Reflections on Trusting Lean

Lorenzo Bresolin and Marcello Mamino

Univesità di Pisa

Abstract. It is known, by works of Werner and Carneiro, that the type theory of Lean (and indeed of **CIC** plus a version of choice) is equiconsistent with $\mathbf{ZFC}_\infty$, that is, **ZFC** + "there are infinitely many strongly inaccessible cardinals". We prove a finer connection between these two formal systems; namely, given any formula φ in the language of second order arithmetic, φ is provable in $\mathbf{ZFC}_\infty$ if and only if *the same* formula, when viewed as a type in Lean, is inhabited.

Keywords: Lean · Automatic Theorem Proving · Calculus of Constructions · Axiomatic Set Theory · Type Theory · Second Order Arithmetic

Introduction

Our work can be illustrated by a thought experiment. We will be dealing with the Lean 4 proof assistant (see [8] for context) but the basic concern and methodology are more general. Now, imagine a mathematician were presented with the construction, in Lean, of an inhabitant p of the propositional type

$$\begin{aligned}&\forall a : \mathtt{Nat}, \exists b : \mathtt{Nat}, \forall c : \mathtt{Nat}, \forall d : \mathtt{Nat},\\ &\quad \neg((c+2) * (d+2) = a + b) \ \wedge\\ &\quad \neg((c+2) * (d+2) = a + b + 2).\end{aligned}$$

Further imagine that our mathematician subscribes to the axioms of **ZFC** and that they are willing to accept the existence of infinitely many strongly inaccessible cardinals. Because of Werner's and Carneiro's work [10, 2], they accept that Lean is free of contradiction. Should they also be satisfied that the twin prime conjecture must be true? We claim yes.

Similarly to other popular proof assistants such as Rocq (formerly Coq), Lean is based on the Calculus of Inductive Constructions (**CIC**), see, for instance, [3] and [4]. We call $\mathbf{CIC}^+$ the variant of **CIC** implemented by Lean 4. This calculus extends **CIC** by the addition of a form of the axiom of choice, propositional extensionality, and proof irrelevance. It has been established by Werner [10] that **CIC** enriched with axioms for choice, and hence $\mathbf{CIC}^+$, proves the consistency of **ZFC** extended with the existence of infinitely many strongly inaccessible cardinals, which we will denote $\mathbf{ZFC}_\infty$. Vice-versa, Carneiro constructs in [2] a set-theoretic model of $\mathbf{CIC}^+$ assuming $\mathbf{ZFC}_\infty$. Thus $\mathbf{CIC}^+$ and $\mathbf{ZFC}_\infty$ are equiconsistent. However mere equiconsistency between two formal theories is in no way assurance enough that they prove the same theorems. For an example, **ZFC** and $\mathbf{ZFC} + \neg\mathrm{Con}(\mathbf{ZFC})$ are equiconsistent, but obviously do not prove the same formulas – and indeed, proving a statement in the latter would hardly be persuasive of that statement's veracity. We propose, here, a more fine grained result: namely that $\mathbf{CIC}^+$ and $\mathbf{ZFC}_\infty$ prove the same theorems in the language of second order arithmetic.

There is a further subtlety. Since the languages of $\mathbf{CIC}^+$ and $\mathbf{ZFC}_\infty$ differ, we should be precise about the meaning of proving *the same theorems*. To carry out our comparison, it is necessary to fix translations from second order arithmetic to set theory and types respectively. We feel that this is one important point that we raise, namely that there are *conventional* or *natural* translations: those that the users of the respective logical systems perceive as such. It is important, in our view, to prove the equivalence of, say, $\mathbf{CIC}^+$ and $\mathbf{ZFC}_\infty$, under the premise that mathematical statement are encoded according to the conventional translations. Given a sentence φ in the language of second order arithmetic, we define translations φ_Z and φ_C of φ in the languages of $\mathbf{ZFC}_\infty$ and $\mathbf{CIC}^+$ respectively. For $\mathbf{ZFC}_\infty$, the obvious choice is to map

the natural numbers to the finite von Neumann ordinals. For $\mathbf{CIC}^+$, using to the finite Von Neumann ordinals constructed inside Werner's model would be artificial; on the contrary, the conventional choice is to map the natural numbers to an appropriate inductive type `Nat`, which is the universally adopted convention in Lean.

