Eliminating Skolem Functions in Peano Arithmetic with Interactive Realizability

Federico Aschieri¹, Margherita Zorzi²

¹Laboratoire PPS, équipe PLR2, Université Paris 7, INRIA & CNRS
²Laboratoire d’Informatique de Paris-Nord UMR CNRS 7030 Institut Galilée – Université Paris-Nord

Abstract. We present a new syntactical proof that first-order Peano Arithmetic with Skolem axioms is conservative over Peano Arithmetic alone for arithmetical formulas. This result – which shows that the Excluded Middle principle can be used to eliminate Skolem functions – has been previously proved by other techniques, among them the epsilon substitution method and forcing. In our proof, we employ Interactive Realizability, a computational semantics for Peano Arithmetic which extends Kreisel’s modified realizability to the classical case.

1 Introduction

For a long time it has been known that intuitionistic realizability can be used as a flexible tool for obtaining a wealth of unprovability, conservativity and proof-theoretic results [23,25]. As title of example, with Kreisel’s modified realizability [17], one can show the unprovability of Markov Principle in Heyting Arithmetic in all finite types (HAω) and the conservativity of HAω with the Axiom of Choice (AC) over HAω for negative formulas. In both cases, one starts by showing that any formula provable in one of those systems can be shown to be realizable in HAω. In the first case, one proves that the realizability of Markov Principle implies the solvability of the Halting Problem, and concludes that Markov Principle is unprovable in HAω. In the second, one exploits the fact that the assertion “t realizes A” is exactly the formula A when A is negative and concludes that HAω proves A.

The situation in classical logic has been very different: for a long time it did not exist any realizability notion suitable to interpret directly classical proofs, let alone proving independence or conservation results. However, recently several classical realizability interpretations have been put forward. Among them: Krivine’s classical realizability [18], which has been shown in [19] to yield striking unprovability results in Zermelo-Fraenkel set theory, and Interactive realizability [1,4,6,7], which has been shown in [3,6] to provide conservation results for Π02-formulas.

Being a tool for extracting programs from proofs, it is however quite natural that Interactive realizability is capable of producing Π02-conservativity results. The aim of this paper is to prove that Interactive realizability can as well be used to prove other conservativity results. In particular, let us consider first-order classical Peano Arithmetic PA, which is HA + EM, where EM is the excluded middle over arithmetical formulas. Then we give a new syntactic proof that PA with the Skolem axiom scheme SK is conservative over PA for arithmetical formulas – a result first syntactically proven by Hilbert and Bernays [19] by means of the epsilon substitution method. The result is particularly interesting since it implies that classical choice principles can be eliminated by using the excluded middle alone. The structure of our proof resembles the pattern of the intuitionistic realizability conservation proofs we have sketched above and allows to obtain a stronger result. Namely, we shall show that if an arithmetical formula A is provable in HAω + EM + SK, then the assertion “t realizes A” is provable in HAω alone. Afterwards, we shall show the provability in

* Supported by ANR COMPLICE project (Implicit Computational Complexity, Concurrency and Extraction), ref.: ANR-08-BLANC-0211-01.
HA^ω + EM of the assertion “(t realizes A) implies A” and thus conclude that HA^ω + EM proves A. Since this latter system is conservative over PA for arithmetical formulas, we obtain the result.

In our opinion, there are at least two reasons our proof technique is interesting. As remarked by Avigad[10], the methods based on the epsilon-method, Herbrand’s Theorem or cut-elimination lead to an exponential increase in the size of the proof, when passing from a proof in HA^ω + EM + SK to a corresponding proof in HA + EM; instead, we conjecture that our transformation is polynomial. To the best of our knowledge, there is only another method that does equally well, which is Avigad’s [10]. The technique of Avigad is related to ours since it uses the method of forcing, in which the conditions are finite approximations of the Skolem functions used in the proof. With forcing one avoids speaking about infinite non-computable objects (i.e. the Skolem functions) and can approximate the original proof. Avigad’s method is very simple and elegant when there is only one Skolem function to eliminate, but it becomes more complicated and difficult to handle when dealing with several Skolem functions. In fact, a nesting of the notion of forcing together with a technical result about elimination of definitions become necessary and the method loses some intuitive appeal. Instead, the use of Interactive realizability allows to deal with all the Skolem functions at the same time, and we conjecture that the resulting proofs are much shorter than the ones obtained by forcing. Moreover, the notion of forcing as an approximation of model-theoretic truth is harder to come up with, and it is much more natural to talk about states and approximations when dealing with programs.

Secondly, the theory of Interactive realizability offers a uniform explanation of a number of different phenomena. Rather than proving each particular meta-theoretic result about classical Arithmetic with an ad-hoc technique, one employs a single methodology. For example, one may prove conservativity of PA over HA for $I^2_2$-formulas by a negative translation followed by Friedman’s translation [13]; one may extract from proofs terms of Gödel’s System T by realizability or functional interpretations [14]; one may prove the result about the elimination of Skolem functions with forcing; one may extract from proofs strategies in backtracking Tarski games by analyzing sequent calculus proofs [12]; one may obtain a simple ordinal analysis of PA + SK by using update procedures [9]. Instead, with the theory of Interactive realizability one obtains all the results above as a consequence of a single concept (see [3,5,7]).

Plan of the paper In Section 2 we review the term calculus $T_{Class}$ in which Interactive realizers are written, namely an extension of Gödel’s system T plus Skolem function symbols for a countable collection of Skolem functions. In Section 3 we recall Interactive realizability, as described in [7], a computational semantics for HA^ω + EM + SK, an arithmetical system with functional variables which includes first-order classical Peano Arithmetic and Skolem axioms. In Section 4 we use Interactive realizability to prove the conservativity of HA^ω + EM + SK over HA^ω + EM for arithmetical formulas. In Section 5 we explain in more detail how to formalize the proofs of Section 4 in HA^ω + EM and HA + EM.

2 The Term Calculus $T_{Class}$

In this section we follow [7] and recall the typed lambda calculi $T$ and $T_{Class}$ in which interactive realizers are written. $T$ is an extension of Gödel’s system T (as presented in Girard [15]) with some syntactic sugar. The basic objects of $T$ are numerals, booleans, and its basic computational constructs are primitive recursion at all types, if-then-else, pairs, as in Gödel’s T. $T$ also includes as basic objects finite partial functions over $\mathbb{N}$ and simple primitive recursive operations over them. $T_{Class}$ is obtained from $T$ by adding on top of it a collection of Skolem function symbols $\Phi_0, \Phi_1, \Phi_2, \ldots$, of type $\mathbb{N} \rightarrow \mathbb{N}$, one for each arithmetical formula. The symbols are inert from the computational point of view and realizers are always computed with respect to some approximation of the Skolem maps represented by $\Phi_0, \Phi_1, \Phi_2, \ldots$.

2.1 Updates

In order to define $T$, we start by introducing the concept of “update”, which is nothing but a finite partial function over $\mathbb{N}$. Realizers of atomic formulas will return these finite partial functions,
or “updates”, as new pieces of information that they have learned about the Skolem function $\Phi_0, \Phi_1, \ldots$. Skolem functions, in turn, are used as “oracles” during computations in the system $T_{\text{class}}$. Updates are new associations input-output that are intended to correct, and in this sense, to update, wrong oracle values used in a computation.

