Computational Interpretations of Analysis via Products of Selection Functions

(Preprint submitted for publication)

Martín Escardó[1] and Paulo Oliva[2]

[1] University of Birmingham
[2] Queen Mary University of London

Abstract. We show that the computational interpretation of full comprehension via two well-known functional interpretations (dialectica and modified realizability) corresponds to two closely related infinite products of selection functions.

1 Introduction

Full classical analysis can be formalised using the language of finite types in Peano arithmetic PA^ω extended with the axiom schema of *full comprehension* (cf. [10])

$$\mathsf{CA} \quad : \quad \exists f^{\mathbb{N}\to\mathbb{B}} \forall n^{\mathbb{N}} (f(n) \leftrightarrow A(n)).$$

As $\forall n^{\mathbb{N}}(A(n) \vee \neg A(n))$ is equivalent to $\forall n^{\mathbb{N}} \exists b^{\mathbb{B}}(b \leftrightarrow A(n))$, full comprehension, in the presence of classical logic, follows from *countable choice* over the booleans

$$\mathsf{AC}^{\mathbb{N}}_{\mathbb{B}} \quad : \quad \forall n^{\mathbb{N}} \exists b^{\mathbb{B}} A(n,b) \to \exists f \forall n A(n, fn).$$

Finally, the negative translation of $\mathsf{AC}^{\mathbb{N}}_{\mathbb{B}}$ follows intuitionistically from $\mathsf{AC}^{\mathbb{N}}_{\mathbb{B}}$ itself together with the classical principle of *double negation shift*

$$\mathsf{DNS} \quad : \quad \forall n^{\mathbb{N}} \neg\neg A(n) \to \neg\neg \forall n A(n),$$

with A being a negated formula. Therefore, full classical analysis can be embedded (via the negative translation) into $\mathsf{HA}^\omega + \mathsf{AC}^{\mathbb{N}}_{\mathbb{B}} + \mathsf{DNS}$, where HA^ω is Heyting arithmetic in the language of all finite types. It then follows that a computational interpretation of theorems in analysis can be obtained via a computational interpretation of the theory $\mathsf{HA}^\omega + \mathsf{AC}^{\mathbb{N}}_{\mathbb{B}} + \mathsf{DNS}$. The fragment $\mathsf{HA}^\omega + \mathsf{AC}^{\mathbb{N}}_{\mathbb{B}}$, excluding the double negation shift, has a very straightforward (modified) realizability interpretation [13], as well as a dialectica interpretation [1, 9]. The remaining challenge is to give a computational interpretation for DNS.

A computational interpretation of DNS was first given by Spector [12], via the dialectica interpretation. Spector devised a form of recursion on well-founded trees, nowadays known as *bar recursion*, and showed that the dialectica interpretation of DNS can be witnesses by such recursion. A computational interpretation of DNS via realizability only came recently, first in [2], via a non-standard form of realizability, and then in [3, 4], via Kreisel's modified realizability. The realizability interpretation of DNS makes use of a new form of bar recursion, termed *modified bar recursion*.

In this article we show that both forms of bar recursion used to interpret classical analysis, via modified realizability and the dialectica interpretation, correspond to two closely related infinite products of selection functions [8].

Notation. We use X, Y, Z for variables ranging over types. Although in PA^ω one does not have dependent types, we will develop the rest of the paper working with types such as $\Pi_{i\in\mathbb{N}} X_i$ rather than its special case X^ω, when all X_i are the same. We often write $\Pi_i X_i$ for $\Pi_{i\in\mathbb{N}} X_i$. Also, we write $\Pi_{i\geq k} X_i$ for $\Pi_i X_{k+i}$, and $\mathbf{0}$ for the constant functional 0 of a particular finite type. If α has type $\Pi_{i\in\mathbb{N}} X_i$ we use the following abbreviations

$$\begin{aligned}
[\alpha](n) &\equiv \langle \alpha(0), \ldots, \alpha(n-1)\rangle, \quad \text{(initial segment of } \alpha \text{ of length } n)\\
\alpha[k,n] &\equiv \langle \alpha(k), \ldots, \alpha(n)\rangle, \quad \text{(finite segment from position } k \text{ to } n)\\
\overline{\alpha, n} &\equiv \langle \alpha(0), \ldots, \alpha(n-1), \mathbf{0}, \mathbf{0}, \ldots\rangle, \quad \text{(infinite extension of } [\alpha](n) \text{ with } \mathbf{0}\text{'s)}\\
\hat{s} &\equiv \langle s_0, \ldots, s_{|s|-1}, \mathbf{0}, \mathbf{0}, \ldots\rangle. \quad \text{(infinite extension of finite seq. } s \text{ with } \mathbf{0}\text{'s)}
\end{aligned}$$

If x has type X_n and s has type $\Pi_{i=0}^{n-1} X_i$ then $s * x$ is the concatenation of s with s, which has type $\Pi_{i=0}^{n} X_i$. Similarly, if x has type X_0 and α has type $\Pi_{i=1}^{\infty} X_i$ then $x * \alpha$ has type $\Pi_{i\in\mathbb{N}} X_i$.

