A Strongly Normalising Curry-Howard Correspondence for IZF Set Theory

Alexandre Miquel

Laboratoire de Recherche en Informatique
Université Paris-Sud, 91405 Orsay Cedex, France
Alexandre.Miquel@lri.fr

Abstract. We propose a method for realising the proofs of Intuitionistic Zermelo-Fraenkel set theory (IZF) by strongly normalising λ-terms. This method relies on the introduction of a Curry-style type theory extended with specific subtyping principles, which is then used as a low-level language to interpret IZF via a representation of sets as pointed graphs inspired by Aczel's hyperset theory.

As a consequence, we refine a classical result of Myhill and Friedman by showing how a strongly normalising λ-term that computes a function of type $\mathbb{N} \to \mathbb{N}$ can be extracted from the proof of its existence in IZF.

1 Introduction

In this paper, we revisit the work of Myhill [12], Friedman [4], McCarty [9] and Krivine [6, 7] related to the study of the computational contents of the proofs of set theory. Unlike the former approaches that are based on variants of Kleene's realisability, we propose a different framework that combines ideas coming from hyperset theory to the theory of type systems [8, 2, 5].

Technically, our work relies on the introduction of a type system which is used as an assembly language to interpret the axioms of intuitionisitic set theory in terms of pointed graphs. Conceptually, this translation is the natural reformulation of Aczel's model of hyperset theory [1] (originally achieved in set theory) in a type-theoretical framework. Of course, building this model in type theory allows us to benefit from the representation of proofs as λ-terms. Moreover, since the type system we are using enjoys the strong normalisation property, any proof of IZF is thus realised by a strongly normalising λ-term via our translation.

Apart from the strong normalisation property for IZF—which is the main contribution of the paper—there are several interests in using a type-theoretical framework as an intermediate language for studying set theory.

On the practical side, one may benefit from the use of many proof-assistants based on type theory to build the corresponding λ-terms explicitly: since the proofs of the axioms of IZF tend to be quite large, writing them down would be almost impossible by hand. By using the method described in this paper, the

author could explicitely construct (and check) all the proof-terms of IZF axioms with the help of the Coq proof-assistant [3].[1]

On a more theoretical side, this decomposition gives new insights about the relationship between set theories and type theories. As far as we know, there is currently no type system whose proof theoretical strength reaches the one of ZF. Despite its non-standard subtyping rules, the extended type system we will present in section 5 deserves the credit of enjoying this property.

2 The Logical Framework

In this section, we introduce the core part of the logical framework we will use in this paper. The initial system, called $F\omega.2$, will be first extended in paragraph 3.2 and then in section 5 to reach the proof theoretical strength of IZF.

2.1 System $F\omega.2$

System $F\omega.2$—or *system $F\omega$ with one universe*[2]—is organised in two syntactic categories: a syntactic category of *object-terms* in order to represent mathematical objects, and a syntactic category of *proof-terms* in order to represent mathematical proofs.

Object terms actually form an autonomous type system (the 'higher part' of system $F\omega.2$) which is completely independent from proof-terms. (Unlike the calculus of constructions, system $F\omega.2$ has no proof-dependent types.) This type system can be seen as an extension of Martin-Löf's logical framework

Object terms

$$\begin{array}{rcl} M,N,T,U,A,B & ::= & \mathsf{Type}_1 \mid \mathsf{Type}_2 \mid \Pi x:T\,.\,U \\ & \mid & x \mid \lambda x:T\,.\,U \mid M(N) \\ & \mid & \mathsf{Prop} \mid A \Rightarrow B \\ & \mid & \forall x:T\,.\,B \mid \exists x:T\,.\,B \end{array}$$

with a primitive type Prop of propositions (below the first universe Type_1), plus extra constructions to represent implication and universal/existential quantification. In this presentation, the letters M, N denote arbitrary object terms, whereas the letters T, U are reserved for types (i.e. the terms of type Type_1 or Type_2) and the letters A, B for propositions (i.e. the terms of type Prop).

As in Martin-Löf's logical framework [8], we make a distinction between a universe Type_1 of *small data-types* and a (top) universe Type_2 of *large data-types*. For convenience, we also consider a cumulativity rule $\mathsf{Type}_1 \subset \mathsf{Type}_2$.[3] Notice that at the level of object terms, Prop is not a sort, and propositions are not

[1] The corresponding proof-scripts can be downloaded on the author's web page at `http://pauillac.inria.fr/~miquel`.

[2] "$F\omega.2$" means: "system F with higher-order twice".

[3] To preserve simplicity, we do not address here the problem of propagating cumulativity through dependent products.

Formation of signatures: $\Sigma \vdash$

$$\frac{}{[] \vdash} \qquad \frac{\Sigma \vdash T : \mathsf{Type}_i}{\Sigma; [x : T] \vdash} \; {\scriptstyle x \notin DV(\Sigma)}$$

Typing rules of object terms: $\Sigma \vdash M : T$

$$\frac{\Sigma \vdash}{\Sigma \vdash \mathsf{Prop} : \mathsf{Type}_1} \qquad \frac{\Sigma \vdash}{\Sigma \vdash \mathsf{Type}_1 : \mathsf{Type}_2} \qquad \frac{\Sigma \vdash M : \mathsf{Type}_1}{\Sigma \vdash M : \mathsf{Type}_2}$$

$$\frac{\Sigma \vdash}{\Sigma \vdash x : T} \; {\scriptstyle (x:T) \in \Sigma} \qquad \frac{\Sigma \vdash T : \mathsf{Type}_i \qquad \Sigma; [x : T] \vdash U : \mathsf{Type}_i}{\Sigma \vdash \Pi x : T \,.\, U : \mathsf{Type}_i} \; {\scriptstyle i \in \{1;2\}}$$

