A Computational Interpretation of Open Induction

Ulrich Berger [1]
University of Wales Swansea
Department of Computer Science
Singleton Park, Swansea SA2 0BD, UK
u.berger@swansea.ac.uk

Abstract

We study the proof-theoretic and computational properties of open induction, a principle which is classically equivalent to Nash-Williams' minimal-bad-sequence argument and also to (countable) dependent choice (and hence contains full classical analysis). We show that, intuitionistically, open induction and dependent choice are quite different: Unlike dependent choice, open induction is closed under negative- and A-translation, and therefore proves the same Π^0_2-formulas (over not necessarily decidable, basic predicates) with classical or intuitionistic arithmetic. Via modified realizability we obtain a new direct method for extracting programs from classical proofs of Π^0_2-formulas using open induction. We also show that the computational interpretation of classical countable choice given by Berardi, Bezem and Coquand [2] can be derived from our results.

1. Introduction

The combination of Gödel's negative- and Friedman's A-translation yields a simple and flexible method for proving the Π^0_2-conservativity of the classical versions of various formal systems over their intuitionistic counterparts. With Kleene/Kreisel (modified) realizabiliy added one obtains a method for extracting computational content from classical proofs which can be applied to, for example, various extensions of Peano arithmetic and Zermelo-Fraenkel set theory [11]. Unfortunately, the method does not work for classical analysis formalized as an extension of Peano arithmetic with function variables plus the axiom scheme of *countable choice*

$\mathrm{AC}\quad \forall n\,\exists x\,A(n,x) \to \exists f\,\forall n\,A(n,fn)$

or *dependent choice*

$\mathrm{DC}\quad \forall n\,\forall x\,\exists y\,A(n,x,y) \to \exists f\,\forall n\,A(n,fn,f(n+1))$.

(the latter scheme is called $\omega\mathrm{AC}$ in [19]; see e.g. [14], or [13] for other, equivalent forms of dependent choice). The method fails in this case because ACN, the negative translation of AC, is not intutitionistically provable (from AC or any other intuitionistically valid principles). The same holds for DCN and DC.

There are, however, other methods: Spector [27] extended Gödel's Dialectica interpretation [12] to classical analysis by interpreting ACN by bar recursion in finite types. Another solution was given by Berardi, Bezem and Coquand [2] who used a special form of realizability to interpret ACN. Oliva and the author [3] gave a modified realizability interpretation of the A-translation of ACN and DCN based on a variant of bar recursion in finite types. Recently, a rather different, more machine oriented interpretation was proposed by Krivine [18].

The significance of Spector's interpretation for reductive proof theory is widely regarded rather limited [10, 1] (but see [19]), and the other interpretations mentioned above do not seem to make new contributions in this respect. However, there *are* improvements concerning the algorithmic behavior of the extracted realizers: In particular the realizer, let us call it Φ, of ACN given by Berardi, Bezem and Coquand is very appealing as it implements a clever 'demand driven' algorithm [2] which seems to be superior over the other solutions (e.g. bar recursion) which rather perform a 'blind' (though terminating) search.

The research presented in our paper was prompted by the desire to find a simple explanation and correctness argument for Φ replacing the somewhat ad-hoc realizability interpretation and complicated proof in [2]. Indeed, we show that Φ is the computational content of the (negative- and A-translated) classical proof of AC using the principle of *open induction* (and a realizer thereof).

The principle of open induction was formulated by Raoult [24] in a classical context and discussed by Coquand [6] from an intuitionistic point of view. It is a classi-

1 Supported by the British Engineering and Physical Sciences Research Council, grant GR/R16020/01

cal reformulation of Nash-Williams minimal-bad-sequence argument which is used in classical proofs of Kruskal's theorem and related theorems [21, 6]. Open induction is the scheme of induction over infinite sequences ordered lexicographically, but restricted to predicates that are open in the pointwise topology. The restriction is necessary because the lexicographic ordering on infinite sequences is not wellfounded for arbitrary predicates.

As we will see, open induction fits well into the extraction method based on negative translation, A-translation and realizability. In particular the modified realizability interpretation of open induction is straightforward and can easily be proven to be correct by open induction. In order to have closure under negative translation we will work with a version of open induction based on an intuitionistically slightly more general (but classically equivalent) notion of open predicates than in [6]. On the other hand we show that open induction is classically equivalent to the (intuitionistically apparently weaker) principle of *update induction* which corresponds to open induction over a wellfounded relation having descending chains of length at most two.

Our paper not only aims at proof-theoretic results, but also at developing methods that are of interest to practical program extraction from proofs. Our interpretation of open induction allows for a direct extraction of programs from classical proofs using the minimal-bad-sequence argument, avoiding the reduction of the minimal-bad-sequence argument to dependent choice. In order to facilitate applications we work with a finite type system and allow free predicate symbols as parameters which are not a-priory assumed to be decidable as well as additional Horn-axioms.

The *main results* of the paper are: The Π^0_2-conservativity of classical open induction over intuitionistic open induction (theorem 3.7), a reduction of classical dependent choice to a restricted form of update induction for Π^0_2-formulas (theorem 3.9), a realization of open and update induction (theorems 4.5 and 4.4), program extraction from classical proofs of Π^0_2-formulas using open induction or dependent choice (theorems 5.2, 5.3 and 5.6), and, finally, the extraction of the Berardi-Bezem-Coquand realizer of ACN from a proof using update induction and thus a definition of this realizer in terms of update recursion.

2. Intuitionistic and classical arithmetic in finite types

We work in the following extension of Heyting arithmetic in finite types. *Types* are the base types $()$ (denoting a singleton set), nat and boole, function types $\rho \to \sigma$, product types $\rho \times \sigma$, and finite sequences ρ^*. We set $\rho^\omega :\equiv \mathsf{nat} \to \rho$. *Terms* are typed lambda terms with the usual constants for the given types including constants for Gödel primitive recursion in all types over the natural numbers, the booleans (i.e. definition by cases), and finite sequences. For a finite sequence $w\colon \rho^*$ and $x\colon \rho$ we let $w{*}x$ be the finite sequence obtained from w by attaching x as last element. If $\alpha\colon \rho^\omega$ is an infinite sequence, then $(w@\alpha)$ denotes the infinite sequence obtained by overriding α with w, that is, $(w@\alpha)k = w_k$ if $k < |w|$ and $= \alpha k$ otherwise. *Atomic formulas* are of the form $P(t_1, \ldots, t_n)$ where P is taken from a given set of *free predicate symbols* each of a fixed arity $(\rho_1, \ldots, \rho_n)$. We do not specify precisely the set of predicate symbols, but assume that it includes a nullary predicate symbol $\bot$, for absurdity, and for each type ρ a binary predicate symbol $=_\rho$ for equality at type ρ. Type information will often be suppressed if it is clear from the context or irrelevant. A boolean equation of the form $t = \mathsf{T}$ will often be abbreviated by t alone. *Formulas* are built from atomic formulas by $\wedge$, $\vee$, $\to$, $\forall x^\rho$ and $\exists x^\rho$. $\neg A$ stands for $A \to \bot$. *Axioms* are the schemes of induction for natural numbers, finite sequence (of arbitrary type) and the booleans (the latter is $A(\mathsf{T}) \wedge A(\mathsf{F}) \to \forall b\, A(b)$), the defining equations for the constants, $\beta\eta$-equality, and a certain set $\mathcal{H}$ of closed Horn-formulas, i.e. formulas of the form $\forall \vec{x}\,(A_0 \wedge \ldots \wedge A_{n-1} \to A_n)$ where the A_i are atomic formulas (the point is that Horn-formulas intuitionistically imply their negative- and A-translations and have no computational content). It is not necessary to completely fix the set $\mathcal{H}$, but it suffices to require that all its members are true in the model $\mathcal{C}$ of total continuous functionals (see section 5) w.r.t. the intended interpretation of the free predicate symbols. The *logical rules* of HA^ω are those of intuitionistic logic with equality. We denote the system described above by $\mathsf{HA}^\omega[\mathcal{H}]$, or HA^ω for short. PA^ω is like HA^ω, but with classical logic. We write $\Gamma \vdash A$ respectively $\Gamma \vdash_{\mathsf{cl}} A$ if A is provable in HA^ω respectively PA^ω from assumptions Γ. $\Gamma \vdash \Delta$ means $\Gamma \vdash D$ for all $D \in \Delta$. Unless stated otherwise, by a 'proof' we will mean a proof in HA^ω.

