Under consideration for publication in Math. Struct. in Comp. Science

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

Federico Aschieri

Laboratoire de l'Informatique du Parallélisme (UMR 5668), équipe Plume
École Normale Supérieure de Lyon – Université de Lyon
46 Allée d'Italie, 69007 Lyon, France
federico.aschieri@ens-lyon.fr

Received 3 March 2012; Revised 23 May 2013

We introduce a 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 and can be seen as programs that learn by interacting with the environment. We show that the realizers of any Π^0_2-formula represent terminating learning processes whose outcomes are numerical witnesses for the existential quantifier of the formula.

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 (Gödel 1990), (Spector 1962) or intuitionistic realizability interpretations combined with Friedman's translation (Friedman 1978). Among the direct interpretations, there are classical realizabilities, Coquand game semantics (Coquand 1995), cut-elimination and normalization of classical proofs (under Curry-Howard correspondence or not), and the epsilon substitution method (Mints et al. 1996) (the Kreisel no-counterexample interpretation (Kreisel 1951) can be obtained as 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

[†] This work was supported by the LABEX MILYON (ANR-10-LABX-0070) of Université de Lyon, within the program "Investissements d'Avenir" (ANR-11-IDEX-0007) operated by the French National Research Agency (ANR)

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. 1998) 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 (Krivine 2009) 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 the program 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 semantically meaningful way. Avigad's presentation of the epsilon substitution method (Avigad 2002) 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 interpreted 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 (according to some current approximation of the truth values of $A(n,m)$ and $A(n,f(n))$) 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 interprets classical proofs; in other words: more efficient programs.

A significant step towards this goal has been taken in (Aschieri and Berardi 2010; Aschieri and Berardi 2012), where it has been introduced a learning-based classical realizability for first-order Heyting Arithmetic HA (also in its finite type version HA^ω) with the excluded middle EM_1 on Σ^0_1-formulas and Skolem axioms SK_1 over quantifier-free for-

mulas. It is a realizability based on states describing 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[†] and it is worth to be studied separately; in game semantics, one represents it as *simple backtracking* (see (Coquand 1991), pag. 90 and (Aschieri 2013)). In more general settings, 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 2002), for an ordinal analysis of this kind of learning, and (Aschieri 2011a; Aschieri 2012), for a type theoretic analysis). In the case of predicative fragments of Analysis, learning is transfinitely nested (see (Aschieri 2011b)).

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 program is very natural to write, it obeys the laws of intuitionistic Heyting semantics (see e.g., (Troelstra and van Dalen 1988)) 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 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 (Kreisel 1959) extended with learning (Aschieri and Berardi 2012). 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 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 2008), but in a Church style) with learning.

We also introduce a technique for witness extraction from proofs of Π^0_2-formulas. That is, given a realizer $t \Vdash \forall x^{\mathbb{N}} \exists y^{\mathbb{N}} P(x,y)$, 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 $P(n,m)$ holds. In particular, the program uses t as a state-extending operator; we shall

[†] 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 directly to a stronger theory, there is a good chance that one is losing some constructive information about the theorems of the former theory.

prove that it is enough to extend the state a finite number of times to get a sufficient amount of information to compute a witness.

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 technique of witness extraction.

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

In this section we introduce the typed lambda calculi that we shall use to define interactive realizability: system $\mathcal{F}$ and $\mathcal{F}_{\mathrm{Class}}$. $\mathcal{F}$ is a completely standard extension of Girard's system F (see (Girard 1989)) 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 (as one may find it in the exposition of (Girard 1989)) with polymorphism. $\mathcal{F}_{\mathrm{Class}}$ 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 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 $\mathbb{N}$.

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

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

4 $\mathbb{U}$ is the set of all updates.

5 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) contained in an update as the code of a possible witness m for a Σ^0_1-formula $\exists y.\mathsf{P}_a(n, y)$. 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 a 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 update whenever $U_1 \cup U_2$ is non-empty.

