An Analysis of Tennenbaum's Theorem in Constructive Type Theory

Marc Hermes
Universität des Saarlandes, Department of Mathematics, Saarbrücken, Germany

Dominik Kirst
Universität des Saarlandes, Saarland Informatics Campus, Saarbrücken, Germany

Abstract

Tennenbaum's theorem states that the only countable model of Peano arithmetic (PA) with computable arithmetical operations is the standard model of natural numbers. In this paper, we use constructive type theory as a framework to revisit and generalize this result.

The chosen framework allows for a synthetic approach to computability theory, by exploiting the fact that, externally, all functions definable in constructive type theory can be shown computable. We internalize this fact by assuming a version of Church's thesis expressing that any function on natural numbers is representable by a formula in PA. This assumption allows for a conveniently abstract setup to carry out elegant computability arguments and a feasible mechanization.

Concretely, we constructivize several classical proofs and present one inherently constructive rendering of Tennenbaum's theorem, all following arguments from the literature. Concerning the classical proofs in particular, the constructive setting allows us to highlight differences in their conclusions which are not visible classically. All versions are accompanied by mechanizations in the Coq proof assistant.

2012 ACM Subject Classification Replace ccsdesc macro with valid one

Keywords and phrases first-order logic, Peano arithmetic, Tennenbaum's theorem, constructive type theory, Church's thesis, synthetic computability, Coq

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Supplementary Material `https://www.ps.uni-saarland.de/extras/tennenbaum`

1 Introduction

In classical logic, it is relatively straightforward to establish the existence of non-standard models of first-order Peano arithmetic (PA), showing that the theory does not possess a unique model up to isomorphism and is therefore not categorical. Following a typical textbook presentation [3], one way to construct a non-standard model is by adding a new constant symbol c to the language of PA together with the enumerable list of new axioms $c \neq 0$, $c \neq 1$, $c \neq 2$, etc. This yields a theory with the property that every finite subset of its axioms is satisfied by the standard model $\mathbb{N}$, since we can always give a large enough interpretation of the constant c in $\mathbb{N}$. Hence by the compactness theorem, the full theory has a model $\mathcal{M}$, which must then be non-standard, as the interpretation of c in $\mathcal{M}$ corresponds to an element which is larger then any number $n : \mathbb{N}$.

This construction comes with some remarkable consequences. Since PA can prove that for every bound n, the products of the form $\prod_{k \leq n} a_k$ exist, the presence of the non-standard element c in $\mathcal{M}$ gives raise to infinite products $\prod_{k \leq c} a_k$. The general PA model $\mathcal{M}$ can therefore exhibit behaviours disagreeing with the usual intuition that computations in PA are finitary, which are largely based on the familiarity with the standard model $\mathbb{N}$.

However, these intuitions are not too far off the mark, as was demonstrated by Stanley Tennenbaum [33] in a remarkable theorem: $\mathbb{N}$ is (up to isomorphism) the only computable model of first-order PA. Here, a model is considered *computable* if its elements can be coded by numbers in $\mathbb{N}$, and the arithmetic operations on model elements can be realized by computable functions on these codes. Usually, this theorem is formulated in a classical framework such as ZF set theory and the precise meaning of *computable* is given by making reference to a concrete model of computation like Turing machines, μ-recursive functions, or the λ-calculus [12, 30]. But as is custom, the computability of a function is rarely proven by exhibiting an explicit construction in the chosen model, but by a call to the *Church-Turing thesis* expressing that every function intuitively computable will be computable in the model.

To offer an alternative and more rigorous perspective, in this paper we revisit Tennenbaum's theorem in constructive type theory. Since we can externally observe that all functions of constructive type theory are computable, we have the freedom to simply treat every function as being computable, without exhibiting any internal representation in a formal model of computation. This is known as the *synthetic* approach to computability [27, 1], simplifying computability arguments to the point where the above mentioned intuitions usually suffice to give complete proofs with no formal gaps, even feasible to mechanize in a proof assistant.

Definitions and notions of computability theory are then also formulated synthetically. This leads to a simplification already as it comes to the statement of Tennenbaum's theorem: in the most natural semantics interpreting the arithmetic operations with type-theoretic functions, simply *all* models are computable and we no longer need "computable model" as part of the theorem statement. We furthermore *internalize* computability by assuming a version of *Church's thesis* [15], an axiom which expresses that *all* functions $\mathbb{N} \to \mathbb{N}$ have a representation in an internally captured formalism, in our case PA. With this setup the, all arguments involving a computability proof reduce to the constructions of type-theoretic functions, giving a formal counterpart to the informal appeal to the Church-Turing thesis.

Based on this framework, we follow the classical presentations of Tennenbaum's theorem [12, 30] to develop constructive versions only relying on *Markov's principle* instead of the *the law of excluded middle.* This yields several classically equivalent variations that differ in the strength of their respective assumptions and conclusions under the constructive lens, which we complement by also adapting the inherently constructive variant given by McCarty [19, 20].

Concretely, our contributions can be summarized as follows:

- We formulate, establish, and compare several versions of Tennenbaum's theorem in the setting of synthetic computability based on constructive type theory.
- We generalize Tennenbaum's theorem to models with decidable divisibility relation that need not be computable in general or even enumerable (Corollary 41).
- We provide a Coq mechanization covering all results studied in this paper.[1]

To make the paper self-contained, we start out in Section 2 by giving a quick introduction to the essential features of constructive type theory, synthetic computability, and the specification of first-order logic inside of the type theory. We continue with a presentation of the first-order axiomatization of PA as given in previous work [13], and of basic results about its standard and non-standard models (Section 4). These are then used in Section 6 to establish results that allow the encoding of predicates on $\mathbb{N}$ in non-standard models, which are essential in the proof of Tennenbaum's theorem. In Section 5 we introduce the chosen formulation of Church's thesis, which is then used (Section 7) to derive Tennenbaum's theorem in several variations. We conclude in Section 8 with observations about these proofs and remarks on

[1] The only facts with no formal counterpart in Coq are clearly marked as "Hypothesis" in this text. The full mechanization is accessible from the web page listed as supplementary material and systematically hyperlinked with the highlighted statements in the PDF version of this paper.

the Coq mechanization as well as related and future work.

2 Preliminaries

2.1 Constructive Type Theory

Our framework is the calculus of inductive constructions [5, 23] which is implemented in the Coq proof assistant [32], providing a predicative hierarchy of *type universes* above a single impredicative universe $\mathbb{P}$ of *propositions* and the capability of inductive type definitions. On type level, we have the unit type $\mathbb{1}$ with a single element, the void type $\mathbb{0}$, function spaces $X \to Y$, products $X \times Y$, sums $X + Y$, dependent products $\forall(x : X).\, A\,x$, and dependent sums $\Sigma(x : X).\, A\,x$. On the propositional level, the notions as listed in the order above, are denoted by the usual logical notation ($\top$, $\bot$, $\to$, $\wedge$, $\vee$, $\forall$, $\exists$).[2] It is important to note that the so-called *large eliminations* from the impredicative $\mathbb{P}$ into higher types of the hierarchy is restricted. In particular it is therefore generally not possible to show $(\exists x.\, p\,x) \to \Sigma x.\, p\,x$.[3] The restriction does however allow for large elimination of the equality predicate $= :\ \forall X.\, X \to X \to \mathbb{P}$, as well as function definitions by well-founded recursion.

We will also use the basic inductive types of *Booleans* ($\mathbb{B} := \mathsf{tt} \mid \mathsf{ff}$), *Peano natural numbers* ($n : \mathbb{N} := 0 \mid n + 1$), the *option type* ($\mathcal{O}(X) := {}^\circ x \mid \emptyset$) and *lists* ($l : \mathsf{List}(X) := [\,] \mid x :: l$). Furthermore, by X^n we denote the type of *vectors* $\vec{v}$ of length $n : \mathbb{N}$ over X.

▶ **Definition 1.** *A proposition $P : \mathbb{P}$ is called* definite *if $P \vee \neg P$ holds and* stable *if $\neg\neg P \to P$. The same terminology is used for predicates $p : X \to \mathbb{P}$ given they are pointwise definite or stable. We furthermore want to recall the following logical principles:*

$$\mathsf{LEM} := \forall P : \mathbb{P}.\, P \vee \neg P \qquad \textit{(Law of Excluded Middle)}$$

$$\mathsf{DNE} := \forall P : \mathbb{P}.\, \neg\neg P \to P \qquad \textit{(Double Negation Elimination)}$$

$$\mathsf{MP} := \forall f : \mathbb{N} \to \mathbb{N}.\, \mathsf{stable}\,(\exists n.\, f n = 0) \qquad \textit{(Markov's Principle)}$$

For convenience, we adapt the reading of double negated statements like $\neg\neg P$ as "*potentially P*"[4] [2].

▶ Remark (Handling $\neg\neg$). Given any propositions A, B we constructively have $(A \to \neg B) \leftrightarrow (\neg\neg A \to \neg B)$, telling us that whenever we are trying to prove a negated goal, we can remove double negations in front of any available assumption. More specifically then, any statement of the form $\neg\neg A_1 \to \ldots \to \neg\neg A_n \to \neg\neg C$, is equivalent to $A_1 \to \ldots \to A_n \to \neg\neg C$ and since $C \to \neg\neg C$ holds, it furthermore suffices to show $A_1 \to \ldots \to A_n \to C$ in this case. In the following, we will make use of these facts without further notice.

