Sets in Types, Types in Sets

Benjamin Werner

INRIA–Rocquencourt BP 105
78 153 LE CHESNAY cedex
FRANCE
benjamin.werner@inria.fr

Abstract. We present two mutual encodings, respectively of the Calculus of Inductive Constructions in Zermelo-Frænkel set theory and the opposite way. More precisely, we actually construct two families of encodings, relating the number of universes in the type theory with the number of inaccessible cardinals in the set theory. The main result is that both hierarchies of logical formalisms interleave w.r.t. expressive power and thus are essentially equivalent. Both encodings are quite elementary: type theory is interpreted in set theory through a generalization of Coquand's simple proof-irrelevance interpretation. Set theory is encoded in type theory using a variant of Aczel's encoding; we have formally checked this last part using the Coq proof assistant.

1 Introduction

This work is an attempt towards better understanding of the expressiveness of powerful type theories. We here investigate the Calculus of Inductive Constructions (CIC); this formalism is, with some variants, the one implemented in the proof systems Coq [6] and Lego. It is essentially a typed λ-calculus with the following features:

- Dependent types, allowing the representation of propositions as types, through the Curry-Howard isomorphism.
- An impredicative level, i.e. the calculus is an extension of Girard's system F, allowing polymorphic types and thus impredicative reasoning.
- A hierarchy of predicative universes, quite similar to Martin-Löfs' [17].
- Inductive types, generalizing the primitive integers of Gödel's system T.

From the point of view of normalization and/or consistency proofs, the combination of these different features is still not fully understood. Known normalization proofs, as well as model constructions, make use of inaccessible cardinals, i.e. go beyond usual set theory. The question we try to address in the present work is whether inaccessible cardinals are necessary to build a model of CIC, or equivalently whether CIC is as strong as usual Zermelo-Frænkel set theory (ZFC).

In the next section we present CIC in a more formal way. In section 3 we give an interpretation of CIC in ZFC; these semantics generalize Coquand's proof-irrelevance semantics and are extremely straightforward. The number of inaccessible cardinals needed is exactly the number of universes of the modelized type theory. In section 4, we do the reverse work by encoding ZFC in CIC, adapting work of Aczel [1–3]. We however have to assume a type-theoretical axiom of choice in CIC to be able to encode full ZFC. We then can encode ZFC with n inaccessible cardinals in CIC equipped with $n+1$ universes. This gives lead to nicely interleaving relative consistency proofs. The situation is summed up in the conclusion.

The paper paper tries to be as self-contained as possible. For matters of space, we keep our description of inductive types informal, relying on examples and referring to the relevant publications. Some basic notions of λ-calculus and set theory are required. The encoding of set theory in CIC is not extensively described but has been formally checked on the Coq proof-assistant; the brave reader can check details looking at the proof-files [21].

2 Definition of the Type Theory

We give a presentation of the type theory in the style of Pure Type Systems (PTS, see [5]). We distinguish two parts: the rules dealing with the formation of function types, λ-abstraction and application on one hand, and inductive types on the other.

2.1 Function types

The *sorts* of the calculus are Prop and Type_i where i is a strictly positive integer; we will generally use s to denote them:

$$s \ := \ \text{Prop} \mid \text{Type}_i$$

We give ourselves a countable set of *variables*, generally denoted by x, y, z, a, b, c, X, etc.

The terms of the calculus are described by the following grammar:

$$t \ := \ x \mid (t\ t) \mid \lambda x : t.t \mid \Pi x : t.t$$

The first three cases correspond to the usual constructors for λ-terms: variable, application and typed abstraction. The term $\Pi x : A.B$ is the type of functions mapping terms of type A to type B. The value x of the argument might occur in B since the calculus allows dependent types. In the case where x does not occur free in B, $\Pi x : A.B$ can be written $A \rightarrow B$.

We write $t[x \setminus u]$ for the term t in which the free occurrences of the variable x are replaced by u. The substitution being defined the usual way. We write $\triangleright_\beta$ for the β-reduction which is defined as the contextual closure of:

$$(\lambda x : A.t\ u) \rightarrow_\beta t[x \setminus u]$$

The reflexive, transitive and symmetric closure of $\triangleright_\beta$ is called β-conversion, written $=_\beta$.

$$(\textsc{Prop})\ [] \vdash \mathrm{Prop} : \mathrm{Type}_1 \qquad (\textsc{Type}\ _i)\ [] \vdash \mathrm{Type}_i : \mathrm{Type}_{i+1}$$

$$(\textsc{Cum-Prop})\ \frac{\Gamma \vdash A : \mathrm{Prop}}{\Gamma \vdash A : \mathrm{Type}_i} \qquad (\textsc{Cum}\ _i)\ \frac{\Gamma \vdash A : \mathrm{Type}_i}{\Gamma \vdash A : \mathrm{Type}_{i+1}}$$

$$(\textsc{Pi-Prop}\ _j)\ \frac{\Gamma \vdash A : \mathrm{Prop} \qquad \Gamma; (x : A) \vdash B : \mathrm{Type}_j}{\Gamma \vdash \Pi x : A.B : \mathrm{Type}_j}$$

$$(\textsc{Pi}\ _{i,j})\ \frac{\Gamma \vdash A : \mathrm{Type}_i \qquad \Gamma; (x : A) \vdash B : \mathrm{Type}_j}{\Gamma \vdash \Pi x : A.B : \mathrm{Type}_{\max(i,j)}}$$

$$(\textsc{Impr})\ \frac{\Gamma \vdash A : \mathrm{Type}_i \qquad \Gamma \vdash P : \mathrm{Prop}}{\Gamma \vdash \Pi x : a.P : \mathrm{Prop}}$$

$$(\textsc{Lam})\ \frac{\Gamma; (x : A) \vdash t : B \qquad \Gamma \vdash \Pi x : A.B : s}{\Gamma \vdash \lambda x : A.t : \Pi x : A.B}$$

$$(\textsc{App})\ \frac{\Gamma \vdash u : \Pi x : A.B \qquad \Gamma \vdash v : A}{\Gamma \vdash (u\ v) : B[x \setminus v]}$$

$$(\textsc{Conv})\ \frac{\Gamma \vdash u : A \qquad \Gamma \vdash B : s \qquad A =_\beta B}{\Gamma \vdash u : B}$$

Fig. 1. Rules for the functional fragment of CIC

The typing rules of the system are given for figure 1. They correspond to Luo's Extended Calculus of Constructions[1] described in [16].