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

1. An update set $U$, shorty 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 $m = m'$. Two updates $U_1, U_2$ are consistent if $U_1 \cup U_2$ is an update.
3. $U$ is the set of all updates.
4. The consistent union $U_1 \cup U_2$ of $U_1, U_2 \in U$ is $U_1 \cup U_2$ minus all triples of $U_2$ which are inconsistent with some triple of $U_1$.

The consistent union $U_1 \cup U_2$ 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 $U_1 \cup U_2 \neq U_2 \cup U_1$. $U$ represents a way of selecting a consistent subset of $U_1 \cup U_2$, such that $U_1 \cup U_2 = \emptyset \implies U_1 = U_2 = \emptyset$.

**2.2 The System $T$**

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

1. The first operation is denoted by the constant $\min : U \to \mathbb{N}$. $\min$ takes as argument an update constant $\overline{U}$; it returns the minimum numeral $a$ such that $(a, n, m) \in U$ for some $n, m \in \mathbb{N}$, if any exists; it returns $0$ otherwise.
2. The second operation is denoted by the constant $\get : U \to \mathbb{N}^3 \to \mathbb{N}$. $\get$ takes as arguments an update constant $\overline{U}$ and three numerals $a, n, l$; it returns $m$ if $(a, n, m) \in U$ for some $m \in \mathbb{N}$ (i.e. if $(a, n)$ belongs to the domain of the partial function $U$); it returns $l$ otherwise.
3. The third operation is denoted by the constant $\mkupd : \mathbb{N}^3 \to U$. $\mkupd$ takes as arguments three numerals $a, n, m$ and transforms them into (the constant coding in $T$) the update $\{(a, n, m)\}$.
4. The forth operation is denoted by the constant $\psi : U^2 \to U$. $\psi$ takes as arguments two update constants and returns the update constant denoting their consistent union.

We observe that the constants $\min, \get, \mkupd, \overline{U}, \psi$, and the type $U$ are just syntactic sugar and may be avoided by coding finite partial functions into natural numbers. System $T$ may thus be coded in Gödel’s $T$.

As proved in [10], $T$ is strongly normalizing, has the uniqueness-of-normal-form property and the following normal form theorem also holds.

**Lemma 1 (Normal Form Property for $T$).** Assume $A$ is either an atomic type or a product type. Then any closed normal term $t \in T$ of type $A$ is: a numeral $n : \mathbb{N}$, or a boolean $\mathbf{true}, \mathbf{false} : \mathbf{bool}$, or an update constant $\overline{U} : U$, or a constant of type $A$, or a pair $\langle u, v \rangle : B \times C$.

**2.3 The System $T_{\text{class}}$**

We now define a classical extension of $T$, that we call $T_{\text{class}}$, with a Skolem function symbol for each arithmetical formula. The elements of $T_{\text{class}}$ will represent (non-computable) realizers.

**Definition 2 (The System $T_{\text{class}}$).** Define $T_{\text{class}} = T + SC$, where $SC$ is a countable set of Skolem function constants, each one of type $\mathbb{N} \to \mathbb{N}$. We assume to have an enumeration $\Phi_0, \Phi_1, \Phi_2, \ldots$ of all the constants in $SC$ (while generic elements of $SC$ will be denoted with letters $\Phi, \Psi, \ldots$).
\[ \begin{align*}
\text{Types} & \quad \sigma, \tau ::= \mathbb{N} \mid \mathbf{U} \mid \sigma \rightarrow \tau \mid \sigma \times \tau \\
\text{Constants} & \quad c ::= \mathbf{R}_c \mid \mathbf{if}_c \mid 0 \mid \mathbf{S} \mid \mathbf{True} \mid \mathbf{False} \mid \mathbf{min} \mid \mathbf{get} \mid \mathbf{mkupd} \mid \mathbf{U} \mid \mathbb{U} (\forall U \in \mathbf{U}) \\
\text{Terms} & \quad t, u ::= c \mid x^\tau \mid tu \mid \lambda x^\tau u \mid (t, u) \mid \pi_0 u \mid \pi_1 u \\
\text{Typing Rules for Variables and Constants} & \quad \begin{array}{rl}
x^\tau : \tau & | 0 : \mathbb{N} \mid \mathbf{S} : \mathbb{N} \rightarrow \mathbb{N} \mid \mathbf{True} : \mathbf{Bool} \mid \mathbf{False} : \mathbf{Bool} \\
& | \mathbf{min} : \mathbb{U} \rightarrow \mathbb{U} \mid \mathbf{get} : \mathbb{U} \rightarrow \mathbb{N} \rightarrow \mathbb{N} \rightarrow \mathbb{N} \mid \mathbf{mkupd} : \mathbb{N} \rightarrow \mathbb{N} \rightarrow \mathbb{U} \\
& | \mathbf{if}_c : \mathbf{Bool} \rightarrow \tau \rightarrow \tau \rightarrow \tau \mid \mathbf{R}_c : \tau \rightarrow (N \rightarrow (\tau \rightarrow \tau)) \rightarrow \mathbb{N} \rightarrow \tau
\end{array}
\]
\[ \text{Reduction Rules} \quad \begin{array}{ll}
R_{cuv0} & \rightarrow u \\
R_{cuvS(t)} & \rightarrow vt(R_{cuvv}) \\
\mathbf{if}_c \mathbf{True} uv & \rightarrow u \\
\mathbf{if}_c \mathbf{False} uv & \rightarrow v \\
\pi_e(u_0, u_1) & \rightarrow u_i, i = 0, 1
\end{array} \]
\[ \text{Typing Rules for Composed Terms} \quad \begin{array}{ll}
l : \sigma \rightarrow \tau & \quad u : \sigma \\
tu : \tau & \quad \lambda x^\tau u : \sigma \rightarrow \tau \\
\langle u, t \rangle : \sigma \times \tau & \quad u : \tau_0 \times \tau_1 \\
\pi_i u : \tau_i & \quad i \in \{0, 1\}
\end{array} \]

Fig. 1. The extension $T$ of Gödel’s system $T$

Every $\Phi \in \mathbf{SC}$ represents a Skolem function for some arithmetical formula $\exists y^\mathbb{N} A(x, y)$, taking as argument a number $x$ and returning some $y$ such that $A(x, y)$ is true if any exists, and an arbitrary value otherwise. In general, there is no set of computable reduction rules for the constants in $\mathbf{SC}$, and therefore no set of computable reduction rules for $T_{	ext{class}}$. Each (in general, non-computable) term $t \in T_{	ext{class}}$ is associated to a set $\{t[s] \mid s \in T, s : \mathbb{N} \rightarrow \mathbb{N}\} \subseteq T$ of computable terms we call its “approximations”, one for each term $s : \mathbb{N} \rightarrow \mathbb{N}$ of $T$, which is thought as a sequence $s_0, s_1, s_2, \ldots$ of computable approximations of the oracles $\Phi_0, \Phi_1, \Phi_2, \ldots$ (with $s_i$ we denote $s(i)$).

**Definition 3 (Approximation at State).**

1. A state is a closed term of type $\mathbb{N}^2 \rightarrow \mathbb{N}$ of $T$. If $i$ is a numeral, with $s_i$ we denote $s(i)$.
2. Assume $t \in T_{	ext{class}}$ and $s$ is a state. The “approximation of $t$ at a state $s$” is the term $t[s]$ of $T$ obtained from $t$ by replacing each constant $\Phi_i$ with $s_i$.