1.1 Background: Selection functions and their binary product

In our recent paper [8] we showed how one can view any element of type $(X \to R) \to R$ as a *generalised quantifier*. The particular case when $R = \mathbb{B}$ corresponds to the types of the usual logical quantifiers $\forall, \exists$. We also showed that some generalised quantifiers $\phi\colon (X \to R) \to R$ are *attainable*, in the sense that for some *selection function* $\varepsilon\colon (X \to R) \to X$, we have

$$\phi p = p(\varepsilon p)$$

for all (generalised) predicates p. In the case when ϕ is the usual existential quantifier, for instance, ε corresponds to Hilbert's epsilon term. Since the types $(X \to R) \to R$ and $(X \to R) \to X$ shall be used quite often, we will abbreviate them as $K_R X$ and $J_R X$, respectively. Moreover, when R is fixed, we often simply write KX and JX, omitting the subscript R. In [8] we also defined products of quantifiers and selection functions:

Definition 1. *Given a quantifier $\phi\colon KX$ and a family of quantifiers $\psi\colon X \to KY$, define a new quantifier $\phi \otimes \psi\colon K(X \times Y)$ as*

$$(\phi \otimes \psi)(p^{X\times Y\to R}) :\overset{R}{=} \phi(\lambda x^X.\psi(x, \lambda y^Y.p(x,y))).$$

Also, given a selection function $\varepsilon\colon JX$ and a family of selection functions $\delta\colon X \to JY$, define a new selection function $\varepsilon \otimes \delta\colon J(X \times Y)$ as

$$(\varepsilon \otimes \delta)(p^{X\times Y\to R}) \overset{X\times Y}{:=} (a, b(a))$$

where $b(x) := \delta(x, \lambda y^Y.p(x,y))$ and $a := \varepsilon(\lambda x^X.p(x, b(x)))$.

One of the results we obtained is that the product of attainable quantifiers is also attainable. This follows from the fact that the product of quantifiers corresponds to the product of selection functions, as made precise in the following lemma:

Lemma 1 ([8], lemma 3.1.2). *Let R be fixed. Given a selection function $\varepsilon : JX$, define a quantifier $\overline{\varepsilon}\colon KX$ as*

$$\overline{\varepsilon}p := p(\varepsilon p).$$

Then for $\varepsilon\colon JX$ and $\delta\colon X \to JY$ we have $\overline{\varepsilon \otimes \delta} = \overline{\varepsilon} \otimes \lambda x.\overline{\delta_x}$.

2 Two Infinite Products of Selection Functions

Given a finite sequence of selection functions (or generalised quantifiers), the binary product defined above can be iterated so as to give rise to a finite product. We have shown that such construction also appears in game theory (backward induction), algorithms (backtracking), and proof theory (interpretation of the infinite pigeon-hole principle). In the following we describe two possible ways of iterating the binary product of selection function an infinite (or unbounded) number of times.

2.1 Explicitly controlled iteration

The finite product of selection functions of Definition 1 can be infinitely iterated in two ways. The first, which we define in this section is via an "explicitly controlled" iteration, which we will show to correspond to Spector's bar recursion. In the following subsection we also define an "implicitly controlled" iteration, which we will show to correspond to modified bar recursion.

Definition 2. *Let $\varepsilon\colon \Pi_{k\in\mathbb{N}}((\Pi_{j<k}X_j) \to JX_k)$ be a family of selection functions. Define the* explicitly controlled infinite product of the selection functions *ε as*

$$\mathsf{EPS}_s(\omega)(\varepsilon) \overset{J(\Pi_{i=|s|}^{\infty}X_i)}{=} \begin{cases} \mathbf{0} & \text{if } \omega_s(\mathbf{0}) < |s| \\ \varepsilon_s \otimes \lambda x^{X_{|s|}}.\mathsf{EPS}_{s*x}(\omega)(\varepsilon) & \text{otherwise,} \end{cases}$$

where $s\colon \Sigma_{k\in\mathbb{N}}(\Pi_{j<k}X_j)$.

Lemma 2. *Let $q\colon \Pi_{i=|s|}^{\infty}X_i \to R$. EPS can be equivalently defined as*

$$\mathsf{EPS}_s(\omega)(\varepsilon)(q) \overset{\Pi_{i=|s|}^{\infty}X_i}{=} \begin{cases} \mathbf{0} & \text{if } \omega_s(\mathbf{0}) < |s| \\ c * \mathsf{EPS}_{s*c}(\omega)(\varepsilon)(q_c) & \text{otherwise,} \end{cases}$$

where $c = \varepsilon_s(\lambda x.q_x(\mathsf{EPS}_{s*x}(\omega)(\varepsilon)(q_x)))$.

Proof. Simply unfolding the definition of the binary product. □

Although we will only need to work with EPS, it will be useful (for the sake of clarity) to define also the conditional product of quantifiers:

Definition 3. *Let* $\phi\colon \Pi_{k\in\mathbb{N}}((\Pi_{j<k}X_j) \to KX_k)$ *be a family of quantifiers. The* explicitly controlled infinite product of quantifiers ϕ *is defined as*

$$\mathsf{EPQ}_s(\omega)(\phi) \overset{K(\Pi_{i=|s|}^{\infty}X_i)}{=} \begin{cases} \lambda q.q(\mathbf{0}) & \text{if } \omega_s(\mathbf{0}) < |s| \\ \phi_s \otimes \lambda x^{X_{|s|}}.\mathsf{EPQ}_{s*x}(\omega)(\phi) & \text{otherwise.} \end{cases}$$

The following lemma explains why EPQ can be defined from EPS if we are working with attainable quantifiers.

Lemma 3. *Assuming* $\forall\alpha\exists n(\omega_{[\alpha](n)}(\mathbf{0}) \leq n)$ *we have* $\mathsf{EPQ}_s(\omega)(\bar{\varepsilon}) = \overline{\mathsf{EPS}_s(\omega)(\varepsilon)}$.

2.2 Implicitly controlled iteration

The implicitly controlled iteration of the binary product of selection functions is defined as follows:

Definition 4. *Let* $\varepsilon\colon \Pi_{k\in\mathbb{N}}((\Pi_{j<k}X_j) \to (JX_k))$ *and* $s\colon \Sigma_{k\in\mathbb{N}}(\Pi_{j<k}X_j)$. *Define the* implicitly controlled infinite product of selection functions IPS *as*

$$\mathsf{IPS}_s(\varepsilon) \overset{J(\Pi_{i=|s|}^{\infty}X_i)}{=} \varepsilon_s \otimes \lambda x^{X_{|s|}}.\mathsf{IPS}_{s*x}(\varepsilon),$$

where $s\colon \Sigma_{k\in\mathbb{N}}(\Pi_{j<k}X_j)$.