A well-formed type is a term of type some sort in a given context. The sorts Type_i are called the *universes*; they are hierarchally embedded in each other through the cumulativity rules $\textsc{Cum}_i$. Universal quantification is represented by the Π-types. An important point is that in Prop, we can quantify over all universes (impredicativity). In Type_i however, quantification is restricted to types living *below* universe i, i.e. over terms of type Type_j with $j \leq i$.

2.2 Inductive types

Inductive types are a very important extension to a type system meant to formalize actual mathematics. Among others, they are used to define data types and logical connectives and they play an essential role in the present work. Coquand and Paulin-Mohring [10] motivate this extension and give a formal description of generic inductive types. The Coq reference manual [6] contains a description of the currently implemented version of inductive definitions due to Paulin-Mohring.

[1] Luo's ECC also includes Σ-types which are a particular case of the inductive types described below.

Notations An inductive type is the smallest type closed by a list of constructors. The most well-known example is maybe the definition of unary integers:

$$\begin{aligned}\text{Inductive } \mathsf{Nat} : \text{Type}_1 := {} & \mathsf{O} : \mathsf{Nat} \\ \mid {} & \mathsf{S} : \mathsf{Nat} \to \mathsf{Nat}.\end{aligned}$$

Which defines three objects Nat, O and S of the given types. To each inductive type are associated elimination schemes corresponding to case analysis and structural recursion and induction. In this case the type of these schemes is:

$$\Pi P : \mathsf{Nat} \to s.(P\ \mathsf{O}) \to (\Pi n : \mathsf{Nat}.(P\ n) \to (P\ (\mathsf{S}\ n))) \to \Pi n : \mathsf{Nat}.(P\ n)$$

with s ranging over the sorts Prop and Type$_i$.

The reduction rules associated to these schemes are defined as usual, and in this case correspond to the recursor R of system T. In what follows however, we will generally not mention the elimination scheme explicitly and rather use a more explicit ML-like notation which, we believe is not ambiguous in the constructions treated below. For instance:

$$\begin{aligned}&\text{Definition } \mathit{add}\ :\ \mathsf{Nat} \to \mathsf{Nat} \to \mathsf{Nat}\ := \\ &\quad \mathsf{O}\ n\ \mapsto\ n \\ &\ \mid\ (\mathsf{S}\ p)\ n\ \mapsto\ (\mathsf{S}\ (\mathit{add}\ p\ n)).\end{aligned}$$

Of course, the corresponding new reduction rules are taken into account by the conversion rule Conv; for instance $(\mathit{add}\ (\mathsf{S}\ \mathsf{O})\ (\mathsf{S}\ \mathsf{O}))$ and $(\mathsf{S}\ (\mathsf{S}\ \mathsf{O}))$ are identified.

Restrictions Two main restrictions assure that inductive definitions do not endanger the consistency of the system. The first one is well-known and of syntactical nature: recursive arguments must occur in strictly positive position in the type of constructors; see [10] for details. Typicaly the following definition is prohibited since the first occurrence of foo in the type of C is negative:

$$\begin{aligned}&\text{Inductive } \mathsf{foo}\ :\ \text{Type}_i\ := \\ &\quad C : (\mathsf{foo} \to \mathsf{foo}) \to \mathsf{foo}.\end{aligned}$$

Semantically, the positivity condition ensures that the inductive definition corresponds to the least fix-point of a *monotone* operator.

The second restriction is more interesting and essential to what follows. If the inductive type is of type Type$_i$, then all arguments of its constructors must live in the same universe or lower, i.e. using the cumulativity rule, the arguments of the constructors must be themselves of type Type$_i$. Releasing this restriction would allow the construction of paradoxes and break the consistency of the system; see [8] for instance. The necessity for this restriction will also appear quite clearly in the proof-irrelevance interpretation and some of its consequences are shown in the next paragraph.

Two definitions of the existential An important point is that, in this paper, we do not use inductive definitions of type Prop. Some propositions certainly are defined inductively, but they can be constructed using an impredicative encoding. The key example is the existential quantification. Given $A : \mathrm{Type}_i$ and $P : A \to s$ we define the proposition $\exists a : A.(P\ a)$ as:

$$\text{Inductive } \exists a : A.(P\ a) : \mathrm{Prop} \;:=\; \quad \exists\_i : (a : A)(P\ a) \to \exists a : A.(P\ a).$$

But this inductive definition is exactly equivalent to:

$$\text{Definition } \exists a : A.(P\ a) := \Pi X : \mathrm{Prop}.(\Pi a : A.(P\ a) \to X) \to X.$$

The important point is that there are no elimination rules of inductive propositions towards Type$_i$.

There is however an alternative definition for the existential quantifier, namely the Σ-type living in Type_i:

$$\text{Inductive } \Sigma a : A.(P\ a) : \mathrm{Type}_i \;:=\; \quad \sigma : (a : A)(P\ a) \to \exists a : A.(P\ a).$$

Each of these two definitions has it advantages and drawbacks. In the first case, the existential lives at the bottom of the universe hierarchy, and thus we can always quantify over it, wherever we are. On the other hand, we cannot extract the *witness* of an existential proof: proving $\exists a : A.(P\ a)$ does not mean that we can exhibit a term of type A.

In the case of the Σ-type, we are able to define π such that $(\pi\ (\sigma\ a\ p)) \triangleright a$. But we cannot consider this Σ-type in types living lower in the hierarchy.