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

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

In fact, the whole realizability semantics is a Monad (Berardi and de' Liguoro 2008). In (Berardi and de' Liguoro 2008), 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 $\mathsf{S}(\mathsf{S}(\ldots 0))$. For notational simplicity, we assume in the following that updates are made of triples of numerals. 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 `True` if $(a, n, m) \in U$ for some numeral m (that is, if the pair (a, n) is in the domain of the partial function U); it returns `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 numeral m (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) 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 system $\mathcal{F}$ 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 $t_1, \ldots, t_n, t \in \mathcal{F}$ 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 $\mathcal{F}$ (a tautology if $n = 0$) if all boolean assignments making $t_1, \ldots, t_n$ equal to $\mathtt{True}$ in $\mathcal{F}$ also make t equal to $\mathtt{True}$ in $\mathcal{F}$. Some terms of Gödel's system T will be used to represent predicates.

Definition 2 (Predicates of system T).

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 $\mathsf{P}_0, \mathsf{P}_1, \mathsf{P}_2, \ldots$ is an arbitrary enumeration of all binary predicates of 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 objects 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 3 (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

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

$$\frac{u : \forall X A}{uB : A[B/X]} \qquad \frac{u : A}{\Lambda X u : \forall X A} \text{ with } X \text{ non free in the types of the free variables of } u$$

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

Reduction Rules All the usual reduction rules for system F (see (Girard 1989)) 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:

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

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

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

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

Fig. 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

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

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 4 (Functional set of rules). Let C be any set of constants, each one of some type $A_1 \to \ldots \to A_n \to A$, for some $A_1, \ldots, A_n, A \in \{\mathtt{Bool}, \mathtt{N}, \mathtt{U}\}$. 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. 4). 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 2005) or just use standard reducibility arguments, as in (Girard 1989). 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 $n : \mathtt{N}$, or a boolean $\mathtt{True}, \mathtt{False} : \mathtt{Bool}$, or an update constant $\overline{U} : \mathtt{U}$, 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 $t = \mathtt{True}, \mathtt{False} : \mathtt{Bool}$ or $t = \overline{U} : \mathtt{U}$ or t is a constant of C we are done. Otherwise either $t = (\mathsf{R}Us_1s_2n)\vec{t}$ or $t = (\mathsf{if}Uba_1a_2)\vec{t}$ or $t = \pi_i(z)\vec{w}$, or $t = \mathsf{is}\,u\,n\,m : \mathtt{N}$, or $t = \mathsf{get}\,u\,n\,m : \mathtt{N}$, or $t = \Cup\, u_1\, u_2 : \mathtt{U}$, 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 $n, m, l : \mathtt{N}$, $b : \mathtt{Bool}$,

$z : A \times B$, $u, u_1, u_2 : \mathtt{U}$, $v_1 : A_1, \ldots v_k : A_k$ 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 $\mathcal{F}_{Class}$*

We now define a classical extension of $\mathcal{F}$, that we call $\mathcal{F}_{\text{Class}}$, with a constant symbol $\Phi : \mathtt{N}^2 \to \mathtt{N}$ 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 5 (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 $\Phi : \mathtt{N}^2 \to \mathtt{N}$ 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 6 (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 $s : \mathtt{S}$; 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 arguments 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 7 (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 U is an update, we denote with $\mathsf{dom}(U)$ the set of pairs of numerals (a, n) such that $(a, n, m) \in U$ and $\mathsf{P}_a nm = \mathtt{True}$; we say that U is *sound* if

for every $(a, n, m) \in U$, we have $\mathsf{P}_a nm = \mathtt{True}$.

2 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 $\oplus t_1 t_2$.

3 For every numeral $a : \mathtt{N}$, 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 $t[s_i]$ 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 correcting 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 8 (Convergence). Assume that $\{s_i\}_{i\in\mathbb{N}}$ is a w.i. sequence of states, and $u \in \mathcal{F}_{\text{Class}}$.

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 of states.

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

Proof. (Classical). For every function S mapping pairs of numerals into numerals, we define some (in general, *non* 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; moreover, $t \neq \Phi$, for $\Phi : \mathtt{N}^2 \to \mathtt{N}$. Then 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 $t[s]$.