2.2 Synthetic Computability

As already expressed in Section 1, the axiom-free type theory allows us to view all functions of the type theory as computable. We then get simplified definitions [10] of the usual notions from computability theory:

[2] Negation $\neg A$ is used as an abbreviation for both $A \to \bot$ and $A \to \mathbb{0}$.
[3] The direction $(\Sigma x.\, p\,x) \to \exists x.\, p\,x$ is however always provable. Intuitively, one can think of $\exists x.\, p\,x$ as stating the mere existence of some value satisfying p, while $\Sigma x.\, p\,x$ is a type that also carries a value satisfying this.
[4] $\neg\neg P$ expresses the impossibility of P being wrong, so it represents a guarantee that P can potentially be shown correct.

▶ **Definition 2** (Enumerability). *Let $p : X \to \mathbb{P}$ be some predicate. We say that p is* enumerable *if there is an* enumerator $f : \mathbb{N} \to \mathcal{O}(X)$ *such that* $\forall x : X.\, p\,x \leftrightarrow \exists n.\, f n = {}^\circ x$.

▶ **Definition 3** (Decidability). *Let $p : X \to \mathbb{P}$ be some predicate. We call $f : X \to \mathbb{B}$ a* decider *for p and write* $\mathsf{decider}\, p\, f$ *iff* $\forall x : X.\, p\,x \leftrightarrow f x = \mathsf{tt}$. *We then define the following notions of decidability:*

- $\mathsf{Dec}\, p := \exists f : X \to \mathbb{B}.\, \mathsf{decider}\, p\, f$
- $\mathsf{Dec}_\Sigma\, p := \Sigma f : X \to \mathbb{B}.\, \mathsf{decider}\, p\, f$
- $\mathsf{dec}(P : \mathbb{P}) := P + \neg P$.

In all cases we will often refer to the predicate or proposition simply as being decidable.

We will expand the synthetic vocabulary with notions for types. In the textbook setting, many of them can only be defined for sets which are in bijection with $\mathbb{N}$, but synthetically they can be handled in a more uniform way.

▶ **Definition 4.** *We call a type X*

- enumerable *if $\lambda x : X.\top$ is enumerable,*
- discrete *if there exists a decider for equality $=$ on X,*
- separated *if there exists a decider for apartness $\neq$ on X,*
- witnessing *if* $\forall p : X \to \mathbb{P}.\, \mathsf{Dec}_\Sigma\, p \to (\exists x.\, p\,x) \to \Sigma x.\, p\,x$.

▶ **Fact 5.** *In the particular type theory we use, $\mathbb{N}$ is witnessing.*

2.3 First-Order Logic

In order to study Tennenbaum's theorem, we need to give a description of the first-order theory of PA and the associated theory of *Heyting arithmetic* (HA), which has the same axiomatization, but uses intuitionistic first-order logic. We follow work in [10, 11, 13] and describe first-order logic inside of the constructive type theory, by inductively defining formulas, terms and the deduction system. We then define a semantics for this logic, which uses Tarski-models and interprets formulas over the respective domain of the model. The type of natural numbers $\mathbb{N}$ will then naturally be a model of HA.

Before specializing to one theory, we keep the definition of first-order logic general and fix some arbitrary signature $\Sigma = (\mathcal{F}; \mathcal{P})$.

▶ **Definition 6** (Terms and Formulas). *We define terms $t : \mathsf{tm}$ and formulas $\varphi : \mathsf{fm}$ inductively.*

$$s, t : \mathsf{tm} ::= x_n \mid f\,\vec{v} \qquad (n : \mathbb{N},\ f : \mathcal{F},\ \vec{v} : \mathsf{tm}^{|f|})$$

$$\alpha, \beta : \mathsf{fm} ::= P\,\vec{v} \mid \alpha \to \beta \mid \alpha \wedge \beta \mid \alpha \vee \beta \mid \forall\,\alpha \mid \exists\,\beta \qquad (P : \mathcal{P},\ \vec{v} : \mathsf{tm}^{|P|}).$$

Where $|f|$ and $|P|$ are the arities of the function symbol f and predicate symbol P, respectively.

We use de Bruijn indexing to formalize the binding of variables to quantifiers. This means that the variable x_n at some position in a formula is *bound* to the n-th quantifier preceding this variable in the syntax tree of the formula. If there is no quantifier binding the variable, it is said to be *free*.

▶ **Definition 7** (Substitution). *Given a variable assignment $\sigma : \mathbb{N} \to \mathsf{tm}$ we recursively define* substitution *on terms by $x_k[\sigma] := \sigma\, k$, $f\,\vec{v} := f(\vec{v}[\sigma])$ and extend this definition to formulas by*

$$\bot[\sigma] := \bot \qquad (P\,\vec{v})[\sigma] := P\,(\vec{v}[\sigma]) \qquad (\alpha \mathbin{\dot\Box} \beta)[\sigma] := \alpha[\sigma] \mathbin{\Box} \beta[\sigma] \qquad (\dot\nabla\, \varphi)[\sigma] := \nabla(\varphi[0; \lambda x.\uparrow(\sigma x)])$$

where $\dot{\Box}$ is any logical connective and $\dot{\nabla}$ any quantifier from the signature. The expression $x;\sigma$ is defined by $(x;\sigma)\,0 := x$, $(x;\sigma)(Sn) := \sigma\,n$ and is simply appending x as the first element of $\sigma : \mathbb{N} \to \mathsf{tm}$. By $\uparrow$ we designate the substitution $\lambda k.\, x_{Sk}$ shifting all variable indices up by one.

▶ **Definition 8** (Natural Deduction). *We define intuitionistic natural deduction $\vdash : \mathsf{List}(\mathsf{fm}) \to \mathsf{fm} \to \mathbb{P}$ inductively by the rules*

$$\frac{\varphi \in \Gamma}{\Gamma \vdash \varphi} \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash \varphi} \qquad \frac{\Gamma, \varphi \vdash \psi}{\Gamma \vdash \varphi \to \psi} \qquad \frac{\Gamma \vdash \varphi \to \psi \quad \Gamma \vdash \varphi}{\Gamma \vdash \varphi}$$

$$\frac{\Gamma \vdash \varphi \quad \Gamma \vdash \psi}{\Gamma \vdash \varphi \wedge \psi} \qquad \frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \varphi} \qquad \frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \psi}$$

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \varphi \vee \psi} \qquad \frac{\Gamma \vdash \psi}{\Gamma \vdash \varphi \vee \psi} \qquad \frac{\Gamma \vdash \varphi \vee \psi \quad \Gamma, \varphi \vdash \theta \quad \Gamma, \psi \vdash \theta}{\Gamma \vdash \theta}$$

$$\frac{\Gamma[\uparrow] \vdash \varphi}{\Gamma \vdash \forall \varphi} \qquad \frac{\Gamma \vdash \forall \varphi}{\Gamma \vdash \varphi[t]} \qquad \frac{\Gamma \vdash \varphi[t]}{\Gamma \vdash \exists \varphi} \qquad \frac{\Gamma \vdash \exists \varphi \quad \Gamma[\uparrow], \varphi \vdash \psi[\uparrow]}{\Gamma \vdash \psi}$$

where we get the classical variant by adding Peirce's rule

$$\frac{}{\Gamma \vdash_c ((\varphi \to \psi) \to \varphi) \to \varphi}$$

We write $\vdash$ for intuitionistic natural deduction and $\vdash_c$ for the classical one.

▶ **Definition 9** (Tarski Semantics). *A* model $\mathcal{M}$ *consists of a type D designating its domain together with functions $f^{\mathcal{M}} : D^{|f|} \to D$ and $P^{\mathcal{M}} : D^{|P|} \to \mathbb{P}$ for all symbols f and P. Abusing notation we will also use $\mathcal{M}$ to refer to the domain. In this context, functions $\rho : \mathbb{N} \to \mathcal{M}$ will be called environments and are used as variable assignments to recursively give interpretations to terms:*

$$\hat{\rho}\, x_k := \rho\, k \qquad \hat{\rho}\,(f\,\vec{v}) := f^{\mathcal{M}}(\hat{\rho}\,\vec{v}) \qquad (v : \mathsf{tm}^n).$$

This is then extended to formulas:

$$\begin{aligned} \mathcal{M} \vDash_\rho P\,\vec{v} &:= P^{\mathcal{M}}(\hat{\rho}\,\vec{v}) & \mathcal{M} \vDash_\rho \alpha \to \beta &:= \mathcal{M} \vDash_\rho \alpha \to \mathcal{M} \vDash_\rho \beta \\ \mathcal{M} \vDash_\rho \alpha \wedge \beta &:= \mathcal{M} \vDash_\rho \alpha \wedge \mathcal{M} \vDash_\rho \beta & \mathcal{M} \vDash_\rho \alpha \vee \beta &:= \mathcal{M} \vDash_\rho \alpha \vee \mathcal{M} \vDash_\rho \beta \\ \mathcal{M} \vDash_\rho \forall \alpha &:= \forall x : D.\ \mathcal{M} \vDash_{x;\rho} \alpha & \mathcal{M} \vDash_\rho \exists \alpha &:= \exists x : D.\ \mathcal{M} \vDash_{x;\rho} \alpha \end{aligned}$$

We then say that a formula φ holds in the model $\mathcal{M}$ *and write $\mathcal{M} \vDash \varphi$ if for every environment ρ we have $\mathcal{M} \vDash_\rho \varphi$. We extend this notation to theories $\mathcal{T} : \mathsf{fm} \to \mathbb{P}$ by writing $\mathcal{M} \vDash \mathcal{T}$ iff $\forall \varphi.\, \mathcal{T}\,\varphi \to \mathcal{M} \vDash \varphi$.*

From the next section onwards, we will no longer explicitly write formulas with deBruijn indices, but will use the conventional notation which uses named variables.