We can notice that if one does not use the impredicative level in CIC, the resulting theory is, in spirit, very close to the one of Martin-Löf [17]. The way we view the impredicative level in the present work is similar to Church's Higher-Order Logic: the objects live in the predicative levels, and Prop is used to express properties about them; we cannot however build objects out of proofs. This idea is obvious in the proof-irrelevance semantics.

3 Proof-Irrelevance Semantics for CIC

3.1 General Idea

In this section, we will write 0 for the empty set, and 1 for the canonical singleton $\{0\}$.

Since Reynolds [19], it is well-known that "polymorphism is not set-theoretic", i.e. it is not possible to interpret impredicative types (here of type Prop) by sets. This will however be possible for the predicative fragment of our theory, in which the objects live. The proof-irrelevance semantics are an interpretation of the type theory in *classical* set theory following two simple ideas:

– Propositions are interpreted either by the empty set or by a canonical singleton, depending upon their validity in the model. As a consequence all proof-terms are identified. In the same way, the interpretation of Prop is $\{0, 1\}$.

– On the predicative level, types are simply interpreted as sets. In particular the function type $\Pi x : A.B$ is interpreted by the full set of set-theoretic function between the interpretations of A and B.

The ideas of these interpretation are very common. They can be found, for instance, in [9,11,13].

We will call $\mathfrak{A}(i)$ the interpretation of Type_i. From the second assertion above and the typing rule $\text{PI}_{i,i}$ we need the following closure condition:

$$\forall \mathcal{A}, \mathcal{B} \in \mathfrak{A}(i). \mathcal{B}^{\mathcal{A}} \in \mathfrak{A}(i).$$

In particular this implies that $2^{\mathcal{A}} \in \mathfrak{A}(i)$, since $|\text{Prop}| = \{0, 1\} \in \mathfrak{A}(i)$. It is actually not too difficult to check that the existence of a set $\mathfrak{A}(i)$ verifying the (first) closure condition above implies the existence of an inaccessible cardinal.

3.2 Inaccessible cardinals

In what follows, we will assume the existence of an inaccessible cardinal to build the interpretation of each universe. The notions of set theory used are quite common (see [14,15] for instance).

Definition 1 (Inaccessible cardinal). An infinite cardinal λ is said to be *i-naccessible* if and only if:

– For any cardinal $\alpha < \lambda$, $2^\alpha < \lambda$.
– Let $(\beta_i)_{i \in I}$ be a family of cardinals $< \lambda$ indexed by a cardinal $I < \lambda$; then $\mathsf{sup}_{i \in i}(\beta_i) < \lambda$.

The main idea behind the notion of inaccessible cardinal is that its existence allows the construction of a set which is a itself a model of ZFC. It therefore enhances the expressive power of the theory. The following constructions are usual in the literature.

We write $\mathcal{P}$ for the powerset.

Definition 2. For every ordinal α, we define a set V_α by induction over α:

– $V_0 \equiv 0$
– $V_\alpha \equiv \bigcup_{\beta < \alpha} \mathcal{P}(V_\beta)$ if $\alpha > 0$.

The following result is a consequence of the foundation axiom and the proofs are well-known.

Definition 3 (Rank of a set). For every set X, there exists a smallest ordinal α such that $X \in V_\alpha$. α is called the *rank* of X, written $\text{rk}(X)$.

Lemma 4. *If λ is an inaccessible cardinal, then V_λ verifies the axioms of ZFC. In particular if $A \in V_\lambda$ and for every $a \in A$, $B_a \in V_\lambda$, then $\Pi_{a \in A} B_A \in V_\lambda$.*

3.3 The Interpretation

From here on, we assume the existence of an increasing sequence of inaccessible cardinals $(\lambda_i)_{i\in\mathbb{N}}$. For every i we define $\mathfrak{A}(i) \equiv V_i$.

We can now define the interpretation. The interpretation $|\Gamma|$ of a context Γ of length n is a set of n-tuples. The interpretation $|\Gamma \vdash t|$ of a judgement $\Gamma \vdash t : T$ does not depend upon T and is a function of domain $|\Gamma|$.

As often for similar constructions, [4,18] among others, we first define the interpretation as a partial function. The definition is a structural induction over the syntax; here, P exclusively denotes propositions (clauses 3 and 4):

$$|[]| \equiv 1 \tag{1}$$

$$|\Gamma; (x : A)| \equiv \{(\gamma, \alpha), \gamma \in |\Gamma| \wedge \alpha \in |\Gamma \vdash A|(\gamma)\} \tag{2}$$

$$|\Gamma \vdash p|(\gamma) \equiv 0 \qquad \text{if } p \text{ is a proof in } \Gamma \tag{3}$$

$$|\Gamma \vdash \Pi x : A.P|(\gamma) \equiv 1 \qquad \text{if } \forall \alpha \in |\Gamma \vdash A|(\gamma).|\Gamma; (x : A) \vdash P|(\gamma, \alpha) = 1 \tag{4}$$

$$|\Gamma \vdash \Pi x : A.P|(\gamma) \equiv 0 \qquad \text{if } \exists \alpha \in |\Gamma \vdash A|(\gamma).|\Gamma; (x : A) \vdash P|(\gamma, \alpha) = 0 \tag{5}$$

$$|\Gamma \vdash \Pi x : A.T|(\gamma) \equiv \Pi_{\alpha \in |\Gamma \vdash A|(\gamma)} |\Gamma; (x : A) \vdash T|(\gamma, \alpha) \tag{6}$$

$$|\Gamma \vdash \mathrm{Prop}|(\gamma) \equiv \{0, 1\} \tag{7}$$

$$|\Gamma \vdash \mathrm{Type}_i|(\gamma) \equiv \mathfrak{A}(i) \tag{8}$$

$$|\Gamma \vdash \lambda x : A.B|(\gamma) \equiv a \in |A|_{\mathcal{I}} \mapsto |B|_{\mathcal{I}; x \leftarrow a} \tag{9}$$