### 3 Interactive Realizability for $\mathbf{HA}^\omega + \mathbf{EM} + \mathbf{SK}$

In this section we introduce a notion of realizability based on interactive learning for $\mathbf{HA}^\omega + \mathbf{EM} + \mathbf{SK}$, Heyting Arithmetic in all finite types (see e.g. Troelstra [26]) plus Excluded Middle and Skolem axiom schemes for all arithmetical formulas. Then we prove our main Theorem, the Adequacy Theorem: “if a closed formula is provable in $\mathbf{HA}^\omega + \mathbf{EM} + \mathbf{SK}$, then it is realizable”.

We first define the formal system $\mathbf{HA}^\omega + \mathbf{EM} + \mathbf{SK}$. We represent atomic predicates of $\mathbf{HA}^\omega + \mathbf{EM} + \mathbf{SK}$ with closed terms of $T_{	ext{class}}$, of type $\mathbf{Bool}$. Terms of $\mathbf{HA}^\omega + \mathbf{EM} + \mathbf{SK}$ are elements of $T_{	ext{class}}$. 
We now define the language of the arithmetical theory $HA$. As usual, one has $F$ written as $\Phi$ we shall denote the formula $F$ where $A$ say that an atomic formula $mkupd_{iut}$ occurs. Such unique formula assume that the function $n$ $N$ into $x$ $L$ $A$ of implication as in bi-intuitionistic logic and means $\rightarrow boolean connectives. As usual, we shall use infix notation: for example, we write $t_1 \Rightarrow_{Bool} t_2$ in place of $\Rightarrow_{Bool} t_1 t_2$ and similarly for the other connectives.

3.1 Language of $HA^\omega + EM + SK$

We now define the language of the arithmetical theory $HA^\omega + EM + SK$.

Definition 4 (Language of $HA^\omega + EM + SK$). The language $L_{Class}$ of $HA^\omega + EM + SK$ is defined as follows.

1. The terms of $L_{Class}$ are all $t \in T_{Class}$.
2. The atomic formulas of $L_{Class}$ are all $Q \in T_{Class}$ such that $Q : Bool$.
3. The formulas of $L_{Class}$ are built from atomic formulas of $L_{Class}$ by the connectives $\vee, \wedge, \rightarrow, \neg, \forall, \exists$ as usual, with quantifiers possibly ranging over variables $x_1, y_2, z_3, \ldots$ of arbitrary finite type $\tau$ of $T_{Class}$.
4. A formula of $L_{Class}$ is said arithmetical if it does not contain constants in $SC$ and all its quantifiers range over the type $N$, i.e. it has one of the following forms: $\forall x^N A, \exists x^N A, A \vee B, A \wedge B, A \rightarrow B, A \wedge B, P$, with $A, B$ arithmetical and $P$ atomic formula of $T$.

We denote with $\bot$ the atomic formula $False$ and with $\neg A$ the formula $A \rightarrow \bot$. $A \wedge B$ is the dual of implication as in bi-intuitionistic logic and means $A$ and the opposite of $B^\prime$. If $F$ is a formula of $L_{Class}$ in the free variables $x_1, \ldots, x_n$ and $t_1 : \tau_1, \ldots, t_n : \tau_n$ are terms of $L_{Class}$, with $F(t_1, \ldots, t_n)$ we shall denote the formula $F[t_1/x_1, \ldots, t_n/x_n]$. Sequences of variable $x_1, \ldots, x_k$ will be written as $x$. We denote with $(x)$ a term of $T$ in the free numeric variables $x$ representing a injection of $N^k$ into $N$. Moreover, for every sequence of numerals $n = n_1, \ldots, n_k$, we define $\langle n \rangle := \langle x \rangle [n/x]$ and assume that the function $n \mapsto \langle n \rangle$ is a bijection.

The Excluded Middle axiom scheme $EM$ is defined as the set of all formulas of the form:

$$\forall x^N. A(x) \lor \neg A(x)$$

where $A$ is an arithmetical formula.

The Skolem axiom scheme $SK$ contains for each arithmetical formula $A(x,y)$ an axiom:

$$\forall x^N. \exists y^N A(x,y) \rightarrow A(x, \Phi(x))$$

with $\Phi \in SC$. We assume that for every $\Phi \in SC$ there is in $SK$ one and only one formula in which $\Phi$ occurs. Such unique formula $A$ is said to be the formula associated to $\Phi$ and $\Phi$ will be sometimes written as $\Phi_A$. If $s$ is a state and $\Phi_1 = \Phi_2$, with $s_A$ we denote $s_1$ and with $mkupd A u t$ we denote $mkupd i u t$.

For each formula $F$ of $L_{Class}$, its involutive negation $F^\perp$ is defined by induction on $F$. First, we say that an atomic formula $P$ is positive if it is of the form $\neg_{Bool} \ldots \neg_{Bool} Q$, $Q$ is not of the form $\neg_{Bool} R$, and the number of $\neg_{Bool}$ in front of $Q$ is even. Then we define:

$$(\neg_{Bool} P)^\perp = P \text{ (if } P \text{ positive) }$$

$$P^\perp = \neg_{Bool} P \text{ (if } P \text{ positive) }$$

$$(A \wedge B)^\perp = A^\perp \lor B^\perp$$

$$(A \rightarrow B)^\perp = A \wedge B$$

$$\forall x^N A^\perp = \exists x^N A$$

As usual, one has $(F^\perp)^\perp = F$.

We now fix a special set of formulas $\Gamma$.

Definition 5 (Set $\Gamma$). We fix an arbitrary finite set $\Gamma$ of arithmetical formulas $A(x,y)$ of $L_{Class}$.
In the following, $\Gamma$ will serve as a parameter in order to relativize the definitions of the realizability relation and of the ordering of states provided in [7]. The idea is that any given proof
in the system $\text{HA}^\omega + \text{EM} + \text{SK}$ uses only a finite number of instances of $\text{EM}$ and $\text{SK}$. Thus, it is enough to specialize the atomic case of the definition of realizability in such a way it refers only to the formulas in $\Gamma$. The restriction is necessary in order to avoid to speak about the truth of an infinite number of formulas, as done in [7]. When we shall have to interpret a particular proof $P$, we will choose $\Gamma$ as containing all the sub-formulas of the classical axioms appearing in $P$.

### 3.2 Truth Value of a Formula in a State

The axioms of the system $\text{HA}^\omega + \text{EM} + \text{SK}$ give a great computational power to the system $\mathcal{T}_{\text{Class}}$: thanks to the use of Skolem functions as oracles, one can “compute” by a term $\chi_F$ of $\mathcal{T}_{\text{Class}}$ the truth value of any arithmetical formula $F$. When one effectively evaluates $\chi_F$ in a particular state $s$, we say that one computes the truth value of a formula $F$ in a state $s$.

**Definition 6 (Truth Value of a Formula $F$ in a State $s$).** For every arithmetical formula $F(x)$ of $\mathcal{L}_{\text{Class}}$ we define, by induction on $F$, a term $\chi_F : \text{Bool}$ of system $\mathcal{T}_{\text{Class}}$, with the same free variables of $F$:

1. $\chi_F(n)$ is a term of $\mathcal{T}_{\text{Class}}$ denoting, in any standard model of $\text{HA}^\omega + \text{EM} + \text{SK}$, the truth value of $F(n)$.
2. $F^*(n)$ is a term of $\mathcal{T}$ computing what would be the truth value of $F(n)$ in some standard model of $\text{HA}^\omega + \text{EM}$ under the (possible false) assumption that the interpretation mapping $\Phi_i$ to $s_i$ satisfies the axioms of SK.

