T'(u)}F'uu'(wu)$
- $L = \Pi_S\ K\ G$
- $\mathsf{q}_R\ w\ w' = \lambda_S\ g$
- $(\mathsf{r}_R\ k)uu' = \mathsf{r}_Fuu'(\mathsf{app}_S\ K\ G\ k\ (\mathsf{q}_Tuu'))$

where G is the function $G\ k = F_0\langle k\rangle(\mathsf{r}_T\ k)$ and g the function $g\ k = \mathsf{q}_F\langle k\rangle(\mathsf{r}_Tk)(w\ \langle k\rangle)(w'\langle k\rangle(\mathsf{r}_Tk))$.

We can check using Lemma 2.1 that $R', L, \mathsf{q}_R, \mathsf{r}_R$ is an element of $\mathsf{Type}'_n(|C|\rho)$.

We get in this way a new model M^* with a projection map $M^* \to M$.

[6] This is well-defined since u is in $\mathsf{Elem}(\langle K\rangle)$ and so 0 is in $\mathsf{Norm}(\langle K\rangle)$.

4. Applications of the normalization model

For the term model M_0, we have an initial map $M_0 \to M_0^*$ which is a section of this projection map. In this case, the contexts of M_0 are the same as telescopes and we have $\langle X \rangle = X$.

For each context Γ of M_0, we can hence compute, using this section, Γ' which is *internally* a dependent family over $|\Gamma|$. *Externally*, this is given by a family of sets $\Gamma'(\Delta, \rho)$ for $\rho : \Delta \to \Gamma$ with restriction maps $\rho' \mapsto \rho'\alpha$ for $\alpha : \Delta_1 \to_S \Delta$.

For A in $\mathsf{Type}(\Gamma)$ let us write $A'\rho\rho'$ for $(\overline{A}\rho\rho').1$ and $\mathsf{r}_A\rho\rho'$ for $(\overline{A}\rho\rho').4$, which, internally, is a function in $\Pi_{k:\mathsf{Neut}(|A|\rho)} A'\rho\rho'\langle k\rangle$, Externally, this can be seen as a function $\mathsf{r}_A(\Delta, \rho)\rho' k$ in $A'(\Delta, \rho)\rho'\langle k\rangle$ for $\rho : \Delta \to \Gamma$ and ρ' in $\Gamma'(\Delta, \rho)$ and k in $\mathsf{Neut}(\Delta, A\rho)$. This function satisfies $(\mathsf{r}_A(\Delta, \rho)\rho' k)\alpha = \mathsf{r}_A(\Delta_1, \rho\alpha)(\rho'\alpha)(k\alpha)$ for $\alpha : \Delta_1 \to_S \Delta$. Similarly we define $\mathsf{q}_A\rho\rho'$ to be $(\overline{A}\rho\rho').3$.

For the two main applications of this normalization model, we first build id'_Γ in $\Gamma'(\Gamma, \mathsf{id})$. The definition is by induction on Γ.

For $\Gamma = ()$ we take[7] $\mathsf{id}'_\Gamma = 0$.

If we have defined id'_Γ in $\Gamma'(\Gamma, \mathsf{id})$ and A is in $\mathsf{Type}(\Gamma)$, let $\Delta = \Gamma.A$. We have $\mathsf{p} : \Delta \to \Gamma$ and $\mathsf{p}_S : \Delta \to_S \Gamma$. Let ρ' be $\mathsf{id}'_\Gamma \mathsf{p}_S$ in $\Gamma'(\Delta, \mathsf{p})$; we can define $\mathsf{id}'_\Delta = \rho', \mathsf{r}_A(\Delta, \mathsf{p})\rho' v_0$.

If A is in $\mathsf{Type}(\Gamma)$ we can compute $\overline{A}\ \mathsf{id}\ \mathsf{id}' = (T', K, \mathsf{q}_T, \mathsf{r}_T)$ and we define $\mathsf{reify}(A)$ to be $(\overline{A}\ \mathsf{id}\ \mathsf{id}').2 = K$. We have $\langle \mathsf{reify}(A)\rangle = A$ since $\langle \mathsf{reify}(A)\rangle = A\ \mathsf{id} = A$. If furthermore a is in $\mathsf{Elem}(\Gamma, A)$ we define $\mathsf{reify}(a)$ in $\mathsf{Norm}(\Gamma, A)$ to be $\mathsf{q}_A\ \mathsf{id}\ \mathsf{id}'\ a\ (\overline{a}\ \mathsf{id}\ \mathsf{id}')$. We have $\langle \mathsf{reify}(a)\rangle = a$ in $\mathsf{Elem}(\Gamma, A)$.