$$|\Gamma \vdash (u\ v)|(\gamma) \equiv |u|(\gamma)(|v|(\gamma)) \tag{10}$$

$$|\Gamma \vdash x|(\gamma) \equiv \gamma_i \tag{11}$$

Interpretation of inductive types The interpretation of each inductive type is defined inductively, in the set-theoretical sense. Again, we avoid detailing a tedious generic definitions and concentrate on an example; consider the definition of lists:

$$\begin{aligned} \text{Inductive } \mathsf{list} : \mathrm{Type}_i := {} & \mathsf{nil} : \mathsf{list} \\ \mid {} & \mathsf{cons} : A \to \mathsf{list} \to \mathsf{list}. \end{aligned}$$

where $A : \mathrm{Type}_i$. We encode the constructor using set-theoretical natural numbers 0,1,2.... The set $|\mathsf{list}|$ is (if it exists) the smallest subset of $\mathfrak{A}(i)$ verifying:

- $0 \in |\mathsf{list}|$
- if $a \in |A|$ and $l \in |\mathsf{list}|$, then $(1, a, l) \in |\mathsf{list}|$.

The two clauses correspond to the two constructors; the interpretation of the latter is natural:

- $|\mathsf{nil}| \equiv 0$
- $|(\mathsf{cons}\ a\ l)| \equiv (1, |a|, |l|)$ or, to be precise, the (curryfied) function which to $a \in |A|$ and $l \in |\mathsf{list}|$ associates $(1, a, l)$.

Note that we deliberately omit the interpretation of the context in this example, since they do not play a relevant role and would complicate notations. This interpretation generalizes smoothly to definitions with arbitrary many constructors of arbitrary arity[2].

The strict positivity condition assures that the inductive definition above corresponds to a monotone operator over sets. Since the arguments of the constructor are all of type Type_i, the soundness result below will ensure that their respective interpretations are elements of $\mathfrak{A}(i)$ and thus this monotone operator will actually admit a least fix-point in $\mathfrak{A}(i)$.

The structural ordering of the elements of the inductive type is reflected by a well-founded ordering of its interpretation. This gives rise to a natural interpretation of the elimination schemes we do not detail here.

Soundness results We show our interpretation is defined and sound on well-formed judgements. In order to treat the conversion rule, we first have to check that the interpretation is stable by reduction. For matters of space we do not detail the parts of the proof dealing with inductive types and the corresponding reductions.

Lemma 5 (Substitution). *Let $\Gamma; (x : A); \Delta \vdash t : T$ and $\Gamma \vdash a : A$ be two derivable judgements. If $\gamma \in |\Gamma|$, $\alpha \in |\Gamma \vdash A|(\gamma)$, $\alpha = |\Gamma \vdash a|(\gamma)$, $(\gamma, \alpha, \delta) \in |\Gamma; (x : A); \Delta|$ and $|\Gamma; (x : A); \Delta \vdash t|(\gamma, \alpha, \delta)$ is defined, then so is $|\Gamma; \Delta[x \setminus a] \vdash t[x \setminus a]|(\gamma, \delta)$ and*

$$|\Gamma; \Delta[x \setminus a] \vdash t[x \setminus a]|(\gamma, \delta) = |\Gamma; (x : A); \Delta \vdash t|(\gamma, \alpha, \delta).$$

By induction over the proof that $|\Gamma; (x : A); \Delta \vdash t|(\gamma, \alpha, \delta)$ is defined.

Lemma 6 (Subject reduction). *Let $\Gamma \vdash u : U$ be a derivable judgement. If $|\Gamma|$ is defined and $|\Gamma \vdash u|(\gamma)$ is defined for any γ in $|\Gamma|$, if $u \rhd_\beta u'$, then*

$$|\Gamma \vdash u'|(\gamma) = |\Gamma \vdash u|(\gamma)$$

and in particular, the left-hand part of the equation is defined.

By induction over the proof that $|\Gamma \vdash u|(\gamma)$ is defined (which follows the structure of u). The key case where u is the reduced redex is treated by the previous lemma.

Corollary 7. *Let $\Gamma \vdash u : U$ and $\Gamma \vdash u' : U'$ be two derivable judgements such that $|\Gamma \vdash u : U|(\gamma)$ and $|\Gamma \vdash u' : U|(\gamma)$ are defined for $\gamma \in |\Gamma|$. Then*

$$|\Gamma \vdash u'|(\gamma) = |\Gamma \vdash u|(\gamma).$$

Immediate, by the previous lemma, subject reduction and confluence of $\rhd_\beta$.

[2] Note however that we take advantage of the fact that every constructor awaits a fixed number of arguments, which is always trues in CIC.

Theorem 8 (Soundness). *Let $\Gamma \vdash t : T$ be a derivable judgement. Then $|\Gamma|$ is defined, and for any element γ of $|\Gamma|$:*

$$|\Gamma \vdash t|(\gamma) \in |\Gamma \vdash T|(\gamma)$$

in particular, both objects are defined.

By induction over the structure of the derivation. The previous corollary takes care of the conversion rule.

Corollary 9. *There is no derivation of $[] \vdash \Pi\alpha : Prop.\alpha$.*

Definition 10. A well-formed context Γ is said to be *consistent* if and only if $|\Gamma|$ is not empty.

Remark 11. If a context Γ is consistent, there is no derivation of $\Gamma \vdash \Pi\alpha :$ Prop.α.