3 Axiomatization of Peano Arithmetic

As a first-order theory, PA has a signature consisting of symbols for the constant zero, the successor function, addition, multiplication and equality:

$$(\mathcal{F}_{\mathsf{PA}}; \mathcal{P}_{\mathsf{PA}}) := (0,\ S, +, \times; =).$$

The finite core of PA axioms consists of statements characterizing the successor function:

$$\text{Disjointness}: \forall x.\, Sx = 0 \to \bot \qquad \text{Injectivity}: \forall xy.\, Sx = Sy \to x = y$$

as well as addition and multiplication:

$$+\text{-base}: \forall x.\, 0 + x = x \qquad +\text{-recursion}: \forall xy.\, (Sx) + y = S(x + y)$$
$$\times\text{-base}: \forall x.\, 0 \times x = 0 \qquad \times\text{-recursion}: \forall xy.\, (Sx) \times y = y + x \times y$$

We then get the full (and infinite) axiomatization of PA by adding the axiom scheme of induction, which in our meta-theory is a type-theoretic function on formulas:

$$\lambda\varphi.\, \varphi[0] \to (\forall x.\, \varphi[x] \to \varphi[Sx]) \to \forall x.\, \varphi[x]$$

If instead of the induction scheme we add the axiom $\forall x.\, x = 0 \lor \exists y.\, x = Sy$, we get the theory Q known as *Robinson arithmetic*. We also add congruence axioms for equality:

$$\text{Reflexivity}: \forall x.\, x = x$$
$$\text{Symmetry}: \forall xy.\, x = y \to y = x$$
$$\text{Transitivity}: \forall xyz.\, x = y \to y = z \to x = z$$
$$S\text{-equality}: \forall xy.\, x = y \to Sx = Sy$$
$$+\text{-equality}: \forall xyuv.\, x = u \to y = v \to x + y = u + v$$
$$\times\text{-equality}: \forall xyuv.\, x = u \to y = v \to x \times y = u \times v.$$

Semantically, we treat equality different compared to other predicate symbols. Instead of being interpreted as a predicate $=^{\mathcal{M}}: \mathcal{M}^2 \to \mathbb{P}$, it will be interpreted as equality in $\mathcal{M}$. This means we are only considering extensional PA models.

▶ **Definition 10.** *We recursively define a function* $\overline{\cdot}: \mathbb{N} \to \mathsf{tm}$ *by* $\overline{0} := 0$ *and* $\overline{n+1} := S\overline{n}$*, giving every natural number a representation as a term. Any term t which is of the form $\overline{n}$ will be called* numeral*.*

We furthermore use notations for expressing *less than* $x < y := \exists k.\, S(x + k) = y$, *less or equal* $x \le y := \exists k.\, x + k = y$ and for *divisibility* $x \mid y := \exists k.\, x \times k = y$.

The formulas of PA can be classified in a hierarchy based on the their computational properties. We will only consider two levels of this hierarchy, namely Δ_0 and Σ_1 formulas:

▶ **Definition 11.** *We will say that a formula φ is Δ_0 if*

- *for every substitution σ which makes $\varphi[\sigma]$ closed, we have* $\mathsf{Q} \vdash \varphi[\sigma] + \mathsf{Q} \vdash \neg\varphi[\sigma]$
- *and* $\mathsf{HA} \vdash \varphi \lor \neg\varphi$.

We will say that a formula is $\exists_1$ if it is of the form $\exists\, \varphi_0$, where φ_0 is Δ_0 and $\exists_n$ if there are n existential quantifiers in front of φ_0. If a formula is $\exists_n$ for any n, it is also called Σ_1.

Note that above definition of Δ_0 formulas is not the syntactical definition of Δ_0 formulas. Instead it singles out the properties which will suffice for the development.

▶ **Lemma 12** (Δ_0-Absoluteness)**.** *Let* $\mathcal{M} \vDash \mathsf{PA}$ *and φ be any closed Δ_0 formula, then* $\mathbb{N} \vDash \varphi \to \mathcal{M} \vDash \varphi$.

Proof. By Definition 11 we have either $\mathsf{PA} \vdash \varphi$ or $\mathsf{PA} \vdash \neg\varphi$. Since $\mathbb{N} \vDash \varphi$ we must have $\mathsf{PA} \vdash \varphi$ and therefore $\mathcal{M} \vDash \varphi$ by soundness. ◀

▶ **Lemma 13.** *For any unary Δ_0 formula $\varphi(x)$ we have* $\mathbb{N} \vDash \exists x.\, \varphi(x) \leftrightarrow \mathsf{PA} \vdash \exists x.\, \varphi(x)$.

Proof. The assumption $\mathbb{N} \vDash \exists x.\, \varphi(x)$ gives us $n : \mathbb{N}$ with $\mathbb{N} \vDash \varphi(\overline{n})$. By Lemma 12 we then have $\mathsf{PA} \vdash \varphi(\overline{n})$, which in turn shows $\mathsf{PA} \vdash \exists x.\, \varphi(x)$. The converse follows by soundness. ◀

▶ **Corollary 14.** *Let* $\mathcal{M} \vDash \mathsf{PA}$ *and φ be any closed $\exists_1$ formula, then* $\mathbb{N} \vDash \varphi \to \mathcal{M} \vDash \varphi$.

4 Standard and Non-standard Models of PA

Starting this section, $\mathcal{M}$ will always designate a PA model.

▶ **Proposition 15.** *We recursively define a function* $\nu : \mathbb{N} \to \mathcal{M}$ *by* $\nu\, 0 := 0^{\mathcal{M}}$ *and* $\nu\,(n+1) := S^{\mathcal{M}}(\nu\ n)$*. We define the predicate* std $:= \lambda e.\, \exists n. \overline{n} = e$ *and refer to* e *as a* standard number *if* std e *and* non-standard *if* $\neg$ std e*. We further have*
(1) $\hat{\rho}\,\overline{n} = \nu\, n$ *for any* $n : \mathbb{N}$ *and environment* $\rho : \mathbb{N} \to \mathcal{M}$*.*
(2) ν *is an injective homomorphism and therefore an* embedding *of* $\mathbb{N}$ *into* $\mathcal{M}$*.*
We take both facts as a justification to abuse notation and also write $\overline{n}$ *for* $\nu\, n$*.*

Usually we would have to write $0^{\mathcal{M}}, S^{\mathcal{M}}, +^{\mathcal{M}}, \times^{\mathcal{M}}, =^{\mathcal{M}}$ for the interpretations of the respective symbols in a model $\mathcal{M}$. For better readability we will however take the freedom to overload the symbols $0, S, +, \cdot, =$ to also refer to these interpretations.

▶ **Definition 16.** $\mathcal{M}$ *is called a* standard model *if there is a bijective homomorphism* $\varphi : \mathbb{N} \to \mathcal{M}$*. We will accordingly write* $\mathcal{M} \cong \mathbb{N}$ *if this is the case.*

We can show that ν is essentially the only homomorphism from $\mathbb{N}$ to $\mathcal{M}$ we need to worry about, since it is unique up to functional extensionality:

▶ **Lemma 17.** *Let* $\varphi : \mathbb{N} \to \mathcal{M}$ *be a homomorphism, then* $\forall x : \mathbb{N}.\, \varphi\, x = \nu\, x$*.*

Proof. By induction on x and using the fact that both are homomorphisms. ◀

We now have two equivalent ways to express standardness of a model.

▶ **Lemma 18.** $\mathcal{M} \cong \mathbb{N} \iff \forall e : \mathcal{M}.\, \mathsf{std}\, e$*.*

Proof. Given $\mathcal{M} \cong \mathbb{N}$, there is an isomorphism $\varphi : \mathbb{N} \to \mathcal{M}$. Since φ is surjective, Lemma 17 implies that ν must also be surjective. For the converse: if ν is surjective, it is an isomorphism since it is injective by Proposition 15. ◀

Having seen that every model contains a unique embedding of $\mathbb{N}$, one may wonder whether there is a formula φ which could define and pick out precisely the standard numbers in $\mathcal{M}$. Lemma 19 gives an answer to this question:

▶ **Lemma 19.** *There is a unary formula* $\varphi(x)$ *with* $\forall e : \mathcal{M}. \big(\mathsf{std}\, e \leftrightarrow \mathcal{M} \vDash \varphi(e)\big)$ *if and only if* $\mathcal{M} \cong \mathbb{N}$*.*

Proof. Given a formula φ with the stated property, we certainly have $\mathcal{M} \vDash \varphi(\overline{0})$ since $\overline{0}$ is a standard number, and clearly $\mathcal{M} \vDash \varphi(x) \implies \mathsf{std}\, x \implies \mathsf{std}\,(Sx) \implies \mathcal{M} \vDash \varphi(Sx)$. Thus by induction in the model, we have $\mathcal{M} \vDash \forall x.\, \varphi(x)$, which is equivalent to $\forall e : \mathcal{M}.\, \mathsf{std}\, e$. The converse is shown by the formula $x = x$. ◀

We now turn our attention to models which are not isomorphic to $\mathbb{N}$.