We remark that thus $F^*(n)$ is only a conditional truth value: if $F^*(n)$ is not the correct truth value of $F(n)$ – it may well happen – then the interpretation mapping $\Phi_i$ in $s_i$ does not satisfy the axioms of SK. This subtle point is what makes possible learning in Interactive realizability: whenever a contradiction follows, realizers are able to effectively find counterexamples to the assertion that the interpretation mapping $\Phi_i$ in $s_i$ satisfies the axioms of SK. We also observe that this way of computing the truth of a formula comes from the epsilon substitution method (see Avigad [9], Mints et al. [20]).

Every state $s$ is considered as an approximation of the Skolem functions denoted by the constants of $\mathcal{S}$: for each formula $A$, $s_A$ may be a correct approximation of $\Phi_A$ on some arguments, but wrong on other ones. More precisely, we are going to consider the set $\text{def}(s)$ of the pairs $(i, n)$

such that $\Phi_i = \Phi_A$ and $A \in \Gamma \Rightarrow \exists y A(n, y) \rightarrow A(n, s_i(n))$ is true as the real “domain” of $s_i$, representing the set of arguments at which $s_i$ is surely a correct approximation of $\Phi_i$, in the sense that $s_i$ returns an appropriate witness if any exists. We point out that if $\Phi_i = \Phi_A$ and $A \notin \Gamma$, then trivially $(i, n) \in \text{def}(s)$. The choice is made just for technical convenience, since one is not interested in the behaviour of $s$ outside $\Gamma$. We also define an ordering between states: we say that $s' \geq s$ if, intuitively, $s'$ is at least as good an approximation as $s$. Thus, we ask that if $s$ is a correct approximation at argument $(i, n)$ also $s'$ is and in particular $s'(n) = s_i(n)$.

**Definition 7 (Domains, Ordering between States).**

We define

$$\text{def}(s) = \{ (i, n) \mid \Phi_i = \Phi_A \text{ and } (A \in \Gamma \Rightarrow \exists y A(n, y) \rightarrow A(n, s_i(n))) \}$$

where $i$ and $n$ range over numerals and sequences of numerals.
2. Let $s$ and $s'$ be two states. We define $s' \geq s$ if and only if for all $(i, \langle n \rangle), (i, \langle n \rangle) \in \text{def}(s)$ implies $s_i(n) = s'_i(n)$.

We remark that by definition, $s' \geq s$ implies $\text{def}(s') \supseteq \text{def}(s)$ and that thanks to the restriction to $T$ the relation $s' \geq s$ is arithmetical, because the condition $(i, \langle n \rangle) \in \text{def}(s)$ is non-trivial only for finitely many $i$. From now onwards, for every pair of terms $t_1, t_2$ of system $T$, we shall write $t_1 = t_2$ if they are the same term modulo the equality rules corresponding to the reduction rules of system $T$ (equivalently, if they have the same normal form).

3.3 Interactive Realizability

For every formula $A$ of $\mathcal{L}_{\text{class}}$, we now define what type $|A|$ a realizer of $A$ must have.

**Definition 8 (Types for realizers).** For each formula $A$ of $\mathcal{L}_{\text{class}}$ we define a type $|A|$ of $\mathcal{T}_{\text{class}}$ by induction on $A$:

$$
|P| = \emptyset, \text{ if } P \text{ is atomic}
$$

$$
|A \land B| = |A| \times |B| \quad |\exists x \forall T A| = \tau \times |A| \quad |A \land B| = |A| \times |B^\bot|
$$

$$
|A \lor B| = \text{Bool} \times (|A| \times |B|) \quad |\forall x \exists T A| = \tau \rightarrow |A| \quad |A \rightarrow B| = |A| \rightarrow |B|
$$

Let now $p_0 := \pi_0 : \sigma_0 \times (\sigma_1 \times \sigma_2) \rightarrow q_0, p_1 := \pi_0 \pi_1 : \sigma_0 \times (\sigma_1 \times \sigma_2) \rightarrow \sigma_1$ and $p_2 := \pi_0 \pi_1 : \sigma_0 \times (\sigma_1 \times \sigma_2) \rightarrow \sigma_2$ be the three canonical projections from $\sigma_0 \times (\sigma_1 \times \sigma_2)$. We define the realizability relation $t \models F$, where $t \in T_{\text{class}}, F \in \mathcal{L}_{\text{class}}$ and $t : |F|$.

**Definition 9 (Interactive Realizability).** Assume $s$ is a state, $t$ is a closed term of $T_{\text{class}}$, $F \in \mathcal{L}_{\text{class}}$ is a closed formula, and $t : |F|$. We define first the relation $t \models_s F$ by induction and by cases according to the form of $F$:

1. $t \models_s Q$ for some atomic $Q$ if and only if $U = \uparrow t[s]$ implies:
   - for every $(i, \langle n \rangle) \in \emptyset$, $\Phi_i = \Phi_A$ for some $A \in \Gamma$, and $A^s(n, s_i(n)) = \text{False}$ and $A^s(n, m) = \text{True}$.
   - $U = \emptyset$ implies $Q[s] = \text{True}$

2. $t \models_s A \land B$ if and only if $p_0 t \models_s A$ and $p_1 t \models_s B$

3. $t \models_s A \lor B$ if and only if either $p_0 t[s] = \text{True}$ and $p_1 t \models_s A$, or $p_0 t[s] = \text{False}$ and $p_2 t \models_s B$

4. $t \models_s A \rightarrow B$ if and only if for all $u$, if $u \models_s A$, then $u \models_s B$

5. $t \models_s A \land B$ if and only if $p_0 t \models_s A$ and $p_1 t \models_s B^\bot$

6. $t \models_s \forall x \exists T A$ if and only if for all closed terms $u : \tau$ of $T$, $t u \models_s A[u/x]$

7. $t \models_s \exists x \forall T A$ if and only for some closed term $u : \tau$ of $T$, $p_0 t[s] = u$ and $p_1 t \models_s A[u/x]$}

We define $t \models F$ if and only for all states $s$ of $T$, $t \models_s F$.

The ideas behind the definition of $\models_s$ in the case of $\text{HA}^\omega + \text{EM} + \text{SK}$ are those we already explained in [7]. A realizer is a term $t$ of $T_{\text{class}}$, possibly containing some non-computable Skolem function of $\text{SC}$; if such a 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 realization 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 some Skolem function $\Phi_i$.

The next proposition tells that realizability at state $s$ respects the notion of equality of $T_{\text{class}}$ 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 in the state $s$. 


We shall show that this will not be the case, and the intuitive reason is that
Adequacy Theorem 1 that with respect to any interpretation of its Skolem functions. Finally, if there existed a counterexample
values of the Skolem functions needed to compute
n
By straightforward induction on
Proof. By straightforward induction on A.

In the following, we use a standard natural deduction system for HAω + EM + SK, together with a term assignment in the spirit of Curry-Howard correspondence for classical logic. We denote with HAω + EM + SK ⊢ t : A the derivability relation in that system, where t is a term of TClaus and A is a formula of Lcous. All details can be found in [3], [7].