We can summarize this discussion as follows.

Theorem 4.1. *For each context Γ, the quotient map $k \mapsto \langle k\rangle$, $\mathsf{Term}(\Gamma, A) \to \mathsf{Elem}(\Gamma, A)$ has a section $a \mapsto \mathsf{reify}(a)$.*

Corollary 4.2. *Equality in M_0 is decidable.*

Proof. If a and b are in $\mathsf{Elem}(\Gamma, A)$ we have $\mathsf{reify}(a) = \mathsf{reify}(b)$ in $\mathsf{Term}(\Gamma, A)$ if, and only if, $a = b$ in $\mathsf{Elem}(\Gamma, A)$. The result then follows from the fact that the equality in $\mathsf{Term}(\Gamma, A)$ is decidable. □

We also can prove that Π is one-to-one for conversions, following P. Hancock's argument presented in [ML75]. The following Lemma follows from the definition of reify.

Lemma 4.3. *For A in $Type(\Gamma)$ and B in $\mathsf{Type}(\Gamma.A)$, we have $\mathsf{reify}(\Pi\ A\ B) = \Pi\ \mathsf{reify}(A)\ \mathsf{reify}(B)$.*

Corollary 4.4. *If $\Pi\ A_0\ B_0 = \Pi\ A_1\ B_1$ in $\mathsf{Type}(\Gamma)$ in the term model, we have $A_0 = A_1$ in $\mathsf{Type}(\Gamma)$ and $B_0 = B_1$ in $\mathsf{Type}(\Gamma.A_0)$.*

Proof. We have $\mathsf{reify}(\Pi\ A_0\ B_0) = \Pi\ \mathsf{reify}(A_0)\ \mathsf{reify}(B_0) = \Pi\ \mathsf{reify}(A_1)\ \mathsf{reify}(B_1) = \mathsf{reify}(\Pi\ A_1\ B_1)$ as syntactical expressions, and hence $\mathsf{reify}(A_0) = \mathsf{reify}(A_1)$. This implies $A_0 = A_1$ in $\mathsf{Type}(\Gamma)$. We then have $\mathsf{reify}(B_0) = \mathsf{reify}(B_1)$, which implies similarly $B_0 = B_1$ in $\mathsf{Type}(\Gamma.A_0)$. □

Corollary 4.5 (Subject reduction)**.** *If $(\lambda b)\ a$ is in $\mathsf{Elem}(\Gamma, D)$ then $b[a]$ is in $\mathsf{Elem}(\Gamma, D)$.*

[7] We defined $\Gamma'(\Delta, \rho)$ to be the constant $1 = \{0\}$ in this case.

Proof. We have b in $\mathsf{Elem}(\Gamma.A, B)$ and a in A' and λb in $\Pi\ A'\ B'$ with $\Pi\ A\ B = \Pi\ A'\ B'$ and $D = B'[a]$. By the previous Corollary, we have $A = A'$ and $B = B'$ and $b[a]$ is in $\mathsf{Elem}(\Gamma, B[a]) = \mathsf{Elem}(\Gamma, B'[a]) = \mathsf{Elem}(\Gamma, D)$. □

We can define a normal form function $\mathsf{nf} : \mathsf{Term}(\Gamma, A) \to \mathsf{Norm}(\Gamma, A)$ by $\mathsf{nf}(k) = \mathsf{reify}(\langle k \rangle)$.

By mutual induction, we can show the following.

Lemma 4.6. *If t is in $\mathsf{Norm}(\Gamma, A)$ then $t = \mathsf{reify}(\langle t \rangle)$ and if k is in $\mathsf{Neut}(\Gamma, A)$ then $\mathsf{r}_A\ \mathsf{id}\ \mathsf{id}'\ k = \overline{\langle k \rangle}\ \mathsf{id}\ \mathsf{id}'$.*

Corollary 4.7. *We have $\mathsf{nf}(\mathsf{nf}(t)) = \mathsf{nf}(t)$ for any t in $\mathsf{Term}(\Gamma, A)$.*

Corollary 4.8. *The section map $\mathsf{reify} : \mathsf{Elem}(\Gamma, A) \to \mathsf{Term}(\Gamma, A)$ is natural in Γ w.r.t. the morphisms in the telescope category $\mathcal{C}$.*