▶ **Fact 20.** *For any* $e : \mathcal{M}$*, we have* $\neg\,\mathsf{std}\, e$ *iff* $\forall n : \mathbb{N}.\ e > \overline{n}$*.*

▶ **Definition 21.** *Founded on the result of Fact 20 we write* $e > \mathbb{N}$ *iff* $\neg\,\mathsf{std}\, e$ *and call the model* $\mathcal{M}$
- non-standard *and write* $\mathcal{M} > \mathbb{N}$ *iff there is* $e : \mathcal{M}$ *such that* $e > \mathbb{N}$*,*
- not standard *and write* $\mathcal{M} \ncong \mathbb{N}$ *iff* $\neg(\mathcal{M} \cong \mathbb{N})$*.*

We will also use the notation $e : \mathcal{M} > \mathbb{N}$ *to express the existence of a non-standard element* e *in* $\mathcal{M}$*.*

Of course we have $\mathcal{M} > \mathbb{N} \to \mathcal{M} \not\cong \mathbb{N}$, but the converse implication does not hold constructively in general, so the distinction becomes meaningful.

▶ **Lemma 22** (Overspill). *If $\mathcal{M} \not\cong \mathbb{N}$ and $\varphi(x)$ is a unary formula with $\mathcal{M} \vDash \varphi(\overline{n})$ for every $n : \mathbb{N}$ then*
(1) $\neg \forall e : \mathcal{M}. \mathcal{M} \vDash \varphi(e) \to \mathsf{std}\, e$
(2) $\mathsf{stable\ std} \to \neg\neg \exists\, e > \mathbb{N}. \mathcal{M} \vDash \varphi(e)$
(3) $\mathsf{DNE} \to \exists\, e > \mathbb{N}. \mathcal{M} \vDash \varphi(e)$.

Proof. (1) Assuming $\forall e : \mathcal{M}.\ \mathcal{M} \vDash \varphi(e) \to \mathsf{std}\, e$ and combining it with our assumption that φ holds on all numerals, Lemma 19 implies $\mathcal{M} \cong \mathbb{N}$, giving us a contradiction. For (2) note that we constructively have the implication

$$\big(\neg \exists e : \mathcal{M}. \neg \mathsf{std}\, e \wedge \mathcal{M} \vDash \varphi(e)\big) \implies \forall e : \mathcal{M}. \mathcal{M} \vDash \varphi(e) \to \neg\neg\, \mathsf{std}\, e$$

and by using the stability of $\mathsf{std}\cdot$ we therefore get a contradiction in the same way as in (1). Statement (3) immediately follows from (2). ◀

In Section 6 we will see a first usage of Overspill to encode predicates by non-standard elements.

5 Church's Thesis

The constructive Church's thesis (CT) [15, 36], states that every function $\mathbb{N} \to \mathbb{N}$ has a representation in a previously chosen, concrete model of computation. In the constructive type theory that we have chosen, it is possible to consistently add CT as an axiom [38, 31]. Given we are treating computability in the context of PA, we choose a version of CT which uses a model of computation based on representing functions by formulas in the language of PA.

▶ **Axiom 23** ($\mathsf{CT_Q}$). *For every function $f : \mathbb{N} \to \mathbb{N}$ there is a binary $\exists_1$ formula $\varphi_f(x, y)$ such that for every $n : \mathbb{N}$ we have $\mathsf{Q} \vdash \forall y.\, \varphi_f(\overline{n}, y) \leftrightarrow \overline{f n} = y$.*

This formulation takes its justification from the standard result establishing the representability of μ-recursive functions by Σ_1 formulae in Q [29, 22], combined with the DPRM theorem [6, 7, 18, 16] to get the desired $\exists_1$ formula. We can use $\mathsf{CT_Q}$ to establish the representability of decidable and enumerable predicates in Q [26].

▶ **Definition 24.** *Let $p : \mathbb{N} \to \mathbb{P}$, then we call p* weakly representable *by $\varphi_p(x)$ if $\forall n : \mathbb{N}.\, p\, n \leftrightarrow \mathsf{Q} \vdash \varphi_p(\overline{n})$, and* strongly representable *if $p\, n \to \mathsf{Q} \vdash \varphi_p(\overline{n})$ and $\neg p\, n \to \mathsf{Q} \vdash \neg\varphi_p(\overline{n})$ for every $n : \mathbb{N}$.*

▶ **Lemma 25** (Representability Theorem (RT)). *Assume $\mathsf{CT_Q}$, and let $p : \mathbb{N} \to \mathbb{P}$ be given.*
- *If p is decidable, it is strongly representable by a unary $\exists_1$ formula.*
- *If p is enumerable, it is weakly representable by a unary $\exists_2$ formula.*

Proof. If p is decidable there is a function $f : \mathbb{N} \to \mathbb{N}$ such that $\forall x : \mathbb{N}.\, p\, x \leftrightarrow f x = 0$ and by $\mathsf{CT_Q}$ there is a binary $\exists_1$ formula $\varphi_f(x, y)$ representing f. We then define $\varphi_p(x) := \varphi_f(x, \overline{0})$ and get

$$p\, n \implies f n = 0 \implies \mathsf{Q} \vdash \overline{f n} = \overline{0} \implies \mathsf{Q} \vdash \varphi_f(\overline{n}, \overline{0}) \implies \mathsf{Q} \vdash \varphi_p(\overline{n})$$
$$\neg p\, n \implies f n \neq 0 \implies \mathsf{Q} \vdash \neg(\overline{f n} = \overline{0}) \implies \mathsf{Q} \vdash \neg\varphi_f(\overline{n}, \overline{0}) \implies \mathsf{Q} \vdash \neg\varphi_p(\overline{n})$$

Which shows that p is strongly representable.

If p is enumerable there is a function $f : \mathbb{N} \to \mathbb{N}$ such that $\forall x : \mathbb{N}.\, p\,x \leftrightarrow \exists n.\, f n = Sx$ and by CT_Q there is a binary $\exists_1$ formula $\varphi_f(x, y)$ representing f. We then define $\varphi_p(x) := \exists\, n.\, \varphi_f(n, Sx)$ giving us

$$\begin{aligned}\mathsf{Q} \vdash \varphi_p(\overline{x}) &\iff \mathsf{Q} \vdash \exists\, n.\, \varphi_f(n, S\overline{x}) \iff \exists n\!:\!\mathbb{N}.\, \mathsf{Q} \vdash \varphi_f(\overline{n}, S\overline{x})\\ &\iff \exists n\!:\!\mathbb{N}.\, \mathsf{Q} \vdash \overline{fn} = S\overline{x} \iff \exists n\!:\!\mathbb{N}.\, fn = Sx \iff p\,x\end{aligned}$$

This shows that p is weakly representable by a $\exists_2$ formula. ◀

6 Coding Predicates

There is a standard way in which finite sets of natural numbers can be encoded by a single natural number. This is readily established in $\mathbb{N}$, and can then be carried over with relative ease to any PA model. Overspill has interesting consequences when it comes to this encoding, as for models $\mathcal{M} \not\cong \mathbb{N}$, it allows the potential encoding of any predicate $p\!:\!\mathbb{N} \to \mathbb{P}$.

For the natural number version of the encoding, we only need some injective function $\pi\!:\!\mathbb{N} \to \mathbb{N}$ whose image consists only of prime numbers.

▶ **Lemma 26** (Finite Coding in $\mathbb{N}$). *Given any predicate $p\!:\!\mathbb{N} \to \mathbb{P}$ and bound $n\!:\!\mathbb{N}$, we have*

$$\neg\neg\, \exists c\!:\!\mathbb{N}\ \forall u\!:\!\mathbb{N}.\big(u < n \to (p\,u \leftrightarrow \pi_u \mid c)\big) \wedge \big(\pi_u \mid c \to u < n\big)$$

i.e. up to the specified bound n, the code c *is divisible by the prime π_u if and only iff p holds on $u\!:\!\mathbb{N}$. The second part of the conjunction assures that no primes bigger then π_n are present in the code. Note that if p is definite, we can drop the $\neg\neg$.*

Proof. We do a proof by induction on n. For $n = 0$ we can choose $c = 1$. For the induction step we first note that $\neg\neg(p\,n \vee \neg p\,n)$ is constructively provable and that the induction hypothesis as well as the goal come with double negations at the front. Using $p\,n \vee \neg p\,n$ we can now consider two cases. If $\neg p\,n$ we can simply take the code c given by the induction hypothesis. If $p\,n$, we set the new code to be $c \cdot \pi_n$. In both cases the separate parts of the conjunction are checked by making use of the fact that π is an injective prime function. ◀

To formulate this result in a generic model $\mathcal{M} \vDash \mathsf{PA}$, we require an object level representation of the prime function. We can easily get such a representation, by usage of CT_Q:

▶ **Fact 27.** *There is a binary formula Π representing the injective prime function π in* Q.