Observe that the implementation of second order arithmetic in a given logical system can be nuanced, so it is not a choice that can be sensibly made in an arbitrary manner. For instance, inside Lean itself, it is not a given that the finite von Neumann ordinals in a model of **ZFC** can be seen as a model of second order arithmetic. In fact, Lean can construct a model of $\mathbf{ZFC} + \neg\mathrm{Con}(\mathbf{ZFC})$. This is, *a fortiori*, a model of **ZFC**, yet its finite von Neumann ordinals satisfy $\neg\mathrm{Con}(\mathbf{ZFC})$, as opposed to `Nat`.

With the notation established above, our main result is as follows:

> *Let φ be a sentence in the language of second order arithmetic. Then $\mathbf{ZFC}_\infty \vdash \varphi_Z$ if and only if $\mathbf{CIC}^+$ proves that the type φ_C is inhabited (i.e. $\mathbf{CIC}^+ \vdash e : \varphi_C$ for some expression e).*

The proof shows the following chains of implications

$$\begin{array}{ccccc} \mathbf{ZFC}_n \vdash \varphi_Z & \implies & \mathbf{CIC}^+ \vdash (\mathcal{M}_{\mathbf{ZFC}_n} \models \varphi_Z) & \implies & \mathbf{CIC}^+ \vdash \varphi_C \\ \mathbf{CIC}^+ \vdash \varphi_C & \implies & \mathbf{ZFC}_\infty \vdash (\mathcal{N}_{\mathbf{CIC}^+} \models \varphi_C) & \implies & \mathbf{ZFC}_\infty \vdash \varphi_Z \end{array}$$

where $\mathcal{M}_{\mathbf{ZFC}_n}$ is Werner's model of **ZFC** with n inaccessibles in **CIC**, and $\mathcal{N}_{\mathbf{CIC}^+}$ is Carneiro's set theoretic model of $\mathbf{CIC}^+$. The implication $\mathbf{CIC}^+ \vdash (\mathcal{M}_{\mathbf{ZFC}_n} \models \varphi_Z) \Rightarrow \mathbf{CIC}^+ \vdash \varphi_C$ has been implemented in Lean 4.

Our proof entails the feasibility of automatic translations from Lean 4 proofs of second order arithmetic statements to, for instance, Isabelle/ZF (and vice-versa). This is a nontrivial endeavor, as it requires to formalize Carneiro's work in Isabelle/ZF, thus obtaining a Lean 4 kernel which is formally verified in Isabelle/ZF. A similar work has been carried out by Krauss and Schropp [6], providing an automatic translator from Isabelle/HOL to Isabelle/ZF. If at all practical, the benefit of such a translator would be the ability to translate complex Lean 4 proofs to independently verifiable set theoretic ones.

1 Definition of $\mathbf{PA}^2$ and Translations

We will follow Carneiro's formalization of Lean, see [2], as essentially the type theory of **CIC** plus some additional axioms. To summarize, the grammar of $\mathbf{CIC}^+$ is as follows:

$$\begin{array}{rll} \ell = u \mid 0 \mid S\ell \mid \max(\ell,\ell) \mid \mathrm{imax}(\ell,\ell) & & \text{(universe level)} \\ \Gamma = () \mid \Gamma, x : e & & \text{(context)} \\ e = x \mid U_\ell \mid e\ \ e \mid \lambda x : e \mapsto e \mid \forall x : e, e \mid & & \\ \quad \mu x : F, K \mid \mathtt{c}_{\mu x:F,K} \mid \mathrm{rec}_{\mu x:F,K} & & \text{(expression)} \\ K = 0 \mid (\mathtt{c} : e) + K & & \\ F = U_\ell \mid \forall x : e, F & & \end{array}$$

U_ℓ are the universes; in particular U_0 is the sort of propositions. The usual definitions apply for $\alpha \to \beta$ and the inductive types $a = b$, $\alpha \times \beta$, and $\exists a : A, P\ a$. $\mathbf{CIC}^+$ is endowed with $(\beta), (\eta)$ reduction rules, as well as rules for recursors, and proof irrelevance. To this, we add the axioms of propositional extensionality and choice. We denote by $\mathbf{CIC}^+_n$ (for n natural) the fragment of **CIC** that only uses universes up to U_n.

Recall that the type $\mathtt{Nat} : U_1$ has an inductive definition given by

$$\mathtt{Nat} := \mu t : U_1, (\mathtt{Zero} :\ \ t) + (\mathtt{Succ} :\ \ t \to t)$$

and sum and product are defined via the recursor.

