Interactive Realizability for Second-Order Heyting Arithmetic with EM1 and SK1

Federico Aschieri

Laboratoire PPS, équipe PI.R2
Université Paris 7, INRIA and CNRS

Abstract

We introduce a classical realizability semantics based on interactive learning for full second-order Heyting Arithmetic with excluded middle and Skolem axioms over Σ^0_1-formulas. Realizers are written in a classical version of Girard's System F. Since the usual computability semantics does not apply to such a system, we introduce a constructive forcing/computability semantics: though realizers are not computable functional in the sense of Girard, they can be forced to be computable. We apply these semantics to show how to extract witnesses from realizable Π^0_2-formulas. In particular a constructive and efficient method is introduced. It is based on a new "(state-extending-continuation)-passing-style translation" whose properties are described with the constructive forcing/computability semantics.

Keywords: interactive learning-based realizability, classical second-order arithmetic
2010 MSC: 03F03, 03F30, 03F55

1. Introduction

In the past years, several computational interpretations of classical logic have been put forward. Under a first classification, they fall into two large categories: *direct* and *indirect* interpretations. Among the indirect interpretations one finds the negative translations followed either by Dialectica interpretations ([15], [32]) or intuitionistic realizability interpretations combined with Friedman's translation [14]. Among the direct interpretations, there are classical realizabilities, Coquand game semantics [13], cut-elimination and normalization of classical proofs (under Curry-Howard correspondence of not), and the epsilon substitution method [26] (the Kreisel no-counterexample interpretation [22] is an easy corollary of the other ones).

Such a variety is surprising and on a first sight these interpretations may appear completely different, but it is becoming evident that some unifying concepts exist. Maybe the most general and powerful one is the concept of *learning*. That is, the computational content of classical proofs can be described in terms of learning programs, that acquire new knowledge about non-computable functions by an intelligent process of making hypotheses and testing them in search of counterexamples. As soon as one adopts this conceptual perspective, all the computational interpretations become clearly related and appear as technical variations on a same theme: learning.

One one hand, indirect interpretations yield, as a result of negative translation, programs using continuations. The deep reason continuations are used is that they are natural tools for implementing backtracking, i.e. the mechanism by which learning programs make guesses, learn about their mistakes and correct them thanks to the new acquired knowledge. In Berardi et al. [9] realizability interpretation of the negative translation of the axiom of countable choice, continuations are used to build finite approximations of choice functions. The Dialectica interpretation and the Friedman translation from the computational point of view provide ways to capture counterexamples to the hypotheses made by learning programs.

On the other hand, direct interpretations exploit the fact that classical principles have a remarkably immediate computational content when considered as learning devices. In the case of Krivine classical realizability [25] the excluded middle $A \vee \neg A$ is interpreted as a program that assume $\neg A$ as a working hypothesis: if at some point of the computation encounters evidence for A (i.e., a realizer of A), then it backtracks, erases everything that depended on the hypothesis $\neg A$ and acquires a realizer of A as new knowledge. Coquand game semantics interprets the excluded middle basically with the same spirit, but in a more intuitive way. The epsilon substitution method instead exploits the learning content of Skolem axioms, which are formulas of the form

$$\forall x \forall y.A(x, y) \implies A(x, f(x))$$

The function f is intepreted as an approximation of some choice function, mapping x to a witness (if any exists) for the formula $\exists y A(x, y)$. Whenever an instance

$$A(n, m) \implies A(n, f(n))$$

of a Skolem axiom is false, one can correct the function f as to output m on input n.

1.1. Realizability Based on Interactive Learning

Given this strong evidence that learning is the *key* for understanding the computational content of classical proofs, an important goal is to formulate classical realizability semantics explicitly based on learning. Such semantics should describe: first, the nature of the knowledge that programs coming from classical proofs acquire during computations; secondly, how this knowledge evolves during computations. As a consequence of this approach, it should be possible to develop a much finer understanding and control of the backtracking mechanism that interpret classical proofs; in other words: more efficient programs.

A significant step towards this goal has been taken in Aschieri and Berardi [4], where it has been introduced a learning-based classical realizability for first-order Heyting Arithmetic HA with the excluded middle EM_1 on Σ^0_1-formulas and Skolem axioms SK_1 over quantifier-free formulas. It is a realizability based on states, which describe the current knowledge of realizers. The reason why such a fragment has been isolated and studied is that just *monotonic* learning is sufficient to interpret it. By monotonicity of learning, we intend that learning programs can only increase their knowledge and once acquired, the knowledge is correct forever. This is the most simple instance of learning, it has special properties[1] and it is worth to be studied separately. In more general settings, in fact, learning is more complex. For example, in the case of full first-order Peano Arithmetic, learning is finitely *nested*, that is knowledge is stratified and the correctness of what is learned at any level depends on what has been learned at the previous levels (see Avigad [6], for an ordinal analysis of this kind of learning, and Aschieri [1], [3], for a type theoretic analysis). In the case of predicative fragments of Analysis, learning is transfinitely nested (see Aschieri [2]).

The crucial contribution of Interactive realizability is that it decomposes into two conceptual steps the extraction of programs from classical proofs. The idea is that, first, one extracts an ideal program, obtained with free use of oracles and Skolem functions: this programs is very natural to write, it obeys the laws of intuitionistic Heyting semantics (see e.g., Troesltra [33]) and is easy to understand. Then, classical principles suggest how to approximate in a very efficient way the oracles and Skolem functions used in the computation. The result is a model of intelligent programs, able to correct themselves and to learn from the mistakes they make when trying to achieve some goal defined by the usual Heyting intuitionistic reading of logical sentences. In hindsight, our interpretation can be seen as modern version of the epsilon substitution method, refined and rebuilt around the Curry-Howard correspondence for classical logic.

In this work, we extend the Interactive realizability for $\mathsf{HA} + \mathsf{EM}_1 + \mathsf{SK}_1$ to second-order intuitionistic Heyting Arithmetic HAS plus EM_1 and SK_1. The Arithmetic $\mathsf{HA} + \mathsf{EM}_1 + \mathsf{SK}_1$ is realized by adding an oracle for the Halting problem to Gödel's system T and by computationally interpreting EM_1 as a device which effectively learns oracle values during calculations. The resulting notion of realizability is just Kreisel modified realizability [23] extended with learning. It is thus natural to realize the theory $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ by adding to Girard's system F an oracle for the Halting problem and try again to interpret the excluded

[1] In the field of proof mining, one is indeed interested in exploiting every special property of the theory one is studying. If a constructive interpretation generalizes to a stronger theory, there is a good chance that one is losing some constructive information about the theorems of the former theory.

middle as a learning device: it is indeed the approach followed in this work. Again, the resulting notion of realizability will be a natural extension of intuitionistic realizability for HAS (for which we basically follow the formulation of Oliva and Streicher [29], but in a Church style) with learning.

We also introduce a new technique for witness extraction from proofs of Π^0_2-formulas. That is, given a realizer $t \Vdash \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$, with P atomic predicate, we extract a non-trivial program taking as input any number n and yielding as output a number m such that Pnm holds. With a classical meta-theory, the problem is easy. Constructively, t can be seen as a state-extending operator; classically, it can be proved that it is enough to extend the state a certain number of times to get a sufficient amount of information to compute a witness. From the constructive point of view, however, the problem is non-trivial, since t is a term of a classical version of system F and it is not possible to apply directly the computability method of Girard to reason about t. Our new technique is a significant refinement and extension of the one in Aschieri [3] and its correctness proof is completely intuitionistic. It is composed of two ingredients:

- First, we define a constructive forcing/computability semantics, explaining that although an interactive realizer is not computable, it can be *constructively forced to behave* like a computable functional. We call our notion of forcing *Constructive forcing* as opposed to *Classical forcing*, which has its origins in Cohen's work in set theory ([12]). The difference between the two versions of forcing is that while classical forcing is a relation between conditions and formulas, constructive forcing is a relation between conditions and proofs (actually, proof terms thanks to Curry-Howard); moreover, while in classical forcing quantification over future conditions is unrestricted, in constructive forcing the future that can be considered is only the future given by a continuation of the computation. The result is that the second has an intuitionistic meta-theory while the first's is classical. This means that constructive forcing has a direct computational content. Despite all these differences, it is clear that constructive forcing is an effective version of classical forcing, hence the name.
- Secondly, we define a new continuation-passing-style translation, which, in particular, manipulates state-extending continuations. When applied to a realizer t, it produces a program that *forces* the computability of t. Moreover, that program is able to backtrack efficiently at the right points of computations and does not forget precious information when backtracking (a common defect among computational interpretations of classical logic, such as [25, 9]).

These properties represent a first concrete evidence that realizability based on states and learning can lead to the implementation of more efficient programs already in pure lambda calculus – not to mention abstract machines and imperative features that may raise the bar of efficiency even higher. Constructive forcing, moreover, is quite a general technique and it can be iterated in order to work for all reasonable systems of predicative Analysis, as clearly showed in [2].

1.2. Plan of the Paper

In section §2 we introduce the term calculus in which realizers will be written, namely an extension of Girard's system F plus a constant symbol for a Skolem function Φ.

In section §3, we introduce our notion of realizability based on interactive-learning for $\mathsf{HAS}+\mathsf{EM}_1+\mathsf{SK}_1$.

In section §4, we present our techniques of witness extraction.

In section §5, we briefly compare Interactive realizability with other related computational interpretations.

In section §Appendix A, we show how to realize also the ex-falso-quodlibet axiom.

2. The Term Calculus $\mathcal{F}_{\text{Class}}$

In this section we introduce the typed lambda calculi that we shall use to define interactive realizability: system $\mathcal{F}$ and $\mathcal{F}_{\text{Class}}$. $\mathcal{F}$ is a completely standard extension of Girard's system F (see Girard [20]) with some syntactic sugar: numerals, booleans, primitive recursion at all types, if-then-else, pairs, finite partial functions over $\mathbb{N}$ and simple primitive recursive operations over them. Equivalently, $\mathcal{F}$ may be seen as an extension of Gödel's system T (we refer to the exposition in Girard [20]) with polymorphism. $\mathcal{F}_{\text{Class}}$ is obtained from $\mathcal{F}$ by adding on top of it a Skolem function symbol $\Phi : \mathtt{N} \to (\mathtt{N} \to \mathtt{N})$ of the same Turing degree of an oracle for the Halting problem. The symbol is totally inert from the computational point of view and so realizers will always be computed with respect to some approximation of the Skolem function represented by Φ.

2.1. Updates

In order to define $\mathcal{F}$, we have first to define the concept of update, which is nothing but a finite partial function over $\mathbb{N}$. We use the appellative "update", because realizers of atomic formulas will return finite partial functions as new pieces of information that they have learned about the Skolem function Φ; updates represent new associations input-output that are intended to correct (and in this sense, *update*) wrong oracle values used in computations.

Definition 1 (Updates and Consistent Union). *We define:*

1. *A binary* predicate *of* T *is any closed normal term* $P : \mathtt{N}^2 \to \mathtt{Bool}$ *of Gödel's system* T*.*

2. *We assume* $\mathsf{P}_0, \mathsf{P}_1, \mathsf{P}_2, \dots$ *is an arbitrary enumeration of all binary predicates of* T*.*

3. *An update set* U*, shortly an* update*, is a finite set of triples of natural numbers representing a finite partial function from* $\mathbb{N}^2$ *to* $\mathbb{N}$*. We say that* U *is* sound *if for every* $(i, n, m) \in U$*, we have* $\mathsf{P}_i nm = \mathtt{True}$*.*

4. *Two triples* (a, n, m) *and* (a', n', m') *of numbers are* consistent *if* $a = a'$ *and* $n = n'$ *implies* $m = m'$*.*

5. *Two updates* U_1, U_2 *are consistent if* $U_1 \cup U_2$ *is an update.*

6. $\mathbb{U}$ *is the set of all updates.*

7. *The* consistent union $U_1 \,\mathcal{U}\, U_2$ *of* $U_1, U_2 \in \mathbb{U}$ *is* $U_1 \cup U_2$ *minus all triples of* U_2 *which are inconsistent with some triple of* U_1*.*

We think of a triple (a, n, m) of a sound update as the code of a witness for $\exists y.\mathsf{P}_a(n, y)$. The fact that every update is a partial *function* allows in each update at most one witness for each formula $\exists y.\mathsf{P}_a(n, y)$.

$U_1 \,\mathcal{U}\, U_2$ is an non-commutative operation: whenever a triple of U_1 and a triple of U_2 are inconsistent, we arbitrarily keep the triple of U_1 and we reject the triple of U_2, therefore for some U_1, U_2 we have $U_1 \,\mathcal{U}\, U_2 \neq U_2 \,\mathcal{U}\, U_1$. $\mathcal{U}$ is a "learning strategy", a way of selecting a consistent subset of $U_1 \cup U_2$.

It is immediate to show that $\mathcal{U}$ is an associative operation on the set of updates, with neutral element $\emptyset$, with upper bound $U_1 \cup U_2$, and returning a non-empty udpate whenever $U_1 \cup U_2$ is non-empty.

Lemma 1. *Assume* $i \in \mathbb{N}$ *and* $U_1, \dots, U_i \in \mathbb{U}$*.*

1. $U_1 \,\mathcal{U}\, \dots \mathcal{U}\, U_i \subseteq U_1 \cup \dots \cup U_i$
2. $U_1 \,\mathcal{U} \dots \mathcal{U}\, U_i = \emptyset$ *implies* $U_1 = \dots = U_i = \emptyset$*.*

In fact, the whole realizability semantics is a Monad [10]. In [10], it is proved that a fragment of our realizability semantics is parametric with respect to the definition we choose for $\mathcal{U}$. Any associative operation $\mathcal{U}$, with neutral element $\emptyset$ and satisfying the two properties of Lemma 1, defines a different but sound realizability semantics, corresponding to a different "learning strategy".

2.2. The System $\mathcal{F}$

System $\mathcal{F}$ is formally described in figure 1. A numeral is a term of the form $\mathsf{S}(\mathsf{S}(\ldots 0))$. Terms of the form $\mathsf{if}\, T\, t_1\, t_2\, t_3$ will be written in the more legible form $\mathsf{if}\ t_1\ \mathsf{then}\ t_2\ \mathsf{else}\ t_3$, whenever T can be inferred from the context. For every update $U \in \mathbb{U}$, there is in $\mathcal{F}$ a constant $\overline{U} : \mathtt{U}$, where $\mathtt{U}$ is a new base type representing $\mathbb{U}$. We write $\varnothing$ for $\overline{\emptyset}$. In $\mathcal{F}$, there are four operations involving updates (see figure 1):

1. The first operation is denoted by the constant $\mathsf{is} : \mathtt{U} \to \mathtt{N}^2 \to \mathtt{Bool}$. is takes as arguments an update constant $\overline{U}$ and two numerals a, n; it returns $\mathtt{True}$ if $(a, n, m) \in U$ for some $m \in \mathtt{N}$ (that is, if the pair (a, n) is in the domain of the partial map U); it returns $\mathtt{False}$ otherwise.

2. The second operation is denoted by the constant $\mathsf{get} : \mathtt{U} \to \mathtt{N}^2 \to \mathtt{N}$. get takes as arguments an update constant $\overline{U}$ and two numerals a, n; it returns m if $(a, n, m) \in U$ for some $m \in \mathtt{N}$ (that is, if (a, n) belongs to the domain of the partial function U); it returns 0 otherwise.

3. The third operation is denoted by the constant $\mathsf{mkupd} : \mathtt{N}^3 \to \mathtt{U}$. mkupd takes as arguments three numerals a, n, m and transforms them into (the constant coding in $\mathcal{T}$) the update $\{(a, n, m)\}$.

4. The forth operation is denoted by the constant $\Cup : \mathtt{U}^2 \to \mathtt{U}$. $\Cup$ takes as arguments two update constants and returns the update constant denoting their consistent union.

We observe that the constants $\mathsf{is}, \mathsf{get}, \mathsf{mkupd}$ are just syntactic sugar and may be avoided by coding finite partial functions into natural numbers. We assume having in Gödel's T some terms $\Rightarrow_{\mathtt{Bool}}: \mathtt{Bool} \to \mathtt{Bool} \to \mathtt{Bool}, \neg_{\mathtt{Bool}} : \mathtt{Bool} \to \mathtt{Bool}, \vee_{\mathtt{Bool}} : \mathtt{Bool} \to \mathtt{Bool} \to \mathtt{Bool} \ldots$, implementing boolean connectives. If $t_1, \ldots, t_n, t \in \mathsf{T}$ have type $\mathtt{Bool}$ and are made from free variables all of type $\mathtt{Bool}$, using boolean connectives, we say that t is a tautological consequence of $t_1, \ldots, t_n$ in T (a tautology if $n = 0$) if all boolean assignments making $t_1, \ldots, t_n$ equal to $\mathtt{True}$ in T also make t equal to $\mathtt{True}$ in T.

As usual when working with polymorphic lambda calculus, one can define sum types $A+B$ and existential types $\exists X A$. We shall need $A + B$ in order to define functionals which return either object of type A or of type B, without knowing in advance of which type. This situation occurs when one needs to define a realizer a disjunction $A \vee B$, which has either to return a realizer of A or a realizer of B. Similarly, we shall need $\exists X A$ in order to define functionals which return a pair of a type B and an object of type $A[B/X]$, without knowing in advance the type B. This situation occurs when one needs to define a realizer of a formula $\exists X A$, which has either to return a type B and a realizer of $A[B/X]$, for some B.

Definition 2 (Sum Types, Existential Types). 1. *For all types A, B of system $\mathcal{F}$, we define a type*

$$A + B := \forall X.(A \to X) \to (B \to X) \to X$$

where X is a variable not occurring free in A, B.

2. *For every term $u : A$ of system F, we define a term*

$$\iota_{0,A,B}(u) := \Lambda X \lambda f^{A \to X} \lambda g^{B \to X} f u$$

of type $A + B$.

3. *For every term $u : B$ of system F, we define a term*

$$\iota_{1,A,B}(u) := \Lambda X \lambda f^{A \to X} \lambda g^{B \to X} g u$$

of type $A + B$.

Types

$$A, B ::= X \mid \mathtt{N} \mid \mathtt{Bool} \mid \mathtt{U} \mid A \to B \mid A \times B \mid \forall X A$$

Constants

$$c ::= \mathsf{R} \mid \mathsf{if} \mid 0 \mid \mathsf{S} \mid \mathtt{True} \mid \mathtt{False} \mid \mathsf{mkupd} \mid \mathsf{is} \mid \mathsf{get} \mid \Cup \mid \overline{U} \text{ (for every } U \in \mathbb{U})$$

Terms

$$t, u ::= c \mid x^A \mid tu \mid \lambda x^A u \mid tA \mid \Lambda X u \mid \langle t, u\rangle \mid \pi_i u$$

Typing Rules for Variables and Constants

$$\begin{aligned}
x^A &: A\\
0 &: \mathtt{N}\\
\mathsf{S} &: \mathtt{N} \to \mathtt{N}\\
\mathtt{True} &: \mathtt{Bool}\\
\mathtt{False} &: \mathtt{Bool}\\
\overline{U} &: \mathtt{U} \text{ (for every } U \in \mathbb{U})\\
\Cup &: \mathtt{U} \to \mathtt{U} \to \mathtt{U}\\
\mathsf{is} &: \mathtt{U} \to \mathtt{N} \to \mathtt{N} \to \mathtt{Bool}\\
\mathsf{get} &: \mathtt{U} \to \mathtt{N} \to \mathtt{N} \to \mathtt{N}\\
\mathsf{mkupd} &: \mathtt{N} \to \mathtt{N} \to \mathtt{N} \to \mathtt{U}\\
\mathsf{if} &: \forall X.\ \mathtt{Bool} \to X \to X \to X\\
\mathsf{R} &: \forall X.\ X \to (\mathtt{N} \to (X \to X)) \to \mathtt{N} \to X
\end{aligned}$$

Typing Rules for Composed Terms

$$\frac{t : A \to B \qquad u : A}{tu : B} \qquad \frac{u : B}{\lambda x^A u : A \to B}$$

$$\frac{u : \forall X A}{uB : A[B/X]} \qquad \frac{u : A}{\Lambda X u : \forall X A}\ X \notin FreeVarTypes(u)$$

$$\frac{u : A \qquad t : B}{\langle u, t\rangle : A \times B} \qquad \frac{u : A_0 \times A_1}{\pi_i u : A_i}\ i \in \{0, 1\}$$

Reduction Rules All the usual reduction rules for system F (see Girard [20]) plus the rules for recursion, if-then-else and projections

$$\mathsf{R}Tuv0 \mapsto u \qquad \mathsf{R}Tuv\mathsf{S}(t) \mapsto vt(\mathsf{R}Tuvt)$$

$$\mathsf{if}T\,\mathtt{True}\,u\,v \mapsto u \qquad \mathsf{if}T\,\mathtt{False}\,u\,v \mapsto v \qquad \pi_i\langle u_0, u_1\rangle \mapsto u_i, i = 0, 1$$

plus the following ones, assuming a, n, m be numerals:

$$\mathsf{is}\,\overline{U}\,a\,n \mapsto \begin{cases} \mathtt{True} & \text{if } \exists m.\ (a, n, m) \in U\\ \mathtt{False} & \text{otherwise}\end{cases}$$

$$\mathsf{get}\,\overline{U}\,a\,n \mapsto \begin{cases} m & \text{if } \exists m.\ (a, n, m) \in U\\ 0 & \text{otherwise}\end{cases}$$

$$\overline{U}_1 \Cup \overline{U}_2 \mapsto \overline{U_1 \mathcal{U} U_2}$$

$$\mathsf{mkupd}\,a\,n\,m \mapsto \overline{U}, \text{with } U = \{(a, n, m)\}$$

Figure 1: System $\mathcal{F}$

4. *For all types A of $\mathcal{F}$, we define a type*

$$\exists X A := \forall Y.(\forall X.A \to Y) \to Y$$

where Y is a variable not occurring free in A.

5. *For every type B and term u of type $A[B/X]$, we define a term*

$$\langle B, u\rangle := \Lambda Y \lambda x^{\forall X.A \rightarrow Y} xBu$$

of type $\exists XA$.

System $\mathcal{F}$ is obtained from system T adding polymorphism and new operations on atomic types. The following definition formalizes what has been done and it useful for defining arbitrary extensions of $\mathcal{F}$ with arbitrary functions over natural numbers; we shall need such extensions for adjoining non-computable functions to $\mathcal{F}$.

Definition 3 (Functional set of rules). *Let C be any set of constants, each one of some type $A_1 \rightarrow \ldots \rightarrow A_n \rightarrow A$, for some $A_1, \ldots, A_n, A \in \{\texttt{Bool}, \texttt{N}, \texttt{U}\}$. We say that $\mathcal{R}$ is a* functional set of reduction rules *for C if $\mathcal{R}$ consists, for all $c \in C$ and all $a_1 : A_1, \ldots, a_n : A_n$ closed normal terms of $\mathcal{F}$, of exactly one rule $ca_1 \ldots a_n \mapsto a$, for some closed normal term $a : A$ of $\mathcal{F}$.*

Any extension of $\mathcal{F}$ with constants and even non-computable functional sets of rules, is strongly normalizing and has the uniqueness-of-normal-form property.

Theorem 2. *Assume that $\mathcal{R}$ is a functional set of reduction rules for C (def. 3). Then $\mathcal{F} + C + \mathcal{R}$ enjoys strong normalization and weak-Church-Rosser (uniqueness of normal forms) for all closed terms of atomic types.*

Proof. For strong normalization, see Berger [11] or just use standard reducibility arguments, as in Girard [20]. Weak Church-Rosser is also standard.

The following normal form theorem also holds.

Lemma 3 (Normal Form Property for $\mathcal{F} + C + \mathcal{R}$). *Assume that $\mathcal{R}$ is a functional set of reduction rules for C. Assume A is either an atomic type or a product type. Then any closed normal term $t \in \mathcal{F}$ of type A is: a numeral $n : \texttt{N}$, or a boolean $\texttt{True}, \texttt{False} : \texttt{Bool}$, or an update constant $\overline{U} : \texttt{U}$, or a constant of type A, or a pair $\langle u, v\rangle : B \times C$.*

Proof. By induction over t. For some sequence $\vec{v}$ of closed types and terms, either t is $(\lambda x.u)\vec{v}$, or t is $(\Lambda X.u)\vec{v}$ or t is $\langle u, w\rangle\vec{v}$, or t is $x\vec{v}$ for some variable x, or t is $c\vec{v}$ for some constant c.
If $t = (\lambda x.u)\vec{v}$, then t has an arrow type if $\vec{v} = \emptyset$, while t is not normal if $\vec{v} \neq \emptyset$.
If $t = (\Lambda X.u)\vec{v}$, then t has universal type if $\vec{v} = \emptyset$, while t is not normal if $\vec{v} \neq \emptyset$.
If $t = \langle u, w\rangle\vec{v}$, then $\vec{v} = \emptyset$ and we are done.
If $t = x(\vec{v})$ then t is not closed.
The only case left is $t = c\vec{v} : A$. If $t = 0$ we are done, if $t = \mathsf{S}(u)$ we apply the induction hypothesis to u, if $t = \texttt{True}, \texttt{False} : \texttt{Bool}$ or $t = \overline{u} : \texttt{U}$ or t is a constant of C we are done. Otherwise either $t = (\mathsf{R}U s_1 s_2 n)\vec{t}$ or $t = (\mathsf{if} U b a_1 a_2)\vec{t}$ or $t = \pi_i(z)\vec{w}$, or $t = \mathsf{is}\, u\, n\, m : \texttt{N}$, or $t = \mathsf{get}\, u\, n\, m : \texttt{N}$, or $t = \mathbb{U}(u_1, u_2) : \texttt{U}$, or $t = \mathsf{mkupd}\, n\, m\, l$ or $t = c\vec{v}$, with $c \in C$ and $v_1 : A_1, \ldots, v_k : A_k$, and A_i atomic for every i . The proper subterms $n, m, l : \texttt{N}$, $b : \texttt{Bool}$, $z : A \times B$, $u, u_1, u_2 : \texttt{U}$, $v_1 : A_1, \ldots v_k : A_k$ of t have atomic or product type and are closed normal. By induction hypothesis they are, respectively, numerals, booleans, pairs, constants. In all cases, t is not normal.

2.3. The System $\mathcal{F}_{\text{Class}}$

We now define a classical extension of $\mathcal{F}$, that we call $\mathcal{F}_{\text{Class}}$, with a constant symbol $\Phi : \texttt{N}^2 \rightarrow \texttt{N}$ denoting a non-computable map of the same Turing degree of an oracle for the Halting problem. We shall use the elements of $\mathcal{F}_{\text{Class}}$ to represent non-computable realizers.

Definition 4 (Systems $\mathcal{F}_{\text{Class}}$ and $\mathcal{T}_{\text{Class}}$). *Define $\mathcal{F}_{\text{Class}} = \mathcal{F} + \Phi$ and $\mathcal{T}_{\text{Class}} = \mathsf{T} + \Phi$, where $\Phi : \texttt{N}^2 \rightarrow \texttt{N}$ is a new constant symbol and T is Gödel's system.*

For every numeral a, Φa – which we shall denote with Φ_a – represents a *Skolem function* for the formula $\exists y^{\mathtt{N}}\, \mathsf{P}_a xy$, taking as argument a number x and returning some y such that $\mathsf{P}_a xy$ if any exists, and an arbitrary value otherwise. There is no set of computable reduction rules for the constant Φ, and therefore no set of computable reduction rules for $\mathcal{F}_{\text{Class}}$.

Each (in general, non-computable) term $t \in \mathcal{F}_{\text{Class}}$ is associated to a set $\{t[s] \mid s \in \mathcal{F}, s : \mathtt{N}^2 \to \mathtt{N}\} \subseteq \mathcal{F}$ of computable terms we call its "approximations", one for each term $s : \mathtt{N}^2 \to \mathtt{N}$ of $\mathcal{F}$, which is thought as a computable approximation of the oracle Φ.