This now makes it possible to express "π_u divides c" by $\exists\, p.\ \Pi(u, p) \wedge p \mid c$, where we will abuse notation and simply write $\Pi(u) \mid c$ for this. With Π then, we can take the coding result established for $\mathbb{N}$ and use it to show a similar result in any model of PA.

▶ **Lemma 28** (Finite Coding in $\mathcal{M} \vDash \mathsf{PA}$). *For any binary formula $\alpha(x, y)$ and $n\!:\!\mathbb{N}$ we have*

$$\mathcal{M} \vDash \forall\, b\, \neg\neg\, \exists\, c\, \forall\, u < \overline{n}.\ \alpha(u, b) \leftrightarrow \Pi(u) \mid c.$$

If $\mathcal{M} \vDash \alpha(\overline{u}, b)$ is definite for every $u\!:\!\mathbb{N}$, $b\!:\!\mathcal{M}$, we can drop the $\neg\neg$ in the above.

Proof. Let $b : \mathcal{M}$, then define the predicate $p := \lambda u : \mathbb{N}.\, \mathcal{M} \vDash \alpha(\overline{u}, b)$. Then Lemma 26 potentially gives us a code $a\!:\!\mathbb{N}$ for p up to the bound n. It now suffices to show that the actual existence of $a\!:\!\mathbb{N}$ already implies

$$\mathcal{M} \vDash \exists\, c\, \forall\, u < \overline{n}.\ \alpha(u, b) \leftrightarrow \Pi(u) \mid c.$$

And indeed, we can verify that $c = \overline{a}$ shows the existential claim: given $u{:}\mathcal{M}$ with $\mathcal{M} \vDash u < \overline{n}$ we can conclude that u must be a standard number $\overline{u}$. We then have the equivalences

$$\mathcal{M} \vDash \alpha(\overline{u}, b) \iff p\,u \iff \pi_u \mid a \iff \mathcal{M} \vDash \Pi(\overline{u}) \mid \overline{a}$$

since a is coding p and Π is representing π. ◀

▶ **Lemma 29.** *If* std $\cdot$ *is stable,* $\mathcal{M} \not\cong \mathbb{N}$ *and* $\alpha(x)$ *a unary formula, we have*

$$\neg\neg\,\exists c{:}\mathcal{M}\ \forall u{:}\mathbb{N}.\ \ \mathcal{M} \vDash \alpha(\overline{u}) \leftrightarrow \Pi(\overline{u}) \mid c.$$

Proof. Using Lemma 28 for the present case where α is unary, we get

$$\mathcal{M} \vDash \neg\neg\,\exists c\,\forall u < \overline{n}.\ \alpha(u) \leftrightarrow \Pi(u) \mid c$$

for every $n{:}\mathbb{N}$, so by Lemma 22 (Overspill) we get

$$\begin{aligned} &\neg\neg\,\exists e > \mathbb{N}.\ \mathcal{M} \vDash \neg\neg\,\exists c\,\forall u < e.\ \alpha(u) \leftrightarrow \Pi(u) \mid c \\ \Longrightarrow\ &\neg\neg\,\exists c{:}\mathcal{M}\,\forall u{:}\mathbb{N}.\ \mathcal{M} \vDash \alpha(\overline{u}) \leftrightarrow \Pi(u) \mid c. \end{aligned}$$

Where we used that given $\forall u{:}\mathcal{M} < e.\,(\ldots)$ we can show $\forall u{:}\mathbb{N}.\,(\ldots)$, since we have $e > \mathbb{N}$ and therefore $\overline{u} < e$ for any $u{:}\mathbb{N}$ by Fact 20. ◀

▶ **Lemma 30.** *If* std $\cdot$ *is stable,* $\mathcal{M} \not\cong \mathbb{N}$ *and* $\mathcal{M} \vDash \alpha(\overline{u}, b)$ *is definite for every* $b{:}\mathcal{M}$*,* $u{:}\mathbb{N}$ *, then we have*

$$\neg\neg\,\forall b{:}\mathcal{M}\ \exists c{:}\mathcal{M}\ \forall u{:}\mathbb{N}.\ \ \mathcal{M} \vDash \alpha(\overline{u}, b) \leftrightarrow \Pi(\overline{u}) \mid c.$$

Proof. Similar to the proof of Lemma 29, but we make use of the definiteness to get the stronger result out of Lemma 28 and then use Overspill to conclude. ◀

7 Tennenbaum's Theorem

We will now present several proofs of Tennenbaum's theorem, differing in the assumptions they make and the strength of their results. All of the proofs have in common that they start by the assumption $\mathcal{M} > \mathbb{N}$ to then make use of the coding lemma to encode a particular formula by an element of the model. In Section 7.1 we will assume enumerability of the model, enabling a direct diagonal argument. This proof–idea can be found in [3]. In Section 7.2 we look at the proof approach that is most prominently found in the literature [30, 12] and uses the existence of recursively inseparable sets. A refinement of this proof was proposed in [17] and circumvents the usage of Overspill. In our constructive setting, this will lead to a perceivable difference when it comes to the strength of the result. Lastly we look at the consequences of Tennenbaum's theorem for HA, once the underlying semantics is made constructive.

7.1 Via a Diagonal Argument

We start by noting that every PA model can prove the most basic fact about divisibility.

▶ **Lemma 31** (Euclidean Lemma)**.** *Given* $e, d{:}\mathcal{M}$ *we have*

$$\mathcal{M} \vDash \exists r\,q.\, e = q \cdot d + r\ \wedge\ (0 < d \to r < d)$$

and the uniqueness property telling us that if $r_1, r_2 < d$ *then* $q_1 \cdot d + r_1 = q_2 \cdot d + r_2$ *implies* $q_1 = q_2$ *and* $r_1 = r_2$*.*

Proof. For Euclid's lemma, there is a standard proof by induction on $e : \mathcal{M}$. The uniqueness claim requires some results about the order relation $<$. ◀

▶ **Lemma 32.** *If $\mathcal{M} \vDash \mathsf{PA}$ is enumerable and discrete, then $\lambda n : \mathbb{N}\, d : \mathcal{M}.\, \mathcal{M} \vDash \overline{n} \mid d$ has a decider.*

Proof. Let $n : \mathbb{N}$ and $d : \mathcal{M}$ be given. By the Euclidean Lemma 31 we have $\exists q, r : \mathcal{M}.\, e = q \cdot d + r$. This existence is propositional, so presently we cannot use it to give a decision for $e \mid d$. Since $\mathcal{M}$ is enumerable, there is a surjective function $g : \mathbb{N} \to \mathcal{M}$ and the above existence therefore shows $\exists q, r : \mathbb{N}.\, e = (g\, q) \cdot d + (g\, r)$. Since equality is decidable in $\mathcal{M}$ and $\mathbb{N}^2$ is witnessing, we get $\Sigma q, r : \mathbb{N}.\, e = (g\, q) \cdot d + (g\, r)$, giving us computational access to r, now allowing us to construct the decision. By the uniqueness part of Lemma 31 we have $g\, r = 0 \leftrightarrow e \mid d$, so the decidability of $e \mid d$ is entailed by the decidability of $g\, r = 0$. ◀

▶ **Lemma 33.**
- *If $\mathsf{std}\,\cdot$ is stable, then so is $\mathcal{M} \cong \mathbb{N}$.*
- *Assuming MP and discreteness of $\mathcal{M}$, then $\mathsf{std}\,\cdot$ is stable.*

Proof. The first statement is trivial by Lemma 18. For the second, recall that $\mathsf{std}\, e$ stands for $\exists n : \mathbb{N}.\, \overline{n} = e$. Since $\overline{n} = e$ in $\mathcal{M}$ is decidable, the stability follows from Fact 5. ◀

▶ **Theorem 34.** *Assuming MP, if $\mathcal{M} \vDash \mathsf{PA}$ is enumerable and discrete, then $\mathcal{M} \cong \mathbb{N}$.*

Proof. By Lemma 33 our goal is equivalent to $\neg\neg \mathcal{M} \cong \mathbb{N}$. So assume $\mathcal{M} \ncong \mathbb{N}$ and try to derive $\bot$. Given the enumerability, there is a surjective function $g : \mathbb{N} \to \mathcal{M}$. We use this to define the predicate $p := \lambda n : \mathbb{N}.\, \neg\, \mathcal{M} \vDash \overline{\pi_n} \mid g\, n$, which has a decider by Lemma 32. By RT then, there is a formula φ_p strongly representing p. Under the given assumptions, we can use the coding Lemma 29, giving us a code $c_p : \mathcal{M}$ such that $\forall u : \mathbb{N}.\, \mathcal{M} \vDash \varphi_p(\overline{u}) \leftrightarrow \Pi(\overline{u}) \mid c_p$. By surjectivity of g there is $c : \mathbb{N}$ with $g\, c = c_p$, which gives us

$$\neg\, \mathcal{M} \vDash \overline{\pi_c} \mid g\, c \implies \mathsf{Q} \vdash \varphi_p(\overline{c}) \implies \mathcal{M} \vDash \varphi_p(\overline{c}) \implies \mathcal{M} \vDash \Pi(\overline{c}) \mid g\, c$$
$$\neg\neg\, \mathcal{M} \vDash \overline{\pi_c} \mid g\, c \implies \mathsf{Q} \vdash \neg\, \varphi_p(\overline{c}) \implies \neg\, \mathcal{M} \vDash \varphi_p(\overline{c}) \implies \neg\, \mathcal{M} \vDash \Pi(\overline{c}) \mid g\, c$$