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 $(a,n) \in \mathsf{dom}(s_i)$ and $(a,n) \in \mathsf{dom}(s_j)$ imply $s_i(a,n) = s_j(a,n)$, since $\{s_i\}_{i\in\mathbb{N}}$ is weakly increasing. 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 w 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, by the *Claim*, the normal forms 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 notion of realizability based on interactive learning 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 $\mathtt{Bool}$.

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 $\exists X A$, where

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

and

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

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

$$S(n,m) := \big(\mathsf{P}_a nm \wedge \forall x^{\mathtt{N}}. x \leq m \to \neg \mathsf{P}_a nx\big) \vee \big(\forall y^{\mathtt{N}} \neg \mathsf{P}_a ny \wedge m = 0\big)$$

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 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 (Krivine 2009) realizing the formula SK_1 as it is written. This is due to the fact that in order to represent Arithmetic in pure second-order logic, one needs to relativize all the first-order quantifiers to the usual predicate $\mathsf{Nat}(x)$, which defines the concept of natural number. This is not enough, though, because one then must ensure that for every first-order term t in the language it is possible to prove $\mathsf{Nat}(t)$. There is not any problem in the case of a term language with function symbols representing recursive functions; however, if Φ is added, there would be the axiom $\forall x.\mathsf{Nat}(x) \to \mathsf{Nat}(\Phi x)$. But then, by the standard results on storage operators showing that it is always possible to effectively recover the number denoted by t from a realizer of $\mathsf{Nat}(t)$, one would obtain that Φ is computable, if the axiom is realizable.

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

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

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

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

4 As usual, if A and B are formulas of $\mathcal{L}_{\text{Class}}$ 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 $\bot$ the atomic formula $\mathtt{False}$. We shall write natural number variables in lower case characters $x^{\mathtt{N}}, y^{\mathtt{N}}, z^{\mathtt{N}}, \alpha^{\mathtt{N}}$ 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 $\forall x^{\mathtt{N}} A$. If P is an atomic formula of $\mathcal{L}_{\text{Class}}$ in the free variables $x_1, \dots, x_n$ and $t_1, \dots, t_n$ are terms of $\mathcal{L}_{\text{Class}}$, with $P(t_1, \dots, t_n)$ we shall denote the atomic formula $P[t_1/x_1, \dots, t_n/x_n]$.

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

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

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" $A[s] : \mathtt{Bool}, A[s] \in \mathcal{F}$ 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 (Oliva and Streicher 2008), for example, who essentially follow (Krivine 1990)). 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, for quantification over predicates, one may define real-

izability 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 $s : \mathtt{S}$. 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 10 (s-Saturated Sets).

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

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

3 Let $\mathcal{L}^+_{\text{Class}}$ be the language resulting from $\mathcal{L}_{\text{Class}}$ by adding a predicate constant symbol $\mathscr{F}$ for every triple made of a function F, type A and state s such that $F : \mathbb{N} \to \mathsf{Sat}_A(s)$; we denote with $[\![\mathscr{F}]\!]$ the function F, with $|\mathscr{F}|$ the type A and with $\mathsf{st}(\mathscr{F})$ the state s.

Remark 3. We shall not assign boolean values to propositions of the form $\mathscr{F}n$, let alone arbitrary formulas, since we are interested in determining what is a construction of a formula rather than when a formula is true. Realizability will indeed give a meaning to the expression $\mathscr{F}n$ by the set $[\![\mathscr{F}]\!](n)$, which specifies in an extensional way what it means to "construct" $\mathscr{F}n$ in a world/state s. More in general, the meaning of an arbitrary formula in a state s will just be a description of what is a construction of it in the state s, and thus will be determined by all its possible realizers in the state s. The (unconditional) meaning of an arbitrary formula will just be the set of programs that are valid realizers of it in every state s; if this set is not empty, the formula is consider "true" in our framework.

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

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

2 $|Xt| = X$,

3 $|\mathscr{F}t| = |\mathscr{F}|$,

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 $t \Vdash C$ is closed, where $t \in \mathcal{F}_{\text{Class}}$, $C \in \mathcal{L}^+_{\text{Class}}$ is closed and $t : |C|$.