Definition 5 (Approximation at state s). *We define:*

1. *A* state *is a closed term of type* $\mathtt{N}^2 \to \mathtt{N}$ *of* $\mathcal{F}$*. We define* $\mathtt{S} := \mathtt{N}^2 \to \mathtt{N}$*.*
2. *Assume* $t \in \mathcal{F}_{\text{Class}}$ *and* s *is a state. The "approximation of* t *at state* s*" is the term* $t[s]$ *of* $\mathcal{F}$ *obtained from* t *by replacing each constant* Φ *with* s*.*

We interpret any $t[s] \in \mathcal{F}$ as a learning process evaluated with respect to the information taken from an approximation s of Φ. Here we consider an approximation of Φ to be an arbitrary term $s : \mathtt{N}^2 \to \mathtt{N}$; s may be correctly in agreement with Φ on some arguments, but wrong on other ones. Consequently, we are going to consider the set of (a, n) such that $\mathsf{P}_a n s_a(n) = \mathtt{True}$ as the real "domain" of s (again, with $s_a(n)$ we shall sometimes denote san). We are also going to define a term $\oplus$, which takes as argument a term $f : \mathtt{N}^2 \to \mathtt{N}$ and an update $\overline{U}$, and changes the values of f according to U. This is one of the fundamental operations of our computational model: realizers will compute updates to correct wrong values of oracle approximations with new good values that they have previously learned and stored in the updates. Last, using Φ, we are going to define for every numeral a the oracle X_a, which takes as argument a numeral n and returns the truth value of $\exists y^{\mathtt{N}}\, \mathsf{P}_a ny$.

Definition 6 (Domain, Updates of Functions, Oracle X_a). *We define:*

1. *If* s *is a state, we denote with* $\mathsf{dom}(s)$ *the set of pairs of numerals* (a, n) *such that* $\mathsf{P}_a n s_a(n) = \mathtt{True}$*. If* $\overline{U}$ *is an update constant, we denote with* $\mathsf{dom}(\overline{U})$ *the set of pairs of numerals* (a, n) *such that* $(a, n, m) \in U$ *and* $\mathsf{P}_a nm = \mathtt{True}$*.*

2. *If* s *is a state and* $\overline{U}$ *is an update constant, we define*
$$\overline{U} \preceq s \iff \mathsf{dom}(\overline{U}) \subseteq \mathsf{dom}(s)$$

3. *We define a term* $\oplus : (\mathtt{N}^2 \to \mathtt{N}) \to \mathtt{U} \to (\mathtt{N}^2 \to \mathtt{N})$ *as follows:*
$$\oplus := \lambda f^{\mathtt{N}^2 \to \mathtt{N}} \lambda u^{\mathtt{U}} \lambda x^{\mathtt{N}} \lambda y^{\mathtt{N}}\ \mathsf{if}\ (\mathsf{is}\, u\, x\, y)\ \mathsf{then}\ (\mathsf{get}\, u\, x\, y)\ \mathsf{else}\ fxy$$
We will write $t_1 \oplus t_2$ *in place of* $\oplus t_1 t_2$*.*

4. *Given any numeral* i*, we define a term* $\oplus_i : (\mathtt{N}^2 \to \mathtt{N}) \to \mathtt{U} \to (\mathtt{N}^2 \to \mathtt{N})$ *as follows:*
$$\oplus_i := \lambda f^{\mathtt{N}^2 \to \mathtt{N}} \lambda u^{\mathtt{U}} \lambda x^{\mathtt{N}} \lambda y^{\mathtt{N}}\ \mathsf{if}\ (i = x \wedge_{\mathtt{Bool}} \mathsf{is}\, u\, x\, y \wedge_{\mathtt{Bool}} \mathsf{P}_i\, y\, (\mathsf{get}\, u\, x\, y))\ \mathsf{then}\ (\mathsf{get}\, u\, x\, y)\ \mathsf{else}\ fxy$$
We will write $t_1 \oplus_i t_2$ *in place of* $\oplus_i t_1 t_2$*.*

5. *For every numeral* a*, we define a term* $\mathsf{X}_a : \mathtt{N} \to \mathtt{Bool}$ *as follows:*
$$\mathsf{X}_a := \lambda x^{\mathtt{N}}\ \mathsf{P}_a x(\Phi_a x)$$

We introduce now a notion of convergence for families of terms $\{t[s_i]\}_{i \in \mathbb{N}} \subseteq \mathcal{F}$, defined by some $t \in \mathcal{F}_{\text{Class}}$ and indexed over a set $\{s_i\}_{i \in \mathbb{N}}$ of states. Informally, "t convergent" means that the normal form of $t[s]$

eventually stops changing when the approximation s_i of Φ gets better and better. If s, r are states, we formalize what it means that r is at least as good an approximation as s by defining:

$$s \leq r \iff \forall a, n.\ s_a(n) \neq r_a(n) \implies (a, n) \notin \mathsf{dom}(s) \wedge (a, n) \in \mathsf{dom}(r)$$

Intuitively, if $s \leq r$, then r can be obtained by changing some of the values of s that make s a wrong approximation of Φ. We say that a sequence $\{s_i\}_{i\in\mathbb{N}}$ of states is a weakly increasing chain of states (is w.i. for short), if $s_i \leq s_{i+1}$ for all $i \in \mathbb{N}$.

Definition 7 (Convergence). *Assume that $\{s_i\}_{i\in\mathbb{N}}$ is a w.i. sequence of states, and $u, v \in \mathcal{F}_{\mathit{Class}}$.*

1. *u converges in $\{s_i\}_{i\in\mathbb{N}}$ if $\exists i \in \mathbb{N}. \forall j \geq i.\ u[s_j] = u[s_i]$ in $\mathcal{F}$.*

2. *u converges if u converges in every w.i. sequence sequence of closed type-$\mathtt{N} \to \mathtt{N}$ terms of $\mathcal{F}$.*

We remark that if u is convergent, we do not ask that u is convergent to the *same* value on *all* w.i. chain of oracle approximations. The value attained by u may depend on the information contained in the particular chain from which u gets the knowledge.

Theorem 4 (Convergence Theorem). *Assume $t \in \mathcal{F}_{\mathit{Class}}$ is a closed term of atomic type A ($A \in \{\mathtt{Bool}, \mathtt{N}, \mathtt{U}\}$). Then t is convergent.*

Proof. *(Classical).* For every $S : \mathbb{N}^2 \to \mathbb{N}$, we define some (in general, *not* computable) functional set of reduction rules $\mathcal{R}(S)$ for Φ and for $\mathcal{F}_{\text{Class}}$.

$$\mathcal{R}(S) := \{\Phi_a n \mapsto m \mid S(a, n) = m\}$$

If s is a state, we define

$$\mathcal{R}(s) := \{\Phi_a n \mapsto m \mid s_a(n) = m, \text{ with } a, n, m \text{ numerals}\}$$

By theorem 2, for any S and s, $\mathcal{F}_{\text{Class}} + \mathcal{R}(S)$ and $\mathcal{F}_{\text{Class}} + \mathcal{R}(s)$ are strongly normalizing and weak-CR for all closed terms of atomic type.
Claim. If s is a state, the normal form of t in $\mathcal{F}_{\text{Class}} + \mathcal{R}(s)$ is equal to the normal form of $t[s]$ in $\mathcal{F}$.
Proof of the Claim. By induction over the size of the reduction tree of t. If t is normal, then by lemma 3 it is a numeral, a boolean or a constant. Hence, $t \neq \Phi$ (for $\Phi : \mathtt{N}^2 \to \mathtt{N}$), and we are done, since $t = t[s]$. If t is not normal, then it reduces in one step reduction to some t'. If we show that $t[s]$ reduces $t'[s]$ in $\mathcal{F}$, we obtain the claim by induction hypothesis. The only non self-evident case is when a redex not already in t is contracted in $t[s]$. Such redex must be of the form $\Phi_a n$, with n numeral, and t' results from t by replacing the redex with m, where m is a numeral and $s_a(n) = m$. But then $t'[s]$ can as well be obtained from $t[s]$ by normalizing the redex $s_a(n)$ of $t[s]$.

Assume now w is the (unique, by weak-CR) normal form of t in $\mathcal{F}_{\text{Class}} + \mathcal{R}(s)$. By the *Claim*, w is the normal form of $t[s]$ in $\mathcal{F}$. For the rest of the proof, let $\{s_i\}_{i\in\mathbb{N}}$ be a w.i. chain of states. Define

$$S_\omega(a, n) := \begin{cases} m & \text{if } \exists i.\ (a, n) \in \mathsf{dom}(s_i) \wedge s_i(a, n) = m \\ s_0(a, n) & \text{otherwise} \end{cases}$$

S_ω is a well defined function, because $(a, n) \in \mathsf{dom}(s_i), (a, n) \in \mathsf{dom}(s_j)$ implies $s_i(a, n) = s_j(a, n)$, since $\{s_i\}_{i\in\mathbb{N}} \in$ w.i. By strong normalization, t has a finite reduction in normal form in $\mathcal{F}_{\text{Class}} + \mathcal{R}(S_\omega)$. Therefore in this reduction are used only *finitely many* reduction rules from $\mathcal{R}(S_\omega)$. Moreover, if $\Phi_a n \mapsto m$ is any of such rules, then either there exists i such that $(a, n) \in \mathsf{dom}(s_i)$ and $s_i(a, n) = m$, and so for every $j \geq i$, $s_i(a, n) = s_j(a, n)$ (for $\{s_i\}_{i\in\mathbb{N}} \in$ w.i.); or it does not, and so for every j, $s_j(a, n) = s_0(a, j)$ (again, $\{s_i\}_{i\in\mathbb{N}} \in$ w.i.). Therefore, all the reduction rules used to obtain the normal form a of t in $\mathcal{F}_{\text{Class}} + \mathcal{R}(S_\omega)$ are already in $\mathcal{R}(s_n)$, for some numeral n, and thus w is the normal form of t also in $\mathcal{F}_{\text{Class}} + \mathcal{R}(s_n)$, and in $\mathcal{F}_{\text{Class}} + \mathcal{R}(s_m)$ for all $m \geq n$. Thus, the normal form in $\mathcal{F}$ of all $t[s_m]$ with $m \geq n$ are the same, as we wished to show.

Remark 1. The idea of the proof of theorem 4 corresponds exactly to the intuition that during any computation, the oracle Φ is consulted a finite number of times and hence asked for a finite number of values. When the approximation s_n of Φ is great enough, we can substitute Φ with s_n and we obtain the same oracle values and hence the same results.

3. An Interactive Learning-Based Notion of Realizability for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$

In this section we introduce a learning-based notion of realizability for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$, Second Order Heyting Arithmetic plus Excluded Middle on Σ^0_1-formulas

$$\mathsf{EM}_1 := \forall x^{\mathtt{N}}.\ \exists y^{\mathtt{N}} \mathsf{P}_a xy \vee \forall y^{\mathtt{N}} \neg \mathsf{P}_a xy$$

and Skolem axioms

$$\mathsf{SK}_1 := \forall x^{\mathtt{N}} \forall y^{\mathtt{N}}.\ \mathsf{P}_a xy \to \mathsf{P}_a x \Phi_a(x)$$

then we prove our main Theorem, the Adequacy Theorem: *"if a closed arithmetical formula is provable in* $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$*, then it is realizable"*.

We first define the formal system $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$. We represent atomic predicates of $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ with (in general, non-computable) closed terms of $\mathcal{T}_{\text{Class}}$ of type `Bool`. Terms of $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ may include the function symbol Φ, denoting the Skolem function for $\exists y^{\mathtt{N}} \mathsf{P}_a xy$.

Remark 2. Our realizability can be formulated already for the standard language of Arithmetic: we add non computable functions to the language for greater generality and to interpret Skolem axioms in addition to EM_1. Of course, HAS already proves the comprehension axiom and thus $\mathsf{HAS} + \mathsf{EM}_1$ proves the existence of a Skolem function for $\exists y^{\mathtt{N}} \mathsf{P}_a xy$, which is the formula

$$A := \exists X.\ \mathsf{fun}(X) \wedge \forall x^{\mathtt{N}} \forall y^{\mathtt{N}}.\ \mathsf{P}_a xy \to (\exists z^{\mathtt{N}} (x, z) \in X \wedge \mathsf{P}_a xz)$$

with

$$\mathsf{fun}(X) \equiv \forall x^{\mathtt{N}} \exists y^{\mathtt{N}}.\ (x, y) \in X \wedge \forall z^{\mathtt{N}}.\ (x, z) \in X \to y = z$$

Indeed, as a witness for X one may take the formula

$$S(n, m) := \forall x^{\mathtt{N}}.\ \exists y^{\mathtt{N}} \mathsf{P}_a xy \to (\exists z^{\mathtt{N}}.\ (n, z) \in X \wedge z \leq y) \wedge \forall y^{\mathtt{N}} \neg \mathsf{P}_a xy \to (x, 0) \in X$$

Then using EM_1 one may prove $A[S/X]$ and thus $\exists X A$.

However, such an approach is fairly inefficient, because it requires to *define* a particular Skolem function, and it usually ends by defining a map always returning the minimum witness, as in the example, or by imposing some arbitrary criterion on the possible witnesses.

Instead, our approach of adding SK_1 is far more direct and efficient: our realizer of SK_1 will not waste resources by always constructing a minimum witness, but it will keep the first one it finds. The addition of non-computable functions to the language of Arithmetic requires in fact a non-trivial modification of the realizability semantics (for example, there is no formulation of Krivine classical realizability [25] realizing the formula SK_1 as it is written).

3.1. Language of $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$

We now define the language of the arithmetical theory $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$.

Definition 8 (Language of $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$). *The language $\mathcal{L}_{Class}$ of $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ is defined as follows.*

1. *The terms of $\mathcal{L}_{Class}$ are all $t \in \mathcal{T}_{Class}$, such that $t : \mathtt{N}$ and $FreeVar(t) \subseteq \{x_1^{\mathtt{N}}, \ldots, x_n^{\mathtt{N}}\}$ for some $x_1, \ldots, x_n$.*

2. *The atomic formulas of $\mathcal{L}_{\textit{Class}}$ are all $Q \in \mathcal{T}_{\textit{Class}}$ and Xt such that $Q : \texttt{Bool}$, $FreeVar(Q) \subseteq \{x_1^{\texttt{N}}, \ldots, x_n^{\texttt{N}}\}$ for some $x_1, \ldots, x_n$ and X is a predicate variable.*

3. *The formulas of $\mathcal{L}_{\textit{Class}}$ are built from atomic formulas of $\mathcal{L}_{\textit{Class}}$ by the connectives $\vee, \wedge, \rightarrow \forall, \exists$ as usual, with quantifiers possibly ranging over second-order predicate variables.*

4. *As usual, if A and B are formulas of $\mathcal{L}_{\textit{Class}}$ and X is a set variable, we denote with $A[\lambda z B(z)/X]$ the formula obtained from A by replacing all its atomic subformulas of the form Xt with $B[t/z]$ (without capturing free variables of B).*

We denote with $\perp$ the atomic formula $\texttt{False}$. We shall write natural number variables in lower case characters $x^{\texttt{N}}, y^{\texttt{N}}, z^{\texttt{N}}, \alpha^{\texttt{N}}$ and predicate variables in upper case characters X, Y, Z. We shall often omit the types of natural number variables, writing for instance $\forall x.A$ in place of $\forall x^{\texttt{N}} A$. If P is an atomic formula of $\mathcal{L}_{\text{Class}}$ in the free variables $x_1, \ldots, x_n$ and $t_1, \ldots, t_n$ are terms of $\mathcal{L}_{\text{Class}}$, with $P(t_1, \ldots, t_n)$ we shall denote the atomic formula $P[t_1/x_1, \ldots, t_n/x_n]$.

We defined $\Rightarrow_{\texttt{Bool}}$: $\texttt{Bool}, \texttt{Bool} \rightarrow \texttt{Bool}$ as a term implementing implication, therefore, to be accurate, formulas of the form $\mathsf{P}_a(t, u) \Rightarrow_{\texttt{Bool}} \mathsf{P}_a(t, \Phi_a t)$ are not an implication between two atomic formulas, but they are equal to the single atomic formula Q, where

$$Q := \Rightarrow_{\texttt{Bool}} (\mathsf{P}_a t u)(\mathsf{P}_a t(\Phi_a t))$$

Any atomic formula A of $\mathcal{L}_{\text{Class}}$ is a boolean term of $\mathcal{T}_{\text{Class}}$, therefore for any state s we may form the "approximation" $A[s] : \texttt{Bool}, A[s] \in \mathcal{F}$ of A. In $A[s]$ we replace the Skolem function Φ we have in A by its approximation s.

Our definition of realizability provides a formal semantics for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$, and therefore also for the more usual language of Arithmetic $\mathsf{HAS} + \mathsf{EM}_1$, in which all terms represent recursive maps.

From now onwards, for every pair of terms t_1, t_2 of system $\mathcal{F}$, we shall write $t_1 = t_2$ if they are the same term modulo the equality rules corresponding to the reduction rules of system $\mathcal{F}$ (equivalently, if they have the same normal form).

3.2. s-Saturated Sets

We now turn to the definition of a fundamental concept: the notion of s-saturated set of type A. The concept arises naturally if one tries, as a first thought, to define realizability for second-order formulas $\forall X A$ in terms of realizability of all formulas $A[\lambda z B(z)/X]$. Of course, such definition would not be a well-founded one with respect to the logical structure of formulas. In the case of second-order Heyting Arithmetic, one usually takes an extensional approach to solve this issue (see [29], for example). Since the computational meaning of a formula is determined by the set of its realizers, one replaces the quantification over formulas with quantification over saturated sets of lambda terms, i.e. arbitrary sets of terms closed under intensional equality. Then, one may define realizability for $\forall X A$ in terms of realizability of $A[F/X]$, for every function F from natural numbers to saturated set of terms. The idea is that F represents an abstract proposition over natural numbers.

Interactive realizability, however, is relativized to states, i.e. to terms $s : \texttt{N}^2 \rightarrow \texttt{N}$. Consequently, also intensional equality of terms of $\mathcal{F}_{\text{Class}}$ and hence saturation are relativized to states. The idea is that each world/state s determines a notion of equality between terms of $\mathcal{F}_{\text{Class}}$, because during computations one replaces Φ with its approximation s. That is, two terms t, u of $\mathcal{F}_{\text{Class}}$ are intensionally equal in the state s if $t[s] = u[s]$ in $\mathcal{F}$.

Definition 9 (s-Saturated Sets). *Let s be any state.*

1. *A s-*saturated set of type *A (A closed) is any set S of closed type-A terms of $\mathcal{F}_{\textit{Class}}$ such that if $t \in S$ and $t[s] = u[s]$ in $\mathcal{F}$, then $u \in S$.*

2. *For every type A, we denote with* $\mathsf{Sat}_A(s)$ *the set of all* s-saturated set of type A.

3. *Let* $\mathcal{L}^+_{\textit{Class}}$ *the language resulting from* $\mathcal{L}_{\textit{Class}}$ *by adding a constant symbol* $\dot{F}$ *for every function* $F : \mathbb{N} \to \mathsf{Sat}_A(s)$.

3.3. Interactive Realizability

For every formula A of $\mathcal{L}^+_{\text{Class}}$, we are now going to define what type $|A|$ a realizer of A must have.

Definition 10 (Types for realizers). *For each formula A of $\mathcal{L}^+_{\textit{Class}}$ we define a type $|A|$ of $\mathcal{F}_{\textit{Class}}$ by induction on A:*

1. $|P| = \mathtt{U}$,

2. $|Xt| = X$,

3. $|\dot{F}t| = C$ *if* $F : \mathbb{N} \to \mathsf{Sat}_C(s)$,

4. $|A \wedge B| = |A| \times |B|$,

5. $|A \vee B| = \mathtt{Bool} \times (|A| + |B|)$,

6. $|A \to B| = |A| \to |B|$,

7. $|\forall x A| = \mathtt{N} \to |A|$,

8. $|\forall X A| = \forall X |A|$,

9. $|\exists x A| = \mathtt{N} \times |A|$,

10. $|\exists X A| = \exists X |A|$

We now define the realizability relation $t \Vdash C$, where $t \in \mathcal{F}_{\text{Class}}$, $C \in \mathcal{L}^+_{\text{Class}}$ and $t : |C|$.

Definition 11 (Interactive Realizability). *Assume s is a state, t is a closed term of $\mathcal{F}_{\textit{Class}}$, $C \in \mathcal{L}^+_{\textit{Class}}$ is a closed formula, and $t : |C|$. We define first the relation $t \Vdash_s C$ by induction and by cases according to the form of C:*

1. $t \Vdash_s Q$ *if and only if:*
   - $t[s] = \overline{U}$ *implies that U is sound and* $\mathsf{dom}(U) \cap \mathsf{dom}(s) = \emptyset$
   - $t[s] = \varnothing$ *implies* $Q[s] = \mathtt{True}$

2. $t \Vdash_s \dot{F}u$ *if and only if for some numeral n, $u[s] = n$ and $t \in F(n)$*

3. $t \Vdash_s A \wedge B$ *if and only if* $\pi_0 t \Vdash_s A$ *and* $\pi_1 t \Vdash_s B$

4. $t \Vdash_s A \vee B$ *if and only if either* $\pi_0 t[s] = \mathtt{True}$, $\pi_1 t[s] = \iota_{0,|A|,|B|}(u)$ *and* $u \Vdash_s A$, *or* $\pi_0 t[s] = \mathtt{False}$, $\pi_1 t[s] = \iota_{1,|A|,|B|}(v)$ *and* $v \Vdash_s B$

5. $t \Vdash_s A \to B$ *if and only if for all u, if $u \Vdash_s A$, then $tu \Vdash_s B$*

6. $t \Vdash_s \forall xA$ *if and only if for all numerals* n, $tn \Vdash_s A[n/x]$

7. $t \Vdash_s \exists xA$ *if and only for some numeral* n, $\pi_0 t[s] = n$ *and* $\pi_1 t \Vdash_s A[n/x]$

8. $t \Vdash_s \forall XA$ *if and only if for every type* B *and* $F : \mathbb{N} \to \mathsf{Sat}_B(s)$, $tB \Vdash_s A[\dot{F}/X]$

9. $t \Vdash_s \exists XA$ *if and only if* $t = \langle B, u\rangle$ *and* $u \Vdash_s A[\dot{F}/X]$, *for some* $F : \mathbb{N} \to \mathsf{Sat}_B(s)$

We define $t \Vdash A$ *if and only if for all states* s, $t \Vdash_s A$.

The ideas behind the definition of $\Vdash_s$ are the following. A realizer is a term t of $\mathcal{F}_{\text{Class}}$, possibly containing the non-computable Skolem function Φ; if such function was computable, t would be an intuitionistic realizer. Since in general t is not computable, we calculate its approximation $t[s]$ at state s. t is an intelligent, self-correcting program, representing a proof/construction depending on the state s. The realizer *interacts* with the environment, which may provide a counter-proof, a counterexample invalidating the current construction of the realizer. But the realizer is always able to turn such a negative outcome into a positive information, which consists in some new piece of knowledge learned about the Skolem function Φ.

There are two important concepts that are useful to understand the interaction of a realizer with the environment: a realizer receives as input *tests* and produces as output *predictions*.

- *Predictions*.
  - A realizer t of $A \vee B$ uses s to predict which one between A and B is realizable: if $\pi_0 t[s] = \texttt{True}$ then A is realizable, and if $\pi_0 t[s] = \texttt{False}$ then B is realizable.
  - A realizer u of $\exists xA$ uses s to compute $\pi_0 u[s] = n$ and to predict that n is a witness for $\exists xA$ (i.e. that $A[n/x]$ is realizable).
  - A realizer of $\exists XA$ predicts the existence of a witness $F : \mathbb{N} \to \mathsf{Sat}_B(s)$ for $\exists XA$ (i.e. that $A[\dot{F}/X]$ is realizable).
- *Tests*.
  - A realizer t of a universal formula $\forall x^{\mathtt{N}} A$ or $\forall XA$ takes a natural number n or a saturated set F as a challenge coming from the environment to provide a construction of $A[n/x]$ or $A[\dot{F}/X]$, whose correctness will be tested at the end of computation.
  - A realizer of $A \to B$ takes a realizer of A as a challenge coming from the environment to provide a construction of B, whose correctness will be tested at the end of the computation.
  - A realizer of $A \wedge B$ may be challenged to construct A as well as B, and again the correctness of the construction will be tested at the end of computation.
  - A realizer of an atomic formula Q comes after a series of predictions and challenges that have been provided to test the construction of a complex formula; the realizer performs a final test and computes the formula Q in the state s as an experiment. Since predictions of realizers need not be always correct, it is possible that a realized atomic formula is actually false; we may have $t \Vdash_s Q$ and $Q[s] = \texttt{False}$ in $\mathcal{F}$. If Q, though predicted to be true, is instead false, then a counterexample has been encountered; this means that the approximation s of Φ is still inadequate. In this case, $t[s] \neq \varnothing$ by definition of $t \Vdash_s Q$, and the atomic realizer t takes s and extends it to a larger state s', union of s and $t[s]$. That is to say: the construction of a realizer is wrong in a particular state, the realizer must learn from its mistakes. The point is that after every learning, the actual state grows, and if we ask to the same realizer new predictions, we will obtain "better" answers.

Indeed, we can say more about this last point. Suppose for instance that $t \Vdash A \vee B$ and let $\{s_i\}_{i \in \mathbb{N}}$ be a w.i. sequence of states. Then, since $t : \texttt{Bool} \times |A| + |B|$, then $\pi_0 t : \texttt{Bool}$ is a closed term of $\mathcal{F}_{\text{Class}}$, converging in $\{s_i\}_{i\in\mathbb{N}}$ to a boolean by theorem 4; thus the sequence of predictions $\{\pi_0 t[s_i]\}_{i\in\mathbb{N}}$ eventually stabilizes, and hence a witness is eventually learned in the limit.

In the atomic case, in order to have $t \Vdash_s Q$, we stipulate as second requirement that if $t[s] = \varnothing$, then $Q[s] = \texttt{True}$ in $\mathcal{F}$. Together with the first requirement, that is to say: if $t[s]$ contains no *new* information about Φ for correcting wrong values of s, then t must ensure the truth of Q with respect to s. Hence search for truth will be for us *computation of a zero* – a s such that $t[s] = \varnothing$ – driven by the excluded-middle instances and the Skolem axioms used by proofs, rather than exhaustive search for counterexamples.

The next proposition tells that realizability at state s respects the notion of equality of $\mathcal{F}_{\text{Class}}$ terms, when the latter is relativized to state s. That is, if two terms are equal at the state s, then they realize the same formulas at the state s. Hence, the extensional notion of s-saturated set comprehends the intensional notion of realizability of a formula at state s.

Proposition 1 (Saturation). *The following hold:*

1. $\{t \mid t \Vdash_s A\} \in \mathsf{Sat}_{|A|}(s)$
2. *If $u[s] = v[s]$, then $\{t \mid t \Vdash_s B[u/x]\} = \{t \mid t \Vdash_s B[v/x]\}$*

Proof. By straightforward induction on A.

The next proposition tells that, in the context of realizability, quantification over maps from $\mathbb{N}$ to s-saturated sets definable through realizability is the same as quantification over definable sets of natural numbers. It is crucial in order to prove that $t \Vdash_s \forall X A$ implies $t \Vdash_s A[\lambda x B(x)/X]$ for every formula $B(x)$ in the only free variable x.

Proposition 2 (Comprehension). *Let $B(x)$ be a formula of $\mathcal{L}^+_{\mathit{Class}}$ in the only free natural number variable x. Define $\mathsf{B} : \mathbb{N} \to \mathsf{Sat}_{|B|}(s)$ as*

$$\mathsf{B} := n \mapsto \{t \mid t \Vdash_s B(n)\}$$

Then for every t

$$t \Vdash_s A[\dot{\mathsf{B}}/X] \iff t \Vdash_s A[\lambda x B(x)/X]$$

Proof. By induction on A.

1. $A = P(\vec{u})$, with P predicate of $\mathcal{T}_{\text{Class}}$. Then, $A[\dot{\mathsf{B}}/X] = A[\lambda x B(x)/X]$ and trivially

$$t \Vdash_s A[\dot{\mathsf{B}}/X] \iff t \Vdash_s A[\lambda x B(x)/X]$$

2. $A = Yu$, with Y predicate variable. Then, if $Y \neq X$, the thesis is trivial, since we would have

$$A[\dot{\mathsf{B}}/X] = Yu = A[\lambda x B(x)/X]$$

   So let us suppose $Y = X$ and that for some numeral n, $u[s] = n$. Then

$$A[\dot{\mathsf{B}}/X] = \dot{\mathsf{B}}u$$
$$A[\lambda x B(x)/X] = B(u)$$

   and so

$$\begin{aligned} t \Vdash_s A[\dot{\mathsf{B}}/X] &\iff t \Vdash_s \dot{\mathsf{B}}u \\ &\iff t \in \mathsf{B}(n) = \{v \mid v \Vdash_s B(n)\} \\ &\iff t \Vdash_s B(n) \\ &\overset{\text{prop. 1}}{\iff} t \Vdash_s B(u) = A[\lambda x B(x)/X] \end{aligned}$$

3. The other cases are straightforward.

Example 1. The most remarkable feature of our Realizability Semantics is the existence of a realizer for EM_1. Assume that P_a is a predicate of T and define

$$\mathsf{E}_a := \lambda\alpha^{\mathtt{N}}\langle \mathsf{X}_a\alpha,\ \mathsf{if}\ \mathsf{X}_a\alpha\ \mathsf{then}\ \iota_{0,A,B}(\langle \Phi_a\alpha,\ \varnothing\rangle)\ \mathsf{else}\ \iota_{1,A,B}(\lambda n^{\mathtt{N}}\ \mathsf{if}\ \mathsf{P}_a\alpha n\ \mathsf{then}\ \mathsf{mkupd}\, a\,\alpha\, n\ \mathsf{else}\ \varnothing)\rangle$$

with $A = \mathtt{N} \times \mathtt{U}$ and $B = \mathtt{N} \to \mathtt{U}$.
Indeed E_a realizes its associated instance of EM_1.