Definition 12. We call *Type-theoretical Description Axiom* on level i (TTDA$_i$), the following proposition:

$$\begin{aligned}\Pi A, B : \mathrm{Type}_i.\Pi P : A \to B \to \mathrm{Prop}.(\Pi a : A.\exists b : B.(P\ a\ b)) \to \\ \exists f : A \to B.(\Pi a : A.(P\ a\ (f\ a)))\end{aligned}$$

Theorem 13. *The following context built up from instances of $TTDA_i$ and the excluded middle is consistent:*

$$(e : \Pi P : Prop.P \vee \neg P); (a_1 : TTDA_1); (a_2 : TTDA_2); \ldots ; (a_n : TTDA_n)$$

It is obvious that $|\Pi P : \mathrm{Prop}.P \vee \neg P| = 1$. One easily checks that $|\mathrm{TTDA}_i| = 1$ is equivalent to the usual set-theoretical axiom of choice; it is thus considered true, since we work in ZFC[3]. The following alternative type-theoretic formulation of the choice axiom will also be useful.

Definition 14. We call *Type-theoretical Choice Axiom* on the level i (TTCA$_i$), the following proposition:

$$\begin{aligned}\Pi A : \mathrm{Type}_i.\Pi R : A \to A \to \mathrm{Prop}.(equiv\ A\ R) \to \\ \exists f : A \to A.\Pi x, y : A.(R\ x\ y) \to (f\ x) =_A (f\ y)\end{aligned}$$

where $(equiv\ A\ R)$ expresses that R is an equivalence relation over A, namely:

$$\begin{aligned} equiv\ :=\ \lambda A : \mathrm{Type}_i.\lambda R : A \to A \to \mathrm{Prop}. &\\ &((x : A)(R\ x\ x)) \\ &\wedge ((x, y : A)(R\ x\ y) \to (R\ y\ x)) \\ &\wedge ((x, y, z : A)(R\ x\ y) \to (R\ y\ z) \to (R\ x\ z))\end{aligned}$$

[3] To be precise, the interpretation of TTDA$_i$ is equivalent to the axiom of choice restricted to the elements of $\mathfrak{A}(i)$; it is however an easy and usual result that the latter is a consequence of the general set-theoretical axiom of choice.

Again, one easily checks that $|TTCA_i| = 1$ is a consequence of the axiom of choice. We might thus conclude:

Theorem 15. *The following context built up from instances of $TTDA_i$, $TTCA_i$ and the excluded middle is consistent:*

$$\begin{aligned}(e : \Pi P : Prop.P \vee \neg P);\ &(a_1 : TTDA_1); (a_2 : TTDA_2); \ldots ; (a_n : TTDA_n);\\ &(a_1 : TTCA_1); (a_2 : TTCA_2); \ldots ; (a_n : TTCA_n.\end{aligned}$$

Let us write CIC$_i$ for the fragment of CIC where we only use universes up to Type$_i$, and ZFC$_i$ for ZFC equipped with i inaccessible cardinals. We can remark that we can build the interpretation for CIC$_i$ using only $i-1$ inaccessibles; which allows us to state a finer version of the theorem:

Theorem 16. *If ZFC$_{i-1}$ is consistent, then so is the following context of CIC$_i$:*

$$\begin{aligned}(e : \Pi P : Prop.P \vee \neg P);\ &(a_1 : TTDA_1); (a_2 : TTDA_2); \ldots ; (a_n : TTDA_n);\\ &(a_1 : TTCA_1); (a_2 : TTCA_2); \ldots ; (a_n : TTCA_n.\end{aligned}$$

3.4 Comment

The type-theoretical description axiom is valid in the model because the set-theoretical axiom of choice is valid in ZFC, but also, and mainly, because the function type $\Pi x : A.B$ is interpreted by the full space of set-theoretical functions. Actually we might view the hypothesis TTDA as a way to express, in the type theory, that the model is full. In the next section we develop this point by showing that TTDA is actually a sufficient constraint to force the model to be full, since adding TTDA to the type theory allows to encode full ZF.

It is certainly not obvious how to build a model for the Calculus of Inductive Constructions which would not be full (and not require the existence of inaccessibles). Actually, it is, to our knowledge, an open problem whether this is at all possible.

We should also say a word about the link with normalization proofs. Altenkirch [4] has presented a new technique in which proving normalization for a type theory essentially boils down to the construction of a certain kind of model. This idea has been used by Melliès and Werner in a normalization proof for Pure Type Systems [18]. Since, in the latter work, inductive types where not considered, it was possible to avoid using inaccessibles at the cost of a notable complication of the model construction. We mention this because such normalization proofs can be particularly well be built up along the proof-irrelevance interpretation. For matters of space, and since this is not the primary topic of the present paper, we do not deal further with normalization here.

4 Encoding ZFC in CIC

In this section, we present an adaptation of Peter Aczel's encoding of set theory in type theory. We have formalized and checked our version of the encoding using the Coq theorem prover[21].

Before describing the technical differences with Aczel's original work, we should mention the different motivations that drive us here. Aczel uses Martin-Löf predicative type theory; he wanted to demonstrate the pertinence this theory as a foundational formalism and was mainly interested in constructivity. It was therefore much more important to him to obtain a constructive type theory than to study the links with the usual classical (and impredicative and non-constructive) Zermelo-Frænkel set theory. Here, we are more primitively interested in "brute force" expressive power and impredicativity.

We parametrize our development by a universe index i. The reader might find more details in the Coq proof-file [21] and Aczel's original work [1–3].

4.1 The sets

Peter Aczel's encoding is a beautyful and very refined piece of type theory. The main idea is that sets can be build up inductively following the foundation axiom: the elements are structurally smaller than the set which contains them.

$$\begin{array}{l}\text{Inductive } \mathsf{Set} \ : \ \text{Type}_{i+1} \ := \\ \quad \mathsf{sup} : \Pi A : \text{Type}_i.(A \rightarrow \mathsf{Set}) \rightarrow \mathsf{Set}.\end{array}$$

Intuitively $(\mathsf{sup}\ A\ f)$ is the set whose elements are the objects of the form $(f\ a)$ where a ranges over the type A; mixing type and set theory notations we could write it $\{f(a), a : A\}$. Note that $(sup\ A\ f)$ contains at most as many elements as the type A (less if, for instance, f is a constant function).