Lemma 4. *Let* $q\colon \Pi_{i=|s|}^{\infty}X_i \to R$. IPS *can be equivalently defined as*

$$\mathsf{IPS}_s(\varepsilon)(q)(n) \overset{X_{|s|+n}}{=} \varepsilon_{s*t_{s,n}}(\lambda x^{X_{|s|+n}}.q_{t_{s,n}*x}(\mathsf{IPS}_{s*t_{s,n}*x}(\varepsilon)(q_{t_{s,n}*x})))$$

where $t_{s,n} := [\mathsf{IPS}_s(\varepsilon)(q)](n)$.

The functional IPS generalises Escardó's [6] construction that selection functions for a sequence of spaces can be combined into a selection function for the product space.

Proposition 1. IPS *(with* $R = \mathbb{B}$ *and* ε_s *dependent only on* $|s|$*) is primitive recursively equivalent to Escardo's* Π *functional of [6]:*

$$\Pi(\varepsilon)(q)(n) \overset{X_n}{=} \varepsilon_n(\lambda x^{X_n}.q_{n,x}(\Pi(\lambda i.\varepsilon_{n+i+1})(q_{n,x})))$$

where

$$q_{n,x}(\alpha^{\Pi_{i=n+1}^{\infty}X_i}) \overset{\mathbb{B}}{:=} q\left(\lambda i.\begin{cases} \Pi(\varepsilon)(q)(i) & i < n \\ x & i = n \\ \alpha(i-n-1) & i > n \end{cases}\right).$$

Proof. For one direction we take $\Pi(\varepsilon)(q) := \mathsf{IPS}_{\langle\rangle}(\varepsilon)(q)$. For the other direction take $\mathsf{IPS}_s(\{\varepsilon_n\}_{n\in\mathbb{N}})(q) := \Pi(\{\varepsilon_{|s|+n}\}_{n\in\mathbb{N}})(q)$. □

3 Dialectica Interpretation of Classical Analysis

We now show how the unbounded iteration EPS of the binary product of selection functions is *precisely* what is needed to solve Spector's equations (which arise from the dialectica interpretation of full classical analysis).

Theorem 1 (cf. lemma 11.5 of [11]). *Let* $q\colon \Pi_{i=0}^{\infty} X_i \to R$ *and* $\omega\colon \Pi_{i=0}^{\infty} X_i \to \mathbb{N}$ *and* $\varepsilon\colon \Pi_{i=0}^{\infty} JX_i$ *be given. Define*

$$\begin{aligned} \alpha &:= \mathsf{EPS}_{\langle\,\rangle}(\omega)(\varepsilon)(q) \\ p_n(x) &:= \mathsf{EPQ}_{[\alpha](n)*x}(\omega)(\overline{\varepsilon})(q_{[\alpha](n)*x}), \end{aligned}$$

identifying ε_s *with* $\varepsilon_{|s|}$. *Then, for all* $n \leq \omega(\alpha)$ *we have*

$$\begin{aligned} \alpha(n) &\overset{X_n}{=} \varepsilon_n(p_n) \\ p_n(\alpha(n)) &\overset{Y}{=} q\alpha. \end{aligned}$$

Proof. First, let us show by induction that for all n the following holds:

$$(i) \quad \alpha = [\alpha](n) * \mathsf{EPS}_{[\alpha](n)}(\omega)(\varepsilon)(q_{[\alpha](n)}).$$

If $n = 0$ this follows by definition. Assume this holds for n we wish to show it for $n+1$. Consider two cases.

(a) If $\omega(\overline{\alpha, n}) < n$ then $\mathsf{EPS}_{[\alpha](n)}(\omega)(\varepsilon)(q_{[\alpha](n)}) = \mathbf{0}$ and hence $\alpha \overset{\text{(IH)}}{=} \overline{\alpha, n} = \overline{\alpha, n+1}$. Therefore, $\omega(\overline{\alpha, n+1}) = \omega(\overline{\alpha, n}) < n < n+1$. So,

$$[\alpha](n+1) * \mathsf{EPS}_{[\alpha](n+1)}(\omega)(\varepsilon)(q_{[\alpha](n+1)}) = \overline{\alpha, n+1} = \overline{\alpha, n} = \alpha.$$

(b) If, on the other hand, $\omega(\overline{\alpha, n}) \geq n$, then

$$\alpha \overset{\text{(IH)}}{=} [\alpha](n) * \mathsf{EPS}_{[\alpha](n)}(\omega)(\varepsilon)(q_{[\alpha](n)}) = [\alpha](n) * c * \mathsf{EPS}_{[\alpha](n)*c}(\omega)(\varepsilon)(q_{[\alpha](n)*c}),$$

where $c = \alpha(n)$. Hence $\alpha = [\alpha](n+1) * \mathsf{EPS}_{[\alpha](n+1)}(\omega)(\varepsilon)(q_{[\alpha](n+1)})$.

Now, let $n := \omega(\alpha)$. We argue that (ii) $\omega(\overline{\alpha, n}) \geq n$. Otherwise, assuming $\omega(\overline{\alpha, n}) = \omega_{[\alpha](n)}(\mathbf{0}) < n$ we would have, by (i), that $\alpha = \overline{\alpha, n}$. And hence, by extensionality, $n > \omega_{[\alpha](n)}(\mathbf{0}) = \omega(\alpha) = n$, a contradiction.