Since $\mathcal{M} \vDash \Pi(\overline{u}) \mid g\, c \leftrightarrow \overline{\pi_u} \mid g\, c$, this entails the contradictory statement $p\, c \Longleftrightarrow \neg p\, c$. ◀

7.2 Via Inseparable Predicates

The usual proof of Tennenbaum's theorem [12, 30] uses the existence of recursively inseparable sets and non-standard coding to establish the existence of a non-recursive set. If we then were to again assume enumerability and discreteness of $\mathcal{M}$, we could easily reach the same conclusion as in Theorem 34. In the following however, we want to highlight that the proof which uses inseparable sets allows for a characterization of $\mathcal{M} \cong \mathbb{N}$ only making reference to the decidability of divisibility by numerals:

▶ **Definition 35.** *For $d : \mathcal{M}$ define the predicate $\overline{\cdot} \mid d := \lambda n : \mathbb{N}.\, \mathcal{M} \vDash \overline{n} \mid d$.*

So in particular we will not assume enumerability or discreteness of $\mathcal{M}$.

▶ **Definition 36.** *A pair $A, B : \mathbb{N} \to \mathbb{P}$ of predicates is called* inseparable *iff*
(1) *they are disjoint, meaning $\forall n : \mathbb{N}.\, \neg(A\, n \land B\, n)$*
(2) *there is no decidable $D : \mathbb{N} \to \mathbb{P}$ which includes A i.e. $\forall n : \mathbb{N}.\, A\, n \to D\, n$ and is disjoint from B i.e. $\forall n : \mathbb{N}.\, \neg(B\, n \land D\, n)$.*

▶ **Lemma 37.** *There are inseparable enumerable predicates $A, B : \mathbb{N} \to \mathbb{P}$.*

Proof. We use an enumeration $\Phi_n : \mathsf{fm}$ of formulas to define disjoint predicates $A := \lambda n : \mathbb{N}.\, \mathsf{Q} \vdash \neg\, \Phi_n(\overline{n})$ and $B := \lambda n{:}\mathbb{N}.\, \mathsf{Q} \vdash \Phi_n(\overline{n})$. Since proofs over Q can be enumerated, A and B are enumerable. Assume we are given a decidable predicate D which includes A and is disjoint from B. Using RT and the enumeration, there is $d{:}\mathbb{N}$ such that Φ_d strongly represents D. This gives us $D\,d \implies \mathsf{Q} \vdash \Phi_d(\overline{d}) \implies B\,d$, contradicting the disjointness of B and D, therefore showing $\neg D\,d$. Furthermore, representability gives us $\neg D\,d \implies \mathsf{Q} \vdash \neg\Phi_d(\overline{d}) \implies A\,d$ and since A is included in D, this shows $\neg D\,d \implies D\,d$. Overall this gives us a contradiction. ◀

▶ **Corollary 38.** *There is a pair $\alpha(z), \beta(z)$ of unary $\exists_2$ formulas such that $A{:=}\lambda n{:}\mathbb{N}.\, \mathsf{Q} \vdash \alpha(\overline{n})$ and $B := \lambda n{:}\mathbb{N}.\, \mathsf{Q} \vdash \beta(\overline{n})$ are inseparable and enumerable.*

Proof. We get the desired formulas by using the weak representability of Lemma 25 on the predicates given by Lemma 37. ◀

▶ **Lemma 39.** *Assuming stability of* $\mathsf{std}\,\cdot$ *and $\mathcal{M} \ncong \mathbb{N}$, then $\neg\neg\, \exists d{:}\mathcal{M}.\, \neg\mathsf{Dec}(\overline{\cdot} \mid d)$.*

Proof. By Corollary 38 there are inseparable formulas $\exists x, y.\, \alpha_0(x, y, z)$ and $\exists x, y.\, \beta_0(x, y, \overline{n})$ such that α_0, β_0 are Δ_0. Since they are disjoint, we have:

$$\mathbb{N} \vDash \forall\, x\, y\, u\, v\, z < \overline{n}.\, \neg(\alpha_0(x, y, z) \wedge \beta_0(u, v, z))$$

for every bound $n{:}\mathbb{N}$. By Lemma 12 we then get

$$\mathcal{M} \vDash \forall\, x\, y\, u\, v\, z < \overline{n}.\, \neg(\alpha_0(x, y, z) \wedge \beta_0(u, v, z))$$

and using Overspill we therefore potentially have $e{:}\mathcal{M}$ with

$$\mathcal{M} \vDash \forall\, x\, y\, u\, v\, z < e.\, \neg(\alpha_0(x, y, z) \wedge \beta_0(u, v, z))$$

showing the disjointness of α_0, β_0 when everything is bounded by e. We now define the predicate $X := \lambda n{:}\mathbb{N}.\, \mathcal{M} \vDash \exists\, x, y < e.\, \alpha_0(x, y, \overline{n})$ and note that

- If $\mathsf{Q} \vdash \exists\, x, y.\, \alpha_0(x, y, \overline{n})$ there are $m_1, m_2 {:} \mathbb{N}$ with $\mathbb{N} \vDash \alpha_0(\overline{m_1}, \overline{m_2}, \overline{n})$ and $\mathcal{M} \vDash \alpha_0(\overline{m_1}, \overline{m_2}, \overline{n})$ by Lemma 12. We therefore get $X n$.
- Assume that $X n \wedge \mathsf{Q} \vdash \exists\, x, y.\, \beta_0(x, y, \overline{n})$. Then similarly to above, there are $m_1, m_2 {:} \mathbb{N}$ with $\mathcal{M} \vDash \beta_0(\overline{m_1}, \overline{m_2}, \overline{n})$, showing $\mathcal{M} \vDash \exists\, x, y < e.\, \beta_0(x, y, \overline{n})$. Together with $X n$ this contradicts the disjointness of α_0, β_0 under the bound e.

Due to the inseparability of the given formulas, this shows that X cannot be decidable and by Lemma 30 there is now potentially a code $d{:}\mathcal{M}$ with $X n \Leftrightarrow \mathcal{M} \vDash \overline{\pi_n} \mid d$. ◀

▶ **Fact 40.** *For every $e{:}\mathcal{M}$ we have* $\mathsf{std}\, e \to \mathsf{Dec}(\overline{\cdot} \mid e)$.

▶ **Corollary 41.** *Given* MP *and discrete $\mathcal{M}$, we have $\mathcal{M} \cong \mathbb{N}$ iff $\forall d{:}\mathcal{M}.\, \neg\neg\mathsf{Dec}(\overline{\cdot} \mid d)$.*

Proof. The first implication follows by Fact 40. For the converse, note that the contraposition of Lemma 39 shows $\forall d{:}\mathcal{M}.\, \neg\neg\mathsf{Dec}(\overline{\cdot} \mid d) \to \neg\neg\mathcal{M} \cong \mathbb{N}$ where the conclusion is equivalent to $\mathcal{M} \cong \mathbb{N}$ due to Lemma 33. ◀

7.3 Variants of the Theorem

We now investigate two further variants of the theorem, by assuming the existence of formulas which satisfy a stronger notion of inseparability and that the coding lemma can be proven inside of PA.

▶ **Definition 42.** *Two formulas $\alpha(x), \beta(x)$ are called* HA-inseparable *if $\lambda n{:}\mathbb{N}.\,\mathsf{Q} \vdash \alpha(\overline{n})$ and $\lambda n{:}\mathbb{N}.\,\mathsf{Q} \vdash \beta(\overline{n})$ are inseparable and one can also show $\mathsf{HA} \vdash \neg\exists\, x.\, \alpha(x) \wedge \beta(x)$.*

▶ **Hypothesis 43.** *There are Δ_0 formulas α_0, β_0 such that $\exists z.\, \alpha_0(z, x), \exists z.\, \beta_0(z, x)$ are* HA-*inseparable.*

▶ **Hypothesis 44.** *For any binary Δ_0 formula $\varphi(x, y)$* HA *can prove the following coding lemma on the object level: $\mathsf{HA} \vdash \forall\, n\, b\, \exists\, c\, \forall\, u < n.\ (\exists z < b.\, \varphi(z, u)) \leftrightarrow \Pi(u) \mid c$.*

According to [21], one way of establishing Hypothesis 43 is by taking the construction of inseparable formulas as seen earlier, and internalizing it within HA. Similarly, Hypothesis 44 is justified by noting that its proof should be an internalized version of the proof of Lemma 26.