A good first example is the construction of the pair-set, corresponding to the set-theoretical axiom of pairing. Since the set $\{E, E'\}$ has atmost two elements, the obvious choice is to use the booleans as base type:

$$\begin{array}{l}\text{Definition } \mathsf{Pair} \ : \ \mathsf{Set} \rightarrow \mathsf{Set} \rightarrow \mathsf{Set} \ := \\ \quad \mathsf{fun}\ E_1\ E_2 \mapsto (\mathsf{sup\ bool}\ (\mathsf{fun\ true} \mapsto E_1 \\ \qquad\qquad\qquad\qquad\qquad\qquad\quad \mid \mathsf{false} \mapsto E_2)).\end{array}$$

Another one is the empty set, which uses the empty type[4]:

$$\text{Definition } \mathsf{Empty} := (\mathsf{sup\ bot\ fun} : \mathsf{bot} \rightarrow \mathsf{Set}).$$

4.2 The propositions

Before we can prove the definitions above actually verify the corresponding set-theoretic axioms, we have to decide how to translate the propositions of set theory. Set theory is a first order theory with two (binary) predicates: membership and equality. One first defines equality, by structural recursion, in a way which captures the extentionality axiom:

$$\begin{array}{l}\text{Definition } \mathsf{Eq} \ : \ \mathsf{Set} \rightarrow \mathsf{Set} \rightarrow \text{Prop} \ := \\ \quad \mathsf{fun}\ (\mathsf{sup}\ A\ f)\ (\mathsf{sup}\ B\ g) \mapsto ((\Pi a : A.\exists b : B.(\mathsf{Eq}\ (f\ a)\ (g\ b))) \\ \qquad\qquad\qquad\qquad\qquad\qquad \wedge\ (\Pi b : B.\exists a : A.(\mathsf{Eq}\ (f\ a)\ (g\ b))))\end{array}$$

[4] The empty type bot is the inductive type with no constructor.

On top of this, one easily defines membership:

$$\begin{array}{l}\text{Definition } \mathsf{In} \ : \ \mathsf{Set} \rightarrow \mathsf{Set} \rightarrow \text{Prop} \ := \\ \quad \mathsf{fun}\ E\ (\mathsf{sup}\ A\ f) \mapsto \exists a : A.(\mathsf{Eq}\ E\ (f\ a)).\end{array}$$

It is in these two last definitions that we made a choice different from Aczel's: we chose to represent the propositions of set theory by objects of type Prop as opposed to Aczel who translates propositions to objects either of type Type_i or of type Type_{i+1}.

In any case, given the two definitions above, we can check the construction of the unordered pair actually is a witness of the corresponding axiom of Zermelo set theory by proving the three following lemmas:

$$\begin{array}{l}(A, B : \mathsf{Set})(\mathsf{In}\ A\ (\mathsf{Pair}\ A\ B) \\ (A, B : \mathsf{Set})(\mathsf{In}\ B\ (\mathsf{Pair}\ A\ B) \\ (A, B, C : \mathsf{Set})(\mathsf{In}\ C\ (\mathsf{Pair}\ A\ B) \rightarrow (\mathsf{Eq}\ A\ C) \vee (\mathsf{Eq}\ B\ C)\end{array}$$

Note that equality over sets is not represented by the usual Leibniz equality. Thus, we have to prove that all our encodings are extentional. For example:

$$\Pi A, A', B : \mathsf{Set}.(\mathsf{In}\ A\ B) \rightarrow (\mathsf{Eq}\ A\ A') \rightarrow (\mathsf{In}\ A'\ B).$$

4.3 Comparing the two approaches

From a constructive point of view, the main drawback of our encoding is that, since we cannot extract the existential witness of existential proofs, it is to prove $\exists X : \mathsf{Set}.(P\ X)$ and to actually exhibit a term E of type Set together with a proof of $(P\ E)$. A side effect is the difficulty we have proving the replacement schemata as described in section 4.5.

The advantage is that, as opposed to Aczel, we gain unbounded quantification (over all sets) thanks to impredicativity and avoid cumbersome distinctions between restricted and unrestricted formulas, leading to various formulations of the comprehension scheme.

We should however remark that in many cases, we can avoid relying on the usual replacement scheme, using the higher-order features of CIC instead.

4.4 The other computational constructions

The other constructions underlying the set theory Z, namely union, comprehension scheme and the powerset can then be defined without difficulty. Possible definitions are:

$$\begin{array}{l}\text{Definition } \mathsf{Power}\ :\ \mathsf{Set} \to \mathsf{Set} \to \mathsf{Set}\ := \\ \quad \mathsf{fun}\ E \mapsto (\mathsf{sup}\ (\mathsf{Set} \to \mathrm{Prop}) \\ \qquad\qquad\qquad \lambda P : \mathsf{Set} \to \mathrm{Prop}.(\mathsf{Compr}\ P\ E)).\end{array}$$

$$\begin{array}{l}\text{Definition } \mathsf{Union}\ :\ \mathsf{Set} \to \mathsf{Set}\ := \\ \quad \mathsf{fun}\ (\mathsf{sup}\ A\ f) \mapsto (\mathsf{sup}\ \ \Sigma a : A.(\pi_1\ (f\ a)) \\ \qquad\qquad\qquad\qquad\qquad \mathsf{fun}(\sigma\ a\ b) \mapsto (\pi_2\ (f\ a)\ b)).\end{array}$$

$$\begin{array}{l}\text{Definition } \mathsf{Comp}\ :\ \mathsf{Set} \to (\mathsf{Set} \to \mathrm{Prop}) \to \mathsf{Set}\ := \\ \quad \mathsf{fun}\ (\mathsf{sup}\ A\ f) \mapsto (\mathsf{sup}\ \ \Sigma a : A.(P\ (f\ a)) \\ \qquad\qquad\qquad\qquad\qquad \mathsf{fun}(\sigma\ a\ p) \mapsto (f\ a)).\end{array}$$

From there we can, for instance, define the intersection set in the usual way:

$$\begin{array}{l}\text{Definition } \mathsf{Inter}\ :\ \mathsf{Set} \to \mathsf{Set}\ := \\ \quad \mathsf{fun}\ E \mapsto (\mathsf{Comp}\ (\mathsf{Union}\ E)\ \lambda e : \mathsf{Set}.\Pi a : \mathsf{Set}.(\mathsf{In}\ a\ E) \to (\mathsf{In}\ e\ a)).\end{array}$$

Of course, one then has to check the usual properties for all these constructions.

