On the computational content of the Axiom of Choice

Stefano Berardi
Chalmers
Torino University *

Marc Bezem
Utrecht University †

Thierry Coquand
Chalmers
University of Gothenburg ‡

Abstract

We present a possible computational content of the negative translation of classical analysis with the Axiom of Choice. Our interpretation seems computationally more direct than the one based on Gödel's Dialectica interpretation [10, 18]. Interestingly, this interpretation uses a refinement of the realizibility semantics of the absurdity proposition, which is *not* interpreted as the empty type here. We also show how to compute witnesses from proofs in classical analysis, and how to interpret the axiom of Dependent Choice and Spector's Double Negation Shift.

Introduction

It is well-known that the Axiom of Choice [15] is validated by the Brouwer–Heyting–Kolmogoroff explanation of the logical constants [3]. In view of the negative interpretation of classical arithmetic into intuitionistic arithmetic [6], one would expect that it is possible to make constructive sense of the Axiom of Choice as used in informal mathematics, for instance in the use of Zorn's Lemma, or in establishing the existence of a well-ordering on the reals.

This, however, appears to be non-trivial. The combination of the Axiom of Choice *and* the Excluded Middle turns out to be extremely problematic from a constructive point of view. To make constructive sense of such a combination can actually be seen as one the main aims of Hilbert's programme [8, 9]. We address here the more modest question of the analysis of the computational content of the Axiom of Choice, by giving a novel realizability interpretation of the negative translation of the Axiom of Choice.

Our paper is organized as follows. After a presentation of the formal system under consideration, we enumerate some known difficulties in combining the Axiom of Choice and the Excluded Middle. They appear in quite different forms in different formalisms: as a non-conservativity result, as the impossibility of a recursive realization, and as a problem with a formulation in sequent calculus. The central and longer part is then a description of a possible realizability interpretation, with a precise and detailed proof of correctness. This proof of correctness is non-constructive. (We use an intuitionistic meta-theory throughout this paper, unless explicitely stated otherwise.) We show next that this interpretation can be generalized to the case of the axiom of Dependent Choice, and in this case, reduce intuitionistically its correctness to a principle of bar induction. We also give an interpretation of the Double Negation Shift and try a comparison with Spector's bar recursive interpretation of DNS [18, 10, 11], which suggests a computational content of the negative interpretation of Axiom of Choice based on Gödel's Dialectica translation. We end by an heuristic explanation of our realizability interpretation, based on a game-theoretical analysis of proofs.

*Dip. Informatica, C.so Svizzera 185, 10149 Torino, Italy, e-mail `stefano@cs.chalmers.se`.
†Department of Philosophy, P.O. Box 80126, 3508 TC Utrecht, The Netherlands, e-mail `bezem@phil.ruu.nl`.
‡Department of Computer Sciences, S-41296, Gothenburg, Sweden, e-mail `coquand@cs.chalmers.se`.

1 Presentation of HA^ω

1.1 Types

The types of HA^ω are N and with τ, τ' also $\tau \to \tau'$. Here and below types will be denoted by lower case greek letters $\tau, \tau', \ldots$.

1.2 Terms

The terms of HA^ω are built from (typed) variables and constants using lambda abstraction and (well typed) application. There are countably many variables $x, y, z, \ldots$ for each type τ. The constants are: $\mathsf{0} : \mathsf{N}$, $\mathsf{s} : \mathsf{N} \to \mathsf{N}$ and $\mathsf{R}_\tau : \tau \to (\mathsf{N} \to \tau \to \tau) \to \mathsf{N} \to \tau$ for every type τ. Terms are denoted by $M, M', N, \ldots$, and $M : \tau$ expresses that the term M has type τ.

1.3 Formulae

Prime formulae are equations of the form $M = M'$, with $M, M' : \mathsf{N}$. Higher type equations, say $M = M'$ with $M, M' : \mathsf{N} \to \mathsf{N}$, are abbreviations of equations of lowest type, such as $Mx = M'x$ with x fresh. The set of formulae of HA^ω is generated in the usual way from the prime formulae by the boolean connectives $\wedge, \Rightarrow, \bot$ and the quantifiers $\forall, \exists$. We use $\phi, \phi', \ldots$ to denote formulae. We abbreviate $\phi \Rightarrow \bot$ by $\neg\phi$.

1.4 Theory

The theory HA^ω, intuitionistic higher-order arithmetic, is built up from three parts: (i) axioms and rules for first order many-sorted intuitionistic predicate logic; (ii) equality axioms and the axiom schema of induction; (iii) lambda calculus axioms and rules and the defining equations of the constants R_τ, $\mathsf{R}_\tau xy\mathsf{0} = x$, $\mathsf{R}_\tau xy(\mathsf{s}z) = yz(\mathsf{R}_\tau xyz)$. Thus our theory HA^ω essentially coincides with HA^ω from [23], the only difference being that we consider $\vee$ as defined and use the lambda version instead of the combinator version. The theory HA^ω_c is HA^ω with classical logic; HA^ω_-, *minimal higher-order arithmetic*, is HA^ω without the axiom schema $\bot \Rightarrow \phi$.

1.5 The Axiom of Choice

Theories of classical (intuitionistic) analysis can be obtained from HA^ω_c (HA^ω) by adding the Axiom of Choice. The Axiom of Choice of types τ, τ', denoted by $\mathsf{AC}(\tau, \tau')$, is the axiom schema

$$[\forall x : \tau \; \exists y : \tau' \; \phi(x, y)] \Rightarrow \exists f : \tau \to \tau' \; \forall x : \tau \; \phi(x, fx)$$

(schematic in formula ϕ). Here we will mainly consider $\mathsf{AC}(\mathsf{N}, \tau)$ (schematic in τ), that we may abbreviate to AC. The axiom AC is sufficiently strong to formalize a large part of classical analysis. Intuitionistically, AC is not a strong axiom, as may be expected from the Brouwer–Heyting–Kolmogorov interpretation of a $\forall\exists$ prefix. More formally, it follows from results of Goodman [7] that adding $\mathsf{AC}(\mathsf{N}, \mathsf{N})$ and $\mathsf{AC}(\mathsf{N}, \mathsf{N} \to \mathsf{N})$ to HA^ω is conservative over Heyting Arithmetic.

1.6 A negative interpretation

We will use the notation $\nabla \; \phi$ for $\neg\neg\phi$. This notation is justified by the fact that $\neg\neg$ can be thought of as a modal operator on formulae; we can prove indeed that $\nabla \; \phi$ follows from ϕ, and $\nabla \; \psi$ from $\nabla \; \phi$ and $\phi \Rightarrow \nabla \; \psi$.

Since absurdity is not interpreted by the empty type, we cannot realize $\bot \Rightarrow \phi$ for all ϕ. We overcome this problem by exercising some care in the negative interpretation. The idea is essentially due to Kolmogorov [13]. We employ the fact that $\bot \Rightarrow \nabla\ \phi$ can be proved for all ϕ without using the axiom schema $\bot \Rightarrow \phi$. Although our prime formulae are decidable, the negative interpretation of a prime formula ϕ will be $\nabla\ \phi$.

As negative interpretation we use a standard version of the double negation translation, i.e. prefixing prime formulae and $\exists$ by ∇. The negative interpretation of a formula ϕ, denoted by $\mathsf{G}\ \phi$, is inductively defined by (here and below, $\equiv$ denotes syntactical identity):

- $\mathsf{G}\ \bot \equiv \bot$
- $\mathsf{G}\ \phi \equiv \nabla\ \phi$ if ϕ is a prime formula
- $\mathsf{G}\ [\phi \Rightarrow \psi] \equiv [\mathsf{G}\ \phi] \Rightarrow \mathsf{G}\ \psi$
- $\mathsf{G}\ [\phi \wedge \psi] \equiv \mathsf{G}\ \phi \wedge \mathsf{G}\ \psi$
- $\mathsf{G}\ \forall x : \tau\ \phi \equiv \forall x : \tau\ \mathsf{G}\ \phi$
- $\mathsf{G}\ \exists x : \tau\ \phi \equiv \nabla\ \exists x : \tau\ \mathsf{G}\ \phi$

The negative interpretation satisfies the following preservation property: if ϕ is provable in HA^ω_c, then $\mathsf{G}\ \phi$ is provable in HA^ω_-. In the presence of AC, one cannot expect such a simple preservation result since, as we shall see below, AC is classically much stronger than intuitionistically.

The theory $\mathsf{HA}^\omega_- + \mathsf{G}\ \mathsf{AC}$ will be called *negative analysis*. We need the following technical results.

1.1. Lemma. *The following schemata are provable in* HA^ω_-*:*

(i) $\bot \Rightarrow \nabla\ \phi$*;*

(ii) $\neg\nabla\ \phi \iff \neg\phi$*;*

(iii) $\phi \Rightarrow \nabla\ \phi$*;*

(iv) $\mathsf{G}\ \phi \iff \nabla\ \mathsf{G}\ \phi$*;*

(v) $\mathsf{G}\ \exists x : \tau\ \phi \iff \nabla\ \exists x : \tau\ \phi$*, if* ϕ *prime formula;*

(vi) $\phi \Rightarrow \mathsf{G}\ \phi$*, if* ϕ *formula not containing* $\Rightarrow$*;*

(vii) $\mathsf{G}\ \neg\phi \iff \neg\mathsf{G}\ \phi$*.*

Proof. The clauses (i)-(iii) and (vi),(vii) are trivial. For (iv), observe that the standard proof of this fact by formula induction does not use the axiom schema $\bot \Rightarrow \phi$. For all ϕ we have $\nabla\ \exists x : \tau\ \nabla\ \phi \iff \nabla\ \exists x : \tau\ \phi$ via $\neg\exists x : \tau\ \phi \iff \forall x : \tau\ \neg\phi$ using (ii), so we have (v). $\square$

The negative interpretation of AC, $\mathsf{G}\ \mathsf{AC}$, reads:

$$[\forall x : \mathsf{N}\ \nabla\ \exists y : \tau\ \mathsf{G}\ \phi(x, y)] \Rightarrow \nabla\ \exists f : \mathsf{N} \to \tau\ \forall x : \mathsf{N}\ \mathsf{G}\ \phi(x, fx).$$

By the stability[1] of formulae after the negative interpretation, which follows from Lemma 1.1 (iv), $\mathsf{G}\ \mathsf{AC}$ is subsumed by the following axiom schema:

$$[\forall x : \mathsf{N}\ \nabla\ \exists y : \tau\ \neg\phi(x, y)] \Rightarrow \nabla\ \exists f : \mathsf{N} \to \tau\ \forall x : \mathsf{N}\ \neg\phi(x, fx)$$

[1] A formula is stable iff it is equivalent to the negation of another formula.