We define the language of second order arithmetic following [9]:

Definition 1. *A second order arithmetic formula is a formula of two-sorted first-order logic (where the sorts are denoted s_1, called the numbers, and s_2, called the subsets) in the language $\mathcal{L}_{\mathbf{PA}^2} = \{0, S, +, \times, \varepsilon\}$ (where $0 : s_1$, $S : s_1 \to s_1$, $+, \times : s_1 \times s_1 \to s_1$ and $\varepsilon \subset s_2 \times s_1$). (the intended meaning of $\varepsilon(P, t)$ is that t is an element of the set P, which we'll sometimes also write as $P(t)$ for brevity.)*

An arithmetic sentence is a second order arithmetic formula with no free variables. The set of arithmetic formulas is denoted $\mathcal{F}_{\mathbf{PA}^2}$.

We will usually indicate with $x_1, x_2, \ldots$ the number variables and with $P_1, P_2, \ldots$ the subset variables. We will use the symbols $\forall, \exists$ to quantify over numbers and $\forall^2, \exists^2$ to quantify over subsets. Without loss of generality, we may restrict the logical connectives to the complete set $\{\to, \bot\}$.

To illustrate the expressivity of second order arithmetic: the standard model, where the numbers are in $\mathbb{N}$ and the subsets range over $\wp(\mathbb{N})$, is bi-interpretable with the first order structure $(\mathbb{R}, +, \cdot)$ expanded by a predicate for $\mathbb{Z} \subset \mathbb{R}$. The sets that this structure can describe are precisely the projective sets of descriptive set theory [5, exercise 37.6]. Most subsets of $\mathbb{R}^n$ used in everyday mathematics, such as the graphs of continuous functions, are Borel sets, thus also projective sets. One limitation of this logic is that, although it can construct specific real functions, it does not allow quantification over the set of all functions $\mathbb{R} \to \mathbb{R}$.

Now, for any formula $\varphi \in \mathcal{F}_{\mathbf{PA}^2}$, we are going to define its set-theoretic interpretation $\varphi_Z \in \mathcal{F}_{\mathbf{ZFC}_\infty}$ as a formula in the language $\{\in\}$; and similarly, the type-theoretic interpretation of $\varphi \in \mathcal{F}_{\mathbf{PA}^2}$ will be an expression φ_C of $\mathbf{CIC}^+$. The type of φ_C will be the type $\mathscr{C} \to U_0$ of functions from an appropriate type of contexts to propositions. When φ is a sentence, φ_C will turn out to be a constant function, which can therefore be identified with a propositional type representing the sentence φ.

Definition 2 ($\mathbf{PA}^2 \rightsquigarrow \mathbf{ZFC}_\infty$). *Given an arithmetic formula $\varphi \in \mathcal{F}_{\mathbf{PA}^2}$ we let φ_Z be the customary translation of φ through the set-theoretical interpretation of second order arithmetic having sorts ω (the set of finite Von Neumann ordinals) and $\wp(\omega)$.*

The formal definition of this translation is straightforward yet rather long: it can be obtained by the implementation of `AriFormula2_to_ZFC'` in the Lean code [1].

Definition 3 ($\mathbf{PA}^2 \rightsquigarrow \mathbf{CIC}^+$). *The type $\mathscr{C}$ of contexts is* $(\texttt{Nat} \to \texttt{Nat}) \times (\texttt{Nat} \to \texttt{Set Nat})$. *Given an arithmetic formula $\varphi \in \mathcal{F}_{\mathbf{PA}^2}$, we define the function $\varphi_C : \mathscr{C} \to U_0$ by structural recursion on the formula φ. The sorts are* `Nat` *and* `Set Nat`*. Terms are mapped sending the symbols for 0, $+$ and $\times$ to the corresponding operators defined pointwise on $\mathscr{C} \to$* `Nat`*, and the variables to the projections $(x_i)_C \langle \mathcal{C}_1, \mathcal{C}_2 \rangle \equiv \mathcal{C}_1\, i$, $(P_j)_C \langle \mathcal{C}_1, \mathcal{C}_2 \rangle \equiv \mathcal{C}_2\, j$. Syntactic elements are mapped to the natural type theoretic constructions as follows:*