$$\frac{\Sigma; [x : T] \vdash M : U}{\Sigma \vdash \lambda x : T \,.\, M : \Pi x : T \,.\, U} \qquad \frac{\Sigma \vdash M : \Pi x : T \,.\, U \qquad \Sigma \vdash N : T}{\Sigma \vdash M(N) : U\{x := N\}}$$

$$\frac{\Sigma \vdash A : \mathsf{Prop} \qquad \Sigma \vdash B : \mathsf{Prop}}{\Sigma \vdash A \Rightarrow B : \mathsf{Prop}} \qquad \frac{\Sigma; [x : T] \vdash B : \mathsf{Prop}}{\Sigma \vdash Q\, x : T \,.\, B : \mathsf{Prop}} \; {\scriptstyle Q \in \{\forall;\exists\}}$$

$$\frac{\Sigma \vdash M : T}{\Sigma \vdash M : T'} \; {\scriptstyle T =_\beta T'}$$

Formation of logical contexts: $\Sigma \vdash \Gamma \; \mathsf{ctx}$

$$\frac{\Sigma \vdash}{\Sigma \vdash [] \; \mathsf{ctx}} \qquad \frac{\Sigma \vdash \Gamma \; \mathsf{ctx} \qquad \Sigma \vdash A : \mathsf{Prop}}{\Sigma \vdash \Gamma; [\xi : A] \; \mathsf{ctx}} \; {\scriptstyle \xi \notin DV(\Gamma)}$$

Typing rules of proof-terms: $\langle \Sigma \rangle \Gamma \vdash t : A$

$$\frac{\Sigma \vdash \Gamma \; \mathsf{ctx}}{\langle \Sigma \rangle \Gamma \vdash \xi : A} \; {\scriptstyle (\xi:A) \in \Gamma} \qquad \frac{\langle \Sigma \rangle \Gamma \vdash t : A}{\langle \Sigma \rangle \Gamma \vdash t : A'} \; {\scriptstyle A =_\beta A'}$$

$$\frac{\langle \Sigma \rangle \Gamma; [\xi : A] \vdash t : B}{\langle \Sigma \rangle \Gamma \vdash \lambda \xi \,.\, t : A \Rightarrow B} \qquad \frac{\langle \Sigma \rangle \Gamma \vdash t : A \Rightarrow B \qquad \langle \Sigma \rangle \Gamma \vdash u : A}{\langle \Sigma \rangle \Gamma \vdash tu : B}$$

$$\frac{\langle \Sigma; [x : T] \rangle \Gamma \vdash t : B}{\langle \Sigma \rangle \Gamma \vdash t : \forall x : T \,.\, B} \; {\scriptstyle x \notin FV(\Gamma)} \qquad \frac{\langle \Sigma \rangle \Gamma \vdash t : \forall x : T \,.\, B \qquad \Sigma \vdash N : T}{\langle \Sigma \rangle \Gamma \vdash t : B\{x := N\}}$$

$$\frac{\Sigma; [x : T] \vdash B : \mathsf{Prop} \quad \Sigma \vdash N : T \qquad \langle \Sigma \rangle \Gamma \vdash t : B\{x := N\}}{\langle \Sigma \rangle \Gamma \vdash t : \exists x : T \,.\, B}$$

$$\frac{\Sigma; [x : T] \vdash A : \mathsf{Prop} \quad \Sigma \vdash B : \mathsf{Prop} \qquad \langle \Sigma \rangle \Gamma \vdash t : \forall x : T \,.\, (A \Rightarrow B)}{\langle \Sigma \rangle \Gamma \vdash t : (\exists x : T \,.\, A) \Rightarrow B}$$

Fig. 1. Typing rules of system $F\omega.2$

data-types. (Of course, propositions will be considered as data-types, but only at the level of proof-terms.)

Formally, the type system of object terms is based on a judgement $\Sigma \vdash M : T$ which expresses that the term M has type T under the assumptions in Σ. The corresponding typing assumptions are regrouped in *signatures*

Signatures $\qquad \Sigma \quad ::= \quad [x_1 : T_1; \ldots; x_n : T_n]$

that are finite ordered lists of declarations of the form $(x : T)$. In order to ensure that the terms T involved in such declarations are well-formed types, we also need a judgement $\Sigma \vdash$ which expresses that the signature Σ is well-formed. The typing rules for both judgements are then defined by mutual recursion thanks to the rules given in Fig. 1.

The proof system (i.e. the 'lower part') of system $F\omega.2$ follows the typing discipline *à la* Curry, so that proofs terms are actually pure λ-terms:

Proof-terms $\qquad t, u \quad ::= \quad \xi \quad | \quad \lambda\xi\,.\,t \quad | \quad tu$

By this, we mean that the universal and existential quantifications are treated as infinitary intersection and union types respectively, so that in practice the corresponding introduction and elimination rules have no impact on proof-terms. On the other hand, implication is introduced and eliminated as usual, by the means of λ-abstraction and application.

Proof-terms depend on assumptions that are declared in *logical contexts*:

Logical contexts $\qquad \Gamma \quad ::= \quad [\xi_1 : A_1; \ldots; \xi_k : A_k]\,.$

Since the object-terms A_i involved in such declarations cannot depend on proof-variables, the order of these declarations is irrelevant. To ensure that each A_i is a well-formed proposition (in a given signature), we introduce a judgement $\Sigma \vdash \Gamma$ ctx which expresses that the logical context Γ is well-formed in the signature Σ. Finally, the last judgement $\langle\Sigma\rangle\Gamma \vdash t : A$ expresses that in the signature Σ, the proof-term t is a proof of the sequent $\Gamma \vdash A$. The rules of inference for both judgements are given in Fig. 1.