Definition 12 (Interactive Realizability). Assume s is a state, t is a closed term of $\mathcal{F}_{\text{Class}}$, $C \in \mathcal{L}^+_{\text{Class}}$ 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 $t \Vdash_s \mathscr{F} u$ if and only if $\mathsf{st}(\mathscr{F}) = s$ and for some numeral n, $u[s] = n$ and $t \in [\![\mathscr{F}]\!](n)$

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 $t \Vdash_s \forall x A$ 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 every $\mathscr{F}$, if $|\mathscr{F}| : B$ and $\mathsf{st}(\mathscr{F}) = s$, then $tB \Vdash_s A[\mathscr{F}/X]$

9 $t \Vdash_s \exists X A$ if and only if $t = \langle B, u \rangle$ and $u \Vdash_s A[\mathscr{F}/X]$, for some $|\mathscr{F}| : B$ with $\mathsf{st}(\mathscr{F}) = s$

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 $\exists x A$ uses s to compute $\pi_0 u[s] = n$ and to predict that n is a witness for $\exists x A$ (i.e. that $A[n/x]$ is realizable).
- A realizer of $\exists X A$ predicts the existence of a witness $[\![\mathscr{F}]\!] : \mathbb{N} \to \mathsf{Sat}_B(s)$ for $\exists X A$ (i.e. that $A[\mathscr{F}/X]$ is realizable).

— *Tests.*

- A realizer t of a universal formula $\forall x^{\mathbb{N}} A$ or $\forall X A$ takes a natural number n or constant $\mathscr{F}$ as a challenge coming from the environment to provide a construction of $A[n/x]$ or $A[\mathscr{F}/X]$, 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 or $\mathscr{F} u$ comes after a series of predictions and challenges that have been provided to test the construction of a complex formula; now it is performed a final test. In the case of $\mathscr{F} u$, one just verifies that the realizer satisfies the notion of construction embodied by the function $[\![\mathscr{F}]\!]$. In the case of a formula Q of $\mathcal{T}_{\text{Class}}$, the realizer computes the formula Q in the state s as an experiment. Since the 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 $t : \mathtt{Bool}\times|A|+|B|$, then $\pi_0 t : \mathtt{Bool}$ 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 $Q[s] = \mathtt{True}$ 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 $\mathcal{L}^+_{\text{Class}}$ in the only free natural number variable x. Let $\mathscr{B}$ be such that $|\mathscr{B}| = |B|$, $\mathsf{st}(\mathscr{B}) = s$ and

$$[\![\mathscr{B}]\!] = n \mapsto \{t \mid t \Vdash_s B(n)\}$$

Then for every t

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

Proof. By induction on A.

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

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

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

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

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

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

and so

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

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, as we explain in a moment, 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 2012). □

E_a works as follows. It uses the oracle X_a (definition 7) 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 7 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 (Sorensen and Urzyczyn 2006), 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

$$\frac{\Gamma \vdash P_1 \ \Gamma \vdash P_2 \ \cdots \ \Gamma \vdash P_n}{\Gamma \vdash P}$$

where $P_1, \ldots, P_n, P$ 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, $P_1\sigma = \ldots = P_n\sigma = \mathtt{True}$ implies $P\sigma = \mathtt{True}$. Let now $\mathtt{eq} : \mathtt{N}^2 \to \mathtt{Bool}$ a term of Gödel's system T representing equality between natural numbers. Among the Post rules, we have the Peano axioms

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

and axioms of equality

$$\frac{}{\Gamma \vdash \mathtt{eq}\ x\,x} \qquad \frac{\Gamma \vdash \mathtt{eq}\,x\,y \ \ \Gamma \vdash \mathtt{eq}\,y\,z}{\Gamma \vdash \mathtt{eq}\ x\,z} \qquad \frac{\Gamma \vdash P(x) \ \ \Gamma \vdash \mathtt{eq}\,x\,y}{\Gamma \vdash P(y)}$$

and for every P_1, P_2 such that $P_1 = P_2$ is an equation of Gödel's system T (equivalently, P_1, P_2 have the same normal form in T), we have the rule

$$\frac{\Gamma \vdash P_1}{\Gamma \vdash P_2}$$

We also have a Post rule

$$\frac{\Gamma \vdash P_1 \ \Gamma \vdash P_2 \ \cdots \ \Gamma \vdash P_n}{\Gamma \vdash P}$$

for every classical propositional tautology $P_1 \to \ldots \to P_n \to P$, where for $i = 1, \ldots, n$, P_i, P 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

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

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

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

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 we may derive $\mathsf{HAS} \vdash P \vee \neg_{\mathtt{Bool}} P$ 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 $P_1, \ldots, P_n$ spell a true conclusion P.