Proposition 3 (Realizer E_a of EM_1).

$$\mathsf{E}_a \Vdash \forall x.\ \exists y\ \mathsf{P}_a(x,y) \vee \forall y \neg_{\mathtt{Bool}} \mathsf{P}_a(x,y)$$

Proof. As in Aschieri and Berardi [4].

E_a works as follows. It uses the oracle X_a to make predictions about which one between $\exists y\, \mathsf{P}_a(m,y)$ and $\forall y \neg_{\mathtt{Bool}} \mathsf{P}_a(m,y)$ is true. X_a, in turn, relies on the approximation s of Φ to make its own prediction. If $\mathsf{X}_a m[s] = \mathtt{False}$, given any n, $\neg_{\mathtt{Bool}} \mathsf{P}_a(m,n)$ is predicted to be true; if it is not the case, we have a counterexample and $\mathsf{mkupd}\, a\, m\, n$ returns $\{(a,m,n)\}$, that is, it requires to correct the state s as to output n on input (a,m). On the contrary, if $\mathsf{X}_a m[s] = \mathtt{True}$, there is unquestionable evidence that $\exists y\, \mathsf{P}_a(m,y)$ holds; namely, $\mathsf{P}_a m s_a(m) = \mathtt{True}$ by definition 6 of X; then $\Phi_a m[s] = s(m)$ is computed and returned.
This is the basic mechanism by which learning is implemented: every state extension is linked with an assumption about an instance of EM_1 which has been used and turned out to be wrong (this is the only way to come across a counterexample); in next computations, the actual state will be bigger, the realizer will not do the same error, and hence will be "wiser".

3.4. Curry-Howard Correspondence for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$

In figure 2, we define a standard natural deduction system for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ (see [31], for example) together with a term assignment in the spirit of Curry-Howard correspondence for classical logic.

We replace purely universal axioms (i.e., Π^0_1 axioms) with Post rules, which are inferences of the form

$$\frac{\Gamma \vdash A_1\ \Gamma \vdash A_2\ \cdots\ \Gamma \vdash A_n}{\Gamma \vdash A}$$

where $A_1, \ldots, A_n, A$ are *first-order* atomic formulas of $\mathcal{L}_{\text{Class}}$ (i.e., without set variables) such that for every substitution $\sigma = [n_1/x_1^{\mathtt{N}}, \ldots, n_k/x_k^{\mathtt{N}}\ s/\Phi]$ of numerals $n_1, \ldots, n_k$ and states s, $A_1\sigma = \ldots = A_n\sigma = \mathtt{True}$ implies $A\sigma = \mathtt{True}$. Let now $\mathsf{eq} : \mathtt{N}^2 \to \mathtt{Bool}$ a term of Gödel's system T representing equality between natural numbers. Among the Post rules, we have the Peano axioms

$$\frac{\Gamma \vdash \mathsf{eq}\ \mathsf{S}(x)\,\mathsf{S}(y)}{\Gamma \vdash \mathsf{eq}\ x\, y} \qquad \frac{\Gamma \vdash \mathsf{eq}\ 0\,\mathsf{S}(x)}{\Gamma \vdash \bot}$$

and axioms of equality

$$\frac{}{\Gamma \vdash \mathsf{eq}\ x\, x} \qquad \frac{\Gamma \vdash \mathsf{eq}\, x\, y \quad \Gamma \vdash \mathsf{eq}\, y\, z}{\Gamma \vdash \mathsf{eq}\ x\, z} \qquad \frac{\Gamma \vdash A(x) \quad \Gamma \vdash \mathsf{eq}\, x\, y}{\Gamma \vdash A(y)}$$

and for every A_1, A_2 such that $A_1 = A_2$ is an equation of Gödel's system T (equivalently, A_1, A_2 have the same normal form in T), we have the rule

$$\frac{\Gamma \vdash A_1}{\Gamma \vdash A_2}$$

We add also have a Post rule

$$\frac{\Gamma \vdash A_1\ \Gamma \vdash A_2\ \cdots\ \Gamma \vdash A_n}{\Gamma \vdash A}$$

for every classical propositional tautology $A_1 \to \ldots \to A_n \to A$, where for $i = 1, \ldots, n$, A_i, A are atomic formulas obtained as combination of other atomic formulas by the Gödel's system T boolean connectives. As title of example, we have the rules

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash P} \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \Rightarrow_{\mathtt{Bool}} B} \frac{\Gamma \vdash A \wedge_{\mathtt{Bool}} B}{\Gamma \vdash A}$$

Finally, we have a Post rule of case reasoning for booleans. For any atomic formula P and any formula $A[P]$ we have:

$$\frac{\Gamma \vdash A[\mathtt{True}] \quad \Gamma \vdash A[\mathtt{False}]}{\Gamma \vdash A[P]}$$

The connectives $\vee_{\mathtt{Bool}}$ and $\vee$ have the same meaning but they are syntactically different: for every atomic formula P, we consider $P \vee_{\mathtt{Bool}} \neg_{\mathtt{Bool}} P$ an atomic formula and $P \vee \neg_{\mathtt{Bool}} P$ a compound formula. $P \vee_{\mathtt{Bool}} \neg_{\mathtt{Bool}} P$ is an axiom, while may derive $\mathsf{HA}^\omega \vdash P \vee \neg_{\mathtt{Bool}} P$ by case reasoning.

The term decorating the conclusion of a Post rule is of the form $u_1 \Cup \cdots \Cup u_n$. In this case, we have n different realizers, whose learning capabilities are put together through a sort of union. By Lemma 1, if $u_1 \Cup \cdots \Cup u_n[s] = \varnothing$, then $u_1[s] = \ldots = u_n[s] = \varnothing$, i.e. all u_i "have nothing to learn". In that case, each u_i must guarantee A_i to be true, and therefore the conclusion of the Post rule is true, because true premises $A_1, \ldots, A_n$ spell a true conclusion A.

We observe that the full *ex-falso-quodlibet* axiom is not in our system, since there are no rules neither axioms to derive $\bot \to Xt$. In fact, we have the ex-falso-quodlibet restricted to first-order formulas. In section Appendix A, we shall address this (not very significant) issue.

We now prove our main result of this section: every theorem of $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ is realizable. As usual in adequacy proofs for realizability, we prove a stronger version of the theorem, suitable to be proved by induction on proofs.

Theorem 5 (Adequacy Theorem). *Suppose that $\Gamma \vdash w : A$ in the system $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$, with $\Gamma = x_1 : A_1, \ldots, x_n : A_n$, and that the free variables of the formulas occurring in Γ and A are among $\alpha_1 : \mathtt{N}, \ldots, \alpha_k : \mathtt{N}, X_1, \ldots, X_m$. Fix any state s, numerals $n_1, \ldots, n_k$ and $F_1 \in \mathbb{N} \to \mathsf{Sat}_{B_1}(s), \ldots, F_m \in \mathbb{N} \to \mathsf{Sat}_{B_m}(s)$. For every formula C, let $\overline{C} := C[n_1/\alpha_1 \cdots n_k/\alpha_k\ \dot{F}_1/X_1 \cdots \dot{F}_m/X_m]$. If $t_1, \ldots, t_n$ are terms such that*

$$t_1 \Vdash_s \overline{A_1}, \ldots, t_n \Vdash_s \overline{A_n}$$

then

$$w[B_1/X_1 \cdots B_m/X_m][t_1/x_1^{|\overline{A_1}|} \cdots t_n/x_n^{|\overline{A_n}|}\ n_1/\alpha_1 \cdots n_k/\alpha_k] \Vdash_s \overline{A}$$

Proof. Notation: for any term v, we denote

$$v[B_1/X_1 \cdots B_m/X_m][t_1/x_1^{|\overline{A_1}|} \cdots t_n/x_n^{|\overline{A_n}|}\ n_1/\alpha_1 \cdots n_k/\alpha_k]$$

with $\overline{v}$. We have

$$|\overline{C}| = |C[\dot{F}_1/X_1 \cdots \dot{F}_m/X_m]| = |C|[B_1/X_1 \cdots B_m/X_m]$$

for all formulas C. We denote with $=$ the provable equality in $\mathcal{F}_{\text{Class}}$. We proceed by induction on w. Consider the last rule in the derivation of $\Gamma \vdash w : A$:

1. If it is the rule for variables, then for some i, $w = x_i^{|A_i|}$ and $A = A_i$. So $\overline{w} = t_i \Vdash_s \overline{A_i} = \overline{A}$.

2. If it is the $\wedge I$ rule, then $w = \langle u, t \rangle$, $A = B \wedge C$, $\Gamma \vdash u : B$ and $\Gamma \vdash t : C$. Therefore, $\overline{w} = \langle \overline{u}, \overline{t} \rangle$. By induction hypothesis, $\pi_0 \overline{w} = \overline{u} \Vdash_s \overline{B}$ and $\pi_1 \overline{w} = \overline{t} \Vdash_s \overline{C}$; so, by definition, $\overline{w} \Vdash_s \overline{B} \wedge \overline{C} = \overline{A}$.

3. If it is a $\wedge E$ rule, say left, then $w = \pi_0 u$ and $\Gamma \vdash u : A \wedge B$. So $\overline{w} = \pi_0 \overline{u} \Vdash_s \overline{A}$, because $\overline{u} \Vdash_s \overline{A} \wedge \overline{B}$ by induction hypothesis.

Contexts With Γ we denote contexts of the form $x_1 : A_1, \ldots, x_n : A_n$, with $x_1, \ldots, x_n$ proof variables and $A_1, \ldots, A_n$ formulas of $\mathcal{L}_{\text{Class}}$.

Axioms $\Gamma, x : A \vdash x^{|A|} : A$

Conjunction $\dfrac{\Gamma \vdash u : A \quad \Gamma \vdash t : B}{\Gamma \vdash \langle u, t \rangle : A \wedge B}$ $\quad \dfrac{\Gamma \vdash u : A \wedge B}{\Gamma \vdash \pi_0 u : A}$ $\quad \dfrac{\Gamma \vdash u : A \wedge B}{\Gamma \vdash \pi_1 u : B}$

Implication $\dfrac{\Gamma \vdash u : A \to B \quad \Gamma \vdash t : A}{\Gamma \vdash ut : B}$ $\quad \dfrac{\Gamma, x : A \vdash u : B}{\Gamma \vdash \lambda x^{|A|} u : A \to B}$

Disjunction Intro. $\dfrac{\Gamma \vdash u : A}{\Gamma \vdash \langle \mathtt{True}, \iota_{0,|A|,|B|}(u) \rangle : A \vee B}$ $\quad \dfrac{\Gamma \vdash u : A}{\Gamma \vdash \langle \mathtt{False}, \iota_{1,|A|,|B|}(u) \rangle : A \vee B}$

Disjunction Elim. $\dfrac{\Gamma \vdash u : A \vee B \quad \Gamma \vdash w_1 : A \to C \quad \Gamma \vdash w_2 : B \to C}{\Gamma \vdash (\pi_1 u)|C| w_1 w_2 : C}$

Universal Quantification (1) $\dfrac{\Gamma \vdash u : \forall \alpha^{\mathtt{N}} A}{\Gamma \vdash ut : A[t/\alpha^{\mathtt{N}}]}$ $\dfrac{\Gamma \vdash u : A}{\Gamma \vdash \lambda \alpha^{\mathtt{N}} u : \forall \alpha^{\mathtt{N}} A}$

where t is a term of $\mathcal{L}_{\text{Class}}$ and $\alpha^{\mathtt{N}}$ does not occur free in any formula B occurring in Γ.

Existential Quantification (1) $\dfrac{\Gamma \vdash u : A[t/\alpha^{\mathtt{N}}]}{\Gamma \vdash \langle t, u \rangle : \exists \alpha^{\mathtt{N}} A}$ $\quad \dfrac{\Gamma \vdash u : \exists \alpha^{\mathtt{N}} A \quad \Gamma \vdash t : \forall \alpha^{\mathtt{N}}. A \to C}{\Gamma \vdash t(\pi_0 u)(\pi_1 u) : C}$

where, in the second rule, $\alpha^{\mathtt{N}}$ is not free in C nor in any formula B occurring in Γ.

Universal Quantification (2) $\dfrac{\Gamma \vdash u : \forall X A}{\Gamma \vdash u|B| : A[\lambda x B(x)/X]}$ $\dfrac{\Gamma \vdash u : A}{\Gamma \vdash \Lambda X u : \forall X A}$

where $B(x)$ is a formula of $\mathcal{L}_{\text{Class}}$ and in the second rule X does not occur free in any formula occurring in Γ.

Existential Quantification (2) $\dfrac{\Gamma \vdash u : A[\lambda x B(x)/X]}{\Gamma \vdash \langle |B|, u \rangle : \exists X A}$ $\quad \dfrac{\Gamma \vdash u : \exists X A \quad \Gamma \vdash t : \forall X. A \to C}{\Gamma \vdash u|C|t : C}$

where, in the second rule, X is not free in C nor in any formula occurring in Γ.

Induction $\dfrac{\Gamma \vdash u : A(0) \quad \Gamma \vdash v : \forall \alpha. A(\alpha) \to A(\mathsf{S}(\alpha))}{\Gamma \vdash \lambda \alpha^{\mathtt{N}}. \mathsf{R}|A| u v \alpha : \forall \alpha A}$

Post Rules $\dfrac{\Gamma \vdash u_1 : A_1 \; \Gamma \vdash u_2 : A_2 \; \cdots \; \Gamma \vdash u_n : A_n}{\Gamma \vdash u_1 \Cup u_2 \Cup \cdots \Cup u_n : A}$

where $n > 0$ and $A_1, A_2, \ldots, A_n, A$ are first-order atomic formulas of $\mathcal{L}_{\text{Class}}$, and the rule is a Post rule for equality, for a Peano axiom, for a classical propositional tautology or for booleans.

Post Rules with no Premises $\dfrac{}{\Gamma \vdash \varnothing : A}$

where A is a first-order atomic formula of $\mathcal{L}_{\text{Class}}$ and an axiom of equality or a classical propositional tautology.

EM1 $\dfrac{}{\Gamma \vdash \mathsf{E}_a : \forall x. \, \exists y \; \mathsf{P}_a(x, y) \vee \forall y \neg_{\mathtt{Bool}} \mathsf{P}_a(x, y)}$

SK1 $\dfrac{}{\Gamma \vdash \lambda x^{\mathtt{N}} \lambda y^{\mathtt{N}} \mathsf{if} \; (\mathsf{P}_a x y \Rightarrow_{\mathtt{Bool}} \mathsf{P}_a x (\Phi_a x)) \; \mathsf{then} \; \varnothing \; \mathsf{else} \; (\mathsf{mkupd} \, a \, x \, y) : \forall x \forall y. \; \mathsf{P}_a(x, y) \Rightarrow_{\mathtt{Bool}} \mathsf{P}_a(x, \Phi_a x)}$

Figure 2: Term Assignement Rules for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$

4. If it is the $\to E$ rule, then $w = ut$, $\Gamma \vdash u : B \to A$ and $\Gamma \vdash t : B$. So $\overline{w} = \overline{u}\overline{t} \Vdash_s \overline{A}$, for $\overline{u} \Vdash_s \overline{B} \to \overline{A}$ and $\overline{t} \Vdash_s \overline{B}$ by induction hypothesis.

5. If it is the $\to I$ rule, then $w = \lambda x^{|B|} u$, $A = B \to C$ and $\Gamma, x : B \vdash u : C$. Suppose now that $t \Vdash_s \overline{B}$; we have to prove that $\overline{w} t \Vdash_s \overline{C}$. By induction hypothesis on u, $\overline{u} \Vdash_s \overline{C}$. By trivial equalities and

induction hypothesis

$$\begin{aligned}\overline{w}t &= (\lambda x^{|\overline{B}|}\overline{u})t \\ &= u[B_1/X_1 \cdots B_m/X_m][t/x^{|\overline{B}|}\ t_1/x_1^{|\overline{A_1}|} \cdots t_n/x_n^{|\overline{A_n}|}\ n_1/\alpha_1 \cdots n_k/\alpha_k] \\ &\Vdash_s \overline{C}\end{aligned}$$

Therefore, $\overline{w}t \Vdash_s \overline{C}$.

6. If it is a $\vee I$ rule, say left, then $w = \langle \mathtt{True}, \iota_{0,|B|,|C|}(u)\rangle$, $A = B \vee C$ and $\Gamma \vdash u : B$. So, $\overline{w} = \langle \mathtt{True}, \iota_{0,|\overline{B}|,|\overline{C}|}(\overline{u}), \rangle$ and hence $\pi_0\overline{w} = \mathtt{True}$. We indeed verify that $\overline{u} \Vdash_s \overline{B}$ with the help of induction hypothesis.

7. If it is a $\vee E$ rule, then
$$w = (\pi_1 u)|D|w_1 w_2$$
and $\Gamma \vdash u : B \vee C$, $\Gamma \vdash w_1 : B \to D$, $\Gamma \vdash w_2 : C \to D$, $A = D$.
Assume $\pi_0\overline{u}[s] = \mathtt{True}$. By inductive hypothesis $\overline{u} \Vdash_s \overline{B} \vee \overline{C}$. Therefore,
$$\pi_1\overline{u}[s] = \iota_{0,|\overline{B}|,|\overline{C}|}(v) \tag{1}$$
and $v \Vdash_s \overline{B}$. Hence, by definition 2 of $\iota_{0,|\overline{B}|,|\overline{C}|}$ and by (1)
$$\begin{aligned}\overline{w}[s] &= \iota_{0,|\overline{B}|,|\overline{C}|}(v)|\overline{D}|\overline{w}_1\overline{w}_2 \\ &= (\Lambda X \lambda f^{|\overline{B}| \to X} \lambda g^{|\overline{C}| \to X}\ fv)|\overline{D}|\overline{w}_1\overline{w}_2 \\ &= \overline{w}_1 v[s]\end{aligned}$$
By induction hypothesis $\overline{w}_1 \Vdash_s \overline{B} \to \overline{D}$. Therefore, $\overline{w}_1 v \Vdash_s \overline{D}$. Thus, by saturation (proposition 1), $\overline{w} \Vdash_s \overline{D}$.
Symmetrically, if $\pi_0\overline{u}[s] = \mathtt{False}$, we obtain again $\overline{w} \Vdash_s \overline{D}$.

8. If it is the (first order) $\forall E$ rule, then $w = ut$, $A = B[t/\alpha^{\mathtt{N}}]$ and $\Gamma \vdash u : \forall \alpha^{\mathtt{N}} B$. So, $\overline{w} = \overline{u}\overline{t}$. For some numeral n, we have $n = \overline{t}[s]$. By inductive hypothesis $\overline{u} \Vdash_s \forall \alpha \overline{B}$ and so $\overline{u}n \Vdash_s \overline{B}[n/\alpha^{\mathtt{N}}]$. Since $\overline{u}\overline{t}[s] = \overline{u}n[s]$, by saturation (proposition 1), we conclude that $\overline{u}\overline{t} \Vdash_s \overline{B}[\overline{t}/\alpha^{\mathtt{N}}]$.

9. If it is the (first order) $\forall I$ rule, then $w = \lambda\alpha^{\mathtt{N}} u$, $A = \forall\alpha^{\mathtt{N}} B$ and $\Gamma \vdash u : B$ (with α not occurring free in the formulas of Γ). So, $\overline{w} = \lambda\alpha^{\mathtt{N}}\overline{u}$, since $\alpha \neq \alpha_1, \ldots, \alpha_n$. Let n be a numeral; we have to prove that $\overline{w}n = \overline{u}[n/\alpha^{\mathtt{N}}] \Vdash_s \overline{B}[n/\alpha^{\mathtt{N}}]$, which amounts to show that the induction hypothesis can be applied to u. For this purpose, it is enough to observe that for $i = 1, \ldots, n$
$$t_i \Vdash_s \overline{A_i} = \overline{A_i}[n/\alpha^{\mathtt{N}}]$$

10. If it is the (first order) $\exists E$ rule, then
$$w = t(\pi_0 u)(\pi_1 u)$$
$\Gamma \vdash t : \forall\alpha^{\mathtt{N}}. B \to A$ and $\Gamma \vdash u : \exists\alpha^{\mathtt{N}} B$. Assume $n = \pi_0\overline{u}[s]$, for some numeral n. By induction hypothesis $\overline{t} \Vdash_s \forall\alpha^{\mathtt{N}}. \overline{B} \to \overline{A}$ and $\overline{u} \Vdash_s \exists\alpha^{\mathtt{N}} \overline{B}$. Therefore, $\pi_1\overline{u} \Vdash_s \overline{B}[n/\alpha^{\mathtt{N}}]$ and
$$\overline{t}n(\pi_1\overline{u}) \Vdash_s \overline{A}[n/\alpha^{\mathtt{N}}] = \overline{A}$$
by for α does not occur free in $\overline{A}$. Since $\overline{w}[s] = \overline{t}n(\pi_1\overline{u})[s]$, we obtain by saturation (proposition 1) $\overline{w} \Vdash_s \overline{A}$.

11. If it is the (first order) $\exists I$ rule, then $w = \langle t, u\rangle$, $A = \exists\alpha^{\mathtt{N}} B$, $\Gamma \vdash u : B[t/\alpha^{\mathtt{N}}]$. So, $\overline{w} = \langle \overline{t}, \overline{u}\rangle$; and, indeed
$$\pi_1 \overline{w} = \overline{u} \Vdash_s \overline{B}[\pi_0 \overline{w}/\alpha^{\mathtt{N}}] = \overline{B}[\overline{t}/\alpha^{\mathtt{N}}]$$
since by induction hypothesis $\overline{u} \Vdash_s \overline{B}[\overline{t}/\alpha^{\mathtt{N}}]$. We obtain the thesis by saturation (proposition 1).

12. If it is the (second order) $\forall E$ rule, then $w = u|C|$, $A = B[\lambda x C(x)/X]$ and $\Gamma \vdash u : \forall X B$. So, $\overline{w} = \overline{u}|\overline{C}|$. Define
$$\mathsf{C} := n \mapsto \{t \mid t \Vdash_s \overline{C}(n)\}$$
By inductive hypothesis $\overline{u} \Vdash_s \forall X \overline{B}$ and so $\overline{u}|\overline{C}| \Vdash_s \overline{B}[\dot{\mathsf{C}}/X]$. By proposition 2, we conclude that
$$\overline{u}|\overline{C}| \Vdash_s \overline{B}[\lambda x \overline{C}(x)/X]$$

13. If it is the (second order) $\forall I$ rule, then $w = \Lambda X u$, $A = \forall X B$ and $\Gamma \vdash u : B$ (and X does not occur free in the formulas of Γ). So, $\overline{w} = \Lambda X \overline{u}$, since $X \neq X_1, \ldots, X_m$. Let $F : \mathbb{N} \to \mathsf{Sat}_C(s)$; we have to prove that $\overline{w}C = \overline{u}[C/X] \Vdash_s \overline{B}[\dot{F}/X]$, which amounts to show that the induction hypothesis can be applied to u. For this purpose, it is enough to observe that for $i = 1, \ldots, n$
$$t_i \Vdash_s \overline{A_i} = \overline{A_i}[\dot{F}/X]$$

14. If it is the (second order) $\exists E$ rule, then
$$w = u|A|t$$
$\Gamma \vdash t : \forall X.\, B \to A$ and $\Gamma \vdash u : \exists X B$, with X not occurring free in A. By inductive hypothesis on u, $\overline{u} \Vdash_s \exists X \overline{B}$; hence $\overline{u} = \langle C, v\rangle$ and $v \Vdash_s \overline{B}[\dot{F}/X]$, for some $F : \mathbb{N} \to \mathsf{Sat}_C(s)$. By induction hypothesis on t, $\overline{t} \Vdash_s \forall X.\, \overline{B} \to \overline{A}$ and hence
$$\overline{t} C v \Vdash_s \overline{A}[\dot{F}/X] = \overline{A}$$
Moreover
$$\overline{w} = \langle C, v\rangle|\overline{A}|\overline{t} \overset{\text{def. } 2}{=} (\Lambda Y \lambda x^{\forall X.\, \overline{B} \to Y}\, x C v)|\overline{A}|\overline{t} = \overline{t} C v$$
We thus obtain by saturation (proposition 1)
$$\overline{w} \Vdash_s \overline{A}$$

15. If it is the (second order) $\exists I$ rule, then $w = \langle |C|, u\rangle$, $A = \exists X B$, $\Gamma \vdash u : B[\lambda x C(x)/X]$. So, $\overline{w} = \langle |\overline{C}|, \overline{u}\rangle$. Moreover, by induction hypothesis
$$\overline{u} \Vdash_s \overline{B}[\lambda x \overline{C}(x)/X]$$
Define
$$F := n \mapsto \{t \mid t \Vdash_s \overline{C}(n)\}$$
By proposition 1, $\dot{F} \in \mathsf{Sat}_{|\overline{C}|}(s)$ and by proposition 2
$$\overline{u} \Vdash_s \overline{B}[\dot{F}/X]$$
which is the thesis.

16. If it is the induction rule, then $w = \lambda\alpha^{\mathtt{N}} \mathsf{R}|B| u v \alpha$, $A = \forall \alpha B$, $\Gamma \vdash u : B(0)$ and $\Gamma \vdash v : \forall\alpha. B(\alpha) \to B(\mathsf{S}(\alpha))$. So, $\overline{w} = \lambda\alpha^{\mathtt{N}} \mathsf{R}|\overline{B}|\overline{u}\overline{v}\alpha$. Now let n be a numeral. A plain induction on n shows that
$$\overline{w} n = \mathsf{R}|\overline{B}|\overline{u}\overline{v} n \Vdash_s \overline{B}[n/\alpha]$$
for $\overline{u} \Vdash_s \overline{B}(0)$ and $\overline{v} i \Vdash_s \overline{B}(i) \to \overline{B}(\mathsf{S}(i))$ for all numerals i by induction hypothesis.

17. If it is a Post rule, then $w = u_1 \Cup u_2 \Cup \cdots \Cup u_n$ and $\Gamma \vdash u_i : A_i$. So, $\overline{w} = \overline{u}_1 \Cup \overline{u}_2 \Cup \cdots \Cup \overline{u}_n$. First, suppose that, for $i = 1, \ldots, n$, $\overline{u}_i[s] = \overline{U_i}$ and $\overline{w}[s] = \overline{U}$. By induction hypothesis, $\mathsf{dom}(\overline{U_i}) \cap \mathsf{dom}(s) = \emptyset$, and thus also $\mathsf{dom}(\overline{U}) \cap \mathsf{dom}(s) = \emptyset$. Suppose now that $\overline{U} = \varnothing$; then we have to prove that $\overline{A}[s] = \mathtt{True}$. It suffices to prove that
$$\overline{A}_1[s] = \overline{A}_2[s] = \cdots = \overline{A}_n[s] = \mathtt{True}$$
Indeed, we have $\overline{U_1} = \cdots = \overline{U_n} = \varnothing$ and by induction hypothesis $\overline{A}_1[s] = \cdots = \overline{A}_n[s] = \mathtt{True}$, since $\overline{u}_i \Vdash_s \overline{A}_i$, for $i = 1, \ldots, n$.