Remark. Since HA^ω has no axioms expressing decidability or extensionality of equality it is similar to what is called *neutral Heyting arithmetic* in [28]. Note however that some extensionality is available through the Horn-theory $\mathcal{H}$: If $\mathcal{H}$ contains all equations, $r = s$, with $\mathrm{Ext} \vdash r = s$ where Ext is the principle of extensionality

$$\mathrm{Ext} \quad \forall x^\rho\, fx =_\sigma gx \to f = g$$

(which is valid in $\mathcal{C}$, but not a Horn formula), then HA^ω is closed under the following *weak extensionality rule* (see e.g. [16]): If $\Delta \vdash \forall x\, rx = sx$ where Δ is a set of quantifier free formulas, then $\Delta \vdash r = s$. Note however that quantifier free formulas are *not* decidable in general, that is, we cannot necessarily prove the law of excluded middle for them.

We now recall some well-known facts about negative- and A-translation adapted to our version of HA^ω. Gödel's

negative translation, $A^{\neg\neg}$, double negates all atomic, disjunctive and existential subformulas of a formula.

Lemma 2.1 *(a)* $\vdash \neg\neg A^{\neg\neg} \to A^{\neg\neg}$.

(b) If $\Gamma \vdash_{\mathsf{cl}} A$, *then* $\Gamma^{\neg\neg} \vdash A^{\neg\neg}$.

Proof. (a) is proved by induction on A, (b) is proved by induction on derivations using (a) and the fact that $H \vdash H^{\neg\neg}$ for Horn-formulas H.

For formulas A, B we define B_A (the A-translation of B) as the formula obtained from B by replacing $\bot$ by A and every other atomic subformula C by $C \vee A$.

Lemma 2.2 *(a)* $\vdash A \to B_A$.

(b) If $\Gamma \vdash B$, *then* $\Gamma_A \vdash B_A$.

Proof. (a) is proved by induction on B, (b) is proved by induction on derivations using the fact that $H \vdash H_A$ for Horn-formulas H.

Next we make precise what we mean by a Σ-formula. A type without $\to$ is called a *data type*, and a predicate of arity $(\tau_1, \dots, \tau_n)$ where all τ_i are data types is called a *data predicate*. A formula is called a *data formula* if all predicate symbols occurring in it are data predicates. A Σ-*formula* is a data formula which is $\to \forall$ *free*, i.e. Σ-formulas are built from atomic data formulas by conjunction, disjunction and existential quantification. Clearly any Σ-formula is intuitionistically equivalent to a formula of the form $\exists \vec{x}\, A$ where A is a quantifier free data formula. The imposed restriction on the arities of predicates will entail a continuity property (lemma 4.3) which is needed in the correctness proof for the realizer of open induction.

Lemma 2.3 *Let* B *be a* $\to \forall$ *free formula.*

(a) $\vdash B^{\neg\neg} \leftrightarrow \neg\neg B$.

(b) $\vdash B_A \leftrightarrow B \vee A$.

Proof. Easy induction on B.

We say an axiom system Γ is *closed under negative translation* respectively *closed under* A*-translation* if $\Gamma \vdash \Gamma^{\neg\neg}$ respectively $\Gamma \vdash \Gamma_A$ for every Σ-formula A.

Theorem 2.4 (Friedman) *Suppose* Γ *is closed under* A*-translation. Then* Γ *is closed under* Markov's rule, *i.e. if* $\Gamma \vdash \neg\neg A$, *then* $\Gamma \vdash A$, *for every* Σ*-formula* A.

Proof. Assume that Γ is closed under A-translation and $\Gamma \vdash \neg\neg A$ where A is a Σ-formula. Then, by lemma 2.2 (b), $\Gamma_A \vdash (\neg\neg A)_A$, that is, $\Gamma_A \vdash (A_A \to A) \to A$. Since Γ is closed under A-translation and, by lemma 2.3 (b), $A_A \to A$ is provable, it follows $\Gamma \vdash A$.

Corollary 2.5 *Let* Γ *be closed under negative translation and* A*-translation. Then* $\Gamma \vdash_{\mathsf{cl}} A$ *implies* $\Gamma \vdash A$ *for every* Σ*-formula* A.

Proof. Lemmas 2.1 (b), 2.3 (a) and theorem 2.4.

An example of an axiom system Γ satisfying the hypotheses of corollary 2.5 is *transfinite induction* on a (decidable) relation $<: \rho \times \rho \to \mathsf{boole}$,

$$\mathrm{TI}(<) \quad \forall x\, (\forall y < x\, A(y) \to A(x)) \to \forall x\, A(x)$$

where A ranges over arbitrary predicates. As already remarked, dependent choice, DC, and countable choice, AC both do *not* satisfy the hypotheses of corollary 2.5, neither does the scheme of extensionality, Ext. All these principles fail to be closed under negative translation.

3. Open induction and update induction

The principle of open induction we study here is, from an intuitionistic point of view, slightly more general than the one in [6]. For any predicate B of arity (ρ^*) and any quantifier $\mathsf{Q} \in \{\forall, \exists\}$ we define a predicate B^{Q} of arity (ρ^ω) by $B^{\mathsf{Q}}(\alpha) :\equiv \mathsf{Q}n\, B(\overline{\alpha}n)$ where $\overline{\alpha}n :\equiv [\alpha 0, \dots, \alpha(n-1)]$. A predicate U over ρ^ω is called *open* if there are a predicate C and a Σ-predicate B such that $U(\alpha)$ is equivalent to $C^\forall(\alpha) \to B^\exists(\alpha)$ (in [6] open predicates are of the form $B^\exists(\alpha)$ with decidable B). For a decidable relation $<: \rho \times \rho \to \mathsf{boole}$ on ρ we define a binary relation $<_{\mathsf{lex}}$ on ρ^ω (the lexicographic extension of $<$) by $\beta <_{\mathsf{lex}} \alpha :\equiv \exists n\, (\overline{\beta}n = \overline{\alpha}n \wedge \beta n < \alpha n)$. The principle of *open induction* is

$$\mathrm{OI} \quad \forall\alpha\, (\forall\beta <_{\mathsf{lex}} \alpha\, U(\beta) \to U(\alpha)) \to \forall\alpha\, U(\alpha).$$

where U ranges over open predicates and $<$ is assumed to be provably wellfounded, i.e. transfinite induction for $<$, $\mathrm{TI}(<)$, is provable.

Remarks. 1. $<_{\mathsf{lex}}$ is not wellfounded (except when $<$ is trivial). For example, in Cantor space we have the infinite descending sequence

$$10000\ldots >_{\mathsf{lex}} 01000\ldots >_{\mathsf{lex}} 00100\ldots >_{\mathsf{lex}} \quad . \quad . \quad .$$

2. Classically, any predicate of the form $U(\alpha) \equiv C^\forall(\alpha) \to B^\exists(\alpha)$ (with C and B arbitrary) is equivalent to a predicate of the form $U(\alpha) \equiv \neg C^\forall(\alpha)$ and hence open.