Remark 4. We observe that the full *ex-falso-quodlibet* axiom is not in our system, since there are no rules neither axioms to derive $\bot \to Xt$. Nevertheless, we have the ex-falso-quodlibet restricted to first-order formulas, thus we cannot say to be in minimal Arithmetic, where usually $\bot$ is replaced by $0 = \mathsf{S}(0)$. We leave the axiom out of the system because 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 is not realizable according to our semantics, since there is no closed term t of type $|\forall X. \bot \to X0| = \forall X. \mathtt{U} \to X$ in system $\mathcal{F}$ (otherwise $\Lambda X. tX\varnothing$ would have type $\forall X X$, which is not possible by normalization and the argument used for proving lemma 3). 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 $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$.

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 assume $|\mathscr{F}_1| = B_1, \mathsf{st}(\mathscr{F}_1) = s \ldots, |\mathscr{F}_m| = B_m, \mathsf{st}(\mathscr{F}_m) = s$. For every formula C, let $\overline{C} := C[n_1/\alpha_1 \cdots n_k/\alpha_k \ \mathscr{F}_1/X_1 \cdots \mathscr{F}_m/X_m]$. 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

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

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 $\Gamma \vdash w : A$:

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

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

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

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

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

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

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

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

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

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

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

Post Rules $$\frac{\Gamma \vdash u_1 : P_1 \;\; \Gamma \vdash u_2 : P_2 \;\; \cdots \;\; \Gamma \vdash u_n : P_n}{\Gamma \vdash u_1 \Cup u_2 \Cup \cdots \Cup u_n : P}$$

where $n > 0$ and $P_1, P_2, \ldots, P_n, P$ 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 $$\frac{}{\Gamma \vdash \varnothing : P}$$

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

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

SK1

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

Fig. 2. Term Assignement Rules for $\mathsf{HAS} + \mathsf{EM}_1 + \mathsf{SK}_1$

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.

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 3 of $\iota_{0,|\overline{B}|,|\overline{C}|}$ and by (1)

$$\begin{aligned}\overline{w}[s] &= \iota_{0,|\overline{B}|,|\overline{C}|}(v)|\overline{D}|\overline{w}_1[s]\overline{w}_2[s]\\ &= (\Lambda X \lambda f^{|\overline{B}|\to X} \lambda g^{|\overline{C}|\to X}\ fv)|\overline{D}|\overline{w}_1[s]\overline{w}_2[s]\\ &= \overline{w}_1[s]v = \overline{w}_1 v[s]\end{aligned}$$

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 $w = ut$, $A = B[t/\alpha]$ and $\Gamma \vdash u : \forall\alpha B$. 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 $\overline{u}n \Vdash_s \overline{B}[n/\alpha]$. Since $\overline{u}\overline{t}[s] = \overline{u}n[s]$, by saturation (proposition 1), we conclude that $\overline{u}\overline{t} \Vdash_s \overline{B}[\overline{t}/\alpha]$.

9 If it is the (first order) $\forall I$ rule, then $w = \lambda\alpha^{\mathtt{N}}u$, $A = \forall\alpha B$ 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 $\overline{w}n = \overline{u}[n/\alpha] \Vdash_s \overline{B}[n/\alpha]$, which amounts to show that the induction hypothesis can be applied to u. For this purpose, it is enough to observe that for $i = 1, \ldots, n$

$$t_i \Vdash_s \overline{A_i} = \overline{A_i}[n/\alpha]$$

10 If it is the (first order) $\exists E$ rule, then

$$w = t(\pi_0 u)(\pi_1 u)$$

$\Gamma \vdash t : \forall\alpha.\, B \to A$ and $\Gamma \vdash u : \exists\alpha\, B$. Assume $n = \pi_0\overline{u}[s]$, for some numeral n. By induction hypothesis $\overline{t} \Vdash_s \forall\alpha.\, \overline{B} \to \overline{A}$ and $\overline{u} \Vdash_s \exists\alpha\, \overline{B}$. Therefore, $\pi_1\overline{u} \Vdash_s \overline{B}[n/\alpha]$ and

$$\overline{t}n(\pi_1\overline{u}) \Vdash_s \overline{A}[n/\alpha] = \overline{A}$$