18. If is the excluded middle axiom EM_1, then $w \Vdash_s \mathsf{EM}_1$: this is Proposition 3.

19. If it is a Φ-axiom rule, then
$$w = \lambda x^{\mathtt{N}} \lambda y^{\mathtt{N}} \mathsf{if}\ (\mathsf{P}_a xy \Rightarrow_{\mathtt{Bool}} \mathsf{P}_a x(\Phi_a x))\ \mathsf{then}\ \varnothing\ \mathsf{else}\ (\mathsf{mkupd}\, a\, x\, y)$$
and
$$A = \forall x \forall y.\ \mathsf{P}_a(x, y) \Rightarrow_{\mathtt{Bool}} \mathsf{P}_a(x, \Phi_a x)$$
Let n, m be two arbitrary numerals. We have to prove that
$$\overline{w} nm \Vdash_s \mathsf{P}_a(n, m) \Rightarrow_{\mathtt{Bool}} \mathsf{P}_a(n, \Phi_a n)$$
There are two cases:
    (a) $\mathsf{P}_a(n, m) \Rightarrow_{\mathtt{Bool}} \mathsf{P}_a(n, s_a n) = \mathtt{True}$. In this case, $\overline{w} nm[s] = \varnothing$ and we have only to check that $\mathsf{dom}(s) \cap \mathsf{dom}(\varnothing) = \emptyset$, which is trivial.

    (b) $\mathsf{P}_a(n, m) \Rightarrow_{\mathtt{Bool}} \mathsf{P}_a(n, s_a n) = \mathtt{False}$. Then, $\mathsf{P}_a nm = \mathtt{True}$ and $\mathsf{P}_a n s_a(n) = \mathtt{False}$. Moreover
    $$\overline{w} nm[s] = \mathsf{mkupd}\, a\, n\, m = \overline{U}$$
    with $U = \{(a, n, m)\}$. We have first to check that U is sound (see definition 1): this follows from $\mathsf{P}_a nm = \mathtt{True}$. Then we have to verify that $\mathsf{dom}(s) \cap \mathsf{dom}(\overline{U}) = \emptyset$: indeed, $\mathsf{dom}(\overline{U}) = \{(a, n)\}$, and by definition 6, $\mathsf{P}_a n s_a(n) = \mathtt{False}$ implies $(a, n) \notin \mathsf{dom}(s)$. Finally, we have to check that $\overline{U} \neq \varnothing$, which is indeed true.

As corollary of the Adequacy theorem 5, we obtain the main theorem.

Theorem 6. *If A is a closed formula such that $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1 \vdash t : A$, then $t \Vdash A$.*

4. Witness Extraction with Interactive Realizability

In this section, we turn our attention to a crucial problem, which is an important test for any realizability semantics of classical Arithmetic: the witness extraction problem for Π^0_2-formulas. Given a realizer $t \Vdash \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$, where P is an atomic recursive predicate, one is asked to extract from t a non-trivial program taking as input a numeral n and yielding as output a witness for the formula $\exists y^{\mathtt{N}} Pmy$ (that is, a numeral m such that $Pnm = \mathtt{True}$). In the case of Interactive realizability, the problem of computing that witness can be reduced to finding a "zero" for a suitable term u of type $\mathtt{U}$, that is a state $s : \mathtt{S}$ such that $u[s] = \varnothing$. Indeed, given any numeral n and state s, the following implications hold:
$$t \ \Vdash\ \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$$
$$\Longrightarrow$$
$$t \ \Vdash_s\ \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$$
$$\Longrightarrow$$

$$tn \Vdash_s \exists y^{\mathtt{N}} Pny$$
$$\Longrightarrow$$
$$\pi_0(tn)[s] = m \wedge \pi_1(tn) \Vdash_s Pnm$$
$$\Longrightarrow$$
$$\pi_1(tn)[s] = \varnothing \implies Pnm = \mathtt{True}$$

Therefore, if s is a zero of $\pi_1(tn)$, then $\pi_0(tn)$ is equal in the state s to some witness m of the formula $\exists y^{\mathtt{N}} Pny$. Intuitively, a zero for $\pi_1(tn)$ represents a sufficient amount of information to compute the required witness.

Given a term $u : \mathtt{U}$ and a state s, we write $u[s] \preceq s$ if and only if $u[s] = \overline{U}$, for some update U, and $\mathsf{dom}(U) \subseteq \mathsf{dom}(s)$. If $u[s] \preceq s$, we call s a *prefix point* of u. A prefix point s of u represents a state that u is not able to extend with new information; in particular, if $u \Vdash_s Q$, for some atomic formula Q, then s is a zero of u. In the rest of the paper, we will show how to compute a prefix point for any closed term $v : \mathtt{U}$ of system $\mathcal{F}_{\text{Class}}$. We propose two methods for accomplishing this task; they construct exactly the same witnesses and implement the very same idea, but they are radically different from the efficiency point of view. The *Iterative Method* is very simple, easy to understand and provides an algorithm that can be directly written in pure lambda calculus; but the proof of correctness uses the Axiom of Choice. The *State-Extending-Continuation-Passing-Style Method* is more sophisticated, dramatically more efficient and provides an algorithm that can be represented in System $\mathcal{F}$; the correctness proof is purely intuitionistic. But the first method is useful to understand the second, which is nothing but an optimization of the first.

Remark 3. For simplicity, for the rest of the paper we assume that the product and boolean types of $\mathcal{F}$, its booleans, pairs and if constructs are not primitive notions, but they are all defined (in the standard way, see for example Girard [20]).

4.1. The Iterative Method

If $t \Vdash \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$ and $u := \pi_0(tn)$, the interpretation of $u : \mathtt{U}$ is that of a state-extending operator. That is, given a state s, since $u[s]$ represents new information improving the approximation s of the oracle Φ, it is natural to associate to u the state-extending operator $\lambda s^{\mathtt{S}}.\, s \oplus u[s]$. The idea of the Iterative Method is to start from an arbitrary state and apply this state-extending operator until a zero of u is reached. Such series of state extensions represents a terminating learning process.

Theorem 7 (Zero Theorem). *Let P be an atomic predicate of Gödel's T and suppose $t \Vdash \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$. Let n be any numeral, define $u := \pi_0(tn)$ and let s be any state. Define, by induction on n, a sequence $\{s_n\}_{n\in\mathbb{N}}$ of states as follows:*

$$s_0 := s$$
$$s_{n+1} := s_n \oplus u[s_n] \overset{def\ 6}{=} \lambda x^{\mathtt{N}} \lambda y^{\mathtt{N}}\ \mathsf{if}\ (\mathsf{is}\, u[s_n]\, x\, y)\ \mathsf{then}\ (\mathsf{get}\, u[s_n]\, x\, y)\ \mathsf{else}\ s_n(x,y)$$

Then, $\{s_n\}_{n\in\mathbb{N}}$ is weakly increasing and there exists a number k such that $u[s_k] = \varnothing$.

Proof. We first prove that $s_0, s_1, s_2, \ldots$ is a weakly increasing chain. Suppose $s_i(a,n) \neq s_{i+i}(a,n)$: we have to prove that $(a,n) \in \mathsf{dom}(s_{i+1})$ and $(a,n) \notin \mathsf{dom}(s_i)$. By definition of s_{i+1}, if it were $\mathsf{is}\, u[s_i]\, a\, n = \mathtt{False}$, then we would have $s_i(a,n) = s_{i+i}(a,n)$, contradiction. Thus, $\mathsf{is}\, u[s_i]\, a\, n = \mathtt{True}$, and if we choose an update U such that $\overline{U} = u[s_i]$, we have:

$$\mathsf{is}\, \overline{U}\, a\, n = \mathtt{True}$$

that is, $(a,n) \in \mathsf{dom}(U)$, and for some m, $(a,n,m) \in U$. If we let $l = u[s_i]$, then $u \Vdash_{s_i} Pnl$; this means that U is sound and $\mathsf{dom}(s_i) \cap \mathsf{dom}(U) = \emptyset$. From $\mathsf{dom}(s_i) \cap \mathsf{dom}(U) = \emptyset$ and $(a,n) \in \mathsf{dom}(U)$ we obtain $(a,n) \notin \mathsf{dom}(s_i)$. From U is sound and $(a,n,m) \in U$ we obtain $\mathsf{P}_a nm = \mathtt{True}$. By definition,

$$s_{i+1}(a,n) = \mathsf{get}\, u[s_i]\, a\, n = \mathsf{get}\, \overline{U}\, a\, n = m$$

Therefore, $s_{i+1}(a,n) = m$ and by definition 6 of dom, we have that $(a,n) \in \mathsf{dom}(s_{i+1})$. We conclude that $s_0, s_1, s_2, \ldots$ is weakly increasing.
Now, by theorem 4, u converges over the chain $\{s_i\}_{i\in\mathbb{N}}$: there exists $k \in \mathbb{N}$ such that for every $j \geq k$, $u[s_j] = u[s_k]$. By choice of k

$$\begin{aligned} s_{k+1} \oplus u[s_{k+1}] &= (s_k \oplus u[s_k]) \oplus u[s_{k+1}] \\ &= (s_k \oplus u[s_k]) \oplus u[s_k] \\ &= s_k \oplus u[s_k] \\ &= s_{k+1} \end{aligned}$$

and hence it must be that $u[s_{k+1}] = \varnothing$, which is the thesis.

We are now able to extract from any realizer $t \Vdash \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$, with P atomic predicate of Gödel's T, a recursive map f from the set of numerals to the set of numerals, such that $Pnf(n) = \mathtt{True}$ for all numerals n.

Theorem 8 (Witness Extraction via the Iterative Method). *Suppose that $t \Vdash \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$, with P atomic predicate of Gödel's T. Then, from t one can effectively define a recursive function f from the set of numerals to the set of numerals such that for every numeral n, $Pn(f(n)) = \mathtt{True}$.*

Proof. Let

$$v := \lambda m^{\mathtt{N}}\ \pi_1(tm)$$

v is of type $\mathtt{N} \to \mathtt{U}$. By the Zero Theorem 7, there exists a recursive function zero from the set of numerals to the set of states such that $vn[\mathsf{zero}(n)] = \varnothing$ for every numeral n. Define the function

$$f := m \mapsto \pi_0(tm)[\mathsf{zero}(m)]$$

and fix a numeral n. By unfolding the definition of realizability with respect to $\mathsf{zero}(n)$, we have that

$$tn \Vdash_{\mathsf{zero}(n)} \exists y^{\mathtt{N}} Pny$$

and hence

$$\pi_1(tn) \Vdash_{\mathsf{zero}(n)}\ Pn(f(n))$$

that is to say

$$vn[\mathsf{zero}(n)] = \varnothing \implies Pn(f(n)) = \mathtt{True}$$

and therefore

$$Pn(f(n)) = \mathtt{True}$$

which is the thesis.

The recursive function f of theorem 8 is not directly representable in $\mathcal{F}$, since it uses unbounded iteration to compute zeros of atomic realizers, as in the proof of the zero theorem 7. However, f is easily representable in pure lambda calculus, by means of any fixed point combinator. It is remarkable that one does not need control operators at all and f can be written directly in a purely functional language (differently from what happens with Krivine classical realizability [25, 27]). f can even be represented: i) in Spector's system B (see Aschieri[2]); ii) in pure system $\mathcal{F}$, by using the methods of Aschieri [3] and of this paper: this is possible because the number of times that a state-extending operator is iterated can be bounded constructively.

One may then wonder how it is implemented backtracking in our extracted programs: control operators have precisely that function in [25]. The answer is that backtracking is implemented automatically in the iteration of the state-extending operator u of theorem 7. More precisely, let us consider the chain $\{s_n\}_{n\in\mathbb{N}}$ defined in the statement of theorem 7. When u is evaluated in s_n, and it is different from $\varnothing$, then $\mathsf{dom}(s_{n+1}) = \mathsf{dom}(s_n \oplus u[s_n])$ is strictly larger than $\mathsf{dom}(s_n)$. In particular, for some pair of numerals (i,j), there is a k such that (i,j,k) belongs to the update denote by $u[s_n]$ and it holds that $k = s_{n+1}(i,j) \neq s_n(i,j)$.

The normalization of $u[s_{n+1}]$ is perfectly equal to the normalization of $u[s_n]$ up to the first point in which $s_{n+1}(i,j)$ is computed: this is the *backtracking point*. Instead of putting control operators in $u[s_n]$ to save all possible backtracking points and to jump directly to the right one (which is discovered by reducing $u[s_n]$ to an update), the Iterative Method recomputes $u[s_{n+1}]$ from scratch. In this way, the backtracking point is trivially rencountered, but with a great waste of resources.

The aim of the next section is to define in $\mathcal{F}$ a more efficient program, that is able to jump to the right backtracking point while iterating the state-extending operator u.

4.2. The State-Extending-Continuation-Passing-Style Method

The State-Extending-Continuation-Passing-Style method defines a translation $[\![\_]\!]$ of the terms of System $\mathcal{F}_{\text{Class}}$ into $\mathcal{F}$. $[\![t]\!]$ will be a program that manipulates continuation on states, and uses them to implement backtracking. As a tool for describing and proving the properties of the translation we shall use, once again, (intuitionistic) realizability combined with a constructive notion of forcing. The use of the notation $[\![\_]\!]$ is not casual, since this translation has a nice model theoretic explanation in terms of non-standard natural numbers (see Aschieri [3]).

The idea of the State-Extending-Continuation-Passing-Style method is to interpret the oracle Φ_a as a kind of control operator. During computations, whenever it is asked the value of $\Phi_a(n)$, one uses the approximation $s_a(n)$ given by some state s. But if the final value of a computation is an update U, it might contain a triple (a,n,m): the value of s at point (a,n) was incorrect and must be corrected with the value m. The idea is that the program interpreting $\Phi_a(n)$ should look at the future of the computation: if any state $s' > s$ is ever encountered such that a triple $(a,n,m) \in \mathsf{dom}(s')$, then it returns m, otherwise it return $s_n(m)$. The future of the computation, as usual, is given by a state-extending continuation.

Definition 12 (State-Extending Continuations). *A closed term $k : \mathtt{S} \to \mathtt{S}$ of $\mathcal{F}$ is said to be* state-extending continuation *if for all states s, $s \leq k(s)$. We denote with $\mathbb{K}$ the set of all state-extending continuations.*

There is, however, a complication. The idea explained above works with terms of the form $\Phi_a(n)$, where n is a numeral, but not with terms of the form $\Phi_a(t)$, when $t : \mathtt{N}$ is a term of $\mathcal{F}_{\text{Class}}$. The problem arises when computing $t[s] = n$. If the domain of the state s' returned by the current continuation $k \in \mathbb{K}$ applied to the current state s contains (a,n,m), then one has to evaluate once again t in the state s'. If $t[s'] = n$, the program interpreting $\Phi_a(t)$ returns m. But it may happen that $t[s'] = n' \neq n$. In this case, the program interpreting $\Phi_a(t)$ must look again in the future $s'' = k(s')$, check whether $(a,n',m) \in \mathsf{dom}(s'')$ and so on... How do we know, *constructively*, that one cannot end stuck in an infinite loop? The key idea is that, starting from any state, one can *force* t to have a value which is going to be stable in all the future states of the computation. To express this precisely, we need the notion of *modulus of forcing*.

4.2.1. Moduli of Forcing and Constructive Forcing at Type $\mathtt{N}$

Classically, a term $t : \mathtt{N}$ of System $\mathcal{F}_{\text{Class}}$ denotes a natural number, when a Skolem function is chosen for interpreting Φ. But from the constructive point of view, t is not a natural number, for we cannot compute its value. Worse of all, t may have infinite values, one for each state! More precisely, in our realizability model, there is no way whatsoever even in the classical sense to associate to t a definite natural number as a value: we consider as states all possible computable approximations of all possible Skolem functions that may interpret Φ, and thus different states may approximate different functions. Constructively, however, we can *force t to behave like a single natural number* for "the rest of computation". In particular, a state s *forces t to belong to* $\mathbb{N}$ if for all $k \in \mathbb{K}$ there exists a state $s' \geq s$ such that t is equal to the same numeral when evaluated in any state r such that $s' \leq r \leq k(s')$. In other words, there is an extension s' of s such that t will have the same values in all the states of the computation produced after s'.

Intuitively, a modulus of forcing represents the computational content of that notion of forcing. The concept of modulus of forcing (due to Berardi) has been extensively motivated and exploited in Aschieri [3] in order to define a state-extending-continuation-passing-syle translation of System $\mathcal{T}_{\text{Class}}$ into Gödel's System $\mathcal{T}$.

Definition 13 (Modulus of Forcing and Constructive Forcing at Type $\mathtt{N}$**).** *Let t be a closed term of $\mathcal{F}_{\scriptscriptstyle Class}$ and of type* $\mathtt{N}$*.*

1. *For all states s, r, we define*

$$t \downarrow [s, r] \stackrel{def}{\equiv} \forall q^{\mathtt{S}}.\ s \leq q \leq r \to t[q] = t[s]$$

2. *A term $\mathcal{M} : (\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})$ of $\mathcal{F}$ is a* modulus of forcing *for t if:*

   (a) $\forall k \in \mathbb{K}.\ \mathcal{M}_k \in \mathbb{K}$

   (b) $\forall k \in \mathbb{K}.\ \forall s^{\mathtt{S}}.\ t \downarrow [\mathcal{M}_k(s), k(\mathcal{M}_k(s))]$

3. *Whenever $\mathcal{M}$ is a modulus of forcing for t, we write $\mathcal{M} \Vdash t \in \mathtt{N}$ and we say that $\mathcal{M}$* forces t to belong *to* $\mathtt{N}$*.*

As in [3], we now show that given two terms t_1 and t_2, if each one of them has a modulus of forcing, then there is a modulus of forcing that works simultaneously for both of them. In particular, we can define a binary operation $\sqcup$ between moduli of forcing such that, for every pair of moduli $\mathcal{M}, \mathcal{N}$, $\mathcal{M} \sqcup \mathcal{N}$ is "more general" than both $\mathcal{M}$ and $\mathcal{N}$. Here, for every $\mathcal{M}_1, \mathcal{M}_2$, we call $\mathcal{M}_2$ more general than $\mathcal{M}_1$, if for every term t, if $\mathcal{M}_1$ is a modulus of forcing for t then also $\mathcal{M}_2$ is a modulus of forcing for t. We this terminology, we may see $\mathcal{M} \sqcup \mathcal{N}$ as an upper bound of the set $\{\mathcal{M}, \mathcal{N}\}$, with respect to the partial order induced by the relation "to be more general than". For any terms $u, v : \mathtt{S}$, we define $u \circ v := \lambda s^{\mathtt{S}}.\ u(v[s])$.

Proposition 4 (Joint Forcing). *Suppose $\mathcal{M} \Vdash t_1 \in \mathtt{N}$ and $\mathcal{N} \Vdash t_2 \in \mathtt{N}$. Define*

$$\mathcal{M} \sqcup \mathcal{N} := \lambda k^{\mathtt{S} \to \mathtt{S}} \lambda s^{\mathtt{S}} \mathcal{N}_h(\mathcal{M}_{k \circ \mathcal{N}_k}(z))$$

Then $\mathcal{M} \sqcup \mathcal{N} \Vdash t_1 \in \mathtt{N}$ and $\mathcal{M} \sqcup \mathcal{N} \Vdash t_2 \in \mathtt{N}$.

Proof. Set

$$\mathcal{L} := \mathcal{M} \sqcup \mathcal{N}$$

First, we check property (1) of definition 13. For all $k \in \mathbb{K}$, $\mathcal{N}_k \in \mathbb{K}$ by definition 13 point (1) and so $k \circ \mathcal{N}_k \in \mathbb{K}$. Thus, for all $k \in \mathbb{K}$ and state s

$$\mathcal{L}_k(s) = \mathcal{N}_k(\mathcal{M}_{k \circ \mathcal{N}_k}(s)) \geq s$$

since $\mathcal{M}$ has property (1) of definition 13 and hence $\mathcal{M}_{k \circ \mathcal{N}_k} \in \mathbb{K}$. Therefore, for all $k \in \mathbb{K}$, $\mathcal{L}_k \in \mathbb{K}$ and we are done.
Secondly, we check property 2 of definition 13. Fix a continuation $k \in \mathbb{K}$ and a state s. Since $\mathcal{M} \Vdash t_1 \in \mathtt{N}$, we have that

$$t_1 \downarrow [\mathcal{M}_{k \circ \mathcal{N}_k}(s), k \circ \mathcal{N}_k(\mathcal{M}_{k \circ \mathcal{N}_k}(s))] \tag{2}$$

Moreover, since $\mathcal{N} \Vdash t_2 \in \mathtt{N}$, we have

$$t_2 \downarrow [\mathcal{N}_k(\mathcal{M}_{k \circ \mathcal{N}_k}(s)), k(\mathcal{N}_k(\mathcal{M}_{h \circ \mathcal{N}_k}(s)))] \tag{3}$$

But the starting point of the interval in (3) is greater or equal to the starting point of the interval in (2), for $\mathcal{N}_k \in \mathbb{K}$, while their ending points are equal. Hence also

$$t_1 \downarrow [\mathcal{N}_k(\mathcal{M}_{k \circ \mathcal{N}_k}(s)), k(\mathcal{N}_k(\mathcal{M}_{k \circ \mathcal{N}_k}(s)))]$$

and hence both t_1 and t_2 are constant in the interval $[\mathcal{L}_k(s), k(\mathcal{L}_k(s))]$ by definition of $\mathcal{L}$.

The notion of modulus of forcing, as explained in [3], gives rise to an interesting non-standard model of Gödel's System $\mathcal{T}$. In that model, integers are pairs $(\mathcal{M}, f)$, where f is a function from states to numbers and $\mathcal{M}$ is a modulus of forcing of such function. It is natural to associate to every term of $\mathcal{F}_{\text{Class}}$ the non-standard numbers it represents.

Definition 14 (Non-Standard Natural Numbers). *We define*

$$^{*}\mathtt{N} := \{\langle \mathcal{M}, t\rangle \mid \mathcal{M} = \langle \mathcal{L}, g\rangle, g : \mathtt{S} \to \mathtt{N} \in \mathcal{F}, \forall s^{\mathtt{S}}\, g(s) = t[s] \text{ and } \mathcal{L} \Vdash t \in \mathtt{N}\}$$

(all terms in the definition are supposed to be closed).

The notion of modulus of forcing, if extended to terms of type $\mathtt{U}$, is sufficient to write down in $\mathcal{F}$ a program that implements the Iterative Method. Indeed, that is the approach followed in [3]. But in order to perform better, we need the additional notion of *modulus of prefix point*.

4.2.2. Moduli of Prefix Point and Constructive Forcing at Type $\mathtt{U}$

An idea which is related to the concept of modulus of prefix point was introduced in Berardi and De' Liguoro [10]. Interestingly, as well as the notion of modulus of forcing is not sufficient to write an efficient optimization of the Iterative Method, the notion of modulus of prefix point alone misses the goal. But when combined together, the two notions are able to produce the desired result.

Intuitively, a modulus of prefix point represents the computational content of the following constructive notion of forcing for terms t of type $\mathtt{U}$: a state s *forces* t to belong to $\mathtt{U}$ if for all $k \in \mathbb{K}$ there exists $s' \geq s$ such that $k(s')$ is a prefix point of t. In other words, there is an extension s' of s such that the final state of the computation that starts from s' will be a prefix point of t. The idea is that the interpretation of the type $\mathtt{U}$ is the set of all terms of type $\mathtt{U}$ having "stable" prefix points reachable from any state.

Definition 15 (Modulus of Prefix Point and Constructive Forcing at Type $\mathtt{U}$). *A term $\mathcal{M} : (\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})$ of $\mathcal{F}$ is a* modulus of prefix point *for t if:*

1. $\forall k \in \mathbb{K}.\ \mathcal{M}_k \in \mathbb{K}$

2. $\forall k \in \mathbb{K}\ \ \forall s^{\mathtt{S}}.\ s' = k(\mathcal{M}_k(s)) \implies t[s'] \preceq s'$

Whenever $\mathcal{M}$ is a modulus of prefix point for t, we write $\mathcal{M} \Vdash t \in \mathtt{U}$ and we say that $\mathcal{M}$ forces t to belong to $\mathtt{U}$.

It is natural to associate to every term of type $\mathtt{U}$ all its possible moduli of prefix point.

Definition 16 (Non-Standard Updates). *We define*

$$^{*}\mathtt{U} := \{\langle \mathcal{M}, t\rangle \mid \mathcal{M} \Vdash t \in \mathtt{U}\}$$

(all terms in the definition are supposed to be closed)

4.2.3. Constructive Forcing at All Types

Now that we have a notion of forcing at base types, we lift it to all types by means of intuitionistic realizability. Since we are in an impredicative setting, we first have to put some general conditions that our notion of forcing/realizability must satisfy. As usual, we define a notion of candidate, which in our case is a set of pair of terms closed by the intensional equality of $\mathcal{F}$ and equipped with a monad operation (in the sense of Moggi [28]). This operation takes as input a non-standard natural number $(\mathcal{M}, g)$ and term $\mathcal{L} : \mathtt{N} \to U$ and return a term of type U. Intuitively, it means that: i) in some way one may apply functions which takes as arguments natural numbers to non-standard numbers; ii) if one can force some property to be true for all natural numbers, then one can force it to be true for all non-standard natural numbers. The property ii) will be essential when interpreting recursion over non-standard numbers.

Definition 17 (Forcing Candidates). *Let $\mathcal{C}$ be a set of terms of system $\mathcal{F}_{\text{Class}}$ of type $U \times V$, closed under the intensional equality of $\mathcal{F}$ (i.e. if $t \in \mathcal{C}$ and $t = t'$, then $t' \in \mathcal{C}$). Let*

$$\mathcal{N} : \mathsf{Cand}_U := ((\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})) \to (\mathtt{S} \to \mathtt{N}) \to (\mathtt{N} \to U) \to U$$

be a closed term of system $\mathcal{F}$. We define the relation $\mathcal{N} \Vdash \mathcal{C} \in \mathsf{Cand}_{U\times V}$ – representing the notion "$\mathcal{N}$ realizes that $\mathcal{C}$ is a forcing candidate *of type $U \times V$" – to hold whenever the following three conditions:*

1. $\mathcal{M} \Vdash t \in \mathtt{N}$;
2. *for all states s, $g(s) = t[s]$;*
3. *$\mathcal{L} : \mathtt{N} \to U$ is a term of $\mathcal{F}$ such that for every numeral m, $\langle \mathcal{L}m, u(m)t_1 \dots t_n\rangle \in \mathcal{C}$*

imply that

$$\langle \mathcal{N}\mathcal{M}g\mathcal{L}, u(t)t_1 \dots t_n\rangle \in \mathcal{C}$$

(where $t_1 \dots t_n$ is any sequence of types or terms). If $\mathcal{N} \Vdash \mathcal{C} \in \mathsf{Cand}_{U\times V}$, then $\mathcal{C}$ is said to be a forcing candidate *(of type $U \times V$). Whenever the type of the terms in $\mathcal{C}$ is known, we shall just write $\mathcal{N} \Vdash \mathcal{C} \in \mathsf{Cand}$.*

For convenience, we now introduce to the language a type constant symbol for every forcing candidate. For each type T in this extended language, we define two types of $\mathcal{F}$: a type $|T|$, which represents the type in $\mathcal{F}$ of a term of type T in the extended language, and a type $[\![T]\!]$, which represents the interpretation of T in the realizability model.

Definition 18 (Extended Types, Interpretation of Types). *We define:*

1. *For every forcing candidate $\mathcal{C}$ of type $U \times V$, we introduce a type constant $\dot{\mathcal{C}}$ to the language of types of system $\mathcal{F}_{\text{Class}}$ and we define $|\dot{\mathcal{C}}| := V$. An* extended type *is defined by induction as either a type constant $\dot{\mathcal{C}}$, or a type variable, or an expression $U \to V$ or $\forall X\, U$, with U and V extended types and X a type variable. We define $|\forall X\, U| := \forall X\, |U|$, $|U \to V| := |U| \to |V|$, $|X| := X$, for X variable.*

2. *For every extended type T, we define a type $[\![T]\!]$ by induction on T as follows.*
   (a) *$T = X$, with X atomic. Then*
   $$[\![X]\!] := X$$
   (b) *$T = \mathtt{N}$. Then*
   $$[\![\mathtt{N}]\!] := ((\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})) \times (\mathtt{S} \to \mathtt{N})$$
   (c) *$T = \mathtt{U}$. Then*
   $$[\![\mathtt{U}]\!] := (\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})$$
   (d) *$T = \dot{\mathcal{C}}$, where $\mathcal{C}$ is a forcing candidate of type $U \times V$. Then*
   $$[\![\dot{\mathcal{C}}]\!] := U$$
   (e) *$T = A \to B$. Then*
   $$[\![A \to B]\!] := [\![A]\!] \to [\![B]\!]$$
   (f) *$T = \forall X\, A$. Then*
   $$[\![\forall X\, A]\!] := \forall X.\, \mathsf{Cand}_X \to [\![B]\!]$$

We are now ready to define our forcing/realizability relation. Intuitively, $\mathcal{M} \Vdash t \downarrow T$ means that $\mathcal{M}$ *forces the term $t \in \mathcal{F}_{\text{Class}}$ to behave like a computable functional of higher-type.* The notion can be seen as a constructive extension of the standard Tait-Girard computability/reducibility predicate, defined in order to deal with non-computable functionals. It also appears as a kind of constructively refined version of Goodman's forcing [16] (see section 5 for more about this point). Here, we could have also talked about "higher-type convergence" since the notion can be as a tool for reformulating and proving constructively the Converge Theorem 4.