Par abus de langage we will from now on denote this schema by $\mathsf{G\,AC}$.

We can now extend the preservation property above to: if ϕ is provable in classical analysis, then $\mathsf{G}\ \phi$ is provable in negative analysis. The proof goes by inspection of the standard preservation proof, taking care that $\bot\Rightarrow\phi$ is avoided using the lemma above.

2 Classical Logic and the Axiom of Choice

In this section, we enumerate some known difficulties in giving a constructive interpretation of the Axiom of Choice in the presence of classical logic. The main remark is that $\mathsf{G\,AC}$ fails to be an intuitionistic consequence of the Axiom of Choice[2]. We point out three independent reasons:

- $\mathsf{G\,AC}$ makes problematic the constructive interpretation of the notion of function; actually, it refutes Church's Thesis;
- $\mathsf{G\,AC}$ is proof theoretically very strong, we can actually interpret by $\mathsf{G\,AC}$ impredicative second-order comprehension, so that one cannot give any predicative interpretation of $\mathsf{G\,AC}$;
- a standard computational interpretation of classical arithmetic, the one using infinitary propositional calculus [17, 21], when extended to quantification over functions, fails to interpret the Axiom of Choice.

2.1 $\mathsf{G\,AC}$ refutes Church's Thesis

In the classical system $\mathsf{HA}^\omega_c+\mathsf{AC}$, we may define the characteristic function of any predicate. Indeed, for any statement $\phi(x)$, we have

$$\exists f\ \forall x\ [\phi(x)\iff [f(x)=0]]$$

since, by the Excluded Middle,

$$\forall x\ \exists y\ [\phi(x)\iff [y=0]].$$

If we take for ϕ some non-decidable formula, then we may prove in this way the existence of non-recursive functions in $\mathsf{HA}^\omega_c+\mathsf{AC}$.

In other words, we refute in $\mathsf{HA}^\omega_c+\mathsf{AC}$ the formula

$$\mathsf{CT}\equiv\forall f:\mathsf{N}\to\mathsf{N}\ \exists e:\mathsf{N}\ \forall x:\mathsf{N}\ \exists y:\mathsf{N}\ [T(e,x,y)\wedge U(y)=f(x)]$$

where $T(e,x,y)$ is the standard Kleene predicate (which can be formulated as an equation) and $U(y)=z$ means that the output of the computation with code y is z. This formula CT expresses Church's Thesis in an intuitionistic way, stating that every function f has a recursive code e.

This simple remark has some deep consequences. Since CT is a formula not containing $\Rightarrow$, it implies its negative translation $\mathsf{G\,CT}$ by lemma 1.1 (vi). We can thus prove $\bot$ in the system $\mathsf{HA}^\omega_-+\mathsf{G\,AC}+\mathsf{CT}$. In other words, if we assume $\mathsf{G\,AC}$, we can prove in HA^ω_- that not all functions are recursive (even though we cannot exhibit a counterexample!).

A direct corollary is that there cannot be any recursive realization [12, 24] of $\mathsf{G\,AC}$. For exactly this reason, and because the semantics of the system NuPrl is based on recursive realizability, the work [16, 4] restricts itself to a fragment of classical logic that does not include the Axiom of Choice.

[2]This is to be contrasted with the induction schema over integers, whose negative interpretation is an instance of the induction schema itself.

This often forces one to encode functions as relations, and this encoding may be unnatural (see the discussion of Higman's lemma in [16]).

It is interesting to analyse in what way our interpretation avoids this difficulty. The main obstacle for recursive realizability is the equivalence between the realizability of ϕ and the realizability of $\nabla\ \phi$, which comes from the fact that the absurd proposition has no realizers. Our interpretation introduces potential "realizers" for the absurd proposition, and because of this, $\nabla\ \phi$ can be realizable though ϕ is not realizable. What is hard to understand intuitively is why taking into account possible "absurd situations" (that will actually never happen) has such an effect.

2.2 Proof Theoretic Strength of G AC

A second difficulty in interpreting G AC comes from the fact that in intuitionistic logic, G AC is proof-theoretically much stronger than AC.

We have seen already that $\mathsf{HA}^\omega + \mathsf{AC}$ is conservative over HA [7]. In contrast, by using the fact that in $\mathsf{HA}^\omega_c + \mathsf{AC}$ we may define the characteristic function of any predicate, it is not difficult to see that we can interpret the second-order Comprehension Axiom in $\mathsf{HA}^\omega_- + \mathsf{G\ AC}$. This system is proof theoretically much stronger than HA (see, for instance, [20], Section 8.5), which implies that there cannot be any predicative way of modeling $\mathsf{HA}^\omega_- + \mathsf{G\ AC}$.

These remarks may explain a comment of Gödel after presenting the negative interpretation [6]: "Intuitionism appears to introduce genuine restrictions only for analysis and set theory; these restrictions, however, are due to the rejection, not of the principle of Excluded Middle, but of notions introduced by impredicative definitions."

It can be shown that the negative interpretation extends to impredicative calculi, and by Gödel's own interpretation of the Axiom of Choice by constructible sets, it is even possible to make sense in this way of the full higher-order arithmetic with the Excluded Middle and the Axiom of Choice in an "intuitionistic" impredicative higher-order arithmetic. (See the introductory note to 1933*e* in [6]).

An important aspect that is not covered in this comment however is a more "intensional" aspect of logical systems, independent of their proof-theoretic strength. From an intuitionistic viewpoint, the proof object, that counts as evidence for the truth of a proposition, is of primary importance.

2.3 Problem in sequent calculus

Let us analyse now another way to get a possible computational content from classical arguments by considering Gentzen's sequent calculus formulation of classical arithmetic. One elegant approach is to use derivations in infinitary propositional logic [17, 21]. There is a general cut-elimination result for this logic which gives a computational interpretation of proofs. If we introduce countable disjunction and conjunction, we can interpret in this way classical quantification over natural numbers and give a computational interpretation of proofs in classical arithmetic.

A natural attempt is then to introduce disjunction and conjunction over the set of all number theoretic functions, and hope that we can interpret in this way classical quantification over number theoretic functions. However, as noticed in Tait's paper [21], there is a natural formulation of the Axiom of Choice in infinitary propositional logic. We start from atoms ϕ_{mn}, $\overline{\phi_{mn}}$, and axioms

$$\Gamma, \phi_{mn}, \overline{\phi_{mn}}$$

and we ask whether it is possible to build a cut-free proof, uniformly in ϕ, of

$$\bigvee_m \bigwedge_n \overline{\phi_{mn}},\ \bigvee_f \bigwedge_m \phi_{mf(m)}, \tag{1}$$

from these axioms. It is not difficult to see that there cannot be such a cut-free proof. We will explain this in some detail.[3] Assume by contradiction Δ is a cut-free proof of (1).

First, observe that all conjunctions in (1) are countable, so by the subformula property all conjunctions in Δ are countable, so that Δ is at most countably branching. Since Δ is also well founded, it follows that Δ is countable and that at most countably many functions $f_0, f_1, \ldots$ appear in Δ.

Second, replace every subformula of the form $\bigvee_f \ \phi(f)$ in Δ by $\bigvee_k \ \phi(f_k)$ and call the resulting tree Δ'. Observe that $\bigvee_f \ \phi(f)$ can only be introduced in Δ from some $\phi(f_k)$. It follows that Δ' is indeed a proof of

$$\bigvee_m \bigwedge_n \overline{\phi_{mn}}, \ \bigvee_k \bigwedge_m \ \phi_{mf_k(m)}.$$

Third, the above conclusion of Δ' is false. This can be seen by taking f to be a function diagonalizing over $f_0, f_1, \ldots$, e.g. $f(n) = f_n(n)+1$, and $\phi_{mn} \iff f(m) = n$. Indeed, $\bigvee_m \bigwedge_n \overline{\phi_{mn}}$ is false since f is total, and $\bigvee_k \bigwedge_m \ \phi_{mf_k(m)}$ is false since f is by construction different from all f_k's.

3 A realizability interpretation

3.1 General description

Realizability, due to Kleene, aims at formalizing the notion of constructive truth, see [24] for an overview. A realizability interpretation interprets a logic, usually an extension of Heyting Arithmetic, in a programming language. More specifically, to each formula ϕ of the logic is associated a type $|\phi|$ of the programming language. One then defines by formula induction when a program of type $|\phi|$ realizes the formula ϕ. Intuitively, it means that this program is a constructive justification of the formula ϕ. To each proof of a closed formula ϕ is associated a program of type $|\phi|$ which realizes the formula ϕ.

3.2 The programming language $\mathcal{P}$

The programming language $\mathcal{P}$ extends the types, terms and equations of HA^ω with type constants for interpreting $\bot$ and equations, type constructors for lists and pairs, term constants associated to the new types, general recursion and infinite terms to form choice sequences.

3.2.1 Types of $\mathcal{P}$

The types of $\mathcal{P}$ are $\mathsf{N}, \mathsf{Unit}, \mathsf{Abs}$ and with τ, τ' also $\tau \to \tau'$, $\tau \times \tau'$ (cartesian product) and $[\tau]$ (lists over type τ). The type Unit serves to interpret prime formulae, and the type Abs to interpret $\bot$. The type Abs will not be empty. Like in the case of HA^ω, types will be denoted by lower case greek letters $\tau, \tau', \ldots$. The types of HA^ω will be called N-types.

3.2.2 Terms of $\mathcal{P}$

The terms of $\mathcal{P}$ are built from (typed) variables and constants using lambda abstraction, (well typed) application, and the formation of infinite terms: if $M_0, M_1, \ldots$ is an infinite sequence of terms of type τ, then $(\lambda\!\!\lambda x.M_x)$ is a term of type $\mathsf{N} \to \tau$. (The infinite terms are *not* for computational purposes, they only play a role in the termination proof.) There are countably many variables

[3] Cf. [21], Tait describes this result as known already, but does not give further references.