Then, it follows easily that, if $n \leq \omega(\alpha)$,

$$\begin{aligned} \alpha(n) &\overset{(i)}{=} \mathsf{EPS}_{[\alpha](n)}(\omega)(\varepsilon)(q_{[\alpha](n)})(0) \\ &\overset{(ii)}{=} (\varepsilon_n \otimes \lambda x.\mathsf{EPS}_{[\alpha](n)*x}(\omega)(\varepsilon))(q_{[\alpha](n)})(0) \\ &= \varepsilon_n(\lambda x.q_{[\alpha](n)*x}(\mathsf{EPS}_{[\alpha](n)*x}(\omega)(\varepsilon)(q_{[\alpha](n)*x}))) \\ &= \varepsilon_n(\lambda x.\overline{\mathsf{EPS}_{[\alpha](n)*x}(\omega)(\varepsilon)}(q_{[\alpha](n)*x})) \\ &= \varepsilon_n(\lambda x.\mathsf{EPQ}_{[\alpha](n)*x}(\omega)(\overline{\varepsilon})(q_{[\alpha](n)*x})) = \varepsilon_n(p_n). \end{aligned}$$

For the second equality, we have

$$\begin{aligned} p_n(\alpha(n)) &= \mathsf{EPQ}_{[\alpha](n+1)}(\omega)(\overline{\varepsilon})(q_{[\alpha](n+1)}) \\ &= \overline{\mathsf{EPS}_{[\alpha](n+1)}(\omega)(\varepsilon)}(q_{[\alpha](n+1)}) \\ &= q_{[\alpha](n+1)}(\mathsf{EPS}_{[\alpha](n+1)}(\omega)(\varepsilon)(q_{[\alpha](n+1)})) \\ &= q([\alpha](n+1) * \mathsf{EPS}_{[\alpha](n+1)}(\omega)(\varepsilon)(q_{[\alpha](n+1)})) \overset{(i)}{=} q(\alpha). \end{aligned}$$

That concludes the proof. □

Remark 1. The theorem above has a very natural game theoretic reading. Following [8], each ε_n should be thought of as the selection function defining an outcome quantifier for round n. The functional q maps infinite plays to the outcome of the game. The construction used in the theorem for α and p_n calculates an infinite play α of the game which is optimal up to the point $n = \omega(\alpha)$. If ω is thought of as deciding when the game is terminated, then we have in fact an optimal play in the game.

Remark 2. Note that we are only using the conditional product of quantifiers for the sake of clarity. As shown in Lemma 3, any use of EPQ above can be replaced by an instance of EPS. Therefore, the recursion schema for EPS is precisely what is needed to solve Spector's equations. In this way, also the use of bar induction (in the proof of Lemma 3) is avoided.

3.1 Relation to Spector's bar recursion

As we have shown above, the conditional product of selection functions is precisely what is needed for a computational interpretation of classical analysis. Spector, however, describing the recursion schema used in his solution, formulated first the general "construction by bar recursion" as

$$\mathsf{BR}_s(\omega)(\phi)(g) \overset{R}{=} \begin{cases} g(s) & \text{if } \omega_s(\mathbf{0}) < |s| \\ \phi_s(\lambda x^{X_{|s|}}.\mathsf{BR}_{s*x}(\omega)(\phi)(g)) & \text{otherwise.} \end{cases}$$

Then, Spector explicitly says that only a "restricted form" of this is used. It is this restricted form that we shall from now on call "Spector's bar recursion":

Definition 5. *Let $R = \Pi_{i=0}^{\infty} X_i$ and $\phi_s \colon KX_{|s|}$ and $p \colon \Pi_{i=0}^{\infty} X_i \to \mathbb{N}$. Spector's bar recursion [12] is the following recursion schema*

$$\mathsf{SBR}_s(\phi)(\omega) \overset{R}{=} \begin{cases} \hat{s} & \text{if } \omega_s(\mathbf{0}) < |s| \\ \mathsf{SBR}_{s*c}(\phi)(\omega) & \text{otherwise,} \end{cases}$$

where $c = \phi_s(\lambda x.\mathsf{SBR}_{s*x}(\phi)(\omega))(|s|)$.

As we have shown above, however, it is EPS which is the precise bar recursive schema needed for solving Spector's equations. Here is how SBR generalises EPS:

Proposition 2. EPS *is primitive recursively definable in* SBR.

Remark 3. It is open whether EPS defines SBR primitive recursively. We believe this is not the case, since in SBR we are given quantifiers, from which we might not be able to re-construct selection functions to apply EPS. If not, then it is fair to say that the *precise* computational content of full classical analysis (from the dialectica interpretation point-of-view) is the conditional product of selection functions EPS, and not Spector's bar recursion SBR.

4 Realizability Interpretation of Classical Analysis

We have seen (Section 3 above) that the conditional iterated product of selection functions is precisely what is needed for the dialectica interpretation of classical countable choice. In this section we show that when interpreting countable choice via modified realizability, one seems to need the *unrestricted* iterated product of selection functions. Assuming continuity, one may say that the infinite iterated product is *implicitly* controlled, by the continuity of the functional q.

As discussed in the introduction, a computational interpretation of full classical analysis can be reduced to an interpretation of the double negation shift DNS. Given that the formula A (in DNS) can be assumed to be negated, i.e. of the form $\neg B$, DNS is equivalent to

$$\forall n((A(n) \to \bot) \to A(n)) \to \forall n A(n) \to \bot) \to \forall n A(n),$$

since, because $A = \neg B$, we have (in minimal logic) both $\bot \to A(n)$ and $\bot \to \forall n A(n)$. Moreover, since the negative translation brings us into minimal logic, falsity $\bot$ can be replaced by an arbitrary Σ^0_1-formula[3] R. Recall that $J_R A$ abbreviates $(A \to R) \to A$. The resulting principle we obtain is what we shall call J-shift

$$J\text{-shift} \quad : \quad \forall n J_R A(n) \to J_R \forall n A(n).$$

Note that $A(n)$ now can be an arbitrary formula (not necessarily a negated formula).