Definition 19 (Constructive Forcing at Higher Types). *We define:*

Let T be a closed extended type, and $\mathcal{M} : [\![T]\!]$ and $t : |T|$ be closed terms respectively of $\mathcal{F}$ and $\mathcal{F}_{Class}$. We define the relation $\mathcal{M} \Vdash t \downarrow T$ – representing the notion "$\mathcal{M}$ forces t to be a computable functional of type *T" – by induction on the type T as follows:*

1. *$T = \mathtt{N}$. Then,*

$$\mathcal{M} \Vdash t \downarrow \mathtt{N} \iff \mathcal{M} = \langle \mathcal{L}, g \rangle, \forall s^{\mathtt{S}} g(s) = t[s] \text{ and } \mathcal{L} \Vdash t \in \mathtt{N} \text{ (definition 13)}$$

2. *$T = \mathtt{U}$. Then, $\mathcal{M} \Vdash t \downarrow \mathtt{U}$ is given by definition 15. That is,*

$$\mathcal{M} \Vdash t \downarrow \mathtt{U} \iff \mathcal{M} \text{ is a moduls of prefix point for } t$$

3. *$T = \dot{\mathcal{C}}$, $|\dot{\mathcal{C}}| = U$. Let $\mathcal{M} : U$. Then*

$$\mathcal{M} \Vdash t \downarrow \dot{\mathcal{C}} \iff \langle \mathcal{M}, t \rangle \downarrow \mathcal{C}$$

4. *$T = A \to B$. Let $\mathcal{M} : [\![A \to B]\!]$. Then*

$$\mathcal{M} \Vdash t \downarrow A \to B \iff (\forall u.\, \mathcal{N} \Vdash u \downarrow A \implies \mathcal{M}\mathcal{N} \Vdash tu \downarrow B)$$

5. *$T = \forall X\, A$. Let $\mathcal{M} : [\![\forall X A]\!]$. Then*

$$\mathcal{M} \Vdash t \downarrow \forall X A \iff (\forall \mathcal{C}.\, \mathcal{N} \Vdash \mathcal{C} \downarrow \mathsf{Cand}_{U \times V} \implies \mathcal{M} U \mathcal{N} \Vdash tV \downarrow A[\dot{\mathcal{C}}/X])$$

4.3. Non-Standard Updates are Forcing Candidates

We are now going to prove that $^*\mathtt{U}$ (see definition 16) is a forcing candidate. Therefore, we must define an operation of "application" $\nabla_{^*\mathtt{U}}$, that applies a term $\mathcal{L} : \mathtt{N} \to [\![\mathtt{U}]\!]$ to a non-standard natural number $(\mathcal{M}, g)$. The goal is to define a modulus of prefix point for a term $u(t)$, with $t : \mathtt{N}$, out of $\mathcal{L}$ and $(\mathcal{M}, g)$, assuming that for every numeral n, $\mathcal{L}n$ is a modulus of prefix point for $u(n)$, $\mathcal{M}$ is a modulus of forcing for t and for all states s, $g(s) = t[s]$. We have to receive as input a $k \in \mathbb{K}$ and a state s; we have to compute a state $s' \geq s$ such that $k(s')$ is a prefix point for $u(t)$. Since computing this prefix point requires in a somewhat circular manner to know the value of g in this prefix point (value which will be given to $\mathcal{L}$), the idea is first to force g to be constant for the rest of computation: this is obtained by defining a continuation k', which represents the sequence of operations: evaluate g in a state r obtaining n, apply $\mathcal{L}$ to n and use the result to extend the state r to some r', such that $k(r')$ is a prefix point of $u(n)$. This continuation k', composed with k, is given to $\mathcal{M}$ which will force g to be constant in such continuation. Finally, g is transformed into a natural number in the state computed by $\mathcal{M}$ and given to $\mathcal{L}$. Details follow.

Proposition 5 ($^*\mathtt{U}$ is a Forcing Candidate). *Let $\mathcal{L} : \mathtt{N} \to (\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})$, $\mathcal{M} : (\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})$ and $g : \mathtt{S} \to \mathtt{N}$ be term variables. Define*

$$k' := \lambda s^{\mathtt{S}}.\, (\mathcal{L}_{g(s)})ks$$
$$k'' := \mathcal{M}_{k \circ k'}$$

and

$$\mathcal{L} \nabla_{\mathtt{U}} (\mathcal{M}, g) := \lambda k^{\mathtt{S} \to \mathtt{S}}\, k' \circ k''$$

Then

$$\lambda \mathcal{M}\, \lambda g\, \lambda \mathcal{L}.\, \mathcal{L} \nabla_{\mathtt{U}} (\mathcal{M}, g) \Vdash {}^*\mathtt{U} \in \mathsf{Cand}$$

Proof. According to definition 17, suppose that:

1. $\mathcal{M} \Vdash t \in \mathtt{N}$;

2. for all states s, $g(s) = t[s]$;
3. $\mathcal{L} : \mathtt{N} \to (\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})$ is a term of $\mathcal{F}$ such that for every numeral m, $\langle \mathcal{L}m, u(m)t_1 \ldots t_n \rangle \in {}^*\mathtt{U}$

We have to prove that

$$\mathcal{L}\nabla_{\mathtt{U}}(\mathcal{M}, g) = \lambda k^{\mathtt{S}\to\mathtt{S}}\, k' \circ k'' \in {}^*\mathtt{U}$$

So, let $k \in \mathbb{K}$ and $s : \mathtt{S}$. $k' \circ k'' \in \mathbb{K}$ follows from hypotheses on $\mathcal{M}$ and $\mathcal{L}$ and definition of k', k''. We thus have to show that

$$u(t)t_1 \ldots t_n[s] \preceq s$$

Since $k'' = \mathcal{M}_{k\circ k'}$ and $\mathcal{M} \Vdash t \in \mathtt{N}$, we obtain that

$$t \downarrow [k''(s), k \circ k'(k''(s))]$$

Let $m = g(k''(s))$. Since $k' = \lambda s^{\mathtt{S}}. (\mathcal{L}_{g(s)})ks$, we have

$$k'(k''(s)) = (\mathcal{L}_m k)(k''(s))$$

By hypothesis (3.) on $\mathcal{L}$, $\mathcal{L}_m$ is a modulus of prefix point for $u(m)t_1 \ldots t_n$. Letting

$$s' = k((\mathcal{L}_m k)(k''(s))) = k(k'(k''(s)))$$

we get that

$$u(m)t_1 \ldots t_n[s'] \preceq s'$$

By $k, k' \in \mathbb{K}$, it holds that $k''(s) \leq k'(k''(s)) \leq k(k'(k''(s)))$. Therefore

$$t[s'] = t[k''(s)] = g(k''(s)) = m$$

It follows that

$$u(t)t_1 \ldots t_n[s'] = u(m)t_1 \ldots t_n[s'] \preceq s'$$

which is the thesis.

4.4. Non-Standard Natural Numbers are Forcing Candidates

We now prove that ${}^*\mathtt{N}$ (see definition 14) is a forcing candidate. The construction involved is analogous to the one used in proving that ${}^*\mathtt{U}$ is a forcing candidate.

Proposition 6 (Non-Standard Natural Numbers are Forcing Candidates). *Let* $\mathcal{L} : \mathtt{N} \to ((\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S}) \times \mathtt{S} \to \mathtt{N})$, $\mathcal{M} : (\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})$ *and* $g : \mathtt{S} \to \mathtt{N}$ *be term variables. Define*

$$\begin{aligned} k' &:= \lambda s^{\mathtt{S}}. (\pi_0 \mathcal{L}_{g(s)})ks \\ k'' &:= \mathcal{M}_{k\circ k'} \end{aligned}$$

and

$$\mathcal{L}\nabla_{^*\mathtt{N}}(\mathcal{M}, g) := \langle \lambda k^{\mathtt{S}\to\mathtt{S}}\, k' \circ k'', \lambda s^{\mathtt{S}}. (\pi_1 \mathcal{L}_{g(s)})s \rangle$$

Then

$$\lambda\mathcal{M}\, \lambda g\, \lambda\mathcal{L}.\ \mathcal{L}\nabla_{\mathtt{N}}(\mathcal{M}, g) \Vdash {}^*\mathtt{N} \in \mathsf{Cand}$$

Proof. According to definition 17, suppose that:

1. $\mathcal{M} \Vdash t \in \mathtt{N}$;
2. for all states s, $g(s) = t[s]$;
3. $\mathcal{L} : \mathtt{N} \to ((\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S}) \times \mathtt{S} \to \mathtt{N})$ is a term of $\mathcal{F}$ such that for every numeral m, $\langle \mathcal{L}m, u(m)t_1 \ldots t_n \rangle \in {}^*\mathsf{Nat}$.

We have to prove that
$$\mathcal{L}\triangledown_{\mathtt{N}}(\mathcal{M},g) = \langle \lambda k^{\mathtt{S}\to\mathtt{S}}\, k' \circ k'', \lambda s^{\mathtt{S}} \pi_1 \mathcal{L}_{g(s)} \rangle \in {}^*\mathtt{N}$$
and therefore that: i) $\lambda k^{\mathtt{S}\to\mathtt{S}}\, k' \circ k''$ is a modulus of forcing for $u(t)t_1 \dots t_n$; ii) that when evaluated in any state s, the former is equal to $(\pi_1 \mathcal{L}_{g(s)})s$. ii) is immediate: since $g(s) = t[s]$, and by hypothesis (3.) for every numeral m, $(\pi_1 \mathcal{L}_m)s = u(m)t_1 \dots t_n[s]$, we get
$$(\pi_1 \mathcal{L}_{g(s)})s = u(g(s))t_1 \dots t_n[s] = u(t)t_1 \dots t_n[s]$$
We now prove i). Let $k \in \mathbb{K}$ and $s : \mathtt{S}$. The fact that $k' \circ k'' \in \mathbb{K}$ follows from hypotheses on $\mathcal{M}$ and $\mathcal{L}$ and by inspection of the definition of k', k''. It remains to show that
$$u(t)t_1 \dots t_n \downarrow [k'(k''(s)), k \circ k'(k''(s))]$$
Since $k'' = \mathcal{M}_{k \circ k'}$ and $\mathcal{M} \Vdash t \in \mathtt{N}$, we obtain that
$$t \downarrow [k''(s), k \circ k'(k''(s))]$$
Let $m = g(k''(s))$. Since $k' = \lambda s^{\mathtt{S}}. (\pi_0 \mathcal{L}_{g(s)})ks$, we have
$$k'(k''(s)) = (\pi_0 \mathcal{L}_m)k(k''(s))$$
Since $\pi_0 \mathcal{L}_m \Vdash u(m)t_1 \dots t_n \in \mathtt{N}$, we get that
$$u(m)t_1 \dots t_n \downarrow [k'(k''(s)), k(k'(k''(s)))]$$
By $k, k' \in \mathbb{K}$, it holds that $k''(s) \leq k'(k''(s)) \leq k(k'(k''(s)))$. Therefore for every s' in the interval $[k'(k''(s)), k \circ k'(k''(s))]$
$$t[s'] = t[k''(s)] = g(k''(s)) = m$$
It follows that
$$u(t)t_1 \dots t_n \downarrow [k'(k''(s)), k \circ k'(k''(s))]$$
which is the thesis.

4.5. The Interpretations of Types are Forcing Candidates

We are going to prove that the interpretation of types in our realizability model is a forcing candidate. Before, we need some notation.

Notation. In the following, we shall assume that the type variables of $\mathcal{F}_{\text{Class}}$ are $X_0, X_1, \dots, X_n \dots$ (but when the index is not important, we shall denote them with generical metavariables $X, Y, \dots$). To each type variable X_i we associate a term variable $x_i^{\mathsf{Cand}_{X_i}}$. Moreover, we assume to have for each forcing candidate $\mathcal{C}$ of type $U \times V$ a term variable $x^{\dot{\mathcal{C}}}$ of type U.

We start by defining the operation $\triangledown$ of "application" of a term $\mathcal{L} : \mathsf{Nat} \to [\![T]\!]$ to a non-standard number.

Definition 20 (Collection of Moduli Turned into a Single Modulus). *Let T be an extended type. Let $\mathcal{L} : \mathsf{Nat} \to [\![T]\!]$, $\mathcal{M} : (\mathtt{S} \to \mathtt{S}) \to (\mathtt{S} \to \mathtt{S})$ and $g : \mathtt{S} \to \mathsf{Nat}$ be term variables. We define by induction on T a term $\mathcal{L}\triangledown_T(\mathcal{M}, g)$ of type $[\![T]\!]$.*

1. *$T = \mathtt{N}$ or $T = \mathtt{U}$. Then $\mathcal{L}\triangledown_T(\mathcal{M}, g)$ has already been defined in propositions 5, 6.*

2. *T is a variable X_i. Then*
$$\mathcal{L}\triangledown_T(\mathcal{M}, g) := x_i^{\mathsf{Cand}_{X_i}} \mathcal{M} g \mathcal{L}$$

3. $T = \dot{\mathcal{C}}$, *with* $\mathcal{C}$ *forcing candidate of type* $U \times V$. *Then*

$$\mathcal{L}\triangledown_T(\mathcal{M}, g) := x^{\dot{\mathcal{C}}}\mathcal{M}g\mathcal{L}$$

4. $T = A \to B$. *Then*

$$\mathcal{L}\triangledown_T(\mathcal{M}, g) := \lambda\mathcal{N}^{[\![A]\!]}(\lambda m^{\mathtt{N}}\mathcal{L}_m\mathcal{N})\triangledown_B(\mathcal{M}, g)$$

5. $T = \forall X_i\, A$. *Then*

$$\mathcal{L}\triangledown_T(\mathcal{M}, g) := \Lambda X_i \lambda x_i^{\mathsf{Cand}_{X_i}}(\lambda m^{\mathtt{N}}\mathcal{L}_m X_i x_i)\triangledown_A(\mathcal{M}, g)$$

Remark 4. The normal form of the term $\mathcal{L}\triangledown_T(\mathcal{M}, g)$ is simply of the shape:

$$\{\Lambda X_1\}\lambda y_1^{A_1} \dots \{\Lambda X_n\}\lambda y^{A_n}.\, x^B\mathcal{M}g(\mathcal{L}\{X_1\}y_1 \dots \{X_n\}y_n)$$

where the notation $\{\Lambda X_i\}$ means that ΛX_i may or may not appear.

We now prove that the operation $\triangledown_T$ realizes that the interpretation of types is a forcing candidate.

Lemma 9 (The Interpretation of Types are Forcing Candidates). *Assume* T *is an extended type. The following hold:*

1. *If* X_i *occurs free in* T *but not in* U_i, $\mathcal{M}, g$ *are closed terms and* $\mathcal{N}_i \Vdash \mathcal{C}_i \in \mathsf{Cand}_{U_i \times V_i}$, *then*

$$\mathcal{L}\triangledown_T(\mathcal{M}, g)[U_i/X_i][\mathcal{N}_i/x_i^{\mathsf{Cand}_{U_i}}] = \mathcal{L}[U_i/X_i]\triangledown_{T[\dot{\mathcal{C}}_i/X_i]}(\mathcal{M}, g)][\mathcal{N}_i/x^{\dot{\mathcal{C}}_i}]$$

2. *Suppose that for* $i = 1, \dots, n$, $\mathcal{N}_i \Vdash \mathcal{C}_i \in \mathsf{Cand}_{U_i \times V_i}$ *and that each candidate constant of* T *is equal to some* $\dot{\mathcal{C}}_i$. *Suppose that for every numeral* m, $\mathcal{L}_m \Vdash u(m)t_1 \dots t_k \downarrow T$, $\mathcal{M}$ *is a modulus of forcing for* t *and* $g = \lambda s^{\mathtt{S}}\, t[s]$. *Then*

$$\mathcal{L}\triangledown_T(\mathcal{M}, g)\,[\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \dots \mathcal{N}_n/x^{\dot{\mathcal{C}}_n}] \Vdash u(t)t_1 \dots t_k \downarrow T$$

3. *Suppose that for* $i = 1, \dots, n$, $\mathcal{N}_i \Vdash \mathcal{C}_i \in \mathsf{Cand}_{U_i \times V_i}$ *and that each candidate constant of* T *is equal to some* $\dot{\mathcal{C}}_i$. *Define*

$$\triangledown^T := \lambda\mathcal{M}^{(\mathtt{S}\to\mathtt{S})\to(\mathtt{S}\to\mathtt{S})}\,\lambda g^{\mathtt{N}\to\mathtt{S}}\,\lambda\mathcal{L}^{\mathtt{N}\to[\![T]\!]}\mathcal{L}\triangledown_T(\mathcal{M}, g)$$

Then

$$\triangledown^T[\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \dots \mathcal{N}_n/x^{\dot{\mathcal{C}}_n}] \Vdash \{\langle\mathcal{N}, t\rangle \mid \mathcal{N} \Vdash t \downarrow T\} \in \mathsf{Cand}_{[\![T]\!]\times|T|}$$

4. *Let*

$$\mathcal{C} := \{\langle\mathcal{N}, t\rangle \mid \mathcal{N} \Vdash t \downarrow B\}$$

Then

$$\mathcal{N} \Vdash t \downarrow A[\dot{\mathcal{C}}/X] \iff \mathcal{N} \Vdash t \downarrow A[B/X]$$

Proof. First of all we define the following abbreviation:

$$[\vec{\mathcal{N}}/\vec{x}^{\vec{\mathcal{C}}}] := [\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \dots \mathcal{N}_n/x^{\dot{\mathcal{C}}_n}]$$

1. By induction on T (which is not a constant since X_i occurs in T).

(a) T is a variable. Then $T = X_i$, since X_i occurs free in T. Moreover,

$$\begin{aligned}
\mathcal{L}\nabla_T(\mathcal{M},g)[U_i/X_i][\mathcal{N}_i/x_i^{\mathsf{Cand}_{U_i}}] &= x_i^{\mathsf{Cand}_{X_i}}\mathcal{M}g\mathcal{L}[U_i/X_i][\mathcal{N}_i/x_i^{\mathsf{Cand}_{U_i}}] \\
&= \mathcal{N}_i\mathcal{M}g(\mathcal{L}[U_i/X_i][\mathcal{N}_i/x_i^{\mathsf{Cand}_{U_i}}]) \\
&= x^{\dot{\mathcal{C}}_i}\mathcal{M}g(\mathcal{L}[U_i/X_i])[\mathcal{N}_i/x^{\dot{\mathcal{C}}_i}] \\
&= \mathcal{L}[U_i/X_i]\nabla_{\dot{\mathcal{C}}_i}(\mathcal{M},g)[\mathcal{N}_i/x^{\dot{\mathcal{C}}_i}]
\end{aligned}$$

which is the thesis.

(b) $T = A \to B$. Then, since $[\![A]\!][U_i/X_i] = [\![A[\dot{\mathcal{C}}_i/X_i]]\!]$, we have

$$\begin{aligned}
\mathcal{L}\nabla_T(\mathcal{M},g)[U_i/X_i][\mathcal{N}_i/x_i^{\mathsf{Cand}_{U_i}}] &= \lambda\mathcal{N}^{[\![A]\!]}(\lambda m^{\mathsf{Nat}}\mathcal{L}_m\mathcal{N})\nabla_B(\mathcal{M},g)[U_i/X_i][\mathcal{N}_i/x_i^{\mathsf{Cand}_{U_i}}] \\
&\overset{\text{ind.}}{=} \lambda\mathcal{N}^{[\![A[\dot{\mathcal{C}}_i/X_i]]\!]}((\lambda m^{\mathsf{Nat}}\mathcal{L}_m\mathcal{N})[U_i/X_i]\nabla_{B[\dot{\mathcal{C}}_i/X_i]}(\mathcal{M},g)[\mathcal{N}_i/x^{\dot{\mathcal{C}}_i}]) \\
&= \lambda\mathcal{N}^{[\![A[\dot{\mathcal{C}}_i/X_i]]\!]}(\lambda m^{\mathsf{Nat}}\mathcal{L}_m[U_i/X_i]\mathcal{N})\nabla_{B[\dot{\mathcal{C}}_i/X_i]}(\mathcal{M},g)[\mathcal{N}_i/x^{\dot{\mathcal{C}}_i}] \\
&= \mathcal{L}[U_i/X_i]\nabla_{T[\dot{\mathcal{C}}_i/X_i]}(\mathcal{M},g)[\mathcal{N}_i/x^{\dot{\mathcal{C}}_i}]
\end{aligned}$$

(c) $T = \forall X_j A$. Then $X_i \neq X_j$, since X_i occurs free in T. Moreover,

$$\begin{aligned}
\mathcal{L}\nabla_T(\mathcal{M},g)[U_i/X_i][\mathcal{N}_i/x_i^{\mathsf{Cand}_{U_i}}] &= \Lambda X_j\lambda x_j^{\mathsf{Cand}_{X_j}}(\lambda m^{\mathsf{Nat}}\mathcal{L}_m X_j x_j)\nabla_A(\mathcal{M},g)[U_i/X_i][\mathcal{N}_i/x_i^{\mathsf{Cand}_{U_i}}] \\
&= \Lambda X_j\lambda x_j^{\mathsf{Cand}_{X_j}}((\lambda m^{\mathsf{Nat}}\mathcal{L}_m X_j x_j)\nabla_A(\mathcal{M},g)[U_i/X_i][\mathcal{N}_i/x_i^{\mathsf{Cand}_{U_i}}]) \\
&\overset{\text{ind.}}{=} \Lambda X_j\lambda x_j^{\mathsf{Cand}_{X_j}}(\lambda m^{\mathsf{Nat}}\mathcal{L}_m X_j x_j)[U_i/X_i]\nabla_{A[\dot{\mathcal{C}}_i/X_i]}(\mathcal{M},g)[\mathcal{N}_i/x^{\dot{\mathcal{C}}_i}] \\
&= \Lambda X_j\lambda x_j^{\mathsf{Cand}_{X_j}}(\lambda m^{\mathsf{Nat}}\mathcal{L}_m[U_i/X_i]X_j x_j)\nabla_{A[\dot{\mathcal{C}}_i/X_i]}(\mathcal{M},g)[\mathcal{N}_i/x^{\dot{\mathcal{C}}_i}] \\
&= \mathcal{L}[U_i/X_i]\nabla_{\forall X_j A[\dot{\mathcal{C}}_i/X_i]}(\mathcal{M},g)[\mathcal{N}_i/x^{\dot{\mathcal{C}}_i}]
\end{aligned}$$

2. By induction on T (which by assumption on $\mathcal{L}$ is closed and hence cannot be a variable).

(a) $T = \mathtt{N}$ or $T = \mathtt{U}$. The thesis follows by propositions 5 and 6.

(b) $T = \dot{\mathcal{C}}_i$. Then

$$\begin{aligned}
\mathcal{L}\nabla(\mathcal{M},g)[\vec{\mathcal{N}}/\vec{x}^{\vec{\mathcal{C}}}] &= x_i^{\dot{\mathcal{C}}_i}\mathcal{M}g\mathcal{L}[\vec{\mathcal{N}}/\vec{x}^{\vec{\mathcal{C}}}] \\
&= \mathcal{N}_i\mathcal{M}g\mathcal{L}
\end{aligned}$$

Since $\mathcal{N}_i \Vdash \mathcal{C}_i \in \mathsf{Cand}_{U_i \times V_i}$, we obtain by definition 17 that

$$\langle \mathcal{N}_i\mathcal{M}g\mathcal{L}, u(t)t_1\dots t_k\rangle \in \mathcal{C}$$

and therefore by definition 19

$$\mathcal{N}_i\mathcal{M}g\mathcal{L} \Vdash u(t)t_1\dots t_k \downarrow \dot{\mathcal{C}}$$

which is the thesis.

(c) $T = C \to B$. Suppose $\mathcal{I} \Vdash t_{k+1} \downarrow C$. We have to show that

$$\begin{aligned}
&\mathcal{L}\triangledown_T(\mathcal{M}, g)[\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \ldots \mathcal{N}_n/x^{\dot{\mathcal{C}}_n}]\mathcal{I} \\
&= (\lambda \mathcal{H}^{[\![C]\!]}(\lambda m^{\mathsf{Nat}}\mathcal{L}_m\mathcal{H})\triangledown_B(\mathcal{M}, g))\mathcal{I}[\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \ldots \mathcal{N}_n/x^{\dot{\mathcal{C}}_n}] \\
&= (\lambda m^{\mathsf{Nat}}\mathcal{L}_m\mathcal{I})\triangledown_B(\mathcal{M}, g)[\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \ldots \mathcal{N}_n/x^{\dot{\mathcal{C}}_n}] \\
&\Vdash u(t)t_1 \ldots t_k t_{k+1} \downarrow B
\end{aligned}$$

By hypothesis, for every numeral m, $\mathcal{L}_m \Vdash u(m)t_1 \ldots t_k \downarrow C \to B$. Therefore, for every numeral m

$$\mathcal{L}_m\mathcal{I} \Vdash u(m)t_1 \ldots t_k t_{k+1} \downarrow B$$

By induction hypothesis

$$(\lambda m^{\mathsf{Nat}}\mathcal{L}_m\mathcal{N}_n)\triangledown_B(\mathcal{M}, g)[\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \ldots \mathcal{N}_n/x^{\dot{\mathcal{C}}_n}] \Vdash u(t)t_1 \ldots t_k \downarrow B$$

which is the thesis.

(d) $T = \forall X_{n+1}A$. Suppose that $\mathcal{N}_{n+1} \Vdash \mathcal{C}_{n+1} \in \mathsf{Cand}_{U_{n+1}\times V_{n+1}}$. We have to show that

$$\begin{aligned}
&\mathcal{L}\triangledown_T(\mathcal{M}, g)[\vec{\mathcal{N}}/\vec{x}^{\vec{\mathcal{C}}}]U_{n+1}\mathcal{N}_{n+1} \\
&= (\Lambda X_{n+1}\lambda x_{n+1}^{\mathsf{Cand}_{X_{n+1}}}(\lambda m^{\mathsf{Nat}}\mathcal{L}_m X_{n+1}x_{n+1})\triangledown_A(\mathcal{M}, g)[\vec{\mathcal{N}}/\vec{x}^{\vec{\mathcal{C}}}])U_{n+1}\mathcal{N}_{n+1} \\
&= (\lambda m^{\mathsf{Nat}}\mathcal{L}_m X_{n+1}x_{n+1})\triangledown_A(\mathcal{M}, g)[U_{n+1}/X_{n+1}][\mathcal{N}_{n+1}/x_{n+1}^{\mathsf{Cand}_{X_{n+1}}}][\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \ldots \mathcal{N}_n/x^{\dot{\mathcal{C}}_n}] \\
&\overset{(1.)}{=} (\lambda m^{\mathsf{Nat}}\mathcal{L}_m U_{n+1}\mathcal{N}_{n+1})\triangledown_{A[\dot{\mathcal{C}}/X_{n+1}]}(\mathcal{M}, g)[\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \ldots \mathcal{N}_{n+1}/x^{\dot{\mathcal{C}}_{n+1}}] \\
&= (\lambda m^{\mathsf{Nat}}\mathcal{L}_m U_{n+1}\mathcal{N}_{n+1})\triangledown_{A[\dot{\mathcal{C}}/X_{n+1}]}(\mathcal{M}, g)[\mathcal{N}_1/x^{\dot{\mathcal{C}}_1} \ldots \mathcal{N}_{n+1}/x^{\dot{\mathcal{C}}_{n+1}}] \\
&\Vdash u(t)t_1 \ldots t_k V_{n+1} \downarrow A[\dot{\mathcal{C}}_{n+1}/X_{n+1}]
\end{aligned}$$

By hypothesis on $\mathcal{L}$, for every numeral m,

$$\mathcal{L}_m U_{n+1}\mathcal{N}_{n+1} \Vdash u(m)t_1 \ldots t_k V_{n+1} \downarrow A[\dot{\mathcal{C}}_{n+1}/X_{n+1}]$$

Therefore, the thesis follows by induction hypothesis.