- $\bot_C \ \langle \mathcal{C}_1, \mathcal{C}_2 \rangle \equiv \bot$
- $(\varphi \to \delta)_C \ \langle \mathcal{C}_1, \mathcal{C}_2 \rangle \equiv (\varphi_C \ \langle \mathcal{C}_1, \mathcal{C}_2 \rangle) \to (\delta_C \ \langle \mathcal{C}_1, \mathcal{C}_2 \rangle)$;
- $(\exists x_i, \varphi)_C \ \langle \mathcal{C}_1, \mathcal{C}_2 \rangle \equiv \exists n : \texttt{Nat},\ \varphi_C \ \langle \mathcal{C}_1[n/i], \mathcal{C}_2 \rangle$;
- $(\forall x_i, \varphi)_C \ \langle \mathcal{C}_1, \mathcal{C}_2 \rangle \equiv \forall n : \texttt{Nat},\ \varphi_C \ \langle \mathcal{C}_1[n/i], \mathcal{C}_2 \rangle$;
- $(\exists^2 P_i, \varphi)_C \ \langle \mathcal{C}_1, \mathcal{C}_2 \rangle \equiv \exists Q : \texttt{Set Nat},\ \varphi_C \ \langle \mathcal{C}_1, \mathcal{C}_2[Q/i] \rangle$;
- $(\forall^2 P_i, \varphi)_C \ \langle \mathcal{C}_1, \mathcal{C}_2 \rangle \equiv \forall Q : \texttt{Set Nat},\ \varphi_C \ \langle \mathcal{C}_1, \mathcal{C}_2[Q/i] \rangle$;

where $\mathcal{C}_1[n/i]$ is the function that agrees with $\mathcal{C}_1$ except when the argument is i, and gives n when the argument is i, and similarly for $\mathcal{C}_2[Q/i]$.

When φ is a sentence, it easy to see that φ_C is necessarily a constant function. By abuse of notation we will denote value assumed by this function by the same symbol φ_C.

Remark 1. Notably, when φ is a sentence, a syntactic representation of the propositional type φ_C can be obtained by little more than just copying the formula. Indeed it is exactly because the translation is so trivial that we dare say that Lean 4, or $\mathbf{CIC}^+$, is designed to encourage its user to consider φ_C and φ, intuitively, the *same* formula.

Our main result is the following:

Theorem 1. *Let φ be an arithmetic sentence (i.e. a formula with no free variables). Then* $\mathbf{ZFC}_\infty$ *proves φ_Z if and only if φ_C is inhabited in* $\mathbf{CIC}^+$.

In the next sections we will outline the proof of each direction of the equivalence.

2 Every Arithmetic Formula Provable in $\mathbf{ZFC}_\infty$ Is Provable in $\mathbf{CIC}^+$

Theorem 1⇒. *Let φ be an arithmetic sentence, and assume that* $\mathbf{ZFC}_\infty$ *proves φ_Z. Then φ_C is inhabited in* $\mathbf{CIC}^+$.

By [10], $\mathbf{ZFC}_n$ can be coded into $\mathbf{CIC}^+_{n+2}$ as follows:

$$\mathtt{SET}_n := \mu t : U_{n+1}, (\mathtt{sup} : \forall A : U_n, (A \to t) \to t)$$

$$\begin{aligned}\mathtt{EXT}_n\ (\mathtt{sup}\ A\ f)\ (\mathtt{sup}\ B\ g) := \\ (\forall a : A, \exists b : B, \mathtt{EXT}_n\ (f\ a)\ (g\ b)) \ \wedge \\ (\forall b : B, \exists a : A, \mathtt{EXT}_n\ (f\ a)\ (g\ b))\end{aligned}$$

$$\mathtt{IN}_n\ x\ (\mathtt{sup}\ A\ f) := \exists a : A, \mathtt{EXT}_n\ x\ (f\ a).$$

Fact 1 (Werner, 1997 [10]). *Let $1 \leq n \in \mathbb{N}$. For each closed formula ψ in the language $\{\in\}$ of set theory, construct a propositional type $\psi_{\mathcal{M}}$ by interpreting the equality with $\mathtt{EXT}_n$, the inclusion with $\mathtt{IN}_n$ and by letting the quantifiers range over $\mathtt{SET}_n$. Then, whenever ψ is an axiom of* $\mathbf{ZFC}_{n-1}$, *one can construct in* $\mathbf{CIC}^+_{n+1}$ *an inhabitant of $\psi_{\mathcal{M}}$.*

Note that it is an immediate consequence of Fact 1 that $\mathbf{ZFC}_\infty \models \varphi_Z$ implies that, for n large enough, $(\varphi_Z)_{\mathcal{M}}$ is inhabited, where $\mathcal{M}$ is the model of $\mathbf{ZFC}_n$ in $\mathbf{CIC}^+_{n+2}$ given by Fact 1. As such, to show Theorem 2 it is sufficient to exhibit an inhabitant of the type $(\varphi_Z)_{\mathcal{M}} \to \varphi_C$.