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 $w = \langle t, u\rangle$, $A = \exists\alpha B$, $\Gamma \vdash u : B[t/\alpha]$. So, $\overline{w} = \langle\overline{t}, \overline{u}\rangle$; and

$$\pi_1\overline{w} = \overline{u} \Vdash_s \overline{B}[\overline{t}/\alpha]$$

by induction hypothesis. Assume $\pi_0\overline{w}[s] = \overline{t}[s] = n$, with n numeral. We obtain $\pi_1\overline{w} \Vdash_s \overline{B}[n/\alpha]$ 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, $\overline{w} = \overline{u}|\overline{C}|$. Let $\mathscr{C}$ be such that $|\mathscr{C}| = |\overline{C}|$, $\mathsf{st}(\mathscr{C}) = s$ and

$$[\![\mathscr{C}]\!] := n \mapsto \{t \mid t \Vdash_s \overline{C}(n)\}$$

($\mathscr{C}$ is well defined by proposition 1). By inductive hypothesis $\overline{u} \Vdash_s \forall X\overline{B}$ and so $\overline{u}|\overline{C}| \Vdash_s \overline{B}[\mathscr{C}/X]$. 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 $X \neq X_1, \ldots, X_m$. Suppose $|\mathscr{F}| = C$ and $\mathsf{st}(\mathscr{F}) = s$; we have to prove that $\overline{w}C = \overline{u}[C/X] \Vdash_s \overline{B}[\mathscr{F}/X]$, which amounts to show that the induction hypothesis can be applied to u. For this purpose, it is enough to observe that for $i = 1, \ldots, n$

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

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 $v \Vdash_s \overline{B}[\mathscr{F}/X]$, where $|\mathscr{F}| = C$ and $\mathsf{st}(\mathscr{F}) = s$. By induction hypothesis on t, $\overline{t} \Vdash_s \forall X.\, \overline{B} \to \overline{A}$ and hence

$$\overline{t}Cv \Vdash_s \overline{A}[\mathscr{F}/X] = \overline{A}$$

Moreover

$$\overline{w} = \langle C, v\rangle |\overline{A}|\overline{t} \stackrel{\text{def. } 3}{=} (\Lambda Y \lambda x^{\forall X.\, \overline{B} \to Y}\, xCv)|\overline{A}|\overline{t} = \overline{t}Cv$$

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

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

Let $\mathscr{F}$ be such that $|\mathscr{F}| = |\overline{C}|$, $\mathsf{st}(\mathscr{F}) = s$ and

$$[\![\mathscr{F}]\!] := n \mapsto \{t \mid t \Vdash_s \overline{C}(n)\}$$

($\mathscr{F}$ is well defined by proposition 1). By proposition 2

$$\overline{u} \Vdash_s \overline{B}[\mathscr{F}/X]$$

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 $\Gamma \vdash u_i : P_i$. 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, $\overline{u}_i \Vdash_s \overline{P}_i$, for $i = 1, \ldots, n$. Therefore, $\mathsf{dom}(U_i) \cap \mathsf{dom}(s) = \emptyset$, and thus also $\mathsf{dom}(U) \cap \mathsf{dom}(s) = \emptyset$. Suppose now that $\overline{U} = \varnothing$; then we have to prove that $\overline{P}[s] = \mathtt{True}$. It suffices to prove that

$$\overline{P}_1[s] = \overline{P}_2[s] = \cdots = \overline{P}_n[s] = \mathtt{True}$$