3. It is a corollary of (2.).

4. By induction on A.

(a) A is a type variable. If $A \neq X$, the thesis is obvious. Suppose $A = X$. Then $A[\dot{\mathcal{C}}/X] = \dot{\mathcal{C}}$ and $A[B/X] = B$. Therefore

$$\mathcal{N} \Vdash t \downarrow A[\dot{\mathcal{C}}/X] \iff \mathcal{N} \Vdash t \downarrow \dot{\mathcal{C}} \iff \langle \mathcal{N}, t\rangle \in \mathcal{C} \iff \mathcal{N} \Vdash t \downarrow B = A[B/X]$$

(b) The other cases are straightforward.

4.5.1. Forcing of Constants

From now on, we devote to the definition of our state-extending-continuation-passing-style translation $[\![\_]\!]$ of terms of System $\mathcal{F}_{\text{Class}}$ into System $\mathcal{F}$. For simplicity, the translation is defined for *proof-like terms* of $\mathcal{F}_{\text{Class}}$: a term t is said to be proof-like if: i) every occurrence in t of the constants Φ_i or mkupd is of the form Φ_i or mkupd_i, where i is some numeral; ii) no update constant different from $\varnothing$ occurs in t. Indeed, that is the syntactic form of every interactive realizer extracted from some proof in $\mathsf{HAS}+\mathsf{EM}_1+\mathsf{SK}_1$.

Notation. For notational convenience and to define in a more readable way terms of type $A\times B\to C$, for any variables $x_0 : A$ and $x_1 : B$ we define

$$\lambda\langle x_0, x_1\rangle^{A\times B}u := \lambda x^{A\times B}u[\pi_0 x/x_0\ \pi_1 x/x_1]$$

where x is a fresh variable not appearing in u. We observe that for any terms t_0, t_1

$$(\lambda\langle x_0, x_1\rangle^{A\times B}u)\langle t_0, t_1\rangle = u[t_0/x_0\ t_1/x_1]$$

For each proof-like constant $c : T$ of $\mathcal{F}$, we now define a term $[\![c]\!]$, which is intended to satisfy the relation $[\![c]\!] \Vdash c \downarrow T$. $[\![c]\!]$ can be seen as the non-standard version of the operation denoted by c. More precisely:

- $[\![\Phi_i]\!]$, as explained before, is a sort of control operator and it represents the computational content of the excluded middle. Given a numeral n, a continuation $k \in \mathbb{K}$ and a current state s, it has the task of choosing a state where to approximate $\Phi_i(n)$. It does that by looking at the future of the computation $s' = k(s)$; if this future state s' is defined on argument (i, n), while the current state s is not, it returns the state s' and Φ_i is approximated with $(s')_i(n)$; otherwise if returns the current state s and Φ_i is approximated with $s_i(n)$. But we observe that the argument taken by Φ_i is in general a term t of $\mathcal{F}_{\text{Class}}$. Therefore, $[\![\Phi_i]\!]$ first of all forces t to be constant for the rest of the computation and then translate it into a numeral; then it applies the previous considerations.
- $[\![\mathsf{mkupd}_i]\!]$ represents the computational content of the Skolem axiom relative to the predicate P_i. Given a continuation $k \in \mathbb{K}$, a current state s, and two terms $t_1, t_2 : \mathtt{N}$, it has the task of computing a state $s' \geq s$ such that $\mathsf{mkupd}_i t_1 t_2[k(s')] \preceq k(s')$. Again, when t_1, t_2 are numerals n_1, n_2, the task is easy: it suffices to take $s' = s \oplus_i \{(i, n_1, n_2)\}$ (remember definition 6 of $\oplus_i$); indeed

  $$\mathsf{mkupd}_i n_1 n_2[k(s')] = \{(i, n_1, n_2)\} \preceq s \oplus_i \{(i, n_1, n_2)\} = s' \leq k(s')$$

  But t_1, t_2 in general are terms of $\mathcal{F}_{\text{Class}}$: it might happen that $t_1[s] = n_1, t_2[s] = n_2$, $t_1[s'] = n_1' \neq n_1, t_2[s'] = n_2'$ and

  $$\mathsf{mkupd}_i t_1 t_2[k(s')] = \{(i, n_1', n_2')\} \not\preceq s \oplus_i \{(i, n_1, n_2)\} = s' \leq k(s')$$

  As before, the solution is to force t_1, t_2 to behave like a pair n_1, n_2 for the rest of the computation, which is defined as k composed with $\lambda s^{\mathsf{S}}.\, s \oplus_i \mathsf{mkupd}_i\, t_1[s]t_2[s]$. Then $[\![\mathsf{mkupd}_i]\!]$ applies the considerations of the easy case.
- $[\![\mathsf{R}]\!]$ is recursion over non-standard natural numbers. The problem is that one does not know how to iterate some functional a "number" of times given by a term $t : \mathtt{N}$ of System $\mathcal{F}_{\text{Class}}$. Thus $[\![\mathsf{R}_A]\!]$ uses the realizer of a forcing candidate of type A to force t to behave like a fixed number n for the rest of the computation and iterates the functional n times.

Definition 21 (Forcing of Constants). *We define for every proof-like constant $c : T$ of $\mathcal{F}_{\text{Class}}$ a closed term $[\![c]\!] : [\![T]\!]$, accordingly to the form of c.*

1. $c = 0$. *Let $u : \mathtt{N}$ be any term of System $\mathcal{F}$. We define*

   $$u^* := \langle \lambda k^{\mathsf{S}\to\mathsf{S}}\lambda s^{\mathsf{S}} s, \lambda s^{\mathsf{S}} u\rangle$$

   Then

   $$[\![0]\!] := 0^*$$

2. $c = \varnothing : \mathtt{U}$. *Then*

$$[\![\varnothing]\!] := \lambda k^{\mathsf{S}\to\mathsf{S}}\lambda s^{\mathsf{S}}.\, s$$

3. $c = \mathsf{S} : \mathtt{N} \to \mathtt{N}$. *Then*

$$[\![\mathsf{S}]\!] := \lambda\langle\mathcal{M}, g\rangle^{[\![\mathtt{N}]\!]}\langle\mathcal{M}, \lambda s^{\mathsf{S}}.\, \mathsf{S}(g(s))\rangle$$

4. $c = \Phi_i : \mathtt{N} \to \mathtt{N}$. *Let*

$$\mathcal{L} := \lambda n^{\mathtt{N}}\langle\lambda k^{\mathsf{S}\to\mathsf{S}}\lambda s^{\mathsf{S}}\,\mathsf{if}\ s_i(n) = (ks)_i(n)\ \mathsf{then}\ s\ \mathsf{else}\ ks, \lambda s^{\mathsf{S}}\, s_i(n)\rangle$$

Then

$$[\![\Phi]\!] := \lambda\langle\mathcal{M}, g\rangle^{[\![\mathtt{N}]\!]}\mathcal{L}\triangledown_{\mathtt{N}}(\mathcal{M}, g)$$

5. $c = \Cup : \mathtt{U} \to \mathtt{U} \to \mathtt{U}$. *Let*

$$k' := \mathcal{N}_k$$
$$k'' := \mathcal{M}_{k\circ k'}$$

Then

$$[\![\Cup]\!] := \lambda\mathcal{M}^{[\![\mathtt{U}]\!]}\lambda\mathcal{N}^{[\![\mathtt{U}]\!]}.\, \lambda k^{\mathsf{S}\to\mathsf{S}}k' \circ k''$$

6. $c = \mathsf{mkupd}_i : \mathtt{N}^2 \to \mathtt{U}$. *Let*

$$k' := \lambda s^{\mathsf{S}}.\, s \oplus_i \mathsf{mkupd}_i\, g_1(s)g_2(s)$$
$$k'' := (\mathcal{M}_1 \sqcup \mathcal{M}_2)_{k\circ k'}$$

$$[\![\mathsf{mkupd}_i]\!] := \lambda\langle\mathcal{M}_1, g_1\rangle^{[\![\mathtt{N}]\!]}\lambda\langle\mathcal{M}_2, g_2\rangle^{[\![\mathtt{N}]\!]}\,\lambda k^{\mathsf{S}\to\mathsf{S}}.\, k' \circ k''$$

7. $c = \mathsf{R} : \forall A.\, A \to (\mathtt{N} \to A \to A) \to \mathtt{N} \to A$. *Define*

$$\mathcal{N} := \lambda n^{\mathtt{N}}\ \mathsf{R}_U\mathcal{I}(\lambda n^{\mathtt{N}}\mathcal{L}n^*)n$$

with

$$U := [\![A]\!] \to (\mathtt{N} \to [\![A]\!] \to [\![A]\!]) \to \mathtt{N} \to [\![A]\!]$$

Then

$$[\![\mathsf{R}]\!] := \Lambda A\,\lambda\mathcal{H}^{\mathsf{Cand}_A}\lambda\mathcal{I}^{[\![A]\!]}\,\lambda\mathcal{L}^{[\![\mathtt{N}\to A\to A]\!]}\,\lambda\langle\mathcal{M}, g\rangle^{[\![\mathtt{N}]\!]}.\, \mathcal{H}\mathcal{M}g\mathcal{N}$$

We now prove that for any constant $c : T$, $[\![c]\!]$ forces c to be a computable functional of type T.

Proposition 7. *For every proof-like constant $c : T$, $[\![c]\!] \Vdash c \downarrow T$.*

Proof. We proceed by cases, accordingly to the form of c.

1. $c : \mathtt{N}$, with $c = 0$. By definition 21

$$[\![0]\!] = \langle\lambda k^{\mathsf{S}\to\mathsf{S}}\lambda s^{\mathsf{S}}\, s, \lambda s^{\mathsf{S}}\, 0\rangle$$

We have therefore to prove that $\lambda k^{\mathsf{S}\to\mathsf{S}}\lambda s^{\mathsf{S}}\, s$ is a modulus of forcing for 0, which is trivially true, since $0[s] = 0$ for every state s and that $(\lambda s^{\mathsf{S}}0)s = 0[s]$, which is again trivial. We conclude $[\![0]\!] \Vdash 0 \downarrow \mathtt{N}$.

2. $c = \varnothing$. By definition 21 of $[\![\varnothing]\!]$

$$[\![\varnothing]\!] := \lambda k^{\mathsf{S}\to\mathsf{S}}\lambda s^{\mathsf{S}}.\, s$$

Let $k \in \mathbb{K}$ and s be a state. Let

$$s' = k([\![\varnothing]\!]ks) = k(s)$$

According to definition 19, we have to prove that

$$[\![\varnothing]\!] \Vdash \varnothing \downarrow \mathtt{U}$$

and thus that $[\![\varnothing]\!]k \in \mathbb{K}$, which is trivially true, and that

$$\varnothing = \varnothing[s'] \preceq s$$

which is again trivially true, since $\mathsf{dom}(\varnothing) = \emptyset \subseteq \mathsf{dom}(s)$.

3. $c = \mathsf{S} : \mathtt{N} \to \mathtt{N}$. By definition 21

$$[\![\mathsf{S}]\!] = \lambda\langle\mathcal{M}, g\rangle^{[\![\mathtt{N}]\!]}\langle\mathcal{M}, \lambda s^{\mathsf{S}}.\, \mathsf{S}(g(s))\rangle$$

Suppose $\langle\mathcal{M}, g\rangle \Vdash t \downarrow \mathtt{N}$. Then $\mathcal{M}$ is a modulus of forcing for t and for all states s, $t[s] = g(s)$. Obviously, $\mathcal{M}$ is also a modulus of forcing for $\mathsf{S}(t)$. Moreover, for all states s, $\mathsf{S}(t)[s] = \mathsf{S}(g(s))$. Hence

$$[\![\mathsf{S}(t)]\!]\langle\mathcal{M}, g\rangle \Vdash \mathsf{S}(t) \downarrow \mathtt{N}$$

which is the thesis.

4. $c = \Phi_i$. Let $t : \mathtt{N}$ and suppose $\langle\mathcal{M}, g\rangle \Vdash t \downarrow \mathtt{N}$. By definition 19, we have to prove that

$$[\![\Phi_i]\!]\langle\mathcal{M}, g\rangle \Vdash \Phi t \downarrow \mathtt{N}$$

By definition 21 of $[\![\Phi]\!]$

$$[\![\Phi_i]\!]\langle\mathcal{M}, g\rangle = \mathcal{L}\triangledown_{\mathtt{N}}(\mathcal{M}, g)$$

with

$$\mathcal{L} := \lambda n^{\mathtt{N}}\langle\lambda k^{\mathsf{S}\to\mathsf{S}}\lambda s^{\mathsf{S}}\,\mathsf{if}\ s_i(n) = (ks)_i(n)\ \mathsf{then}\ s\ \mathsf{else}\ ks, \lambda s^{\mathsf{S}}\, s_i(n)\rangle$$

Since $\langle\mathcal{M}, g\rangle \Vdash t \downarrow \mathtt{N}$, if we prove that for every numeral n, $\mathcal{L}_n \Vdash \Phi n \downarrow \mathtt{N}$, we obtain by lemma 9 (2.) that $\mathcal{L}\triangledown_{\mathtt{N}}(\mathcal{M}, g) \Vdash \Phi t \downarrow \mathtt{N}$ and we are done. First we have to check that for all s^{S}

$$\pi_1\mathcal{L}_n[s] = s_i(n) = \Phi_i(n)[s]$$

which is true. It remains us to show that, given any numeral n,

$$\pi_0\mathcal{L}_n = \lambda k^{\mathsf{S}\to\mathsf{S}}\lambda s^{\mathsf{S}}\,\mathsf{if}\ s_i(n) = (ks)_i(n)\ \mathsf{then}\ s\ \mathsf{else}\ ks$$

is a modulus of forcing for $\Phi_i(n)$. We have to prove that given any $k \in \mathbb{K}$ and state s,

$$\Phi_i(n) \downarrow [(\pi_0\mathcal{L}_n)_k(s), k((\pi_0\mathcal{L}_n)_k(s))]$$

We have two possibilities:
i) $s_i(n) = (ks)_i(n)$. Since $s \le k(s)$, we have either $(i, n) \in \mathsf{dom}(s)$ and so

$$\forall q^{\mathsf{S}}.\ s \le q \le k(s) \implies q_i(n) = s_i(n)$$

or $(i, n) \notin \mathsf{dom}(s)$ and hence $(i, n) \notin \mathsf{dom}(k(s))$ and therefore

$$\forall q^{\mathsf{S}}.\ s \le q \le k(s) \implies q_i(n) = (ks)_i(n)$$

Therefore

$$\begin{aligned}\Phi_i(n) \downarrow &[s, k(s)]\\ =&[(\pi_0\mathcal{L}_n)_k(s), k((\pi_0(\mathcal{L}_n)_k(s))]\end{aligned}$$

by definition of $\mathcal{L}$.
ii) $s_i(n) \ne (ks)_i(n)$. Since $s \le k(s)$, we have $(i, n) \in \mathsf{dom}(k(s))$. Since $k(s) \le k(ks)$, we have $(ks)_i(n) = (k(ks))_i(n)$ and with the same reasoning as above we obtain

$$\begin{aligned}\Phi_i(n) \downarrow &[k(s), k(k(s))]\\ =&[(\pi_0\mathcal{L}_n)_k(z), k((\pi_0\mathcal{L}_n)_k(z))]\end{aligned}$$

5. $c = \mathbb{U} : \mathtt{U} \to \mathtt{U} \to \mathtt{U}$. Let

$$k' := \mathcal{N}_k$$
$$k'' := \mathcal{M}_{k \circ k'}$$

Then, by definition 21

$$[\![\mathbb{U}]\!] = \lambda \mathcal{M}^{[\![\mathtt{U}]\!]} \lambda \mathcal{N}^{[\![\mathtt{U}]\!]}. \lambda k^{\mathtt{S} \to \mathtt{S}} k' \circ k''$$

Suppose $\mathcal{M} \Vdash t_1 \downarrow \mathtt{U}$ and $\mathcal{N} \Vdash t_2 \downarrow \mathtt{U}$ and let $k \in \mathbb{K}$ and s be a state. We must show that

$$s' = k([\![\mathbb{U}]\!] \mathcal{M} \mathcal{N} k s) \implies \mathbb{U} t_1 t_2 [s'] \preceq s'$$

By definition of s' and $[\![\mathbb{U}]\!]$, we have that

$$s' = k(k' \circ k''(s))$$

By definition of $\mathcal{M} \Vdash t_1 \downarrow \mathtt{U}$ and of k'', we get $t_1[s'] \preceq s'$. By definition of $\mathcal{N} \Vdash t_2 \downarrow \mathtt{U}$ and k', we get $t_2[s'] \preceq s'$. Since, the update $\mathbb{U} t_1 t_2 [s']$ is contained in the union of $t_1[s']$ and $t_2[s']$, we have that $\mathbb{U} t_1 t_2 [s'] \preceq s'$, which is the thesis.

6. $c = \mathsf{mkupd}_i : \mathtt{N}^2 \to \mathtt{U}$. Let

$$k' := \lambda s^{\mathtt{S}}.\, s \oplus_i \mathsf{mkupd}_i\, g_1(s) g_2(s)$$
$$k'' := (\mathcal{M}_1 \sqcup \mathcal{M}_2)_{k \circ k'}$$

By definition 21

$$[\![\mathsf{mkupd}_i]\!] = \lambda \langle \mathcal{M}_1, g_1 \rangle^{[\![\mathtt{N}]\!]} \lambda \langle \mathcal{M}_2, g_2 \rangle^{[\![\mathtt{N}]\!]}\, \lambda k^{\mathtt{S} \to \mathtt{S}}.\, k' \circ k''$$

Suppose $\langle \mathcal{M}_1, g_1 \rangle \Vdash t_1 \downarrow \mathtt{N}$ and $\langle \mathcal{M}_2, g_2 \rangle \Vdash t_2 \downarrow \mathtt{N}$ and let $k \in \mathbb{K}$ and s be a state. We must show that

$$s' = k([\![\mathsf{mkupd}_i]\!] \langle \mathcal{M}_1, g_1 \rangle \langle \mathcal{M}_2, g_2 \rangle k s) \implies \mathsf{mkupd}_i t_1 t_2 [s'] \preceq s'$$

By definition of s' and $[\![\mathsf{mkupd}_i]\!]$, we have that

$$s' = k(k' \circ k''(s))$$

By definition of $\langle \mathcal{M}_1, g_1 \rangle \Vdash t_1 \downarrow \mathtt{N}$ and $\langle \mathcal{M}_2, g_2 \rangle \Vdash t_2 \downarrow \mathtt{N}$ and by proposition 4, we deduce that $\mathcal{M}_1 \sqcup \mathcal{M}_2$ is a modulus of forcing for both t_1 and t_2. Therefore, by definition of k''

$$t_1, t_2 \downarrow [k''(s), k \circ k'(k''(s))] = [k''(s), s']$$

We may suppose that $\mathsf{P}_i t_1 t_2 [s'] = \mathtt{True}$ (otherwise, $\mathsf{mkupd}_i t_1 t_2 [s'] = \varnothing$ and the thesis is obvious). Thus,

$$\mathsf{P}_i t_1 t_2 [k''(s)] = \mathsf{P}_i t_1 t_2 [s'] = \mathtt{True}$$

Therefore,

$$\begin{aligned} \mathsf{mkupd}_i t_1 t_2 [s'] &= \mathsf{mkupd}_i t_1 t_2 [k''(s)] \\ &= \{(i, t_1[k''(s)], t_2[k''(s)])\} \\ &\preceq k''(s) \oplus_i \{(i, t_1[k''(s)], t_2[k''(s)])\} \\ &= k'(k''(s)) \end{aligned}$$

Since $s' \geq k'(k''(s))$, we obtain $\mathsf{mkupd}_i t_1 t_2 [s'] \preceq s'$, which is the thesis.

7. $c = \mathsf{R} : \forall A.\, A \to (\mathtt{N} \to A \to A) \to \mathtt{N} \to A$. Let

$$\mathcal{N} := \lambda n^{\mathtt{N}}\, \mathsf{R}_U \mathcal{I} (\lambda n^{\mathtt{N}} \mathcal{L} n^*) n$$

with

$$U := [\![A]\!] \to (\mathtt{N} \to [\![A]\!] \to [\![A]\!]) \to \mathtt{N} \to [\![A]\!]$$

By definition 21

$$[\![\mathsf{R}]\!] = \Lambda A\, \lambda \mathcal{H}^{\mathsf{Cand}_A}\, \lambda \mathcal{I}^{[\![A]\!]}\, \lambda \mathcal{L}^{[\![\mathtt{N} \to A \to A]\!]}\, \lambda \langle \mathcal{M}, g\rangle^{[\![\mathtt{N}]\!]}.\, \mathcal{H}\mathcal{M}g\mathcal{N}$$

Suppose $\mathcal{H} \Vdash \mathcal{C} \in \mathsf{Cand}_{V\times W}$, $\mathcal{I} \Vdash u \downarrow \dot{\mathcal{C}}$, $\mathcal{L} \Vdash v \downarrow \mathtt{N} \to \dot{\mathcal{C}} \to \dot{\mathcal{C}}$ and $\langle \mathcal{M}, g\rangle \Vdash t \downarrow \mathtt{N}$. We have to prove that

$$[\![\mathsf{R}]\!]_V \mathcal{H}\mathcal{I}\mathcal{L}\langle \mathcal{M}, g\rangle = \mathcal{H}\mathcal{M}g\mathcal{N} \Vdash \mathsf{R}_W uvt \downarrow \dot{\mathcal{C}}$$

If we show that for all numerals n, $\mathcal{N}_n \Vdash \mathsf{R}_W uvn \downarrow \dot{\mathcal{C}}$, then for all numerals n, $\langle \mathcal{N}_n, \mathsf{R}_W uvn\rangle \in \mathcal{C}$ and by definition 17 of $\mathcal{H} \Vdash \mathcal{C} \in \mathsf{Cand}_{V\times W}$ we obtain that

$$\langle \mathcal{H}\mathcal{M}g\mathcal{N}, \mathsf{R}_W uvt\rangle \in \mathcal{C}$$

which is the thesis. We prove that by induction on n.
If $n = 0$, then

$$\mathcal{N}_0 = \mathsf{R}_U \mathcal{I}(\lambda n^{\mathtt{N}} \mathcal{L} n^*)0 = \mathcal{I} \Vdash u = \mathsf{R}_W uv0 \downarrow \dot{\mathcal{C}}$$

If $n = \mathsf{S}(m)$, then

$$\begin{aligned}
\mathcal{N}_{\mathsf{S}(m)} &= \mathsf{R}_U \mathcal{I}(\lambda n^{\mathtt{N}} \mathcal{L} n^*)\mathsf{S}(m) \\
&= (\lambda n^{\mathtt{N}} \mathcal{L} n^*)m(\mathsf{R}_U \mathcal{I}(\lambda n^{\mathtt{N}} \mathcal{L} n^*)m) \\
&= \mathcal{L}m^*(\mathsf{R}_U \mathcal{I}(\lambda n^{\mathtt{N}} \mathcal{L} n^*)m) \\
&= \mathcal{L}m^*\mathcal{N}_m
\end{aligned}$$

By induction hypothesis, $\mathcal{N}_m \Vdash \mathsf{R}_U uvm \downarrow \dot{\mathcal{C}}$. Moreover, $m^* \Vdash m \downarrow \mathtt{N}$ and by hypothesis $\mathcal{L} \Vdash v \downarrow \mathtt{N} \to \dot{\mathcal{C}} \to \dot{\mathcal{C}}$. Hence

$$\mathcal{L}m^*\mathcal{N}_m \Vdash vm(\mathsf{R}_W uvm) = \mathcal{R}_W uv\mathsf{S}(m) \downarrow \dot{\mathcal{C}}$$

which is the thesis.

4.5.2. The Adequacy Theorem

We are finally ready to define the translation of every term of $\mathcal{F}_{\text{Class}}$.

Definition 22 (Realizers of Forcing for Terms of $\mathcal{F}_{\text{Class}}$). *For every proof-like term $v : T$ of system $\mathcal{F}_{\text{Class}}$, we define a term $[\![v]\!] : [\![T]\!]$ by induction on v and by cases as follows:*

1. *$v = c$, with c constant. We define $[\![c]\!]$ as in definition 21.*

2. *$v = z^A$, z variable. Then*
$$[\![z^A]\!] := z^{[\![A]\!]}$$

3. *$v = ut$. Then*
$$[\![ut]\!] := [\![u]\!][\![t]\!]$$

4. *$v = \lambda z^A u$. Then*
$$[\![\lambda z^A u]\!] := \lambda z^{[\![A]\!]}\ [\![u]\!]$$

5. *$v = uV$, with $u : \forall X_i U$ and $T = U[V/X_i]$. Then*
$$[\![uV]\!] := [\![u]\!][\![V]\!]\triangledown^V$$

6. $v = \Lambda X_i\, u$. *Then*

$$[\![\Lambda X_i\, u]\!] := \Lambda X_i\, \lambda x_i^{\mathsf{Cand}_{X_i}} [\![u]\!]$$

We are now able to prove our main theorem. For every closed term u of $\mathcal{F}_{\text{Class}}$, $[\![u]\!]$ forces u to be a computable functional.

Theorem 10 (Adequacy Theorem). *Let $w : A$ be a proof-like term of $\mathcal{F}_{\text{Class}}$ and let $z_1^{A_1}, \ldots, z_n^{A_n}$ contain all the free variables of w and $X_1, \ldots, X_m$ contain all the free type variables of $A_1, \ldots, A_n, A$. Then*

$$[\![\Lambda X_1 \ldots \Lambda X_m \lambda z_1^{A_1} \ldots \lambda z_n^{A_n} w]\!] \Vdash \Lambda X_1 \ldots \Lambda X_m \lambda z_1^{A_1} \ldots \lambda z_n^{A_n} w \downarrow \forall X_1, \ldots, X_m A_1 \to \ldots \to A_n \to A$$

Proof. Suppose

$$\mathcal{H}_1 \Vdash \mathcal{C}_1 \in \mathsf{Cand}_{U_1 \times V_1}, \ldots, \mathcal{H}_m \Vdash \mathcal{C}_m \in \mathsf{Cand}_{U_m \times V_m}$$

For any type B, set $\overline{B} := B[\dot{\mathcal{C}}_1/X_1 \ldots \dot{\mathcal{C}}_m/X_m]$, $\widetilde{B} := B[V_1/X_1 \ldots V_m/X_m]$ and $\widehat{B} := [U_1/X_1 \ldots U_m/X_m]$. (thus $\mathcal{M}_i : [\![\overline{A_i}]\!] = \widehat{[\![A_i]\!]}$ and $t_i : |\overline{A_i}| = \widetilde{A}_i$). Suppose also

$$\mathcal{M}_1 \Vdash t_1 \downarrow \overline{A_1}, \ldots, \mathcal{M}_n \Vdash t_n \downarrow \overline{A_n}$$

We have to prove that

$$[\![w]\!][U_1/X_1][\mathcal{H}_1/x_1^{\mathsf{Cand}_{U_1}}] \ldots [U_m/X_m][\mathcal{H}_m/x_m^{\mathsf{Cand}_{U_m}}][\mathcal{M}_1/z_1^{\widehat{[\![A_1]\!]}} \ldots \mathcal{M}_n/z_n^{\widehat{[\![A_n]\!]}}]$$

$$\Vdash$$

$$w[V_1/X_1 \ldots V_m/X_m][t_1/z_1^{\widetilde{A}_1} \ldots t_n/z_n^{\widetilde{A}_n}] \downarrow \overline{A}$$

For any term v, we set

$$\widetilde{v} := v[V_1/X_1 \ldots V_m/X_m][t_1/z_1^{\widetilde{A}_1} \cdots t_n/z_n^{\widetilde{A}_n}]$$

and

$$\widehat{[\![v]\!]} := [\![v]\!][U_1/X_1][\mathcal{H}_1/x_1^{\mathsf{Cand}_{U_1}}] \ldots [U_m/X_m][\mathcal{H}_m/x_m^{\mathsf{Cand}_{U_m}}][\mathcal{M}_1/z_1^{\widehat{[\![A_1]\!]}} \ldots \mathcal{M}_n/z_n^{\widehat{[\![A_n]\!]}}]$$

With that notation, we have to prove that $\widehat{[\![w]\!]} \Vdash \widetilde{w} \downarrow \overline{A}$. The proof is by induction on w and proceeds by cases, accordingly to the form of w.

1. $w = c$, with c constant. Since $[\![c]\!]$ is closed and c does not have free variables, by proposition 7

   $$\widehat{[\![c]\!]} = [\![c]\!] \Vdash c = \widetilde{c} \downarrow \overline{A}$$

   which is the thesis.