The main theorem about interactive realizability is the Adequacy Theorem: if a closed formula is provable in HAω + EM + SK, then it is realizable (see [7] for a proof).

Theorem 1 (Adequacy Theorem). If A is a closed formula such that HAω + EM + SK ⊢ t : A and all the subformulas of the instances of EM and SK used in the derivation belong to Γ, then t ⊨ A.

4 Conservativity of HAω + EM + SK over HAω + EM (HA + EM)

The aim of this section is to use Interactive realizability in order to prove that for every arithmetical formula A, if HAω + EM + SK ⊢ A then HAω + EM ⊢ A (HA + EM ⊢ A). Since we know by the Adequacy Theorem[1] that HAω + EM + SK ⊢ t : A implies HAω + EM + SK ⊢ t : A and HAω proves t ⊨ A, our goal is to show in HAω + EM that t ⊨ A implies A.

The intuitive reason this latter result is true is the following: one can always find an approximation s of the Skolem functions of t which is good enough to contain all the information needed by t to compute the true witnesses for A against any particular purported counterexample. The idea is that one has only to collect finitely many values of each Skolem function called during the execution of the program represented by t. To this end, it suffices to invoke the excluded middle a number of times which, intuitively, can be expressed in a proof formalizable in HAω + EM. This is possible because HAω + EM is strong enough to prove the normalization of each term t of TClaus with respect to any interpretation of its Skolem functions. Finally, if there existed a counterexample to A, it would be possible to falsify the construction of the realizer t in the state s. Since t is a self-correcting program, it would be able to correct one of the values of s it has used in the computation of some witness for A. But s is constructed as to be correct on all the values used by t, which entails a contradiction.

For example, let A = ∃x∀y∃zB(x, y, z). Then one can find a state s which contains all the values of the Skolem functions needed to compute u = π0[s]. Suppose a counterexample m to the formula ∃y∃zB(n, y, z) existed. Then one can find a state s’ ≥ s which contains all the values of the Skolem functions needed to compute l = π0(⟨π1l⟩m)[s’]. Now, we would have that B(n, m, l) is false; thus, π1(⟨π1l⟩m)[s’] would be equal to some update U containing some corrections to s’. We shall show that this will not be the case, and the intuitive reason is that s’ can be chosen as to be correct everywhere it is needed.

We now elaborate our argument. We start with a definition axiomatizing the informal concept that a state s contains all the information needed to compute the normal form of a term t of ground type. Namely, if for every s’ extending s the evaluation of t in the state s’ gives the same result obtained evaluating t in s, then we may assume all the relevant information is already in s.

Definition 10 (Definition of a term in a state s). For every state s and term t of TClaus of atomic type, we define t ↪ s (and we say “t is defined in s”) as the statement: for all states s’ ≥ s, t[s’] = t[s].

Remark 1. There is another, perhaps more intuitive way to express the concept of “being defined in the state s”. For every state s we may define a binary reduction relation t ∴ s ⊆ TClaus × TClaus as follows: t ∴ s u if either t → u in TClaus or u is obtained from t by replacing one of its subterms Φi(n) with a numeral m = s_i(n) such that (i, n) ∈ def(s). Then one could say that t is defined in s if t ∴ s where s is either a numeral, a boolean or an update. Though this approach works well, it is unsuitable to be directly formalized in HAω, because in that system one cannot express this syntactical reasoning on terms.
We now define for every type $\tau$ a set of “computable” terms of type $\tau$ by means of the usual Tait-style computability predicates [22]. In our case, following the approach of the previous discussion, we consider a term $t$ of ground type to be computable if for every state $s$, one can find a state $s' \geq s$ such that $t$ is defined in $s'$. The notion is lifted to higher types as usual.

**Definition 11 (Computable terms).**

For every type $\tau$ of $T_{\text{class}}$, we define a set of closed terms of $T_{\text{class}}$ of type $\tau$ as follows:

- $\llbracket N \rrbracket = \{ n : N \mid \text{for all states } s \text{ there is a state } s' \geq s \text{ such that } t \downarrow s' \}$
- $\llbracket \text{Bool} \rrbracket = \{ t : \text{Bool} \mid \text{for all states } s \text{ there is a state } s' \geq s \text{ such that } t \downarrow s' \}$
- $\llbracket U \rrbracket = \{ t : U \mid \text{for all states } s \text{ there is a state } s' \geq s \text{ such that } t \downarrow s' \}$
- $\llbracket \tau \rightarrow \sigma \rrbracket = \{ t \mid \forall u \in \llbracket \tau \rrbracket \ t u \in \llbracket \sigma \rrbracket \}$
- $\llbracket \tau \times \sigma \rrbracket = \{ t \mid \pi_0 t \in \llbracket \tau \rrbracket \text{ and } \pi_1 t \in \llbracket \sigma \rrbracket \}$

In order to show that every term $t$ in $T_{\text{class}}$ is computable, as usual we need to prove that the set of computable terms is saturated with respect to some suitable relation. In our case, two terms are related if they are equal in all states greater than some state.

**Theorem 2 (Computability Theorem).**

Let $v : \tau$ be a term of $T_{\text{class}}$ and suppose that all the free variables of $v$ are among $x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}$. If $t_1 \in \llbracket \sigma_1 \rrbracket, \ldots, t_n \in \llbracket \sigma_n \rrbracket$, then $v[t_1/x_1^{\sigma_1}, \ldots, t_n/x_n^{\sigma_n}] \in \llbracket \tau \rrbracket$.

**Proof.** We proceed by induction on $v$. If $v = t$ and $t \in \llbracket \tau \rrbracket$, then $u \in \llbracket \tau \rrbracket$ by trivial application of Lemma 2.

**Notation 1** For any term $w$ in $T_{\text{class}}$, we denote $w[t_1/x_1^{\sigma_1}, \ldots, t_n/x_n^{\sigma_n}]$ with $\overline{w}$.

1. $v$ is a variable $x_i^{\sigma_i} : \sigma_i$ and $\tau = \sigma_i$. Then, $\overline{v} = t_1 \in \llbracket \sigma_i \rrbracket = \llbracket \tau \rrbracket$.
2. $v$ is 0, True, False, $U$: trivial.
3. $v$ is $uvv$, then by means of typing rules, $u : \sigma \rightarrow \tau$, $w : \sigma$. Since by induction hypothesis $\overline{\tau} \in \|\sigma \rightarrow \tau\|$ and $\overline{w} \in \|\sigma\|$, we obtain $\overline{v} = \overline{uvw} \in \|\overline{\tau}\|$.

4. $v$ is $\lambda x.\tau'.u : \tau_1 \rightarrow \tau_2$. Then, by means of typing rules, $u : \tau_2$. Suppose now, for a term $t : \tau_1$ in $\mathcal{T}_{\text{class}}$, that $t \in \|\tau_1\|$. We have to prove that $\overline{vt} \in \|\tau_2\|$. We have:

$$\overline{vt} = (\lambda x.\tau'u)[t_1/x^{\tau_1}_1 \cdots t_n/x^{\tau_1}_n]t = (\lambda x.\tau_1u)t[t_1/x^{\tau_1}_1 \cdots t_n/x^{\tau_1}_n] = u[t/x^{\tau_1}][t_1/x^{\tau_1}_1 \cdots t_n/x^{\tau_1}_n] = u[t/x^{\tau_1}t_1/x^{\tau_1}_1 \cdots t_n/x^{\tau_1}_n]$$

By induction hypothesis, this latter term belongs to $\|\tau_2\|$. We conclude $\overline{vt} \in \|\tau_2\|$. 