We indeed show slightly more:

Lemma 1. *Let $n \geq 1$ be arbitrary, and $\mathtt{SET} := \mathtt{SET}_n$. There are functions $\mathtt{num}:\mathtt{Nat} \to \mathtt{SET} \to U_0$ and $\mathtt{sub}:\mathtt{Set\ Nat} \to \mathtt{SET} \to U_0$ so that the following holds. For any arithmetic formula φ, let x_i denote the free number variables of φ and P_j the free subset variables of φ (for i, j ranging over some finite subsets of the integers), and let $(n_i : \mathtt{Nat})$, $(Q_j : \mathtt{Set\ Nat})$, $(y_i : \mathtt{SET})$, $(z_j : \mathtt{SET})$ be s.t.*

(i) $\mathtt{num}\ n_i\ y_i$ *is inhabited for each i*
(ii) $\mathtt{sub}\ Q_j\ z_j$ *is inhabited for each j.*

Then the following propositional type is inhabited:

$$\varphi_C[n_i/x_i \ldots, Q_j/P_j \ldots] \leftrightarrow (\varphi_Z)_{\mathcal{M}}[y_i/x_i \ldots, z_j/P_j \ldots].$$

We provide a Lean 4 formalization of this lemma and of Theorem 1⇒ [1]; references to the Lean code will be provided in footnotes.

The proof[1] is carried out by structural recursion on φ; the general sketch is as follows:

(i) Construct $\mathtt{num}$ by structural recursion on its first argument, so that intuitively $\mathtt{num}\ n\ x$ is true when "x is the n-th finite ordinal" is true in $\mathcal{M}$; and similarly produce $\mathtt{sub}$ s.t. $\mathtt{sub}\ P\ x$ is true when $(x \subset \omega)_{\mathcal{M}}$ and, for any n, "x contains the n-th finite ordinal"$_{\mathcal{M}}$ iff $P\ n$. Then, one verifies that $\mathtt{num}$, $\mathtt{sub}$ are the graphs of bijective maps which are well-behaved w.r.t. the operations of arithmetic, hence that $\mathtt{Nat}$ is isomorphic to $(x : \mathtt{SET}) \times (x \in \omega)_{\mathcal{M}}$, and that $\mathtt{Set\ Nat}$ is isomorphic to $(x : \mathtt{SET}) \times (x \subset \omega)_{\mathcal{M}}$.[2]

[1] In the code, `ZFC_iff_Lean'`.

[2] In the code, those two relations are provided by `is_number` and `is_real` resp. The fact that those maps are bijections that preserve the structure is provided by `isoN` and the theorems from `plus_is_sum` to `succ_is_succ`.

(ii) Construct an inductive type `Form` of arithmetic formulas (with variables indexed with `Nat`), and associate a $\bar{\varphi} : \texttt{Form}$ to each formula φ.[3]

(iii) Construct valuations $\texttt{val}_C : (\texttt{Nat} \to \texttt{Nat}) \to (\texttt{Nat} \to \texttt{Set Nat}) \to \texttt{Form} \to U_0$ that, given an $f : \texttt{Nat} \to \texttt{Nat}$ valuation of the number variables, $g : \texttt{Nat} \to \texttt{Set Nat}$ valuation of the subset variables and $\bar{\varphi} : \texttt{Form}$, outputs a true proposition iff $\varphi_C[(f\ i)/x_i \ldots, (g\ j)/P_j \ldots]$ is true; and similarly $\texttt{val}_Z : (f : \texttt{Nat} \to \texttt{SET}) \to (g : \texttt{Nat} \to \texttt{SET}) \to (\bar{\varphi} : \texttt{Form})$ that outputs a true proposition iff $(\varphi_Z)_{\mathcal{M}}[(f\ i)/x_i \ldots, (g\ j)/P_j \ldots]$ is true.[4]

(iv) Finally check that $(\texttt{val}_C\ f\ g\ \bar{\varphi}) \leftrightarrow (\texttt{val}_Z\ f'\ g'\ \bar{\varphi})$ is inhabited as long as $\texttt{num}\ (f\ i)\ (f'\ i)$ whenever $x_i \in FV(\varphi)$ and $\texttt{sub}\ (g\ j)\ (g'\ j)$ whenever $P_j \in FV(\varphi)$, by structural recursion on $\bar{\varphi}$.