Lemma 3.1 *Open predicates are extensional, i.e. if* $U(\alpha)$ *is open, then we have*

$$\forall n\, \alpha n = \beta n \to U(\alpha) \to U(\beta)$$

Proof. This follows from the fact that in $U(\alpha)$ the variable α occurs only in contexts of the form $\overline{\alpha}n$ where n is not lambda-abstracted.

As observed by Coquand [6], the principle of open induction can be reduced intuitionistically to the following

principle of relativized bar induction [19] (also called extended bar induction [29]). We show that this still holds for our more general notion of open predicate. Let B, S, P be predicates on ρ^* where B is a Σ-predicate and call P *hereditary on* S if $\forall w \in S\,(\forall x\,(S(w{*}x) \to P(w{*}x)) \to P(w))$. Then the principle of *relativized bar induction* of type ρ is

$$\mathrm{RBI} \quad S([]) \wedge (S^\forall \subseteq B^\exists) \wedge (S \cap B \subseteq P) \wedge \\ \wedge\,(P \text{ hereditary on } S) \to P([])$$

Classically, this principle follows from dependent choice. The question whether it is constructively acceptable is discussed in [19]. The following proposition and its proof are a straightforward generalization of Coquand's result [6].

Proposition 3.2 *Relativized bar induction proves open induction.*

Proof. Let $<$ be a wellfounded relation and U an open predicate, i.e. $U(\alpha) \equiv C^\forall(\alpha) \to B^\exists(\alpha)$ with a Σ-predicate B. Assume U is 'progressive', that is $\forall\alpha\,(\forall\beta <_{\mathsf{lex}} \alpha\, U(\beta) \to U(\alpha))$. Let $\alpha_0 \colon \rho^\omega$ be arbitrary with $C^\forall(\alpha_0)$. We have to show $B^\exists(\alpha_0)$. For $w \colon \rho^*$ we set $P(w) :\equiv \forall\alpha\, U(w@\alpha)$. It suffices to prove $P([])$. We will do this by relativized bar induction with the predicates B and P as given respectively defined above and $S(w) :\equiv C(w) \wedge (w = [] \vee \exists v, x\,(w = v{*}x \wedge \forall y < x\, P(v{*}y)))$. We need to verify the four premises of (RBI). $S([])$ holds because we assumed $C^\forall(\alpha_0)$. To show $S^\forall \subseteq B^\exists$ we assume $S^\forall(\alpha)$. Then $C^\forall(\alpha)$. Hence, because U is progressive, it suffices to show $\forall\beta <_{\mathsf{lex}} \alpha\, U(\beta)$. Let $\beta <_{\mathsf{lex}} \alpha$, say $\overline{\beta}n = \overline{\alpha}n$ and $\beta n < \alpha n$. Since $S^\forall(\alpha)$ we have $S(\overline{\alpha}(n{+}1))$ and hence $P(\overline{\beta}(n{+}1))$ because $\beta n < \alpha n$. Therefore $U(\beta)$. $S \cap B \subseteq P$ holds trivially since even $B \subseteq P$. Finally we need to show that P is hereditary on S. Let $w \in S$ such that $S(w{*}x) \to P(w{*}x)$ for all x. In order to show $P(w)$ it suffices to show $\forall x\, P(w{*}x)$. We do this by induction on the wellfounded relation $<$. We need to show $P(w{*}x)$ under the induction hypothesis $\forall y < x\, P(w{*}y)$. By definition of P we may in addition assume $C(w{*}x)$. But then $S(w{*}x)$ and hence also $P(w{*}x)$.

Remark. As shown by Coquand [7], open induction for ordinary open predicates (i.e. predicates of the form $U(\alpha) \equiv B^\exists(\alpha)$ with decidable B) on Cantor space can be proven by (ordinary) bar induction with decidable bar. The computational content of this form of open induction has been analyzed by Mahboubi [20].

We also consider a principle similar to open induction which we call *update induction*. Update induction is concerned with *partial sequences of type* ρ by which we mean objects of type $(\mathsf{boole} \times \rho)^\omega$ (the idea behind the type $\mathsf{boole} \times \rho$ is to simulate a type $1 + \rho$ or Haskell's Maybe ρ). For any partial sequence α the sequence $\mathsf{dom}(\alpha) \colon \mathsf{boole}^\omega$ and $\mathsf{val}(\alpha) \colon \rho^\omega$ are defined by $\alpha n = (\mathsf{dom}(\alpha)n, \mathsf{val}(\alpha)n)$. $\mathsf{dom}(\alpha)$ represents (the characteristic function of) the domain of α. So, we regard α as 'defined' at n if and only if $\mathsf{dom}(\alpha)n = \mathsf{T}$ and in that case $\mathsf{val}(\alpha)n$ is its value. We will often write '$n \in \mathsf{dom}(\alpha)$' for '$\mathsf{dom}(\alpha)n = \mathsf{T}$' and '$\alpha[n]$' instead of '$\mathsf{val}(\alpha)n$'. If $n \notin \mathsf{dom}(\alpha)$, i.e. $\mathsf{dom}(\alpha)n = \mathsf{F}$, and x is of type ρ, then we call the partial sequence $\alpha^x_n :\equiv \overline{\alpha}n{*}(\mathsf{T}, x)@\alpha$, i.e. $\alpha^x_n n = (\mathsf{T}, x)$ and $\alpha^x_n k = \alpha k$ for $k \neq n$, an *update* of α. We write $\beta <_{\mathsf{up}} \alpha$ if β is an update of α. The principle of update induction is now

$$\mathrm{UI} \quad \forall\alpha\,(\forall\beta <_{\mathsf{up}} \alpha\, U(\beta) \to U(\alpha)) \to \forall\alpha\, U(\alpha)$$

where U ranges over open predicates.

Proposition 3.3 *Open induction proves update induction.*

Proof. The binary relation $<$ on $\mathsf{boole} \times \rho$ defined by $(b, y) < (a, x) :\equiv b = \mathsf{T} \wedge a = \mathsf{F}$ is clearly decidable and wellfounded, and $\beta <_{\mathsf{up}} \alpha$ implies $\beta <_{\mathsf{lex}} \alpha$.

Proposition 3.4 *The principles of dependent choice, open induction and update induction are classically (i.e. provably in* PA^ω*) equivalent.*

Proof. DC *implies* OI. We repeat the (well-known) argument given in [6]. One proves the contrapositive of OI, i.e. the minimal-bad-sequence argument. Assume $\alpha_0 \colon \rho^\omega$ is an infinite bad sequence, that is, $\neg U(\alpha_0)$ holds. Using dependent choice and the minimal element principle for $<$ (i.e. transfinite induction on $<$) one constructs an infinite sequence $\alpha \colon \rho^\omega$ such that for each n the finite sequence $\overline{\alpha}(n{+}1)$ can be extended to an infinite bad sequence, but for each $y < \alpha n$ the sequence $\overline{\alpha}n{*}y$ cannot. Because U is open α is bad and, by construction of α, all $\beta <_{\mathsf{lex}} \alpha$ are good, i.e. $U(\beta)$ holds. Alternatively one can prove RBI from DC, as remarked earlier, and use proposition 3.2.

OI *implies* UI. Proposition 3.3.