The following variant of Tennenbaum's theorem is based on an observation by Makholm [17]. Most importantly, it avoids the usage of Overspill, by using Hypothesis 44. In contrast to the result in Section 7.1 we want to highlight that the next theorem does not presuppose MP or the stability of $\mathsf{std}\,\cdot$.

▶ **Theorem 45** (Makholm)**.** *We have $\mathcal{M} > \mathbb{N}$ if and only if $\exists d{:}\mathcal{M}.\neg\mathsf{Dec}(\overline{\cdot} \mid d)$.*

Proof. First note that the converse follows from Fact 40. Now assume we have $e{:}\mathcal{M} > \mathbb{N}$. By Hypothesis 43 there are HA-inseparable $\exists_1$ formulas $\exists z.\, \alpha_0(z, x)$ and $\exists z.\, \beta_0(z, x)$, where α_0, β_0 are binary Δ_0 formulas. Then let $X := \lambda n{:}\mathbb{N}.\, \mathcal{M} \vDash \exists\, z < e.\, \alpha_0(z, \overline{n})$.

- If $\mathsf{Q} \vdash \exists\, z.\, \alpha_0(z, \overline{n})$ there is $m{:}\mathbb{N}$ with $\mathbb{N} \vDash \alpha_0(\overline{m}, \overline{n})$ and $\mathcal{M} \vDash \alpha_0(\overline{m}, \overline{n})$ by Lemma 12. We therefore get Xn.
- Assuming $Xn \wedge \mathsf{Q} \vdash \exists\, z.\, \beta_0(z, \overline{n})$, then similarly to above, there is $m{:}\mathbb{N}$ with $\mathcal{M} \vDash \beta_0(\overline{m}, \overline{n})$, showing $\mathcal{M} \vDash \exists\, z < e.\, \beta_0(z, \overline{m})$. But together with Xn this contradicts the deductive disjointness property of the HA-inseparable formulas α_0 and β_0.

Due to the inseparability of the given $\exists_1$ formulas, this shows that X is not decidable. Using soundness on Hypothesis 44 for $\varphi := \alpha_0$ and $n, b := e$, we get $\mathcal{M} \vDash \exists\, c\, \forall\, u < e.\ \big(\exists\, z < e.\, \alpha_0(z, u)\big) \leftrightarrow \Pi(u) \mid c$. So there is a code $c{:}\mathcal{M}$ such that X is coded by it, showing that $\overline{\cdot} \mid c$ cannot be decidable. ◀

▶ **Corollary 46.** *We have $\forall e{:}\mathcal{M}.\neg\neg\,\mathsf{std}\, e$ iff $\forall d{:}\mathcal{M}.\neg\neg\mathsf{Dec}(\overline{\cdot} \mid d)$.*

McCarty [21, 20] considered Tennenbaum's theorem with constructive semantics. Instead of models placed in classical set-theory, he assumes an intuitionistic theory (e.g. IZF), making the interpretation of the object-level disjunction much stronger. We simulate this in our type theory by assuming the following choice principle:

▶ **Definition 47.** *By $\mathsf{AUC}_{\mathbb{N},\mathbb{B}}$ we denote the principle of unique choice:*

$$\forall R.\, (\forall x\, \exists! y.\, Rxy) \to \exists f : \mathbb{N} \to \mathbb{B}.\, \forall x.\, Rx(fx)$$

Note that CT and $\mathsf{AUC}_{\mathbb{N},\mathbb{B}}$ combined prove the negation of LEM [8]. In the following, we are therefore strictly anti-classical and in particular this means that there is no classical model of PA.

▶ **Lemma 48.** *For any formula $\varphi(x, y)$ we have $\mathcal{M} \vDash \forall\, b.\, \neg\neg\forall\, x, y < b.\, \varphi(x, y) \vee \neg\varphi(x, y)$.*

Proof. Single instances of the law of excluded middle are provable under double negation. We can then use this in combination with an induction on the bound b to prove the claim. ◀

▶ **Lemma 49.** *Assuming $\mathsf{AUC}_{\mathbb{N},\mathbb{B}}$ and $\mathcal{M} > \mathbb{N}$, we have $\forall d{:}\mathcal{M}.\neg\neg\mathsf{Dec}(\overline{\cdot} \mid d)$.*

Proof. Let $d\!:\!\mathcal{M}$ be given and assume $e\!:\!\mathcal{M} > \mathbb{N}$. Then we have $e + d + 1 > \mathbb{N}$ and using Lemma 48 we get

$$\begin{aligned}
&\mathcal{M} \vDash \forall b.\, \neg\neg\forall x, y < b.\, \varphi(x, y) \vee \neg\varphi(x, y)\\
\implies\; &\neg\neg\, \mathcal{M} \vDash \forall x, y < (e + d + 1).\ \varphi(x, y) \vee \neg\varphi(x, y)\\
\implies\; &\neg\neg\forall n\!:\!\mathbb{N}.\, \mathcal{M} \vDash \varphi(\overline{n}, d) \vee \neg\varphi(\overline{n}, d)\\
\implies\; &\neg\neg\forall n\!:\!\mathbb{N}.\, \mathcal{M} \vDash \varphi(\overline{n}, d) + \neg\, \mathcal{M} \vDash \varphi(\overline{n}, d)
\end{aligned}$$

where the last implication is possible, since $\mathsf{AUC}_{\mathbb{N},\mathbb{B}}$ implies the decidability of definite propositions. For the choice $\varphi(x, y) := x \mid y$ we then get the desired result. ◀

▶ **Corollary 50.** *Assuming* $\mathsf{AUC}_{\mathbb{N},\mathbb{B}}$*, then for every* $\mathcal{M} \vDash \mathsf{HA}$ *we have* $\neg\, \mathcal{M} > \mathbb{N}$.

Proof. Assuming $\mathcal{M} > \mathbb{N}$, Lemma 49 entails $\neg\exists d : \mathcal{M}.\neg\mathsf{Dec}(\overline{\cdot} \mid d)$, in contradiction to Theorem 45. ◀

▶ **Corollary 51** (McCarty)**.** *Given* $\mathsf{AUC}_{\mathbb{N},\mathbb{B}}$ *and* MP*,* HA *is categorical.*

Proof. Given that $\mathsf{HA} \vdash \forall xy.\, x = y \vee \neg\, x = y$, $\mathsf{AUC}_{\mathbb{N},\mathbb{B}}$ entails that every model $\mathcal{M} \vDash \mathsf{HA}$ is discrete, showing the stability of $\mathsf{std}\cdot$ by Lemma 33. Combined with Corollary 50 this shows $\mathcal{M} \cong \mathbb{N}$. ◀

8 Discussion

8.1 General Remarks

In Section 7, we presented several proofs of Tennenbaum's theorem which we summarize in the below table, listing their assumptions[5,6] on the left and the conclusion on the right.

MP	$\mathsf{AUC}_{\mathbb{N},\mathbb{B}}$	discrete	HA-insep.	Conclusion	from
•		•		$\mathbb{N} \cong \mathcal{M}$ iff $\mathcal{M}$ enumerable	Theorem 34
•		•		$\mathcal{M} > \mathbb{N} \to \neg\neg\exists d.\neg\mathsf{Dec}(\overline{\cdot} \mid d)$	Lemma 39
			•	$\mathcal{M} > \mathbb{N} \leftrightarrow \exists d.\neg\mathsf{Dec}(\overline{\cdot} \mid d)$	Theorem 45
	•		•	$\mathbb{N} \cong \mathcal{M}$	Corollary 51

First note that since PA can show definiteness of equality, the above listed assumption of the model $\mathcal{M}$ being discrete is equivalent to $\mathcal{M}$ being separated. Comparing Theorem 45 to Theorem 34 and Lemma 39 we see that its conclusion is constructively stronger. The noteworthy observation about Theorem 45 is that it cannot be reached by the proofs given in Section 7.2, as they crucially dependent on Overspill and therefore MP and discreteness. The result only becomes possible once we use a stronger notion of inseparability for formulas and avoid the usage of Overspill. As was pointed out by McCarty in [21], a weaker version of CT suffices for his proof. Analogously, a weaker version of CT_Q, called WCT_Q:

For every function $f : \mathbb{N} \to \mathbb{N}$ there *potentially* is a binary $\exists_1$ formula $\varphi_f(x, y)$ such that for every $n\!:\!\mathbb{N}$ we have $\mathsf{Q} \vdash \forall y.\, \varphi_f(\overline{n}, y) \leftrightarrow \overline{fn} = y$,

suffices for all of the proofs that we have presented. This only needs few changes of the presented proofs and we verified this in the Coq project.[7]

---

[5] We do not list the global assumption CT_Q. Additionally, we leave out Hypothesis 44, as it is expected to be provable.

[6] In the pdf they are linked back to their definitions.

[7] We could have presented all of the results with respect to WCT_Q. We opted against this in favor for CT_Q, to avoid additional handling of double negations and to keep the proofs more readable.