Although the logic of system $F\omega.2$ is ultimately built on implication and quantifiers, the connectives $\wedge$, $\vee$ and units $\top$, $\bot$ are definable by the means of standard second-order encodings, as well as Leibniz equality.

The Primitive Existential Quantifier Although the existential quantifier could have been defined purely in terms of $\forall$ and $\Rightarrow$ (by the mean of the standard second-order encoding of the existential quantifier), system $F\omega.2$ introduces a primitive form of existential quantification that behaves exactly as an infinitary union type constructor. (See the last two proof-typing rules of Fig. 1.)

In practice, the primitive form of existential quantifier—which restores a symmetry between quantifiers in a spirit which is very close to the one of standard realisability—tends to produce more compact proof-terms than the second-order

encoding. But the main reason for using the primitive form of existential quantification is that denotations of propositions become much more readable in the normalisation model, a point which is important when studying intuitionistic forms of the axiom of choice as we will do in paragraphs 4.3 and 4.4.

From the point of view of provability of course, both forms of existential quantification are logically equivalent, and the full system with primitive form of existential quantification is nothing else but a conservative extension of the system restricted to $\forall$ and $\Rightarrow$ only.

3 Interpreting Zermelo's Set Theory

Equality & Compatibility axioms

(Refl)	$\forall x$	$x = x$
(Sym)	$\forall x, y$	$x = y \;\Rightarrow\; y = x$
(Trans)	$\forall x, y, z$	$x = y \;\Rightarrow\; y = z \;\Rightarrow\; x = z$
(Compat-L)	$\forall x, y, z$	$x = y \;\Rightarrow\; y \in z \;\Rightarrow\; x \in z$
(Compat-R)	$\forall x, y, z$	$x \in y \;\Rightarrow\; y = z \;\Rightarrow\; x \in z$

Zermelo's axioms

(Ext)	$\forall a, b \quad (\forall x \;\; x \in a \Leftrightarrow x \in b) \;\Rightarrow\; a = b$
(Pair)	$\forall a, b \; \exists c \; \forall x \quad x \in c \;\Leftrightarrow\; x = a \vee x = b$
(Select)	$\forall a \; \exists b \; \forall x \quad x \in b \;\Leftrightarrow\; x \in a \wedge \phi$ where ϕ is any formula such that $b \notin FV(\phi)$
(Power)	$\forall a \; \exists b \; \forall x \quad x \in b \;\Leftrightarrow\; (\forall y \;\; y \in x \Rightarrow y \in a)$
(Union)	$\forall a \; \exists b \; \forall x \quad x \in b \;\Leftrightarrow\; (\exists y \;\; y \in a \wedge x \in y)$
(Infinity)	$\exists a \quad \varnothing \in a \;\wedge\; (\forall x \;\; x \in a \;\Rightarrow\; x \cup \{x\} \in a)$

Fig. 2. Axioms of Zermelo's set theory

In this section, we briefly explain how to encode Intuitionistic Zermelo's set theory in system $F\omega.2$. We only give the basics of the encoding whose implementation details can be found in the author's PhD thesis [11].

Formally, Intuitionistic Zermelo's set theory (IZ) is the intuitionisitic first-order theory based on two binary predicates '$=$' (equality) and '$\in$' (membership) whose axioms are given in Fig. 2.

3.1 Sets as Pointed Graphs

A set is represented in system $F\omega.2$ as a *pointed graph*, that is, as a triple (X, A, a) where:

1. $X : \mathsf{Type}_1$ is the *carrier* (i.e. the type of *vertices*)
2. $A : X \to X \to \mathsf{Prop}$ is the *edge relation* (or *local membership*)
3. $a : X$ is the *root* of the pointed graph.

Intuitively, a pointed graph (X, A, a) can be thought as a kind of transitive closure of a set whose structure is given by the relation $A(x, y)$ (that can be read as: 'x is a local element of y') and by the root a (i.e. the entry point in the transitive closure).

Notice that unlike the terms X, A and a, the triple (X, A, a) is not a real object of system $F\omega.2$ (which does not provide any pairing mechanism) but only an informal notation to group related components. Similarly, we also introduce the following shorthands

$$\begin{array}{rcl}
\forall(X, A, a).\phi & \triangleq & \forall X : \mathsf{Type}_1 \,.\, \forall A : (X \to X \to \mathsf{Prop}) \,.\, \forall a : X \,.\, \phi \\
\exists(X, A, a).\phi & \triangleq & \exists X : \mathsf{Type}_1 \,.\, \exists A : (X \to X \to \mathsf{Prop}) \,.\, \exists a : X \,.\, \phi \\
\lambda(X, A, a).M & \triangleq & \lambda X : \mathsf{Type}_1 \,.\, \lambda A : (X \to X \to \mathsf{Prop}) \,.\, \lambda a : X \,.\, M
\end{array}$$

to denote the universal and existential quantifications as well as the λ-abstraction over the class of pointed graphs.