In the end, we will have constructed an inhabitant `C_iff_Z` of the propositional type

$$\begin{aligned}
&\forall \bar{\varphi} : \texttt{Form}, \forall f : \texttt{Nat} \to \texttt{Nat},\ \forall g : \texttt{Nat} \to \texttt{Set Nat},\\
&\qquad \forall f' : \texttt{Nat} \to \texttt{SET},\ \forall g' : \texttt{Nat} \to \texttt{SET},\\
&\qquad\quad (h : \forall n : \texttt{Nat}, \text{“}x_n \text{ is free in } \varphi\text{”} \to \texttt{num}\ (f\ n)\ (f'\ n)) \to\\
&\qquad\quad (h' : \forall n : \texttt{Nat}, \text{“}P_n \text{ is free in } \varphi\text{”} \to \texttt{sub}\ (g\ n)\ (g'\ n)) \to\\
&\qquad\qquad \big((\texttt{val}_C\ f\ g\ \bar{\varphi}) \leftrightarrow (\texttt{val}_Z\ f'\ g'\ \bar{\varphi})\big)
\end{aligned}$$

Proof (Theorem 1⇒). [5] Let φ be an arithmetic sentence, and assume $\mathbf{ZFC}_\infty \models \varphi_Z$. From Fact 1, then $(\varphi_Z)_{\mathcal{M}}$ is inhabited, let $\alpha : (\varphi_Z)_{\mathcal{M}}$. Applying `C_iff_Z` $\bar{\varphi}$ to arbitrary contexts we obtain an inhabitant of $\varphi_C \leftrightarrow (\varphi_Z)_{\mathcal{M}}$. We conclude, applying the $\leftarrow$ arrow of this to α. □

3 Every Arithmetic Formula Provable by Lean is Provable in $\mathbf{ZFC}_\infty$

Theorem 1⇐. *Let φ be an arithmetic formula with no free variables, and assume that φ_C is inhabited in $\mathbf{CIC}^+$. Then, $\mathbf{ZFC}_\infty$ proves φ_Z.*

To prove that, we will follow the following scheme:

(i) Following Carneiro [2], for each φ one can construct, assuming the existence of n strongly inaccessible cardinals for some sufficiently large n (which may, of course, depend on φ) a set $[\![\varphi]\!]$ which $\mathbf{ZFC}_n$ proves to be nonempty precisely if φ_C is inhabited.

(ii) We then need to show that

$$\mathbf{ZFC}_n \vdash ([\![\varphi]\!] \neq \varnothing) \longrightarrow \varphi_Z.$$

From that, we can conclude that $\mathbf{ZFC}_n \vdash \varphi_Z$ whenever φ_C is inhabited.

Carneiro describes, in [2, section 5], a model of the fragment $\mathbf{CIC}^+_W$ of $\mathbf{CIC}^+$ that uses only the W-type as a general inductive type, plus a handful of simple constructors. W-types are those produced by the W constructor, as defined in the following definition:

Definition 4. *(Martin-Löf 1982, [7]) Let $\Gamma \vdash a : U_\ell$ and $\Gamma,\ a : U_\ell \vdash b : U'_\ell$; then*

$$Wx : a, b\ := \mu t : U_{\max(\ell,\ell',1)},\ (\texttt{sup:}\ a \to (b \to t) \to t).$$

Then, [2] shows a reduction of the type theory of Lean onto this fragment. Technically, this means that we need to work with the model of $\mathbf{CIC}^+_W$ directly.

Let us begin with the definition of $\mathbf{N}$ obtained as the special case of the translation described in [2, section 5.2] applied to the inductive type `Nat`.

$$\mathbf{N} : U_1\ := Wx : \top \oplus \top, \texttt{rec}_\oplus\ (\lambda\_ : \top\ \mapsto \bot)\ (\lambda\_ : \top\ \mapsto \top)\ x.$$

[3] In the code, this inductive type is `AriFormula2`, which uses the inductive types `AriTerm` and `AriTerm2` to represent terms of first and second sort resp.

[4] In the code, those maps are `AriFormula2_to_Lean'` and `AriFormula2_to_ZFC'` resp.

[5] In the code, this is provided by `ZFC_iff_Lean`.

One can compute the $\mathbf{CIC}^+_W$ terms $0 : \mathbf{N}$, $S : \mathbf{N} \to \mathbf{N}$, $+ : \mathbf{N} \to \mathbf{N} \to \mathbf{N}$, $\times : \mathbf{N} \to \mathbf{N} \to \mathbf{N}$ which correspond to the usual operators on `Nat` via the reduction to $\mathbf{CIC}^+_W$. To any arithmetic formula φ we associate $\varphi_W : \mathscr{C} \to U_0$ in the in the same manner as φ_C, except by quantifying over $\mathbf{N}$ instead of `Nat`.