UI *implies* DC. Assume (1) $\forall n\, \forall x^\rho\, \exists y^\rho\, A(n, x, y)$. Let us call a partial sequence α of type ρ a 'partial choice function' if $0 \in \mathsf{dom}(\alpha)$ and for all n if $n + 1 \in \mathsf{dom}(\alpha)$ then $n \in \mathsf{dom}(\alpha)$ and $A(n, \alpha[n], \alpha[n + 1])$ holds. Note that being *not* a partial choice function is an open property of partial sequences. Clearly there exist partial choice functions, for example any partial sequence with domain $\{0\}$ is one. By update induction and classical logic it follows that there is a partial choice function α such that all updates of α aren't partial choice functions. It suffices to show that α is total, since then for $f :\equiv \mathsf{val}(\alpha)$ we have $\forall n\, A(n, fn, f(n + 1))$. Assume α were not total. let n be minimal such that $n + 1 \notin \mathsf{dom}(\alpha)$. By (1) there exists y such that $A(n, \alpha[n], y)$ holds. Clearly $\beta :\equiv \alpha^y_{n+1}$ is a partial choice function which updates α contradicting the assumption on α.

A closer look at the third part of the proof above shows that we in fact proved intuitionistically the negative version of dependent choice from a special instance of update induction. A similar fact holds for the negative version of

countable (independent) choice. We call a predicate $U(\alpha)$ 1-*open* if $U(\alpha)$ is of the form $D(\alpha) \to B^{\exists}(\alpha)$ where B is a Σ-predicate and $D(\alpha) \equiv \forall n\,(n \in \mathsf{dom}(\alpha) \to D_0(n, \alpha[n])$ with an arbitrary predicate D_0. 2-open predicates are defined similarly, but with $D(\alpha) \equiv \forall n > 0\,(n \in \mathsf{dom}(\alpha) \to n-1 \in \mathsf{dom}(\alpha) \wedge D_0(n, \alpha[n-1], \alpha[n])$. Clearly, every p-open predicate is open. We let UI_p denote update induction restricted to p-open predicates.

Proposition 3.5 *(a)* UI_1 *proves*

$$\mathrm{ACN} \quad \forall n\,\neg\neg\exists x\,A(n,x) \to \neg\neg\exists f\,\forall n\,A(n,fn)$$

(b) UI_2 *proves*

$$\mathrm{DCN} \quad \forall n\,\forall x\,\neg\neg\exists y\,A(n,x,y) \to \neg\neg\exists f\,\forall n\,A(n,fn,f(n+1))$$

Proof. To prove (a), assume (1) $\forall n\,\neg\neg\exists x\,A(n,x)$ and (2) $\neg\exists f\,\forall n\,A(n,fn)$. We call α a partial choice function if $A(n,\alpha[n])$ holds for all $n \in \mathsf{dom}(\alpha)$. Clearly, not being a partial choice function is a 1-open property. We prove by update induction that there is no partial choice function (which is absurd). Assume that α is a partial choice function. By (2) it suffices to show $\mathsf{dom}(\alpha)n = \mathsf{T}$ for every n, since then $\forall n\,A(n,\mathsf{val}(\alpha)n)$. Assume $\mathsf{dom}(\alpha)n = \mathsf{F}$ (equality between booleans is decidable!). By (1) it suffices to show $\neg\exists x\,A(n,x)$. Assume $A(n,x)$ for some x. Then $\beta :\equiv \alpha^x_n$ is a partial choice function updating α, which contradicts the (update) induction hypothesis. Part (b) is proved similarly along the lines of the third part of proposition 3.4. Note that the proof of (b) involves induction on natural numbers while the proof of (a) does not.

Lemma 3.6 *Open induction and update induction are both closed under negative translation and A-translation.*

Proof. First note that the negative translation and the A-translation of an open predicate are open. Furthermore, by lemma 3.1, $\beta <_{\mathsf{lex}} \alpha$ is equivalent to the $\to\,\forall$ free formula $\exists n,y,\gamma\,(y < \alpha n \wedge \beta = \overline{\alpha}n{*}y@\gamma)$. Therefore, we we have, by lemma 2.3, that $(\beta <_{\mathsf{lex}} \alpha)^{\neg\neg} \leftrightarrow \neg\neg\beta <_{\mathsf{lex}} \alpha$ and $(\beta <_{\mathsf{lex}} \alpha)_A \leftrightarrow \beta <_{\mathsf{lex}} \alpha \vee A$. From this and the lemmas 2.1 (a), and 2.2 (a) it follows easily that the negative- respectively A-translation of open induction for U is implied by open induction for $\neg\neg U$ respectively U_A. The same argument works for update induction since $\beta <_{\mathsf{up}} \alpha$ is equivalent to the $\to\,\forall$ free formula $\exists n\,(\mathsf{dom}(\alpha)n = \mathsf{F} \wedge \beta = \overline{\alpha}n{*}(\mathsf{T}, \beta[n])@\alpha)$.

Theorem 3.7 *Classical open induction is conservative over intuitionistic open induction for Σ-formulas. More precisely, a proof of* $\mathrm{OI} \vdash_{\mathsf{cl}} A$ *where A is a Σ-formula translates via negative and A-translation into a proof of* $\mathrm{OI} \vdash A$. *A corresponding result holds for update induction.*

Proof. Lemma 3.6 and corollary 2.5.

Lemma 3.8 *Update induction restricted to p-open predicates, where $p = 1, 2$, is closed under A-translation.*

Proof. It is easy to see that the A-translation (but unfortunately in general not the negative translation) of a p-open predicate is again p-open. The rest is similar to the proof of lemma 3.6.

Theorem 3.9 *Let A be a Σ-formula.*

(a) If $\mathrm{AC} \vdash_{\mathsf{cl}} A$, *then* $\mathrm{UI}_1 \vdash A$.

(b) If $\mathrm{DC} \vdash_{\mathsf{cl}} A$, *then* $\mathrm{UI}_2 \vdash A$.

Proof. (a) If $\mathrm{AC} \vdash_{\mathsf{cl}} A$, then $\mathrm{ACN} \vdash \neg\neg A$, by lemmas 2.1 and 2.3. With proposition 3.5 it follows $\mathrm{UI}_1 \vdash \neg\neg A$ and hence $\mathrm{UI}_1 \vdash A$, by lemma 3.8 and theorem 2.4. The argument for (b) is similar.

4. Realizability

Since HA^ω has a well-known computational interpretation, e.g. in terms of realizability, theorem 3.9 reduces the problem of extracting computational content from proofs of Σ-formulas using dependent choice and classical logic to the problem of realizing update induction. In order to carry this out we will work with (a formalized version of) Kreisel's *modified realizability* [17] as described e.g. in [28]. For every formula A and term r (of a type determined by A) the formula $r\,\mathbf{mr}\,A$ is defined by induction on A (w.l.o.g. we assume r to be in long $\beta\eta$ normal form):

$$\begin{array}{rcl}
r^{()}\,\mathbf{mr}\,A & \equiv & A \quad \text{if } A \text{ is atomic,} \\
(r,s)\,\mathbf{mr}\,(A \wedge B) & \equiv & r\,\mathbf{mr}\,A \wedge s\,\mathbf{mr}\,B \\
(b^{\mathsf{boole}}, r, s)\,\mathbf{mr}\,(A \vee B) & \equiv & (b = \mathsf{T} \wedge r\,\mathbf{mr}\,A) \vee \\
 & & \quad (b = \mathsf{F} \wedge s\,\mathbf{mr}\,B) \\
r^{\rho\to\sigma}\,\mathbf{mr}\,(A \to B) & \equiv & \forall x^\rho\,(x\,\mathbf{mr}\,A \to rx\,\mathbf{mr}\,B) \\
(r,s)\,\mathbf{mr}\,\exists x\,A(x) & \equiv & s\,\mathbf{mr}\,A(r) \\
r^{\rho\to\sigma}\,\mathbf{mr}\,\forall x^\rho\,A(x) & \equiv & \forall x\,(rx\,\mathbf{mr}\,A(x))
\end{array}$$

Theorem 4.1 (Soundness of realizability) *Assume $B_1, \ldots, B_n \vdash A$. Then $x_1\,\mathbf{mr}\,B_1, \ldots, x_n\,\mathbf{mr}\,B_n \vdash r\,\mathbf{mr}\,A$ for some term r extracted from the given derivation.*

Proof. Induction on derivations using the fact that $\vdash r\,\mathbf{mr}\,H \leftrightarrow H$ for Horn-formulas H.