8.2 Coq Mechanization

The Coq development is not axiom free as the results crucially depend on the axiom $\mathsf{CT_Q}$ and the usage of MP for some of the results. Apart from this, there are two statements in Section 7.3 we have labeled as hypothesis, and which were also taken as additional axioms in the Coq development. They are expected to be provable and would usually be treated as facts and simply used, but since our treatment is backed up by a development in the proof assistant, we wanted to make these assumptions very explicit. In total, the development counts roughly 4600 lines of code. 2300 loc on the specification of first-order logic and basic results about PA models were reused from earlier work [13]. The present project differs from this development in the regard that the equality symbol is not interpreted as a predicate, but as the equality on the underlying model domain. The various coding lemmas from Section 6 took 530 loc to be formalized and all variants of Tennenbaum's theorem come to a total of only 800 lines, showing the advantages of a synthetic approach to computability.

8.3 Related Work

Presentations of first-order logic in the context of proof-checking have already been discussed and used by Shankar in [28], Paulson [24] and O'Connor [22], and the particular mechanization of first-order logic we use is based on [10, 11, 13]. Classical proofs of Tennenbaum's theorem can be found in [3, 30, 12]. There are also refinements of the theorem which show that computability of either operation suffices [19] as well as a weaker induction scheme [37, 4]. Constructive accounts were given by McCary [20, 21] and Plisko [25], and a relatively recent investigation into Tennenbaum phenomena by Godziszewski and Hamkins in [34]. Relevant work concerning synthetic computability are [27, 1] and for an account of Church's thesis in constructive mathematics we refer to Kreisel and Troelstra [15, 35]. Investigations into CT and its connections to other axioms of synthetic computability theory are found in [9].

8.4 Future Work

We would like to give a proper formalization of the arithmetic hierarchy, which would allow us to prove **??** and to conduct an analysis concerning the strength of the induction scheme needed to establish Tennenbaum's theorem. We would like to further justify $\mathsf{CT_Q}$ by starting off with the more conventional formulation of CT for Turing machines and verifying that it yields $\mathsf{CT_Q}$. To eliminate some of the assumptions made in Section 7.3, we also want to mechanize proofs of Hypothesis 43 and Hypothesis 44. A more satisfying rendering of McCarty's result will be achieved by changing Definition 9, putting the interpretations of formulas on the type level instead of the propositional level therefore removing the need to assume $\mathsf{AUC}_{\mathbb{N},\mathbb{B}}$. The presented versions of Tennenbaum's theorem do not explicitly mention the computability of addition or multiplication of the model, and as mentioned in Section 1 this is due to the chosen synthetic approach. To make these assumptions explicit again, we could assume an version of CT which makes reference to a T predicate [14, 8], and expresses that every T-computable function is representable in Q. We can then then distinguish between addition or multiplication being T-computable and formalize the result that T-computability of either operation leads to the model being standard [19].

References

1 Andrej Bauer. First steps in synthetic computability theory. *Electronic Notes in Theoretical Computer Science*, 155:5–31, 2006.

2 Andrej Bauer. Intuitionistic mathematics for physics, 2008. URL: `http://math.andrej.com/2008/08/13/intuitionistic-mathematics-for-physics/`.

3 George S Boolos, John P Burgess, and Richard C Jeffrey. *Computability and logic*. Cambridge university press, 2002.

4 Patrick Cegielski, Kenneth McAloon, and George Wilmers. Modèles récursivement saturés de l'addition et de la multiplication des entiers naturels. In *Studies in Logic and the Foundations of Mathematics*, volume 108, pages 57–68. Elsevier, 1982.

5 Thierry Coquand and Gérard Huet. *The calculus of constructions*. PhD thesis, INRIA, 1986.

6 Martin Davis, Yuri Matijasevič, and Julia Robinson. Hilbert's tenth problem. Diophantine equations: positive aspects of a negative solution. American Math. Soc Providence, R. I, 1976.

7 Martin Davis, Hilary Putnam, and Julia Robinson. The decision problem for exponential Diophantine equations. *Annals of Mathematics*, pages 425–436, 1961.

8 Yannick Forster. Church's thesis and related axioms in coq's type theory. *arXiv preprint arXiv:2009.00416*, 2020.

9 Yannick Forster. Church's Thesis and Related Axioms in Coq's Type Theory. In Christel Baier and Jean Goubault-Larrecq, editors, *29th EACSL Annual Conference on Computer Science Logic (CSL 2021)*, volume 183 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 21:1–21:19, Dagstuhl, Germany, 2021. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. URL: `https://drops.dagstuhl.de/opus/volltexte/2021/13455`, `doi:10.4230/LIPIcs.CSL.2021.21`.

10 Yannick Forster, Dominik Kirst, and Gert Smolka. On synthetic undecidability in Coq, with an application to the Entscheidungsproblem. In *Proceedings of the 8th ACM SIGPLAN International Conference on Certified Programs and Proofs*, pages 38–51, 2019.

11 Yannick Forster, Dominik Kirst, and Dominik Wehr. Completeness theorems for first-order logic analysed in constructive type theory: Extended version. *Journal of Logic and Computation*, 31(1):112–151, 2021.

12 Richard Kaye. Tennenbaum's theorem for models of arithmetic. *Set Theory, Arithmetic, and Foundations of Mathematics. Ed. by J. Kennedy and R. Kossak. Lecture Notes in Logic. Cambridge*, pages 66–79, 2011.

13 Dominik Kirst and Marc Hermes. Synthetic Undecidability and Incompleteness of First-Order Axiom Systems in Coq. In Liron Cohen and Cezary Kaliszyk, editors, *12th International Conference on Interactive Theorem Proving (ITP 2021)*, volume 193 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 23:1–23:20, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. URL: `https://drops.dagstuhl.de/opus/volltexte/2021/13918`, `doi:10.4230/LIPIcs.ITP.2021.23`.

14 Stephen Cole Kleene. Recursive predicates and quantifiers. *Transactions of the American Mathematical Society*, 53(1):41–73, 1943.

15 Georg Kreisel. Church's thesis: a kind of reducibility axiom for constructive mathematics. In *Studies in Logic and the Foundations of Mathematics*, volume 60, pages 121–150. Elsevier, 1970.

16 Dominique Larchey-Wendling and Yannick Forster. Hilbert's tenth problem in Coq. In *4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019*, volume 131, pages 27–1. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019.

17 Henning Makholm. Tennenbaum's theorem without overspill. Mathematics Stack Exchange. (version: 2014-01-24). URL: `https://math.stackexchange.com/q/649457`.

18 Yuri V. Matijasevič. *Enumerable sets are Diophantine. Soviet Mathematics: Doklady*, 11:354–357, 1970.

19 Kenneth McAloon. On the complexity of models of arithmetic. *The Journal of Symbolic Logic*, 47(2):403–415, 1982.

20 Charles McCarty. Variations on a thesis: intuitionism and computability. *Notre Dame Journal of Formal Logic*, 28(4):536–580, 1987.

21 Charles McCarty. Constructive validity is nonarithmetic. *The Journal of Symbolic Logic*, 53(4):1036–1041, 1988. URL: `http://www.jstor.org/stable/2274603`.

22 Russell O'Connor. Essential incompleteness of arithmetic verified by Coq. In *International Conference on Theorem Proving in Higher Order Logics*, pages 245–260. Springer, 2005.

23 Christine Paulin-Mohring. Inductive definitions in the system Coq rules and properties. In *International Conference on Typed Lambda Calculi and Applications*, pages 328–345. Springer, 1993.

24 Lawrence C Paulson. A mechanised proof of Gödel's incompleteness theorems using nominal Isabelle. *Journal of Automated Reasoning*, 55(1):1–37, 2015.

25 Valerii Egorovich Plisko. Constructive formalization of the Tennenbaum theorem and its applications. *Mathematical notes of the Academy of Sciences of the USSR*, 48(3):950–957, 1990.

26 Panu Raatikainen. Gödel's Incompleteness Theorems. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2021 edition, 2021.

27 Fred Richman. Church's thesis without tears. *The Journal of symbolic logic*, 48(3):797–803, 1983.

28 Natarajan Shankar. *Proof-checking metamathematics (theorem-proving)*. PhD thesis, The University of Texas at Austin, 1986.

29 Peter Smith. *An introduction to Gödel's theorems*. Cambridge University Press, 2013.

30 Peter Smith. Tennenbaum's theorem. 2014.

31 Andrew Swan and Taichi Uemura. On church's thesis in cubical assemblies, 2019. `arXiv:1905.03014`.

32 The Coq Development Team. The coq proof assistant, January 2022. `doi:10.5281/zenodo.5846982`.

33 Stanley Tennenbaum. Non-archimedean models for arithmetic. *Notices of the American Mathematical Society*, 6(270):44, 1959.

34 Michał Tomasz Godziszewski and Joel David Hamkins. Computable quotient presentations of models of arithmetic and set theory. *arXiv e-prints*, pages arXiv–1702, 2017.

35 Anne S Troelstra. *Metamathematical investigation of intuitionistic arithmetic and analysis*, volume 344. Springer Science & Business Media, 1973.

36 AS Troelstra and D van Dalen. Constructivism in mathematics, vol. 1. number 121 in studies in logic and the foundations of mathematics, 1988.

37 George Wilmers. Bounded existential induction. *The Journal of Symbolic Logic*, 50(1):72–90, 1985.

38 Norihiro Yamada. Game semantics of martin-löf type theory, part iii: its consistency with church's thesis, 2020. `arXiv:2007.08094`.