What we gain from those contortions is that now φ_W is a type of the fragment $\mathbf{CIC}^+_W$; so we would like to use it in place of φ_C. We can do this as long as $\varphi_C \leftrightarrow \varphi_W$ is inhabited; for then φ_W would be inhabited iff φ_C is.

The following lemma is proved by a straightforward structural induction:

Lemma 2. *There are bijections a : `Nat` $\to \mathbf{N}$ and A : (`Set Nat`) $\to$ (`Set` $\mathbf{N}$) s.t. the following holds. Whenever φ is an arithmetic formula with free number variables x_i and free subset variables P_j, and n_i : `Nat`..., Q_j : `Set Nat`..., then $\varphi_C[n_i/x_i \dots, Q_j/P_j \dots] \leftrightarrow \varphi_W[(a\, n_i)/x_i \dots, (A\, Q_j)/P_j \dots]$ is inhabited.*

In particular, if φ is an arithmetic sentence, then the propositional type $\varphi_C \leftrightarrow \varphi_W$ is inhabited.

Given a (strictly) increasing sequence of cardinals $(\gamma_n)_{n\in\omega}$, we say that it is *k-correct* if γ_i is inaccessible for all $0 \le i < k$.

Fact 2 (Carneiro [2], 2019). *Let $(\kappa_n)_{n\in\omega}$ be an increasing succession of strong limit cardinals, and let $U_0 = \{\varnothing, \{\bullet\}\}$, $U_{n+1} = V_{\kappa_{n+1}}$, $U_\omega = \bigcup_n U_n$ (where $\bullet = \varnothing$ is the proof object). Then there exist maps (where $|\Gamma|$ denotes the length of a given context Γ)*

$$\begin{aligned} \Gamma \text{ well-formed context} \quad &\mapsto \quad [\![\Gamma]\!] \subset U_\omega^{|\Gamma|} \\ \big(\Gamma \text{ well-formed context}, \gamma \in [\![\Gamma]\!], \\ \quad e \text{ s.t. } \Gamma \vdash e : \alpha \text{ for some } \alpha\big) \quad &\mapsto \quad [\![\Gamma \vdash e]\!]_\gamma \in U_\omega \end{aligned}$$

(where e ranges over the expressions of $\mathbf{CIC}^+_W$) so that:

i. if $\Gamma \vdash \alpha : U_0$, then $[\![\Gamma \vdash \alpha]\!]_\gamma \subseteq \{\bullet\}$
ii. if $\Gamma \vdash e : \alpha$, and if κ_n is k-correct for a sufficiently large k, then for all $\gamma \in [\![\Gamma]\!]$, $[\![\Gamma \vdash e]\!]_\gamma \in [\![\Gamma \vdash \alpha]\!]_\gamma$
iii. if $\Gamma \vdash e \equiv e'$, and if κ_n is k-correct for a sufficiently large k, then for all $\gamma \in [\![\Gamma]\!]$, $[\![\Gamma \vdash e]\!]_\gamma = [\![\Gamma \vdash e']\!]_\gamma$.

In the proof of Fact 2 in [2], $[\![\Gamma]\!], [\![\Gamma \vdash e]\!]_\gamma$ are defined by mutual recursion. The intended meaning is that $[\![\Gamma \vdash e]\!]_\gamma$ is the set that represents e whenever the variables that appear free in it (which are typed in the context Γ) are interpreted in accordance with γ; and $[\![\Gamma]\!]$ is the set of vectors of the possible set-theoretical interpretations of the variables it types.

In the rest of this section, we will use the particular definition of $[\![\cdot]\!]$ given in the proof of Fact 2.

In particular, by explicitly doing the calculations, one finds that

Lemma 3. *There is a countable set $\tilde{\omega}$ s.t., for any well-formed context Γ and $\gamma \in [\![\Gamma]\!]$, $[\![\Gamma \vdash \mathbf{N}]\!]_\gamma = \tilde{\omega}$. Furthermore, there is a bijection $\omega \to \tilde{\omega}$, $n \mapsto \overline{n}$ s.t.:*