Lemma 4.2 *(a) $\vdash A \leftrightarrow \exists x\,(x\,\mathbf{mr}\,A)$ for every $\to\,\forall$ free formula A.*

(b) If B is a Σ-formula, so is the formula $r\,\mathbf{mr}\,B$.

Proof. Induction on formulas.

From theorem 3.9 and the theorem and lemma above it follows that a realizability interpretation of restricted update induction, UI_2, will provide us with a method of extracting from every proof of $\mathrm{DC} \vdash_{\mathsf{cl}} \exists x\,A(x)$ where A is a

Σ-formula a term r such that $A(r)$ holds. Similarly, by theorem 3.9, a realization of open induction will allow us to extract from a proof $\mathrm{OI} \vdash_{\mathsf{cl}} \exists x\, A(x)$ a term r such that $A(r)$ holds. The reason why we treat the schemes DC and OI separately, even though by proposition 3.4 they are classically equivalent, is computational: For DC we just need a realizer of UI_2 which, as we will see, is simpler than the realizer of OI. On the other hand, the classical reduction of OI to DC is hard to formalize (minimal-bad-sequence argument, or proposition 3.2), thereore, given a classical proof using open induction (e.g. Kruskal's theorem), it might be advantageous to extract a program directly using a realizer of OI.

Troelstra [28] observed that transfinite induction, $\mathrm{TI}(<)$, on a decidable wellfounded relation $<$ can be realized by a recursive functional

$$\mathrm{TR} \quad \mathsf{R}fx = fx(\lambda y.\mathsf{if}\ y < x\ \mathsf{then}\ \mathsf{R}fy)$$

where 'if b then z^σ' is shorthand for 'if b then z else 0^σ' with some (arbitrary) closed term 0^σ. A similar recursion scheme was proof-theoretically analyzed by Schwichtenberg and Wainer [25]. For realizing open and update induction we can use the same idea, however, we have to take into account that $<_{\mathsf{up}}$ and $<_{\mathsf{lex}}$ are not decidable and not well-founded for arbitrary predicates. Note that update induction can be equivalently stated as

$$\mathrm{UI} \quad \forall\alpha\,(\forall n \notin \mathsf{dom}(\alpha)\,\forall x\, U(\alpha^x_n) \to U(\alpha)) \to \forall\alpha\, U(\alpha)$$

This suggests to add for each pair of types ρ, τ where τ is a data type a constant $\mathsf{R}^{\mathsf{u}}{}_{\rho,\tau}$ for *update recursion* with the defining equation

$$\mathrm{UR} \quad \mathsf{R}^{\mathsf{u}} f\alpha =_\tau f\alpha(\lambda n, x.\mathsf{if}\ n \notin \mathsf{dom}(\alpha)\ \mathsf{then}\ \mathsf{R}^{\mathsf{u}} f\alpha^x_n)$$

where f is of type ρ^ω. Similarly, open induction is equivalent to

$$\mathrm{OI} \quad \forall\alpha\,(\forall n, y, \gamma\,(y < \alpha n \to U(\overline{\alpha}n{*}y@\gamma)) \to U(\alpha)) \to \forall\alpha\, U(\alpha)$$

suggesting constants $\mathrm{OR}_{\rho,\tau,<}$ for *open recursion*

$$\mathrm{OR} \quad \mathsf{R}^{\mathsf{o}} f\alpha =_\tau f\alpha(\lambda n, y, \gamma.\mathsf{if}\ y{<}\alpha n\ \mathsf{then}\ \mathsf{R}^{\mathsf{o}} f(\overline{\alpha}n{*}y@\gamma))$$

where again τ is a data type and f is of type ρ^ω. We will show that open induction can indeed be realized using open recursion, and the restricted forms of update induction, UI_1 and UI_2 (which are sufficient by theorem 3.9), are realized by update recursion (full update induction, however, seems to require a realizer similar to open recursion).

In order to prove that the realizers we will construct are correct we need the following continuity principle for functionals F of type $\rho^\omega \to \mathsf{nat}$

$$\mathrm{Cont} \quad \forall F, \alpha\, \exists n\, \forall\beta\,(\overline{\alpha}n =_\rho \overline{\beta}n \to F\alpha =_{\mathsf{nat}} F\beta)$$

This principle holds in all constructively meaningful models of HA^ω. In particular it holds in the model of continuous functionals [15, 17] (see also section 5).

Lemma 4.3 *For every Σ-formula A we have* $\mathrm{Cont} \vdash \forall\alpha\,(A(\alpha) \to \exists n\, \forall\beta\,(\overline{\alpha}n = \overline{\beta}n \to A(\beta)))$.

Proof. It easy to see that Cont can be generalized to functionals F of type $\rho^\omega \to \tau$ where τ is a data type. It follows that the assertion holds for atomic Σ-formulas (because of the restriction on arities of predicates), and hence, by induction, for all Σ-formulas.

Theorem 4.4 $\mathrm{Cont} + \mathrm{OI} + \mathrm{OR} \vdash \Phi\,\mathbf{mr}\,\mathrm{OI}$ *for some term Φ explicitely definable from open recursion.*

Proof. Let $U(\alpha)$ be open, that is, of the form $C^\forall(\alpha) \to B^\exists(\alpha)$ where B is a Σ-predicate. Let σ and τ be the types corresponding to the formulas C and B respectively. Note that τ is a data type. Using the open recursor $\mathsf{R}^{\mathsf{o}}{}_{\rho\times\sigma,\tau,<'}$, where $(y,z) <' (y_1,z_1) :\equiv y < y_1$, we can define a term Φ such that the equation

$$\Phi f\alpha\delta =_\tau f\alpha(\lambda n, y^\rho, \gamma, \eta.\ \mathsf{if}\ y < \alpha n\ \mathsf{then}\ \Phi f(\overline{\alpha}n{*}y@\gamma)(\overline{\delta}n@\eta))\delta$$

follows from OR. In order to prove that Φ realizes OI for the given predicate U we assume that f realizes the progressiveness of U, i.e.

$$f\,\mathbf{mr}\,\forall\alpha\,(\forall n, y, \gamma\,(y < \alpha n \to U(\overline{\alpha}n{*}y@\gamma)) \to U(\alpha))$$