A very nice construction is the set of natural numbers, corresponding to the axiom of infinity, obtained using the type of natural numbers:

$$\begin{array}{l}\text{Definition } \mathsf{enc}\ :\ \mathsf{Nat} \to \mathsf{Set}\ := \\ \qquad \mathsf{O} \mapsto \mathsf{Empty} \\ \quad (\mathsf{S}\ n) \mapsto (\mathsf{Union}\ (\mathsf{Pair}\ (\mathsf{enc}\ n)\ (\mathsf{Power}\ (\mathsf{enc}\ n)))).\end{array}$$

$$\text{Definition } \mathsf{NAT} := (\mathsf{sup}\ \mathsf{Nat}\ \mathsf{enc}).$$

All these definitions are basically Aczel's ones, except the powerset which obviously strongly relies on impredicativity. A first consequence is:

Theorem 17. *The set theory Z can be encoded in CCI$_2$+EM.*

4.5 Non-computational constructions: replacement and choice

The situation is more complicated regarding the replacement schemata and the (set theoretical) axiom of choice. Both axioms rely on assumptions of the form $\forall\mathcal{A}.\exists\mathcal{B}.\ldots$. Since we work with a non-computational existential quantifier, we have no chance to actually build a set out of this assumption. We can however *prove*, using TTDA, the set theoretical axiom. The proofs are quite straightforward but too long to be detailed here; we refer to the proof-file [21] for details.

Formulations of the replacement schemata The encoding of the following collection scheme can be proven in CIC, assuming TTDA. It is parametrized by a binary predicate P:

$$(\forall X\,.\,\exists Y\,.\,P(X,Y)) \Rightarrow \forall E\,.\,\exists A\,.\,(\forall x \in E\,.\,\exists y \in A\,.\,P(x,y)).$$

Furthermore, suppose P is a functional predicate, namely

$\forall x, y, y' \,.\, P(x,y) \wedge P(x,y') \Rightarrow y = y'$.

Then, assuming TTDA and the excluded middle, we can prove the usual replacement schemata:

$\forall X \,.\, \exists Y \,.\, \forall y \,.\, (y \in Y \iff \exists x \in X \,.\, P(x,y))$.

We can then state:

Theorem 18. *The set theory ZF can be encoded in CCI_2+EM+$TTDA_3$.*

A possible encoding of the axiom of choice Let us, for instance, consider the following formulation of the set-theoretical axiom of choice:

Let E be set such that:
- *all elements of E are non-empty (i.e. bear one element)*
- *the intersection of two elements of E is non-empty*

Then there exists a set X such that the intersection of X with any element of E bears exactly one element.

Viewing this statement in the encoding, we may consider E is of the form $(\mathsf{sup}\ A\ f)$. Using TTDA, we can prove the existence of a function g of type $A \to \mathsf{Set}$, such that for any inhabitant a of type A, we have $(\mathsf{In}\ (g\ a)\ (f\ a))$. Suppose however that a and b are two different inhabitants of A, such that $(\mathsf{Eq}\ (f\ a)\ (f\ b))$; we cannot conclude that $(\mathsf{Eq}\ (g\ a)\ (g\ b))$. The set $(\mathsf{sup}\ A\ g)$ is therefore not an adequate witness for the axiom of choice: its intersection with the element $(f\ a)$ of E might possibly bear several elements (here $(g\ a)$ and $(g\ b)$). We have to further assume TTCA, to deal correctly with this extentionality problem and prove the lemma above. The conclusion then is:

Theorem 19. *The set theory ZFC can be encoded in CCI_3 using the assumptions EM+$TTDA_3$ $TTCA_3$.*

4.6 Inaccessible Cardinals

Up to here, we have only used two universes Type_i and Type_{i+1}. As a consequence CIC_2 (with axioms) is sufficient to encode ZFC. We now show that we are able to build explicit encodings for inaccessible cardinals.

The idea is remarkably simple and builds on the only typing rule we have not used so far, namely cumulativity. We duplicate the whole encoding in a new universe; for simplicity of notations we suppose that i is at least 2 and redefine sets one level below:

$$\begin{array}{l}\text{Inductive } \mathsf{Set}' \ :\ \mathrm{Type}_i \ := \\ \quad \mathsf{sup}' : \Pi A : \mathrm{Type}_{i-1}.(A \to \mathsf{Set}') \to \mathsf{Set}'.\end{array}$$

The new type Set' then corresponds to $\mathfrak{A}(1)$, the set of "small sets" which is obviously closed for all set-theoretical axioms. Indeed we have an obvious injection from Set' to Set:

$$\begin{array}{l}\text{Definition } inj \ :\ \mathsf{Set}' \to \mathsf{Set} \ := \\ \quad \mathsf{fun}\ (\mathsf{sup}'\ A\ f) \mapsto (\mathsf{sup}\ A\ \lambda a : A.(inj\ (f\ a))).\end{array}$$

And we can actually define the big set of small sets:

$$\mathsf{Big} \;:=\; (\mathsf{sup}\ \mathsf{Set}'\ inj).$$

And, for instance, it is surprisingly simple to prove that Big is closed for the powerset:

$$\Pi E : \mathsf{Set}.(\mathsf{In}\ E\ \mathsf{Big}) \rightarrow (\mathsf{In}\ (\mathsf{Power}\ E)\ \mathsf{Big}).$$

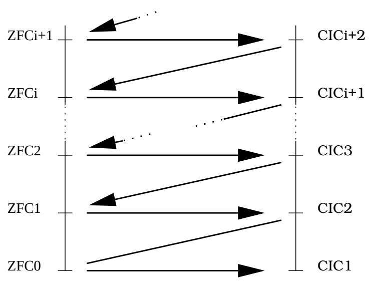
Of course this duplication of the encoding can be repeated several times using several universes.