2. $w = z_i^{A_i}$, for some $1 \leq i \leq n$. Then

   $$\widehat{[\![w]\!]} = z_i^{\widehat{[\![A_i]\!]}}[\mathcal{M}_1/z_1^{\widehat{[\![A_1]\!]}} \ldots \mathcal{M}_n/z_n^{\widehat{[\![A_n]\!]}}] = \mathcal{M}_i \Vdash t_i = z_i^{\widetilde{A}_i}[t_1/z_1^{\widetilde{A}_1} \cdots t_n/z_n^{\widetilde{A}_n}] = \widetilde{w} \downarrow \overline{A}$$

   which is the thesis.

3. $w = ut$. By induction hypothesis, $\widehat{[\![u]\!]} \Vdash \widetilde{u} \downarrow \overline{B} \to \overline{A}$ and $\widehat{[\![t]\!]} \Vdash \widetilde{t} \downarrow \overline{B}$. So

   $$\widehat{[\![ut]\!]} = \widehat{[\![u]\!]}\, \widehat{[\![t]\!]} \Vdash \widetilde{u}\, \widetilde{t} = \widetilde{w} \downarrow \overline{A}$$

   which is the thesis.

4. $w = \lambda z^B u$. Suppose $\mathcal{M} \Vdash t \downarrow \overline{B}$. We have to prove that $\widehat{[\![w]\!]}\mathcal{M} \Vdash \widetilde{w}t \downarrow \overline{A}$. By induction hypothesis

$$\widehat{[\![\lambda z^B u]\!]}\mathcal{M} = (\lambda z^{\widehat{[\![B]\!]}}\widehat{[\![u]\!]})\mathcal{M} = \widehat{[\![u]\!]}[\mathcal{M}/z^{\widehat{[\![B]\!]}}] \Vdash \widetilde{u}[t/z^{\widetilde{B}}] = \widetilde{w}t \downarrow \overline{A}$$

which is the thesis.

5. $w = uV$, with $u : \forall XB$ and $A = B[V/X]$. Define

$$\mathcal{C} := \{\langle \mathcal{N}, t\rangle \mid \mathcal{N} \Vdash t \downarrow \overline{V}\}$$

By lemma 9, point 3 and since $[\![\overline{V}]\!] = \widehat{[\![V]\!]}$, $|\overline{V}| = \widetilde{V}$, we obtain

$$\nabla^{\overline{V}}[\mathcal{H}_1/z^{\dot{\mathcal{C}}_1}\ldots\mathcal{H}_m/z^{\dot{\mathcal{C}}_m}] \Vdash \mathcal{C} \in \mathsf{Cand}_{\widehat{[\![V]\!]}\times\widetilde{V}}$$

Moreover, we may suppose that none of the variables $z_i^{\widetilde{A}_i}$ occurs in ∇^V and none of the X_i occurs in U_i or $\mathcal{H}_i$. Therefore, by repeated application of lemma 9, point 1,

$$\begin{aligned}\widehat{(\nabla^V)} &= \nabla^V[U_1/X_1][\mathcal{H}_1/x_1^{\mathsf{Cand}_{U_1}}]\ldots[U_m/X_m][\mathcal{H}_m/x_m^{\mathsf{Cand}_{U_m}}]\\ &= \nabla^{\overline{V}}[\mathcal{H}_1/x_1^{\dot{\mathcal{C}}_1}\ldots\mathcal{H}_m/x_m^{\dot{\mathcal{C}}_m}]\end{aligned}$$

By induction hypothesis, $\widehat{[\![u]\!]} \Vdash \widetilde{u} \downarrow \forall X\overline{B}$. Therefore,

$$\widehat{[\![uV]\!]} = \widehat{[\![u]\!]}\;\widehat{[\![V]\!]}\;\widehat{(\nabla^V)} \Vdash \widetilde{u}\widetilde{V} \downarrow \overline{B}[\dot{\mathcal{C}}/X]$$

By lemma 9, point 4, we obtain

$$\widehat{[\![uV]\!]} \Vdash \widetilde{u}\widetilde{V} \downarrow \overline{B}[\overline{V}/X] = \overline{A}$$

which is the thesis.

6. $w = \Lambda X_i u$ and $A = \forall X_i\, B$, $i > n$. Since $\widehat{[\![\Lambda X_i u]\!]} = \Lambda X_i \lambda x_i^{\mathsf{Cand}_{X_i}}\widehat{[\![u]\!]}$, we have to suppose that $\mathcal{H} \Vdash \mathcal{C} \in \mathsf{Cand}_{U\times V}$, and then show

$$\widehat{[\![u]\!]}[U/X_i][\mathcal{H}/x_i^{\mathsf{Cand}_U}] \Vdash \widetilde{u}[V/X] \downarrow \overline{B}[\dot{\mathcal{C}}/X_i]$$

But this follows by the induction hypothesis.

4.5.3. Witness Extraction with the State-Extending-Continuation-Passing-Style Translation Method

At the end, we are able to show that our translation can be used for program extraction with Interactive realizability.

Theorem 11 (Program Extraction via SECPS-Translation). *Let t be a proof-like term of $\mathcal{F}_{\text{Class}}$ and suppose that $t \Vvdash \forall x^{\mathtt{N}}\exists y^{\mathtt{N}} Pxy$, with P atomic predicate of Gödel's T. Let $r : \mathsf{S}$ be any state. Define a term* wit *of $\mathcal{F}$ as follows:*

$$\mathsf{wit} := \lambda x^{\mathtt{N}}.\, \pi_0(tx)[[\![\lambda n^{\mathtt{N}}\pi_1(tn)]\!]x^*(\lambda z^{\mathsf{S}}\, z)r]$$

Then, for all numerals m, $Pm(\mathsf{wit}\, m) = \mathtt{True}$.

Proof. Let m be any numeral and let

$$s := [\![\lambda n^{\mathtt{N}}\pi_1(tn)]\!]m^*(\lambda z^{\mathsf{S}}\, z)r$$

Since $tm \Vvdash_s \exists y^{\mathtt{N}} Pmy$ and $\mathsf{wit}\, m = \pi_0(tm)[s]$, we have that

$$\pi_1(tm) \;\Vvdash_s\; Pm(\mathsf{wit}\, m)$$

which by definition 11 in particular means that

$$\pi_1(tm)[s] = \varnothing \implies Pm(\mathsf{wit}\, m) = \mathtt{True}$$

Thus, in order to obtain the thesis, it is enough to show that $\pi_1(tm)[s] = \varnothing$. By the Adequacy theorem 10,

$$[\![\lambda n^{\mathtt{N}}\pi_1(tn)]\!] \Vdash \lambda n^{\mathtt{N}}\pi_1(tn) \downarrow \mathtt{N} \to \mathtt{U}$$

Since $m^* \Vdash m \downarrow \mathtt{N}$, we obtain that

$$[\![\lambda n^{\mathtt{N}}\pi_1(tn)]\!]m^* \Vdash \pi_1(tm) \downarrow \mathtt{U}$$

Therefore, by definition 19, $\pi_1(tm)[s] \preceq s$ which means that

$$\mathsf{dom}(\pi_1(tm)[s]) \subseteq \mathsf{dom}(s)$$

Since $\pi_1(tm) \Vvdash_s Pm(\mathsf{wit}\, m)$, we also have that $\mathsf{dom}(\pi_1(tm)[s]) \cap \mathsf{dom}(s) = \emptyset$. Therefore, $\pi_1(tm)[s] = \varnothing$.

5. Conclusions and Related Works

We conclude the paper with some important remarks.

5.1. *Interactive Realizability and Krivine Classical Realizability*

We have introduced a new realizability interpretation based on states and learning for the classical system of second-order Arithmetic $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$. Then we have developed in full detail two program extraction techniques. The second one is a new CPS-translation and looks particularly interesting: it has a nice model theoretic meaning and it appears as a first step towards a more efficient computational interpretation of classical logic. Indeed, the defect of programs extracted from classical proofs is that they waste too many resources: when they backtrack, they tend to “forget” precious information.

For example, let us examine Krivine classical realizability [25] interpretation of excluded-middle with cc. Consider a computation of the form

$$\mathsf{cc}\, t \star \pi \succ t \star \mathsf{k}_\pi.\pi \succ \ldots \succ \mathsf{k}_\pi \star u.\rho \succ u \star \pi$$

The penultimate instruction executed in this computation is an operation of backtracking: the process takes the term u and put it in the context π of the first instruction. Thus, everything between the first and last instruction of the process is discarded, except for u. But it is unreasonable that everything must be erased; it might happen, for example, that between the first and last instruction some witness for some instance of the excluded middle EM_1 is learned.

Krivine’s realizability interpretation of choice axioms, such as countable choice and SK_1, suffers other inefficiency problems which, on the contrary, the interpretation of the excluded middle does not. Indeed, the choice functions are *defined* in the realizability model through the minimum principle. As explained in remark 2, this approach leads to a waste of resources: a choice function is allowed to return any witness, while imposing some arbitrary criterion over the witnesses that must be returned may force a realizer to change witness even if its current one is correct.

Realizability based on states seems an answer to this kind of efficiency issues. In classical logic, for writing down more efficient programs, it seems necessary to describe exactly: a) what the programs learn; b) how the knowledge of programs varies during the execution. The programs obtained by our SECPS-translation not only are able to backtrack as programs with cc can do, but also keep all the useful information coming from a failure. In fact, when they backtrack (i.e. when the program $[\![\Phi_i]\!]$ is executed), they restart the computation into a state coming from the execution of a state-extending continuation and thus possibly much bigger than the current one. That is, even in a dead branch of the execution, something useful might be learned and has to be kept. Moreover, no arbitrary definition of the choice function that interprets SK_1 is given and realizers are free to keep any witness they find during computations.

5.2. Interactive Realizability and Bar Recursive Interpretations of Analysis

As another example of inefficiency problems in classical logic, let us consider Spector's computational interpretation of Analysis by means of bar recursion [32]. Albeit it provides a very interesting computational reduction of the countable choice axiom to recursion over well-founded trees, Spector's bar recursion is dramatically inefficient. Choice functions are approximated not just in the values that must be used in some particular computation, but instead finite initial segments of the shape $f(0), f(1), f(2), \ldots, f(n)$ are built, regardless the fact that some of these values are not at all asked.

A nice solution to this problem has been provided by Berardi, Bezem and Coquand [9]. They have introduced a much more efficient bar recursion, that indeed may be called: *demand-driven* bar recursion. As the name suggests, choice functions are approximated only in the values that are asked during the computation. Its equational axiom (following Berger [11]) is:

$$(\mathsf{DDBR})YGHs = Y(\lambda n^{\mathtt{N}}\mathsf{if}\ n \in \mathsf{domain}(s)\ \mathsf{then}\ s(n)\ \mathsf{else}\ Gn(\lambda z^{A}(\mathsf{DDBR})YGHs \cup (n, z)))$$

where s is a finite partial function $\mathtt{N} \to A$, with $n \in \mathsf{domain}(s)$ and $s \cup (n, z)$ having the expected meaning, and $Y : (\mathtt{N} \to A) \to \mathtt{N}$, $G : \mathtt{N} \to (A \to \mathtt{N}) \to A$.

Unfortunately, with each recursive call of DDBR, the functional Y is recomputed from scratch even if the partial function s changes only in one point (it becomes $s \cup (n, z)$). This inefficiency is similar to the one of the Iterative Method that we have pointed out in section 4.1. Since the SECPS-translation was devised precisely to solve this issue, it provides a more efficient computational interpretation than demand-driven bar recursion (of course in our limited setting).

5.3. Comparison with Goodman's Forcing

Also Goodman [16] gave a forcing definition of what it means that a functional of finite type depending on some non-recursive function is "computable". The worlds considered by Goodman are partial functions order by inclusion and for him a term t of type $\mathtt{N}$ is "forced to be computable of type $\mathtt{N}$ by a partial function p" if t is defined in p, that is, if p contains all the values need for reducing t to a natural number. Instead, we use as states total functions; but we can easily rephrase Goodman's concept of being defined in our framework.

Since Goodman's definition refers to finite types, we concentrate on terms of $\mathcal{T}_{\text{Class}}$. We say that a term $t \in \mathcal{T}_{\text{Class}}$ is *defined in a state* $s : \mathtt{S}$, if for all $s' \geq s$, $t[s'] = t[s]$. Intuitively, if t is defined in s, then s contains all the information necessary to compute t, since increasing the information of s yields the same value. Then, Goodman's forcing relation $s\,\mathsf{gf}\,t \in A$, where t is a term of $\mathcal{T}_{\text{Class}}$ of type A and s is a state, can be defined as:

$$s\ \mathsf{gf}\ t \in \mathtt{N} \iff t \text{ is defined in } s$$

$$s\ \mathsf{gf}\ t \in A \to B \iff (\forall s' \geq s.\, s'\ \mathsf{gf}\ A \implies \exists s'' \geq s'.\, s''\ \mathsf{gf}\ B)$$

Then one defines

$$\mathsf{gf}\ t \in A \iff \forall s \exists s' \geq s.\, s'\ \mathsf{gf}\ t \in A$$

Unfortunately, from the constructive point of view, already the notion $s\,\mathsf{gf}\,t \in \mathtt{N}$ is unsustainably strong. Indeed, to prove $\mathsf{gf}\ t \in \mathtt{N}$ one needs a classical metatheory!

However, if one takes the Kreisel's no-counterexample-interpretation [23] of $\mathsf{gf}\ t \in \mathtt{N}$, one can define

$$s \Vdash t \in \mathtt{N} \iff \forall k \in \mathbb{K}\, \exists s' \geq s.\, t \downarrow [s, k(s)]$$

and $\Vdash t \in \mathtt{N}$ as $\forall s.\, s \Vdash t \in \mathtt{N}$. Then, applying modified realizability, one obtains our definition 13 of forcing $\mathcal{M} \Vdash t \in \mathtt{N}$. To put it in another way, one replaces the unrestricted quantification over future worlds, typical of forcing semantics, with a restricted one: the future that can be considered is only the future of the computation.

5.4. Interactive Realizability, Avigad's Forcing and Update Procedures

Avigad [7] showed how to use forcing to eliminate the use of Skolem axioms in classical proofs. Unfortunately, the resulting proofs are classical, because they still use excluded middle. Anyway, Avigad's forcing have an interesting computational interpretation, because composing it with any negative translation of classical logic into intuitionistic logic allows one to extract programs from the original proofs that use Skolem axioms.

While at the time of [4] it was not completely clear how Interactive realizability relates to forcing, it is now evident that the theory of Interactive realizability, as developed in this paper, is a direct constructivization of Avigad's forcing. In fact, we have proved conservativity of $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ over HAS for Π^0_2-formulas.

Another contribution of this paper is related to Avigad's update procedures. The concept of update procedure [2, 6] provides an axiomatization of the computational content of the epsilon substitution method. However, with the current technology it was not possible to extract update procedures from impredicative systems. Here, we have showed how to do that, since realizers of Σ^0_1-formulas *are* update procedures.

5.5. Interactive Realizability and Friedman's Translation

The same result of Aschieri and Berardi [5] holds here: Interactive realizability can be characterized as a new way of using Friedman's translation, which allows to extract programs from classical proofs without passing from logical negative translations.

5.6. Forcing Semantics, Evolution of Knowledge and Interactive Realizability

There is another observation which is worth to be done. A limit of the intuitionistic forcing semantics (Goodman, Kripke [24] etc.) is that while they depict, correctly, knowledge and mathematical ability as something which grows in time, they remain completely silent about *how this knowledge evolves*. Indeed, Interactive Realizability shows that the computational content of a classical proof is a state-extending operator. Moreover, we have shown that the interactive realizers can be given a constructive forcing semantics explaining their stability with respect any evolution of the future given by any continuation. In our opinion, Kripke semantics is much more interesting in the classical setting than in the intuitionistic one!

Appendix A. Interactive Realizability for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ plus Ex-Falso-Quodlibet

As remarked in section 3.4, the system $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ does not prove the full *ex-falso-quodlibet* principle:

$$\forall X. \bot \to X0$$

There were good reasons to leave it out of our system. The axiom is not very interesting from the computational point of view, because it is usually interpreted by dummy realizers. On top of that, the ex-falso also creates annoying technical issues, since there are no closed terms of type $|\forall X. \bot \to X0| = \forall X. \mathtt{U} \to X$ in system $\mathcal{F}$. Furthermore, the trivial translation mapping atomic formulas of the form Xt to $Xt \vee \bot$ eliminates the need of the full ex-falso axiom in the system $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$.

In our framework as well, the ex-falso may be interpreted by dummy realizers, albeit in a slightly less trivial way than in intuitionistic logic. In this appendix, we sketch two approaches. First, we show how to formulate Interactive realizability in such a way that the ex-falso-quodlibet axiom is realizable by a term of System $\mathcal{F}$ (and for avoiding to change notations, we shall just "overwrite" the definitions of section 3). Then, we show how the first approach can be drastically simplified, but at the cost of adding dummy constants to pure $\mathcal{F}$.

Appendix A.1. First Approach: Realizing Ex-Falso-Quodlibet in pure $\mathcal{F}$

In classical realizability, it is not entirely trivial to realize an axiom of the form $\bot \to A$, since in any state s there is a realizer of $\bot$: it is enough to take a sound update $\overline{U}$ such that $\mathsf{dom}(U) \cap \mathsf{dom}(s) = \emptyset$. Therefore, given such an update U, one must show that is possible to define a dummy realizer of A. Indeed, in a first-order setting that is very easy. But in second-order logic it is technically more complicated and one needs to charge every s-saturated set to provide a constructor, taking an update and returning an element of the set.

Definition 23 (s-Saturated Sets). *Let A be a closed type of $\mathcal{F}$ and s be any state. Let S be a set of closed type-A terms of $\mathcal{F}_{Class}$ such that if $t \in S$ and $t[s] = u[s]$ in $\mathcal{F}$, then $u \in S$.*

1. *We say that a closed term $v : \mathtt{U} \to A$ of System $\mathcal{F}$* realizes that S is a s-saturated set*, and write $v \Vdash S \in \mathsf{Sat}_A(s)$, if for all non-empty sound updates U such that* $\mathsf{dom}(U) \cap \mathsf{dom}(s) = \emptyset$*, it is true that $v\overline{U} \in S$.*

2. *Let $\mathcal{L}^+_{Class}$ the language resulting from $\mathcal{L}_{Class}$ by adding a constant symbol $\dot{F}$ for every function $F : \mathbb{N} \to \mathsf{Sat}_A(s)$. We write $v \Vdash F : \mathbb{N} \to \mathsf{Sat}_A(s)$ if for all numbers n, $v \Vdash F(n) \in \mathsf{Sat}_A(s)$*

For every formula A of $\mathcal{L}^+_{\text{Class}}$, we are now going to define what type $|A|$ a realizer of A must have. The difference with section 3 is just in the types $|\forall X A|, |\exists X A|$ which must account for an extra argument, representing a realizer of a s-saturated set.

Definition 24 (Types for realizers). *For each formula A of $\mathcal{L}^+_{Class}$ we define a type $|A|$ of $\mathcal{F}_{Class}$ by induction on A:*

1. $|P| = \mathtt{U}$, *when P is atomic,*
2. $|Xt| = X$,
3. $|\dot{F}t| = C$ *if* $F : \mathbb{N} \to \mathsf{Sat}_C(s)$,
4. $|A \wedge B| = |A| \times |B|$,
5. $|A \vee B| = \mathtt{Bool} \times (|A| + |B|)$,
6. $|A \to B| = |A| \to |B|$,
7. $|\forall x A| = \mathtt{N} \to |A|$,
8. $|\forall X A| = \forall X. (\mathtt{U} \to X) \to |A|$,
9. $|\exists x A| = \mathtt{N} \times |A|$,
10. $|\exists X A| = \exists X. (\mathtt{U} \to X) \times |A|$

We now define the realizability relation $t \Vdash C$, where $t \in \mathcal{F}_{\text{Class}}$, $C \in \mathcal{L}^+_{\text{Class}}$ and $t : |C|$. The difference with section 3 lies just in the definition of $t \Vdash_s \forall X A, \exists X A$.

Definition 25 (Interactive Realizability). *Assume s is a state, t is a closed term of $\mathcal{F}_{Class}$, $C \in \mathcal{L}^+_{Class}$ is a closed formula, and $t : |C|$. We define first the relation $t \Vdash_s C$ by induction and by cases according to the form of C:*

1. *$t \Vdash_s Q$ if and only if:*
   - *$t[s] = \overline{U}$ implies that U is sound and* $\mathsf{dom}(U) \cap \mathsf{dom}(s) = \emptyset$
   - *$t[s] = \varnothing$ implies $Q[s] =$* $\mathtt{True}$
2. *$t \Vdash_s \dot{F}u$ if and only if for some numeral n, $u[s] = n$ and $t \in F(n)$*
3. *$t \Vdash_s A \wedge B$ if and only if $\pi_0 t \Vdash_s A$ and $\pi_1 t \Vdash_s B$*

4. $t \Vdash_s A \vee B$ *if and only if either* $\pi_0 t[s] = \texttt{True}$, $\pi_1 t[s] = \iota_{0,|A|,|B|}(u)$ *and* $u \Vdash_s A$, *or* $\pi_0 t[s] = \texttt{False}$, $\pi_1 t[s] = \iota_{1,|A|,|B|}(v)$ *and* $v \Vdash_s B$

5. $t \Vdash_s A \to B$ *if and only if for all* u, *if* $u \Vdash_s A$, *then* $tu \Vdash_s B$

6. $t \Vdash_s \forall x A$ *if and only if for all numerals* n, $tn \Vdash_s A[n/x]$

7. $t \Vdash_s \exists x A$ *if and only for some numeral* n, $\pi_0 t[s] = n$ *and* $\pi_1 t \Vdash_s A[n/x]$

8. $t \Vdash_s \forall X A$ *if and only if for all* v, *if* $v \Vdash F \in \mathbb{N} \to \mathsf{Sat}_B(s)$ *then* $tBv \Vdash_s A[\dot{F}/X]$

9. $t \Vdash_s \exists X A$ *if and only if* $t = \langle B, u\rangle$, $\pi_0 u \Vdash F \in \mathbb{N} \to \mathsf{Sat}_B(s)$ *and* $\pi_1 u \Vdash_s A[\dot{F}/X]$

We define $t \Vdash A$ *if and only if for all states* s, $t \Vdash_s A$.

We are going to prove that interpretations of formulas in our realizability model are saturated sets. Before, we need some notation.

Notation. In the following, we shall assume that the type variables of $\mathcal{F}_{\text{Class}}$ are $X_0, X_1, \ldots, X_n \ldots$ (but when the index is not important, we shall denote them with generical metavariables $X, Y, \ldots$). To each type variable X_i we associate a term variable $x_i^{\mathtt{U}\to X_i}$. Moreover, we assume to have for each $F : \mathbb{N} \to \mathsf{Sat}_A(s)$ a term variable $x^{\dot{F}}$ of type $\mathtt{U} \to A$.

We start by defining the terms dum_A that will used as constructors, taking updates as input and returning dummy members of saturated sets as output.

Definition 26 (Dummy Realizers). *Let* A *a be a formula of* $\mathcal{L}^+_{\text{Class}}$ *and* z *a variable of type* $\mathtt{U}$. *We define by induction on* A *a term* $\mathsf{dum}_A(z)$ *of type* $|A|$.

1. $A = P$, *with* P *atomic. Then*
$$\mathsf{dum}_A(z) = z$$

2. $A = X_i t$, *where* X_i *is a variable. Then*
$$\mathsf{dum}_A(z) := x_i^{\mathtt{U}\to X_i} z$$

3. $A = \dot{F} t$. *Then*
$$\mathsf{dum}_A(z) := x^{\dot{F}} z$$

4. $A = B \to C$. *Then*
$$\mathsf{dum}_A(z) := \lambda y^{|B|}\, \mathsf{dum}_C(z) \text{ \textit{(with y fresh)}}$$

5. $A = B \wedge C$. *Then*
$$\mathsf{dum}_A(z) := \langle \mathsf{dum}_B(z), \mathsf{dum}_C(z)\rangle$$

6. $A = B \vee C$. *Then*
$$\mathsf{dum}_A(z) := \langle \texttt{True}, \iota_{0,|B|,|C|}(\mathsf{dum}_B(z))\rangle$$

7. $A = \forall x^{\mathtt{N}} B$. *Then*
$$\mathsf{dum}_A(z) := \lambda y^{\mathtt{N}}\, \mathsf{dum}_B(z)$$

8. $A = \forall X_i B$. *Then*
$$\mathsf{dum}_A(z) := \Lambda X_i\, \lambda x_i^{\mathtt{U}\to X_i} \mathsf{dum}_B(z)$$

9. $A = \exists x^{\mathtt{N}} B$. *Then*
$$\mathsf{dum}_A(z) := \langle 0, \mathsf{dum}_B(z) \rangle$$

10. $A = \exists X_i B$. *Then*
$$\mathsf{dum}_A(z) := \langle \mathtt{U}, \langle \lambda z^{\mathtt{U}} z, \mathsf{dum}_B(z)[\mathtt{U}/X_i][\lambda z^{\mathtt{U}} z / x_i^{\mathtt{U} \to \mathtt{U}}] \rangle \rangle$$

We now prove that the dummy term $\mathsf{dum}_A(\overline{U})$ realizes in a state s any closed formula A of $\mathcal{L}_{\text{Class}}$, whenever U is sound, non-empty and $\mathsf{dom}(U) \cap \mathsf{dom}(s) = \emptyset$.

Lemma 12 (Dummy Realizers). *Assume A is a formula of $\mathcal{L}^{+}_{\textit{Class}}$ without free number variables. The following hold:*

1. *If X_i occurs free in A and $v_i \Vdash F_i \in \mathbb{N} \to \mathsf{Sat}_{V_i}(s)$, then*
$$\mathsf{dum}_A(z)[V_i/X_i][v_i/x_i^{\mathtt{U} \to V_i}] = \mathsf{dum}_{A[\dot{F}_i/X_i]}(z)[v_i/x^{\dot{F}_i}]$$

2. *Suppose that for $i = 1, \ldots, n$, $v_i \Vdash F_i \in \mathbb{N} \to \mathsf{Sat}_{V_i}(s)$ and that each set constant of A is equal to some $\dot{F}_i$. Let U be a non-empty sound update such that $\mathsf{dom}(U) \cap \mathsf{dom}(s) = \emptyset$. Then*
$$\mathsf{dum}_A(\overline{U})[v_1/x^{\dot{F}_1} \ldots v_n/x^{\dot{F}_n}] \ \Vdash_s \ A$$

Proof. 1. By induction on A (which is not of the form $\dot{F}t$ or P, with P atomic predicate, since X_i occurs in A).

(a) $A = X_i t$, with X_i second-order variable. Then,
$$\begin{aligned}\mathsf{dum}_A(z)[V_i/X_i][v_i/x_i^{\mathtt{U} \to V_i}] &= x_i^{\mathtt{U} \to X_i} z[V_i/X_i][v_i/x_i^{\mathtt{U} \to V_i}] \\ &= v_i z \\ &= x^{\dot{F}_i} z[v_i/x^{\dot{F}_i}] \\ &= \mathsf{dum}_{A[\dot{F}/X_i]}(z)[v_i/x^{\dot{F}_i}]\end{aligned}$$

which is the thesis.

(b) $A = B \to C$. Then, since $|B|[V_i/X_i] = |B[\dot{F}_i/X_i]|$, we have
$$\begin{aligned}\mathsf{dum}_A[V_i/X_i][v_i/x_i^{\mathtt{U} \to V_i}](z) &= \lambda y^{|B|} \mathsf{dum}_C(z)[V_i/X_i][v_i/x_i^{\mathtt{U} \to V_i}] \\ &\overset{\text{ind.}}{=} \lambda y^{|B[\dot{F}_i/X_i]|}\, \mathsf{dum}_{C[\dot{F}_i/X_i]}(z)[v_i/x^{\dot{F}_i}] \\ &= \mathsf{dum}_{A[\dot{F}_i/X_i]}[v_i/x^{\dot{F}_i}]\end{aligned}$$

which is the thesis.

(c) $A = \forall X_j B$. Then $X_i \neq X_j$, since X_i occurs free in A. Moreover,
$$\begin{aligned}\mathsf{dum}_A(z)[V_i/X_i][v_i/x_i^{\mathtt{U} \to V_i}] &= \Lambda X_j \lambda x_j^{\mathtt{U} \to X_j} (\mathsf{dum}_B(z)[V_i/X_i][v_i/x_i^{\mathtt{U} \to V_i}]) \\ &\overset{\text{ind.}}{=} \Lambda X_j \lambda x_j^{\mathtt{U} \to X_j} \mathsf{dum}_{B[\dot{F}_i/X_i]}(z)[v_i/x^{\dot{F}_i}] \\ &= \mathsf{dum}_{A[\dot{F}_i/X_i]}(z)[v_i/x^{\dot{F}_i}]\end{aligned}$$

which is the thesis.