We have to show $\Phi f\,\mathbf{mr}\,\forall\alpha\, U(\alpha)$, which expands to $\forall\alpha,\delta\,(\delta\,\mathbf{mr}\,C^\forall(\alpha) \to \Phi f\alpha\delta\,\mathbf{mr}\,B^\exists(\alpha))$. Note that, identifying (α,δ) with $\lambda k(\alpha k, \delta k)$ and by lemmas 4.2 (b) and 4.3, the predicate $V(\alpha,\delta) :\equiv \delta\,\mathbf{mr}\,C^\forall(\alpha) \to \Phi f\alpha\delta\,\mathbf{mr}\,B^\exists(\alpha)$ is open, which allows us to argue by open induction. Assume $\delta\,\mathbf{mr}\,C^\forall(\alpha)$. We need to prove $\Phi f\alpha\delta\,\mathbf{mr}\,B^\exists(\alpha)$. By Φ's defining equation and the assumption that f realizes the progressiveness of U this reduces to showing $\Phi f(\overline{\alpha}n{*}y@\gamma)(\overline{\delta}n@\eta)\,\mathbf{mr}\,B^\exists(\overline{\alpha}n{*}y@\gamma)$ for all n, y, γ, η such that $y < \alpha n$ and $\eta\,\mathbf{mr}\,C^\forall(\overline{\alpha}n{*}y@\gamma)$. By open induction hypothesis it suffices to show $\overline{\delta}n@\eta\,\mathbf{mr}\,C^\forall(\overline{\alpha}n{*}y@\gamma)$, i.e. $\forall k\,((\overline{\delta}n@\eta)k\,\mathbf{mr}\,C(\overline{\overline{\alpha}n{*}y@\gamma}k))$. For $k < n$ this holds because $\delta\,\mathbf{mr}\,C^\forall(\alpha)$, for $k \geq n$ this follows from the assumption $\eta\,\mathbf{mr}\,C^\forall(\overline{\alpha}n{*}y@\gamma)$.

Theorem 4.5 $\mathrm{Cont} + \mathrm{UI}_p + \mathrm{UR} \vdash \Phi\,\mathbf{mr}\,\mathrm{UI}_p$ *($p = 1, 2$) for a term Φ explicitely definable from update recursion.*

Proof. The proof is similar to the previous proof, but slightly more technical. We only prove the case $p = 2$, the case $p = 1$ being similar. Let $U(\alpha)$ (α of type $(\mathsf{boole}\times\rho)^\omega$) be 2-open, that is, of the form $D(\alpha) \to B^\exists(\alpha)$ where B is a Σ-predicate and $D(\alpha) \equiv \forall n > 0\,(n \in \mathsf{dom}(\alpha) \to n-1 \in \mathsf{dom}(\alpha) \wedge D_0(n, \alpha[n-1], \alpha[n]))$. Let σ and τ be the types

corresponding to the formulas D_0 and B respectively. Using the update recursor $\mathsf{R}^{\mathsf{u}}{}_{\rho\times\sigma,\tau}$ we can explicitely define a term Φ such that

$$\Phi f\alpha\delta =_\tau f\alpha(\lambda n, x^\rho, \eta.\mathsf{if}\ n \notin \mathsf{dom}(\alpha)\ \mathsf{then}\ \Phi f\alpha^x_n\delta^{\eta n}_n)\delta$$

follows from UR. In order to prove that Φ realizes UI for the given predicate U we assume that f realizes the progressiveness of U, i.e.

$$f\,\mathbf{mr}\,\forall\alpha\,(\forall n, x\,(n \notin \mathsf{dom}(\alpha) \to U(\alpha^x_n)) \to U(\alpha)).$$

We have to show $\forall\alpha,\delta\,(\delta\,\mathbf{mr}\,D(\alpha) \to \Phi f\alpha\delta\,\mathbf{mr}\,B^\exists(\alpha))$. We again identify (α,δ) with $\lambda k(\alpha k, \delta k)$ and see that the predicate $V(\alpha,\delta) :\equiv \delta\,\mathbf{mr}\,D(\alpha) \to \Phi f\alpha\delta\,\mathbf{mr}\,B^\exists(\alpha)$ is 2-open, so we may argue by update induction. Assume $\delta\,\mathbf{mr}\,D(\alpha)$, i.e. $\forall k > 0\,(k \in \mathsf{dom}(\alpha) \to k-1 \in \mathsf{dom}(\alpha) \wedge \delta k\,\mathbf{mr}\,D_0(k, \alpha[k-1], \alpha[k]))$. We need to prove $\Phi f\alpha\delta\,\mathbf{mr}\,B^\exists(\alpha)$. By the given assumptions we need to show $\Phi f\alpha^x_n\delta^{\eta n}_n\,\mathbf{mr}\,B^\exists(\alpha^x_n)$ for all n, x, η such that $n \notin \mathsf{dom}(\alpha)$ and $\eta\,\mathbf{mr}\,D(\alpha^x_n)$. By update induction hypothesis it suffices to show $\delta^{\eta n}_n\,\mathbf{mr}\,D(\alpha^x_n)$. Assume $0 < k \in \mathsf{dom}(\alpha^x_n)$. We need to show $k-1 \in \mathsf{dom}(\alpha^x_n)$ and $\delta^{\eta n}_n\,\mathbf{mr}\,D_0(k, \alpha^x_n[k-1], \alpha^x_n[k])$. From $n \notin \mathsf{dom}(\alpha)$ and $\delta\,\mathbf{mr}\,D(\alpha)$ it follows $k \le n$. If $k < n$ we use the assumption $\delta\,\mathbf{mr}\,D(\alpha)$. If $k = n$, then we are done because $\eta\,\mathbf{mr}\,D(\alpha^x_n)$.

5. Program extraction

By program extraction in general we mean the process of extracting from a proof of $\forall x\,\exists y\,A(x,y)$ a term Φ such that $\forall x\,A(x,\Phi x)$ holds in a certain structure and the value of Φx can be computed for every instance of x. A natural structure for interpreting finite type languages is the model $\mathcal{C}$ of *total continuous functionals* of Kleene [15] and Kreisel [17] (see also [22]). This model satisfies the continuity principle and dependent choice (the latter holds because in $\mathcal{C}$ a type ρ^ω is interpreted as the set of *all* sequences $\alpha\colon\mathbb{N}\to\mathcal{C}^\rho$). Ershov [9] showed that $\mathcal{C}$ is the extensional collapse of the total elements of the domain-theoretic model $\hat{\mathcal{C}}$ of *partial continuous functionals* [26] (where, of course, 'total' is meant here in the domain-theoretic sense and not in the sense of the proof of proposition 3.4). The model $\hat{\mathcal{C}}$ has the advantage that every computable functional of a type $\rho\to\rho$ has a least fixed point. Hence for any extension of Gödel's system T (i.e the term system described in section 2) by constants c with recursive defining equations, $c = rc$, every closed term t has a natural value $t^{\hat{\mathcal{C}}}$ in $\hat{\mathcal{C}}$. It follows, by Ershov's result, that if a constant given by a recursive equation as above has a total solution in $\hat{\mathcal{C}}$ it has a solution in $\mathcal{C}$ as well.

Proposition 5.1 *The update recursor* R^{u} *and the open recursor* R° *are total and hence exist in the total continuous functionals.*

Proof. Let $f \in \hat{\mathcal{C}}$ of type $(\mathsf{boole}\times\rho)^\omega \to (\rho\to\tau)^\omega \to \tau$ be total. For $\alpha\in\hat{\mathcal{C}}$ of type $(\mathsf{boole}\times\rho)^\omega$ we define $U(\alpha) \equiv \alpha$ total $\to \mathsf{R}^{\mathsf{u}} f\alpha$ total. Since R^{u} is continuous, τ is a data type and $\mathcal{C}$ satisfies Cont, it follows that $U(\alpha)$ is an open property. From the recursion equation UR it is clear that $U(\alpha)$ is 'update progressive' (i.e. satisfies the hypothesis of update induction). Therefore, by update induction, $U(\alpha)$ holds for all α. For R° the proof is similar.