Proof. If $\alpha : \Delta \to \Gamma$ is a morphism in $\mathcal{C}$ and a is in $\mathsf{Elem}(\Gamma, A)$ and $t = \mathsf{reify}(a)$ we have t in $\mathsf{Norm}(\Gamma, A)$ and $t\alpha$ in $\mathsf{Norm}(\Delta, A\alpha)$ with $\langle t\alpha \rangle = \langle t \rangle \alpha = a\alpha$. By the previous Lemma, we get $\mathsf{reify}(a\alpha) = \mathsf{reify}(\langle t\alpha \rangle) = t\alpha = \mathsf{reify}(a)\alpha$. □

This implies that, in the presheaf model $\hat{\mathcal{C}}$ the interpretation map $\mathsf{Term}(T) \to \mathsf{Elem}(T)$ for T in Type has a section $\mathsf{Elem}(T) \to \mathsf{Term}(T)$. Furthermore, $\mathsf{Norm}(T)$, which has internally a decidable equality, is isomorphic to $\mathsf{Elem}(T)$.

5. Conclusion

This note can be seen as a weak "positive" complement of the "negative" result in [AC20], in the sense, that, in the absence of the problematic cast function analysed in [AC20], we do have normalization and decidability of conversion.

Our argument extends to the addition of dependent sum types with surjective pairing, or inductive types. In general, inductive types have to be declared in some universe U_n with $n > 0$.

Note that it is possible to define the absurd proposition $\bot$ in U_0 as $\Pi_{X:\mathsf{U}_0} X$ and to add the large elimination rule $\bot \to A$ for *any* type A while preserving decidability of equality.

A natural question is what happens if we consider a proof *relevant* impredicative type of propositions. In a companion paper, we show that the present technique extends also to this case.

Acknowledgement

Many thanks to Daniel Gratzer for many discussions on the topic of normalization. In particular, Daniel convinced me that one could define the set of normal terms in such a way that one gets exactly the β-normal η-*long* normal terms (ideas also present in [Ste22]), contrary to what I was doing in [Coq19], and this was crucial for some results in the last section. Many thanks also to the referee for many relevant comments.

References

[AC20] Andreas Abel and Thierry Coquand. Failure of normalization in impredicative type theory with proof-irrelevant propositional equality. *Log. Methods Comput. Sci.*, 16(2), 2020. doi:10.23638/LMCS-16(2:14)2020.

[ACP09] Andreas Abel, Thierry Coquand, and Miguel Pagano. A modular type-checking algorithm for type theory with singleton types and proof irrelevance. In Pierre-Louis Curien, editor, *Typed Lambda Calculi and Applications, 9th International Conference, TLCA 2009, Brasilia, Brazil, July 1-3, 2009. Proceedings*, volume 5608 of *Lecture Notes in Computer Science*, pages 5–19. Springer, 2009. doi:10.1007/978-3-642-02273-9_3.

[Acz98] Peter Aczel. On Relating Type Theories and Set Theories. In T. Altenkirch, B. Reus, and W. Naraschewski, editors, *Types for Proofs and Programs*, pages 33–46. Springer, 1998.

[Car19] M. Carneiro. The type theory of lean. Master Thesis, Carnegie-Mellon University, 2019.

[CG90] Th. Coquand and J. Gallier. A Proof of Strong Normalisation for the Theory of Constructions using a Kripke-Like Interpretation. In *Proceeding of first meeting on Logical Framework*, 1990.

[Coq86] Th. Coquand. An Analysis of Girard's Paradox. In *Proceedings of the Symposium on Logic in Computer Science (LICS '86), Cambridge, Massachusetts, USA, June 16-18, 1986*, pages 227–236. IEEE Computer Society, 1986.

[Coq19] Th. Coquand. Canonicity and normalization for dependent type theory. *Theor. Comput. Sci.*, 777:184–191, 2019.

[dB95] N.G. de Bruijn. Some extensions of Automath: The AUT-4 family. In J. Geuvers R. Nederpelt and R. de Vrijer, editors, *Selected Papers on Automath*, pages 283–288. Elsevier, 1995.

[Ehr88] Th. Ehrhard. Une sémantique catégorique des types dépendents. PhD thesis, 1988.

[Gir71] J.-Y. Girard. Une extension de l'interprétation de Gödel à l'analyse, et son application à l'élimination des coupures dans l'analyse et la théorie des types.(An extension of Gödel's interpretation to analysis and its application to cut elimination in analysis and type theory). Proc. 2nd Scandinav. Logic Sympos. 1970, Studies Logic Foundations Math. 63, 63-92 (1971)., 1971.