5. $v$ is $(u, w) : \tau_0 \rightarrow \tau_1$. By means of typing rules, $u : \tau_0$, $w : \tau_1$ and by induction hypothesis $\pi_0\overline{v} = \overline{v} \in \|\tau_0\|$ and $\pi_1\overline{v} = \overline{w} \in \|\tau_1\|$. The thesis $\overline{v} \in \|\tau_0 \times \tau_1\|$ follows by definition.

6. $v$ is $\pi_i(u) : \tau_i$, $i = 0, 1$, where $u : \tau_0 \rightarrow \tau_1$. $\pi_i\overline{u} \in \|\tau_i\|$ because $\overline{u} \in \|\tau_0 \rightarrow \tau_1\|$ by induction hypothesis.

7. $v$ is if :: $\text{Bool} \rightarrow \tau \rightarrow \tau \rightarrow \tau$. Suppose that $u \in \|\text{Bool}\|$, $u_1 \in \|\tau\|$, $u_2 \in \|\tau\|$. Then, for all states $s$ there exists $s' \geq s$ such that $u \downarrow s'$. We have to prove that if $\tau u_1 u_2 \in \|\tau\|$. Let $s$ be a state and let $s' \geq s$ be such that $u \downarrow s'$. If $u[s'] = \text{True}$, then for all $s'' \geq s'$, if $\tau u_1 u_2[s''] = u_1[s'']$ and $u_1 \in \|\tau\|$. If $u[s'] = \text{False}$, then for all $s'' \geq s'$, if $\tau u_1 u_2[s''] = u_2[s'']$ and $u_2 \in \|\tau\|$. By Lemma 2, we conclude if $\tau u_1 u_2 \in \|\tau\|$.

8. $v$ is $R_\tau : \tau \rightarrow (\text{Nat} \rightarrow (\tau \rightarrow \tau)) \rightarrow \text{Nat} \rightarrow \tau$. Suppose that $u \in \|\tau\|$, $w \in \|\text{Nat} \rightarrow (\tau \rightarrow \tau)\|$, $z \in \|\text{Nat}\|$. We have to prove that $R_\tau uwz \in \|\tau\|$. By a plain induction, it is possible to prove, for each numeral $n$, $R_\tau uwn \in \|\tau\|$. Let $s$ be a state and let $s'' \geq s$ be such that $z \downarrow s'$. Let $z[s'] = n$ with $n$ numeral. Then for all $s'' \geq s'$,

$$R_\tau uwz[s''] = R_\tau uwn[s''] \in \|\tau\|$$

By Lemma 2, we conclude $R_\tau uwn \in \|\tau\|$. 

9. $v$ is min :: $\text{Nat} \rightarrow \text{Nat}$. Suppose, for a term $u$ in $\mathcal{T}_{\text{class}}$, that $u \in \|\text{Nat}\|$. Let $s$ be a state. Since $u \in \|\text{Nat}\|$, there exists an update $U$ such that for all $s'' \geq s$, $u[s''] = U$. Then for all $s'' \geq s'$, min $u[s''] = \min U = n$ for some numeral $n$. By definition of $\|\text{Nat}\|$, min $u \in \|\text{Nat}\|$. 

10. $v$ is $\psi : \text{Nat} \rightarrow \text{Nat} \rightarrow \text{Nat}$. Suppose that $u_1 \in \|\text{Nat}\|$ and $u_2 \in \|\text{Nat}\|$. We have to prove that $\psi u_1 u_2 \in \|\text{Nat}\|$. Let $s$ be a state. Since $u_1 \in \|\text{Nat}\|$ there exists $s' \geq s$ such that $u_1 \downarrow s'$. Since $u_2 \in \|\text{Nat}\|$, there exists $s'' \geq s'$ such that $u_2 \downarrow s''$. Therefore, there exist two constants $U_1$ and $U_2$ such that for all $s''' \geq s''$, $u_1[s'''] = U_1$ and $u_2[s'''] = U_2$. Finally, for all $s''' \geq s''$, $\psi u_1 u_2[s'''] = \psi U_1 U_2 = U_3$ and by definition of $\|\text{Nat}\|$, $\psi u_1 u_2 \in \|\text{Nat}\|$. 

11. $v$ is $S, \text{mkup}d$ or $\text{get}$. Analogous to the previous case.

12. $v$ is a constant $\Phi_i : \text{Nat} \rightarrow \text{Nat}$ in $\mathcal{S}\mathcal{C}$. Suppose now, for a term $u : \text{Nat}$, that $u \in \|\text{Nat}\|$. We have to prove that $\Phi_i u \in \|\text{Nat}\|$. Let $s$ be a state. We must show that there exists a $s' \geq s$ such that $\Phi_i u \downarrow s'$. Since $u \in \|\text{Nat}\|$, there exists a state $s' \geq s$ such that $u \downarrow s'$. Let $n = u[s']$, with $n$ numeral, and $m = s'_i(n)$. Let $\Phi_i = \Phi_{A(x,y)}$. If $A \notin \Gamma$, then trivially $(i, n) \in \text{def}(s')$ by definition. Therefore for all $s'' \geq s'$, $\Phi_i u[s''] = s'_i(n) = m$ and we are done. Hence, we may assume $A \in \Gamma$. There are two cases, and this is the only point of this proof in which we use EM.

(a) $A(n, m)$ is true. Therefore, for all $s'' \geq s'$, $s'_i(n) = m$ because $(i, n) \in \text{def}(s')$. Thus, for all $s'' \geq s'$, $\Phi_i u[s''] = s'_i(n) = m$, which is the thesis.
Thus, we can show with the excluded middle that for every state $s$, since $t = s' \geq s$, that when we evaluate $B$, in a state $s$ which contains all the information needed to evaluate $t$. According to the Definition 6 of the truth value $A^*$ of a formula $A$ in a state $s$, when we compute $A^*$ we need only a finite number of Skolem function values, one for each quantifier of $A$. Thus, we can show with the excluded middle that for every state $s$ there exists a state $s' \geq s$ such that when we evaluate $A$ in the state $s'$ we obtain the real truth value of $A$.

**Proposition 2.** Let $A(x)$ be any arithmetical formula and $n$ be numerals. For every state $s$, there exists a state $s' \geq s$ such that $A^*(n) = \text{True}$ if and only if $A(n)$ is true.

**Proof.** We prove the thesis by induction on $A$. Let $s$ be any state. The cases in which $A$ is atomic or $A = B \lor C, B \land C, B \rightarrow C$ are trivial. Let us consider those in which $A$ starts with a quantifier.

- $A(n) = \exists y^B(B(n, y))$. By the excluded middle, we extend $s$ to a state $s' \geq s$ such that $m = s'_B(n)$ implies that $\exists y^B(B(n, y) \rightarrow B(n, m))$

By induction hypothesis, there exists a state $s'' \geq s'$ such that $B(n, m)$ is true if and only if $B''(n, m) = \chi_B(n, m)[s''] = \text{True}$

Assuming $\Phi_i = \Phi_B$, since $(i, (n)) \in \text{def}(s')$, we have $s''_B(n) = s'_B(n)$. Since $A''(n) = \chi_B(n, \Phi_B(n))[s''] = \chi_B(n, m)[s'']$ and $A(n)$ is equivalent to $B(n, m)$, we obtain the thesis.