$x, y, z, \ldots$ for each type τ. The set of constants of $\mathcal{P}$ extends that of HA^ω with constants R_τ for types τ that are not N-types, $() : \mathsf{Unit}$, $\mathsf{Dummy} : \mathsf{Abs}$, $\mathsf{Axiom}_1, \mathsf{Axiom}_2 : \mathsf{N}{\to}\mathsf{Abs}$, constants for general recursion (fixpoint combinators of all appropriate types) and constants for pairing and projection and list construction and destruction. Terms are again denoted by $M, M', N, \ldots$, and $M : \tau$ expresses that the term M has type τ.

3.2.3 Equations of $\mathcal{P}$

The only formulae of $\mathcal{P}$ are equations of the form $M = M'$, with M, M' terms of $\mathcal{P}$ of the same type, not necessarily N.

3.2.4 Theory of $\mathcal{P}$

The theory of $\mathcal{P}$ is equational, built up from the usual lambda calculus axioms and rules, defining equations for the constants R_τ as in HA^ω, but now for all types τ of $\mathcal{P}$, pairing axioms and list axioms and axioms for general recursion as usual, and the following axiom schema for infinite terms:

$$(\beta\!\!\beta) \qquad (\lambda\!\!\lambda x.M_x)\underline{k} = M_k$$

(schematic in τ, $M_0, M_1, \ldots$ of type τ and natural number k).

3.2.5 Pragmatics of $\mathcal{P}$

We allow ourselves a liberal use of $\mathcal{P}$. We will assume that all terms are well typed and we will reduce type information to a minimum that is required to reconstruct the type of a well typed term from the context. For every natural number k, we abbreviate $\mathsf{s}(\ldots(\mathsf{s}\ \mathsf{0})\ldots)$ (with k times s) by $\underline{k}$, called the numeral $\underline{k}$. The term $\underline{k}_\tau$ of N-type τ is defined inductively by: $\underline{k}_\mathsf{N} \equiv \underline{k}$; $\underline{k}_{\tau\to\tau'} \equiv \lambda x : \tau.\underline{k}_{\tau'}$. We write pairs as (M, N) and triples as (M, N, P) instead of $(M, (N, P))$. We even write $\lambda(x, y).M$ instead of $\lambda z : \tau{\times}\tau'.M'$, where M' is obtained from M by replacing x by the first projection of z and y by the second. Lists are denoted by $[M_1, \ldots, M_n]$ ($M_i : \tau$ for $1 \le i \le n$). Adding a term H at the beginning of a list L will be denoted by $H{:}L$.

Instead of the explicit use of fixpoint combinators we define terms by giving the recursion equations. As an example we define a term $\mathsf{get} : \mathsf{N}{\to}[\mathsf{N}{\times}\tau{\times}\tau']{\to}(\tau{\to}\tau'{\to}\tau''){\to}\tau''{\to}\tau''$, which will play a role in the sequel. The term $(\mathsf{get}\ x\ l\ f\ a)$ searches the list l for the first triple whose first component matches x; if such a triple is found, then the output is f applied to the second and third component of the triple, otherwise the output is a. Formally,

$$\begin{array}{lcl} \mathsf{get}\ x\ []\ f\ a & = & a \\ \mathsf{get}\ x\ ((x', y, y'){:}l)\ f\ a & = & \mathsf{if}\ (x = x')\ \mathsf{then}\ (f\ y\ y')\ \mathsf{else}\ (\mathsf{get}\ x\ l\ f\ a) \end{array}$$

Here and below $\mathsf{if}\ (M = M')\ \mathsf{then}\ldots\mathsf{else}\ldots$ (with M, M' of type N) is a sugared version of a well-known primitive recursive term.

3.2.6 Known facts about $\mathcal{P}$

There exists a reduction relation $\longrightarrow$ on the terms of $\mathcal{P}$ such that the reflexive, symmetric and transitive closure of $\longrightarrow$ coincides with $=$ (convertibility) on the terms of $\mathcal{P}$. Moreover, $\longrightarrow$ satisfies:

(i) the Church Rosser Theorem;
(ii) every closed normal form of type N is a numeral $\underline{k}$;
(iii) every closed normal form of type Unit is $()$;
(iv) every closed normal form of type Abs is either Dummy or of the form $\mathsf{Axiom}_1\underline{k}$ or $\mathsf{Axiom}_2\underline{k}$;
(v) the Continuity Lemma: let $M : (\mathsf{N}\to\tau)\to\tau'$ and $N : \mathsf{N}\to\tau$ be such that MN has a closed normal form. Then there exists a natural number m such that for all $N' : \mathsf{N}\to\tau$ with $N\underline{i} = N'\underline{i}$ for all $i < m$ we have $MN = MN'$. In particular we have extensionality: if $N\underline{i} = N'\underline{i}$ for all natural numbers i, then $MN = MN'$.

3.3 Realizability

In this subsection a realizability interpretation of $\mathsf{HA}^\omega + \mathsf{G\ AC}$ in $\mathcal{P}$ will be given. It consists of a mapping of formulas of HA^ω to types of $\mathcal{P}$ together with a realizability relation between programs in $\mathcal{P}$ and formulas of HA^ω, where the program has the type to which the formula is mapped. The main result will be that every theorem of $\mathsf{HA}^\omega + \mathsf{G\ AC}$ can be realized in $\mathcal{P}$. The difficult step in proving this result is the realization of $\mathsf{G\ AC}$.

3.3.1 Mapping formulas of HA^ω to types of $\mathcal{P}$

The idea behind this mapping is usually referred to as "forgetting dependencies", due to Martin-Löf. By formula induction we define a type $|\phi|$ of $\mathcal{P}$ for every formula ϕ of HA^ω:

- $|M = M'| \equiv \mathsf{Unit}$
- $|\ \bot\ | \equiv \mathsf{Abs}$
- $|\phi \Rightarrow \psi| \equiv |\phi|\to|\psi|$
- $|\phi \wedge \psi| \equiv |\phi|\times|\psi|$
- $|\forall x : \tau\ \ \phi| \equiv \tau\to|\phi|$
- $|\exists x : \tau\ \ \phi| \equiv \tau\times|\phi|$

Note that the domains of quantification in HA^ω are types of HA^ω, and hence of $\mathcal{P}$, so that the mapping $|\ \ |$ is well defined.

3.3.2 Reducibility in $\mathcal{P}$

In order to define the realizability relation we need a notion of reducibility for closed terms of $\mathcal{P}$ of N-type. By induction on the N-type we define:

- $M : \mathsf{N}$ is reducible iff M reduces to a numeral
- $M : \tau\to\tau'$ is reducible iff MN is reducible for every reducible $N : \tau$

In the sequel, we shall need the following technicalities.

3.1. Definition. Two expressions E and E' (terms or formulae) are called *related* if they are syntactically identical up to the indices of the constants Axiom_i. Note that related terms are of the same type.

3.2. LEMMA. *If M and M' are two related terms of type* N*, then $M = \underline{n}$ iff $M' = \underline{n}$. If M and M' are two related terms of type* Unit*, then $M = ()$ iff $M' = ()$.*

PROOF. By induction on the length of reduction sequences. □

3.3. LEMMA. *If M and M' are two related terms of* N*-type, then M is reducible iff M' is reducible.*

PROOF. By an easy induction on the common N-type of M and M', using Lemma 3.2 for the base case N. □

3.3.3 Realizability relation

What follows is essentially the notion of modified realizability, due to Kreisel, with realizing objects from the programming language $\mathcal{P}$. We give an inductive definition of "M realizes ϕ", where ϕ is a closed formula of HA^ω with possibly closed reducible terms of $\mathcal{P}$ occurring in the prime constituents of ϕ, and $M : |\phi|$ a closed term of $\mathcal{P}$.

$M : \mathsf{Abs}$ realizes $\bot$	iff	$M = \mathsf{Axiom}_i \underline{k}$ for some $i = 1, 2$ and numeral $\underline{k}$
$M : \mathsf{Unit}$ realizes $M_1 = M_2$	iff	$M = ()$ and M_1 and M_2 reduce in $\mathcal{P}$ to the same numeral
$M : \lvert\phi\rvert \to \lvert\psi\rvert$ realizes $\phi \Rightarrow \psi$	iff	MN realizes ψ whenever N realizes ϕ
$M : \lvert\phi \wedge \psi\rvert$ realizes $\phi \wedge \psi$	iff	$M = (N, P)$ with N realizes ϕ and P realizes ψ
$M : \tau \to \lvert\phi\rvert$ realizes $\forall x : \tau\ \phi$	iff	MN realizes $\phi[x := N]$ for all *reducible* $N : \tau$
$M : \tau \times \lvert\phi\rvert$ realizes $\exists x : \tau\ \phi$	iff	$M = (N, P)$ with $N : \tau$ *reducible* and P realizes $\phi[x := N]$

Note that the above definition uses reducibility for N-types only. In the case $M_1 = M_2$ above, the equation is of type N. The terms M_1 and M_2 come from (possibly) open terms of HA^ω in which closed reducible terms of $\mathcal{P}$ are substituted for the variables. Thus M_1 and M_2 are closed and reducible, since all constants of HA^ω are reducible constants of $\mathcal{P}$. Hence we can verify $M_1 = M_2$ in $\mathcal{P}$, relying on Fact (i) and (ii) from 3.2.6.

In the sequel, we shall need the following technical lemma.

3.4. LEMMA. *If ϕ and ϕ' are two related formulae, and M, M' two related terms, then M realizes ϕ iff M' realizes ϕ'.*

PROOF. By an easy induction on the realization relation, using Lemma 3.2 and Lemma 3.3. □

3.4 Main Result and applications

In this subsection we formulate the main result, sketch a proof and give two applications of the main result. The essential and difficult step in the proof of the main result, the realization of $\mathsf{G\,AC}$ is given in the next subsection.