Indeed, we have $\overline{U_1} = \cdots = \overline{U_n} = \varnothing$ and thus $\overline{P}_1[s] = \cdots = \overline{P}_n[s] = \mathtt{True}$, by $\overline{u}_i \Vdash_s \overline{P}_i$, 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 7): this follows from $\mathsf{P}_a nm = \mathtt{True}$. Then we have to verify that $\mathsf{dom}(s) \cap \mathsf{dom}(U) = \emptyset$: indeed, $\mathsf{dom}(U) = \{(a,n)\}$, and by definition 7, $\mathsf{P}_a n s_a(n) = \mathtt{False}$ 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 $t \Vdash \forall x^{\mathbb{N}} \exists y^{\mathbb{N}} Pxy$, where $Pxy : \mathtt{Bool}$ is a term of Gödel's T, 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 $\exists y^{\mathbb{N}} Pny$ (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:

$$t \Vdash \forall x^{\mathbb{N}} \exists y^{\mathbb{N}} Pxy$$

$$\Longrightarrow$$

$$t \Vdash_s \forall x^{\mathbb{N}} \exists y^{\mathbb{N}} Pxy$$

$$\Longrightarrow$$

$$tn \Vdash_s \exists y^{\mathbb{N}} Pny$$

$$\Longrightarrow$$

$$\pi_0(tn)[s] = m \wedge \pi_1(tn) \Vdash_s Pnm$$

$$\Longrightarrow$$

$$\pi_1(tn)[s] = \varnothing \implies Pnm = \mathtt{True}$$

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. In the rest of the paper, we will show how to compute such a zero. We propose the *Iterative Method*, which is very simple, easy to understand and provides an algorithm that can be directly written in pure lambda calculus. However, the proof of correctness uses the Axiom of Choice. The problems of finding an algorithm directly formalizable in system $\mathcal{F}$ and of proving constructively its correctness will be treated in a forthcoming paper.

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 $Pxy : \mathtt{Bool}$ be a term of Gödel's T and suppose $t \Vdash \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$. Let n be any numeral, define $u := \pi_1(tn)$ and let s be any state. Define, by induction on n, a sequence $\{s_n\}_{n\in\mathbb{N}}$ of states as follows:

$$s_0 := s$$

$$s_{n+1} := s_n \oplus u[s_n] \overset{\text{def } 7}{=} \lambda x^{\mathtt{N}} \lambda y^{\mathtt{N}}\ \mathsf{if}\ (\mathsf{is}\, u[s_n]\, x\, y)\ \mathsf{then}\ (\mathsf{get}\, u[s_n]\, x\, y)\ \mathsf{else}\ s_n(x,y)$$

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 consider the 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 $l = \pi_0(tn)[s_i]$, 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 7 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 $Pxy : \mathtt{Bool}$ term 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 $t \Vdash \forall x^{\mathtt{N}} \exists y^{\mathtt{N}} Pxy$, where $Pxy : \mathtt{Bool}$ is a term 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 to write f quite directly in a purely functional language (differently from what happens with Krivine classical realizability (Krivine 2009; Miquel 2009)).

One may then wonder how it is implemented backtracking in our extracted programs: control operators have precisely that function in (Krivine 2009). 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 denoted 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 rencountered, but with a waste of resources. Again, this issue will be addressed in a forthcoming paper.

References

F. Aschieri, S. Berardi (2010), *Interactive Learning-Based Realizability for Heyting Arithmetic with EM1*, Logical Methods in Computer Science, 2010.

F. Aschieri (2011a), *Learning, Realizability and Games in Classical Arithmetic*, PhD Thesis, 2011. http://arxiv.org/abs/1012.4992

F. Aschieri (2011b), *Transfinite Update Procedures for Predicative Systems of Analysis*, Proceedings of Computer Science Logic, 2011.

F. Aschieri (2012), *A Constructive Analysis of Learning in Peano Arithmetic*, Annals of Pure and Applied Logic, 2011, vol. 162, n. 11, 2012.

F. Aschieri, S. Berardi (2012), *A New Use of Friedman's Translation: Interactive Realizability*, in: Logic, Construction, Computation, Berger et al. eds, Ontos-Verlag Series in Mathematical Logic, 2012.