(d) The other cases are trivial.

2. By induction on A (which by assumption is closed).

(a) $A = P$, with P atomic. Then

$$\mathsf{dum}_A(\overline{U})[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}] = \overline{U} \ \Vdash_s \ P$$

by definition 25 of realizability and since $U \neq \emptyset$.

(b) $A = \dot{F}_i t$. Then

$$\begin{aligned}\mathsf{dum}_A(\overline{U})[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}] &= x_i^{\dot{F}_i}\overline{U}[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}] \\ &= v_i\overline{U}\end{aligned}$$

Letting $t[s] = n$, since $v_i \ \Vdash \ F_i(n) \in \mathsf{Sat}_{V_i}(s)$, we obtain by definition 23 that

$$v_i\overline{U} \in F_i(n)$$

and therefore by definition 25 of realizability

$$v_i\overline{U} \ \Vdash_s \ \dot{F}(t)$$

which is the thesis.

(c) $A = B \to C$. Suppose $t \Vdash_s B$. We have to show that

$$\begin{aligned}&\mathsf{dum}_A(\overline{U})[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}]t \\ &= (\lambda y^{|B|}\mathsf{dum}_C(\overline{U}))t[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}] \\ &= \mathsf{dum}_C(\overline{U})[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}] \\ &\Vdash_s C\end{aligned}$$

which is true by induction hypothesis.

(d) $A = \forall X_{n+1}B$. Suppose that $v_{n+1} \ \Vdash \ F_{n+1} \in \mathbb{N} \to \mathsf{Sat}_{V_{n+1}}(s)$. We have to show that

$$\begin{aligned}&\mathsf{dum}_A(\overline{U})[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}]V_{n+1}v_{n+1} \\ &= (\Lambda X_{n+1}\lambda x_{n+1}^{\mathtt{U}\to X_{n+1}}\, \mathsf{dum}_B(\overline{U})[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}])V_{n+1}v_{n+1} \\ &= \mathsf{dum}_B(\overline{U})[V_{n+1}/X_{n+1}][v_{n+1}/x_{n+1}^{\mathtt{U}\to X_{n+1}}][v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}] \\ &\overset{\text{by (1.)}}{=} \mathsf{dum}_{B[\dot{F}/X_{n+1}]}(\overline{U})[v_1/x^{\dot{F}_1} \dots v_{n+1}/x^{\dot{F}_{n+1}}] \\ &\Vdash_s \ A[\dot{F}_{n+1}/X_{n+1}]\end{aligned}$$

which is true by induction hypothesis.

(e) $A = \exists X_{n+1}A$. Let F_{n+1} be the function mapping any number to the set of all closed terms of $\mathcal{F}_{\text{Class}}$ of type $\mathtt{U}$. Then $\lambda z^{\mathtt{U}}z \Vdash F_{n+1} \in \mathbb{N} \to \mathsf{Sat}_{\mathtt{U}}(s)$. We have to show that

$$\begin{aligned}&\mathsf{dum}_A(\overline{U})[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}] \\ &= \langle \mathtt{U}, \langle \lambda z^{\mathtt{U}}z, \mathsf{dum}_B(\overline{U})[\mathtt{U}/X_{n+1}][\lambda z^{\mathtt{U}}z/x_{n+1}^{\mathtt{U}\to\mathtt{U}}]\rangle\rangle[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}] \\ &\overset{\text{by (1.)}}{=} \langle \mathtt{U}, \langle \lambda z^{\mathtt{U}}z, \mathsf{dum}_{B[\dot{F}_{n+1}/X_{n+1}]}(\overline{U})[v_1/x^{\dot{F}_1} \dots v_n/x^{\dot{F}_n}\lambda z^{\mathtt{U}}z/x^{\dot{F}_{n+1}}]\rangle\rangle\end{aligned}$$

By induction hypothesis

$$\mathsf{dum}_{B[\dot{F}_{n+1}/X_{n+1}]}(\overline{U})[v_1/x^{\dot{F}_1}\dots v_n/x^{\dot{F}_n}\lambda z^{\mathtt{U}}z/x^{\dot{F}_{n+1}}] \Vdash_s B[\dot{F}_{n+1}/X_{n+1}]$$

which is the thesis.

(f) The other cases are trivial.

The following proposition remains valid.

Proposition 8 (Comprehension). *Let $B(x)$ be a formula of $\mathcal{L}^{+}_{Class}$ in the only free natural number variable x. Define $\mathsf{B} : \mathbb{N} \to \mathsf{Sat}_{|B|}(s)$ as*

$$\mathsf{B} := n \mapsto \{t \mid t \Vdash_s B(n)\}$$

Then for every t

$$t \Vdash_s A[\dot{\mathsf{B}}/X] \iff t \Vdash_s A[\lambda x B(x)/X]$$

Proof. Same proof of proposition 2.

In figure A.3 we have modified the term assignment rules of figure 2 in order to take account of the ex-falso-quodlibet and of the new definition of realizability. The changes are very minor and involve only the second-order quantifiers.

Ex-Falso-Quodlibet Axiom $\dfrac{}{\Gamma \vdash \Lambda X \lambda x^{\mathtt{U}\to X}\lambda z^{\mathtt{U}}.\, xz : \forall X.\, \bot \to X0}$

Universal Quantification (2) $\dfrac{\Gamma \vdash u : \forall X A}{\Gamma \vdash u|B|\lambda z^{\mathtt{U}}\mathsf{dum}_B(z) : A[\lambda x B(x)/X]}$ $\qquad \dfrac{\Gamma \vdash u : A}{\Gamma \vdash \Lambda X \lambda x^{\mathtt{U}\to X} u : \forall X A}$

where $B(x)$ is a formula of $\mathcal{L}_{\text{Class}}$ and X does not occur free in any formula occurring in Γ.

Existential Quantification (2) $\dfrac{\Gamma \vdash u : A[\lambda x B(x)/X]}{\Gamma \vdash \langle |B|, \langle \lambda z^{\mathtt{U}}\mathsf{dum}_B(z), u\rangle\rangle : \exists X A}$ $\qquad \dfrac{\Gamma \vdash u : \exists X A \quad \Gamma \vdash t : \forall X.\, A \to C}{\Gamma \vdash u|C|(\Lambda X \lambda x^{(\mathtt{U}\to X)\times|A|} tX(\pi_0 x)(\pi_1 x)) : C}$

where X is not free in C nor in any formula occurring in Γ.

Figure A.3: Term Assignement Rules for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1 + \forall X.\, \bot \to X0$

We now are now able to prove that every theorem of $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1 + \forall X.\, \bot \to X0$ is realizable. As usual in adequacy proofs for realizability, we prove a stronger version of the theorem, suitable to be proved by induction on proofs.

Theorem 13 (Adequacy Theorem). *Suppose that $\Gamma \vdash w : A$ in the system $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1 + \forall X.\, \bot \to X0$, with $\Gamma = z_1 : A_1, \dots, z_n : A_n$, and that the free variables of the formulas occurring in Γ and A are among $\alpha_1 : \mathtt{N}, \dots, \alpha_k : \mathtt{N}, X_1, \dots, X_m$. Fix any state s, numerals $n_1, \dots, n_k$ and $F_1 : \mathbb{N} \to \mathsf{Sat}_{B_1}(s), \dots, F_m : \mathbb{N} \to \mathsf{Sat}_{B_m}(s)$. For every formula C, let $\overline{C} := C[n_1/\alpha_1 \cdots n_k/\alpha_k\ \dot{F}_1/X_1 \cdots \dot{F}_m/X_m]$. Suppose $t_1, \dots, t_n, v_1, \dots, v_m$ are terms such that*

$$\text{for } i = 1, \dots, n,\ t_i \Vdash_s \overline{A_i},\ \ v_i \Vdash F_i \in \mathbb{N} \to \mathsf{Sat}_{B_i}(s)$$

Then

$$w[B_1/X_1 \cdots B_m/X_m][v_1/x_1^{\mathtt{U}\to B_1} \dots v_m/x_m^{\mathtt{U}\to B_m}][t_1/z_1^{|\overline{A_1}|} \cdots t_n/z_n^{|\overline{A_n}|}\ n_1/\alpha_1 \cdots n_k/\alpha_k] \Vdash_s \overline{A}$$

Proof. Notation: for any term v, we denote

$$v[B_1/X_1 \cdots B_m/X_m][v_1/x_1^{\mathtt{U}\to B_1} \dots v_m/x_m^{\mathtt{U}\to B_m}][t_1/z_1^{|\overline{A_1}|} \cdots t_n/z_n^{|\overline{A_n}|}\ n_1/\alpha_1 \cdots n_k/\alpha_k]$$

with $\overline{v}$. We have

$$|\overline{C}| = |C[\dot{F}_1/X_1 \cdots \dot{F}_m/X_m]| = |C|[B_1/X_1 \cdots B_m/X_m]$$

for all formulas C. We denote with $=$ the provable equality in $\mathcal{F}_{\text{Class}}$. We proceed by induction on w. Consider the last rule in the derivation of $\Gamma \vdash w : A$:

1. If it is the (second order) $\forall E$ rule, then $w = u|C|\lambda z^{\mathtt{U}}\mathsf{dum}_C(z)$, $A = B[\lambda x C(x)/X]$ and $\Gamma \vdash u : \forall X B$. Since the only free term variables in $\lambda z^{\mathtt{U}}\mathsf{dum}_C(z)$ are of the form $x_i^{\mathtt{U}\to X_i}$, we have

$$\overline{w} = \overline{u}|\overline{C}| \left(\lambda z^{\mathtt{U}}\mathsf{dum}_C(z)[B_1/X_1 \cdots B_m/X_m][v_1/x_1^{\mathtt{U}\to B_1} \ldots v_m/x_m^{\mathtt{U}\to B_m}]\right)$$

   Define

$$\mathsf{C} := n \mapsto \{t \mid t \Vdash_s \overline{C}(n)\}$$

   By repeated application of lemma 12, point 1) and some rearrangement of the substitution, we get

$$\begin{aligned}&\lambda z^{\mathtt{U}}\,\mathsf{dum}_C(z)[B_1/X_1 \cdots B_m/X_m][v_1/x_1^{\mathtt{U}\to B_1} \ldots v_m/x_m^{\mathtt{U}\to B_m}]\\ =&\lambda z^{\mathtt{U}}\,\mathsf{dum}_{\overline{C}}(z)[v_1/x^{\dot{F}_1} \ldots v_m/x^{\dot{F}_m}]\end{aligned}$$

   By lemma 12, point 2), we obtain that

$$\lambda z^{\mathtt{U}}\,\mathsf{dum}_{\overline{C}}(z)[v_1/x^{\dot{F}_1} \ldots v_m/x^{\dot{F}_m}] \Vdash \mathsf{C} \in \mathbb{N} \to \mathsf{Sat}_{|\overline{C}|}(s)$$

   By inductive hypothesis $\overline{u} \Vdash_s \forall X\overline{B}$ and so $\overline{w} \Vdash_s \overline{B}[\dot{\mathsf{C}}/X]$. By proposition 11, we conclude that

$$\overline{w} \Vdash_s \overline{B}[\lambda x\overline{C}(x)/X]$$

2. If it is the (second order) $\forall I$ rule, then $w = \Lambda X_k \lambda x^{\mathtt{U}\to X_k} u$, $A = \forall X_k B$ and $\Gamma \vdash u : B$ (and X does not occur free in the formulas of Γ). So, $\overline{w} = \Lambda X_k \lambda x_k^{\mathtt{U}\to X_k}\,\overline{u}$, since $X_k \neq X_1, \ldots, X_m$. Suppose $v \Vdash F \in \mathbb{N} \to \mathsf{Sat}_C(s)$; we have to prove that $\overline{w}Cv = \overline{u}[C/X_k][v/x_k^{\mathtt{U}\to X_k}] \Vdash_s \overline{B}[\dot{F}/X]$, which amounts to show that the induction hypothesis can be applied to u. For this purpose, it is enough to observe that for $i = 1, \ldots, n$

$$t_i \Vdash_s \overline{A_i} = \overline{A_i}[\dot{F}/X]$$

3. If it is the (second order) $\exists E$ rule, then $\Gamma \vdash t : \forall X.\, B \to A$, $\Gamma \vdash u : \exists X B$ and

$$w = u|A|(\Lambda X \lambda x^{(\mathtt{U}\to X)\times|B|} tX(\pi_0 x)(\pi_1 x))$$

   with X not occurring free in A nor in the formulas of Γ. By inductive hypothesis on u, $\overline{u} \Vdash_s \exists X\overline{B}$; hence $\overline{u} = \langle C, v\rangle$, $\pi_0 v \Vdash F \in \mathbb{N} \to \mathsf{Sat}_C(s)$ and $\pi_1 v \Vdash_s \overline{B}[\dot{F}/X]$. By induction hypothesis on t, $\overline{t} \Vdash_s \forall X.\, \overline{B} \to \overline{A}$ and hence

$$\overline{t}C(\pi_0 v)(\pi_1 v) \Vdash_s \overline{A}[\dot{F}/X] = \overline{A}$$

   Moreover

$$\begin{aligned}\overline{w} &= \langle C, v\rangle|\overline{A}|\overline{t}\\ &\overset{\text{def. 2}}{=} (\Lambda Y \lambda x^{\forall X.\,(\mathtt{U}\to X)\times\overline{B}\to Y}\, xCv)|\overline{A}|(\Lambda X \lambda x^{(\mathtt{U}\to X)\times|\overline{B}|} tX(\pi_0 x)(\pi_1 x))\\ &= (\Lambda X \lambda x^{(\mathtt{U}\to X)\times|\overline{B}|} tX(\pi_0 x)(\pi_1 x))Cv\\ &= \overline{t}C(\pi_0 v)(\pi_1 v)\end{aligned}$$

   We thus obtain by saturation (proposition 1)

$$\overline{w} \Vdash_s \overline{A}$$

4. If it is the (second order) $\exists I$ rule, then $\Gamma \vdash u : B[\lambda x C(x)/X]$,

$$w = \langle |C|, \langle \lambda z^{\mathtt{U}} \mathsf{dum}_C(z), u\rangle\rangle$$

and $A = \exists X B$. So,

$$\overline{w} = \langle |\overline{C}|, (\lambda z^{\mathtt{U}} \mathsf{dum}_C(z)[B_1/X_1 \cdots B_m/X_m][v_1/x_1^{\mathtt{U}\to B_1} \ldots v_m/x_m^{\mathtt{U}\to B_m}]), \overline{u}\rangle\rangle$$

Moreover, by induction hypothesis

$$\overline{u} \Vdash_s \overline{B}[\lambda x \overline{C}(x)/X]$$

Define

$$\mathsf{C} := n \mapsto \{t \mid t \Vdash_s \overline{C}(n)\}$$

We have already proved that

$$\begin{aligned} &\lambda z^{\mathtt{U}}\, \mathsf{dum}_C(z)[B_1/X_1 \cdots B_m/X_m][v_1/x_1^{\mathtt{U}\to B_1} \ldots v_m/x_m^{\mathtt{U}\to B_m}] \\ =&\lambda z^{\mathtt{U}}\, \mathsf{dum}_{\overline{C}}(z)[v_1/x^{\dot{F_1}} \ldots v_m/x^{\dot{F_m}}] \\ \Vdash& \mathsf{C} \in \mathbb{N} \to \mathsf{Sat}_{|\overline{C}|}(s) \end{aligned}$$

Moreover, by proposition 11, $\overline{u} \Vdash_s \overline{B}[\dot{\mathsf{C}}/X]$, and thus we have the thesis.

5. The other cases are treated as in the proof of theorem 5.

As corollary of the Adequacy theorem 13, we obtain the main theorem.

Theorem 14. *If A is a closed formula such that $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1 + \forall X.\bot \to X0 \vdash t : A$, then $t \Vdash A$.*

Appendix A.1.1. Second Approach: Dummy Constants

Since in our first solution the ex-falso-quodlibet axiom is interpreted in a dummy way, it is perhaps more efficient and more direct to introduce dummy constants encoding the realizers of our first solution. This approach is more in line with the approaches exposed in Girard [20] (§10.6) and Schwichtenberg and Troelstra [30] in the case of HAS.

We add to system $\mathcal{F}_{\text{Class}}$ a constant c of type $\forall X.\mathtt{U} \to X$. For every type A and term $v : \mathtt{U}$, we shall denote $\mathsf{c}Av$ with c_v^A. Let $\mathcal{R}$ be the set of the following reduction rules for the constant c:

$$\begin{aligned} \mathsf{c}_v^{A \to B} t &\mapsto \mathsf{c}_v^B \\ \pi_i \mathsf{c}_v^{A_1 \times A_2} &\mapsto \mathsf{c}_v^{A_i} \\ \mathsf{c}_v^{\forall X A} B &\mapsto \mathsf{c}_v^{A[B/X]} \\ \mathsf{c}_v^{\mathtt{N}} &\mapsto 0 \\ \mathsf{c}_v^{\mathtt{Bool}} &\mapsto \mathtt{True} \\ \mathsf{c}_v^{\mathtt{U}} &\mapsto v \end{aligned}$$

As the rules suggest themselves, c_v^A is intended to be a constant functional always returning fixed values of atomic types. The idea is that the terms of the form c_v^A will be used as dummy realizers of the ex-falso-quodlibet axiom, taking the role of the dummy terms $\mathsf{dum}_A(v)$ of section Appendix A.1. The extra power of the constant c allows to relieve saturated sets of providing constructors for dummy terms, thus simplifying the definitions of section Appendix A.1.

Now we have to redefine the concept of saturated set and define a realizability notion which is perfectly equal to the one for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ but the range of the notion of saturated set occurring in the definition. The types $|A|$ of realizers remain the same of the definition 10 of section 3, differently from what was done in section Appendix A.1.

Definition 27 (Interactive Realizability). *We define:*

1. *For every closed type A, let $\mathsf{Sat}_A(s)$ be the set of sets S of closed type-A terms of $\mathcal{F}_{Class} + \mathcal{C}$ such that:*

   i) if $t \in S$ and $t[s] = u[s]$ in $\mathcal{F} + \mathsf{c} + \mathcal{R}$, then $u \in S$;

   ii) if $U \neq \emptyset$, U is sound and $\mathsf{dom}(s) \cap \mathsf{dom}(U) = \emptyset$, then $\mathsf{c}_U^A \in S$.

2. *Assume s is a state, t is a closed term of $\mathcal{F}_{Class} + \mathsf{c}$, $D \in \mathcal{L}^+_{Class}$ is a closed formula, and $t : |D|$. We first define by induction on D the relation $t \Vvdash_s D$:*
   (a) *$t \Vvdash_s Q$ if and only if:*
      - *$t[s] = \overline{U}$ implies that U is sound and $\mathsf{dom}(U) \cap \mathsf{dom}(s) = \emptyset$*
      - *$t[s] = \varnothing$ implies $Q[s] = \mathtt{True}$*

   (b) *$t \Vvdash_s \dot{F}u$ if and only if for some numeral n, $u[s] = n$ and $t \in F(n)$*

   (c) *$t \Vvdash_s A \wedge B$ if and only if $\pi_0 t \Vvdash_s A$ and $\pi_1 t \Vvdash_s B$*

   (d) *$t \Vvdash_s A \vee B$ if and only if either $\pi_0 t[s] = \mathtt{True}$, $\pi_1 t[s] = \iota_{0,|A|,|B|}(u)$ and $u \Vvdash_s A$, or $\pi_0 t[s] = \mathtt{False}$, $\pi_1 t[s] = \iota_{1,|A|,|B|}(v)$ and $v \Vvdash_s B$*

   (e) *$t \Vvdash_s A \to B$ if and only if for all u, if $u \Vvdash_s A$, then $tu \Vvdash_s B$*

   (f) *$t \Vvdash_s \forall x A$ if and only if for all numerals n, $tn \Vvdash_s A[n/x]$*

   (g) *$t \Vvdash_s \exists x A$ if and only for some numeral n, $\pi_0 t[s] = n$ and $\pi_1 t \Vvdash_s A[n/x]$*

   (h) *$t \Vvdash_s \forall X A$ if and only if for every type B and $F : \mathbb{N} \to \mathsf{Sat}_B(s)$, $tB \Vvdash_s A[\dot{F}/X]$*

   (i) *$t \Vvdash_s \exists X A$ if and only if $t = \langle B, u \rangle$ and $u \Vvdash_s A[\dot{F}/X]$, for some $F : \mathbb{N} \to \mathsf{Sat}_B(s)$*

   We define $t \Vvdash A$ if and only if for all closed $s : \mathtt{N} \to \mathtt{N}$ of $\mathcal{F}$, $t \Vvdash_s A$.

Given any state s, one may now define a dummy realizer of any formula and thus of the ex-falso axiom.

Proposition 9 (Dummy Realizers). *The following hold:*

1. *Suppose U is a sound non-empty update and s is a state such that $\mathsf{dom}(s) \cap \mathsf{dom}(U) = \emptyset$. Then*

$$\mathsf{c}_U^{|A|} \Vvdash_s A$$

2.

$$\Lambda X \lambda x^{\mathtt{U}} \mathsf{c}_x^X \Vvdash_s \forall X. \bot \to X0$$

Proof. 1. is proved by routine induction on A and 2. is immediate.

We are now able to prove that the following propositions 2 still holds that is, the new notion of saturated set captures the notion of $\Vvdash$.

Proposition 10 (Saturation). *The following hold:*

1. $\{t \mid t \Vvdash_s A\} \in \bot\mathsf{Sat}_{|A|}(s)$

2. *If* $u[s] = v[s]$, *then* $\{t \mid t \Vdash_s B[u/x]\} = \{t \mid t \Vdash_s B[v/x]\}$

PROOF. By straightforward induction on A.

Proposition 11 (Comprehension). *Let* $B(x)$ *be a formula of* $\mathcal{L}^{+}_{Class}$ *in the only free natural number variable* x. *Define* $\mathsf{B} : \mathbb{N} \to \bot\mathsf{Sat}_{|B|}(s)$ *as*

$$\mathsf{B} := n \mapsto \{t \mid t \Vdash_s B(n)\}$$

Then for every t

$$t \Vdash_s A[\dot{\mathsf{B}}/X] \iff t \Vdash_s A[\lambda x B(x)/X]$$

PROOF. As the proof of proposition 2.

Let now $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1 + \forall X. \bot \to X0$ the type system obtained from the one in figure 2 for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$ by adding the rule:

$$\frac{}{\Gamma \vdash \Lambda X \lambda x^{\mathtt{U}}\, \mathsf{c}_x^X : \forall X. \bot \to X0}$$

We finally have the main theorem:

Theorem 15 (Adequacy). *If* A *is a closed formula such that* $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1 + \forall X. \bot \to X0 \vdash t : A$, *then* $t \Vdash A$.

PROOF. The proof is exactly the same of theorem 14.

[1] F. Aschieri, *Learning, Realizability and Games in Classical Arithmetic*, PhD Thesis, 2011. http://arxiv.org/abs/1012.4992
[2] F. Aschieri, *Transfinite Update Procedures for Predicative Systems of Analysis*, Proceedings of Computer Science Logic, 2011.
[3] F. Aschieri, *A Constructive Analysis of Learning in Peano Arithmetic*, Annals of Pure and Applied Logic, 2011, doi: 10.1016/j.apal.2011.12.004.
[4] F. Aschieri, S. Berardi, *Interactive Learning-Based Realizability for Heyting Arithmetic with* EM_1, Logical Methods in Computer Science, 2010.
[5] F. Aschieri, S. Berardi, *A New Use of Friedman's Translation: Interactive Realizability*, draft, 2011.
[6] J. Avigad, *Update Procedures and 1-Consistency of Arithmetic*, Mathematical Logic Quarterly, volume 48, 2002.
[7] J. Avigad, *Eliminating Definitions and Skolem functions in First-Order Logic*, ACM Transactions on Computational Logic, (4), 2003.
[8] J. Avigad, *A realizability Interpretation for Classical Arithmetic*, in Buss, Hjek, and Pudlk eds., Logic Colloquium '98, Lecture Notes in Logic 13, AK Peters, 57-90, 2000.
[9] S. Berardi, M. Bezem, T. Coquand, *On the Computational Content of the Axiom of Choice*, Journal of Symbolic Logic, vol. 63, n. 2, 1998.
[10] S. Berardi and U. de' Liguoro, *A Calculus of Realizers for EM1 Arithmetic*, Computer Science Logic, Lecture Notes in Computer Science, vol. 5213, 2008.
[11] U. Berger, *Strong Normalization for Applied Lambda Calculi*, Logical Methods in Computer Science, 2005.
[12] P. Cohen, *Set Theory and the Continuum Hypothesis*, Dover ed., 1966.
[13] T. Coquand, *A Semantic of Evidence for Classical Arithmetic*, Journal of Symbolic Logic 60, pag 325-337,1995.
[14] H. Friedman, *Classically and Intuitionistically Provable Recursive Functions*, Lecture Notes in Mathematics, 1978, Volume 669/1978, 21-27.
[15] K. Gödel *Uber eine bisher noch nicht ben utzte Erweiterung des niten Standpunktes*, Dialectica 12, 280287 (reproduced with English translation, in [Gödel 1990], 240251.
[16] Nicolas D. Goodman, *Relativized Realizability in Intuitionistic Arithmetic of All Finite Types*,Journal of Symbolic Logic 43, 1, pag. 23-44 (1978).
[17] W. Felscher, *Dialogues as a Foundation for Intuitionistic Logic*, Handbook of Philosophical Logic, 2nd Edition, Volume 5, pages 115-145 2002 Kluwer Academic.
[18] G. Gentzen, *Die Widerspruchsfreiheit der reinen Zahlentheorie*. Mathematische Annalen, 112:493-565, 1935. English translation: The consistency of elementary number theory, in Szabo [465], pages 132-200.
[19] G. Gentzen, *Untersuchungen fiber das logische Schliessen. Mathematische Zeitschrift, 39:176-210, 405-431, 1935. English translation: Investigations into logical deduction, in Szabo [465], pages 68-131*
[20] J.-Y. Girard, *Proofs and Types*, Cambridge University Press (1989).
[21] U. Kohlenbach, *Applied Proof Theory*, Springer-Verlag, Berlin, Heidelberg, 2008.
[22] G. Kreisel, *On the interpretation of non-finitist proofs, part I.* J. Symbolic Logic 16, pp.241-267, 1951.

[23] G. Kreisel, *Interpretation of analysis by means of constructive functionals of finite types*, Heyting, A. (ed.), Constructivity in Mathematics, pp. 101128. North- Holland, Amsterdam (1959).
[24] S. Kripke, *Semantical Analysis of Intuitionistic Logic I*, In Formal Systems and Recursive Functions, edited by M. Dummett and J. N. Crossley. Amsterdam: North-Holland Publishing Co. 1965.
[25] J-L. Krivine, *Realizability in Classical Logic*, in Interactive models of computation and program behaviour. Panoramas et synthses, Socit Mathmatique de France, 27, p. 197-229 (2009).
[26] G. Mints, S. Tupailo, W. Bucholz, *Epsilon Substitution Method for Elementary Analysis*, Archive for Mathematical Logic, volume 35, 1996
[27] A. Miquel, *Relating classical realizability and negative translation for existential witness extraction.* In Typed Lambda Calculi and Applications (TLCA 2009), pp. 188-202, 2009.
[28] E. Moggi, *Notions of Computations and Monads*, Journal of Logic and Computation, 93(1),1991.
[29] P. Oliva, T. Striecher, *On Krivine Realizability Interpretation of Second-Order Classical Arithmetic*, Fundamenta Informaticae, 2008.
[30] H. Schwichtenberg, A. Troelstra, *Basic Proof Theory*, Cambridge University Press, 1996
[31] M. H. Sorensen, P. Urzyczyn, *Lectures on the Curry-Howard isomorphism*, Studies in Logic and the Foundations of Mathematics, vol. 149, Elsevier, 2006.
[32] C. Spector, *Provably Recursive Functionals of Analysis: a Consistency Proof of Analysis by an Extension of Principles in Current Intuitionistic Mathematics*, Dekker (ed.), Recursive Function Theory: Proceedings of Symposia in Pure Mathematics, vol. 5. AMS, Providence, 1962
[33] A. Troelstra, D. van Dalen, *Constructivism in Mathematics, vol. I*, North-Holland, 1988.
[34] A. Troelstra, *Metamathematical Investigations of Intuitionistic Arithmetic and Analysis*, Lecture Notes in Mathematics, Springer-Verlag, Berlin-Heidelber-NewYork, 1973.