- $[\![\Gamma \vdash 0]\!]_\gamma = \overline{0}$.
- $\tilde{S} = [\![\Gamma \vdash S]\!]_\gamma$ *is the map $\tilde{\omega} \to \tilde{\omega}$ given by $\tilde{S}(\overline{n}) = \overline{n+1}$.*
- $\tilde{+} = [\![\Gamma \vdash +]\!]_\gamma$ *is the map $\tilde{\omega} \to \tilde{\omega}^{\tilde{\omega}}$ given by $\tilde{+}(\overline{n})(\overline{m}) = \overline{n+m}$.*
- $\tilde{\times} = [\![\Gamma \vdash \times]\!]_\gamma$ *is the map $\tilde{\omega} \to \tilde{\omega}^{\tilde{\omega}}$ given by $\tilde{\times}(\overline{n})(\overline{m}) = \overline{nm}$.*

We can now undertake the step (ii) of the proof:

Lemma 4. *Let φ be an arithmetic formula with free number variables in $(x_i)_{i=1\dots p}$ and free subset variables in $(P_j)_{j=1\dots q}$; let moreover $n_1, \dots, n_p \in \omega$, $A_1, \dots, A_q \in \wp(\omega)$; then, if κ_n is k-correct for a sufficiently large k,*

$$\bullet \in [\![x_i : \mathbf{N} \dots, P_j : \texttt{Set } \mathbf{N} \dots \vdash \varphi_W]\!]_{\overline{n_i}\dots, \overline{A_j}\dots} \iff \varphi_Z[n_i/x_i \dots, A_j/P_j \dots]$$

The proof is by structural recursion. The base step follows from showing that, for any term t, one has $[\![x_i : \mathbf{N} \ldots \vdash t_W]\!]_{\overline{n_i}\ldots} = \overline{t_Z[n_i/x_i \ldots]}$ which follows using the fact that $\tilde{\omega}$ is "well-behaved" with respect to the arithmetic operations.

Proof (Theorem 1⇐). Let φ be an arithmetic formula with no free variables, and assume that φ_C is inhabited by $e : \varphi_C$. Now, *working in* $\mathbf{ZFC}_\infty$, we set out to prove φ_Z.

Using the Lemma 2, we can pick an $\vdash e' : \varphi_W$. In order to use the map $[\![\cdot]\!]$, let v be an universe valuation that sets all universe variables to 0, and $\vdash \langle e' \rangle_v : \varphi_W$ (see the theorems 6.2 and 6.3 in [2]). Now assume that κ_n is k-correct for a sufficiently large k as to satisfy the hypothesis of Fact 2. Let $p = [\![\vdash \langle e' \rangle_v]\!]_{()}$, and by Fact 2,(ii) one has that $p \in [\![\vdash \varphi_W]\!]_{()}$; but, by (i), $[\![\vdash \varphi_W]\!]_{()} \subseteq \{\bullet\}$, hence $p = \bullet$, and by the Lemma 4 then φ_Z is true.

We have now proven that $\mathbf{ZFC}_\infty$ shows φ_Z if φ_C is inhabited. □

References

1. Bresolin, L.: (2026), https://github.com/lrzbrsln/LeanPA
2. Carneiro, M.: The Type Theory of Lean (2019), https://github.com/digama0/lean-type-theory/releases/tag/v1.0
3. Coquand, T., Huet, G.: The calculus of constructions. Information and Computation **76**(2), 95–120 (1988)
4. Coquand, T., Paulin, C.: Inductively defined types. In: Martin-Löf, P., Mints, G. (eds.) COLOG-88. pp. 50–66. Springer Berlin Heidelberg (1990)
5. Kechris, A.: Classical Descriptive Set Theory. Graduate texts in mathematics, Springer-Verlag (1995)
6. Krauss, A., Schropp, A.: A mechanized translation from higher-order logic to set theory. In: Kaufmann, M., Paulson, L.C. (eds.) Interactive Theorem Proving. pp. 323–338. Springer Berlin Heidelberg (2010)
7. Martin-Löf, P.: Constructive Mathematics and Computer Programming. In: Studies in Logic and the Foundations of Mathematics. p. 153–175. Elsevier (1982)
8. Moura, L.d., Ullrich, S.: The Lean 4 Theorem Prover and Programming Language. In: Platzer, A., Sutcliffe, G. (eds.) Automated Deduction – CADE 28. pp. 625–635. Springer International Publishing (2021)
9. Simpson, S.G.: Subsystems of Second Order Arithmetic. Perspectives in Logic, Cambridge University Press, 2 edn. (2009)
10. Werner, B.: Sets in types, types in sets. In: Theoretical Aspects of Computer Software. p. 530–546. Springer Berlin Heidelberg (1997)