Extensional equality of set theory is interpreted as *bisimilarity* in the class of pointed graphs. This relation denoted by $(X, A, a) \approx (Y, B, b)$ is defined by:

$$\begin{array}{l}
(X, A, a) \approx (Y, B, b) \quad \triangleq \\
\quad \exists R : (X \to Y \to \mathsf{Prop}) \,. \\
\qquad \big(\forall x, x' : X \,.\, \forall y : Y \,. \quad A(x', x) \wedge R(x, y) \Rightarrow \exists y' : Y \,.\; B(y', y) \wedge R(x', y')\big) \wedge \\
\qquad \big(\forall y, y' : Y \,.\, \forall x : X \,. \quad B(y', y) \wedge R(x, y) \Rightarrow \exists x' : X \,.\; A(x', x) \wedge R(x', y')\big) \wedge \\
\qquad R(a, b) \,.
\end{array}$$

Membership is then interpreted as *shifted bisimilarity*, namely:

$$(X, A, a) \in (Y, B, b) \quad \triangleq \quad \exists b' : B \,. \quad B(b', b) \;\wedge\; (X, A, a) \approx (Y, B, b') \,.$$

Once the interpretation of equality and membership has been defined, it is straightforward to translate any formula ϕ of set theory as a proposition ϕ^* in system $F\omega.2$. For that, we consider a fixed mapping that associates three distinct variables $X_i : \mathsf{Type}_1$, $A_i : X_i \to X_i \to \mathsf{Prop}$ and $a_i : X_i$ of the type system $F\omega.2$ to each variable x_i of the first-order language of set theory, and we set:

$$\begin{array}{rcll}
(x_i = x_j)^* & \triangleq & (X_i, A_i, a_i) \approx (X_j, A_j, a_j) & \\
(x_i \in x_j)^* & \triangleq & (X_i, A_i, a_i) \in (X_j, A_j, a_j) & \\
(\phi \diamond \psi)^* & \triangleq & \phi^* \diamond \psi^* & (\diamond \in \{\Rightarrow;\ \wedge;\ \vee\}) \\
(Qx\ \phi)^* & \triangleq & Q(X, A, a) \,.\, \phi^* & (Q \in \{\forall;\ \exists\})
\end{array}$$

It is straightforward to check that this translation validates the equality and compatibility axioms of Fig. 2.

3.2 Soundness of Zermelo's Axioms

The main interest of interpreting sets as pointed graphs and equality as bisimilarity is that it automatically validates the axiom of extensionality:

Proposition 1 (Extensionality) — *The translation of the extensionality axiom is provable in system $F\omega.2$.*

As pointed out by [12, 9, 6], the interpretation of the extensionality axiom is the cornerstone of any computational interpretation of (the proofs of) set theory. On the other hand, proving that the bisimilarity relation is extensional w.r.t. the shifted bisimilarity relation is quite easy, and the formalisation of this proof in type-theory gives the corresponding λ-term for free.

The other axioms of Zermelo express the possibility of constructing new sets from other sets by several means. Except for the axiom of infinity, all the corresponding constructions have a natural translation in terms of pointed graphs that can be formalised in system $F\omega.2$, so that:

Proposition 2 (Finitary Zermelo axioms) — *The translation of* (Pair), (Power), (Union) *and of each instance of* (Select) *is provable in $F\omega.2$.*

However, the axiom of infinity poses another problem, since its interpretation in terms of pointed graphs requires an infinite small data-type whose existence cannot be proved in the logical framework we presented in section 2.[4]

As proposed in [11], a way to solve this problem is to add an extra universe below the universe Type_1 (i.e. a universe Type_0) from which one easily reconstructs a type of numerals by suitable encodings (so that the full construction of a model of IZ actually takes place in system $F\omega.3$).

In this paper, we consider a simpler solution by extending our logical framework with primitive numerals. For that, we introduce a small type $\mathsf{Nat} : \mathsf{Type}_1$ with two constructors $\mathsf{0} : \mathsf{Nat}$ and $\mathsf{S} : \mathsf{Nat}{\to}\mathsf{Nat}$, as well as two primitive functions $\mathsf{pred} : \mathsf{Nat}{\to}\mathsf{Nat}$ and $\mathsf{null} : \mathsf{Nat}{\to}\mathsf{Prop}$ with the computational rules

$$\mathsf{pred}(\mathsf{0}) \to_\beta \mathsf{0} \qquad \mathsf{pred}(\mathsf{S}(M)) \to_\beta M \qquad \mathsf{null}(\mathsf{0}) \to_\beta \top \qquad \mathsf{null}(\mathsf{S}(M)) \to_\beta \bot$$

from which we easily derive that S is injective, and that $\mathsf{S}(n) \neq \mathsf{0}$ for any $n : \mathsf{Nat}$ (where $\neq$ denotes the negation of Leibniz equality). In this framework, the induction principle comes for free provided we restrict all the quantifications with the predicate $\mathsf{wf\_nat}$ defined by

$$\mathsf{wf\_nat}(n) \quad \stackrel{\Delta}{\equiv} \quad \forall P : (\mathsf{Nat}{\to}\mathsf{Prop})\,.\;\; P(\mathsf{0}) \Rightarrow (\forall p : \mathsf{Nat}\,.\, P(p) \Rightarrow P(\mathsf{S}(p))) \Rightarrow P(n)\,.$$

Using this, it is then easy to build a pointed graph that represents the set ω of von Neumann numerals, so that:

[4] A simple counter-model is the obvious extension of the finitary boolean model of Church's theory of simple types to system $F\omega.2$, in which small types are interpreted by hereditarily finite sets whereas large types are interpreted by the elements of a fixed set-theoretical universe.

Proposition 3 (Infinity) — *The translation of the axiom of infinity is provable in system $F\omega.2$ extended with primitive numerals.*

3.3 Beyond Zermelo

It is natural to ask whether the former soundness result can be extended to IZF, which is obtained by adding the *collection scheme* to IZ:

$$(\textsc{Coll}) \qquad \forall a \quad (\forall x{\in}a\ \exists y\ \ \phi) \ \ \Rightarrow \ \ \exists b\ \ \forall x{\in}a\ \exists y{\in}b\ \ \phi$$

(where ϕ is an arbitrary formula such that $b \notin FV(\phi)$).