3.5. THEOREM. *Every theorem of $\mathsf{HA}^\omega_- + \mathsf{G\,AC}$ can be realized by a term in $\mathcal{P}$ in which no constants Axiom_i occur ($i = 1, 2$).*

PROOF. Apart from the realization of $\mathsf{G\,AC}$, the proof is more or less standard. For example, the axiom $\forall x : \mathsf{N}\ \neg(\mathsf{s}x = \mathsf{0})$ is realized by $M \equiv \lambda x : \mathsf{N}\ \lambda h : \mathsf{Unit}.\mathsf{Dummy}$. Indeed, $M\underline{n}$ realizes $\neg(\mathsf{s}\underline{n} = \mathsf{0})$ for every natural number n, since nothing realizes $\mathsf{s}\underline{n} = \mathsf{0}$ (here we use Fact (i) and (ii) from 3.2.6). □

3.4.1 Application 1: the consistency of analysis

The main result immediately implies the consistency of analysis. Assume $\bot$ is provable in $\mathsf{HA}^\omega_c + \mathsf{AC}$. Then $\mathsf{G}\ \bot$, i.e. $\bot$, is provable in $\mathsf{HA}^\omega_- + \mathsf{G\ AC}$. Hence $\bot$ is realizable by a term of $\mathcal{P}$ in which no constants Axiom_i occur $(i = 1, 2)$. This is impossible by the definition of realization.

3.4.2 Application 2: how to compute witnesses with AC and classical logic

Assume a formula $\phi(x)$ of HA^ω, with $x : \mathsf{N}$, is decidable, i.e. of the form $M_\phi x = \mathsf{0}$ for suitable closed term M_ϕ of HA^ω. We will freely write $\phi(x)$ instead of $M_\phi x = \mathsf{0}$. Assume $\exists x : \mathsf{N}\ \phi(x)$ is a theorem of $\mathsf{HA}^\omega_c + \mathsf{AC}$. Then $\nabla\ \exists x : \mathsf{N}\ \mathsf{G}\ \phi(x)$ is a theorem of $\mathsf{HA}^\omega_- + \mathsf{G\ AC}$. By Lemma 1.1 (v) we have that $\nabla\ \exists x : \mathsf{N}\ \phi(x)$ is a theorem of $\mathsf{HA}^\omega_- + \mathsf{G\ AC}$, and hence realizable by a term not containing constants Axiom_i $(i = 1, 2)$, say by M. We have that

$$N \equiv \lambda(x, h) : \mathsf{N}\times\mathsf{Unit.if}\ \phi(x)\ \mathsf{then}\ (\mathsf{Axiom}_1 x)\ \mathsf{else\ Dummy}$$

realizes $\neg\exists x : \mathsf{N}\ \phi(x)$. So MN realizes $\bot$ and must hence be convertible to $\mathsf{Axiom}_1\underline{n}$ for some numeral $\underline{n}$. We claim $\phi(\underline{n})$, i.e. $\underline{n}$ is a witness. Consider the following extensionally equal terms:

$$\begin{aligned} F &\equiv \lambda x : \mathsf{N}\lambda h : \mathsf{Unit.if}\ \phi(x)\ \mathsf{then}\ (\mathsf{Axiom}_1 x)\ \mathsf{else\ Dummy} \\ F' &\equiv \lambda x : \mathsf{N}\lambda h : \mathsf{Unit.if}\ \phi(x)\ \mathsf{then}\ (\mathsf{if}\ \phi(x)\ \mathsf{then}\ (\mathsf{Axiom}_1 x)\ \mathsf{else\ Dummy})\ \mathsf{else\ Dummy} \end{aligned}$$

Since F and F' are extensionally equal we have by extensionality:

$$\mathsf{Axiom}_1\underline{n} = MN = M(\lambda(x, h).Fxh) = M(\lambda(x, h).F'xh)$$

Note that F' can be obtained from F by replacing Axiom_1 by $\lambda x.\mathsf{if}\ \phi(x)\ \mathsf{then}\ (\mathsf{Axiom}_1 x)\ \mathsf{else\ Dummy}$. Since M does not contain the constant Axiom_1, it follows that

$$\mathsf{Axiom}_1\underline{n} = \mathsf{if}\ \phi(\underline{n})\ \mathsf{then}\ (\mathsf{Axiom}_1\underline{n})\ \mathsf{else\ Dummy}.$$

Using Fact (iv) from 3.2.6 we get $\phi(\underline{n})$.

3.5 Realization of G AC

Recall that G AC is the following schema:

$$[\forall x : \mathsf{N}\ \nabla\ \exists y : \tau\ \neg\phi(x, y)] \Rightarrow \nabla\ \exists f : \mathsf{N}\to\tau\ \forall x : \mathsf{N}\ \neg\phi(x, fx)$$

We start with some preliminary calculations:

$$|\forall x : \mathsf{N}\ \nabla\ \exists y : \tau\ \neg\phi(x, y)| \equiv \mathsf{N}\to((\tau\times(|\phi|\to\mathsf{Abs}))\to\mathsf{Abs})\to\mathsf{Abs}$$

$$|\neg\exists f : \mathsf{N}\to\tau\ \forall x : \mathsf{N}\ \neg\phi(x, fx)| \equiv ((\mathsf{N}\to\tau)\times(\mathsf{N}\to|\phi|\to\mathsf{Abs}))\to\mathsf{Abs}$$

A realizer of G AC should be a term M such that MHP realizes $\bot$ whenever H realizes $\forall x : \mathsf{N}\ \nabla\ \exists y : \tau\ \neg\phi(x, y)$ and P realizes $\neg\exists f : \mathsf{N}\to\tau\ \forall x : \mathsf{N}\ \neg\phi(x, fx)$. Moreover, M should not contain constants Axiom_i $(i = 1, 2)$. The general idea is to approximate a function witnessing $\exists f : \mathsf{N}\to\tau\ \forall x : \mathsf{N}\ \neg\phi(x, fx)$ by means of a list L of triples (X, Y, Z), where $X : \mathsf{N}$ and $Y : \tau$ are reducible, and Z realizes $\neg\phi(X, Y)$. Given such a list $L = [(X_1, Y_1, Z_1), \ldots, (X_n, Y_n, Z_n)]$ with all the X_i's distinct, we consider a function $\mathsf{fun}\ L : \mathsf{N}\to\tau$ which maps X_i to Y_i $(1 \le i \le n)$ and takes function values $\underline{0}_\tau$ in arguments different from all X_i's. Formally:

$$\mathsf{fun}\ l\ x = \mathsf{get}\ x\ l\ (\lambda y : \tau\lambda z : |\phi|\to\mathsf{Abs}.y)\ \underline{0}_\tau$$

Furthermore, we consider a function $\lambda x : \mathsf{N}.\mathsf{rea}\ L\ x\ A : \mathsf{N}{\to}|\phi|{\to}\mathsf{Abs}$ which maps X_i to the realizer Z_i $(1 \leq i \leq n)$ and takes values A (to be specified below) in arguments different from all X_i's. Formally:

$$\mathsf{rea}\ l\ x\ a = \mathsf{get}\ x\ l\ (\lambda y : \tau \lambda z : |\phi|{\to}\mathsf{Abs}.z)\ a$$

Consider

$$P(\mathsf{fun}\ L, \lambda x : \mathsf{N}.\mathsf{rea}\ L\ x\ A).$$

If $\lambda x : \mathsf{N}.\mathsf{rea}\ L\ x\ A$ realizes $\forall x\ \neg\phi(x, \mathsf{fun}\ L\ x)$, then we would have that $P(\mathsf{fun}\ L, \lambda x : \mathsf{N}.\mathsf{rea}\ L\ x\ A)$ realizes $\bot$ and we would be done. However, this is in general not the case since we cannot choose A such that A realizes $\neg\phi(x, \underline{0}_\tau)$ for all x different from all X_i's. We claim that, as A may depend on x, there is a possibility to construct A in such a way that it allows us to compute a better approximation of the function witnessing $\exists f : \mathsf{N}{\to}\tau\ \forall x : \mathsf{N}\ \neg\phi(x, fx)$. The type of A must be $|\phi|{\to}\mathsf{Abs}$, so we must have

$$A \equiv \lambda x' : |\phi|.\cdots,$$

where $\cdots$ is of type Abs. It is tempting to fill in $\cdots$ with $(\mathsf{Axiom}_i x)$. The resulting term

$$P(\mathsf{fun}\ L, \lambda x.\mathsf{rea}\ L\ x\ (\lambda x' : |\phi|.\mathsf{Axiom}_i x))$$

of type Abs is not a solution, since it contains Axiom_i, but it will play an important role in the discussion below. Note that $\lambda x : \mathsf{N}.\mathsf{rea}\ L\ x\ A$ only accesses A in case x does not occur as first component of a triple in L. The basic intuition is that, if the above term reduces to $\mathsf{Axiom}_i \underline{k}$, then this tells us that P needs more information about its arguments, in particular it needs a function value and a realizer for the argument k.

Observe that, so far, H realizing $\forall x : \mathsf{N}\ \nabla\ \exists y : \tau\ \neg\phi(x, y)$ has not been used. For filling in $\cdots$ we use H. Recall that the type of H is $\mathsf{N}{\to}((\tau{\times}(|\phi|{\to}\mathsf{Abs})){\to}\mathsf{Abs}){\to}\mathsf{Abs}$. The obvious way to continue is putting

$$A \equiv \lambda x' : |\phi|.H\ x\ \cdots.$$

Now $\cdots$ is of type $(\tau{\times}(|\phi|{\to}\mathsf{Abs})){\to}\mathsf{Abs}$ and is hence of the form $\lambda(y, z).\cdots$, so that we have

$$A \equiv \lambda x' : |\phi|.H\ x\ (\lambda(y, z).\cdots),$$

where $\cdots$ is again of type Abs. The crucial idea is now to put

$$A \equiv \lambda x' : |\phi|.H\ x\ (\lambda(y, z).\cdots((x, y, z){:}L)),$$

where $\cdots$ stands for a recursive call of the whole procedure described above.

This informal discussion motivates the following recursive definition:

$$\Phi\ p\ h\ l\ =\ p\ (\mathsf{fun}\ l, \lambda x.\mathsf{rea}\ l\ x\ (\lambda x'.h\ x\ (\lambda(y, z).\Phi\ p\ h\ ((x, y, z){:}l))))$$

We shall prove that, given H and P as above, $\Phi\ P\ H\ []$ realizes $\bot$. Thus $\lambda h \lambda p.\Phi\ p\ h\ []$ realizes $\mathsf{G}\,\mathsf{AC}$.