Theorem 5.2 *From a derivation* $\mathrm{OI} \vdash_{\mathsf{cl}} \forall x^\rho\,\exists y^\tau\,A(x,y)$ *where* $A(x,y)$ *is a* Σ*-formula and* τ *is a data type one can extract a closed open recursive term* $\Phi^{\rho\to\tau}$ *such that* $\mathrm{Cont}+\mathrm{OI}+\mathrm{OR} \vdash \forall x^\rho\,A(x,\Phi x)$ *and hence* $\forall x^\rho\,A(x,\Phi x)$ *holds in* $\mathcal{C}$.

Proof. By theorem 3.7 we have $\mathrm{OI} \vdash \forall x^\rho\,\exists y^\tau\,A(x,y)$ and by theorem 4.4 we can extract an open recursive term Φ such that $\mathrm{Cont}+\mathrm{OI}+\mathrm{OR} \vdash \Phi\,\mathbf{mr}\,\forall x^\rho\,\exists y^\tau\,A(x,y)$. Φ has type $\rho\to\tau\times\sigma$ where σ is the type corresponding to $A(x,y)$. Set $\Phi' \equiv \lambda x\pi_0(\Phi x)$ where $\pi_0(.)$ is the left projection. Then, by lemma 4.2 (a), $\mathrm{Cont}+\mathrm{OI}+\mathrm{OR} \vdash \forall x^\rho\,A(x,\Phi' x)$ and hence $\forall x^\rho\,A(x,\Phi' x)$ holds in $\mathcal{C}$ because $\mathcal{C}$ is a model of $\mathrm{Cont}+\mathrm{OI}+\mathrm{OR}$.

Theorem 5.3 *From a derivation* $\mathrm{DC} \vdash_{\mathsf{cl}} \forall x^\rho\,\exists y^\tau\,A(x,y)$ *where* $A(x,y)$ *is as above one can extract a closed update recursive term* $\Phi^{\rho\to\tau}$ *such that* $\mathrm{Cont}+\mathrm{UI}+\mathrm{UR} \vdash \forall x^\rho\,A(x,\Phi x)$ *and hence* $\forall x^\rho\,A(x,\Phi x)$ *holds in* $\mathcal{C}$.

Proof. Similar to the proof of theorem 5.2.

In order to interpret realizers as programs we need a correct and terminating operational semantics for terms (that may contain open or update recursors). To this end we will use Plotkin's adequacy theorem [23] relating the operational call-by-name semantics and the denotational domain semantics of PCF. Modulo some obvious adjustments the theorem applies our situation (i.e. extensions of Gödel's system T by recursively defined constants). The defining equations of the constants can be read as term rewriting rules which, together with the usual conversion rules for λ-calculus, can be used to define a call-by-name operational semantics. We call a closed normal term of a data type built from the constructors of data types (i.e. the boolean constants, zero, successor, the empty list, cons and pairing) a *numeral*. We will write $\underline{n}$ for numerals of any data type (not only nat). It is easy to see that any closed term of a data type is a numeral. Numerals are in a one-to-one correspondence with their values which are exactly the total elements in $\hat{\mathcal{C}}$ of a data type. We identify numerals with their values.

Theorem 5.4 (Plotkin) *Let* t *be a closed term and* $\underline{n}$ *a numeral of the same data type. Then* t *has value* $\underline{n}$ *iff* t *normalizes to* $\underline{n}$.

Proposition 5.5 *Every update recursive term of a data type has a unique normal form.*

Proof. This follows immediately from theorem 5.4 and proposition 5.1.

Theorem 5.6 *From any given derivation* $\mathrm{OI}(\mathrm{DC}) \vdash_{\mathsf{cl}} \forall x^\rho \exists y^\tau A(x,y)$ *where* $A(x,y)$ *is a* Σ*-formula and* τ *is a data type one can extract a closed open (update) recursive term* $\Phi^{\rho\to\tau}$ *such that for each closed term* r^ρ *the term* Φr *reduces to a numeral* $\underline{n}$ *such that* $A(r,\underline{n})$ *holds in* $\mathcal{C}$.

Proof. Immediate, by theorem 5.2 (5.3) and proposition 5.5.

6. The Berardi-Bezem-Coquand functional

We now show that the computational interpretations of countable choice given by Berardi, Bezem and Coquand [2] can be derived from ours. In [2] essentially a realizer of ACN_A, i.e.

$$\forall n\,((\exists x\, B_A(n,x) \to A) \to A) \\ \to (\exists f\, \forall n\, B_A(n,fn) \to A) \to A$$

(B arbitrary, A a Σ-formula) was given. We show how to extract this realizer from a proof using update induction. If we A-translate proposition 3.5 we obtain a proof of ACN_A from update induction applied to a 1-open predicate. Extracting a program from this proof using the update recursive realizer of UI_1 given in the proof of theorem 4.5, we obtain the following realizer of ACN_A. Let ρ,σ,τ be the types of x, B_A, A respectively and assume $F\colon \mathsf{nat} \to (\rho\times\sigma\to\tau)\to\tau$ realizes $\forall n\,((\exists x\, B_A(n,x)\to A)\to A)$ and $G\colon \rho^\omega\times\sigma^\omega\to\tau$ realizes $\exists f\,\forall n\, B_A(n,fn)\to A$. Identifying the types $\rho^\omega\times\sigma^\omega$ and $(\rho\times\sigma)^\omega$ we can also say $G\colon(\rho\times\sigma)^\omega\to\tau$. Let furthermore $H\colon\tau\to\sigma$ be a realizer of $A\to B_A(n,x)$ (see lemma 2.2 (a); H is independent of n and x). From F, G and H a realizer of A is constructed as $\Psi\lambda n.(\mathsf{F},0^{\rho\times\sigma})$ where $\Psi\colon(\mathsf{boole}\times(\rho\times\sigma))^\omega\to\tau$ is recursively defined by

$$\Psi\eta =_\tau G(\lambda n.\\ \quad \mathsf{if}\ n\in\mathsf{dom}(\eta)\ \mathsf{then}\ \eta[n]\ \mathsf{else}\ (0^\rho, H(Fn\lambda z^{\rho\times\sigma}.\Psi\eta^z_n)))$$

If we unfold the term $\Psi\lambda n.(\mathsf{F},0^{\rho\times\sigma})$ a finite number of times, it is clear that Ψ will always be called with an argument η of finite domain. Hence we may replace η by a variable f of type $(\mathsf{nat}\times(\rho\times\sigma))^*$, and define

$$\Phi f =_\tau G(\lambda n.\\ \mathsf{if}\ n\in\mathsf{dom}(f)\ \mathsf{then}\ f[n]\ \mathsf{else}\ (0^\rho, H(Fn\lambda z^{\rho\times\sigma}.\Phi f{*}(n,z)))$$

where now '$n\in\mathsf{dom}(f)$' means that the finite sequence f contains a member of the form (n,z) and in that case $f[n]$ stands for z (in order to determine z uniquely we can, e.g., take the rightmost pair (n,z) in f). Except for some inessential notational differences this is exactly the functional defined in [2]. By Scott's fixed point induction one easily proves $\Phi f = \Psi\tilde{f}$ where $\tilde{f}n := \mathsf{if}\ n\in\mathsf{dom}(f)\ \mathsf{then}\ (\mathsf{T}, f[n])\ \mathsf{else}\ (\mathsf{F},0^{\rho\times\sigma})$ (i.e. one proves by induction on k that $\Phi^k f = \Psi^k\tilde{f}$ where Φ^k,Ψ^k are the approximation of Φ,Ψ, respectively, determined by their recursive definitions). In particular $\Phi[] = \Psi\lambda n.(\mathsf{F},0^{\rho\times\sigma})$. Hence $\Phi[]$ realizes A.