Unfortunately, the answer is negative, since the soundness of (Coll) would entail the relative consistency of ZF w.r.t. $F\omega.2$ with primitive numerals (via Gödel's negation translation, which maps provable formulas of ZF to provable formulas of IZF). On the other hand, our type system can be seen as a subsystem of the calculus of constructions with universes whose strong normalisation property (and logical consistency) has been proved in ZF [10], so that its proof-theoretical strength is actually less than the one of ZF.[5]

4 The Normalisation Model $\mathcal{M}$

This section is devoted to the construction of a strong normalisation model $\mathcal{M}$ of $F\omega.2$. The main interest of such a model is not only that it constitutes the main device for proving the strong normalisation property of system $F\omega.2$ but that it naturally validates more propositions than the syntax, and thus suggests extensions of it.

4.1 Interpreting Object-Terms

The model $\mathcal{M}$ is defined in classical set theory with axiom of choice (ZFC). To interpret type-theoretical universes, we also assume the following axiom:

Axiom 4 — *There exists two nested ZF-universes.*

By ZF-universe (or set-theoretical universe), we mean any transitive set $\mathfrak{U}$ that fulfils the following conditions:

1. $A \in \mathfrak{U} \ \Rightarrow\ \mathfrak{P}(A) \in \mathfrak{U}$
2. $A \in \mathfrak{U} \ \wedge\ (B_x)_{x\in A} \in \mathfrak{U}^A \ \Rightarrow\ (\bigcup_{x\in A} B_x) \in \mathfrak{U}$
3. $\omega \in \mathfrak{U}$

ZF-universes are closed under all the operations that can be defined in ZFC, among which the generalised cartesian product that will be used to interpret dependent products:

$$A \in \mathfrak{U} \ \wedge (B_x)_{x\in A} \in \mathfrak{U}^A \quad \Rightarrow \quad \Big(\prod_{x\in A} B_x\Big) \in \mathfrak{U}.$$

[5] We conjecture that system $F\omega.2$ with primitive numerals has the same proof-theoretical strength as higher-order Zermelo's set theory.

In the following, we assume that $\mathfrak{U}_1$ and $\mathfrak{U}_2$ are two ZF-universes such that $\mathfrak{U}_1 \in \mathfrak{U}_2$. Since we want to interpret any signature, we must prohibit empty types by setting $[\![\mathsf{Type}_i]\!] = \mathfrak{U}_i \setminus \{\varnothing\}$ for $i \in \{1;2\}$. (Notice that these sets are still closed under generalised cartesian products, thanks to the axiom of choice.)

Propositions are interpreted as saturated sets [10]. The set of all saturated sets, denoted by $\mathfrak{SAT}$, is closed under arbitrary (non-empty) unions and intersections, and forms a complete lattice whose top element is SN, and whose bottom element is the set *Neut* of neutral terms. Moreover, $\mathfrak{SAT}$ is closed under the construction $S \to T$ defined by

$$S \to T \quad \triangleq \quad \{t \in \Lambda; \quad \forall u \in S \quad tu \in T\} \quad \in \quad \mathfrak{SAT}\,.$$

Let $\mathcal{M} = \mathfrak{U}_2$. A valuation is a function $\rho : \mathcal{V} \to \mathcal{M}$, where $\mathcal{V}$ denotes the set of object-variables. The interpretation $(M, \rho) \mapsto [\![M]\!]_\rho$ is defined by

$$[\![\mathsf{Type}_i]\!]_\rho \;=\; \mathfrak{U}_i \setminus \{\varnothing\} \qquad [\![\mathsf{Prop}]\!]_\rho \;=\; \mathfrak{SAT} \qquad [\![x]\!]_\rho \;=\; \rho(x)$$

$$[\![\Pi x : T\,.\,U]\!]_\rho = \prod_{v \in [\![T]\!]_\rho} [\![U]\!]_{(\rho; x \leftarrow v)} \qquad [\![\lambda x : T\,.\,M]\!]_\rho = \big(v_{\in [\![T]\!]_\rho} \mapsto [\![M]\!]_{(\rho; x \leftarrow v)}\big)$$

$$[\![M(N)]\!]_\rho = [\![M]\!]_\rho([\![N]\!]_\rho) \qquad [\![A \Rightarrow B]\!]_\rho = [\![A]\!]_\rho \to [\![B]\!]_\rho$$

$$[\![\forall x : T\,.\,A]\!]_\rho = \bigcap_{v \in [\![T]\!]_\rho} [\![A]\!]_{(\rho; x \leftarrow v)} \qquad [\![\exists x : T\,.\,A]\!]_\rho = \bigcup_{v \in [\![T]\!]_\rho} [\![A]\!]_{(\rho; x \leftarrow v)}$$

whereas the constants Nat, $\mathsf{0}$, S, pred and null are interpreted in the obvious way. Notice that the right-hand side of the equation which gives the interpretation of the application may be undefined, so that the function $\rho \mapsto [\![M]\!]_\rho$ is partial.

Each signature Σ is interpreted as the set $[\![\Sigma]\!]$ of all valuations ρ such that $\rho(x) \in [\![T]\!]$ for each declaration $(x : T) \in \Sigma$ (i.e. the set of *adapted valuations*). We finally get by a straightforward induction:

Proposition 5 (Soundness of typing) — *If $\Sigma \vdash M : T$, then for all $\rho \in [\![\Sigma]\!]$, the denotations $[\![M]\!]_\rho$ and $[\![T]\!]_\rho$ are well-defined, and $[\![M]\!]_\rho \in [\![T]\!]_\rho$.*

As for any strong normalisation model, our model enjoys the crucial property that the denotation of a (well-formed) type is always inhabited. For this reason, any well-formed signature admits at least an adapted valuation.

4.2 The Normalisation Invariant

As well as we interpreted signatures as sets of adapted valuations, each logical context $\Gamma = [\xi_1 : A_1; \ldots; \xi_n : A_n]$ is now interpreted as a set of adapted substitutions. Formally, $[\![\Gamma]\!]_\rho$ is defined (for a given valuation ρ) as the set of all substitutions $\sigma = [\xi_1 := u_1; \ldots; \xi_n := u_n]$ such that $u_i \in [\![A_i]\!]_\rho$ for all $i = 1..n$.