Theorem 2 (cf. [3], theorem 3). $\mathsf{IPS}_{\langle\rangle}$ *modified realizes* J-shift.

Proof. We assume continuity and relativised bar induction (as in [3]). Let

$$\varepsilon_n \ \mathsf{mr} \ (A(n) \to R) \to A(n)$$
$$q \quad \mathsf{mr} \ \forall n A(n) \to R.$$

We show $\forall s \in S \, \forall n P(s,n)$ by relativised bar induction, where

$$P(s,n) \equiv (s * \mathsf{IPS}_s(\varepsilon)(q_s))(n) \, \mathsf{mr} \, A(n)$$

and the predicate used in the relativisation is

$$s \in S \equiv \forall n < |s| \, (s_n \, \mathsf{mr} \, A(n)).$$

We write $\alpha \in S$ as an abbreviation for $\forall n([\alpha](n) \in S)$. We now prove the two assumptions of the bar induction:

(i) $\underline{\forall \alpha \in S \, \exists k \forall t \succeq [\alpha](k) \, \forall n P(t,n)}$, where $t \succeq s$ means that t is an extension of the finite sequence

[3] This is known as the (refined) A-translation [5], and is useful to analyse proofs of Π^0_2 theorems in analysis.

s. Given α we pick k to be the point of continuity of q on α. The result follows simply unfolding the definition of IPS.

(ii) $\underline{\forall s{\in}S(\forall t, x(s*t*x{\in}S \to \forall n P(s*t*x,n)) \to \forall n P(s,n))}$. Fix $s{\in}S$ and assume

$$\text{(a) } \forall t, x(s*t*x{\in}S \to \forall n P(s*t*x,n)).$$

We prove $\forall n P(s,n)$ by course-of-values induction. Assume $\forall k{<}n\, P(s,k)$, i.e.

$$\text{(b) } \forall k{<}n\, ((s*\mathsf{IPS}_s(\varepsilon)(q_s))(k)\,\mathsf{mr}\,A(k)).$$

We want to show $(s*\mathsf{IPS}_s(\varepsilon)(q_s))(n)\,\mathsf{mr}\,A(n)$. If $n<|s|$ we are done, since in this case $(s*\mathsf{IPS}_s(\varepsilon)(q_s))(n)=s_n$ (and $s\in S$). Assume $n\geq|s|$. Then, our goal becomes

$$\varepsilon_n(\lambda x^{X_n}.q_{s*t_{s,n}*x}(\mathsf{IPS}_{s*t_{s,n}*x}(\varepsilon)(q_{s*t_{s,n}*x})))\,\mathsf{mr}\,A(n),$$

where $t_{s,n}=[\mathsf{IPS}_s(\varepsilon)(q_s)](n-|s|)$. That follows from

$$\lambda x^{X_n}.q_{s*t_{s,n}*x}(\mathsf{IPS}_{s*t_{s,n}*x}(\varepsilon)(q_{s*t_{s,n}*x}))\,\mathsf{mr}\,A(n)\to R$$

which, by definition, is

$$\forall x^{X_n}(x\,\mathsf{mr}\,A(n)\ \to\ q_{s*t_{s,n}*x}(\mathsf{IPS}_{s*t_{s,n}*x}(\varepsilon)(q_{s*t_{s,n}*x}))\,\mathsf{mr}\,R).$$

Fix x such that $x\,\mathsf{mr}\,A(n)$. By our assumption (b) we have that $s*t_{s,n}*x\in S$. And by assumption (a) we get $(s*t_{s,n}*x*\mathsf{IPS}_{s*t_{s,n}*x}(\varepsilon)(q_{t_{s,n}*x}))\,\mathsf{mr}\,\forall n A(n)$. The proof is then concluded by the assumption that $q\,\mathsf{mr}\,\forall n A(n)\to R$. □

Remark 4. We analyse the J-shift in more details in [7], where a proof translation based on the construction JX is also defined.

4.1 Relation to modified bar recursion

We now show that IPS and modified bar recursion are in fact primitive recursively inter-definable. Modified bar recursion [3], when generalised to the language of dependent types, can be formulated as

$$\mathsf{MBR}_s(\varepsilon)(q)(n)\overset{X_n}{=}\begin{cases} s_n & \text{if } n<|s| \\ \varepsilon_s(\lambda x^{X_{|s|}}.q(\mathsf{MBR}_{s*x}(\varepsilon)(q)))(n-|s|) & \text{otherwise,}\end{cases}$$

where $\varepsilon_s\colon (X_{|s|}\to R)\to \Pi_{j\geq|s|}X_j$. The following lemma says that MBR is equivalent to a variant which can make use of any value bar recursively computed, and not just the immediate children $s*x$ of the node s. We are assuming that types are restricted so that finite sequences of X_k's can be coded as single elements.

Lemma 5 ([3], lemma 2). MBR *is primitive recursively equivalent to*

$$\mathsf{MBR}^0_s(\varepsilon)(q)(n)\overset{X_n}{=}\begin{cases} s_n & \text{if } n<|s| \\ \varepsilon_s(\lambda r^{\Pi_{k=|s|}^{j-1}X_k}\lambda x^{X_j}.q(\mathsf{MBR}^0_{s*r*x}(\varepsilon)(q)))(n-|s|) & \text{otherwise.}\end{cases}$$

The next theorem essentially says that MBR is also equivalent to a variant which makes use of course-of-values recursion to access values previously computed, i.e. in order to define the point n of the infinite sequence $\mathsf{MBR}^1_s(\varepsilon)(q)$ we are allowed to use $\mathsf{MBR}^1_s(\varepsilon)(q)(k)$ for $k < n$.