At the current day, we have not explicitly proven in Coq the set Big implies the existence of an inaccessible cardinal. The main reason is that this would imply developing the whole theory of ordinals in ZFC encoded in Coq. Precisely, such a development is under way, along the lines of Paul Taylor's definition of constructive ordinals [20]. It seems however reasonable to anticipate the formal result of proving the existence of inaccessible ordinals in Coq, which is our final relative consistency result:

Theorem 20. *The set theory ZFC with n inaccessible cardinals can be encoded in*
CIC_{n+2}*+EM+*$TTDA_{n+2}$.

5 Conclusion

Building on ideas of Aczel, Coquand and others, we have presented two families of relative consistency proofs between ZFC and CIC, depending on how many inaccessible cardinals (resp. universes) we assume. This shows these two families have interleaving logical strengths; the situation is summed up in the following figure.



Both proofs are quite simple in spirit, and even the details are still much less cumbersome than what metatheoretical reasoning can often be.

Some work remains to be done. The formal proof of the existence of inaccessible cardinals of course, but also we believe the axioms used in coq in order to encode ZF and ZFC might be slightly simplified. To be precise, we conjecture:

1. When encoding ZF in CIC, it should be possible to rely on a weaker (but possibly more verbose) axiom than TTDA, whose justification unpleasantly relies on the set-theoretic axiom of choice.

2. Along the lines of the Diaconescu-Goodman-Myhill paradox, it should be possible to *prove* the excluded middle in CIC under the assumptions TTDA and TTCA.

More generally, it remains an open problem, up-to-where choice axioms are necessary for CIC to achieve the expressiveness of ZF or ZFC [5].

This work does not give birth to fundamental new concepts. We hope however it helps to bring various pieces of mathematics together. Especially we believe it sheds some new lights on the concept of type universes, which are often difficult to grasp.

Acknowledgements

I am grateful to Peter Aczel, Thierry Coquand and Samuel Lacas for helpful exchanges on the topic. The idea of investigating this topic was given to me by Christine Paulin-Mohring and Martin Hoffman.

References

1. P. Aczel. The Type Theoretic Interpretation of Constructive Set Theory. *Logic Colloquium 77*, Macintyre, Pacholsky and Paris (editors), Springer, 1977.
2. P. Aczel. The Type Theoretic Interpretation of Constructive Set Theory: Choice Principles. In *The L. E. J. Brouwer Centenary Symposium*, A. S. Troelstra and D. van Dalen (editors), North-Holland, 1982.
3. P. Aczel. The Type Theoretic Interpretation of Constructive Set Theory: Inductive Definitions. Proceedings of Methodology and Philosophy of Sciences, 1985.
4. T. Altenkirch. *Constructions, Inductive Types and Strong Normalization.* Ph.D. Thesis, University of Edinburgh, 1993.
5. H. Barendregt. Lambda Calculi with Types. In *Handbook of Logic in Computer Science*, Vol II, Elsevier, 1992
6. B. Barras et al. The Coq Proof Assistant Reference Manual, Version 6.1. INRIA Technical Report, 1996.
7. A. Church. A Formulation of the Simple Theory of Types. *Journal of Symbolic Logic*, 5 (1), pp. 56-68, 1940.
8. T. Coquand. An Analysis of Girard's Paradox. Proceeding of LICS, IEEE press, 1985.
9. T. Coquand. Metamathematical Investigations of a Calculus of Constructions. In P. Oddifredi (editor), *Logic and Computer Science.* Academic Press, 1990. Rapport de recherche INRIA 1088.
10. T. Coquand and C. Paulin-Mohring. Inductively defined types. In P. Martin-Löf et G. Mints (editors), *Proceedings of Colog'88.* Springer-Verlag, LNCS 417, 1990.
11. P. Dybjer. Inductive sets and families in Martin-Löf's type theory and their set-theoretic semantics. In G. Huet and G. Plotkin (editors), *Logical Frameworks*, Cambridge University Press, 1991.
12. J.-Y. Girard. *Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur*, Thèse d'Etat, Université Paris 7, 1972.

[5] Or alternatively whether one can exhibit a relative consistency proof of CIC+TTDA+TTCA in CIC.

13. D. Howe. On Computational Open-Endedness in Martin-Löf's Type Theory. Proceedings of LICS, 1991.
14. J.-L. Krivine. *Théorie Axiomatique des Ensembles*, Presses Universitaires de France, 1969.
15. K. Kunen. *Set Theory, An Introduction – Independence Proofs*, North-Holland, 1980.
16. Z. Luo. *An Extended Calculus of Constructions.* Ph.D. Thesis, University of Edinburgh, 1990.
17. P. Martin-Löf. *Intuitionistic Type Theory.* Studies in Proof Theory, Bibliopolis, 1984.
18. P.-A. Melliès and B. Werner. A Generic Normalization Proof for Pure Type Systems. Submitted and vailable on `http://pauillac.inria.fr/~werner/`, 1996.
19. J. C. Reynolds. Polymorphism is not Set-theoretic. Proceedings Int. Symp. on Semantics of Data Types, Sophia-Antipolis, LNCS 173, Springer, 1984.
20. P. Taylor. Intuitionistic Sets and Ordinals. *Journal of symbolic Logic*, 61(3), pp. 705-744, 1996.
21. B. Werner. An Encoding of ZFC Set Theory in Coq. Coq proof-file, available on `http://pauillac.inria.fr/ werner/` or as part of Coq distribution, 1997.