We can now express the strong normalisation invariant of (Curry-style) system $F\omega.2$ as follows:

Proposition 6 (Strong normalisation invariant) — *If* $\langle\Sigma\rangle\Gamma\vdash t:A$ *is derivable, then for all* $\rho\in[\![\Sigma]\!]$ *and for all* $\sigma\in[\![\Gamma]\!]_\rho$ *one has* $t[\sigma]\in[\![A]\!]_\rho$.

By instantiating this result to an arbitrary adapted valuation $\rho\in[\![\Sigma]\!]$ and to the identity substitution (adapted to Γ) we conclude that:

Corollary 7 (Strong normalisation) — *In* $F\omega.2$*, all the typable proof-terms are strongly normalising* λ*-terms.*

Truth in the Model Although the strong normalisation model $\mathcal{M}$ is quite close of a realisability model (think of the interpretation of $\forall$, $\exists$ and $\Rightarrow$), an important difference with the standard realisability approach is that in $\mathcal{M}$, the denotation of a proposition is never empty, since a saturated set contains at least all the neutral terms. To define a suitable notion of truth in the model, one has to exclude such 'paraproofs' by only considering *closed* terms.

Formally, we will say that a proposition A is true in the model $\mathcal{M}$ if its denotation $[\![A]\!]\in\mathfrak{SAT}$ contains at least a closed proof-term. Notice that the proposition $\bot=\forall p:\mathsf{Prop}\,.\,p$ (whose denotation is *Neut*) is false in the model, which shows that system $F\omega.2$ is logically consistent.

4.3 Axiom of Choice and Uniform Collection

An example of a true but non-provable proposition is the following formulation of the axiom of choice in $F\omega.2$:

$$(\forall x:T\,.\,\exists x:U\,.\,A(x,y))\;\Rightarrow\;\exists f:T\to U\,.\,\forall x:T\,.\,A(x,f(x))$$

where T and U are arbitrary data-types. Although this proposition is not provable in $F\omega.2$, it is straightforward to check that in the model $\mathcal{M}$, its denotation contains the closed proof-term $\lambda\xi\,.\,\xi$ since both members of the corresponding implication have the very same denotation, namely:

$$\bigcap_{x\in T}\bigcup_{y\in U}A(x,y)\quad=\quad\bigcup_{f\in U^T}\bigcap_{x\in T}A(x,f(x))\,.$$

(In the model of course, the left-to-right inclusion relies on the axiom of choice.)

This fact suggests that we can add the following proof principle

$$\frac{\langle\Sigma\rangle\Gamma\vdash t:\forall x:T\,.\,\exists y:U\,.\,A(x,y)}{\langle\Sigma\rangle\Gamma\vdash t:\exists f:(T\to U)\,.\,\forall x:T\,.\,A(x,f(x))}$$

to our system without breaking the normalisation invariant.

The Uniform Collection Scheme Coming back to our translation of set theory, it is interesting to notice that this additional rule is actually sufficient for proving a weak form of collection scheme—that will be called here the *uniform collection scheme*—whose statement is the following:

$$(\forall x\;\exists y\;\phi(x,y))\quad\Rightarrow\quad\forall a\;\exists b\quad\forall x{\in}a\;\exists y{\in}b\quad\phi(x,y)$$

This statement is weaker than the collection scheme since it relies on a totality assumption which is stronger than that of collection, by requiring that the binary relation $\phi(x, y)$ should be total not only on a, but on the whole universe. (Of course, the uniform collection is classically equivalent to the collection scheme.)

4.4 An Intuitionistic Choice Operator

The difficulty in realising the full collection scheme is that for any $x \in a$, the implicit witness y given by the proof of $\exists y\ \phi(x, y)$ does not only depend on x, but also on the proof of the relativisation $x \in a$. To overcome this difficulty, we propose to express this dependency by using a trick inspired by Krivine's interpretation of the denumerable axiom of choice [6, 7].

For that, we extend system $F\omega.2$ with an *intuitionistic choice operator* that associates to any predicate $A(x)$ a sequence of objects $\boldsymbol{\epsilon}x\,{:}\,T\,.\,A(x) : \mathsf{Nat}{\to}T$ whose intended meaning is: if A holds for some $x : T$, then A holds for some element of the sequence $\boldsymbol{\epsilon}x\,{:}\,T\,.\,A(x)$.

Formally, the interpretation of this new construction in the model $\mathcal{M}$ relies on a fixed enumeration $(t_n)_{n\in\omega}$ of all the λ-terms. The construction $\boldsymbol{\epsilon}x\,{:}\,T\,.\,A(x)$ is then interpreted as a function $f \in T^\omega$ (defined by using the axiom of choice) such that for all $n \in \omega$, either:

1. $f(n)$ is some $x \in T$ such that $t_n \in A(x)$ if such an element exists; or
2. $f(n)$ is an arbitrary element of T otherwise.

Once the function $f \in T^\omega$ that interprets the construction $\boldsymbol{\epsilon}x\,{:}\,T\,.\,A(x)$ has been defined, it is straightforward to check that $\bigcup_{x\in T} A(x) \subset \bigcup_{n\in\omega} A(f(n))$. (The converse inclusion also holds, but is not of interest.) Coming back to our logical framework, this inclusion means that the model validates the typing rule

$$\frac{\langle\Sigma\rangle\Gamma \vdash t : \exists x\,{:}\,T\,.\,A(x)}{\langle\Sigma\rangle\Gamma \vdash t : \exists n\,{:}\,\mathsf{Nat}\,.\,A\big((\boldsymbol{\epsilon}x\,{:}\,T\,.\,A(x))(n)\big)}$$

which can thus be added to our type system without harm for the strong normalisation property.