Lemma 6. MBR^0 *is primitive recursively equivalent to*

$$\mathsf{MBR}^1_s(\varepsilon)(q)(n) \overset{X_n}{=} \begin{cases} s_n & \text{if } n < |s| \\ \varepsilon_s(r_{s,n}, \lambda r^\eta, x^{X_j}.q(\mathsf{MBR}^1_{s*r*x}(\varepsilon)(q)))(n-|s|) & \text{otherwise,} \end{cases} \tag{1}$$

where $r_{s,n} := \mathsf{MBR}^1_s(\varepsilon)(q)[|s|, n-1]$ *and* $\eta = \Pi_{k=|s|}^{j-1} X_k$.

Corollary 1. MBR *primitive recursively defines* IPS.

Theorem 3. IPS *primitive recursively defines* MBR.

Proof. Let $X'_i \equiv \mathbb{B} \times X_i$, and assume $\langle b^{\mathbb{B}}, x^{X_i} \rangle$ denotes pairing and $(a^{\mathbb{B}\times X_i})_0 \colon \mathbb{B}$ and $(a^{\mathbb{B}\times X_i})_1 \colon X_i$ are the projections. A family of mappings

$$\varepsilon_s \colon (X_{|s|} \to R) \to \Pi_{j\geq|s|} X_j$$

can be turned into a family of selection functions $\tilde{\varepsilon}_s \colon J(\Pi_{j\geq|s|} X'_j)$ as

$$\tilde{\varepsilon}_s(f^{\Pi_{j\geq|s|}X'_j \to R}) \overset{\Pi_{j\geq|s|}X'_j}{:=} \lambda j.\langle 1, \varepsilon_s(\lambda x^{X_{|s|}}.f(\tilde{x}))(j)\rangle,$$

where $\tilde{x}$ is the infinite sequence of pairs starting with $\langle 0, x\rangle$ and followed by $\langle 0, \mathbf{0}\rangle$. Intuitively, $\langle 1, a\rangle$ means that a is a computed ε-value, whereas $\langle 0, a\rangle$ means that a is a recursive call. We have that $\mathsf{IPS}_{\langle\rangle}(\tilde{\varepsilon}) \colon \Pi_i \Pi_{j\geq i} X'_j$. Also, a predicate $q \colon \Pi_i X_i \to R$ can be turned into a predicate $\tilde{q} \colon \Pi_i \Pi_{j\geq i} X'_j \to R$ as $\tilde{q}(\alpha) := q(\tilde{\alpha})$ where

$$\tilde{\alpha}(i) \overset{X_i}{:=} \begin{cases} (\alpha(i)(0))_1 & \text{if } (\alpha(i)(0))_0 = 0 \\ (\alpha(k)(i-k))_1 & \text{else,} \end{cases}$$

k being the least point smaller than i where $(\alpha(k)(k))_0 = 1$. Intuitively, the construction $\tilde{\alpha}$ takes a matrix $\alpha \colon \Pi_i \Pi_{j\geq i} X'_j$ and returns a sequence in $\Pi_i X_i$, by returning elements from the first row while flags 0 are found, and then elements of column k, where k is the first column with flag 1. We claim that MBR can be defined as

$$\mathsf{MBR}_s(\varepsilon)(q) \overset{\Pi_j X_j}{:=} s * (\mathsf{IPS}_{\tilde{s}}(\tilde{\varepsilon})(\widetilde{q_s})(0))^1$$

where, for $\alpha \colon \Pi_{i\geq k} X'_i$ we have $\alpha^1 \colon \Pi_{i\geq k} X_i$ (by simple projection), and $\tilde{s}$ is the embedding of $s \colon \Pi_{i<n} X_i$ into $\Pi_{i<n} \Pi_j X'_{i+j}$ as $(\tilde{s})_i := \widetilde{s_i}$. Note that

(*i*) $\tilde{s} * \tilde{x} = \widetilde{s * x}$,

(*ii*) $(\tilde{q})_{\tilde{s}} = \widetilde{q_s}$.

Unfolding definitions

$$\begin{aligned}
\mathsf{MBR}_s(\varepsilon)(q) &\overset{\Pi_j X_j}{=} s * (\mathsf{IPS}_{\tilde{s}}(\tilde{\varepsilon})(\widetilde{q_s})(0))^1 \\
&= s * (\tilde{\varepsilon}_s(\lambda\alpha^{\Pi_j X'_{|s|+j}}.(\widetilde{q_s})_\alpha(\mathsf{IPS}_{\tilde{s}*\alpha}(\tilde{\varepsilon})((\widetilde{q_s})_\alpha))))^1 \\
&= s * \varepsilon_s(\lambda x^{X_{|s|}}.(\widetilde{q_s})_{\tilde{x}}(\mathsf{IPS}_{\tilde{s}*\tilde{x}}(\tilde{\varepsilon})((\widetilde{q_s})_{\tilde{x}}))) \\
&\overset{(i)}{=} s * \varepsilon_s(\lambda x^{X_{|s|}}.(\widetilde{q_s})_{\tilde{x}}(\mathsf{IPS}_{\widetilde{s*x}}(\tilde{\varepsilon})((\widetilde{q_s})_{\tilde{x}}))) \\
&\overset{(ii)}{=} s * \varepsilon_s(\lambda x^{X_{|s|}}.(\tilde{q})_{\widetilde{s*x}}(\mathsf{IPS}_{\widetilde{s*x}}(\tilde{\varepsilon})(\widetilde{q_{s*x}}))) \\
&= s * \varepsilon_s(\lambda x^{X_{|s|}}.q_{s*x}((\mathsf{IPS}_{\widetilde{s*x}}(\tilde{\varepsilon})(\widetilde{q_{s*x}})(0))^1)) \\
&= s * \varepsilon_s(\lambda x^{X_{|s|}}.q(\mathsf{MBR}_{s*x}(\varepsilon)(q))).
\end{aligned}$$

That concludes the proof. □

Corollary 2. *The equation defining* IPS *has a solution in the type structure* $\mathcal{M}$ *of the strongly majorizable functionals.*

Proof. This follows from the result in [4] that MBR lives in the model $\mathcal{M}$. □

Acknowledgements. The second author gratefully acknowledges support of the Royal Society (grant 516002.K501/RH/kk).