The first step will be, in the next lemma, to check that each recursive call to Φ on a "good argument" indeed corresponds to an extension of the list L approximating a witness f such that $\forall x : \mathsf{N}.\neg\phi(x, fx)$.

3.6. Lemma. *Let H, P, Φ be as above. Furthermore, let $L = [(X_1, Y_1, Z_1), \ldots, (X_n, Y_n, Z_n)]$ be such that L does not contain Axiom_1 and, for all $1 \leq i \leq n$, $X_i : \mathsf{N}, Y_i : \tau$ are reducible and $Z_i : |\phi|{\to}\mathsf{Abs}$ realizes $\neg\phi(X_i, Y_i)$. If $\Phi\ P\ H\ L$ does not realize $\bot$, then there exist $X : \mathsf{N}, Y : \tau$ and $Z : |\phi|{\to}\mathsf{Abs}$ not containing Axiom_1 such that:*

(i) X *and* Y *are reducible;*

(ii) X *is different from all* X_i*'s;*

(iii) Z *realizes* $\neg\phi(X,Y)$*;*

(iv) $\Phi\ P\ H\ ((X,Y,Z){:}L)$ *does not realize* $\bot$*.*

PROOF. Let conditions be as above. By Lemma 3.4 we may assume that Axiom_1 does not occur in P, H. Recall

$$\Phi\ P\ H\ L\ =\ P\ (\mathsf{fun}\ L, \lambda x.\mathsf{rea}\ L\ x\ (\lambda x'.H\ x\ (\lambda(y,z).\Phi\ P\ H\ ((x,y,z){:}L)))).$$

Let $\Phi'\ P\ H\ L$ be the term discussed above (not depending on H):

$$\Phi'\ P\ H\ L\ =\ P(\mathsf{fun}\ L, \lambda x.\mathsf{rea}\ L\ x\ (\lambda x' : |\phi|.\mathsf{Axiom}_1 x)).$$

By our assumption on L we have that

$$\lambda x : \mathsf{N}.\mathsf{rea}\ L\ x\ (\lambda x' : |\phi|.\mathsf{Axiom}_1 x)$$

realizes $\forall x : \mathsf{N}.\neg\phi(x, \mathsf{fun}\ L\ x)$. Consequently, $\Phi'\ P\ H\ L$ realizes $\bot$, and hence reduces to $\mathsf{Axiom}_i\underline{k}$ for some numeral $\underline{k}$ and $i = 1, 2$. Since the only occurrence of Axiom_1 in $(\Phi'\ P\ H\ L)$ is the one which is explicitly shown, we have that

$$\Phi\ P\ H\ L\ =\ (\Phi'\ P\ H\ L)[\mathsf{Axiom}_1 := \lambda x.H\ x\ (\lambda(y,z).\Phi\ P\ H\ ((x,y,z){:}L))]$$

By our assumption that $\Phi\ P\ H\ L$ does not realize $\bot$, it follows that $i = 1$, and hence

$$\begin{aligned}\Phi\ P\ H\ L\ &=\ (\mathsf{Axiom}_1\underline{k})[\mathsf{Axiom}_1 := \lambda x.H\ x\ (\lambda(y,z).\Phi\ P\ H\ ((x,y,z){:}L))]\\ &=\ H\ \underline{k}\ (\lambda(y,z).\Phi\ P\ H\ ((\underline{k},y,z){:}L)).\end{aligned}$$

Since H realizes $\forall x : \mathsf{N} \nabla \exists y : \tau\ \neg\phi(x,y)$ and $\Phi\ P\ H\ L$ does not realize $\bot$, it follows that $\lambda(y,z).\Phi\ P\ H\ ((\underline{k},y,z) : L)$ does not realize $\neg\exists y : \tau\ \neg\phi(\underline{k},y)$. By the definition of realizability and using classical logic it follows that there exists a reducible $Y : \tau$ and a $Z : |\phi| \to \mathsf{Abs}$ realizing $\neg\phi(\underline{k},Y)$ such that $\Phi\ P\ H\ ((\underline{k},Y,Z){:}L)$ does not realize $\bot$. By Lemma 3.4 we may assume that Axiom_1 does not occur in Y, Z. Now we are done with (i), (iii) and (iv); it remains to prove (ii). To this end, we reason similarly to the argument used in Application 2 above. Consider the following extensionally equal terms:

$$\begin{aligned}F\ &\equiv\ \lambda x.\mathsf{rea}\ L\ x\ (\lambda x' : |\phi|.\mathsf{Axiom}_1 x)\\ F'\ &\equiv\ \lambda x.\mathsf{rea}\ L\ x\ (\lambda x' : |\phi|.\mathsf{if}\ (\mathsf{member}\ x\ L)\ \mathsf{then\ Dummy\ else}\ (\mathsf{Axiom}_1 x))\end{aligned}$$

Here $(\mathsf{member}\ x\ L)$ tests if x occurs as first component of a triple in L. Since the only occurrence of Axiom_1 in F is the one which is explicitly shown, we have that

$$F'\ =\ F[\mathsf{Axiom}_1 := \lambda x.\mathsf{if}\ (\mathsf{member}\ x\ L)\ \mathsf{then\ Dummy\ else}\ (\mathsf{Axiom}_1 x)]$$

Recall that $P(\mathsf{fun}\ L, F) = \Phi'\ P\ H\ L = \mathsf{Axiom}_1\underline{k}$. By extensionality we have $P(\mathsf{fun}\ L, F') = P(\mathsf{fun}\ L, F) = \mathsf{Axiom}_1\underline{k}$. By substitution we get

$$\begin{aligned}P(\mathsf{fun}\ L, F')\ &=\ (\mathsf{Axiom}_1\underline{k})[\mathsf{Axiom}_1 := \lambda x.\mathsf{if}\ (\mathsf{member}\ x\ L)\ \mathsf{then\ Dummy\ else}\ (\mathsf{Axiom}_1 x)]\\ &=\ \mathsf{if}\ (\mathsf{member}\ \underline{k}\ L)\ \mathsf{then\ Dummy\ else}\ (\mathsf{Axiom}_1\underline{k})\end{aligned}$$

By Fact (i) (Church Rosser Theorem) from 3.2.6 we get $\neg(\mathsf{member}\ \underline{k}\ L)$. This is (ii) and completes the proof of the lemma. □

Using the lemma above we can prove by contradiction that $\Phi\ P\ H\ []$ realizes $\bot$. We give an informal argument, which can easily be formalized using the axiom of Dependent Choice and classical logic. The argument is similar to the argument used by Tait in [22]. Suppose $\Phi\ P\ H\ []$ does not realize $\bot$. Then, by the lemma above, there exist X_1, Y_1, Z_1 such that $[(X_1, Y_1, Z_1)]$ satisfies the conditions of the lemma, in particular $\Phi\ P\ H\ [(X_1, Y_1, Z_1)]$ does not realize $\bot$. Applying the lemma again yields X_2, Y_2, Z_2 such that ... Iterating this argument infinitely many times yields an infinite sequence of triples (X_i, Y_i, Z_i) $(i = 1, 2, \ldots)$ such that each finite initial subsequence satisfies the conditions of the lemma. Note that the X_i's all convert to different numerals. Define (classically!) $(\underline{n}, Y'_n, Z'_n)$ to be (X_i, Y_i, Z_i) if $X_i = \underline{n}$ for some i, and $(\underline{n}, \underline{0}_\tau, \mathsf{Axiom}_1\underline{n})$ otherwise. This results in an infinite sequence $\lambda\!\!\lambda n.(\underline{n}, Y'_n, Z'_n)$ such that $\lambda\!\!\lambda n.Y'_n$ is reducible and $\lambda\!\!\lambda n.Z'_n$ realizes $\forall x : \mathsf{N}\ \neg\phi(x, (\lambda\!\!\lambda n.Y'_n)x)$. It follows that $P(\lambda\!\!\lambda n.Y'_n, \lambda\!\!\lambda n.Z'_n)$ realizes $\bot$ and hence reduces to a term of the form $\mathsf{Axiom}_i\underline{k}$. By Fact (v) from 3.2.6, the Continuity Lemma, $P(\lambda\!\!\lambda n.Y'_n, \lambda\!\!\lambda n.Z'_n)$ only depends on a finite initial subsequence of $\lambda\!\!\lambda n.(\underline{n}, Y'_n, Z'_n)$. This conflicts with the above construction of $[(X_1, Y_1, Z_1), \ldots, (X_n, Y_n, Z_n)]$ for n large enough.

3.6 Comments on this interpretation

This interpretation gives a quite direct computational interpretation of the Axiom of Choice. For instance, it allows to interpret directly the first informal proof of Higman's lemma presented in [16], and avoids in this case the encoding of a function as a relation. Observe furthermore that the computation is demand-driven in the sense that the list of triples only contains function values and realizers that are really needed for the computation of a realizer of $\bot$.

It should be noted also that our interpretation works for $\mathsf{AC}(\tau, \tau')$ where $\tau = \mathsf{N}$ but τ' is arbitrary. As it is given, it uses in an essential way the restriction $\tau = \mathsf{N}$, but we shall present below heuristic arguments that suggest a possible extension to higher types.

Already in the case $\mathsf{AC}(\mathsf{N}, \mathsf{N}{\to}\mathsf{N})$, the translation of a function as a relation used in [16] to avoid the use of $\mathsf{AC}(\mathsf{N}, \mathsf{N})$ does not work any more. The fact that the Axiom of Choice $\mathsf{AC}(\mathsf{N}, \mathsf{N})$ is validated by considering functions as relations indeed uses the existence of a well founded total ordering on N, but we don't know of any such effective ordering on $\mathsf{N}{\to}\mathsf{N}$.