5 The Extended Framework $F\omega.2^{++}$

Using the material we presented above, we can now define an extended Curry-style framework called $F\omega.2^{++}$ that actually contains enough proof principles to allow the (translation of the) collection scheme to be derived.

5.1 Syntax and Typing Rules

Object-Terms and Signatures The syntax and typing rules of object-terms and signatures of system $F\omega.2^{++}$ are the one of system $F\omega.2$ extended with primitive numerals and the intuitionistic choice operator discussed in paragraph 4.4 (see Fig. 3). The corresponding reduction rules are the usual β-rule and the reduction rules of the constants pred and null that we gave in paragraph 3.2.

Judgements $\Sigma \vdash$, $\Sigma \vdash M : T$ and $\Sigma \vdash \Gamma$ ctx

Rules of Fig. 1 p. 3 + constants Nat, 0, S, pred, null

+

$$\frac{\Sigma; [x : T] \vdash A : \mathsf{Prop}}{\Sigma \vdash \epsilon x : T \,.\, A : \mathsf{Nat} \to T}$$

Propositional subtyping: $\Sigma \vdash A \leq B$

$$\frac{\Sigma \vdash A : \mathsf{Prop} \qquad \Sigma \vdash A' : \mathsf{Prop}}{\Sigma \vdash A \leq A'}\ {\scriptstyle A =_\beta A'} \qquad \frac{\Sigma \vdash A \leq B \qquad \Sigma \vdash B \leq C}{\Sigma \vdash A \leq C}$$

$$\frac{\Sigma \vdash A' \leq A \qquad \Sigma \vdash B \leq B'}{\Sigma \vdash\ A \Rightarrow B\ \leq\ A' \Rightarrow B'}$$

$$\frac{\Sigma; [x : T] \vdash A \leq B}{\Sigma \vdash\ A\ \leq\ \forall x : T \,.\, B}\ {\scriptstyle x \notin FV(A)} \qquad \frac{\Sigma; [x : T] \vdash A : \mathsf{Prop} \qquad \Sigma \vdash N : T}{\Sigma \vdash\ \forall x : T \,.\, A\ \leq\ A\{x := N\}}$$

$$\frac{\Sigma; [x : T] \vdash A : \mathsf{Prop} \qquad \Sigma \vdash N : T}{\Sigma \vdash\ A\{x := N\}\ \leq\ \exists x : T \,.\, A} \qquad \frac{\Sigma; [x : T] \vdash A \leq B}{\Sigma \vdash \exists x : T \,.\, A \leq B}\ {\scriptstyle x \notin FV(B)}$$

$$\frac{\Sigma; [x : T] \vdash A : \mathsf{Prop} \qquad \Sigma \vdash B : \mathsf{Prop}}{\Sigma \vdash\ \forall x{:}T \,.\, (A \Rightarrow B)\ \leq\ (\exists x{:}T \,.\, A) \Rightarrow B}$$

$$\frac{\Sigma; [x : T;\ y : U] \vdash A : \mathsf{Prop}}{\Sigma \vdash\ \forall x{:}T \,.\, \exists y{:}U \,.\, A\ \leq\ \exists f : (\Pi x{:}T.U) \,.\, \forall x{:}T \,.\, A\{y{:=}f(x)\}}$$

$$\frac{\Sigma; [x : T] \vdash A : \mathsf{Prop}}{\Sigma \vdash\ \exists x : T \,.\, A\ \leq\ \exists n : \mathsf{Nat} \,.\, A\{x := (\epsilon x : T \,.\, A)(n)\}}$$

Typing rules of proofs-terms: $\langle \Sigma \rangle \Gamma \vdash t : A$

$$\frac{\Sigma \vdash \Gamma\ \mathsf{ctx}}{\langle \Sigma \rangle \Gamma \vdash \xi : A}\ {\scriptstyle (\xi : A) \in \Gamma} \qquad \frac{\langle \Sigma \rangle \Gamma; [\xi : A] \vdash t : B}{\langle \Sigma \rangle \Gamma \vdash \lambda \xi \,.\, t : A \Rightarrow B}$$

$$\frac{\langle \Sigma \rangle \Gamma \vdash t : A \Rightarrow B \qquad \langle \Sigma \rangle \Gamma \vdash u : A}{\langle \Sigma \rangle \Gamma \vdash tu : B}$$

$$\frac{\langle \Sigma; [x : T] \rangle \Gamma \vdash t : A}{\langle \Sigma \rangle \Gamma \vdash t : \forall x : T \,.\, A}\ {\scriptstyle x \notin FV(\Gamma)} \qquad \frac{\langle \Sigma \rangle \Gamma \vdash t : A \qquad \Sigma \vdash A \leq A'}{\langle \Sigma \rangle \Gamma \vdash t : A'}$$

Fig. 3. Rules of inference of system $F\omega.2^{++}$

Logical Contexts and Proof-Terms Logical contexts have the same syntax and formation rules as in system $F\omega.2$, and proof-terms are still pure λ-terms.

The main novelty of system $F\omega.2^{++}$ is the introduction of a new form of judgement called *propositional subtyping* and written $\Sigma \vdash A \leq B$. This judgement—whose logical meaning is a *direct implication*—is intended to capture the different inclusions between saturated sets that we pointed out throughout section 4. In particular, this judgement (whose rules of inference are given in Fig. 3) now incorporates the introduction and elimination rules for both quantifiers, as well as a very natural rule that expresses the contravariance of the domain of implication (and the covariance of its codomain) w.r.t. subtyping. The last two subtyping rules express both intuitionistic forms of axiom of choice discussed in paragraphs 4.3 and 4.4. (Notice that the latter is actually sufficient to derive the collection scheme.)