References

1. J. Avigad and S. Feferman. Gödel's functional ("Dialectica") interpretation. In S. R. Buss, editor, *Handbook of proof theory*, volume 137 of *Studies in Logic and the Foundations of Mathematics*, pages 337–405. North Holland, Amsterdam, 1998.
2. S. Berardi, M. Bezem, and T. Coquand. On the computational content of the axiom of choice. *The Journal of Symbolic Logic*, 63(2):600–622, 1998.
3. U. Berger and P. Oliva. Modified bar recursion and classical dependent choice. *Lecture Notes in Logic*, 20:89–107, 2005.
4. U. Berger and P. Oliva. Modified bar recursion. *Mathematical Structures in Computer Science*, 16:163–183, 2006.
5. U. Berger and H. Schwichtenberg. Program extraction from classical proofs. In D. Leivant, editor, *Logic and Computational Complexity Workshop (LCC'94)*, volume 960 of *Lecture Notes in Computer Science*, pages 77–97. Springer, Berlin, 1995.
6. M. H. Escardó. Infinite sets that admit fast exhaustive search. In *Proceedings of LICS*, pages 443–452, 2007.
7. M. H. Escardó and P. Oliva. The peirce translation. *Submitted to CiE 2010*, 2010.
8. M. H. Escardó and P. Oliva. Selection functions, bar recursion, and backward induction. *Mathematical Structures in Computer Science*, to appear.
9. K. Gödel. Über eine bisher noch nicht benützte Erweiterung des finiten Standpunktes. *Dialectica*, 12:280–287, 1958.
10. U. Kohlenbach. Higher order reverse mathematics. In Stephen G. Simpson, editor, *Reverse Mathematics 2001*, volume 21 of *Lecture Notes in Logic*, pages 281–295. ASL, A K Peters, 2005.
11. U. Kohlenbach. *Applied Proof Theory: Proof Interpretations and their Use in Mathematics*. Monographs in Mathematics. Springer, 2008.

12. 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.
13. A. S. Troelstra. *Metamathematical Investigation of Intuitionistic Arithmetic and Analysis*, volume 344 of *Lecture Notes in Mathematics*. Springer, Berlin, 1973.

5 Appendix: Omitted proofs

Proof of Lemma 3. The above equation can be shown by bar induction, using the condition $\forall\alpha\exists n(\omega_{[\alpha](n)}(\mathbf{0}) \leq n)$ as the basis of the induction. If $\omega_s(\mathbf{0}) < |s|$ then, for any given q,

$$\mathsf{EPQ}_s(\omega)(\bar{\varepsilon})(q) = q(\mathbf{0}) = q(\mathsf{EPS}_s(\omega)(\varepsilon)(q)) = \overline{\mathsf{EPS}_s(\omega)(\varepsilon)}(q).$$

If, however, $\omega_s(\mathbf{0}) \geq |s|$, then

$$\begin{aligned}\mathsf{EPQ}_s(\omega)(\bar{\varepsilon})(q) &= (\bar{\varepsilon}_s \otimes \lambda x.\mathsf{EPQ}_{s*x}(\omega)(\bar{\varepsilon}))(q)\\ &\overset{\text{(IH)}}{=} (\bar{\varepsilon}_s \otimes \lambda x.\overline{\mathsf{EPS}_{s*x}(\omega)(\varepsilon)})(q)\\ &\overset{\text{(L1)}}{=} \overline{(\varepsilon_s \otimes \lambda x.\mathsf{EPS}_{s*x}(\omega)(\varepsilon))}(q) \equiv \overline{\mathsf{EPS}_s(\omega)(\varepsilon)}(q),\end{aligned}$$

using the bar recursive induction step. □

Proof of Proposition 2. Looking at the reformulation of EPS described in Lemma 2, we see that at each recursive call to EPS the argument predicate q keeps changing. Consider, however, the following variant where $q\colon \Pi_{i=0}^{\infty} X_i \to R$ stays fixed:

$$\widetilde{\mathsf{EPS}}_s(\omega)(\varepsilon)(q) \overset{\Pi_{i=0}^{\infty} X_i}{\equiv} \begin{cases} \hat{s} & \text{if } \omega_s(\mathbf{0}) < |s| \\ \widetilde{\mathsf{EPS}}_{s*c}(\omega)(\varepsilon)(q) & \text{otherwise,}\end{cases}$$

where $c = \varepsilon_s(\lambda x.q(\widetilde{\mathsf{EPS}}_{s*x}(\omega)(\varepsilon)(q)))$. We argue that EPS can be defined from $\widetilde{\mathsf{EPS}}$ as

$$\mathsf{EPS}_s(\omega)(\varepsilon)(q) \overset{\Pi_{i=|s|}^{\infty} X_i}{:=} \lambda n.\widetilde{\mathsf{EPS}}_s(\omega)(\varepsilon)(q^{|s|})(|s|+n)$$

where $q^n(\alpha) = q(\alpha(n), \alpha(n+1), \ldots)$. If $\omega_s(\mathbf{0}) < |s|$ the result is trivial. Assume $\omega_s(\mathbf{0}) \geq |s|$. Then, we have