4 Axiom of Dependent Choice

We realize now the negative interpretation of DC_τ, the axiom of Dependent Choice at type τ :

$$[\forall n : \mathsf{N}\ \forall x : \tau\ \exists y : \tau\ \phi(n, x, y)] \Rightarrow \exists f : \mathsf{N}{\to}\tau\ \forall n : \mathsf{N}\ \phi(n, f(n), f(n+1)),$$

which, as explained in [11], implies directly the usual versions of the Axiom of Choice and of the axiom of Dependent Choice. Below it is implicit that x and y are of the same type τ, n of type N and that the function $f : \mathsf{N}{\to}\tau$ is such that $f(\mathsf{0}) = a$, where a is some fixed object of type τ.

The axiom of Dependent Choice is commonly used in informal mathematics. As proved in [11], $\mathsf{AC}(\mathsf{N}, \mathsf{N})$ trivially implies DC_N by primitive recursion. For higher types τ, the axiom of Dependent Choice appears to be stronger than the Axiom of Choice. (In [11], Problem 9 asks whether $\mathsf{AC}(\mathsf{N}, \mathsf{N}{\to}\mathsf{N})$ implies $\mathsf{DC}_{\mathsf{N}\to\mathsf{N}}$, and we do not know if this question is still open.)

The negative interpretation of this schema is subsumed by the following axiom schema $\mathsf{G\ DC}$:

$$[\forall n\ \forall x\ \nabla\ \exists y\ \neg\phi(n, x, y)] \Rightarrow \nabla\ \exists f\ \forall n\ \neg\phi(n, f(n), f(n+1)).$$

We shall give a program that realizes this schema. Furthermore, we prove this fact not using classical logic and dependent choices, but intuitionistic meta-reasoning together with a certain strong principle of bar induction.

This interpretation is based on the same idea of partial approximation of a function. An *approximation* of f satisfying $\forall n\ \neg\phi(n, f(n), f(n+1))$ will be given as a term $u = (n, f_n, l)$ where n is a numeral, f_n a term of type τ and l a list of the form

$$[(0, a, q_0), \ldots, (n-1, f_{n-1}, q_{n-1})]$$

such that q_i is of type $|\neg\phi(i, f_i, f_{i+1})|$ for $i < n$ (with the convention that $f_0 = a$, the fixed object of typed τ). It is now standard to redefine fun and get such that

$$\mathsf{fun}\ u\ i$$

is f_i if $i \leq n$, and 0_τ if $i > n$, and that

$$\mathsf{get}\ u\ i\ b$$

is q_i if $i < n$, and b if $n \leq i$.

If $u = (n, f_n, l)$ is such an approximation, and f is of type τ and q of type $|\neg\phi(n, f_n, f)|$, we write $u + (f, q)$ for the approximation $(n+1, f, (n, f_n, q){:}l)$. We say that v is a *direct extension* of u if v is of the form $u + (f, q)$ for some f, q, and we denote this by $u < v$.

We define recursively, when u is of the form (n, f_n, l)

$$\Phi\ P\ H\ u\ = P\ (\mathsf{fun}\ u, \lambda i.\mathsf{get}\ u\ i\ (\lambda x' : |\phi|.H\ (n+1)\ (\lambda(y,z).\Phi\ P\ H\ (u + (y,z))))).$$

It is then possible to show that $\lambda h\lambda p.\Phi\ p\ h\ (0, a, [])$ realizes $\mathsf{G\ DC}$, by a classical argument similar to the one we gave for the negative interpretation of the Axiom of Choice.

For an intuitionistic reduction to the principle of bar induction, we introduce

$$\Phi'\ P\ H\ u\ =\ P\ (\mathsf{fun}\ u, \lambda i.\mathsf{get}\ u\ i\ (\lambda x' : |\phi|.\mathsf{Axiom}_1 i)).$$

As before, we can suppose that Axiom_1 does not occur in P nor in H.

We define a *reducible* approximation as an approximation (n, f_n, l) such that f_n is a reducible term of type τ and l is a list

$$[(0, a, q_0), \ldots, (n-1, f_{n-1}, q_{n-1})]$$

such that f_i is a reducible term of type τ and q_i realizes $\neg\phi(i, f_i, f_{i+1})$ for $i < n$, with $f_0 = a$. Furthermore we ask that Axiom_1 does not appear neither in f_n nor in l.

Note that, for each reducible approximation u, the term $\Phi'\ P\ H\ u$ realizes $\bot$. Furthermore, by a reasoning similar to the one of the previous section, if $\Phi'\ P\ H\ u$ does not reduce to a term of the form $\mathsf{Axiom}_1\ n_0$, then we have $\Phi\ P\ H\ u\ =\ \Phi'\ P\ H\ u$ and so $\Phi\ P\ H\ u$ realizes $\bot$. A natural terminology is to say that the approximation u is *partial* if $\Phi'\ P\ H\ u$ reduces to a term of the form $\mathsf{Axiom}_1\ n_0$, and *complete* otherwise.

We claim that the set of complete approximations is a bar on the set of reducible approximations. This follows from the fact that any sequence of reducible approximations $u_0 < u_1 < u_2 \ldots$ is eventually complete. That is, there exists n_0 such that u_n is complete if $n_0 \leq n$. This is established using the Continuity Lemma as in the previous section.

It is now direct to prove by bar induction that $\Phi\ P\ H\ u$ realizes $\bot$ if u is a reducible approximation. If $u < v$ implies that $\Phi\ P\ H\ v$ realizes $\bot$, it follows indeed from the definition of Φ that $\Phi\ P\ H\ u$ realizes $\bot$.

5 Double Negation Shift

This section will be informal. We show that a slight variation of our interpretation gives also a computational interpretation of Spector's Double Negation Shift [18].

5.1 Exit operator

We first add a new polymorphic constant to our programming language

$$\mathsf{Exit}_\tau : \mathsf{Abs} \to \tau,$$

with new reduction rules

- $\mathsf{Exit}_{\mathsf{Abs}}\ u \longrightarrow u$,
- $(\mathsf{Exit}_{\tau \to \tau'}\ u)\ v \longrightarrow \mathsf{Exit}_{\tau'}\ u$,
- $\mathsf{R}_\tau\ x\ y\ \mathsf{Exit}_{\mathsf{N}}\ u) \longrightarrow \mathsf{Exit}_\tau\ u$,

and similar rules for destructors for pairs and lists.

This notion of reduction may look incomplete. One may expect a rule saying how a term $C[\mathsf{Exit}\ u]$ would reduce, where $C[]$ is a term context. The rules suffice, however, to reduce closed proofs of atomic sentences, and it is possible to show that the Church-Rosser property still holds for this extended language.

The notion of realizability is changed by adding that $\mathsf{Exit}_{\mathsf{Unit}}\ (\mathsf{Axiom}_i\ n_0)$ realizes *any* atomic formula (even if this atomic formula is actually false) for any numeral n_0.

5.2 Double Negation Shift

Double Negation Shift [18], or DNS for short, is the following axiom schema

$$[\forall x : \mathsf{N}\ \nabla\ \phi(x)] \Rightarrow \nabla\ \forall x : \mathsf{N}\ \phi(x).$$

It is direct to show that this schema plus the Axiom of Choice implies intuitionistically the negative interpretation of the Axiom of Choice.

We define recursively

$$\Phi\ P\ H\ l\ =\ P\ (\lambda x.\mathsf{get}\ x\ l\ (\mathsf{Exit}\ (H\ x\ (\lambda y.\Phi\ P\ H\ ((x,y):l)))))),$$

and then it is possible to show that $\lambda h \lambda p.\Phi\ p\ h\ []$ realizes DNS, by a reasoning quite similar as the one we did for the negative interpretation of the Axiom of Choice.

6 Comparison with Spector's work

In [6] Gödel introduced the so-called Dialectica interpretation, a translation of intuitionistic arithmetic into HA^ω, which satisfies the following preservation property. If ϕ is a theorem of intuitionistic arithmetic, then $\phi^D \equiv \exists f\ \forall x\ \phi_D$, with ϕ_D quantifier-free, can be *validated* in HA^ω (f and x may be sequences of variables). This means that there exists a closed term F such that $\forall x\ \phi_D[f := F]$ is a theorem of HA^ω. The types of f and x only depend on the logical structure of ϕ, and F depends on the proof of ϕ. Thus the Dialectica interpretation absorbes the quantifiers of ϕ at the cost of higher types in $\exists f\ \forall x\ \phi_D$.

In [18] Spector extended the Dialectica interpretation to analysis. The crucial step, where [18] goes beyond [6], is the Dialectica interpretation of DNS, which leads to the following formula.

$$\forall a\ y\ d\ \exists n\ z\ f\ [\phi(n, anz, z(anz)) \Rightarrow \phi(yf, f(yf), df))] \tag{2}$$

Here the type of n is N, the type of f is $\mathsf{N}{\to}\tau$ and the types of the other variables can easily be inferred. The problem is to find closed terms yielding n, z, f when applied to a, y, d. This will be done by bar recursion.

Spector solved the system of functional equations arising from (2), i.e.

$$\forall a\ y\ d\ \exists n\ z\ f\ [n = yf \wedge anz = f(yf) \wedge z(anz) = df] \tag{3}$$

by first reducing it to

$$\forall a\ y\ d\ \exists f\ \forall n \leq yf\ \exists z_n\ [anz_n = fn \wedge z_n fn = df] \tag{4}$$

and then solving (4) by bar recursion. Clearly (4) implies (3) by taking $n = yf$ and $z = z_n$.

We introduce the following abbreviation

$$\phi(n, f) \equiv\ \ n \leq yf \Rightarrow \exists\ z_n\ [anz_n = fn \wedge z_n fn = df]$$

Let A, Y, D be given. We shall define by bar recursion a functional Φ which extends every list $L = [F_{m-1}, \ldots, F_0]$ to a functional $F = \Phi L$ satisfying $Fi = F_i$ for all $i < m$ and $\forall n \geq m\ \phi(n, F)$. Then $F = \Phi[]$ solves (4). Assume such $\Phi(X{:}L)$ has already been defined for all X. We are looking for a definition of ΦL. We redefine $\mathsf{fun}\ L$ as the function which extends L with infinitely many function values $\underline{0}_\tau$. If $Y(\mathsf{fun}\ L) < m$, then we put $\Phi L = \mathsf{fun}\ L$ and $F = \Phi L$ vacuously satisfies $\forall n \geq m\ \phi(n, F)$, since $n > YF$ for all $n \geq m$. Now suppose $Y(\mathsf{fun}\ L) \geq m$. By assumption we have for any X that $F = \Phi(X : L)$ satisfies $\forall n \geq m+1\ \phi(n, F)$. So if we put $\Phi L = \Phi(F_m : L)$, then it only remains to determine a suitable F_m such that $F = \Phi(F_m : L)$ satisfies $\phi(m, F)$, i.e. $\exists\ z_m\ [Amz_m = Fm \wedge z_m Fm = DF]$. Take to this end $F_m = AmZ_m$ with $Z_m = \lambda x.D(\Phi(x : L))$, then indeed $Fm = F_m = AmZ_m$ and $Z_m(Fm) = Z_m(F_m) = D(\Phi(F_m{:}L)) = DF$. This completes the definition of Φ, and hence (4) is solved.