If one is just interested in a short and concise proof of the totality and correctness of the realizer in [2] (and not so much in its proof-theoretic explanation), one can directly prove by update induction that the functional Ψ above is total and has the following property: If η is such that for all $n\in\mathsf{dom}(\eta)$ we have $\eta[n]=(x,y)$ with $y\,\mathbf{mr}\,B_A(n,x)$, then $\Psi\eta\,\mathbf{mr}\,A$. Note that we can apply update induction because, by lemma 4.3 the validity of $\Psi\eta\,\mathbf{mr}\,A$ depends only on a finite initial segment of η (see also the proof of theorem 4.5 for a similar argument).

We close with a simple example demonstrating that the functional given by Berardi, Bezem and Coquand may yield indeed better extracted programs than other known realizers of countable choice. Consider the following (silly) classical proof of $\forall n\,\exists m\, n=m$: First prove (in minimal logic using reflexivity of equality) $\forall n\,\neg\neg\exists m\, n=m$. Then apply ACN, the negative translation of countable choice, to obtain $\neg\neg\exists f\,\forall n\,\neg\neg n=fn$. From this we again conclude (in minimal logic) $\forall n\,\neg\neg\exists m\, n=m$. Using a realizer of the A-translation of ACN one obtains a program p with $\forall n\, n=pn$. It is now easy to see that if we use the Berardi-Bezem-Coquand functional, the program p will be operationally equivalent to $\lambda n.n$ (i.e. the number of computation steps is independent of n) whereas the realizer based on bar recursion in [27] and [3] would yield a program which, roughly, given n searches from 0 upwards until it eventually hits n.

7. Conclusion

We introduced a version of the principle of open induction (and a weaker form called update induction) which is closed under negative- and A-translation, and therefore proves the same Σ-formulas classically or intuitionistically. A modified realizability interpretation of open induction together with Plotkin's adequacy theorem provided us with a new and direct method for extracting programs from classical proofs using open induction. We also showed that the computational interpretation of the axiom of countable choice given in [2] can be derived from our interpretation of update induction.

We believe that our results will be useful for extracting interesting new programs from classical proofs of theorems in infinitary combinatory logic, such as Kruskal's theorem

and generalizations thereof. In particular the fact that we may admit free predicate symbols which are axiomatized by Horn-formulas (the Horn-condition can even be slightly weakened [4]), but for which no decision procedures are required, supports the extraction of programs from proofs in abstract mathematics. In contrast, program extraction based on Gödel's Dialectica interpretation [12] *does* use decision algorithms for atomic formulas unless one is satisfied with sequences or sets of candidates of realizers [8], [5], or upper bounds [16] (on the other hand, the Dialectica interpretation easily yields conservativity results for classical Σ^0_1-induction and classical quantifier free choice, but our method doesn't).

An interesting open problem, which is also discussed in [2], is the question of how to extract programs from classical proofs using arbitrary (not necessarily countable) choice. Is there a generalization of open induction which matches this case as well?

Acknowledgments

I am grateful to the anonymous referees for valuable suggestions that helped a lot in improving the paper. One referee spotted an error in an earlier version of theorem 4.5.

References

[1] J. Avigad and S. Feferman. Gödel's functional ("Dialectica") interpretation. In S. Buss, editor, *Handbook of proof theory*, volume 137, pages 337–405. Elsevier, North-Holland, Amsterdam, 1998.

[2] S. Berardi, M. Bezem, and T. Coquand. On the computational content of the axiom of choice. *Journal of Symbolic Logic*, 63(2):600–622, 1998.

[3] U. Berger and P. Oliva. Modified bar recursion and classical dependent choice. In *Logic Colloquium 2001*. Springer, to appear.

[4] U. Berger, H. Schwichtenberg, and W. Buchholz. Refined program extraction from classical proofs. *Annals of Pure and Applied Logic*, to appear.

[5] W. Burr. Diller-Nahm-style functional interpretation of $KP\omega$. *Archive for Mathematical Logic*, 39(8):599–604, 2000.

[6] T. Coquand. Constructive toplogy and combinators. In *Constructivity in Computer Science*, volume 613 of *Lecture Notes in Computer Science*, pages 159–164. Springer, 1991.

[7] T. Coquand. A note on the open induction principle, 1997.

[8] J. Diller and W. Nahm. Eine Variante zur Dialectica-Interpretation der Heyting-Arithmetik endlicher Typen. *Archiv für mathematische Logik und Grundlagenforschung*, 16:49–66, 1974.

[9] Y. Ershov. Model C of partial continuous functionals. In R. Gandy and M. Hyland, editors, *Logic Colloquium 1976*, pages 455–467. North Holland, Amsterdam, 1977.

[10] S. Feferman. Gödel's Dialectica interpretation and its two-way stretch. In G. e. a. Gottlob, editor, *Computational Logic and Proof Theory, Lecture Notes in Computer Science*, volume 713, pages 23–40. Springer, 1993.

[11] H. Friedman. Classically and intuitionistically provably recursive functions. In D. Scott and G. Müller, editors, *Higher Set Theory, Lecture Notes in Mathematics*, volume 669, pages 21–28. Springer, 1978.

[12] K. Gödel. Über eine bisher noch nicht benützte Erweiterung des finiten Standpunktes. *Dialectica*, 12:280–287, 1958.

[13] P. Howard and J. Rubin. *Consequences of the Axiom of Choice*. Math. Surv. and Monographs, vol. 58. American Mathematical Society, 1998.

[14] 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(3):325–358, 1966.

[15] S. C. Kleene. Countable functionals. In A. Heyting, editor, *Constructivity in Mathematics*, pages 81–100. North–Holland, Amsterdam, 1959.

[16] U. Kohlenbach. On uniform weak König's lemma. *Annals of Pure and Applied Logic*, 114:103–116, 2002.

[17] G. Kreisel. Interpretation of analysis by means of constructive functionals of finite types. *Constructivity in Mathematics*, pages 101–128, 1959.

[18] J.-L. Krivine. Countable choice and 'quote'. Unpublished, 2002.

[19] H. Luckhardt. *Extensional Gödel Functional Interpretation – A Consistency Proof of Classical Analysis*, volume 306 of *Lecture Notes in Mathematics*. Springer, 1973.

[20] A. Mahboubi. An induction principle over real numbers. Submitted to *Archive for Mathematical Logic*, 2004.

[21] C. Nash-Williams. On well-quasi-ordering finite trees. *Proc. Cambridge Phil. Soc.*, 59:833–835, 1963.

[22] D. Normann. *Recursion on the countable functionals*, volume 811 of *Lecture Notes in Mathematics*. Springer, 1980.

[23] G. D. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5:223–255, 1977.

[24] J.-C. Raoult. Proving open properties by induction. *Information processing letters*, 29:19–23, 1988.

[25] H. Schwichtenberg and S. Wainer. Ordinal bounds for programs. In P. Clote and J. Remmel, editors, *Feasible Mathematics II*, pages 387–406. Birkhäuser, Boston, 1995.

[26] D. S. Scott. Outline of a mathematical theory of computation. In *4th Annual Princeton Conference on Information Sciences and Systems*, pages 169–176, 1970.

[27] C. Spector. Provably recursive functionals of analysis: a consistency proof of analysis by an extension of principles in current intuitionistic mathematics. In F. D. E. Dekker, editor, *Recursive Function Theory: Proc. Symposia in Pure Mathematics*, volume 5, pages 1–27. American Mathematical Society, Providence, Rhode Island, 1962.

[28] A. Troelstra. *Metamathematical investigation of intuitionistic Arithmetic and Analysis*, volume 344 of *Lecture Notes in Mathematics*. Springer, 1973.

[29] A. Troelstra. Extended bar induction of type zero. In *The Kleene Symposium*. North-Holland, 1980.