$$\begin{aligned}\mathsf{EPS}_s(\omega)(\varepsilon)(q) &= \lambda n.\widetilde{\mathsf{EPS}}_s(\omega)(\varepsilon)(q^{|s|})(|s|+n)\\ &= \lambda n.\widetilde{\mathsf{EPS}}_{s*c}(\omega)(\varepsilon)(q^{|s|})(|s|+n)\\ &= \lambda n.\widetilde{\mathsf{EPS}}_{s*c}(\omega)(\varepsilon)((q_c)^{|s*c|})(|s|+n)\\ &= c * \lambda n.\widetilde{\mathsf{EPS}}_{s*c}(\omega)(\varepsilon)((q_c)^{|s*c|})(|s*c|+n)\\ &= c * \mathsf{EPS}_{s*c}(\omega)(\varepsilon)(q_c)\end{aligned}$$

where $c = \varepsilon_s(\lambda x.q_x(\mathsf{EPS}_{s*x}(\omega)(\varepsilon)(q_x))) = \varepsilon_s(\lambda x.q^{|s|}(\widetilde{\mathsf{EPS}}_{s*x}(\omega)(\varepsilon)(q^{|s|})))$. It is then straightforward to show that $\widetilde{\mathsf{EPS}}$ is definable in SBR as

$$\widetilde{\mathsf{EPS}}_s(\omega)(\varepsilon)(q) := \mathsf{SBR}_s(\phi^{\varepsilon,q})(\omega),$$

where $\phi_s^{\varepsilon,q}(f) \overset{\Pi_{i=0}^{\infty} X_i}{:=} f(\varepsilon_s(\lambda x^{X_{|s|}}.q(f(x))))$. □

Proof of Lemma 6. MBR^1 is clearly more general than MBR^0. For the other direction, consider the following construction

$$\hat{\varepsilon}_s(f)(n) \overset{X_{|s|+n}}{:=} \varepsilon_s(r_{s,n}, f)(n)$$

where $r_{s,n} = h_s(0) * \ldots h_s(n-1)$, and $h_s(i)$ is calculated by course-of-values recursion as

$$h_s(i) = \varepsilon_s([h_s](i), f)(i).$$

Define MBR^1 by MBR^0 as

$$\mathsf{MBR}^1_s(\varepsilon)(q) := \mathsf{MBR}^0_s(\hat{\varepsilon})(q).$$

We show that MBR^1 defined as such satisfies the equation (1). Let $n \geq |s|$ (if $n < |s|$ the result is trivial), we have

$$\begin{aligned}
\mathsf{MBR}^1_s(\varepsilon)(q)(n) &\overset{X_n}{=} \mathsf{MBR}^0_s(\hat{\varepsilon})(q)(n) \\
&= \hat{\varepsilon}_s(\lambda r, x.\mathsf{MBR}^0_{s*r*x}(\hat{\varepsilon})(q))(n-|s|) \\
&= \hat{\varepsilon}_s(\lambda r, x.\mathsf{MBR}^1_{s*r*x}(\varepsilon)(q))(n-|s|) \\
&= \varepsilon_s(r_{s,n}, \lambda r, x.\mathsf{MBR}^1_{s*r*x}(\varepsilon)(q))(n-|s|),
\end{aligned}$$

where $r_{s,n} = \mathsf{MBR}^1_s(\varepsilon)(p)[|s|, n-1]$. □

Proof of Corollary 1. We show that MBR^1 defines IPS. That can be done as

$$\mathsf{IPS}_s(\varepsilon)(q)(n) \overset{X_{|s|+n}}{:=} \mathsf{MBR}^1_s(\tilde{\varepsilon})(q^{|s|})(|s|+n),$$

where $\tilde{\varepsilon}_s(t,f)(i) := \varepsilon_{s*t}(ft)$ and $q^{|s|}(\alpha^{\Pi_i X_i}) \overset{R}{=} q(\lambda n.\alpha(|s|+n))$. So $q\colon \Pi_{i\geq|s|}X_i \to R$ and $q^{|s|}\colon \Pi_i X_i \to R$. Then

$$\begin{aligned}
\mathsf{IPS}_s(\varepsilon)(q)(n) &\overset{X_{|s|+n}}{=} \mathsf{MBR}^1_s(\tilde{\varepsilon})(q^{|s|})(|s|+n) \\
&= \tilde{\varepsilon}_s(\tilde{r}, \lambda r^\eta, x^{X_j}.q^{|s|}(\mathsf{MBR}^1_{s*r*x}(\tilde{\varepsilon})(q^{|s|})))(|s|+n) \\
&= \varepsilon_{s*\tilde{r}}(\lambda x^{X_{|s|+n}}.q^{|s|}(\mathsf{MBR}^1_{s*\tilde{r}*x}(\tilde{\varepsilon})(q^{|s|}))) \\
&= \varepsilon_{s*\tilde{r}}(\lambda x^{X_{|s|+n}}.(q_{\tilde{r}*x})^{|s*\tilde{r}*x|}(\mathsf{MBR}^1_{s*\tilde{r}*x}(\tilde{\varepsilon})((q_{\tilde{r}*x})^{|s*\tilde{r}*x|}))) \\
&= \varepsilon_{s*\tilde{r}}(\lambda x^{X_{|s|+n}}.q_{\tilde{r}*x}(\mathsf{IPS}^1_{s*\tilde{r}*x}(\varepsilon)(q_{\tilde{r}*x}))) \\
&= \varepsilon_{s*t_{s,n}}(\lambda x^{X_{|s|+n}}.q_{t_{s,n}*x}(\mathsf{IPS}_{s*t_{s,n}*x}(\varepsilon)(q_{t_{s,n}*x}))),
\end{aligned}$$

with $\tilde{r} = \mathsf{MBR}^1_s(\tilde{\varepsilon})(q^{|s|})[|s|, |s|+n-1] = [\mathsf{IPS}_s(\varepsilon)(q)](n) = t_{s,n}$. □