F. Aschieri (2013), *Learning Based Realizability for HA + EM1 and 1-Backtracking Games: Soundness and Completeness*, Annals of Pure and Applied Logic, vol. 164, n. 6, 2013.

J. Avigad (2000), *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.

J. Avigad (2002), *Update Procedures and 1-Consistency of Arithmetic*, Mathematical Logic Quarterly, volume 48, 2002.

J. Avigad (2003), *Eliminating Definitions and Skolem functions in First-Order Logic*, ACM Transactions on Computational Logic, vol. 4, 2003.

S. Berardi, M. Bezem, T. Coquand (1998), *On the Computational Content of the Axiom of Choice*, Journal of Symbolic Logic, vol. 63, n. 2, 1998.

S. Berardi and U. de' Liguoro (2008), *A Calculus of Realizers for EM1 Arithmetic*, Proceedings of Computer Science Logic 2008, Lecture Notes in Computer Science, vol. 5213, 2008.

U. Berger (2005), *Strong Normalization for Applied Lambda Calculi*, Logical Methods in Computer Science, 2005.

T. Coquand (1991), *A Semantic of Evidence for Classical Arithmetic*, Proceedings of the Second Workshop on Logical Frameworks, 1991.

T. Coquand (1995), *A Semantic of Evidence for Classical Arithmetic*, Journal of Symbolic Logic, vol. 60, pp. 325-337,1995.

H. Friedman (1978), *Classically and Intuitionistically Provable Recursive Functions*, Lecture Notes in Mathematics, 1978, vol. 669, 21-27.

K. Gödel (1990), *Uber eine bisher noch nicht benutzte Erweiterung des finiten Standpunktes*, Dialectica 12, pp. 280-287 1958.

G. Gentzen (1935a), *Die Widerspruchsfreiheit der reinen Zahlentheorie.* Mathematische Annalen, 112:493-565, 1935.

G. Gentzen (1935b), *Untersuchungen fiber das logische Schliessen.* Mathematische Zeitschrift, 39:176-210, 405-431, 1935.

J.-Y. Girard (1989), *Proofs and Types*, Cambridge University Press (1989).

U. Kohlenbach (2008), *Applied Proof Theory*, Springer-Verlag, Berlin, Heidelberg, 2008.

G. Kreisel (1951), *On the interpretation of non-finitist proofs, part I.* Journal of Symbolic Logic, vol. 16, pp. 241-267, 1951.

G. Kreisel (1959), *Interpretation of analysis by means of constructive functionals of finite types*, in: Heyting, A. (ed.), Constructivity in Mathematics, pp. 101-128, North-Holland, Amsterdam, 1959.

J.-L. Krivine (1990) *Lambda-calcul, types et modèles*, Studies in Logic and Foundations of Mathematics, pp. 1–176. Masson, Paris, 1990.

J-L. Krivine (2009), *Realizability in Classical Logic*, in: Interactive models of computation and program behaviour. Panoramas et synthèses, Sociètè Mathématique de France, 27, pp. 197-229, 2009.

G. Mints, S. Tupailo, W. Bucholz (1996), *Epsilon Substitution Method for Elementary Analysis*, Archive for Mathematical Logic, vol. 35, 1996.

A. Miquel (2009), *Relating classical realizability and negative translation for existential witness extraction.* In Typed Lambda Calculi and Applications (TLCA 2009), pp. 188-202, 2009.

P. Oliva, T. Striecher (2008), *On Krivine Realizability Interpretation of Second-Order Classical Arithmetic*, Fundamenta Informaticae, 2008.

H. Schwichtenberg, A. Troelstra (1996), *Basic Proof Theory*, Cambridge University Press, 1996

M. H. Sorensen, P. Urzyczyn (2006), *Lectures on the Curry-Howard isomorphism*, Studies in Logic and the Foundations of Mathematics, vol. 149, Elsevier, 2006.

C. Spector (1962), *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

A. Troelstra (1973), *Metamathematical Investigations of Intuitionistic Arithmetic and Analysis*, Lecture Notes in Mathematics, Springer-Verlag, Berlin-Heidelber-NewYork, 1973.

A. Troelstra, D. van Dalen (1988), *Constructivism in Mathematics, vol. I*, North-Holland, 1988.