[Gra22] Daniel Gratzer. Normalization for multimodal type theory. In Christel Baier and Dana Fisman, editors, *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, pages 2:1–2:13. ACM, 2022. doi:10.1145/3531130.3532398.

[Hof97] Martin Hofmann. Syntax and semantics of dependent types. In *Semantics and logics of computation (Cambridge, 1995)*, volume 14 of *Publ. Newton Inst.*, pages 79–130. Cambridge Univ. Press, Cambridge, 1997. doi:10.1017/CBO9780511526619.004.

[KHS19] Ambrus Kaposi, Simon Huber, and Christian Sattler. Gluing for type theory. In Herman Geuvers, editor, *4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019, Dortmund, Germany*, volume 131 of *LIPIcs*, pages 25:1–25:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.FSCD.2019.25.

[Miq04] A. Miquel. Lamda-Z: Zermelo's Set Theory as a PTS with 4 Sorts. In Jean-Christophe Filliâtre, Christine Paulin-Mohring, and Benjamin Werner, editors, *Types for Proofs and Programs, International Workshop, TYPES 2004, Jouy-en-Josas, France, December 15-18, 2004, Revised Selected Papers*, volume 3839 of *Lecture Notes in Computer Science*, pages 232–251. Springer, 2004.

[ML75] P. Martin-Löf. An intuitionistic theory of types: Predicative part. In H. E. Rose and J. Sheperdson, editors, *Studies in Logic and the Foundations of Mathematics*, volume 80, pages 73–118. Elsevier, 1975.

[PV07] Erik Palmgren and Steven J. Vickers. Partial horn logic and cartesian categories. *Ann. Pure Appl. Log.*, 145(3):314–353, 2007. doi:10.1016/j.apal.2006.10.001.

[Ste22] Jonathan Sterling. *First Steps in Synthetic Tait Computability: The Objective Metatheory of Cubical Type Theory*. PhD thesis, Carnegie Mellon University, USA, 2022. doi:10.1184/r1/19632681.v1.

[TAS97] M. Hofmann Th. Altenkirch and Th. Streicher. Reduction-free normalisation for system f. Unpublished note, 1997.

Appendix A. Variations on the formulation of the system

Our formal system is not a generalised algebraic theory, presenting the sort $\mathsf{Type}(\Gamma)$ as stratified by $\mathsf{Type}_n(\Gamma)$ and requiring $\mathsf{Elem}(\Gamma, \mathsf{U}_n) = \mathsf{Type}_n(\Gamma)$. It would instead have been possible to use coercion functions $T_n(X)$ in $\mathsf{Type}(\Gamma)$ and $T^l_n(X)$ in $\mathsf{Elem}(\Gamma, \mathsf{U}_l)$ for X in $\mathsf{Elem}(\Gamma, \mathsf{U}_n)$ with $T_l(T^l_n(X)) = T_n(X)$ for $l \leq n$. One would then also need a dependent product operation $\Pi^n\ X\ Y$ in $\mathsf{Elem}(\Gamma, \mathsf{U}_n)$ with $T_n(\Pi^n\ X\ Y) = \Pi\ T_n(X)\ T_n(Y)$ for $n > 0$ and the impredicative dependent product $\Pi^0\ A\ Y$ in $\mathsf{Elem}(\Gamma, \mathsf{U}_0)$ with $T_0(\Pi^0\ A\ Y) = \Pi\ A\ T_0(Y)$. One can then apply e.g. [PV07] to justify the existence of an initial model. We can see our system as an informal notation used to simplify the presentation.

One can wonder how crucial is the use of P. Aczel's encoding of dependent product [Acz98] which justifies the equality $T_0(\Pi^0\ A\ Y) = \Pi\ A\ T_0(Y)$. Without this encoding, we only have one isomorphism between $T_0(\Pi^0\ A\ Y)$, which is a subset of 1, and $\Pi\ A\ T_0(Y)$, which is also a subsingleton, but may not be a subset of 1. The following argument, due to M. Shulman, provides a more modular solution to this issue, which is independent of the way one encodes dependent product in the underlying set theory. One replaces the set model M by a new model M' with the same notion of context but letting $\mathsf{Type}'(\Gamma)$ to be the disjoint sum $\mathsf{Type}(\Gamma) + \mathsf{Elem}(\Gamma, \mathsf{U}_0)$. It is then possible to define by case a new product operation so that we get a strict equality $T_0(\Pi^0\ A\ Y) = \Pi\ A\ T_0(Y)$.