In this framework, the proof-typing rules become simpler than in system $F\omega.2$, for that many logical rules have been incorporated in the propositional subtyping judgement, and are now accessed via a standard subsumption rule.

5.2 Strong Normalisation

From the results of section 4, it is clear that $\mathcal{M}$ is still a strong normalisation model for $F\omega.2^{++}$. The proofs of propositions 5 and 6 are easily adapted to system $F\omega.2^{++}$ by interpreting the new construction $\boldsymbol{\epsilon} x : T \,.\, A$ as explained in 4.4. Notice that in order to prove the strong normalisation invariant for system $F\omega.2^{++}$, we first have to check the soundness of propositional subtyping:

Proposition 8 — *If $\Sigma \vdash A \leq B$, then for all $\rho \in [\![\Sigma]\!]$ one has $[\![A]\!]_\rho \subset [\![B]\!]_\rho$.*

From this, we easily deduce that all the well-formed proof-terms of system $F\omega.2^{++}$ are strongly normalising, and that the system is logically consistent.

5.3 Deriving the Collection Scheme

Of course, the main interest of using system $F\omega.2^{++}$ is that we can now realise the collection scheme, by the means of the intuitionistic choice operator:

Proposition 9 (Collection scheme) — *The translation of each instance of* (Coll) *is provable in system $F\omega.2^{++}$.*

Notice that this result entails that the proof-theoretical strength of system $F\omega.2^{++}$ is at least the one of IZF/ZF.

5.4 Extracting Functions from Proofs

In this paragraph, we aim to show that from any proof (in IZF) of a statement ϕ of the form

$$\phi \quad \equiv \quad \forall x{\in}\omega \;\; \exists y{\in}\omega \;\; \psi(x,y)$$

(where ψ is an arbitrary formula s.t. $FV(\psi) = \{x; y\}$) we can extract a strongly normalising λ-term that computes the corresponding function.

The extraction process relies on the fact that, internally, the set ω of von Neumann numerals is implemented [11] as a pointed graph (X, A, a) equipped with an injection $i : \mathsf{Nat} \to X$ that associates to any object $n : \mathsf{Nat}$ such that $\mathsf{wf\_nat}(n)$ the vertex $i(n) : X$ which represents the corresponding von Neumann numeral in the graph (X, A). Using this, it is easy to derive from a proof of ϕ (in IZF) a proof-term (in $F\omega.2^{++}$) of the statement:

$$\forall n : \mathsf{Nat}\,.\ \mathsf{wf\_nat}(n) \ \Rightarrow\ \exists p : \mathsf{Nat}\,.\ \mathsf{wf\_nat}(p) \wedge \phi^*(X, A, i(n), X, A, i(p))$$

(where ψ^* denotes the translation of the binary relation ψ in $F\omega.2^{++}$). By dropping the second component of the conjunction, we thus get a proof-term

$$\lambda\xi\,.\,t'\xi(\lambda\xi_1, \xi_2\,.\,\xi_1) \quad : \quad (\exists n : \mathsf{Nat}\,.\,\mathsf{wf\_nat}(n)) \ \Rightarrow\ (\exists p : \mathsf{Nat}\,.\,\mathsf{wf\_nat}(p))$$

that obviously computes the desired function, since:

Fact 10 — *The closed inhabitants of the saturated set* $[\![\mathsf{wf\_nat}(x)]\!]_{x \leftarrow n}$ *(for a given* $n \in \omega$*) are the SN-terms whose normal form is Church numeral* $\lceil n \rceil$*.*

References

1. P. Aczel. Non well-founded sets. *Center for the Study of Language and Information*, 1988.
2. H. Barendregt. Introduction to generalized type systems. Technical Report 90-8, University of Nijmegen, Department of Informatics, May 1990.
3. B. Barras, S. Boutin, C. Cornes, J. Courant, J.C. Filliâtre, E. Giménez, H. Herbelin, G. Huet, C. Muñoz, C. Murthy, C. Parent, C. Paulin, A. Saïbi, and B. Werner. The Coq Proof Assistant Reference Manual – Version V6.1. Technical Report 0203, INRIA, August 1997.
4. H. Friedman. Some applications of Kleene's methods for intuitionistic systems. In *Cambridge Summer School in Mathematical Logic*, volume 337 of *Springer Lecture Notes in Mathematics*, pages 113–170. Springer-Verlag, 1973.
5. J.H. Geuvers and M.J. Nederhof. A modular proof of strong normalization for the calculus of constructions. In *Journal of Functional Programming*, volume 1,2(1991), pages 155–189, 1991.
6. J.-L. Krivine. Typed lambda-calculus in classical Zermelo-Fraenkel set theory. *Archive for Mathematical Logic*, 40(3):189–205, 2001.
7. J.-L. Krivine. Dependent choice, 'quote' and the clock. *Theoretical Computer Science*, 2003.
8. P. Martin-Löf. *Intuitionistic Type Theory*. Bibliopolis, Napoli, 1984.
9. D. McCarty. *Realizability and Recursive Mathematics*. PhD thesis, Ohio State University, 1984.
10. P.-A. Melliès and B. Werner. A generic normalization proof for pure type systems. In *Proceedings of TYPES'96*, 1997.
11. A. Miquel. *Le calcul des constructions implicite: syntaxe et sémantique*. PhD thesis, Université Paris VII, 2001.
12. J. Myhill. Some properties of intuitionistic Zermelo-Fraenkel set theory. In *Cambridge Summer School in Mathematical Logic*, volume 337 of *Springer Lecture Notes in Mathematics*, pages 206–231. Springer-Verlag, 1973.