If we resume the definition of Φ, then we get:

$$\Phi\ a\ y\ d\ l = \begin{cases} \mathsf{fun}\ l & \text{if } y(\mathsf{fun}\ l) < m \\ \Phi\ a\ y\ d\ (am(\lambda x.d(\Phi\ a\ y\ d\ (x{:}l))){:}l) & \text{otherwise} \end{cases}$$

with m the length of the list l, which is an instance of the definition schema of bar recursion (see [18],[10]). Using Φ it is easy to find closed terms yielding n, z, f when applied to a, y, d, or in other words to solve (2).

It is not obvious how to compare the bar recursive interpretation of DNS with the realization using the Exit operator. One of the superficial similarities seems to be that the role of the Exit operator in the recursion is taken over by the condition $y(\mathsf{fun}\ l) \geq m$ (m the length of l).

There is an important difference in motivation between the work of Spector and ours. Spector's aim was to prove the consistency of analysis and to characterize its proof theoretic strength in terms of a subrecursive calculus. To this end, bar recursion suits well. Our aim is to explore the computational content of various choice axioms in combination with classical logic in order to extract algorithms from proofs. To this end, the Dialectica interpretation is rather indirect, and the realizability interpretation suits better.

7 Game Semantics

The purpose of this informal section is to explain how the program was found that realizes the negative interpretation of the Axiom of Choice. The importance of this section is that it may indicate a connection with the work on sequential algorithms, and may pave the way for a generalization to the case of the Axiom of Choice at higher-types, i.e. the axiom $\mathsf{AC}(\tau, \tau')$ with $\tau \neq \mathsf{N}$.

7.1 A semantics of classical arithmetic

The first step is to give an intuitive interpretation of cut-free provability as defined by Novikoff [17]. This is defined for an infinitary propositional calculus. Each arithmetical formula can be naturally represented in such a calculus. For instance $\forall k\ \exists n \geq k\ \forall m \geq k\ f(n) \leq f(m)$, which expresses that any function f has a minimum among its values $f(n)$ for $k \leq n$, becomes the propositional formula

$$\bigwedge_k \bigvee_{n \geq k} \bigwedge_{m \geq k} \phi_{nm},$$

where ϕ_{nm} is, for given numerals n and m, the truth value of the (decidable) formula $f(n) \leq f(m)$.

The formulae are defined inductively by

- 0 and 1 are formulae,
- if ϕ_i is a family of formulae over a countable decidable set, then $\bigwedge_i \phi_i$ and $\bigvee_i \phi_i$ are formulae.

Iterated conjunctions (disjunctions) are packed together as one single conjunction (disjunction), so that in every formula conjunctions and disjunctions are alternating. The notion of intuitionistic validity

- 1 is valid,
- $\bigvee_i \phi_i$ is valid iff there exists i_0 such that ϕ_{i_0} is valid,
- $\bigwedge_i \phi_i$ is valid iff ϕ_i is valid for all i,

has a natural game theoretical interpretation. The game is played by two players, $\forall$belard and $\exists$loise, who quarrel about the truth of a given formula. The player $\exists$loise, who argues in favour, plays when the formula is existential, say $\bigvee_i \phi_i$, by choosing an index i_0 ("$\exists$loise's Choice"), after which the game is continued with the formula ϕ_{i_0}. The player $\forall$belard, who argues against, plays when the formula is universal, say $\bigwedge_j \phi_j$, by choosing an index j_0, after which the game is continued with the formula ϕ_{j_0}. The game ends when either $\forall$belard or $\exists$loise chooses the formula 0 or 1. Evidently, the game always terminates. If either $\exists$loise or $\forall$belard chooses 1, then $\exists$loise wins the game and $\forall$belard looses. Dually, if either $\exists$loise or $\forall$belard chooses 0, then $\forall$belard wins the game and $\exists$loise looses.

We have the following completeness result: the player $\exists$loise has a winning strategy for the game associated to the existential formula ϕ if and only if this formula ϕ is intuitionistically valid. (Every formula is equivalent to an existential formula by prefixing it with a singleton disjunction.)

For instance, there is no winning strategy for the formula

$$\bigwedge_k \bigvee_{n \geq k} \bigwedge_{m \geq k} f(n) \leq f(m),$$

because it is not possible to compute effectively such a minimum value for a function f.

The notion of classical validity, as described by Novikoff [17], is inductively defined by the clauses

- 1 is valid,
- $\bigvee_i \phi_i$ is valid iff there exists i_0 such that ϕ_{i_0} is 1, or is an universal formula $\bigwedge_j \phi_{i_0 j}$ such that, for all j, the formula $\phi_{i_0 j} \vee \bigvee_i \phi_i$ is valid,
- $\bigwedge_i \phi_i$ is valid iff ϕ_i is valid for all i.

This definition may seem not well founded, but it correctly defines classical validity as the smallest set satisfying the above closure properties.

For instance, a formula of the form $\psi \vee \bigvee_i \phi_i \vee \overline{\phi_{i_0}}$ is always valid. This is shown by induction on the formula $\bigvee_i \phi_i$. First, recall that $\overline{\phi}$ is the usual defined complement of ϕ, changing $\bigvee$'s into $\bigwedge$'s, 1's in 0's, and conversely. Second, ϕ_{i_0} is either 0 or 1, or a universal formula $\bigwedge_j \phi_{i_0 j}$ by the alternation of $\bigvee$'s and $\bigwedge$'s. In the first two cases, it is trivial to choose the index of a valid disjunct in the formula above. In the third case, choose the index such that we get the formula $\psi \vee \bigvee_i \phi_i \vee \overline{\phi_{i_0}} \vee \phi_{i_0 j}$, which is valid for all j by the induction hypothesis.

The game semantics has to be adapted to the classical notion of validity: in the case of a disjunction $\bigvee_i \phi_i$, the game proceeds with the formula $\bigwedge_j [\phi_{i_0 j} \vee \bigvee_i \phi_i]$. The other cases do not change. A tentative interpretation of this definition is that it describes exactly a winning strategy for a game between $\forall$belard and $\exists$loise. This strategy is suggested by the inductive definition of classical validity: in the case of an existential formula, and after a choice j_0 of $\forall$belard, $\exists$loise should try again and choose a new i in

$$\phi_{i_0 j_0} \vee \bigvee_i \phi_i,$$

using information drawn from $\phi_{i_0 j_0}$ (if this latter formula is 1, then $\exists$loise should choose this formula, of course).

There is such a winning strategy for the formula

$$\bigwedge_k \bigvee_{n \geq k} \bigwedge_{m \geq k} \phi_{nm},$$

with $\phi_{nm} = f(n) \leq f(m)$, which can be described as follows. The game starts by $\forall$belard giving a value k. The player $\exists$loise answers choosing an arbitrary value $n \geq k$, say $n = k$ and $\forall$belard answers by playing $m = m_0 \geq k$. The formula becomes

$$\phi_{k m_0} \vee \bigvee_{n \geq k} \bigwedge_{m \geq k} \phi_{nm}.$$

If $\phi_{k m_0}$ is true, then $\exists$loise wins. If $\phi_{k m_0}$ is false, the strategy for $\exists$loise is to choose the last value given by $\forall$belard. After $\forall$belard chooses a value $m = m_1 \geq k$, the formula becomes

$$\phi_{k m_0} \vee \phi_{m_0 m_1} \vee \bigvee_{n \geq k} \bigwedge_{m \geq k} \phi_{nm}.$$

The player $\exists$loise wins if $f(m_0) \leq f(m_1)$. In this way, to each game corresponds a sequence $m_0, m_1, \ldots$ such that $\phi_{k m_0}$, $\phi_{m_0 m_1}$, $\ldots$ are all false, i.e. such that $f(m_0) > f(m_1) > \ldots$ Since N is well founded, $\exists$loise will win eventually by following this strategy.

7.2 Game semantics and realizability

It is possible to represent ∃loise's strategy by a program that realizes the negative interpretation of the formula $\forall k\ \exists n \geq k\ \forall m \geq k\ f(n) \leq f(m)$, which is $\forall k\ \nabla\ \exists n \geq k\ \forall m \geq k\ \nabla\ f(n) \leq f(m)$. We make the (inessential) simplification that $k = 0$. Consider the following program:

$$\Phi\ P\ n = P\ (n, \lambda m \lambda h.\ \mathsf{if}\ \phi_{nm}\ \mathsf{then}\ (h\ ())\ \mathsf{else}\ (\Phi\ P\ m)),$$

where () is the only constant of type Unit. It is possible to show that $\lambda p.\Phi\ p\ 0$ realizes the formula $\nabla\ \exists n\ \forall m\ \nabla\ f(n) \leq f(m)$. This program follows closely the previous strategy of ∃loise, and the proof that $\lambda p.\Phi\ p\ 0$ realizes the formula $\nabla\ \exists n\ \forall m\ \nabla\ f(n) \leq f(m)$ is similar to the argument showing that this strategy is winning.

7.3 A strategy for the Axiom of Choice

In [5] it was conjectured that it should be possible to extend this interpretation in the case of quantification over functions. The idea would be simply to allow as index set the set of all functions, and, apart from this single change, to keep the same notion of games and strategies. This suggested the realization of the Axiom of Choice given above.