- $A(n) = \forall y^B(B(n, y))$. By the excluded middle, we extend $s$ to a state $s' \geq s$ such that $m = s'_B(n)$ implies that $\exists y^B(B(n, y) \rightarrow B^+(n, m))$

By induction hypothesis, there exists a state $s'' \geq s'$ such that $B^+(n, m)$ is true if and only if $(B^+)''(n, m) = \chi_{B^+}(s'') = \text{True}$

Assuming $\Phi_i = \Phi_{B^+}$, since $(i, (n)) \in \text{def}(s')$, we have $s''_{B^+}(n) = s'_B(n)$. Since $A''(n) = \chi_{B^+}(n, \Phi_{B^+}(n))[s''] = \chi_{B^+}(n, m)[s'']$ we obtain the thesis.

Now we prove a special case of the statement that the realizability of a formula implies the formula itself. Namely, we show that $t$ realizes $\perp$ implies $\perp$. The idea, as we have explained before, is to find a state $s$ which contains all the information needed to evaluate $t$.

**Theorem 3 (Consistency of Interactive Realizability).** For every closed term $t$ of $\mathcal{T}_{\text{class}}$, $t \not\models \perp$. In particular, for every state $s$, there exists a state $s' \geq s$ such that $t \not\models s'$.

**Proof.** Suppose, for the sake of contradiction, that there exists a term $t$ such that $t \not\models \perp$. Let $s$ be any state. Since $t : U$, by theorem 2 we have $t \in \|U\|$ and therefore there exists a state $r \geq s$ such that $t \not\models r$. Let $\|t\| = U$ for some update $U$. Since $t \not\models r \models \perp$, $U$ is non-empty: let $(i, (n, m)) \in U$.

By application of theorem 2, if $\Phi_i = \Phi_A$, there exists a state $q \geq r$ such that $\chi_A(n, m) \not\models q$. By definition, $A^*(n, m) = \chi_A(n, m)[q] = b$

for some boolean $b$. Since $t \not\models q \models \perp$ and $\|q\| = U$ (because $\not\models U$ and $q \geq r$), we obtain by definition of realizability that $b = \text{True}$. Let $q_\perp(n) = l$. We have two possibilities:
1. $A(n,l)$ is false. We define the state

$$s' := \lambda x^B \lambda y^B, if \ x = i \land \text{Boo} \ y = \langle n \rangle then \ m else \ q_s(y)$$

Then, $s' \geq q$, for $A(n,l)$ is false. Moreover, since $\chi_A(n,m) \not\vdash q$, for all $q' \geq q$, $\chi_A(n,m)[q'] = b$; by Proposition 2 there exists $q' \geq q$, such that $\chi_A(n,m)[q'] = \text{True}$ if and only if $A(n,m)$ is true. Since $\chi_A(n,m)[q'] = b = \text{True}$, we have that $A(n,m)$ is true. By assumption on $t$, we have $t \not\vdash s' \perp$ and $t[s'] = \overline{U}$, because $s' \geq r$. Since $s'(n) = m$, by definition of $t \not\vdash s' \perp$ we would have both $A'(n,m) = \text{False}$ and $A'(n,m) = \text{True}$, which is a contradiction.

2. $A(n,l)$ is true. By Proposition 2 there is a state $s' \geq q$ such that $A'(n,l) = \text{True}$. By assumption on $t$, we have $t \not\vdash s' \perp$ and $t[s'] = \overline{U}$. But $q_s(n) = t$, $A(n,l)$ is true and $s' \geq q$; therefore $(i,n) \in \text{def}(q)$ and hence $s'_s(n) = l$. By definition of $t \not\vdash s' \perp$, we would have $A'(n,l) = \text{False}$ and $A'(n,m) = \text{True}$, which is in contradiction with $A'(n,l) = \text{True}$.

Finally, we are in a position to prove in $\text{HA}_\omega + \text{EM}$ that the realizability of a formula $A$ implies its truth. For simplicity we assume $A$ is a $\rightarrow$-free, but the result holds also in the general case.

**Theorem 4 (Soundness of Realizability).** Let $A$ be any $\rightarrow$-free arithmetical formula and suppose $t \vdash A$. Then $A$ is true.

**Proof.** We prove a stronger statement. Let $s$ be a state and suppose that for all $s' \geq s$, $t \not\vdash s' A$. We prove by induction on $A$, that $A$ is true.

- $A = P$, with $P$ atomic. Suppose, by the way of contradiction, that $P$ is false. Then we have that for all $s' \geq s$, $t \not\vdash s' P$, which is impossible by Theorem 3.

- $A = B \land C$. Then, for all $s' \geq s$, $t \not\vdash s' A$ and $t \not\vdash s' B$. By induction hypothesis $A$ and $B$ are true, and we obtain the thesis.

- $A = B \lor C$. By Theorem 2 there exists a state $r \geq s$ such that $p_0 t \not\vdash q$. Let $p_0 t[r] = b$ with $b$ boolean, say $b = \text{True}$. Then, by definition, for every $r' \geq r$, $p_0 t[r'] = \text{True}$ and therefore $t \not\vdash r A$. By induction hypothesis $A$ is true, and we obtain the thesis.

- $A = \forall x^B B$. Let $n$ be any numeral. Then, for all $s' \geq s$, $t n \not\vdash s' B(n)$. By induction hypothesis $B(n)$ is true. Therefore, $\forall x^B B$ is true, and we obtain the thesis.

- $A = \exists x^B B$. By Theorem 2 there exists a state $r \geq s$ such that $\pi_0 t \not\vdash q$. Let $\pi_0 t[r] = n$ with $n$ numeral. Then, by definition, for every $r' \geq r$, $\pi_0 t[r'] = n$ and therefore $t \not\vdash r B(n)$. By induction hypothesis $B(n)$ is true, and we obtain the thesis.

Since all the proofs given in this section are formalizable in $\text{HA}_\omega + \text{EM}$ (see Section 5), we are able to prove the conservativity of $\text{HA}_\omega + \text{EM} + \text{SK}$ over $\text{HA}_\omega + \text{EM}$ for arithmetical formulas.

**Theorem 5 (Conservativity of $\text{HA}_\omega + \text{EM} + \text{SK}$ over $\text{HA}_\omega + \text{EM}$).** Let $A$ be a closed arithmetical formula, and suppose

$$\text{HA}_\omega + \text{EM} + \text{SK} \vdash A$$

Then:

$$\text{HA}_\omega + \text{EM} \vdash A \quad (1)$$  

$$\text{HA} + \text{EM} \vdash A \quad (2)$$

**Proof.**

1. We may assume that $A$ is $\rightarrow$-free. Otherwise,

$$\text{HA}_\omega + \text{EM} \vdash A \leftrightarrow B$$
with $B \to$-free and we consider $B$. Since $\Gamma$ is arbitrary, we may assume that all the subformulas of the instances of $\text{EM}$ and $\text{SK}$ used in the derivation belong to $\Gamma$. By formalization of the Adequacy Theorem \ref{adequacy} in $\text{HA}^\omega$ (see Section \ref{5}), we obtain that $\text{HA}^\omega \vdash t \Vdash A$ for some term $t$ of $T_{\text{class}}$. By formalization of the proof of Theorem \ref{realizability} in $\text{HA}^\omega + \text{EM}$, we obtain that $\text{HA}^\omega + \text{EM} \vdash (t \Vdash A) \to A$. We conclude $\text{HA}^\omega + \text{EM} \vdash A$.

2. There are at least two ways to obtain the thesis. On one hand, we may use \ref{adequacy} and the standard result about the conservativity of $\text{HA}^\omega + \text{EM}$ over $\text{HA} + \text{EM}$ for arithmetical formulas (see for example Troesstra [24]). On the other hand, we may code directly terms of system $T_{\text{class}}$ into natural numbers and then express the proofs of point 1) in $\text{HA} + \text{EM}$ (see Section \ref{5}).

5 Formalization of the Proofs in $\text{PA}$ and in $\text{HA}^\omega + \text{EM}$

In this section we explain how to formalize in $\text{PA}$ and $\text{HA}^\omega + \text{EM}$ the proof of the Adequacy Theorem \ref{adequacy} of Section \ref{3} and the proofs of the Computability Theorem \ref{computability} and the Soundness Theorem \ref{soundness} of Section \ref{4}. We start with the case of $\text{PA}$.

5.1 Formalization in $\text{PA}$

One can routinely code in $\text{PA}$ all the concepts we have so far used. As in Tait [22], one may code the terms of $T_{\text{class}}$ with natural numbers and successively the definition of the realizability and computability predicates, which is what we are going to do. As in Tait [22], one may code directly terms of system $T_{\text{class}}$ into natural numbers and then express the proofs of point 1) in $\text{HA} + \text{EM}$ (see Section \ref{5}).

5.2 Formalization in $\text{HA}^\omega + \text{EM}$

Instead of coding everything into natural numbers, which is of limited practical interest, it is more satisfying to formalize our proofs directly in $\text{HA}^\omega + \text{EM}$. There is no serious obstacle to this end, except for a small formalization issue: the notion $t[s]$ of evaluation of a term $t$ of $T_{\text{class}}$ in a state $s$, which we have heavily used in the definitions of the realizability and computability predicates, is not directly representable in $\text{HA}^\omega + \text{EM}$. To begin with, terms of $T_{\text{class}}$ may contain some constant $\Phi \in \mathcal{SC}$ which does not belong to the language of $\text{HA}^\omega$. This problem is easily solved by considering terms of the form $t[s]$ with $s$ state variable. However, in the definition of Interactive realizability for implication and in the statement of the Computability Theorem one needs to define formulas $x \Vdash A$ and $x \in \|\|$, where $x$ is a variable. In these definitions it is necessary to speak about the substitution of an actual state $s$ in the body of a variable $x$, which is impossible in $\text{HA}^\omega$ (remember that $x$ represents a term $t[s]$ of $T$). This last issue is overcome quite easily by considering in place of a term $t : \tau$ in $T_{\text{class}}$ the term $\lambda s^\omega.t[s] : S \to \tau$, where $S := \mathbb{N}^\omega \to \mathbb{N}$ is the type of states. In this way, one makes explicit the functional dependence of $t$ from the state $s$ and transforms $t$ into an object having a semantical denotation. It is however necessary to slightly adapt the definitions of realizability and computability, which is what we are going to do.

First, we give an alternative definition of Interactive realizability, which is shown in \ref{alternative} to be equivalent to Kreisel’s modified realizability for $\text{HA}^\omega$ applied to some Friedman translation of formulas. We denote with $L$ the restriction of the language $L_{\text{class}}$ to the formulas not containing any Skolem function constant $\Phi \in \mathcal{SC}$.

Definition 12 (Alternative Definition of Interactive Realizability). Assume $s : S$ is a closed term of $T$, $t$ is a closed term of $T$, $D \in L$ is a closed formula of $L$, and $t : [D]$. We define by induction on $D$ the relation $t \Vdash_s D$:

1. $t \Vdash_s Q$ if and only if $t = U$ implies:
   - for every $(i, n, m) \in U$, $\Phi_i = \Phi_A$ for some $A \in \Gamma$, and $A^t(n, s_i(n)) = \text{False}$ and $A^t(n, m) = \text{True}$.
   - $U = \emptyset$ implies $Q = \text{True}$

2. $t \Vdash_s A \land B$ if and only if $\pi_0 t \Vdash_s A$ and $\pi_1 t \Vdash_s B$
Then, for every $B$

Theorem 7 and the fact that it is possible to prove by induction the statement $\forall$.

Theorem 4 can be straightforwardly adapted. In particular, in the base case of the induction one

which is formalizable in that system (see [25]).

4 but with the new specified meaning.

Let $v$.

Lemma 3.

see the full version of this paper [8]).

Definition 13 (Definition of a term in a state $s$).

For every term $t$ of type $S \rightarrow \tau$ and $i \in \{0, 1\}$, we define the following notion of projection:

$$\pi_i t := \lambda s^i \cdot \pi t s$$

Finally, for every constant term $c \notin SC$, we define $c^* := \lambda s^c$. We now adapt Definition 10 and Definition 11. Since there is no possibility of confusion, we maintain the same notations of Section 4 but with the new specified meaning.

Definition 14 (Computable terms).

For every type $\tau$ of $T$, we define a set of closed terms of $T$ of type $S \rightarrow \tau$ as follows:

- $\{ t : S \rightarrow N \mid \text{for all states } s \text{ there is a state } s' \geq s \text{ such that } t \downarrow \}$
- $\{ t : S \rightarrow \text{Bool} \mid \text{for all states } s \text{ there is a state } s' \geq s \text{ such that } t \downarrow \}$
- $\{ t : S \rightarrow U \mid \text{for all states } s \text{ there is a state } s' \geq s \text{ such that } t \downarrow \}$
- $\{ t : \tau \rightarrow \sigma \mid \forall u \in \|\tau\| \hspace{1cm} t \cdot u \in \|\sigma\| \}$
- $\{ t : \tau \times \sigma \mid \forall u \in \|\tau\| \hspace{1cm} \tau_1 t \in \|\tau\| \text{ and } \tau_2 t \in \|\sigma\| \}$

The proofs of Lemmas 2 and of the Computability Theorem can be easily adapted (for details, see the full version of this paper [8]).

Lemma 3. For every term $t : S \rightarrow \rho$ of $T$, if for every state $s$ there exists a state $s' \geq s$ and $u \in \|\rho\|$ such that for all states $s'' \geq s'$, $ts'' = us''$, then $t \in \|\rho\|$.

Theorem 7 (Computability Theorem).

Let $v : \tau$ be a term of $T_{\text{class}}$ and suppose that all the free variables of $v$ are among $x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}$. If $t_1 \in \|\sigma_1\|, \ldots, t_n \in \|\sigma_n\|$, then $\lambda s^v. v[s][t_1 s/x_1^{\sigma_1}, \ldots, t_n s/x_n^{\sigma_n}] \in \|\tau\|$.

The proofs of Proposition 2 and Theorem 8 remain exactly the same, while the proof of Theorem 4 can be straightforwardly adapted. In particular, in the base case of the induction one needs to prove that a term $t$, possibly with free variables of type $N$, is computable. This follows from Theorem 7 and the fact that it is possible to prove by induction the statement $\forall x^\tau. \lambda s^x x \in \|N\|$. 
References
7. F. Aschieri, Interactive Realizability for Classical Peano Arithmetic with Skolem Axioms, Tech. Rep., http://hal.inria.fr/hal-00685360.