There is an essential difficulty, closely connected to the problem in sequent calculus mentioned above, that we cannot require the strategy for ∃loise to win eventually against any player ∀belard. For instance, let us consider the classically valid formula

$$\bigvee_g \bigwedge_k \bigwedge_{m \geq k} f(g(k)) \leq f(m),$$

where g should be a function such that $k \leq g(k)$ for all k. It should be clear that ∀belard may force the game to be infinite by playing successively $k = 0, 1, 2, \ldots$ since at each move ∃loise can only hope to update the value $g(k)$.

To analyse this difficulty, we shall consider the more general case of a game-theoretic justification of the Axiom of Choice

$$\mathsf{AC} = \bigvee_m \bigwedge_x \bigwedge_y \overline{\phi_{mxy}} \vee \bigvee_f \bigwedge_m \bigvee_y \phi_{mf(m)y}.$$

In the sequel, ϕ_{mx} abbreviates $\bigvee_y \phi_{mxy}$; we have taken the instance of AC with this latter formula in order to have alternating quantifiers in case ϕ_{mx} is universal. The player ∃loise has the following natural strategy. ∃loise starts by playing an arbitrary function, for instance the constant function $f_0 = \lambda x.0$. The player ∀belard answers by a value $m = m_0$ and the formula becomes

$$\phi_{m_0 0} \vee \mathsf{AC}$$

The player ∃loise can then play $m = m_0$, and ∀belard answers with two values $x = x_0$ and $y = y_0$. The formula becomes

$$\overline{\phi_{m_0 x_0 y_0}} \vee \phi_{m_0 0} \vee \mathsf{AC}.$$

The player ∃loise backtracks by choosing $f_1 = f_0 + [m_0 \longmapsto x_0]$ which is defined by $f_1(m) = f_0(m)$ for $m \neq m_0$ and $f_1(m) = x_0$ for $m = m_0$. We can suppose that ∀belard will answer with a value $m_1 \neq m_0$. Indeed, if $m_1 = m_0$, then the formula will have the form

$$\overline{\phi_{m_0 x_0 y_0}} \vee \phi_{m_0 x_0} \vee \ldots$$

and we have seen above that ∃loise has always a winning strategy for this kind of formulae.

In this way, we can associate to this game a successive updating of the function f_0

$$f_k = f_0 + [m_0 \longmapsto x_0] + [m_1 \longmapsto x_1] + \ldots + [m_k \longmapsto x_k]$$

where m_{j+1} is $\forall$belard's answer to the move $f = f_j$ of $\exists$loise.

The crucial point now is that $\forall$belard loses as soon as we have $m_{k+1} = m_j$ for some $j \leq k$.

If we assume that the player $\forall$belard can only base his move on finitely many information from the move $f = f_k$ of $\exists$loise, this will happen eventually, by a continuity argument.

Hence, if we change our notion of winning strategy in asking only that the player $\exists$loise has to win eventually against any "finite" player $\forall$belard, this strategy is a winning strategy for the formula AC. The program that realizes the negative interpretation of the Axiom of Choice is a direct translation of this strategy. As in the arithmetical case, the proof of reducibility of this program closely follows the argument that such a strategy is "winning" against any "reasonable" player $\forall$belard.

This argument is only yet heuristic and the main problem is making precise what is a "finite" player $\forall$belard. We have seen that it would be wrong to ask that it wins against *any*, non-computable, strategy. But it is probably too restrictive to ask that it wins eventually against any computable strategy. One may try to define a winning strategy for $\exists$loise as a strategy winning eventually against any continuous strategy for $\forall$belard. We avoided here this difficulty by formulating the interpretation in a functional system, but we can hope for a much more perspicuous analysis of the computation if it is expressed as an interaction between strategies.

Another possibility would be to use sequential algorithms for representing functions. To give a function will then not be an atomic operation, but the function will in general be given "piece by piece". One may then hope to avoid the consideration of continuity.

8 Conclusion

There are many directions in which this work can be improved and some potential connections that seem worthwhile to analyse.

One is the formulation of this work in the framework of sequential algorithms, which is indeed the natural framework in which one can observe "intensional behaviour" of functionals, which has been our main tool in the realization of the negative interpretation of the Axiom of Choice. We expect from such a formulation a much more transparent proof of termination (and it will be interesting to see what becomes in this formalism the difficulty of finding "fresh" constants). We also think that in such a framework, the generalization to higher types, that is, a computational interpretation of $\mathsf{AC}(\tau, \tau')$ for arbitrary τ and τ', should be straightforward. An elegant form would be an interpretation of the notion of dependent types in the framework of sequential algorithms, together with a direct interpretation of $\nabla\ \phi$, and the inference of $\nabla\ \forall x : \tau.\phi(x)$ from $\forall x : \tau.\nabla\ \phi(x)$.

Closely related should be the question of a constructive formulation of our proof of realizability. Can we adapt the method of [2] and avoid the use of infinite terms? Our hope is that our interpretation, computationally more direct than the one of Spector [18, 10], may provide help for a constructive understanding of classical analysis with the Axiom of Choice.

Experiments with some examples, in particular Higman's Lemma, have revealed a computational inefficiency of our interpretation. Intuitively, our algorithms proceed by trial and errors and may "forget too many things" of previous trials. It seems possible to design improved algorithms, that "remember" and can use all previous trials. This problem is closely connected to the problem

of the computational interpretation of the implication $\nabla\ \phi \Rightarrow \nabla\ \psi \Rightarrow \nabla\ (\phi \wedge \psi)$ so that its solution should have consequences even for the problem of the computational content of propositional classical logic.

The previous remark, and metaphors such as "trials and errors", "conjecturing laws that are refined by experimentation", are immediately suggested by trying to understand the computational behaviour of our interpretation. They hint at possible connections with learning theories, such as the one described in [19], where the learning agent may benefit from negative information.

Yet another natural connection is with the work of Hilbert [8, 9] and Ackermann [1]. Our interpretation can be seen as a variation of Hilbert's epsilon method [8, 9], with a (classical) proof of termination. It will be interesting to compare our algorithm with the one described in Ackermann's paper [1] [4].

Acknowledgement

We thank Anne Troelstra for helpful comments and answers to various questions that arose during the preparation of this paper, Loïc Colson for interesting discussions on the axiom of choice and impredicativity at the beginning of this work, and Marco Hollenberg and Jan Springintveld for carefully reading successive draft versions.

References

[1] W. Ackermann. Begründung des Tertium non datur mittels der Hilbertschen Theorie der Widerspruchsfreiheit. Mathematische Annalen, 93, 1924, p. 1-36

[2] M. Bezem. Strong Normalization of Barrecursive Terms Without Using Infinite Terms. Archiv für mathematische Logik und Grundlagenforschung, 25, 1985, p. 175-182

[3] E. Bishop. *Foundations of Constructive Analysis.* New York, McGraw-Hill, 1967.

[4] R. Constable and C. Murthy. Finding Computational Content in Classical Proofs. In G.Huet and G. Plotkin, editors, *Logical Frameworks*, 341 - 362, (1991), Cambridge University Press.

[5] Th. Coquand. A semantics of evidence for classical arithmetic. Journal of Symbolic Logic, to appear, 1994.

[6] K. Gödel. *Collected Work.* Volumes I and II, S. Feferman, J. W. Dawson, S.C. Kleene, G.H. Moore, R. M. Solovay, J. van Heijenoort, editors, Oxford, 1986.

[7] N. Goodman. *Intuitionistic arithmetic as a theory of constructions.* PhD thesis, Stanford University, 1968.

[8] D. Hilbert. Die logischen Grundlagen der Mathematik. Mathematische Annalen, 88, 1923, p. 151-165

[9] D. Hilbert. The foundations of mathematics. In van Heijenoort ed., *From Frege to Gödel*, p. 465-479.

[4] According to [14], if it is well-known that the proof of convergence of Ackermann's algorithm was defective as presented in [1], it is not known yet, not even non-constructively, if the method converges at all.

[10] W.A. Howard. Functional interpretation of bar induction by bar recursion. Compos. Mathematica 20 (1968), 107-124.

[11] W.A. Howard and G. Kreisel. Transfinite induction and bar induction of types zero and one, and the role of continuity in intuitionistic analysis. Journal of Symbolic Logic 31, 1966, p. 325 - 358.

[12] S.C. Kleene. On the interpretation of intuitionistic number theory. Journal of Symbolic Logic 10 (1945), p. 109 - 124.

[13] A.N. Kolmogorov. On the principle of the excluded middle. In *From Frege to Gödel*, J. van Heijenoort, editor, Harvard University Press, Cambridge MA, 1971, pp. 414-437.

[14] G. Kreisel. Mathematical Logic. In *Lectures on Modern Mathematics*, vol. III, ed. Saaty, Wiley, N.Y., 1965, p. 95 - 195.

[15] G.E. Moore. *Zermelo's Axiom of Choice: Its Origins, Development and Influence.* Springer-Verlag, 1982

[16] C. Murthy. *Extracting Constructive Content from Classical Proofs.* Ph. D. Thesis, 1990.

[17] Novikoff P.S. On the consistency of certain logical calculi. Matematiceskij sbornik (Recueil Mathématique) T. 12 (54), p. 230-260, 1943.

[18] C. Spector. Provably recursive functionals of analysis: a consistency proof of analysis by an extension of principles formulated in current intuitionistic mathematics. *Recursive Function Theory*, J.C.E. Dekker ed., Proceedings of Symposia in Pure Mathematics V, AMS, p. 1 - 27, 1961.

[19] D.N. Osherbon, M. Stob and S. Weinstein. *Systems That Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists.* MIT Press, 1986.

[20] J. R. Shoenfield. *Mathematical Logic.* Addison-Wesley, 1967.

[21] W.W. Tait. Normal Derivability in Classical Logic. Springer Lecture Notes in Mathematics 72 (1968), J. Barwise ed., p. 204-236.

[22] W.W. Tait. Normal form theorem for bar recursive functions of finite type. In *Proceedings of the second Scandinavian Logic Symposium,* J.E. Fenstad ed., North Holland, Amsterdam, 1971, pp. 353-367.

[23] A.S. Troelstra. *Metamathematical Investigation of Intuitionistic Arithmetic and Analysis.* Lecture Notes in Mathematics 344, Springer-Verlag, Berlin, 1973.

[24] A.S. Troelstra. Realizability. ILLC Prepublication Series for Mathematical Logic and Foundations ML-92-09.